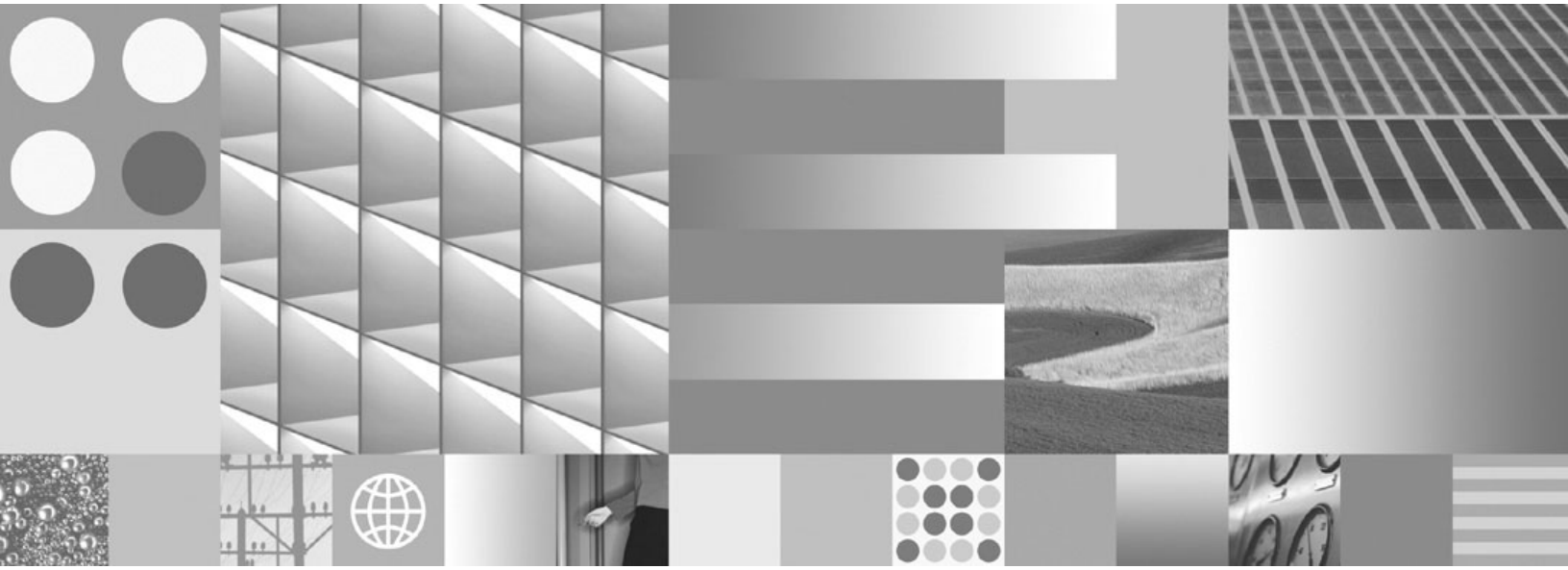
IBM DB2 9.7
for Linux, UNIX, and Windows

バージョン 9 リリース 7



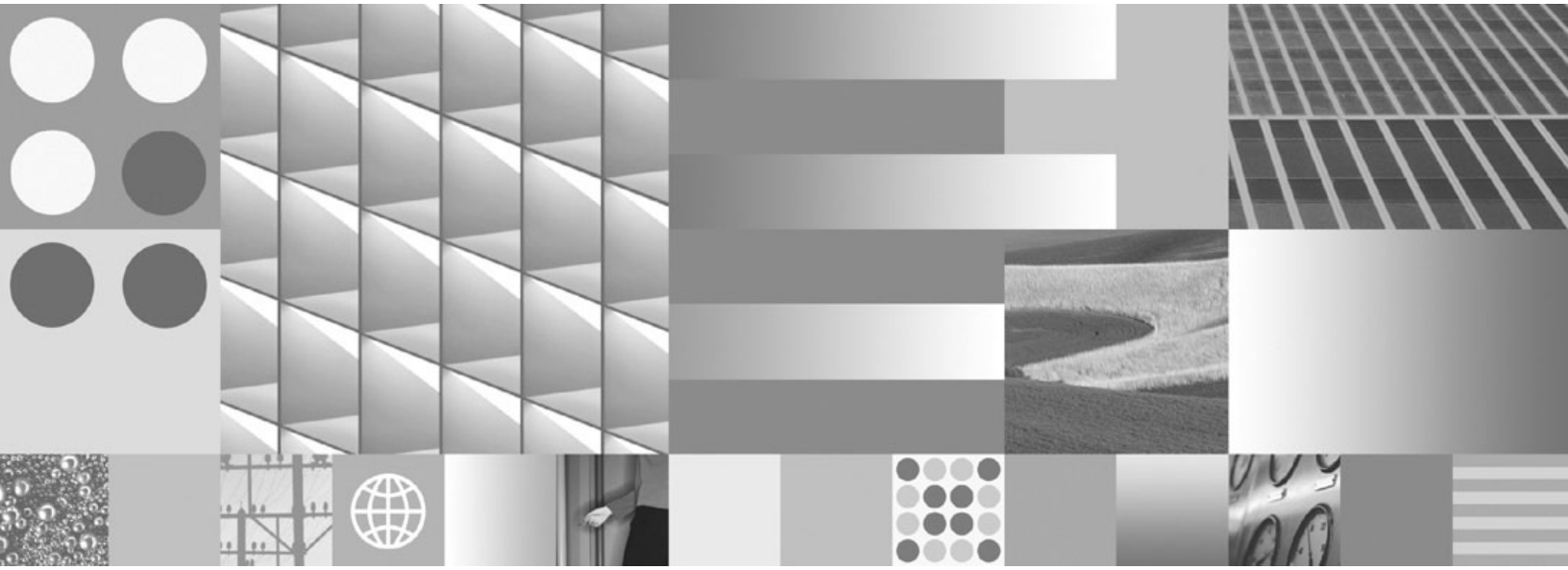
管理ルーチンおよびビュー

2010 年 9 月更新版

SC88-5880-02
(英文原典：SC27-2436-02)

IBM DB2 9.7
for Linux, UNIX, and Windows

バージョン 9 リリース 7



管理ルーチンおよびビュー

2010 年 9 月更新版

SC88-5880-02
(英文原典：SC27-2436-02)

ご注意

本書および本書で紹介する製品をご使用になる前に、 1365 ページの『付録 B. 特記事項』に記載されている情報をお読みください。

本書には、IBM の専有情報が含まれています。その情報は、使用許諾条件に基づき提供され、著作権により保護されています。本書に記載される情報には、いかなる製品の保証も含まれていません。また、本書で提供されるいかなる記述も、製品保証として解釈すべきではありません。

IBM 資料は、オンラインでご注文いただくことも、ご自分の国または地域の IBM 担当員を通してお求めいただくこともできます。

- オンラインで資料を注文するには、www.ibm.com/shop/publications/order にある IBM Publications Center をご利用ください。
- ご自分の国または地域の IBM 担当員を見つけるには、www.ibm.com/planetwide にある IBM Directory of Worldwide Contacts をお調べください。

IBM は、お客様が提供するいかなる情報も、お客様に対してなんら義務も負うことのない、自ら適切と信ずる方法で、使用もしくは配布することができるものとします。

お客様の環境によっては、資料中の円記号がバックスラッシュと表示されたり、バックスラッシュが円記号と表示されたりする場合があります。

原典： SC27-2436-02
IBM DB2 9.7 for Linux, UNIX, and Windows
Version 9 Release 7
Administrative Routines and Views
Updated September, 2010

発行： 日本アイ・ビー・エム株式会社

担当： トランスレーション・サービス・センター

第1刷 2010.9

© Copyright IBM Corporation 2006, 2010.

目次

© Copyright IBM Corp. 2006, 2010

第 1 章 システム定義のルーチンとビュー

システム定義のルーチンとビューでは、DB2® を SQL により管理および使用するための、基本的で使いやすいプログラマチック・インターフェースが提供されます。これは、さまざまな DB2 タスクを実行するための、組み込みビュー、表関数、プロシージャー、およびスカラー関数の集合を対象としています。 例えば、システム定義ルーチンは、表の再編成、モニター・データのキャプチャーおよび検索、または現在の接続のアプリケーション ID の検索などに使用できます。

これらのルーチンおよびビューは、SQL ベースのアプリケーション、DB2 コマンド行、またはコマンド・スクリプトから呼び出すことができます。

システム定義のルーチンおよびビューをアプリケーションで呼び出す場合のベスト・プラクティス

システム定義のルーチンやビューを適切に使用するために、推奨されるコーディングの手法があります。特に、リリースが更新されるにつれて機能強化によってルーチンが変更される可能性があるため、これらの手法は重要です。

システム定義ルーチンまたはビューを使って情報を取得するために照会を発行するとき、SELECT * ... という形式のステートメントを使用しないでください。例えば、次のような照会を発行しないでください。

```
SELECT * FROM TABLE(MON_GET_UNIT_OF_WORK(NULL,-1)) AS t
ORDER BY total_cpu_time DESC
```

その代わりに、SELECT ステートメントで結果列を指定します。こうすることで、アプリケーションは結果列の数と、それらが戻される順序を制御することができます。例えば、

```
SELECT application_handle,
       uow_id,
       total_cpu_time,
       app_rqsts_completed_total,
       rqsts_completed_total
FROM TABLE(MON_GET_UNIT_OF_WORK(NULL,-1)) AS t
ORDER BY total_cpu_time DESC
```

これにより、ルーチン内で列の順序と数が変更された場合の問題を防ぐことができます。ルーチンによって戻される結果列の数は増える可能性があります。例えば 6 つの結果列がルーチンによって戻されるにもかかわらず 5 つのホスト変数だけを提供すると、アプリケーションで問題が発生します。

さらに、ルーチンの出力パラメーターまたは結果列のタイプとサイズが変更されることもあります。例えば、列が変更されて VARCHAR(8) から VARCHAR(128) になったり、INTEGER 列が BIGINT 列になったりする可能性があります。使用される変数のサイズが小さすぎる場合、ルーチンから受け取るデータが切り捨てられる可能性があります。

© Copyright IBM Corp. 2006, 2010

このような変化からアプリケーションを保護するために、C アプリケーションでは準備済み (PREPARE) テートメントを記述 (DESCRIBE) することにより、どの結果列が戻されるのか、およびどのようなタイプとサイズであるのかを判別できます。例えば、SELECT application_handle, uow_id,total_cpu_time FROM TABLE(MON_GET_UNIT_OF_WORK(NULL,-1)) AS t ORDER BY total_cpu_time DESC という照会を記述するには、以下のコード・スニペットを使用できます。

```
strcpy(strStmt, "SELECT application_handle, uow_id,total_cpu_time
   FROM TABLE(MON_GET_UNIT_OF_WORK(NULL,-1))
   AS t ORDER BY total_cpu_time DESC");
EXEC SQL PREPARE stmt FROM :strStmt;
EXEC SQL DESCRIBE stmt into :*pSqlda;
```

SQLDA で戻される情報の使用方法について、詳しくは samples/c/tbread.sqc にある RowDatamemoryAlloc 関数を参照してください。

Java™ および .Net アプリケーションでは、データ・タイプとサイズが問題となる場合には、どんなタイプ/サイズのどんな結果列が戻されるかを判別するためにメタデータを使用できます。例えば、

```
ResultSet rs = pstmt.executeQuery();
ResultSetMetaData rsms = rs.getMetaData();
```

結果セットのメタデータを使用する方法について、詳しくは samples/java/jdbc/TbRead.java にある execPreparedQueryWithUnknownOutputColumn() メソッドを参照してください。

管理ビューの許可

SYSIBMADM スキーマ内のすべての管理ビューについて、ビューに対する SELECT 特権が必要です。これは以下の照会により妥当性検査できますが、この照会では許可 ID、または所属するグループあるいはロールが、SELECT 特権を持っている (つまり、検索基準を満たしており、GRANTEE 列にリストされている) ことを確認します。

```
SELECT GRANTEE, GRANTEETYPE
   FROM SYSCAT.TABAUTH
   WHERE TABSCHEMA = 'SYSIBMADM' AND TABNAME = '<view_name>' AND
   SELECTAUTH <> 'N'
```

ここで <view_name> は管理ビューの名前です。

SYSIBMADM.AUTHORIZATIONIDS、SYSIBMADM.OBJECTOWNERS、および SYSIBMADM.PRIVILEGES を除き、基礎の管理表関数に対する EXECUTE 特権も必要です。基礎の管理表関数は、管理ビューの許可セクションにリストされています。これは以下の照会で妥当性検査することができます。

```
SELECT GRANTEE, GRANTEETYPE
   FROM SYSCAT.ROUTINEAUTH
   WHERE SCHEMA = 'SYSPROC' AND SPECIFICNAME = '<routine_name>' AND
   EXECUTEAUTH <> 'N'
```

ここで <routine_name> は、文書内にリストされる基礎の管理表関数の名前です。

一部の管理ビューでは、ビューに対する SELECT および基礎の管理表関数に対する EXECUTE 以上の追加の権限が必要になります。必要とされるどの追加権限も、ビューに記述される参照情報で文書化される必要があります。

管理ビューと表関数との比較

DB2 バージョン 9.5 では、SQL による DB2 管理機能への使いやすいアプリケーション・プログラミング・インターフェースを提供する、管理ビューが導入されました。

管理ビューは、以下の 3 つのカテゴリーに分けられます。

- カタログ・ビューを基にしたビュー。
- 入力パラメーターを持たない表関数を基にしたビュー。
- 1 つ以上の入力パラメーターを持つ表関数を基にしたビュー。

表関数は追加情報またはパフォーマンス上の利点を提供しないため、管理ビューは、カタログ・ビューを基にしたビューおよび入力パラメーターを持たない表関数を基にしたビューの、優先される唯一の文書化されたインターフェースです。

1 つ以上の入力パラメーターを持つ表関数を基にしたビューの場合、管理ビューと表関数の両方を使用できますが、それぞれは以下のように異なる目標を実現します。

- ADMINTABINFO 管理ビューおよび ADMIN_GET_TAB_INFO_V95 表関数: この管理ビューは、データベース内のすべての表の情報を検索します。 これは、大規模なデータベースではパフォーマンスに大きく影響を与える可能性があります。パフォーマンスへの影響は、表関数を使用し、スキーマ名、表名、またはその両方を入力として指定することにより減らすことができます。
- PDLOGMSGS_LAST24HOURS 管理ビューおよび PD_GET_LOG_MSGS 表関数: 通知ログ・メッセージを検索するこの管理ビューは、直前の 24 時間のデータへの迅速なアクセスを提供しますが、この表関数は指定期間内からデータを検索します。
- すべてのスナップショット・モニター管理ビューおよび表関数 (SNAP* 管理ビュー、SNAP_GET_* 表関数): スナップショット・モニター管理ビューは、各データベース・パーティションからのデータへのアクセスを提供します。これらの表関数は、単一のデータベース・パーティションからのデータ、またはすべてのデータベース・パーティションから集約したデータのいずれかを選択するオプションを提供します。

表関数は新しい情報を戻すためにリリースごとに変更されている可能性があるので、ビューの代わりに表関数を使用するアプリケーションは、変更が必要になる可能性があります。新しい表関数は、元の関数と同じベース名を持ち、それが追加された製品のバージョンの '_Vxx' (_V97 など) という接尾部が付けられます。管理ビューは必ず最新のバージョンの表関数を基にするので、アプリケーションの移植性は向上します。列はリリースごとに変わる場合があるので、管理ビューから特定の列を選択するか、または SELECT * ステートメントがアプリケーションにより使用される場合は結果セットを記述することをお勧めします。

第 2 章 サポートされるシステム定義の SQL ルーチンおよび管理ビュー

以下の表は、サポートされる管理 SQL ルーチンおよび管理ビューについての情報を要約したものです。

表 1. アクティビティー・モニター管理 SQL ルーチン

ルーチン名	スキーマ	説明
AM_BASE_RPT_RECOMS 表関数	SYSPROC	この表関数は、アクティビティー・モニターで使用されるアクティビティー報告書に関する勧告を戻します。
AM_BASE_RPTS 表関数	SYSPROC	この表関数は、アクティビティー・モニターで使用されるアクティビティー報告書を戻します。
AM_DROP_TASK プロシージャー	SYSPROC	このプロシージャーは、モニター・タスクを削除します。
AM_GET_LOCK_CHN_TB プロシージャー	SYSPROC	このプロシージャーは、表フォーマットのアプリケーション・ロック・チェーン・データを戻します。

© Copyright IBM Corp. 2006, 2010

表 1. アクティビティー・モニター管理 SQL ルーチン (続き)

ルーチン名	スキーマ	説明
AM_GET_LOCK_CHNS プロシージャー	SYSPROC	このプロシージャーは、フォーマット設定ストリングを使って、指定のアプリケーション用のロック・チェーンを表示します。
AM_GET_LOCK_RPT プロシージャー	SYSPROC	このプロシージャーは、アプリケーションのロックの詳細を表示します。
AM_GET_RPT プロシージャー	SYSPROC	このプロシージャーは、レポートのアクティビティー・モニター・データを表示します。
AM_SAVE_TASK プロシージャー	SYSPROC	このプロシージャーは、モニター・タスクを作成または変更します。

表 2. ADMIN_CMD ストアード・プロシージャーおよび関連する管理 SQL ルーチン

ルーチン名	スキーマ	説明
ADMIN_CMD プロシージャー	SYSPROC	このプロシージャーを使用すると、管理者は CALL ステートメントで ADMIN_CMD を実行することにより、管理コマンド (DB2 コマンド行プロセッサー (CLP) コマンドを含む) を実行することができます。
ADMIN_EST_INLINE_LENGTH 関数	SYSIBM	この関数は、XML 列、BLOB 列、CLOB 列、または DBCLOB 列に格納されたデータをインライン化するために必要な、インライン長の見積もりを戻します。
ADMIN_GET_DBP_MEM_USAGE 表関数	SYSPROC	この表関数は、指定されたインスタンスの合計メモリー消費量を取得します。
ADMIN_GET_INDEX_COMPRESS_INFO	SYSPROC	この表関数は、非圧縮索引について索引圧縮による節約の可能性を戻すか、カタログからの索引圧縮統計をレポートします。
ADMIN_GET_INDEX_INFO 表関数	SYSPROC	この表関数は、カタログ・ビューで使用できない索引情報を戻します。
ADMIN_GET_MSGS 表関数	SYSPROC	この表関数は、ADMIN_CMD プロシージャーを介して実行されるデータ移動ユーティリティーによって生成されるメッセージを取得するのに使用されます。
ADMIN_IS_INLINED 関数	SYSIBM	この関数は、XML 列、BLOB 列、CLOB 列、または DBCLOB 列のインライン・データに関する状態情報を取得します。
ADMIN_REMOVE_MSGS プロシージャー	SYSPROC	このプロシージャーは、ADMIN_CMD プロシージャーを介して実行されるデータ移動ユーティリティーによって生成されるメッセージをクリーンアップするのに使用されます。
ADMIN_REVALIDATE_DB_OBJECTS プロシージャー	SYSPROC	このプロシージャーは、無効なデータベース・オブジェクトを再度有効にします。
ADMINTABCOMPRESSINFO ビューおよび ADMIN_GET_TAB_COMPRESS_INFO_V97 表関数	SYSIBMADM (管理ビュー)、SYSPROC (表関数)	このビューおよび表関数は、表、マテリアライズ照会表 (MQT)、および階層表の圧縮情報を戻します。

表 2. *ADMIN_CMD ストアード・プロシージャーおよび関連する管理 SQL ルーチン (続き)*

ルーチン名	スキーマ	説明
ADMINTABINFO および ADMIN_GET_TAB_INFO_V97	SYSIBMADM (管理ビュー)、SYSPROC (表関数)	このビューおよび表関数は、表、マテリアライズ照会表 (MQT)、階層表のサイズおよび状態の情報を戻します。
ADMINTEMPCOLUMNS ビューおよび ADMIN_GET_TEMP_COLUMNS 表関数	SYSIBMADM (管理ビュー)、SYSPROC (表関数)	このビューおよび表関数は、作成済み一時表および宣言済み一時表の列属性に関する情報を取得します。
ADMINTEMPTABLES ビューおよび ADMIN_GET_TEMP_TABLES 表関数	SYSIBMADM (管理ビュー)、SYSPROC (表関数)	このビューおよび表関数は、作成済み一時表および宣言済み一時表のインスタンスの表属性およびインスタンス生成時間に関する情報を取得します。

表 3. *管理タスク・スケジューラー・ルーチンおよびビュー*

ルーチン名またはビュー名	スキーマ	説明
ADMIN_TASK_ADD	SYSPROC	このプロシージャーは、管理タスクをスケジュールに入れます。
ADMIN_TASK_LIST	SYSTOOLS	この管理ビューは、スケジューラーで定義されている各タスクに関する情報を検索します。
ADMIN_TASK_REMOVE	SYSPROC	このプロシージャーは、スケジュールされたタスクまたはタスク状況レコードを除去します。
ADMIN_TASK_STATUS	SYSTOOLS	この管理ビューは、各タスクの状況に関する情報を検索します。
ADMIN_TASK_UPDATE	SYSPROC	このプロシージャーは、既存のタスクを更新します。

表 4. *監査ルーチンおよびプロシージャー*

ルーチン名またはビュー名	スキーマ	説明
AUDIT_ARCHIVE プロシージャーおよび表関数	SYSPROC	このプロシージャーおよび表関数は、現在の監査ログをアーカイブします。
AUDIT_DELIM_EXTRACT プロシージャー	SYSPROC	このプロシージャーは、バイナリー・アーカイブ・ログからデータを抽出し、それを区切り文字付きファイルにロードします。
AUDIT_LIST_LOGS 表関数	SYSPROC	この表関数は、現行データベースの、指定されたパスにあるアーカイブ監査ログのリストを戻します。

表 5. *自動保守管理 SQL ルーチンおよびビュー*

ルーチン名またはビュー名	スキーマ	説明
AUTOMAINT_GET_POLICY プロシージャー	SYSPROC	このプロシージャーは、データベースの現在の自動保守設定を取得します。
AUTOMAINT_GET_POLICYFILE プロシージャー	SYSPROC	このプロシージャーは、データベースの現在の自動保守設定を取得します。

表 5. 自動保守管理 SQL ルーチンおよびビュー (続き)

ルーチン名またはビュー名	スキーマ	説明
AUTOMAINT_SET_POLICY プロシージャー	SYSPROC	このプロシージャーは、現在接続中のデータベースの自動保守ポリシー設定を設定します。
AUTOMAINT_SET_POLICYFILE プロシージャー	SYSPROC	このプロシージャーは、現在接続中のデータベースの自動保守設定を設定します。

表 6. 共通 SQL API ストアード・プロシージャー

ルーチン名またはビュー名	スキーマ	説明
CANCEL_WORK プロシージャー	SYSPROC	このプロシージャーは、指定されたアクティビティーをキャンセルします。固有のアクティビティー ID をまったく指定しないと、接続済みアプリケーションのすべてのアクティビティーをキャンセルし、システムでそのアプリケーションを強制的に停止します。
GET_CONFIG プロシージャー	SYSPROC	このプロシージャーは、データ・サーバー構成データを取得します。このデータには、すべてのデータベース・パーティションの `nodes.cfg` ファイル・データ、データベース・マネージャー構成データ、データベース構成データ、およびレジストリー設定が含まれます。
GET_MESSAGE プロシージャー	SYSPROC	このプロシージャーは、簡略メッセージ・テキスト、詳細メッセージ・テキスト、および SQLCODE の SQLSTATE を取得します。
GET_SYSTEM_INFO プロシージャー	SYSPROC	このプロシージャーは、データ・サーバーに関する情報を取得します。この情報には、システム、現行インスタンス、インストール済み DB2 データベース製品、環境変数、使用可能な CPU、および他のシステム情報が含まれます。
SET_CONFIG プロシージャー	SYSPROC	このプロシージャーは、GET_CONFIG プロシージャーによって取得した構成パラメーターを更新します。

表 7. 構成管理 SQL ルーチンおよびビュー

ルーチン名またはビュー名	スキーマ	説明
DB_PARTITIONS 表関数	SYSPROC	この表関数は、表形式の `db2nodes.cfg` ファイルの内容を戻します。
DBCFG 管理ビュー	SYSIBMADM	この管理ビューは、データベース構成情報を戻します。
DBMCFG 管理ビュー	SYSIBMADM	この管理ビューは、データベース・マネージャー構成情報を戻します。
REG_VARIABLES 管理ビュー	SYSIBMADM	この管理ビューは、すべてのデータベース・パーティションから DB2 レジストリー設定値を戻します。

表 8. 環境管理ビュー

ビュー名	スキーマ	説明
ENV_FEATURE_INFO 管理ビュー	SYSPROC	この管理ビューは、ライセンスが必要とされる使用可能なすべてのフィーチャーに関する情報を戻します。
ENV_INST_INFO 管理ビュー	SYSIBMADM	この管理ビューは、現在のインスタンスについての情報を戻します。
ENV_PROD_INFO 管理ビュー	SYSIBMADM	この管理ビューは、インストール済みの DB2 データベース製品についての情報を戻します。
ENV_SYS_INFO 管理ビュー	SYSIBMADM	この管理ビューは、システムについての情報を戻します。
ENV_SYS_RESOURCES 管理ビュー	SYSIBMADM	この管理ビューは、オペレーティング・システム、CPU、メモリー、およびその他のシステム関連情報を戻します。

表 9. ヘルス・スナップショット管理 SQL ルーチン

ルーチン名	スキーマ	説明
HEALTH_CONT_HI 表関数	SYSPROC	この表関数は、データベースのヘルス・スナップショットからコンテナーに関するヘルス・インディケーター情報を載せた表を戻します。
HEALTH_CONT_HI_HIS 表関数	SYSPROC	この表関数は、データベースのヘルス・スナップショットからコンテナーに関するヘルス・インディケーター履歴情報を載せた表を戻します。
HEALTH_CONT_INFO 表関数	SYSPROC	この表関数は、データベースのヘルス・スナップショットからコンテナーに関するロールアップ・アラート状態情報を載せた表を戻します。
HEALTH_DB_HI 表関数	SYSPROC	この表関数は、データベースのヘルス・スナップショットからヘルス・インディケーター情報を載せた表を戻します。
HEALTH_DB_HI_HIS 表関数	SYSPROC	この表関数は、データベースのヘルス・スナップショットからヘルス・インディケーター履歴情報を載せた表を戻します。
HEALTH_DB_HIC 表関数	SYSPROC	この表関数は、データベースのヘルス・スナップショットからコレクション・ヘルス・インディケーターの情報を戻します。
HEALTH_DB_HIC_HIS 表関数	SYSPROC	この表関数は、データベースのヘルス・スナップショットからコレクション・ヘルス・インディケーターの履歴情報を戻します。
HEALTH_DB_INFO 表関数	SYSPROC	この表関数は、1 つまたはすべてのデータベースのヘルス・スナップショットからロールアップ・アラート状態情報を載せた表を戻します。

表 9. ヘルス・スナップショット管理 SQL ルーチン (続き)

ルーチン名	スキーマ	説明
HEALTH_DBM_HI 表関数	SYSPROC	この表関数は、DB2 データベース・マネージャーのヘルス・スナップショットからヘルス・インディケーター情報を載せた表を戻します。
HEALTH_DBM_HI_HIS 表関数	SYSPROC	この表関数は、DB2 データベース・マネージャーのヘルス・スナップショットからヘルス・インディケーター履歴情報を載せた表を戻します。
HEALTH_DBM_INFO 表関数	SYSPROC	この表関数は、DB2 データベース・マネージャーのヘルス・スナップショットからロールアップ・アラート状態情報を載せた表を戻します。
HEALTH_GET_ALERT_ACTION_CFG 表関数	SYSPROC	この表関数は、さまざまなタイプのオブジェクト (データベース・マネージャー、データベース、表スペース、および表スペース・コンテナー)、およびさまざまな構成レベル (インストール・デフォルト、インスタンス、グローバル、およびオブジェクト) のヘルス・アラート・アクション構成設定値を戻します。
HEALTH_GET_ALERT_CFG 表関数	SYSPROC	この表関数は、さまざまなタイプのオブジェクト (データベース・マネージャー、データベース、表スペース、表スペース・コンテナー)、およびさまざまな構成レベル (インストール・デフォルト、グローバル、およびオブジェクト) のヘルス・アラート構成設定値を戻します。
HEALTH_GET_IND_DEFINITION 表関数	SYSPROC	この表関数は、ヘルス・インディケーター定義を戻します。
HEALTH_HI_REC プロシージャー	SYSPROC	このプロシージャーは、特定の DB2 オブジェクト上のアラート状態にあるヘルス・インディケーターに関連した一連の推奨事項を取り出します。
HEALTH_TBS_HI 表関数	SYSPROC	この表関数は、データベースのヘルス・スナップショットから表スペースに関するヘルス・インディケーター情報を載せた表を戻します。
HEALTH_TBS_HI_HIS 表関数	SYSPROC	この表関数は、データベースのヘルス・スナップショットから表スペースに関するヘルス・インディケーター履歴情報を載せた表を戻します。
HEALTH_TBS_INFO 表関数	SYSPROC	この表関数は、データベースのヘルス・スナップショットから表スペースに関するロールアップ・アラート状態情報を載せた表を戻します。

表 10. モニター SQL ルーチン

ルーチン名	スキーマ	説明
EVMON_FORMAT_UE_TO_TABLES プロシージャー	SYSPROC	このプロシージャーは、フォーマットされていないイベント表に保管されているデータを取り出し、その XML 文書を一連のリレーショナル表に移動します。
EVMON_FORMAT_UE_TO_TABLES 表関数	SYSPROC	この表関数は、バイナリー・イベントを未フォーマット・イベント表から抽出し、XML 文書にフォーマットします。
MON_GET_ACTIVITY_DETAILS	SYSPROC	この表関数は、アクティビティーに関する詳細情報を戻します。これには、一般的なアクティビティー情報と、アクティビティーのメトリックの集合が含まれます。
MON_GET_BUFFERPOOL 表関数	SYSPROC	この表関数は、1 つ以上のバッファー・プールのモニター・メトリックを戻します。
MON_GET_CONNECTION 表関数	SYSPROC	この表関数は、1 つ以上の接続のメトリックを戻します。
MON_GET_CONNECTION_DETAILS 表関数	SYSPROC	この表関数は、1 つ以上の接続の詳細メトリックを戻します。
MON_GET_CONTAINER 表関数	SYSPROC	この表関数は、1 つ以上の表スペース・コンテナーのモニター・メトリックを戻します。
MON_GET_EXTENT_MOVEMENT_STATUS 表関数	SYSPROC	この表関数は、エクステント移動操作の状況を戻します。
MON_GET_INDEX 表関数	SYSPROC	この表関数は、1 つ以上の索引のメトリックを戻します。
MON_GET_PKG_CACHE_STMT 表関数	SYSPROC	この表関数は、データベース・パッケージ・キャッシュ内の静的 SQL ステートメントと動的 SQL ステートメントの両方のポイント・イン・タイム・ビューを戻します。
MON_GET_SERVICE_SUBCLASS 表関数	SYSPROC	この表関数は、1 つ以上のサービス・サブクラスのメトリックを戻します。
MON_GET_SERVICE_SUBCLASS_DETAILS 表関数	SYSPROC	この表関数は、1 つ以上のサービス・サブクラスの詳細メトリックを戻します。
MON_GET_TABLE 表関数	SYSPROC	この表関数は、1 つ以上の表のモニター・メトリックを戻します。
MON_GET_TABLESPACE 表関数	SYSPROC	この表関数は、1 つ以上の表スペースのモニター・メトリックを戻します。
MON_GET_UNIT_OF_WORK 表関数	SYSPROC	この表関数は、1 つ以上の作業単位のメトリックを戻します。

表 10. モニター SQL ルーチン (続き)

ルーチン名	スキーマ	説明
MON_GET_UNIT_OF_WORK_DETAILS 表関数	SYSPROC	この表関数は、1 つ以上の作業単位の詳細メトリックを戻します。
MON_GET_WORKLOAD 表関数	SYSPROC	この表関数は、1 つ以上のワークロードのメトリックを戻します。
MON_GET_WORKLOAD_DETAILS 表関数	SYSPROC	この表関数は、1 つ以上のワークロードの詳細メトリックを戻します。

表 11. MQSeries 管理 SQL ルーチン

ルーチン名	スキーマ	説明
MQPUBLISH スカラー関数	DB2MQ, DB2MQ1C	このスカラー関数は、MQSeries ロケーションに対してデータを公開します。
MQREAD スカラー関数	DB2MQ, DB2MQ1C	このスカラー関数は、MQSeries ロケーションからメッセージを戻します。
MQREADALL 表関数	DB2MQ, DB2MQ1C	この表関数は、MQSeries ロケーションからメッセージとメッセージ・メタデータを示した表を戻します。
MQREADALLCLOB 表関数	DB2MQ	この表関数は、指定された MQSeries ロケーションから、メッセージとメッセージ・メタデータの入った表を戻します。
MQREADCLOB スカラー関数	DB2MQ	このスカラー関数は、指定された MQSeries ロケーションからメッセージを戻します。
MQRECEIVE スカラー関数	DB2MQ, DB2MQ1C	このスカラー関数は、MQSeries ロケーションからメッセージを戻し、それに関連したキューからメッセージを除去します。
MQRECEIVEALL 表関数	DB2MQ, DB2MQ1C	この表関数は、MQSeries ロケーションからメッセージとメッセージ・メタデータの入った表を戻し、それに関連したキューからメッセージを除去します。
MQRECEIVEALLCLOB 表関数	DB2MQ	この表関数は、指定された MQSeries ロケーションから、メッセージとメッセージ・メタデータの入った表を戻します。
MQRECEIVECLOB スカラー関数	DB2MQ	このスカラー関数は、指定された MQSeries ロケーションからメッセージを戻します。
MQSEND スカラー関数	DB2MQ, DB2MQ1C	このスカラー関数は、MQSeries ロケーションにデータを送信します。
MQSUBSCRIBE スカラー関数	DB2MQ, DB2MQ1C	このスカラー関数は、特定のトピックに関して公開された MQSeries メッセージにサブスクライブします。
MQUNSUBSCRIBE スカラー関数	DB2MQ, DB2MQ1C	このスカラー関数は、特定のトピックに関して公開された MQSeries メッセージからアンサブスクライブします。

表 12. セキュリティー管理 SQL ルーチンおよびビュー

ルーチン名またはビュー名	スキーマ	説明
AUTH_LIST_AUTHORITIES_FOR_AUTHID 表関数	SYSPROC	この表関数は、データベース構成ファイルにあるか、許可 ID に直接付与されたか、あるいはグループまたはロールを介して間接的に付与された許可 ID によって保持されているすべての権限を戻します。
AUTH_LIST_GROUPS_FOR_AUTHID 表関数	SYSPROC	この表関数は、与えられた許可 ID がメンバーになっているグループのリストを戻します。
AUTH_LIST_ROLES_FOR_AUTHID 関数	SYSPROC	この関数は、与えられた許可 ID がメンバーになっているロールのリストを戻します。
AUTHORIZATIONIDS 管理ビュー	SYSIBMADM	この管理ビューには、現在接続中のデータベースに対して特権または権限を付与された許可 ID のリストが、それらのタイプとともに含まれます。
OBJECTOWNERS 管理ビュー	SYSIBMADM	この管理ビューには、現在接続中のデータベースに関するすべてのオブジェクト所有権情報が含まれます。
PRIVILEGES 管理ビュー	SYSIBMADM	この管理ビューには、現在接続中のデータベースに対する明示的な特権すべてが含まれます。

表 13. スナップショット管理 SQL ルーチンおよびビュー

ルーチン名またはビュー名	スキーマ	説明
APPL_PERFORMANCE 管理ビュー	SYSIBMADM	この管理ビューは、選択行数と読み取り行数の比率の情報をアプリケーションごとに表示します。
APPLICATIONS 管理ビュー	SYSIBMADM	この管理ビューは、接続されているデータベース・アプリケーションに関する情報を戻します。
BP_HITRATIO 管理ビュー	SYSIBMADM	この管理ビューは、データベースでのバッファー・プール・ヒット率 (合計、データ、索引) を戻します。
BP_READ_IO 管理ビュー	SYSIBMADM	この管理ビューは、バッファー・プール読み取りパフォーマンス情報を戻します。
BP_WRITE_IO 管理ビュー	SYSIBMADM	この管理ビューは、バッファー・プールあたりのバッファー・プール書き込みパフォーマンス情報を戻します。
CONTAINER_UTILIZATION 管理ビュー	SYSIBMADM	この管理ビューは、表スペース・コンテナーと使用率についての情報を戻します。
LOCKS_HELD 管理ビュー	SYSIBMADM	この管理ビューは、現在のロック保持数の情報を戻します。
LOCKWAITS 管理ビュー	SYSIBMADM	この管理ビューは、付与待機中ロック数の情報を戻します。

表 13. スナップショット管理 SQL ルーチンおよびビュー (続き)

ルーチン名またはビュー名	スキーマ	説明
LOG_UTILIZATION 管理ビュー	SYSIBMADM	この管理ビューは、現在接続されているデータベースのログ使用率についての情報を戻します。
LONG_RUNNING_SQL 管理ビュー	SYSIBMADM	この管理ビューは、現在接続されているデータベースで実行時間が最も長い SQL ステートメントを戻します。
QUERY_PREP_COST 管理ビュー	SYSIBMADM	この管理ビューは、ステートメントのリストを、ステートメントの準備に必要な時間に関する情報とともに戻します。
SNAP_WRITE_FILE プロシージャー	SYSPROC	このプロシージャーは、システム・スナップショット・データを、インスタンス・ディレクトリーの tmp サブディレクトリーにあるファイルに書き込みます。
SNAPAGENT 管理ビューおよび SNAP_GET_AGENT 表関数	SYSIBMADM (管理ビュー)、SYSPROC (表関数)	この管理ビューおよび表関数は、アプリケーション・スナップショットから、特に agent 論理データ・グループのエージェント情報を戻します。
SNAPAGENT_MEMORY_POOL 管理ビューおよび SNAP_GET_AGENT_MEMORY_POOL 表関数	SYSIBMADM (管理ビュー)、SYSPROC (表関数)	この管理ビューおよび表関数は、エージェント・レベルでのメモリー使用量についての情報を戻します。
SNAPAPPL 管理ビューおよび SNAP_GET_APPL_V95 表関数	SYSIBMADM (管理ビュー)、SYSPROC (表関数)	この管理ビューおよび表関数は、アプリケーション・スナップショットから、特に appl 論理データ・グループのアプリケーション情報を戻します。
SNAPAPPL_INFO 管理ビューおよび SNAP_GET_APPL_INFO_V95 表関数	SYSIBMADM (管理ビュー)、SYSPROC (表関数)	この管理ビューおよび表関数は、アプリケーション・スナップショットから、特に appl_info 論理データ・グループのアプリケーション情報を戻します。
SNAPBP 管理ビューおよび SNAP_GET_BP_V95 表関数	SYSIBMADM (管理ビュー)、SYSPROC (表関数)	この管理ビューおよび表関数は、バッファー・プール・スナップショットから、特に bufferpool 論理データ・グループのバッファー・プール情報を戻します。
SNAPBP_PART 管理ビューおよび SNAP_GET_BP_PART 表関数	SYSIBMADM (管理ビュー)、SYSPROC (表関数)	この管理ビューおよび表関数は、バッファー・プール・スナップショットから、特に bufferpool_nodeinfo 論理データ・グループのバッファー・プール情報を戻します。
SNAPCONTAINER 管理ビューおよび SNAP_GET_CONTAINER_V91 表関数	SYSIBMADM (管理ビュー)、SYSPROC (表関数)	この管理ビューおよび表関数は、tablespace_container 論理データ・グループからの表スペース・スナップショット情報を戻します。
SNAPDB 管理ビューおよび SNAP_GET_DB_V95 表関数	SYSIBMADM (管理ビュー)、SYSPROC (表関数)	この管理ビューおよび表関数は、データベース (dbase) およびデータベース・ストレージ (db_storage_group) 論理グループからのスナップショット情報を戻します。

表 13. スナップショット管理 SQL ルーチンおよびビュー (続き)

ルーチン名またはビュー名	スキーマ	説明
SNAPDB_MEMORY_POOL 管理ビューおよび SNAP_GET_DB_MEMORY_POOL 表関数	SYSIBMADM (管理ビュー)、SYSPROC (表関数)	この管理ビューおよび表関数は、データベース・レベルでのメモリー使用量についての情報を戻します (UNIX® オペレーティング・システムの場合のみ)。
SNAPDBM 管理ビューおよび SNAP_GET_DBM_V95 表関数	SYSIBMADM (管理ビュー)、SYSPROC (表関数)	この管理ビューおよび表関数は、DB2 データベース・マネージャー (dbm) 論理グループのスナップショット・モニター情報を戻します。
SNAPDBM_MEMORY_POOL 管理ビューおよび SNAP_GET_DBM_MEMORY_POOL 表関数	SYSIBMADM (管理ビュー)、SYSPROC (表関数)	この管理ビューおよび表関数は、データベース・マネージャーでのメモリー使用量についての情報を戻します。
SNAPDETAILLOG 管理ビューおよび SNAP_GET_DETAILLOG_V91 表関数	SYSIBMADM (管理ビュー)、SYSPROC (表関数)	この管理ビューおよび表関数は、detail_log 論理データ・グループからのスナップショット情報を戻します。
SNAPDYN_SQL 管理ビューおよび SNAP_GET_DYN_SQL_V95 表関数	SYSIBMADM (管理ビュー)、SYSPROC (表関数)	この管理ビューおよび表関数は、dynsql 論理データ・グループからのスナップショット情報を戻します。
SNAPFCM 管理ビューおよび SNAP_GET_FCM 表関数	SYSIBMADM (管理ビュー)、SYSPROC (表関数)	この管理ビューおよび表関数は、データベース・マネージャー・スナップショットから、特に fcm 論理データ・グループの高速コミュニケーション・マネージャー (FCM) 情報を戻します。
SNAPFCM_PART 管理ビューおよび SNAP_GET_FCM_PART 表関数	SYSIBMADM (管理ビュー)、SYSPROC (表関数)	この管理ビューおよび表関数は、データベース・マネージャー・スナップショットから、特に fcm_node 論理データ・グループの高速コミュニケーション・マネージャー (FCM) 情報を戻します。
SNAPHADR 管理ビューおよび SNAP_GET_HADR 表関数	SYSIBMADM (管理ビュー)、SYSPROC (表関数)	この管理ビューおよび表関数は、データベース・スナップショットから、特に hadr 論理データ・グループの高可用性災害時リカバリー情報を戻します。
SNAPLOCK 管理ビューおよび SNAP_GET_LOCK 表関数	SYSIBMADM (管理ビュー)、SYSPROC (表関数)	この管理ビューおよび表関数は、特に lock 論理データ・グループのロック・スナップショット情報を戻します。
SNAPLOCKWAIT 管理ビューおよび SNAP_GET_LOCKWAIT 表関数	SYSIBMADM (管理ビュー)、SYSPROC (表関数)	この管理ビューおよび表関数は、特に lockwait 論理データ・グループのロック待機スナップショット情報を戻します。
SNAPSTMT 管理ビューおよび SNAP_GET_STMT 表関数	SYSIBMADM (管理ビュー)、SYSPROC (表関数)	この管理ビューおよび表関数は、アプリケーション・スナップショットからステートメントに関する情報を戻します。
SNAPSTORAGE_PATHS 管理ビューおよび SNAP_GET_STORAGE_PATHS 表関数	SYSIBMADM (管理ビュー)、SYSPROC (表関数)	この管理ビューおよび表関数は、特に db_storage_group 論理データ・グループから、データベースの自動ストレージ・パスのリストを、ストレージ・パスごとのファイル・システム情報を含めて戻します。

表 13. スナップショット管理 SQL ルーチンおよびビュー (続き)

ルーチン名またはビュー名	スキーマ	説明
SNAPSUBSECTION 管理ビューおよび SNAP_GET_SUBSECTION 表関数	SYSIBMADM (管理ビュー)、SYSPROC (表関数)	この管理ビューおよび表関数は、アプリケーション・サブセクション情報として、subsection 論理モニター・グループの情報を戻します。
SNAPSWITCHES 管理ビューおよび SNAP_GET_SWITCHES 表関数	SYSIBMADM (管理ビュー)、SYSPROC (表関数)	この管理ビューおよび表関数は、データベース・スナップショット切り替え状態に関する情報を戻します。
SNAPTAB 管理ビューおよび SNAP_GET_TAB_V91 表関数	SYSIBMADM (管理ビュー)、SYSPROC (表関数)	この管理ビューおよび表関数は、table 論理データ・グループからのスナップショット情報を戻します。
SNAPTAB_REORG 管理ビューおよび SNAP_GET_TAB_REORG 表関数	SYSIBMADM (管理ビュー)、SYSPROC (表関数)	この管理ビューおよび表関数は、表再編成情報を戻します。
SNAPTBSP 管理ビューおよび SNAP_GET_TBSP_V91 表関数	SYSIBMADM (管理ビュー)、SYSPROC (表関数)	この管理ビューおよび表関数は、table space 論理データ・グループからのスナップショット情報を戻します。
SNAPTBSP_PART 管理ビューおよび SNAP_GET_TBSP_PART_V91 表関数	SYSIBMADM (管理ビュー)、SYSPROC (表関数)	この管理ビューおよび表関数は、tablespace_nodeinfo 論理データ・グループからのスナップショット情報を戻します。
SNAPTBSP_QUIESCER 管理ビューおよび SNAP_GET_TBSP_QUIESCER 表関数	SYSIBMADM (管理ビュー)、SYSPROC (表関数)	この管理ビューおよび表関数は、表スペース・スナップショットから、静止プログラムに関する情報を戻します。
SNAPTBSP_RANGE 管理ビューおよび SNAP_GET_TBSP_RANGE 表関数	SYSIBMADM (管理ビュー)、SYSPROC (表関数)	この管理ビューおよび表関数は、範囲スナップショットから情報を戻します。
SNAPUTIL 管理ビューおよび SNAP_GET_UTIL 表関数	SYSIBMADM (管理ビュー)、SYSPROC (表関数)	この管理ビューおよび表関数は、utility_info 論理データ・グループからのユーティリティー・スナップショット情報を戻します。
SNAPUTIL_PROGRESS 管理ビューおよび SNAP_GET_UTIL_PROGRESS 表関数	SYSIBMADM (管理ビュー)、SYSPROC (表関数)	この管理ビューおよび表関数は、特に progress 論理データ・グループのユーティリティー進行状況情報を戻します。
TBSP_UTILIZATION 管理ビュー	SYSIBMADM	この管理ビューは、表スペースの構成および使用率の情報を戻します。
TOP_DYNAMIC_SQL 管理ビュー	SYSIBMADM	この管理ビューは、実行数、平均実行時間、ソート数、またはステートメントあたりのソートによってソートできる動的 SQL ステートメントのうち、上位のものを戻します。

表 14. SQL プロシージャー管理 SQL ルーチン

ルーチン名	スキーマ	説明
ALTER_ROUTINE_PACKAGE プロシージャー	SYSPROC	このプロシージャーは、コンパイル済み SQL ルーチンまたはコンパイル済みトリガーに関連付けられたパッケージに使用される値を、再バインドを必要とせずに変更します。

表 14. SQL プロシージャー管理 SQL ルーチン (続き)

ルーチン名	スキーマ	説明
GET_ROUTINE_OPTS スカラー関数	SYSPROC	このスカラー関数は、現行セッションでの SQL プロシージャーの作成に使われる予定のオプションの文字ストリング値を戻します。
GET_ROUTINE_SAR プロシージャー	SYSFUN	このプロシージャーは、最低限同じレベルおよびオペレーティング・システムで実行している別のデータベース・サーバーに、同一のルーチンをインストールするのに必要な情報を戻します。
PUT_ROUTINE_SAR プロシージャー	SYSFUN	このプロシージャーは、データベース・サーバーで SQL ルーチンを作成したり定義したりするのに必要な情報を渡します。
REBIND_ROUTINE_PACKAGE プロシージャー	SYSPROC	このプロシージャーは、SQL プロシージャーに関連したパッケージを再バインドします。
SET_ROUTINE_OPTS プロシージャー	SYSPROC	このプロシージャーは、現行セッションでの SQL プロシージャーの作成に使われる予定のオプションを設定します。

表 15. 段階的な再配分管理 SQL ルーチン

ルーチン名	スキーマ	説明
ANALYZE_LOG_SPACE プロシージャー	SYSPROC	このプロシージャーは、ログ・スペース分析情報を戻します。
GENERATE_DISTFILE プロシージャー	SYSPROC	このプロシージャーは、データ配分ファイルを生成します。
GET_SWRD_SETTINGS プロシージャー	SYSPROC	このプロシージャーは、再配分情報を戻します。
SET_SWRD_SETTINGS プロシージャー	SYSPROC	このプロシージャーは、再配分レジストリーを作成または変更します。
STEPWISE_REDISTRIBUTE_DBPG プロシージャー	SYSPROC	このプロシージャーは、データベース・パーティション・グループに属するものを再配分します。

表 16. ストレージ管理ツール管理 SQL ルーチン

ルーチン名	スキーマ	説明
CAPTURE_STORAGEMGMT_INFO プロシージャー	SYSPROC	このプロシージャーは、所定のルート・オブジェクトのストレージ関連情報を戻します。
CREATE_STORAGEMGMT_TABLES プロシージャー	SYSPROC	このプロシージャーは、ストレージ管理表を作成します。
DROP_STORAGEMGMT_TABLES プロシージャー	SYSPROC	このプロシージャーは、すべてのストレージ管理表をドロップします。

表 17. テキスト検索管理 SQL ルーチン

ルーチン名	スキーマ	説明
SYSTS_ADMIN_CMD ストアード・プロシージャー	SYSPROC	このプロシージャーは、SQL CALL ステートメントを使用してテキスト検索管理コマンドを実行します。
SYSTS_ALTER プロシージャー	SYSPROC	このプロシージャーは、索引の更新特性を変更します。
SYSTS_CLEAR_COMMANDLOCKS プロシージャー	SYSPROC	このプロシージャーは、データベース内の特定のテキスト検索索引またはすべてのテキスト検索索引のすべてのコマンド・ロックを解除します。
SYSTS_CLEAR_EVENTS プロシージャー	SYSPROC	このプロシージャーは、管理に使用される索引のイベント表から、索引付けイベントを削除します。
SYSTS_CREATE プロシージャー	SYSPROC	このプロシージャーは、テキスト検索関数を使用して列データを検索可能にする、テキスト列のテキスト検索索引を作成します。
SYSTS_DISABLE プロシージャー	SYSPROC	このプロシージャーは、現行データベースの DB2 テキスト検索を使用不可にします。
SYSTS_DROP プロシージャー	SYSPROC	このプロシージャーは、表列と関連付けられた既存のテキスト検索索引をドロップします。
SYSTS_ENABLE プロシージャー	SYSPROC	このプロシージャーは、データベース内の表の列に対してテキスト検索索引を作成する前に、正常に実行する必要があります。
SYSTS_UPDATE プロシージャー	SYSPROC	このプロシージャーは、索引が関連付けられるテキスト列の現行の内容を反映するテキスト検索索引を更新します。

表 18. ワークロード管理用の管理 SQL ルーチン

ルーチン名	スキーマ	説明
WLM_CANCEL_ACTIVITY プロシージャー	SYSPROC	このプロシージャーは、指定されたアクティビティーをキャンセルします。
WLM_CAPTURE_ACTIVITY_IN_PROGRESS プロシージャー	SYSPROC	このプロシージャーは、指定されたアクティビティーに関する情報をアクティビティー・イベント・モニターに送信します。
WLM_COLLECT_STATS プロシージャー	SYSPROC	このプロシージャーは、サービス・クラス、ワークロード、作業クラス、およびしきい値キューの統計を統計イベント・モニターに送信し、その統計のメモリー内コピーをリセットします。
WLM_GET_QUEUE_STATS 表関数	SYSPROC	この表関数は、1 つ以上のしきい値キューの基本統計情報を戻します。

表 18. ワークロード管理用の管理 SQL ルーチン (続き)

ルーチン名	スキーマ	説明
WLM_GET_SERVICE_CLASS_AGENTS_V97 表関数	SYSPROC	この表関数は、SERVICE_SUPERCLASS_NAME および SERVICE_SUBCLASS_NAME によって指定されたサービス・クラスで実行しているか、または APPLICATION_HANDLE によって指定されたアプリケーションの代わりに実行している、指定されたパーティション上のエージェントのリストを戻します。
WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES_V97 表関数	SYSPROC	この表関数は、特定のパーティション上の指定されたサービス・クラスで実行しているすべてのワークロード・オカレンスのリストを戻します。
WLM_GET_SERVICE_SUBCLASS_STATS_V97 表関数	SYSPROC	この表関数は、1 つ以上のサービス・サブクラスの基本統計を戻します。
WLM_GET_SERVICE_SUPERCLASS_STATS 表関数	SYSPROC	この表関数は、1 つ以上のサービス・スーパークラスの基本統計を戻します。
WLM_GET_WORK_ACTION_SET_STATS 表関数	SYSPROC	この表関数は、作業アクション・セット内の作業クラスの基本統計を戻します。
WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES_V97 表関数	SYSPROC	この表関数は、指定されたパーティション上の特定のアプリケーションからサブミットされ、また完了していないすべてのアクティビティーのリストを戻します。
WLM_GET_WORKLOAD_STATS_V97 表関数	SYSPROC	この表関数は、1 つ以上のワークロードの基本統計を戻します。
WLM_SET_CLIENT_INFO プロシージャー	SYSPROC	このプロシージャーは、DB2 データベース・サーバーでの現行接続に関連付けられたクライアント情報を設定します。

表 19. その他の管理 SQL ルーチンおよびビュー

ルーチン名またはビュー名	スキーマ	説明
ADMIN_COPY_SCHEMA プロシージャー	SYSPROC	このプロシージャーは、特定のスキーマと、その中に含まれているすべてのオブジェクトをコピーするために使用されます。
ADMIN_DROP_SCHEMA プロシージャー	SYSPROC	このプロシージャーは、特定のスキーマと、その中に含まれているすべてのオブジェクトをドロップするために使用されます。
ADMIN_MOVE_TABLE プロシージャー	SYSPROC	このプロシージャーは、アクティブな表のデータを、同じ名前の新しい表オブジェクトに移動します。データはオンラインのままなので、引き続きアクセス可能です。
ADMIN_MOVE_TABLE_UTIL プロシージャー	SYSPROC	このプロシージャーは、ADMIN_MOVE_TABLE プロシージャーによって使用されるユーザー定義可能値を変更します。

表 19. その他の管理 SQL ルーチンおよびビュー (続き)

ルーチン名またはビュー名	スキーマ	説明
ALTOBJ プロシージャー	SYSPROC	このプロシージャーは、入力 CREATE TABLE ステートメントをターゲット表定義として使用して既存の表を変更します。
APPLICATION_ID スカラー関数	SYSFUN	このスカラー関数は、現行接続のアプリケーション ID を戻します。
COMPILATION_ENV 表関数	SYSPROC	この表関数は、コンパイル環境のエレメントを戻します。
CONTACTGROUPS 管理ビュー	SYSIBMADM	この管理ビューは、連絡先グループのリストを戻します。
CONTACTS 管理ビュー	SYSIBMADM	この管理ビューは、データベース・サーバーで定義されている連絡先のリストを戻します。
DB_HISTORY 管理ビュー	SYSIBMADM	この管理ビューは、現在接続されているデータベース・パーティションに関連付けられた履歴ファイルからの情報を戻します。
DBPATHS 管理ビュー	SYSIBMADM	この管理ビューは、分割ミラー・バックアップなどのタスクに必要なデータベース・パスの値を戻します。
EXPLAIN_FORMAT_STATS スカラー関数	SYSPROC	この新規のスカラー関数は、定様式の統計情報を表示するために使用されます。この情報は構文解析され、特定の照会についてキャプチャーされた Explain スナップショットから抽出されます。
EXPLAIN_GET_MSGS 表関数	スキーマは Explain 表スキーマと同じです。	この表関数は、EXPLAIN_DIAGNOSTIC および EXPLAIN_DIAGNOSTIC_DATA Explain 表を照会し、定様式メッセージを戻します。
GET_DBSIZE_INFO プロシージャー	SYSPROC	このプロシージャーは、データベース・サイズと最大容量を計算します。
NOTIFICATIONLIST 管理ビュー	SYSIBMADM	この管理ビューは、インスタンスの状況が通知される連絡先および連絡先グループのリストを戻します。
PD_GET_DIAG_HIST 表関数	SYSPROC	この表関数は、指定された機能からログ・レコード、イベント・レコード、および通知レコードを戻します。
PDLOGMSGS_LAST24HOURS 管理ビューおよび PD_GET_LOG_MSGS 表関数	SYSIBMADM (管理ビュー)、SYSPROC (表関数)	この管理ビューおよび表関数は、DB2 通知ログに記録された問題判別ログ・メッセージを戻します。この情報は、データベース管理者とシステム管理者が使用するためのものです。
REORGCHK_IX_STATS プロシージャー	SYSPROC	このプロシージャーは、索引統計を調べて、再編成の必要があるかどうかを判別します。
REORGCHK_TB_STATS プロシージャー	SYSPROC	このプロシージャーは、表統計を調べて、再編成の必要があるかどうかを判別します。

表 19. その他の管理 SQL ルーチンおよびビュー (続き)

ルーチン名またはビュー名	スキーマ	説明
SQLERRM スカラー関数	SYSPROC	このスカラー関数には 2 つのバージョンがあります。1 つは、メッセージ・トークンの使用や言語選択などを含む、十分に柔軟性をもたせたメッセージ検索を提供します。もう 1 つは、SQLCODE のみを入力パラメーターとして取り、短メッセージを英語で戻すという、インターフェースが簡単なものです。
SYSINSTALLOBJECTS プロシージャー	SYSPROC	このプロシージャーは、特定のツールで必要なデータベース・オブジェクトを作成またはドロップします。
MON_GET_FCM	SYSPROC	この表関数は、高速コミュニケーション・マネージャー (FCM) に関するメトリックを戻します。
MON_GET_FCM_CONNECTION_LIST	SYSPROC	この表関数は、指定されたメンバーのすべての高速コミュニケーション・マネージャー (FCM) 接続に関するモニター・メトリックを戻します。

第 3 章 アクティビティー・モニター・ルーチン

AM_BASE_RPT_RECOMS - アクティビティー・レポートに関する推奨事項

AM_BASE_RPT_RECOMS 表関数は、アクティビティー・モニターで使用されるアクティビティー・レポートに関する推奨事項を戻します。

構文

スキーマは SYSPROC です。

表関数パラメーター

report-id
レポート ID を指定する、タイプ INTEGER の入力引数。引数が NULL の場合、すべての使用可能なレポートの推奨事項が戻されます。

client-locale
クライアントの言語 ID を指定する、タイプ VARCHAR(33) の入力引数。引数が NULL または空ストリングである場合、デフォルト値は 'En_US' (英語) です。指定したロケールのメッセージ・ファイルがサーバーで使用できない場合には、 'En_US' が使用されます。

許可

AM_BASE_RPT_RECOMS 表関数に対する EXECUTE 特権。

例

例 1: ID が 1 のアクティビティー・モニター・レポート用に推奨事項を要求します (英語)。デフォルトのクライアント言語 ID である 'En_US' が前提となります。

```
SELECT *
   FROM TABLE(SYSPROC.AM_BASE_RPT_RECOMS(1, CAST(NULL AS VARCHAR(33))))
   AS RECOMS
```

例 2: ID が 12 のアクティビティー・モニター・レポート用に推奨事項を要求します (フランス語)。

```
SELECT *
   FROM TABLE(SYSPROC.AM_BASE_RPT_RECOMS(12, CAST('Fr_FR' AS VARCHAR(33))))
   AS RECOMS
```

© Copyright IBM Corp. 2006, 2010

戻される情報

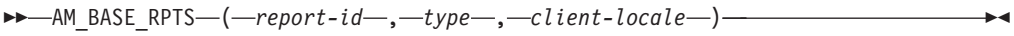
表 20. AM_BASE_RPT_RECOMS 表関数によって戻される情報

列名	データ・タイプ	説明
REPORT_ID	INTEGER	レポート ID。
RECOM_NAME	VARCHAR(256)	推奨事項の名前または簡略説明。
RECOM_DESCRIPTION	CLOB(32K)	推奨事項の詳細記述。

AM_BASE_RPTS - アクティビティー・モニター・レポート

AM_BASE_RPTS 表関数は、アクティビティー・モニターで使用されるアクティビティー・レポートを戻します。

構文



スキーマは SYSPROC です。

表関数パラメーター

report-id
固有のレポート ID を指定する、タイプ INTEGER の入力引数。引数が NULL の場合、レポート ID が示されたレポートが戻されます。

type
レポート・タイプを指定する、タイプ CHAR(4) の入力引数。有効な値は以下のとおりです。

APPL アプリケーション

STMT SQL ステートメント

TRAN トランザクション

CACH 動的 SQL ステートメント・キャッシュ

値は、大文字または小文字で指定できます。引数が NULL または空ストリングである場合、すべてのタイプのレポートが戻されます。

client-locale
クライアントの言語 ID を指定する、タイプ VARCHAR(33) の入力引数。引数が NULL または空ストリングであるか、指定したロケールのメッセージ・ファイルがサーバーで使用できない場合には、'En_US' が使用されます。

許可

AM_BASE_RPTS 表関数に対する EXECUTE 特権。

例

例 1:

```
SELECT * FROM TABLE(SYSPROC.AM_BASE_RPTS(CAST(NULL AS INTEGER),
   CAST(NULL AS CHAR(4)), CAST(NULL AS VARCHAR(33)))) AS REPORTS
```

例 2:

```
SELECT ID, NAME FROM TABLE(SYSPROC.AM_BASE_RPTS(
   CAST(NULL AS INTEGER), CAST('STMT' AS CHAR(4)), 'En_US'))
   AS REPORTS WHERE TYPE = 'STMT'
```

戻される情報

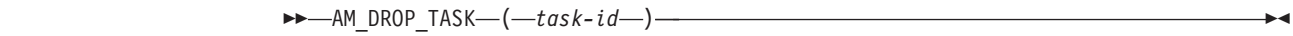
表 21. AM_BASE_RPTS 表関数によって戻される情報

列名	データ・タイプ	説明
ID	INTEGER	固有のレポート ID。
TYPE	CHAR(4)	レポート・タイプ。有効な値は、APPL、 STMT、 TRAN、 CACH です。
NAME	VARCHAR(256)	レポートの名前または簡略説明。
DESCRIPTION	VARCHAR(16384)	レポートの詳細記述。
SWITCHES	VARCHAR(100)	このレポートに必要なモニター・スイッチ。

AM_DROP_TASK - モニター・タスクの削除

AM_DROP_TASK プロシージャーは、モニター・タスクを削除します。戻されるデータはありません。

構文



スキーマは SYSPROC です。

プロシージャー・パラメーター

task-id
固有のモニター・タスク ID を指定する、タイプ INTEGER の入力引数。

許可

AM_DROP_TASK プロシージャーに対する EXECUTE 特権。

例

ID 5 のモニター・タスクをドロップします。

```
CALL SYSPROC.AM_DROP_TASK(5)
```

AM_GET_LOCK_CHN_TB - 表形式のアプリケーション・ロック・チェーン・データの検索

AM_GET_LOCK_CHN_TB プロシージャーは、アプリケーション・ロック・チェーン・データを、タブ区切りフォーマットで戻します。ロック・チェーンは、現行のアプリケーションが、直接的または間接的に、保留しているまたは待機しているすべてのアプリケーションで構成されます。

構文

スキーマは SYSPROC です。

プロシージャー・パラメーター

agent-id
ロック・チェーン・データを検索する対象のアプリケーションのエージェント ID を指定する、タイプ BIGINT の入力引数。

許可

- SYSMON 権限
- AM_GET_LOCK_CHN_TB プロシージャーに対する EXECUTE 特権。

例

エージェント ID 68 に関するロック・チェーン情報を検索します。

```
CALL SYSPROC.AM_GET_LOCK_CHN_TB(68)
```

戻される情報

このプロシージャーにより、次に示されている表が戻されます。表のそれぞれの行は、ロック待機の関係を表します。さらに、結果セットには、保留のみのアプリケーションごとに 1 つの行が示されます。ここでは、 HOLDING_AGENT_ID 列が NULL で、他の 4 つの列は保留のみのアプリケーション用です。

表 22. AM_GET_LOCK_CHN_TB プロシージャーによって戻される情報

列名	データ・タイプ	説明
HOLDING_AGENT_ID	BIGINT	ロックを保留しているアプリケーションのエージェント ID。
AGENT_ID	BIGINT	ロックを待機しているアプリケーションのエージェント ID。
APPL_NAME	VARCHAR(255)	ロックを待機しているアプリケーションの名前。
AUTH_ID	VARCHAR(128)	ロックを待機しているアプリケーションの許可 ID。
APPL_ID	VARCHAR(64)	ロックを待機しているアプリケーションのアプリケーション ID。

AM_GET_LOCK_CHNS - 特定のアプリケーションに関するロック・チェーン情報の検索

AM_GET_LOCK_CHNS プロシージャーは、指定したアプリケーションのロック・チェーンを、フォーマット済みストリングとして戻します。ロック・チェーンは、現行のアプリケーションが、直接的または間接的に、保留しているまたは待機しているすべてのアプリケーションで構成されます。

構文

スキーマは SYSPROC です。

プロシージャー・パラメーター

agent-id
ロック・チェーンが表示されるアプリケーションのエージェント ID を指定する、タイプ BIGINT の入力引数。

lock-chains
指定したアプリケーションの全ロック・チェーンを表示する、タイプ CLOB(2M) の出力引数。

許可

- SYSMON 権限
- AM_GET_LOCK_CHNS プロシージャーに対する EXECUTE 特権。

例

```
CALL SYSPROC.AM_GET_LOCK_CHNS(17,?)

  Value of output parameters
  --------------------------
  Parameter Name  : LOCK_CHAINS
  Parameter Value : >db2bp.exe (Agent ID: 17) (Auth ID: AMUSERC )

<db2bp.exe (Agent ID: 17) (Auth ID: AMUSERC )
    <db2bp.exe (Agent ID: 18) (Auth ID: AMUSERB )
        <db2bp.exe (Agent ID: 16) (Auth ID: AMUSERA )

  Return Status = 0
```

AM_GET_LOCK_RPT - アプリケーション・ロックに関する詳細の検索

AM_GET_LOCK_RPT プロシージャーは、アプリケーションのロック詳細を、 3 つの出力結果セットで戻します。

構文

スキーマは SYSPROC です。

プロシージャー・パラメーター

agent-id
ロック詳細が戻されるアプリケーションのエージェント ID を指定する、タイプ BIGINT の入力引数。

許可

- SYSMON 権限
- AM_GET_LOCK_RPT プロシージャーに対する EXECUTE 特権。

例

```
CALL SYSPROC.AM_GET_LOCK_RPT(68)
```

使用上の注意

このプロシージャーから情報が戻されるようにするには、DFT_MON_LOCK モニター・スイッチをオンにしておかなければなりません。

戻される情報

このプロシージャーは、3 つの結果セット (アプリケーションの一般情報用、アプリケーションが保持するロック用、およびアプリケーションが待機中のロック用) を戻します。

表 23. AM_GET_LOCK_RPT プロシージャーから戻される一般アプリケーション情報

列名	データ・タイプ	説明
AGENT_ID	BIGINT	agent_id - アプリケーション・ハンドル (エージェント ID)
APPL_NAME	VARCHAR(256)	appl_name - アプリケーション名
AUTH_ID	VARCHAR(128)	auth_id - 許可 ID
APPL_ID	VARCHAR(128)	appl_id - アプリケーション ID

表 23. AM_GET_LOCK_RPT プロシージャーから戻される一般アプリケーション情報 (続き)

列名	データ・タイプ	説明
APPL_STATUS	VARCHAR(22)	appl_status - アプリケーション状況。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • BACKUP • COMMIT_ACT • COMP • CONNECTED • CONNECTPEND • CREATE_DB • DECOUPLED • DISCONNECTPEND • INTR • IOERROR_WAIT • LOAD • LOCKWAIT • QUIESCE_TABLESPACE • RECOMP • REMOTE_RQST • RESTART • RESTORE • ROLLBACK_ACT • ROLLBACK_TO_SAVEPOINT • TEND • THABRT • THCOMT • TPREP • UNLOAD • UOWEXEC • UOWWAIT • WAITFOR_REMOTE
COORD_PARTITION_NUM	SMALLINT	coord_node - コーディネーター・ノード
SEQUENCE_NO	VARCHAR(4)	sequence_no - シーケンス番号
CLIENT_PRDID	VARCHAR(128)	client_prdid - クライアント製品/バージョン ID
CLIENT_PID	BIGINT	client_pid - クライアント・プロセス ID

表 23. AM_GET_LOCK_RPT プロシージャーから戻される一般アプリケーション情報 (続き)

列名	データ・タイプ	説明
CLIENT_PLATFORM	VARCHAR(12)	client_platform - クライアント・オペレーティング・プラットフォーム。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。 • AIX® • AIX64 • AS400_DRDA • DOS • DYNIX • HP • HP64 • HPIA • HPIA64 • LINUX • LINUX390 • LINUXIA64 • LINUXPPC • LINUXPPC64 • LINUXX8664 • LINUXZ64 • MAC • MVS_DRDA • NT • NT64 • OS2 • OS390 • SCO • SGI • SNI • SUN • SUN64 • 不明 (UNKNOWN) • UNKNOWN_DRDA • VM_DRDA • VSE_DRDA • WINDOWS • WINDOWS95

表 23. AM_GET_LOCK_RPT プロシージャーから戻される一般アプリケーション情報 (続き)

列名	データ・タイプ	説明
CLIENT_PROTOCOL	VARCHAR(10)	client_protocol - クライアント通信プロトコル。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。 • CPIC • LOCAL • NETBIOS • NPIPE • TCPIP (DB2 Universal Database™、または DB2 UDB の場合) • TCPIP4 • TCPIP6
CLIENT_NNAME	VARCHAR(128)	client_nname モニター・エレメントは使用すべきではありません。返される値は無効な値です。
LOCKS_HELD	BIGINT	locks_held - ロック保持数
LOCK_WAIT_START_TIME	TIMESTAMP	lock_wait_start_time - ロック待機開始タイム・スタンプ
LOCK_WAIT_TIME	BIGINT	lock_wait_time - ロック待機中の時間
LOCK_WAITS	BIGINT	lock_waits - ロック待機数
LOCK_TIMEOUTS	BIGINT	lock_timeouts - ロック・タイムアウト数
LOCK_ESCALS	BIGINT	lock_escals - ロック・エスカレーション数
X_LOCK_ESCALS	BIGINT	x_lock_escals - 排他ロック・エスカレーション数
DEADLOCKS	BIGINT	deadlocks - デッドロック検出数

表 24. AM_GET_LOCK_RPT プロシージャーから戻されるロック保持情報

列名	データ・タイプ	説明
TBSP_NAME	VARCHAR(128)	tablespace_name - 表スペース名
TABSCHEMA	VARCHAR(128)	table_schema - 表スキーマ名
TABNAME	VARCHAR(128)	table_name - 表名

表 24. AM_GET_LOCK_RPT プロシージャーから戻されるロック保持情報 (続き)

列名	データ・タイプ	説明
LOCK_OBJECT_TYPE	VARCHAR(18)	lock_object_type - 待機中のロック対象タイプ。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • AUTORESIZE_LOCK • AUTOSTORAGE_LOCK • BLOCK_LOCK • EOT_LOCK • INPLACE_REORG_LOCK • INTERNAL_LOCK • INTERNALB_LOCK • INTERNALC_LOCK • INTERNALJ_LOCK • INTERNALL_LOCK • INTERNALO_LOCK • INTERNALQ_LOCK • INTERNALP_LOCK • INTERNALS_LOCK • INTERNALT_LOCK • INTERNALV_LOCK • KEYVALUE_LOCK • ROW_LOCK • SYSBOOT_LOCK • TABLE_LOCK • TABLE_PART_LOCK • TABLESPACE_LOCK • XML_PATH_LOCK

表 24. AM_GET_LOCK_RPT プロシージャーから戻されるロック保持情報 (続き)

列名	データ・タイプ	説明
LOCK_MODE	VARCHAR(10)	lock_mode - ロック・モード。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • IN • IS • IX • NON (ロックなしの場合) • NS • NW • S • SIX • U • X • Z
LOCK_STATUS	VARCHAR(10)	lock_status - ロック状況。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • CONV • GRNT
LOCK_ESCALATION	SMALLINT	lock_escalation - ロック・エスカレーション
LOCK_NAME	VARCHAR(32)	lock_name - ロック名
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

表 25. AM_GET_LOCK_RPT プロシージャーから戻されるロック待機情報

列名	データ・タイプ	説明
AGENT_ID_HOLDING_LK	BIGINT	agent_id_holding_lock - ロックを保持しているエージェント ID
APPL_ID_HOLDING_LK	VARCHAR(128)	appl_id_holding_lk - ロックを保持しているアプリケーション ID
LOCK_WAIT_START_TIME	TIMESTAMP	lock_wait_start_time - ロック待機開始タイム・スタンプ
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。
TBSP_NAME	VARCHAR(128)	tablespace_name - 表スペース名
TABSCHEMA	VARCHAR(128)	table_schema - 表スキーマ名
TABNAME	VARCHAR(128)	table_name - 表名

表 25. AM_GET_LOCK_RPT プロシージャーから戻されるロック待機情報 (続き)

列名	データ・タイプ	説明
LOCK_OBJECT_TYPE	VARCHAR(18)	lock_object_type - 待機中のロック対象タイプ。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • AUTORESIZE_LOCK • AUTOSTORAGE_LOCK • BLOCK_LOCK • EOT_LOCK • INPLACE_REORG_LOCK • INTERNAL_LOCK • INTERNALB_LOCK • INTERNALC_LOCK • INTERNALJ_LOCK • INTERNALL_LOCK • INTERNALO_LOCK • INTERNALQ_LOCK • INTERNALP_LOCK • INTERNALS_LOCK • INTERNALT_LOCK • INTERNALV_LOCK • KEYVALUE_LOCK • ROW_LOCK • SYSBOOT_LOCK • TABLE_LOCK • TABLE_PART_LOCK • TABLESPACE_LOCK • XML_PATH_LOCK

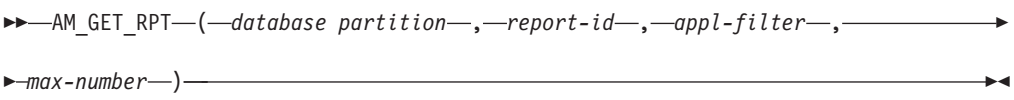
表 25. AM_GET_LOCK_RPT プロシージャーから戻されるロック待機情報 (続き)

列名	データ・タイプ	説明
LOCK_MODE	VARCHAR(10)	lock_mode - ロック・モード。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • IN • IS • IX • NON (ロックなしの場合) • NS • NW • S • SIX • U • X • Z
LOCK_MODE_REQUESTED	VARCHAR(10)	lock_mode_requested - 要求されているロック・モード。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • IN • IS • IX • NON (ロックなしの場合) • NS • NW • S • SIX • U • X • Z
LOCK_ESCALATION	SMALLINT	lock_escalation - ロック・エスカレーション

AM_GET_RPT - アクティビティー・モニター・データの検索

AM_GET_RPT プロシージャーは、レポートのアクティビティー・モニター・データを戻します。

構文



スキーマは SYSPROC です。

プロシージャー・パラメーター

database partition
データベース・パーティション番号を指定するタイプ INTEGER の入力引数。有効な値は、-2 (すべてのデータベース・パーティションの表示) と、既存のデータベース・パーティションのデータベース・パーティション番号です。

report-id
固有のレポート ID を指定する、タイプ INTEGER の入力引数。

appl-filter
アプリケーション・フィルターを指定する、タイプ CLOB(32K) の入力引数。アプリケーション・フィルターとは、3 つの列 (AGENT_ID、APPL_NAME、および AUTH_ID) の一部またはすべてが関係する検索条件のことです。ここで、AGENT_ID と AUTH_ID は整数で、APPL_NAME は文字ストリングです。引数が NULL または空ストリングである場合、フィルター操作は実行されません。

max-number
表示するアプリケーション、ステートメント、またはトランザクションの最大数を指定する、タイプ INTEGER の入力引数。引数が NULL の場合、すべてのアプリケーション、ステートメント、およびトランザクションが表示されます。

許可

- SYSMON 権限
- AM_GET_RPT プロシージャーに対する EXECUTE 特権。

例

```
CALL SYSPROC.AM_GET_RPT(-2, 18,
   CAST('AGENT_ID=29 AND AUTH_ID <> ''dbuser'' AND APPL_NAME LIKE ''db2%'''
       AS CLOB(32K)), 100)
```

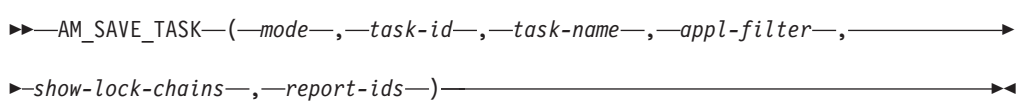
使用上の注意

戻される結果セットは、各レポート ID ごとに異なります。このプロシージャーは、アクティビティー・モニター・グラフィック・ツールをサポートするためのプロシージャーです。解析できるレポートを作成するには、このプロシージャーではなく、スナップショット管理 SQL ルーチンおよびビューを使用する必要があります。

AM_SAVE_TASK - モニター・タスクの作成または変更

AM_SAVE_TASK プロシージャーは、モニター・タスクを作成または変更します。

構文



スキーマは SYSPROC です。

プロシージャー・パラメーター

mode
新規モニター・タスクを作成する ('C') か既存のモニター・タスクを変更する ('M') かを指定する、タイプ CHAR(1) の入力引数。

task-id
固有のモニター・タスク ID を指定する、タイプ INTEGER の入力引数。 *mode* が 'C' である場合、*task-id* の指定された入力は無視されます。新規モニター・タスクの ID は、プロシージャーによって生成されて出力に戻されます。 *mode* が 'M' である場合、変更されるモニター・タスクの ID が指定されます。

task-name
モニター・タスクの名前または簡略説明を指定する、タイプ VARCHAR(128) の入力引数。

appl-filter
アプリケーション・フィルターを指定する、タイプ CLOB(32K) の入力引数。アプリケーション・フィルターとは、3 つの列 (AGENT_ID、APPL_NAME、および AUTH_ID) の一部またはすべてが関係する検索条件のことです。ここで、AGENT_ID と AUTH_ID は整数で、APPL_NAME は文字ストリングです。引数が NULL または空ストリングである場合、フィルター操作は実行されません。

show-lock-chains
ロック・チェーンを表示するかどうかを指定する、タイプ CHAR(1) の入力引数。有効な値は 'Y' および 'N' です。引数が NULL の場合、ロック・チェーンは表示されません。

report-ids
1 つ以上のレポート ID をコンマで区切って指定する、タイプ VARCHAR(3893) の入力引数。

許可

AM_SAVE_TASK プロシージャーに対する EXECUTE 特権。

例

例:

```
CALL SYSPROC.AM_SAVE_TASK('M',11,'Task ABC',CAST (NULL AS CLOB(32K)),
   'N','1,2,4,8,9,12')
```

第 4 章 ADMIN_CMD プロシージャーと関連したルーチン

ADMIN_CMD - 管理コマンドの実行

ADMIN_CMD プロシージャーは、SQL CALL ステートメントを使用して管理コマンドを実行するアプリケーションで使用されます。

構文

スキーマは SYSPROC です。

プロシージャー・パラメーター

command-string
実行する単一のコマンドを指定する、タイプ CLOB (2M) の入力引数。

許可

ADMIN_CMD プロシージャーに対する EXECUTE 特権

このプロシージャーは現在、以下の DB2 コマンド行プロセッサー (CLP) コマンドをサポートしています。

- ADD CONTACT
- ADD CONTACTGROUP
- AUTOCONFIGURE
- BACKUP - オンラインのみ
- DESCRIBE
- DROP CONTACT
- DROP CONTACTGROUP
- EXPORT
- FORCE APPLICATION
- IMPORT
- INITIALIZE TAPE
- LOAD
- PRUNE HISTORY/LOGFILE
- QUIESCE DATABASE
- QUIESCE TABLESPACES FOR TABLE
- REDISTRIBUTE
- REORG INDEXES/TABLE
- RESET ALERT CONFIGURATION

© Copyright IBM Corp. 2006, 2010

- RESET DATABASE CONFIGURATION
- RESET DATABASE MANAGER CONFIGURATION
- REWIND TAPE
- RUNSTATS
- SET TAPE POSITION
- UNQUIESCE DATABASE
- UPDATE ALERT CONFIGURATION
- UPDATE CONTACT
- UPDATE CONTACTGROUP
- UPDATE DATABASE CONFIGURATION
- UPDATE DATABASE MANAGER CONFIGURATION
- UPDATE HEALTH NOTIFICATION CONTACT LIST
- UPDATE HISTORY

注: コマンドによっては、ADMIN_CMD プロシージャーを介して実行するとき、サポートされる構文が若干異なることがあります。

このプロシージャーはまた、CLP ではサポートされていない次のようなコマンドをサポートします。

- GET STMM TUNING DBPARTITIONNUM
- UPDATE STMM TUNING DBPARTITIONNUM

使用上の注意

コマンドの実行に関する情報の検索:

- ADMIN_CMD プロシージャーはサーバー上で実行されるので、ユーティリティー・メッセージもサーバー上で作成されます。 MESSAGES ON SERVER オプション (詳細は、該当するコマンドの項を参照してください) は、メッセージ・ファイルをサーバー上で作成することを指示します。
- コマンドの実行状況は、CALL ステートメントからの結果である SQLCA で戻されます。
- 管理コマンドの実行が正常に完了した場合に、実行状況以外のものがコマンドから戻されるとしたら、そのような追加情報は、結果セット (最大 2 つまで) の形式で戻されます。例えば、EXPORT コマンドが正常に実行された場合、戻される結果セットには、エクスポートされた行の数についての情報が含まれます。一方、RUNSTATS コマンドが正常に実行された場合、結果セットは戻されません。結果セットの情報は、対応するコマンドとともに説明されています。
- 管理コマンドの実行が正常に完了しなかった場合、ADMIN_CMD プロシージャーから SQL20397W 警告メッセージが戻されます。それには、管理コマンドが失敗した理由に関する詳細を示した結果セットが伴います。 ADMIN_CMD プロシージャーを使用するアプリケーションはすべて、このプロシージャーから戻される SQLCODE を検査する必要があります。 SQLCODE が >= 0 の場合、管理コマンドの結果セットを検索する必要があります。 以下の表は、MESSAGES ON SERVER オプションを使用した場合としなかった場合に、どのような情報が戻されるかを示しています。

表 26. ADMIN_CMD プロシージャーから戻される SQLCODE および情報

管理コマンドの実行状況	MESSAGES ON SERVER オプションを指定した場合	MESSAGES ON SERVER オプションを指定しなかった場合
成功	戻された SQLCODE が >= 0 の場合: 追加情報 (結果セット) があれば、それが戻されます。	戻された SQLCODE が >= 0 の場合: 追加情報 (結果セット) があれば、それが戻されます。ただし、MSG_RETRIEVAL 列と MSG_REMOVAL 列は NULL です。
失敗	戻された SQLCODE が 20397 の場合: 追加情報 (結果セット) が戻されます。ただし、データが設定されているのは MSG_RETRIEVAL 列と MSG_REMOVAL 列だけです。	戻された SQLCODE が < 0 の場合: 追加情報 (結果セット) は戻されません。

- 結果セットを CLP からや、JDBC および DB2 CLI アプリケーションなどのアプリケーションから取り出すことはできますが、組み込み C アプリケーションから取り出すことはできません。
- 大文字と小文字を区別する名前および 2 バイト文字セット (DBCS) 名は、¥" MyTabLe ¥" のように円記号および二重引用符で囲む必要があります。

ADMIN_CMD を介して実行されるすべてのコマンドでは、データベースへの接続を確立したユーザー ID が認証で使用されます。

必要な追加の権限 (例えば、データベース・サーバー上のファイル・システムへのアクセスを必要とするコマンドに必要な追加の権限) については、該当するコマンドを解説している参照情報に記載されています。

ユーザー定義関数 (SQLSTATE 38001) またはトリガーからこのプロシージャーを呼び出すことはできません。

ADD CONTACT コマンド (ADMIN_CMD プロシージャーを使用)

システムでローカルに定義できる連絡先リストかまたはグローバル・リストで定義できる連絡先リストに、連絡先を追加します。連絡先とは、スケジューラーおよびヘルス・モニターなどのプロセスが、メッセージを送信する先のユーザーです。

Database Administration Server (DAS) の **contact_host** 構成パラメーターの設定により、リストがローカルかグローバルかが決まります。

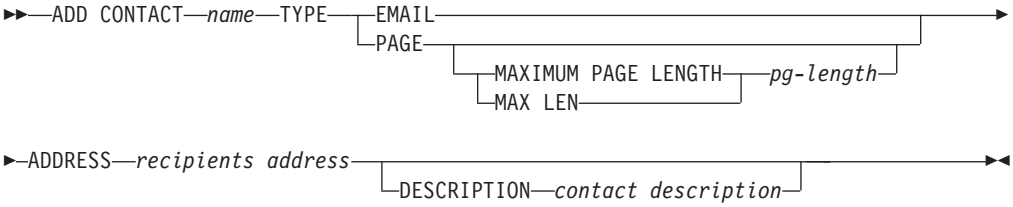
許可

なし

必要な接続

データベース。 DAS が実行中でなければなりません。

コマンド構文



コマンド・パラメーター

ADD CONTACT *name*
追加される連絡先の名前。デフォルトでは、DB2 Administration Server 構成パラメーター **contact_host** が別のシステムを指示していない限り、連絡先はローカル・システムに追加されます。

TYPE 連絡の方法。次のうちいずれかです。

EMAIL
この連絡先には、(**ADDRESS**) に E メールで通知します。

PAGE この連絡先には、**ADDRESS** にページを送信することで通知します。

MAXIMUM PAGE LENGTH *pg-length*
ページング・サービスにメッセージ長に関する制限がある場合、その制限を文字数で指定します。

通知システムは、SMTP プロトコルを使用して、2DB2 Administration Server 構成パラメーター **smtp_server** が指定するメール・サーバーに通知を送信します。 E メールを送信したり、ページャーを呼び出したりするのは、SMTP サーバーの役割です。

ADDRESS *recipients-address*
宛先の SMTP メールボックス・アドレス。例えば、joe@somewhere.org などです。 **smtp_server** DAS 構成パラメーターは、 SMTP サーバーの名前に設定することが必要です。

DESCRIPTION *contact description*
連絡先のテキスト記述。長さは、最大 128 文字です。

例

電子メール・アドレスが testuser@test.com であるユーザー testuser の連絡先を追加します。

```
CALL SYSPROC.ADMIN_CMD
   ('ADD CONTACT testuser TYPE EMAIL ADDRESS testuser@test.com')
```

使用上の注意

DAS が作成されていて実行中でなければなりません。

コマンドの実行状況は、CALL ステートメントからの結果である SQLCA で戻されます。

ADD CONTACTGROUP コマンド (ADMIN_CMD プロシージャーを使用)

ローカル・システムで定義されたグループのリストに、新しい連絡先グループを追加します。連絡先グループとは、スケジューラーおよびヘルス・モニターなどのモニター・プロセスが、メッセージを送信する先のユーザーおよびグループのリストです。

Database Administration Server (DAS) の **contact_host** 構成パラメーターの設定により、リストがローカルかグローバルかが決まります。

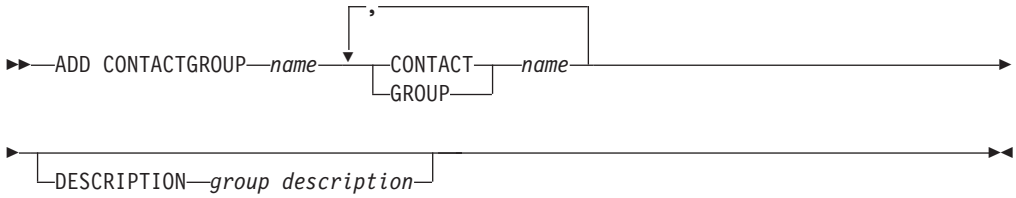
許可

なし

必要な接続

データベース。 DAS が実行中でなければなりません。

コマンド構文



コマンド・パラメーター

ADD CONTACTGROUP *name*
新しい連絡先グループの名前。システム上のグループの集合の中で固有なものでなければなりません。

CONTACT *name*
グループのメンバーである連絡先の名前。グループに追加された後、ADD CONTACT コマンドを使用して連絡先を定義できます。

GROUP *name*
このグループがメンバーである連絡先グループの名前。

DESCRIPTION *group description*
オプション。連絡先グループのテキスト記述。

例

2 つの連絡先 cname1 と cname2 を含む連絡先グループ gname1 を作成します。

```
CALL SYSPROC.ADMIN_CMD( 'add contactgroup gname1 contact cname1, contact cname2' )
```

使用上の注意

DAS が作成されていて実行中でなければなりません。

コマンドの実行状況は、CALL ステートメントからの結果である SQLCA で戻されます。

AUTOCONFIGURE コマンド (ADMIN_CMD プロシージャーを使用)

バッファー・プール・サイズ、データベース構成およびデータベース・マネージャーの構成パラメーターの初期値を計算します。オプションで、これらの推奨値を適用するように指定できます。

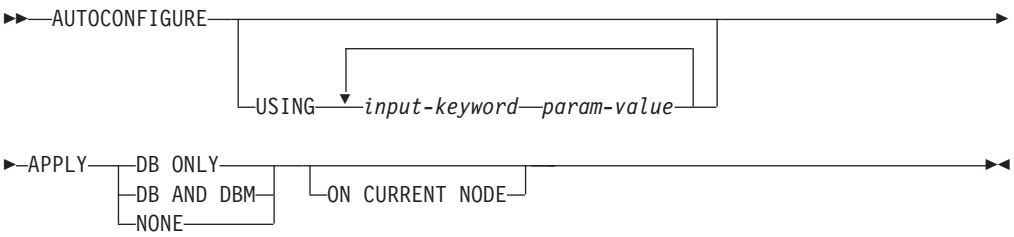
許可

SYSADM

必要な接続

データベース

コマンド構文



コマンド・パラメーター

USING *input-keyword param-value*

表 27. 有効な入力キーワードおよびパラメーター値

キーワード	有効値	デフォルト値	説明
mem_percent	1 から 100	25	データベースに割り当てられているインスタンス・メモリーのパーセンテージ。ただし、CREATE DATABASE コマンドによって構成アドバイザーが呼び出され、**mem_percent** の値を指定しない場合、インスタンスとシステムのメモリー使用量に基づいてパーセンテージが計算され、最大でインスタンス・メモリーの 25% になります。
workload_type	simple、mixed、complex	mixed	単純 (simple) ワークロードは入出力集中の傾向があり大部分がトランザクションであるのに対し、複雑 (complex) ワークロードは CPU 集中の傾向があり大部分が照会です。
num_stmts	1 から 1,000,000	10	作業単位ごとのステートメント数

表 27. 有効な入力キーワードおよびパラメーター値 (続き)

キーワード	有効値	デフォルト値	説明
tpm	1 から 200,000	60	1 分ごとのトランザクション
admin_priority	perfor-mance、recovery、both	both	より良いパフォーマンス (分あたりのより多いトランザクション数) またはより良いリカバリー時間のための最適化
is_populated	yes、no	yes	データベースにデータを移植するかどうか
num_local_apps	0 から 5,000	0	接続されたローカル・アプリケーションの数
num_remote_apps	0 から 5,000	10	接続されたリモート・アプリケーションの数
isolation	RR、 RS、CS、 UR	RR	このデータベースに接続するアプリケーションの最大分離レベル (反復可能読み取り (RR)、読み取り固定 (RS)、カーソル固定 (CS)、非コミット読み取り (UR))。これは、他の構成パラメーターの値を決定するためにのみ使用されます。アプリケーションを特定の分離レベルに制限するよう設定されるものはなく、デフォルト値を使用するのが安全です。
bp_resizeable	yes、no	yes	バッファー・プールのサイズが変更可能かどうか。

APPLY

DB ONLY
現行のデータベース・マネージャーの構成に基づいて、データベース構成およびバッファー・プール設定の推奨値を表示します。データベース構成およびバッファー・プール設定に対して推奨される変更を適用します。

DB AND DBM
データベース・マネージャー構成、データベース構成、およびバッファー・プール設定に対して推奨される変更を、表示および適用します。

NONE 推奨される変更を表示しますが、適用はしません。

ON CURRENT NODE
デフォルトでは、パーティション・データベース環境において、構成アドバイザーによりすべてのノード上のデータベース構成が更新されます。 **ON CURRENT NODE** オプションを指定して実行すると、アドバイザーにより推奨データベース構成が適用される対象となるのは、コーディネーター (接続) ノードだけになります。

バッファー・プールの変更事項は常にシステム・カタログに適用されます。したがってすべてのノードが影響を受けます。 **ON CURRENT NODE** オプションはバッファー・プールの推奨値に影響しません。

例

ADMIN_CMD ストアード・プロシージャーにより、データベースに対して AUTOCONFIGURE を呼び出します。

```
CALL SYSPROC.ADMIN_CMD( 'AUTOCONFIGURE APPLY NONE' )
```

以下は、コマンドによって戻される結果セットの例です。

```
LEVEL     NAME                 VALUE       RECOMMENDED_VALUE  DATATYPE
-----...- -----------------...- -------...- ------------------ --------...-
DBM       ASLHEAPSZ            15          15                 BIGINT
DBM       FCM_NUM_BUFFERS      512         512                BIGINT
...
DB        APP_CTL_HEAP_SZ      128         144                INTEGER
DB        APPGROUP_MEM_SZ      20000       14559              BIGINT
...
BP        IBMDEFAULTBP         1000        164182             BIGINT
```

使用上の注意

- このコマンドは、現在接続されているデータベースのための推奨構成を作成し、このデータベースがインスタンス上で唯一のアクティブ・データベースであると仮定します。セルフチューニング・メモリー・マネージャーを使用可能にしておらず、インスタンス上に複数のアクティブ・データベースがある場合、データベース・メモリー分散を反映する **mem_percent** 値を指定してください。例えば、インスタンス・メモリーの 80% を使用しリソースを平等に共有する、2 つのアクティブ・データベースがインスタンスにある場合、40% (80% を 2 データベースで割る) を **mem_percent** 値に指定します。
- 同じコンピューター上に複数のインスタンスがあり、セルフチューニング・メモリー・マネージャーが使用可能になっていない場合は、各インスタンスで **instance_memory** に固定値を設定するか、データベース・メモリーの配分を反映する **mem_percent** 値を指定してください。例えば、すべてのアクティブ・データベースがコンピューター・メモリーの 80% を使用し、それぞれ 1 つのデータベースを持つインスタンスが 4 つある場合は、20% (80% を 4 データベースで割る) を **mem_percent** 値に指定します。
- AUTOCONFIGURE コマンドによって構成アドバイザーを明示的に呼び出す場合、**DB2_ENABLE_AUTOCONFIG_DEFAULT** レジストリー変数の設定値は無視されます。
- データベースに対して AUTOCONFIGURE コマンドを実行すると、セルフチューニング・メモリー・マネージャーを有効にすることが推奨されます。ただし、**sheapthres** がゼロではないインスタンス内で、あるデータベースに対して AUTOCONFIGURE コマンドを実行した場合、ソート・メモリー・チューニング (**sortheap**) は自動的には有効になりません。ソート・メモリー・チューニング (**sortheap**) を有効にするには、UPDATE DATABASE MANAGER CONFIGURATION コマンドを使用して、**sheapthres** をゼロに設定する必要があります。 **sheapthres** の値を変更すると、これまでの既存データベース内のソート・メモリーの使用に影響を与える可能性があることに注意してください。
- コマンドの実行状況は、CALL ステートメントからの結果である SQLCA で戻されます。
- AUTOCONFIGURE 実行のために ADMIN_CMD プロシージャーの中で実行される SQL は、Query Patroller によってモニターされます。

- AUTOCONFIGURE コマンドの実行終了時には COMMIT ステートメントが発行されます。タイプ 2 接続では、その結果として ADMIN_CMD プロシージャーから理由コード 2 の SQL30090N が戻されます。

結果セット情報

コマンドの実行状況は、CALL ステートメントからの結果である SQLCA で戻されます。実行が成功したなら、以下の結果セットによりコマンドから追加情報が戻されます。

表 28. AUTOCONFIGURE コマンドから戻される結果セット

列名	データ・タイプ	説明
LEVEL	VARCHAR(3)	パラメーターのレベル。以下のいずれか 1 つ。 • BP (バッファー・プール・レベル) • DBM (データベース・マネージャー・レベル) • DB (データベース・レベル)
NAME	VARCHAR(128)	• LEVEL が DB または DBM の場合、これには構成パラメーター・キーワードが含まれています。 • LEVEL が BP の場合、この値にはバッファー・プール名が含まれています。
VALUE	VARCHAR(256)	• LEVEL が DB または DBM で、推奨値が適用された場合、この列の内容は、推奨値適用前に NAME 列の中で示されていた構成パラメーターの値です (つまり古い値が含まれています)。変更が適用されなかった場合、この列の内容は、示されている構成パラメーターについて現在ディスク上にある (据え置き) 値です。 • LEVEL が BP で、推奨値が適用された場合、この列の内容は、推奨値適用前に NAME 列の中で示されていたバッファー・プールのサイズ (ページ数) です (つまり古いサイズが含まれています)。変更が適用されなかった場合、この列の内容は、示されているバッファー・プールの現在のサイズ (ページ数) です。

表 28. AUTOCONFIGURE コマンドから戻される結果セット (続き)

列名	データ・タイプ	説明
RECOMMENDED_VALUE	VARCHAR(256)	• LEVEL が DB または DBM の場合、この列の内容は、パラメーター列で示されている構成パラメーターの推奨値 (または適用された値) です。 • タイプが BP の場合、この列の内容は、パラメーター列で示されているバッファー・プールの推奨サイズ (または適用されたサイズ) です (ページ数)。
DATATYPE	VARCHAR(128)	パラメーターのデータ・タイプ。

BACKUP DATABASE コマンド (ADMIN_CMD プロシージャーを使用)

データベースまたは表スペースのバックアップ・コピーを作成します。

異なるさまざまなオペレーティング・システムおよびハードウェア・プラットフォームの間で DB2 データベース・システムによってサポートされるバックアップ操作の詳細は、『異なるオペレーティング・システムおよびハードウェア・プラットフォーム間のバックアップおよびリストア操作』を参照してください。

有効範囲

パーティション・データベース環境で、データベース・パーティションを指定しない場合、このコマンドはコマンドが実行されたデータベース・パーティションにのみ作用します。

パーティション・バックアップを実行するためのオプションが指定された場合、コマンドを呼び出すことができるのは、カタログ・ノードに対してだけです。すべてのデータベース・パーティション・サーバーをバックアップするためのオプションが指定されているなら、それは db2nodes.cfg ファイルの中にリストされているすべてのデータベース・パーティション・サーバーに影響を与えます。そうでない場合は、コマンドで指定されたデータベース・パーティション・サーバーに作用します。

許可

以下のいずれか。

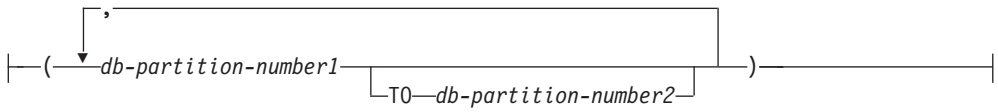
- SYSADM
- SYSCTRL
- SYSMAINT

必要な接続

データベース。既存のデータベース接続は、バックアップ操作の完了後はそのまま残ります。

コマンド構文

Partition number(s):



Open sessions:

Options:

コマンド・パラメーター

DATABASE | DB *database-alias*
バックアップを取るデータベースの別名を指定します。 別名は、サーバーで定義されたローカル・データベースでなければならず、また、ユーザーの現在の接続先のデータベース名でなければなりません。データベース別名がユーザーの接続先のものでない場合は、SQL20322N エラーが戻されます。

ON データベース・パーティションのセットに対してデータベースをバックアップします。この節は、カタログ・パーティションでのみ指定します。

DBPARTITIONNUM *db-partition-number1*
データベース・パーティション・リスト内のデータベース・パーティション番号を指定します。

DBPARTITIONNUMS *db-partition-number1* **TO** *db-partition-number2*
データベース・パーティション番号の範囲を指定します。 *db-partition-number1* から *db-partition-number2* までの間のすべてのパーティションがデータベース・パーティション・リストに含められます。

ALL DBPARTITIONNUMS
`db2nodes.cfg` ファイルで指定されたすべてのパーティションでデータベースをバックアップすることを指定します。

EXCEPT
`db2nodes.cfg` ファイルで指定されたパーティションのうち、データベース・パーティション・リストで指定されたパーティションを除くすべてのパーティションでデータベースをバックアップすることを指定します。

DBPARTITIONNUM *db-partition-number1*
データベース・パーティション・リスト内のデータベース・パーティション番号を指定します。

DBPARTITIONNUMS *db-partition-number1* **TO** *db-partition-number2*
データベース・パーティション番号の範囲を指定します。 *db-partition-number1* から *db-partition-number2* までの間のすべてのパーティションがデータベース・パーティション・リストに含められます。

TABLESPACE *tablespace-name*
バックアップを取る表スペースを指定するときに使用する名前のリスト。

ONLINE
オンライン・バックアップを指定します。サポートされているモードは、デフォルトのこれのみです。 **ONLINE** 節を指定する必要はありません。

INCREMENTAL
累積 (増分) バックアップ・イメージを指定します。増分バックアップ・イメージとは、正常に実行されたフルバックアップ操作のうち最新のものが実行されて以来変更された、すべてのデータベース・データのコピーです。

DELTA

非累積 (差分) バックアップ・イメージを指定します。差分バックアップ・イメージとは、正常に実行された任意のタイプのバックアップ操作のうち最新のものが実行されて以来変更された、すべてのデータベース・データのコピーです。

USE

TSM バックアップに Tivoli® Storage Manager (TSM) 出力を使用することを指定します。

XBSA XBSA インターフェースを使用するように指定します。バックアップ・サービス API (XBSA) は、バックアップやアーカイブの目的でデータ・ストレージ管理を必要とするアプリケーションまたは機能のための、オープン・アプリケーション・プログラミング・インターフェースです。

SNAPSHOT

スナップショット・バックアップを取ることを指定します。

SNAPSHOT パラメーターを以下のいずれかのパラメーターと一緒に使用することはできません。

- **TABLESPACE**
- **INCREMENTAL**
- **WITH** *num-buffers* **BUFFERS**
- **BUFFER**
- **PARALLELISM**
- **COMPRESS**
- **UTIL_IMPACT_PRIORITY**
- **SESSIONS**

スナップショット・バックアップのデフォルトの動作は、データベースを構成するすべてのパスの FULL DATABASE OFFLINE バックアップです。つまり、すべてのコンテナー、ローカル・ボリューム・ディレクトリー、データベース・パス (**DBPATH**)、1 次ログとミラー・ログのパスが含まれます (**INCLUDE LOGS** は、**EXCLUDE LOGS** を明示的に記述しない限り、すべてのスナップショット・バックアップのデフォルトです)。

LIBRARY *library-name*

IBM® Data Server には、以下のストレージ・ハードウェアのための DB2 ACS API ドライバーが組み込まれています。

- IBM TotalStorage® SAN ボリューム・コントローラー
- IBM Enterprise Storage Server® Model 800
- IBM System Storage® DS6000™
- IBM System Storage DS8000®
- IBM System Storage N Series
- NetApp V-series

- NetApp FAS

他のストレージ・ハードウェアを使用していて、そのストレージ・ハードウェア用の DB2 ACS API ドライバーがある場合、**LIBRARY** パラメーターを使用してその DB2 ACS API ドライバーを指定できます。

LIBRARY パラメーターの値は、完全修飾ライブラリー・ファイル名です。

OPTIONS

"*options-string*"
バックアップ操作で使用するオプションを指定します。ストリングは、二重引用符なしで、入力されたとおりに渡されます。

@ *file-name*
バックアップ操作で使用するオプションが、DB2 サーバー上のファイルに含まれていることを指定します。このストリングは、ベンダー・サポートのライブラリーに渡されます。ファイル名は完全修飾ファイル名でなければなりません。

VENDOROPT データベース構成パラメーターを使用して、スナップショット・バックアップ操作のベンダー固有オプションを指定することはできません。これには、バックアップ・ユーティリティーの **OPTIONS** パラメーターを使用する必要があります。

OPEN *num-sessions* **SESSIONS**
DB2 と TSM または他のバックアップ・ベンダー製品との間で作成される入出力セッションの数。このパラメーターは、テープ、ディスク、または他のローカル装置にバックアップする場合には効果はありません。

TO *dir* | *dev*
ディレクトリーまたはテープ装置名のリストです。ディレクトリーが常駐する絶対パスを指定しなければなりません。このターゲット・ディレクトリーまたは装置は、データベース・サーバー上に存在している必要があります。

パーティション・データベースでは、すべてのデータベース・パーティションにターゲット・ディレクトリーまたは装置が存在している必要があり、オプションで共用パスにすることができます。ディレクトリー名や装置名は、データベース・パーティション式を使用して指定できます。データベース・パーティション式について詳しくは、『『自動ストレージ・データベース』』 を参照してください。

このパラメーターは、バックアップ・イメージが複数の宛先ディレクトリーや装置にわたる場合に、それらを指定するために繰り返すことができます。宛先が複数指定されている場合 (例えば、宛先 1、宛先 2、および宛先 3)、宛先 1 が最初にオープンされます。メディア・ヘッダーおよび特殊ファイル (構成ファイル、表スペース表、および履歴ファイルを含む) は、宛先 1 にあります。他の残りの宛先は、オープンされており、これらはバックアップ操作のときに並列で使用されます。 Windows® オペレーティング・システムの場合、汎用テープ装置はサポートされていないので、テープ装置のタイプごとに固有のデバイス・ドライバーが必要です。

テープ装置またはフロッピー・ディスクを使用すると、プロンプトおよびユーザー対話を必要とする可能性があります。その場合は、結果としてエラーが戻されます。

テープ・システムでバックアップ・イメージを固有に参照する機能をサポートしていない場合は、同じテープに同じデータベースの複数のバックアップ・コピーは作成しないことをお勧めします。

LOAD *library-name*
使用するバックアップおよびリストア I/O ベンダー関数を含む共有ライブラリー (Windows オペレーティング・システムでは DLL) の名前。絶対パスで指定することができます。絶対パスを指定していない場合、デフォルトはユーザー出口プログラムが常駐しているパスになります。

WITH *num-buffers* **BUFFERS**
使用するバッファーの数です。値を明示的に指定しない場合、DB2 はこのパラメーターの最適値を自動的に選択します。ただし、バックアップを複数の場所に作成する場合は、パフォーマンスを向上させるために多数のバッファーを使用することができます。

BUFFER *buffer-size*
4 KB ページごとの単位で表した、バックアップ・イメージを作成する際に使用するバッファーのサイズ。値を明示的に指定しない場合、DB2 はこのパラメーターの最適値を自動的に選択します。このパラメーターの最小値は 8 ページです。

さまざまなブロック・サイズのテープを使用する場合は、磁気テープ装置がサポートする範囲内にバッファー・サイズを削減してください。この範囲内でないと、バックアップ操作は正常に実行されることもありますが、作成されたイメージはリカバリー不能になることがあります。

Linux® のほとんどのバージョンでは、SCSI テープ装置でバックアップ操作を行うときに、DB2 のデフォルトのバッファー・サイズを使用すると、エラー SQL2025N、理由コード 75 が表示されます。Linux 内部 SCSI バッファーがオーバーフローするのを防ぐには、以下の公式を使用してください。

```
bufferpages <= ST_MAX_BUFFERS * ST_BUFFER_BLOCKS / 4
```

bufferpages は **BUFFER** パラメーターと共に使用する値であり、`ST_MAX_BUFFERS` と `ST_BUFFER_BLOCKS` は `drivers/scsi` ディレクトリー中の Linux カーネルで定義されています。

PARALLELISM *n*
バックアップ・ユーティリティーによって同時に読み取り可能な表スペースの数を決定します。値を明示的に指定しない場合、DB2 はこのパラメーターの最適値を自動的に選択します。

DEDUP_DEVICE
データ非重複化をサポートするターゲット・ストレージ・デバイス用にバックアップ・イメージのフォーマットを最適化します。バージョン 9.7 フィックスパック 3 以降のフィックスパックで使用可能です。

UTIL_IMPACT_PRIORITY *priority*
バックアップを、指定した優先順位によりスロットル・モードで実行することを指定します。スロットル・モードでは、バックアップ操作によるパフォ

ーマンスの影響を調整できます。優先順位 (priority) は 1 から 100 までの範囲の任意の数であり、 1 が優先順位最低、100 が優先順位最高を意味します。優先順位の値なしで **UTIL_IMPACT_PRIORITY** キーワードが指定された場合は、デフォルトの優先順位 50 でバックアップが実行されます。**UTIL_IMPACT_PRIORITY** を指定しない場合、バックアップは非スロットル・モードで実行されます。バックアップをスロットル・モードで実行するためには、 **util_impact_lim** 構成パラメーターを設定することによって影響ポリシーが定義されていなければなりません。

COMPRESS
バックアップを圧縮することを指定します。

COMPRLIB *name*
圧縮を実行するために使用するライブラリーの名前を指定します (例えば、Windows の場合は `db2compr.dll`、Linux または UNIX システムの場合は `libdb2compr.so`)。この名前は、サーバー上の 1 個のファイルを参照する完全修飾パスでなければなりません。このパラメーターを指定しない場合、デフォルトの DB2 圧縮ライブラリーが使用されます。指定されたライブラリーをロードできない場合、バックアップは失敗します。

EXCLUDE
圧縮ライブラリーをバックアップ・イメージに格納しないことを指定します。

COMPROPTS *string*
バイナリー・データのうち、圧縮ライブラリーの初期設定ルーチンに渡すブロックを記述します。 DB2 はこのストリングをクライアントからサーバーに直接渡すため、バイト反転やコード・ページ変換の問題がある場合は圧縮ライブラリーで処理する必要があります。データ・ブロックの最初の文字が「@」である場合、DB2 は、データの残りの部分をサーバー上に存在するファイルの名前と解釈します。その場合 DB2 は、string の内容をそのファイルの内容で置き換え、そのようにして得られる新しい値を初期設定ルーチンに渡します。*string* の最大長は 1024 バイトです。

EXCLUDE LOGS
バックアップ・イメージにログ・ファイルをまったく含めないことを指定します。オフライン・バックアップ操作の実行の場合、このオプションが指定されていてもいなくても、ログは除外されます (ただし、スナップショット・バックアップは例外です)。

INCLUDE LOGS
ログ・ファイルのうち、特定の整合ポイント・イン・タイムまでこのイメージをリストアおよびロールフォワードするために必要な範囲をバックアップ・イメージに含めることを指定します。オフライン・バックアップの場合、このオプションは無効です。ただし、スナップショット・バックアップの場合は例外で、除外するように明示的に指示しない限りはこのオプションがデフォルトです。

WITHOUT PROMPTING

バックアップは、管理されることなく実行されるため、通常はユーザーの介入を必要とするアクションでエラー・メッセージが戻されるように指定されます。これはデフォルトです。

例

以下は、リカバリー可能データベース用の週次の増分バックアップ・ストラテジーのサンプルです。週 1 回の全データベース・バックアップ操作、1 日 1 回の非累積 (差分) バックアップ操作、および週 2 回の累積 (増分) バックアップ操作が含まれています。

```
(Sun) CALL SYSPROC.ADMIN_CMD('backup db sample online use tsm')
(Mon) CALL SYSPROC.ADMIN_CMD
      ('backup db sample online incremental delta use tsm')
(Tue) CALL SYSPROC.ADMIN_CMD
      ('backup db sample online incremental delta use tsm')
(Wed) CALL SYSPROC.ADMIN_CMD
      ('backup db sample online incremental use tsm')
(Thu) CALL SYSPROC.ADMIN_CMD
      ('backup db sample online incremental delta use tsm')
(Fri) CALL SYSPROC.ADMIN_CMD
      ('backup db sample online incremental delta use tsm')
(Sat) CALL SYSPROC.ADMIN_CMD
      ('backup db sample online incremental use tsm')
```

使用上の注意

バックアップ内のデータは、データベース・サーバーによって保護されるわけではありません。バックアップに LBAC で保護されたデータが含まれる場合は特に、バックアップを適切に保護しておく必要があります。

テープへのバックアップの場合、現在、可変ブロック・サイズの使用はサポートされていません。そのオプションを使用する必要がある場合は、リカバリーが正常に実行されるように十分にテストしたプロシージャーが使用できるようになっていることを確認し、また可変ブロック・サイズを指定して作成されたバックアップ・イメージを使用してください。

可変ブロック・サイズを使用する場合、使用している磁気テープ装置の最大限度以下のバックアップ・バッファー・サイズを指定する必要があります。パフォーマンスを最適化するには、使用している装置のブロック・サイズの最大限度と等しい値をバッファー・サイズとして使用しなければなりません。

スナップショット・バックアップは、ファイラー・システムまたはストレージ・システムで障害が発生した場合に備えて、通常のディスク・バックアップで補完する必要があります。

定期的にデータベースをバックアップしていくと、非常に大きなデータベース・バックアップ・イメージ、多くのデータベース・ログ、およびロード・コピー・イメージが累積する場合があり、これらすべてが大量のディスク・スペースを占めることがあります。これらのリカバリー・オブジェクトの管理方法については、『リカバリー・オブジェクトの管理』を参照してください。

プロキシー・ノードをサポートする TSM 環境では、**OPTIONS** パラメーターを使用してバックアップ操作を使用可能にすることができます。詳しくは、『Tivoli Storage Manager クライアントの構成』のトピックを参照してください。

結果セット情報

コマンドの実行状況は、CALL ステートメントからの結果である SQLCA で戻されます。実行が成功すると、コマンドは追加情報を戻します。バックアップ操作は、バックアップに含まれる各データベース・パーティションがそれぞれ 1 つの行を構成する、1 つの結果セットを戻します。

表 29. バックアップ操作の結果セット

列名	データ・タイプ	説明
BACKUP_TIME	VARCHAR(14)	バックアップ・イメージに名前を付けるときに使用されたタイム・スタンプ・ストリングに対応します。
DBPARTITIONNUM	SMALLINT	エージェントがバックアップ操作を実行したデータベース・パーティション番号。
SQLCODE	INTEGER	指定されたデータベース・パーティションにおけるバックアップ処理の最終的な SQLCODE 結果。
SQLERRMC	VARCHAR(70)	指定されたデータベース・パーティションにおけるバックアップ処理の最終的な SQLERRMC 結果。
SQLERRML	SMALLINT	指定されたデータベース・パーティションにおけるバックアップ処理の最終的な SQLERRML 結果。

非パーティション・データベースをバックアップする場合や、従来の単一パーティションの構文を使用してパーティション・データベースをバックアップする場合は、結果セットは 1 つの行で構成されます。DBPARTITIONNUM には、バックアップされるデータベース・パーティションの ID 番号が含まれます。

SQLCODE、SQLERRMC、および SQLERRML は、指定されたデータベース・パーティションでバックアップによって戻された SQLCA の、同等の名前を持つメンバーを参照します。

DESCRIBE コマンド (ADMIN_CMD プロシージャーを使用)

DESCRIBE コマンドは、表またはビューの列、索引、およびデータ・パーティションに関するメタデータを表示します。このコマンドは、SELECT、CALL、または XQuery ステートメントの出力に関するメタデータも表示できます。

DESCRIBE コマンドを使用して、以下の項目のいずれかに関する情報を表示します。

- SELECT、CALL、または XQuery ステートメントの出力
- 表またはビューの列
- 表またはビューの索引
- 表またはビューのデータ・パーティション

許可

必要な許可は、DESCRIBE コマンドを使用して表示する情報のタイプによって異なります。

- SYSTOOLSTMPSPACE 表スペースが存在する場合、以下の表に示されている権限の 1 つが必要になります。

情報を表示するオブジェクト	必要な特権または権限
SELECT ステートメントまたは XQuery ステートメントの出力	SELECT ステートメント内で参照されている表またはビューごとに、以下のいずれかの特権または権限: • SELECT 特権 • DATAACCESS 権限 • DBADM 権限 • SQLADM 権限 • EXPLAIN 権限
CALL ステートメントの出力	以下の特権または権限のいずれか。 • DATAACCESS 権限 • ストアード・プロシージャーでの EXECUTE 特権
表またはビューの列	SYSCAT.COLUMNS システム・カタログ表に対して、以下のいずれかの特権または権限: • SELECT 特権 • ACCESSCTRL 権限 • DATAACCESS 権限 • DBADM 権限 • SECADM 権限 • SQLADM 権限 **SHOW DETAIL** パラメーターを使用する場合、SYSCAT.DATAPARTITIONEXPRESSION システム・カタログ表に対しても、これらの特権または権限のいずれかが必要になります。 PUBLIC には宣言済み一時表に対するすべての特権が付与されているので、このコマンドを使用して、接続内に存在するすべての宣言済み一時表に関する情報を表示できます。

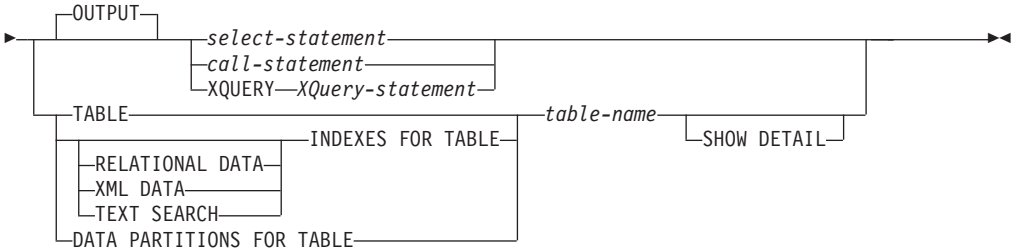
情報を表示するオブジェクト	必要な特権または権限
表またはビューの索引	SYSCAT.INDEXES システム・カタログ表に対して、以下のいずれかの特権または権限: • SELECT 特権 • ACCESSCTRL 権限 • DATAACCESS 権限 • DBADM 権限 • SECADM 権限 • SQLADM 権限 **SHOW DETAIL** パラメーターを使用する場合、GET_INDEX_COLNAMES() UDF に対する EXECUTE 特権も必要です。 PUBLIC には宣言済み一時表に対するすべての特権が付与されているので、このコマンドを使用して、接続内に存在するすべての宣言済み一時表に関する情報を表示できます。
表またはビューのデータ・パーティション	SYSCAT.DATAPARTITIONS システム・カタログ表に対して、以下のいずれかの特権または権限: • SELECT 特権 • ACCESSCTRL 権限 • DATAACCESS 権限 • DBADM 権限 • SECADM 権限 • SQLADM 権限 PUBLIC には宣言済み一時表に対するすべての特権が付与されているので、このコマンドを使用して、接続内に存在するすべての宣言済み一時表に関する情報を表示できます。

- SYSTOOLSTMPSPACE 表スペースが存在しない場合は、上記の権限のいずれかに加えて SYSADM または SYSCTRL 権限も必要になります。

必要な接続

データベース

コマンド構文



コマンド・パラメーター

OUTPUT

ステートメントの出力を記述するよう指示します。このキーワードはオプションです。

select-statement | *call-statement* | **XQUERY** *XQuery-statement*

情報が必要なステートメントを識別します。ステートメントは CLP によって自動的に準備されます。 XQuery ステートメントであることを示すには、ステートメントの先頭にキーワード **XQUERY** を入れます。 DESCRIBE OUTPUT ステートメントで暗黙的な隠し列についての情報が戻されるのは、記述対象の照会の最終的な結果表の SELECT リスト内にその列が明示的に指定されている場合のみです。

TABLE *table-name*

記述する表またはビューを指定します。 *schema.table-name* 形式の完全修飾名を使用しなければなりません。実際の表の代わりに表の別名を使用することはできません。 暗黙的に非表示になっている列に関する情報が戻されます。

DESCRIBE TABLE コマンドは、各列に関する以下の情報をリストします。

- 列名
- タイプ・スキーマ
- タイプ名
- 長さ
- スケール
- NULL 値 (yes/no)

INDEXES FOR TABLE *table-name*

索引を記述する必要がある表またはビューを指定します。 *schema.table-name* という形式の完全修飾名を使用するか、*table-name* のみを指定してデフォルトのスキーマが自動的に使用されるようにできます。実際の表の代わりに表の別名を使用することはできません。

DESCRIBE INDEXES FOR TABLE コマンドは、その表またはビューの各索引に関する以下の情報をリストします。

- 索引スキーマ
- 索引名
- ユニーク規則
- 列の数

- 索引タイプ

DESCRIBE INDEXES FOR TABLE コマンドが **SHOW DETAIL** オプションを指定して使用される場合、索引名が 18 バイトより長いと、索引名は切り捨てられます。索引タイプ・オプションが指定されていない場合、すべての索引タイプ (リレーショナル・データ索引、XML データに対する索引、および Text Search 索引) の情報がリストされます。出力には以下の追加情報が含まれます。

- リレーショナル・データ索引の索引 ID、XML パス索引、XML 領域索引、または XML データに対する索引
- XML データに対する索引のデータ・タイプ
- XML データに対する索引のハッシュ
- XML データに対する索引の最大 VARCHAR 長
- XML データに対する索引に指定された XML パターン
- テキスト検索索引のコード・ページ
- テキスト検索索引の言語
- テキスト検索索引に指定された形式
- テキスト検索索引の更新最小数
- テキスト検索索引の更新頻度
- テキスト検索索引のコレクション・ディレクトリー
- 列名

特定の索引タイプについてだけの情報をリストする索引タイプを指定します。複数の索引タイプの指定はサポートされていません。

RELATIONAL DATA

RELATIONAL DATA 索引タイプ・オプションが **SHOW DETAIL** オプションなしで指定されている場合は、以下の情報だけがリストされます。

- 索引スキーマ
- 索引名
- ユニーク規則
- 列の数

SHOW DETAIL を指定した場合、列名情報もリストされます。

XML DATA

XML DATA 索引タイプ・オプションが **SHOW DETAIL** オプションなしで指定されている場合は、以下の情報だけがリストされます。

- 索引スキーマ
- 索引名
- ユニーク規則
- 列の数
- 索引タイプ

SHOW DETAIL を指定した場合、以下の XML データに対する索引の情報もリストされます。

- 索引 ID
- データ・タイプ
- ハッシュ
- 最大 varchar 長
- XML パターン
- 列名

TEXT SEARCH

TEXT SEARCH 索引タイプ・オプションが **SHOW DETAIL** オプションなしで指定されている場合は、以下の情報だけがリストされます。

- 索引スキーマ
- 索引名

SHOW DETAIL を指定した場合、以下のテキスト検索索引情報もリストされます。

- 列名
- コード・ページ
- 言語
- フォーマット
- 更新最小数
- 更新頻度
- コレクション・ディレクトリー

TEXT SEARCH オプションが指定され、テキスト検索オプションがインストールされていないか正しく構成されていない場合、エラー (SQLSTATE 42724) が戻されます。

列にリストされる情報については、DB2 Text Search の情報を参照してください。

DATA PARTITIONS FOR TABLE *table-name*

データ・パーティションを記述する必要がある表またはビューを指定します。表に含まれるデータ・パーティションごとに表示される情報には、パーティション ID とパーティション・インターバルが含まれます。結果は、パーティション ID の順になっています。 *schema.table-name* 形式の完全修飾名を使用しなければなりません。実際の表の代わりに表の別名を使用することはできません。 *schema* はユーザー名で、その下に表またはビューが作成されます。

DESCRIBE DATA PARTITIONS FOR TABLE コマンドの場合、出力に以下の追加情報を内容とする 2 番目の表を含めることを指定します。

- データ・パーティション順序 ID
- SQL でのデータ・パーティション式

SHOW DETAIL

DESCRIBE TABLE コマンドの場合、表のデータ・パーティション式 (表がデータ・パーティション表でなければ 0 行が戻されるかもしれない) を含む 2 番目の結果セットと共に、以下の追加情報を出力に含めることを指定します。

- CHARACTER、VARCHAR または LONG VARCHAR 列のいずれかが FOR BIT DATA として定義されたかどうか
- 列番号
- 分散キー・シーケンス
- コード・ページ
- デフォルト
- 表パーティションのタイプ (範囲によってパーティション化されている表の場合、元の出力の下にこの出力が表示される)
- パーティション・キー列 (範囲によってパーティション化されている表の場合、元の出力の下にこの出力が表示される)
- 索引に使用される表スペースの ID

例

SELECT ステートメントの出力の記述

次に示すのは、SELECT ステートメントを記述する方法の一例です。

```
CALL SYSPROC.ADMIN_CMD('describe select * from emp_photo')
```

以下はこの SELECT ステートメントの出力例です。

```
Result set 1
--------------

SQLTYPE_ID SQLTYPE      SQLLENGTH SQLSCALE SQLNAME_DATA    ...
---------- ---------..- --------- -------- ------------..- ...
       452 CHARACTER            6        0 EMPNO           ...
       448 VARCHAR             10        0 PHOTO_FORMAT    ...
       405 BLOB            102400        0 PICTURE         ...

  3 record(s) selected.

  Return Status = 0
```

この SELECT ステートメントの出力 (続き)。

```
... SQLNAME_LENGTH SQLDATATYPENAME_DATA    SQLDATATYPENAME_LENGTH
... -------------- ---------------------..- ----------------------
...              5 SYSIBM  .CHARACTER                           18
...             12 SYSIBM  .VARCHAR                             16
...              7 SYSIBM  .BLOB                                13
```

表の記述

非パーティション表の記述。

```
CALL SYSPROC.ADMIN_CMD('describe table org show detail')
```

以下はこの CALL ステートメントの出力例です。

```
Result set 1
--------------
COLNAME     TYPESCHEMA     TYPENAME     FOR_BINARY_DATA     ...
```

```
-------...- ----------...- --------...- ---------------...- ...
DEPTNUMB    SYSIBM         SMALLINT     N                   ...
DEPTNAME    SYSIBM         VARCHAR      N                   ...
MANAGER     SYSIBM         SMALLINT     N                   ...
DIVISION    SYSIBM         VARCHAR      N                   ...
LOCATION    SYSIBM         VARCHAR      N                   ...

  5 record(s) selected.
```

この CALL ステートメントの出力 (続き)。

```
... LENGTH SCALE NULLABLE COLNO PARTKEYSEQ CODEPAGE DEFAULT
... ------ ----- -------- ----- ---------- -------- -------
...      2     0 N            0          1        0 -
...     14     0 Y            1          0     1208 -
...      2     0 Y            2          0        0 -
...     10     0 Y            3          0     1208 -
...     13     0 Y            4          0     1208 -
```

この CALL ステートメントの出力 (続き)。

```
Result set 2
--------------
DATA_PARTITION_KEY_SEQ DATA_PARTITION_EXPRESSION
---------------------- -------------------------

  0 record(s) selected.

Return Status = 0
```

パーティション表の記述。

```
CALL SYSPROC.ADMIN_CMD('describe table part_table1 show detail')
```

以下はこの CALL ステートメントの出力例です。

```
Result set 1
--------------
COLNAME     TYPESCHEMA     TYPENAME FOR_BINARY_DATA ...
-------...- ----------...- -------- --------------- ...
COL1        SYSIBM         INTEGER  N               ...

  1 record(s) selected.
```

この CALL ステートメントの出力 (続き)。

```
... LENGTH SCALE NULLABLE COLNO PARTKEYSEQ CODEPAGE DEFAULT
... ------ ----- -------- ----- ---------- -------- -------
...      4     0 N            0          1        0       -
```

この CALL ステートメントの出力 (続き)。

```
Result set 2
--------------
DATA_PARTITION_KEY_SEQ DATA_PARTITION_EXPRESSION
---------------------- -------------------------
                     1 COL1

  1 record(s) selected
```

表索引の記述

次に示すのは、表索引を記述する方法の一例です。この呼び出しは表 USER1.DEPARTMENT を記述して、2 つのリレーショナル・データ索引、6 つの XML データ索引、2 つのテキスト検索索引、およびシステム索引をリストします。

```
CALL SYSPROC.ADMIN_CMD('describe indexes for table user1.department')
```

以下はこの CALL ステートメントの出力例です。

```
Result set 1
--------------

INDSCHEMA      INDNAME                 UNIQUE_RULE
----------...  ------------------...  ------------------
SYSIBM         SQL070531145253450      DUPLICATES_ALLOWED
SYSIBM         SQL070531145253620      UNIQUE_ENTRIES_ONLY
USER1          RELIDX1                 DUPLICATES_ALLOWED
USER1          RELIDX2                 DUPLICATES_ALLOWED
SYSIBM         SQL070531145253650      PRIMARY_INDEX
USER1          XMLIDX1                 DUPLICATES_ALLOWED
SYSIBM         SQL070531154625650      DUPLICATES_ALLOWED
USER1          XMLIDX2                 DUPLICATES_ALLOWED
SYSIBM         SQL070531154626000      DUPLICATES_ALLOWED
USER1          XMLIDX3                 DUPLICATES_ALLOWED
SYSIBM         SQL070531154626090      DUPLICATES_ALLOWED
USER1          XMLIDX4                 DUPLICATES_ALLOWED
SYSIBM         SQL070531154626190      DUPLICATES_ALLOWED
USER1          XMLIDX5                 DUPLICATES_ALLOWED
SYSIBM         SQL070531154626290      DUPLICATES_ALLOWED
USER1          XMLIDX6                 DUPLICATES_ALLOWED
SYSIBM         SQL070531154626400      DUPLICATES_ALLOWED
USER1          TXTIDX1                 -
USER1          TXTIDX2                 -

  19 record(s) selected.

  Return Status = 0
```

この CALL ステートメントの出力 (続き)。

```
...   COLCOUNT    INDEXTYPE
...   ---------   -------------------------
...           -   XML_DATA_REGIONS
...           1   XML_DATA_PATH
...           1   RELATIONAL_DATA
...           2   RELATIONAL_DATA
...           1   RELATIONAL_DATA
...           1   XML_DATA_VALUES_LOGICAL
...           1   XML_DATA_VALUES_PHYSICAL
...           1   XML_DATA_VALUES_LOGICAL
...           1   XML_DATA_VALUES_PHYSICAL
...           1   XML_DATA_VALUES_LOGICAL
...           1   XML_DATA_VALUES_PHYSICAL
...           1   XML_DATA_VALUES_LOGICAL
...           1   XML_DATA_VALUES_PHYSICAL
...           1   XML_DATA_VALUES_LOGICAL
...           1   XML_DATA_VALUES_PHYSICAL
...           1   XML_DATA_VALUES_LOGICAL
...           1   XML_DATA_VALUES_PHYSICAL
...           1   TEXT_SEARCH
...           1   TEXT_SEARCH
```

データ・パーティションの記述

次に示すのは、データ・パーティションを記述する方法の一例です。

```
CALL SYSPROC.ADMIN_CMD('describe data partitions for table part_table2')
```

以下はこの CALL ステートメントの出力例です。

```
Result set 1
--------------

DATA_PARTITION_ID LOW_KEY_INCLUSIVE LOW_KEY_VALUE ...
----------------- ----------------- ------------ ...
                0 Y                 1            ...
                1 Y                 10           ...
                2 Y                 20           ...

  3 record(s) selected.
```

この CALL ステートメントの出力 (続き)。

```
... HIGH_KEY_INCLUSIVE HIGH_KEY_VALUE
... ------------------ --------------
... N                  10
... N                  20
... N                  40
```

次に示すのは、SHOW DETAIL 節を指定してデータ・パーティションを記述する方法の一例です。

```
CALL SYSPROC.ADMIN_CMD('describe data partitions
    for table part_table2 show detail')
```

以下はこの CALL ステートメントの出力例です。

```
Result set 1
--------------

DATA_PARTITION_ID LOW_KEY_INCLUSIVE LOW_KEY_VALUE ...
----------------- ----------------- ------------ ...
                0 Y                 1            ...
                1 Y                 10           ...
                2 Y                 20           ...

  3 record(s) selected.

Return Status = 0
```

この CALL ステートメントの出力 (続き)。

```
... HIGH_KEY_INCLUSIVE HIGH_KEY_VALUE
... ------------------ --------------
... N                  10
... N                  20
... N                  40
```

この CALL ステートメントの出力 (続き)。

```
Result set 2
--------------

DATA_PARTITION_ID DATA_PARTITION_NAME TBSPID ...
----------------- ------------------- ------ ...
                0 PART0                    3 ...
                1 PART1                    3 ...
                2 PART2                    3 ...

  3 record(s) selected.

Return Status = 0
```

この CALL ステートメントの出力 (続き)。

```
... PARTITION_OBJECT_ID LONG_TBSPID ACCESSMODE  STATUS
... ------------------- ----------- ----------- ------
...                  15           3 FULL_ACCESS
...                  16           3 FULL_ACCESS
...                  17           3 FULL_ACCESS
```

使用上の注意

DESCRIBE コマンドが一時表を作成しようとして失敗した場合、SYSTOOLSTMPSPACE の作成が試みられ、その後、再び一時表の作成が、今回は SYSTOOLSTMPSPACE の中で試みられます。 SYSCTRL または SYSADM 権限は、SYSTOOLSTMPSPACE 表スペースを作成するために必要です。

結果セット情報

コマンドの実行状況は、CALL ステートメントからの結果である SQLCA で戻されます。実行が成功したなら、以下の結果セットによりコマンドから追加情報が戻されます。

- 表 30: DESCRIBE *select-statement*、DESCRIBE *call-statement*、および DESCRIBE XQUERY *XQuery-statement* の各コマンド
- 67 ページの表 31: DESCRIBE TABLE コマンドの結果セット 1
- 68 ページの表 32: DESCRIBE TABLE コマンドの結果セット 2
- 68 ページの表 33: DESCRIBE INDEXES FOR TABLE コマンド
- 70 ページの表 34: DESCRIBE DATA PARTITIONS FOR TABLE コマンドの結果セット 1
- 70 ページの表 35: DESCRIBE DATA PARTITIONS FOR TABLE コマンドの結果セット 2

表 30. DESCRIBE select-statement、DESCRIBE call-statement、および DESCRIBE XQUERY XQuery-statement の各コマンドによって戻される結果セット

列名	データ・タイプ	LOB のみ[1]	説明
SQLTYPE_ID	SMALLINT	いいえ	列のデータ・タイプ。 SQL 記述子域 (SQLDA) の SQLTYPE フィールドに示されているのと同じ。
SQLTYPE	VARCHAR (257)	いいえ	SQLTYPE_ID 値に対応するデータ・タイプ。
SQLLEN	INTEGER	いいえ	列の長さ属性。 SQLDA の SQLLEN フィールドに示されているのと同じ。
SQLSCALE	SMALLINT	いいえ	10 進 (DECIMAL) 値の場合は小数部分の桁数。それ以外のデータ・タイプの場合は 0。
SQLNAME_DATA	VARCHAR (128)	いいえ	列の名前。
SQLNAME_LENGTH	SMALLINT	いいえ	列名の長さ。
SQLDATA_TYPESCHEMA	VARCHAR (128)	はい	データ・タイプ・スキーマ名。
SQLDATA_TYPENAME	VARCHAR (128)	はい	データ・タイプ名。

注: [1]: Yes の場合、記述する LOB データがある場合にのみヌル以外の値が戻されることを示します。

表 31. DESCRIBE TABLE コマンドによって戻される結果セット 1

列名	データ・タイプ	詳細[2]	説明
COLNAME	VARCHAR (128)	いいえ	列名。
TYPESCHEMA	VARCHAR (128)	いいえ	列名が特殊ならスキーマ名が戻され、そうでない場合は 'SYSIBM' が戻されます。
TYPENAME	VARCHAR (128)	いいえ	列タイプの名前。
FOR_BINARY_DATA	CHAR (1)	はい	列が CHAR、VARCHAR、または LONG VARCHAR のいずれかのタイプであり、かつ FOR BIT DATA として定義されているなら 'Y' が戻され、そうでない場合は 'N' が戻されます。
LENGTH	INTEGER	いいえ	データの最大長。 DECIMAL (10 進) データの場合、これは精度を示します。特殊タイプの場合、0 が戻されます。
SCALE	SMALLINT	いいえ	DECIMAL (10 進) データの場合、これはスケールを示します。それ以外のすべてタイプの場合、0 が戻されます。
NULLABLE	CHAR (1)	いいえ	以下のいずれか • 列が NULL 可能の場合、'Y' • 列が NULL 可能でない場合、'N'
COLNO	SMALLINT	はい	列の順序。
PARTKEYSEQ	SMALLINT	はい	表のパーティション・キーの中での列の順序。列がパーティション・キーの一部でない場合は NULL または 0 が戻され、副表および階層表の場合は NULL。
CODEPAGE	SMALLINT	はい	列のコード・ページ。以下のうちいずれか。 • FOR BIT DATA として定義されていない列の場合、データベース・コード・ページの値。 • グラフィック列の場合、DBCS コード・ページの値。 • それ以外の場合は 0。
DEFAULT	VARCHAR (254)	はい	列のデータ・タイプに適した定数、特殊レジスター、または cast 関数で表された表の列のデフォルト値。NULL の場合もあります。

注: [2]: Yes の場合、SHOW DETAIL 節が使用された場合にのみヌル以外の値が戻されることを示します。

表 32. SHOW DETAIL 節が使用された場合に DESCRIBE TABLE コマンドによって戻される結果セット 2

列名	データ・タイプ	説明
DATA_PARTITION_KEY_SEQ	INTEGER	データ・パーティション・キー番号。例えば、最初のデータ・パーティション式の場合は 1、2 番目のデータ・パーティション式の場合は 2。
DATA_PARTITION_EXPRESSION	CLOB (32K)	SQL 構文によるこのデータ・パーティション・キーの式

表 33. DESCRIBE INDEXES FOR TABLE コマンドによって戻される結果セット

列名	データ・タイプ	詳細[3]	索引タイプ・オプション[4、5]	説明
INDSCHEMA	VARCHAR (128)	いいえ	RELATIONAL DATA XML DATA TEXT SEARCH	索引スキーマ名。
INDNAME	VARCHAR (128)	いいえ	RELATIONAL DATA XML DATA TEXT SEARCH	索引名。
UNIQUE_RULE	VARCHAR (30)	いいえ	RELATIONAL DATA XML DATA	以下の値のいずれか。 • DUPLICATES_ALLOWED • PRIMARY_INDEX • UNIQUE_ENTRIES_ONLY
INDEX _PARTITIONING	CHAR(1)	いいえ	N/A	索引のパーティション特性を示します。可能な値は以下のとおりです。 • N = 非パーティション索引 • P = パーティション索引 • ブランク = 索引はパーティション表にはない
COLCOUNT	SMALLINT	いいえ	RELATIONAL DATA XML DATA	キー内の列数と組み込み列 (存在する場合) の数の合計。
INDEX_TYPE	VARCHAR (30)	いいえ	RELATIONAL DATA XML DATA TEXT SEARCH	索引のタイプ: • RELATIONAL_DATA • TEXT_SEARCH • XML_DATA_REGIONS • XML_DATA_PATH • XML_DATA_VALUES_LOGICAL • XML_DATA_VALUES_PHYSICAL
INDEX_ID	SMALLINT	はい	RELATIONAL DATA XML DATA	リレーショナル・データ索引の索引 ID、XML パス索引、XML 領域索引、または XML データに対する索引

表 33. *DESCRIBE INDEXES FOR TABLE* コマンドによって戻される結果セット (続き)

列名	データ・タイプ	詳細[3]	索引タイプ・オプション [4、5]	説明
DATA_TYPE	VARCHAR (128)	はい	XML DATA	XML データに対する索引に指定された SQL データ・タイプ。以下の値のいずれか。 • VARCHAR • DOUBLE • DATE • TIMESTAMP
HASHED	CHAR (1)	はい	XML DATA	XML データに対する索引の値がハッシュされるかどうかを示します。 • 値がハッシュされる場合、'Y'。 • 値がハッシュされない場合、'N'。
LENGTH	SMALLINT	はい	XML DATA	XML データに対する索引の場合、VARCHAR (*integer*) の長さ。その他の場合 0。
PATTERN	CLOB (2M)	はい	XML DATA	XML データに対する索引に指定された XML パターン式。
CODEPAGE	INTEGER	はい	TEXT SEARCH	テキスト検索索引に指定された文書コード・ページ
LANGUAGE	VARCHAR (5)	はい	TEXT SEARCH	テキスト検索索引に指定された文書言語
FORMAT	VARCHAR (30)	はい	TEXT SEARCH	テキスト検索索引に指定された文書フォーマット
UPDATEMINIMUM	INTEGER	はい	TEXT SEARCH	インクリメンタル更新が実行される前のテキスト検索ログ表内の項目の最小数
UPDATEFREQUENCY	VARCHAR (300)	はい	TEXT SEARCH	テキスト索引に更新を適用するために指定されるトリガー基準
COLLECTION DIRECTORY	VARCHAR (512)	はい	TEXT SEARCH	テキスト検索索引ファイルに指定されたディレクトリー
COLNAMES	VARCHAR (2048)	はい	RELATIONAL DATA XML DATA TEXT SEARCH	列名のリスト。それぞれの前に + が付いている場合は昇順、- が付いている場合は降順。

注: [3]: Yes の場合、SHOW DETAIL 節が索引タイプ・オプションの指定なしで使用された場合にのみ、値が戻されることを示します。値は NULL にすることもできます。

注: [4]: DESCRIBE *index-type* INDEXES FOR TABLE を使用した場合に戻される値を示します。例えば、INDEX_ID 値は、TEXT SEARCH が *index-type* として指定された場合は戻されません。 INDEX_ID 値は、RELATIONAL DATA または XML DATA のいずれかが指定される場合に戻されます。

注: [5]: DESCRIBE *index-type* INDEXES FOR TABLE SHOW DETAIL を使用する場合、値は索引タイプがリストされるときにのみ戻されます。例えば、DATA_TYPE

値は、XML DATA が *index-type* として指定された場合に戻されます。
DATA_TYPE 値は、TEXT SEARCH または RELATIONAL DATA のいずれかが *index-type* として指定された場合には戻されません。

表 34. DESCRIBE DATA PARTITIONS FOR TABLE コマンドによって戻される結果セット 1

列名	データ・タイプ	詳細[2]	説明
DATA_PARTITION_ID	INTEGER	いいえ	データ・パーティション ID。
LOW_KEY_INCLUSIVE	CHAR (1)	いいえ	キー値の下限を含む場合は 'Y'、そうでない場合は 'N'。
LOW_KEY_VALUE	VARCHAR (512)	いいえ	このデータ・パーティションのキー値の下限。
HIGH_KEY_INCLUSIVE	CHAR (1)	いいえ	キー値の上限を含む場合は 'Y'、そうでない場合は 'N'。
HIGH_KEY_VALUE	VARCHAR (512)	いいえ	このデータ・パーティションのキー値の上限。

注: [2]: Yes の場合、SHOW DETAIL 節が使用された場合にのみヌル以外の値が戻されることを示します。

表 35. SHOW DETAIL 節が使用されている場合に DESCRIBE DATA PARTITIONS FOR TABLE コマンドによって戻される結果セット 2

列名	データ・タイプ	説明
DATA_PARTITION_ID	INTEGER	データ・パーティション ID。
DATA_PARTITION_NAME	VARCHAR (128)	データ・パーティション名。
TBSPID	INTEGER	このデータ・パーティションが格納されている表スペースの ID。
PARTITION_OBJECT_ID	INTEGER	このデータ・パーティションが格納されている DMS オブジェクトの ID。
LONG_TBSPID	INTEGER	LONG データが格納されている表スペースの ID。
INDEX_TBSPID	INTEGER	索引データが格納されている表スペースの ID。
ACCESSMODE	VARCHAR (20)	データ・パーティションのアクセシビリティーを定義。以下のいずれかが必要です。 • FULL_ACCESS • NO_ACCESS • NO_DATA_MOVEMENT • READ_ONLY

表 35. SHOW DETAIL 節が使用されている場合に DESCRIBE DATA PARTITIONS FOR TABLE コマンドによって戻される結果セット 2 (続き)

列名	データ・タイプ	説明
STATUS	VARCHAR(64)	データ・パーティションの状況。以下のいずれかが必要です。 • NEWLY_ATTACHED • NEWLY_DETACHED: MQT メンテナンスが必要。 • INDEX_CLEANUP_PENDING: 索引クリーンアップのためにのみ SYSDATAPARTITIONS 内のタプルが維持されている、デタッチされたデータ・パーティション。このタプルは、デタッチされたデータ・パーティションを参照する索引レコードがすべて削除された時点で除去されます。 それ以外の場合、この列はブランクです。

DROP CONTACT コマンド (ADMIN_CMD プロシージャーを使用)

ローカル・システムで定義された連絡先のリストから、連絡先を除去します。連絡先とは、スケジューラーおよびヘルス・モニターがメッセージを送信する先のユーザーです。 Database Administration Server (DAS) の **contact_host** 構成パラメーターの設定により、リストがローカルかグローバルかが決まります。

許可

なし

必要な接続

データベース。 DAS が実行中でなければなりません。

コマンド構文

コマンド・パラメーター

CONTACT *name*
ローカル・システムからドロップされる連絡先の名前。

例

サーバー・システム上の連絡先リストから testuser という連絡先をドロップします。

```
CALL SYSPROC.ADMIN_CMD( 'drop contact testuser' )
```

使用上の注意

DAS が作成されていて実行中でなければなりません。

コマンドの実行状況は、CALL ステートメントからの結果である SQLCA で戻されます。

DROP CONTACTGROUP コマンド (ADMIN_CMD プロシージャーを使用)

ローカル・システムで定義された連絡先のリストから、連絡先グループを除去します。連絡先グループには、スケジューラーおよびヘルス・モニターがメッセージを送信する先のユーザーのリストが入っています。 Database Administration Server (DAS) の **contact_host** 構成パラメーターの設定により、リストがローカルかグローバルかが決まります。

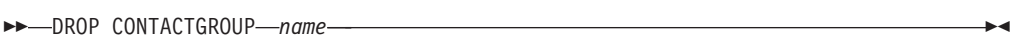
許可

なし

必要な接続

データベース。 DAS が実行中でなければなりません。

コマンド構文



コマンド・パラメーター

CONTACTGROUP *name*
ローカル・システムからドロップされる連絡先グループの名前。

例

連絡先グループ gname1 をドロップします。

```
CALL SYSPROC.ADMIN_CMD( 'drop contactgroup gname1' )
```

使用上の注意

DAS が作成されていて実行中でなければなりません。

コマンドの実行状況は、CALL ステートメントからの結果である SQLCA で戻されます。

EXPORT コマンド (ADMIN_CMD プロシージャーを使用)

データベースから、いくつかある外部ファイル形式のいずれかにデータをエクスポートします。ユーザーは、SQL SELECT ステートメントによって、または型付き表の階層情報によってエクスポートするデータを指定します。データはサーバーにのみエクスポートされます。

79 ページの『エクスポート・ユーティリティー用のファイル・タイプ修飾子』へのクイック・リンク。

許可

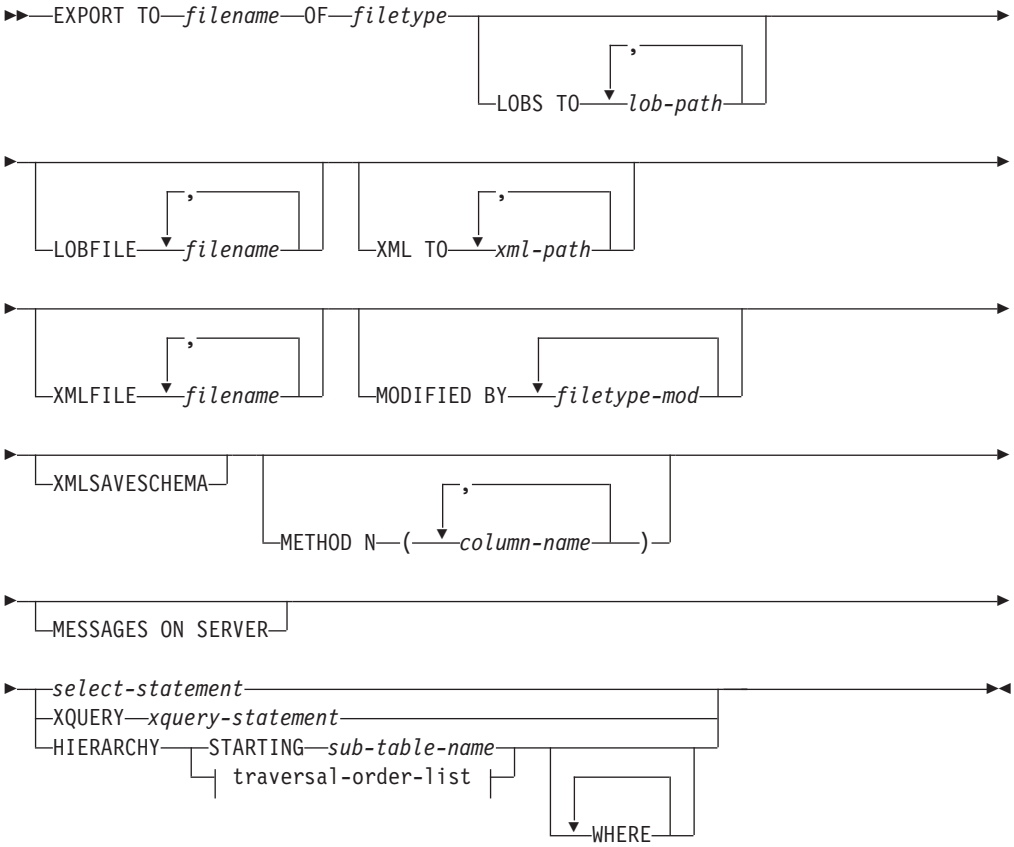
以下のいずれかです。

- DATAACCESS 権限
- 関係するそれぞれの表またはビューに対する CONTROL または SELECT 特権

必要な接続

データベース。 Linux、UNIX、または Windows クライアントから Linux、UNIX、または Windows データベース・サーバーへのユーティリティー・アクセスは、DB2 Connect™ ゲートウェイまたはループバック環境を経由してではなく、エンジンを使用した直接接続でなければなりません。

コマンド構文



traversal-order-list:

コマンド・パラメーター

HIERARCHY *traversal-order-list*

指定した走査順序を使用して副階層をエクスポートします。すべての副表は、PRE-ORDER 方式でリストされていなければなりません。最初の副表名が、SELECT ステートメントのターゲット表名として使用されます。

HIERARCHY STARTING *sub-table-name*

デフォルトの走査順序 (ASC、DEL、または WSF ファイルの OUTER 順序、または PC/IXF データ・ファイルに保管されている順序) を使用して、*sub-table-name* から始まる副階層をエクスポートします。

LOBFILE *filename*

LOB ファイルに 1 つ以上の基本ファイル名を指定します。最初の名前の名前空間がいっぱいになると、2 番目の名前が使用され、以下 3 番目、4 番目と続きます。これによって、LOBSINFILE 動作が暗黙的にアクティブ化されます。

エクスポート操作中に LOB ファイルを作成するときに、まずこのリストから現行パス (*lob-path* で指定されたパス) に現行のベース名を追加してから、最初に 3 桁のシーケンス番号、次に 3 文字の ID lob を追加して、ファイル名が構成されます。例えば、現行 LOB パスがディレクトリー /u/foo/lob/path/ で、現行 LOB ファイル名が bar の場合、 LOB ファイルは、 /u/foo/lob/path/bar.001.lob、 /u/foo/lob/path/bar.002.lob (以下 003、004 と続く) などのように作成されます。 LOB ファイル名の 3 桁のシーケンス番号で 999 が使用されてしまうとこの番号は 4 桁に増え、また、9999 が使用されてしまうと 4 桁が 5 桁に増えます。以下同様にシーケンス番号の桁数が増えていきます。

LOBS TO *lob-path*

LOB ファイルが保管される、ディレクトリーへの 1 つ以上のパスを指定します。それらのパスはサーバーのコーディネーター・パーティション上に存在するものでなければならず、また、完全修飾パスでなければなりません。LOB パスごとに少なくとも 1 つのファイルが存在し、各ファイルには少なくとも 1 つの LOB が入ります。指定できるパスの最大数は 999 です。これによって、LOBSINFILE 動作が暗黙的にアクティブ化されます。

MESSAGES ON SERVER

EXPORT コマンドによってサーバー上に作成されるメッセージ・ファイルを保存することを指定します。戻される結果セットには以下の 2 つの列が含まれます。 1 つは MSG_RETRIEVAL で、これはこの操作中に発生したすべての警告メッセージおよびエラー・メッセージを取り出すのに必要な SQL ステートメントです。もう 1 つは MSG_REMOVAL で、これはメッセージをクリーンアップするために必要な SQL ステートメントです。

この節が指定されていない場合は、ADMIN_CMD プロシージャーから呼び出し元に戻る時点でメッセージ・ファイルが削除されます。結果セット中の MSG_RETRIEVAL および MSG_REMOVAL 列の内容は NULL 値になります。

この節が指定されているかどうかに関係なく、fenced ユーザー ID に、**DB2_UTIL_MSGPATH** レジストリー変数の示すディレクトリーおよびデータのエクスポート先ディレクトリーの下にファイルを作成するための権限が付与されていることが必要です。

METHOD N *column-name*
出力ファイルで使用される 1 つ以上の列名を指定します。このパラメーターが指定されない場合、表の列名が使用されます。このパラメーターは WSF および IXF ファイルでのみ有効ですが、階層データをエクスポートするときは無効です。

MODIFIED BY *filetype-mod*
ファイル・タイプ修飾子オプションを指定します。 79 ページの『エクスポート・ユーティリティー用のファイル・タイプ修飾子』を参照してください。

OF *filetype*
次のような出力ファイルのデータ・フォーマットを指定します。

- DEL (区切り文字付き ASCII フォーマット)。さまざまなデータベース・マネージャーやファイル・マネージャー・プログラムで使用します。
- WSF (ワークシート・フォーマット)。以下のプログラムで使用します。
  - Lotus® 1-2-3®
  - Lotus Symphony™

  BIGINT または DECIMAL データをエクスポートする場合、タイプ DOUBLE の範囲内の値のみが正確にエクスポートされます。この範囲内にない値もエクスポートされますが、オペレーティング・システムによっては、これらの値を再びインポートまたはロードすると、データに間違いが生じる場合があります。

  注: WSF ファイル・フォーマットのサポートは推奨されておらず、今後のリリースで除去される可能性があります。サポートが除去される前に、WSF ファイルの代わりに、サポートされているファイル・フォーマットの使用を開始することを推奨します。
- IXF (統合交換フォーマット、PC バージョン) は、プロプラエタリー・バイナリー・フォーマットです。

select-statement
エクスポートされるデータを戻す SELECT または XQUERY ステートメントを指定します。このステートメントによってエラーが発生する場合、メッセージ・ファイル (または標準出力) にメッセージが書き込まれます。エラー・コードが SQL0012W、SQL0347W、SQL0360W、SQL0437W、または SQL1824W である場合、エクスポート操作は続行します。これ以外のエラー・コードの場合、操作は停止します。

TO *filename*
サーバーにおいてデータのエクスポート先となるファイルの名前を指定します。これは、完全修飾パスでなければならず、サーバー・コーディネーター・パーティション上に存在していなければなりません。

既に存在するファイルの名前を指定した場合、エクスポート・ユーティリティーはファイルの内容を上書きします。つまり情報は追加されません。

XMLFILE *filename*

XML ファイルのための 1 つ以上の基本ファイル名を指定します。最初の名前の名前空間がいっぱいになると、2 番目の名前が使用され、以下 3 番目、4 番目と続きます。

エクスポート操作中に XML ファイルを作成するときに、まずこのリストから現行パス (*xml-path* で指定されたパス) に現行のベース名を追加し、それに 3 桁のシーケンス番号を追加し、さらに 3 文字の ID xml を追加したファイル名が構成されます。例えば、現行 XML パスがディレクトリー /u/foo/xml/path/ で、現行 XML ファイル名が bar の場合、XML ファイルは、/u/foo/xml/path/bar.001.xml、/u/foo/xml/path/bar.002.xml などのように作成されます。

XML TO *xml-path*

XML ファイルが保管されるディレクトリーを指す 1 つ以上のパスを指定します。 XML パスごとに少なくとも 1 つのファイルが存在し、各ファイルには少なくとも 1 つの XQuery データ・モデル (XDM) インスタンスが含まれることになります。複数のパスが指定された場合、XDM インスタンスはそれらのパスに均等に分散されます。

XMLSAVESCHEMA

すべての XML 列について XML スキーマ情報を保存することを指定します。エクスポートされた各 XML 文書のうち、挿入時に XML スキーマに関する妥当性検査が実行されたものについては、そのスキーマの完全修飾 SQL ID が、対応する XML Data Specifier (XDS) 内に SCH 属性として格納されます。エクスポートされた文書に対して XML スキーマに関する妥当性検査が実行されていない場合、またはそのスキーマ・オブジェクトがデータベースにもやは存在しない場合は、対応する XDS に SCH 属性は含められません。

SQL ID のスキーマと名前の各部分は、XML スキーマに対応する SYSCAT.XSROBJECTS カタログ表の行の "OBJECTSCHEMA" および "OBJECTNAME" の値として格納されます。

XMLSAVESCHEMA オプションには、整形式 XML 文書を生成しない XQuery シーケンスとの互換性がありません。

例

次に示すのは、SAMPLE データベースにある STAFF 表から、ファイル myfile.ixf に情報をエクスポートする方法の一例です。これは、IXF フォーマットで出力されます。コマンドを発行する前に、SAMPLE データベースと接続していなければなりません。

```
CALL SYSPROC.ADMIN_CMD ('EXPORT to /home/user1/data/myfile.ixf
   OF ixf MESSAGES ON SERVER select * from staff')
```

使用上の注意

- EXPORT コマンドで使用されるすべてのパスは、サーバー上の有効な完全修飾パスでなければなりません。
- 表に LOB 列が含まれている場合、**LOBS TO** および **LOBFILE** 節を使用して、少なくとも 1 つの完全修飾 LOB パスと LOB 名が指定されていなければなりません。

- エクスポート・ユーティリティーは、操作の開始時に COMMIT ステートメントを発行しますが、これによってタイプ 2 接続の場合に、プロシージャーは SQL30090N、理由コード 2 を戻します。
- UCS-2 データベースから、区切り文字で区切られている ASCII (DEL) ファイルにエクスポートする場合、すべての文字データは、プロシージャーを実行する場所で有効なコード・ページに変換されます。文字ストリングと GRAPHIC ストリング・データのどちらも、サーバーの、同じ SBCS または MBCS コード・ページに変換されます。
- エクスポート操作を開始する前に、すべての表操作が完了し、すべてのロックが解除されていることを確認してください。これは、WITH HOLD でオープンされたすべてのカーソルをクローズした後で COMMIT を発行するか、または ROLLBACK を発行することによって行うことができます。
- SELECT ステートメントでは表の別名を使用できます。
- メッセージ・ファイルに置かれたメッセージには、メッセージ検索サービスから戻される情報が含まれています。各メッセージは新しい行から始まります。
- DEL フォーマット・ファイルへエクスポートするために 254 よりも長い文字データの列が選択されると、エクスポート・ユーティリティーは警告メッセージを生成します。
- PC/IXF インポートは、データベース間でデータを移動する場合に使用します。行区切り文字を含む文字データが区切り文字付き ASCII (DEL) ファイルにエクスポートされ、テキスト転送プログラムによって処理される場合、行区切り文字を含むフィールドは長さが変わることがあります。
- ソースとターゲットのデータベースが両方とも同じクライアントからアクセス可能である場合、ファイルのコピーというステップは必要ありません。
- DB2 Connect を使用して、DB2 for OS/390®、DB2 for VM and VSE、および DB2 for OS/400® などの DRDA® サーバーから表をエクスポートすることができます。 PC/IXF エクスポートだけがサポートされています。
- IXF フォーマットにエクスポートする際に、ID が IXF フォーマットでサポートされる最大サイズを超えている場合、エクスポートは成功しますが、生成されるデータ・ファイルは CREATE モードを使用する後続のインポート操作では使用できません。 SQL27984W が戻されます。
- Windows でディスケットにエクスポートしているときに 1 枚のディスケットの容量を超えるデータを持つ表がある場合、別のディスケットを挿入するようプロンプトが出され、複数パーツ PC/IXF ファイル (マルチボリューム PC/IXF ファイル、または論理的に分割された PC/IXF ファイルとして知られる) が生成されて別々のディスケットに保管されます。最後のファイルを除く各ファイルには、そのファイルが論理的に分割されていること、および次のファイルを検索する場所を示すための DB2 CONTINUATION RECORD (または略して "AC" Record) が書き込まれます。その後、ファイルがインポート・ユーティリティーおよびロード・ユーティリティーによって読み取られるように、ファイルを AIX システムに転送できます。エクスポート・ユーティリティーは、AIX システムから呼び出される場合、複数部分からなる PC/IXF ファイルを作成しません。詳しい使用法については、IMPORT コマンドまたは LOAD コマンドを参照してください。
- エクスポート・ユーティリティーは、提供される SELECT ステートメントが、`SELECT * FROM tablename` という形式である場合、 IXF ファイルの表の NOT NULL WITH DEFAULT 属性を保管します。

- 型付き表をエクスポートする場合、副選択ステートメントは、ターゲット表名と **WHERE** 節を指定することによってのみ表現することができます。階層をエクスポートする場合、全選択と *select-statement* は指定できません。
- IXF 以外のファイル形式の場合は、階層の全探索の方法、およびエクスポートする副表とが DB2 に知らされるよう、全探索順序リストを指定することをお勧めします。このリストを指定しない場合、階層内のすべての表がエクスポートされ、デフォルトの順序は OUTER 順序になります。 OUTER 関数によって指定されるデフォルトの順序を使うこともできます。
- インポート操作の間も同じ走査順序を使用します。ロード・ユーティリティーでは、階層または副階層のロードはサポートされていません。
- 保護行のある表からデータをエクスポートする場合は、セッション許可 ID の保持する LBAC 信用証明情報のために、エクスポートされる行が制限されることがあります。セッション許可 ID に読み取りアクセスがない行はエクスポートされません。エラーも警告も出ません。
- セッション許可 ID の保持する LBAC 信用証明情報のために、エクスポートに含まれている 1 つ以上の保護列からの読み取りが許可されない場合、エクスポートは失敗し、エラー (SQLSTATE 42512) が戻されます。
- export や db2move などのデータ移動ユーティリティーを実行する際に、照会コンパイラーが、基礎照会を基本表よりも MQT に対して実行する方が効率が良いと判別することがあります。そのような場合、照会は据え置きリフレッシュの MQT に対して実行され、ユーティリティーの結果は基礎表内のデータを正確に表していない可能性があります。
- エクスポートするパッケージは DATETIME ISO 形式を使用してバインドされるため、ストリング表記へのキャスト時にすべての日付/ 時間/ タイム・スタンプの値は ISO 形式に変換されます。 CLP パッケージは DATETIME LOC 形式 (ロケール固有形式) を使用してバインドされるため、CLP DATETIME 形式が ISO と異なる場合に CLP とエクスポート間で矛盾した振る舞いが見られる場合があります。例えば、以下の SELECT ステートメントは、以下の期待される結果を戻します。

  ```
  db2 select col2 from tab1 where char(col2)='05/10/2005';
     COL2
     ----------
     05/10/2005
     05/10/2005
     05/10/2005
     3 record(s) selected.
  ```

  しかし、同じ select 節を使用した export コマンドでは、期待される結果が戻されません。

  ```
  db2 export to test.del of del select col2 from test
  where char(col2)='05/10/2005';
     Number of rows exported: 0
  ```

  ここで、LOCALE 日付形式を ISO 形式に置き換えると、以下のように、期待される結果になります。

  ```
  db2 export to test.del of del select col2 from test
  where char(col2)='2005-05-10';
     Number of rows exported: 3
  ```

結果セット情報

コマンドの実行状況は、CALL ステートメントからの結果である SQLCA で戻されます。実行が成功したなら、以下の結果セットによりコマンドから追加情報が戻されます。

表 36. EXPORT コマンドから戻される結果セット

列名	データ・タイプ	説明
ROWS_EXPORTED	BIGINT	エクスポートされた行の総数。
MSG_RETRIEVAL	VARCHAR(512)	このユーティリティーによって作成されたメッセージを取り出すために使用する SQL ステートメント。以下に例を示します。 `SELECT SQLCODE, MSG FROM TABLE (SYSPROC.ADMIN_GET_MSGS ('3203498_txu')) AS MSG`
MSG_REMOVAL	VARCHAR(512)	このユーティリティーによって作成されたメッセージをクリーンアップするために使用する SQL ステートメント。以下に例を示します。 `CALL SYSPROC.ADMIN_REMOVE_MSGS ('3203498_txu')`

エクスポート・ユーティリティー用のファイル・タイプ修飾子

表 37. エクスポート・ユーティリティーで有効なファイル・タイプ修飾子: すべてのファイル・フォーマット

修飾子	説明
lobsinfile	*lob-path* には、LOB データの入ったファイルへのパスを指定します。 各パスには、データ・ファイル内で LOB ロケーション指定子 (LLS) によって示される 1 つ以上の LOB の入った、少なくとも 1 つのファイルが組み込まれます。 LLS は、LOB ファイル・パスに保管されるファイル内の LOB のロケーションのストリング表現です。 LLS の形式は、*filename.ext.nnn.mmm/* です。*filename.ext* は LOB を収めたファイルの名前、 *nnn* はファイル内の LOB のオフセット (バイト単位)、 *mmm* は LOB の長さ (バイト単位) を表します。例えば、ストリング `db2exp.001.123.456/` がデータ・ファイルに保管される場合、 LOB はファイル `db2exp.001` のオフセット 123 に位置し、456 バイト長です。 EXPORT の使用時に **lobsinfile** 修飾子を指定した場合、LOB データは **LOBS TO** 節に指定されたロケーションに置かれます。 指定しない場合、 LOB データはデータ・ファイル・ディレクトリーに送られます。 **LOBS TO** 節は、LOB ファイルが保管されるディレクトリーへのパスを 1 つ以上指定します。 LOB パスごとに少なくとも 1 つのファイルが存在し、各ファイルには少なくとも 1 つの LOB が入ります。 **LOBS TO** または **LOBFILE** オプションによって、LOBSINFILE 動作が暗黙的にアクティブ化されます。 NULL LOB を指定するには、サイズに -1 と入力します。サイズを 0 と指定すると、長さが 0 の LOB として扱われます。長さが -1 の NULL LOB の場合、オフセットとファイル名は無視されます。例えば、NULL LOB の LLS は、db2exp.001.7.-1/ のようになります。
xmlinsepfiles	各 XQuery データ・モデル (XDM) インスタンスが別のファイルに書き込まれます。デフォルトでは、同じファイルの中で複数の値が連結されます。

表 37. エクスポート・ユーティリティーで有効なファイル・タイプ修飾子: すべてのファイル・フォーマット (続き)

修飾子	説明
lobsinsepfiles	各 LOB 値が別のファイルに書き込まれます。デフォルトでは、同じファイルの中で複数の値が連結されます。
xmlnodeclaration	XDM インスタンスが XML 宣言タグなしで書き込まれます。デフォルトでは、XDM インスタンスのエクスポート時に、エンコード属性を指定した XML 宣言タグが先頭に組み込まれます。
xmlchar	XDM インスタンスが文字コード・ページで書き込まれます。文字コード・ページは codepage ファイル・タイプ修飾子で指定されている値であるか、または指定がない場合はアプリケーションのコード・ページであることに注意してください。デフォルトでは、XDM インスタンスは、Unicode で書き込まれます。
xmlgraphic	EXPORT コマンドで xmlgraphic 修飾子を指定すると、アプリケーション・コード・ページまたは codepage ファイル・タイプ修飾子にかかわりなく、エクスポート XML 文書は、UTF-16 コード・ページでエンコードされます。

表 38. エクスポート・ユーティリティーの有効なファイル・タイプ修飾子: DEL (区切り付き ASCII) ファイル形式

修飾子	説明

chardel*x*	*x* は単一文字のストリング区切り文字です。デフォルト値は、二重引用符 (") です。文字ストリングを囲む二重引用符の代わりに指定の文字を使用します。[2] 文字ストリング区切りとして二重引用符を明示的に指定する場合は、以下のように指定します。
	`modified by chardel""`
	以下のように、文字ストリング区切りとして単一引用符 (') を指定することもできます。
	`modified by chardel''`

codepage=*x*	*x* は、ASCII 文字ストリングです。この値は、出力データ・セットに含まれているデータのコード・ページとして解釈されます。エクスポート操作の実行中に、文字データは、アプリケーション・コード・ページからこのコード・ページに変換されます。 DBCS のみ (GRAPHIC)、混合 DBCS、および EUC の場合、区切り文字の範囲は x00 から x3F に制限されます。codepage 修飾子を lobsinfile 修飾子と一緒に使用することはできません。

coldel*x*	*x* は単一文字の列区切り文字です。デフォルト値はコンマ (,) です。列の終わりを示すコンマの代わりに指定の文字を使用します。[2]
	以下の例では coldel; を指定しているので、エクスポート・ユーティリティーは、セミコロン文字 (;) をエクスポート・データの列区切りとして使用します。
	`db2 "export to temp of del modified by coldel;`
	`select * from staff where dept = 20"`

decplusblank	正符号文字。正の 10 進値の接頭部として、正符号 (+) ではなくブランク・スペースを使用します。デフォルトのアクションでは、正の 10 進数の前に正符号 (+) が付けられます。
decpt*x*	*x* は、小数点文字としてピリオドの代わりに使用される単一の置換文字です。デフォルト値はピリオド (.) です。小数点文字として、ピリオドの代わりに指定の文字を使用します。[2]

表 38. エクスポート・ユーティリティーの有効なファイル・タイプ修飾子: DEL (区切り付き ASCII) ファイル形式 (続き)

修飾子	説明
nochardel	列データが文字区切りで囲まれなくなります。DB2 を使用してデータのインポートまたはロードを実行する場合は、このオプションを指定しないでください。これは、区切り文字を持たないベンダー・データ・ファイルをサポートするために用意されています。不適切に使用すると、データが損失または破壊される場合があります。 このオプションを chardelx または nodoubledel と一緒に指定することはできません。これらは、相互に排他的なオプションです。
nodoubledel	二重文字区切りの認識を抑止します。[2]

striplzeros	エクスポートするすべての 10 進数列から先行ゼロを除去します。
	
	以下の例について考慮します。
	
	`db2 create table decimalTable ( c1 decimal( 31, 2 ) )`
	`db2 insert into decimalTable values ( 1.1 )`
	
	`db2 export to data of del select * from decimalTable`
	
	`db2 export to data of del modified by STRIPLZEROS`
	`select * from decimalTable`
	
	最初のエクスポート操作では、エクスポート・ファイル・データの内容が +0000000000000000000000000000001.10 になります。 2 番目の操作は、striplzeros 修飾子以外は最初の操作と同じですが、この場合は、エクスポート・ファイル・データの内容が +1.10 になります。

表 38. エクスポート・ユーティリティーの有効なファイル・タイプ修飾子: DEL (区切り付き ASCII) ファイル形式 (続き)

修飾子	説明

timestampformat="*x*"	*x* は、ソース・ファイルのタイム・スタンプの形式です。[4] 有効なタイム・スタンプ・エレメントは、以下のとおりです。
	
	YYYY - 年 (0000 から 9999 の範囲の 4 桁の数)
	M - 月 (1 から 12 の範囲の 1 桁または 2 桁の数)
	MM - 月 (01 から 12 の 2 桁の数。
	M および MMM とは相互に排他的)
	MMM - 月 (大文字小文字を区別しない月名の 3 文字の省略形。
	M と MM とは相互に排他的)
	D - 日 (1 から 31 の範囲の 1 桁または 2 桁の数)
	DD - 日 (1 から 31 の範囲の 2 桁の数。D とは相互に排他的)
	DDD - 元日から数えた日数 (001 から 366 の範囲の 3 桁の数。
	他の日または月のエレメントとは相互に排他的)
	H - 時 (12 時間制の場合は 0 から 12、24 時間制では 0 から 24 の範囲の 1 桁または 2 桁の数。)
	HH - 時 (12 時間制の場合は 0 から 12、24 時間制では 0 から 24 の範囲の 2 桁の数。)
	H と相互に排他的)
	M - 分 (0 から 59 の範囲の 1 桁または 2 桁の数)
	MM - 分 (0 から 59 の範囲の 2 桁の数。
	M (分) とは相互に排他的)
	S - 秒 (0 から 59 の範囲の 1 桁または 2 桁の数)
	SS - 秒 (0 から 59 の範囲の 2 桁の数。
	S と相互に排他的)
	SSSSS - 夜中の 12 時から数えた秒数
	(00000 から 86399 の範囲の 5 桁の数。
	他の時刻エレメントとは相互に排他的)
	U (1 から 12 時)
	- 小数秒 (U のオカレンス数は、各桁を 0 から 9 の範囲として、
	桁数を表します)
	TT - 午前/午後の指定子 (AM または PM)
	
	タイム・スタンプ・フォーマットの例を以下に示します。
	
	`"YYYY/MM/DD HH:MM:SS.UUUUUU"`
	
	MMM エレメントは、以下の値を生成します。「Jan」、 「Feb」、 「Mar」、 「Apr」、 「May」、 「Jun」、 「Jul」、 「Aug」、 「Sep」、 「Oct」、 「Nov」、および「Dec」。「Jan」は 1 月と等しく、「Dec」は 12 月と等しいです。
	
	以下の例は、「schedule」という表から、ユーザー定義のタイム・スタンプ・フォーマットを示すデータをエクスポートする方法を示しています。
	
	`db2 export to delfile2 of del`
	`modified by timestampformat="yyyy.mm.dd hh:mm tt"`
	`select * from schedule`

表 39. エクスポート・ユーティリティーで有効なファイル・タイプ修飾子: IXF ファイル・フォーマット

修飾子	説明
codepage=*x*	*x* は、ASCII 文字ストリングです。この値は、出力データ・セットに含まれているデータのコード・ページとして解釈されます。エクスポート操作中に、文字データをこのコード・ページからアプリケーション・コード・ページに変換します。 DBCS のみ (GRAPHIC)、混合 DBCS、および EUC の場合、区切り文字の範囲は x00 から x3F に制限されます。

表 40. エクスポート・ユーティリティーの有効なファイル・タイプ修飾子: WSF ファイル形式 [6]

修飾子	説明
1	Lotus 1-2-3 リリース 1、またはLotus 1-2-3 リリース 1a との互換がある WSF ファイルを作成します。[5]この値がデフォルトです。
2	Lotus Symphony リリース 1.0 と互換性のある WSF ファイルを作成します。[5]
3	Lotus 1-2-3 バージョン 2、またはLotus Symphony リリース 1.1 との互換がある WSF ファイルを作成します。[5]
4	DBCS 文字が含まれている WSF ファイルを作成します。

注:

1. サポートされていないファイル・タイプを **MODIFIED BY** オプションで使用しようとしても、エクスポート・ユーティリティーは警告を出しません。サポートされていないファイル・タイプを使おうとすると、エクスポート操作は失敗し、エラー・コードが戻されます。
2. *データ移動のための区切り文字の制約事項*に、区切り文字の指定変更として使用できる文字に適用される制限のリストが示されています。
3. 通常、エクスポート・ユーティリティーは、以下のような形式で書き込みを行います。
   - 日付データ: *YYYYMMDD* の形式
   - 文字 (日付) データ: *YYYY -MM -DD* の形式
   - 時刻データ: *HH.MM.SS* の形式
   - タイム・スタンプ・データ: *YYYY -MM -DD -HH.MM.SS.uuuuuu* の形式

   エクスポート操作のために SELECT ステートメントで指定される日時列に組み込まれたデータも、これらの形式になります。
4. タイム・スタンプ・フォーマットの場合、月の記述子と分の記述子のどちらも文字 M を使用するため、区別があいまいにならないように注意する必要があります。月のフィールドは、他の日付フィールドと隣接していなければなりません。分フィールドは、他の時刻フィールドに隣接していなければなりません。あいまいなタイム・スタンプ形式の例を以下に示します。

   ```
   "M" (月または分のどちらにもとれる)
   "M:M" (月と分の区別がつかない)
   "M:YYYY:M" (両方とも月と解釈される)
   "S:M:YYYY" (時刻値と日付値の両方に隣接している)
   ```

   あいまいな場合、ユーティリティーはエラー・メッセージを報告し、操作は失敗します。

   以下に、明確なタイム・スタンプ・フォーマットを示します。

   ```
   "M:YYYY" (M (月))
   "S:M" (M (分))
   "M:YYYY:S:M" (M (月)....M (分))
   "M:H:YYYY:M:D" (M (分)....M (月))
   ```
5. *filetype-mod* パラメーター・ストリングの中で、Lotus 1-2-3 の場合は L、Symphony の場合は S を指定すれば、これらのファイルを特定の製品に送ることができます。指定できるのは、1 つの値または製品指定子だけです。WSF ファイル・フォーマットのサポートは推奨されておらず、今後のリリースで除去

される可能性があります。サポートが除去される前に、WSF ファイルの代わりに、サポートされているファイル・フォーマットの使用を開始することを推奨します。

6. XML 列では、WSF ファイル形式はサポートされていません。このファイル・フォーマットのサポートは推奨されておらず、今後のリリースで除去される可能性があります。サポートが除去される前に、WSF ファイルの代わりに、サポートされているファイル・フォーマットの使用を開始することを推奨します。
7. **XMLFILE** 節または **XML TO** 節のどちらも指定されていない場合でも、すべての XDM インスタンスは、メイン・データ・ファイルとは別個の XML ファイルに書き込まれます。デフォルトで、XML ファイルはエクスポートされるデータ・ファイルのパスに書き込まれます。 XML ファイルのデフォルトのベース名は、エクスポートされるデータ・ファイル名に拡張子 ".xml" を追加したものとなります。
8. XMLNODECLARATION ファイル・タイプ修飾子が指定されていなければ、すべての XDM インスタンスはエンコード属性を含む XML 宣言を先頭に付けて書き込まれます。
9. XMLCHAR または XMLGRAPHIC ファイル・タイプ修飾子が指定されていなければ、デフォルトで、すべての XDM インスタンスは Unicode で書き込まれます。
10. XML データおよび LOB データのデフォルト・パスは、メイン・データ・ファイルのパスです。デフォルトの XML ファイルのベース名は、メイン・データ・ファイルです。デフォルトの LOB ファイルのベース名は、メイン・データ・ファイルです。例えば、メイン・データ・ファイルが

    ```
    /mypath/myfile.del
    ```

    の場合、XML データおよび LOB データのデフォルト・パスは

    ```
    /mypath"
    ```

    となり、デフォルトの XML ファイルのベース名は

    ```
    myfile.del
    ```

    となり、デフォルトの LOB ファイルのベース名は

    ```
    myfile.del
    ```

    LOB ファイルを生成するには、LOBSINFILE ファイル・タイプ修飾子が指定されている必要があります。
11. エクスポート・ユーティリティーは、数値 ID を各 LOB ファイルまたは XML ファイルに付加します。この ID は、0 が埋め込まれた 3 桁のシーケンス値で、

    ```
    .001
    ```

    から始まります。 999 番目の LOB ファイルまたは XML ファイルの後には、ID に 0 が埋め込まれることはなくなります (例えば、1000 番目の LOG ファイルまたは XML ファイルの拡張子は

    ```
    .1000
    ```

    となります。数値 ID の後に、データ・タイプを表す次のどちらかの 3 文字のタイプ ID が続きます。

.lob

または

.xml

例えば、生成される LOB ファイルの名前は

myfile.del.001.lob

の形式となり、生成される XML ファイルの名前は

myfile.del.001.xml

12. の形式となります。XQuery を指定すると、エクスポート・ユーティリティーは、整形式でない文書である XDM インスタンスをエクスポートできます。ただし、XML 列に含めることができるのは完全な文書だけなので、これらのエクスポートされた文書を XML 列に直接インポートまたはロードすることはできません。

FORCE APPLICATION コマンド (ADMIN_CMD プロシージャーを使用)

システムからローカルまたはリモートのユーザーやアプリケーションを強制終了し、サーバー上での保守を可能にします。

重要: 割り込みが許されない操作 (例えば、RESTORE DATABASE) を強制終了する場合は、その操作を再び実行して正常終了しなければデータベースは使用可能になりません。

有効範囲

このコマンドは、$HOME/sqllib/db2nodes.cfg ファイルにリストされているすべてのデータベース・パーティションに影響を与えます。

パーティション・データベース環境では、このコマンドを発行するのは、強制終了するアプリケーションのコーディネーター・データベース・パーティションからでなくてもかまいません。パーティション・データベース環境内の任意のノード (データベース・パーティション・サーバー) から発行することができます。

許可

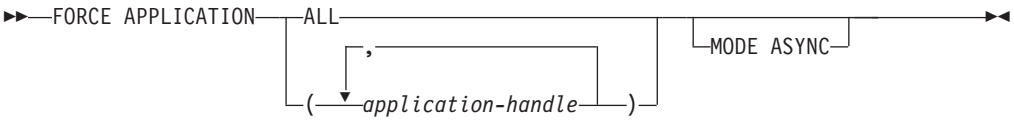
以下のいずれか。

- SYSADM
- SYSCTRL
- SYSMAINT

必要な接続

データベース

コマンド構文



コマンド・パラメーター

FORCE APPLICATION

ALL すべてのアプリケーションがデータベースから切断されます。これにより、ADMIN_CMD プロシージャーが実行されている接続がクローズされることがあります。その場合、強制終了オペレーションが正常に完了した時点で、ADMIN_CMD プロシージャーについては SQL1224N エラーが戻されることになります。

application-handle
エージェントの終了を指定します。 LIST APPLICATIONS コマンドを使用して値をリストします。

MODE ASYNC
コマンドは、指定したすべてのユーザーが終了するのを待たずに戻ってきます。コマンドは、機能を正常に発行するか、またはエラー (無効な構文などの) を発見するとすぐに戻ります。

現在サポートしているモードはこのモードだけです。

例

次の例は、*application-handle* の値が 41408 と 55458 の 2 つのユーザーをデータベースから強制的に切断します。

```
CALL SYSPROC.ADMIN_CMD( 'force application ( 41408, 55458 )' )
```

使用上の注意

データベース・マネージャーは、db2start を必要とせずに、後続のデータベース・マネージャー操作を処理できるようにするため、アクティブなままになっています。

データベースの整合性を確保するため、終了できるのは、アイドル中のユーザー、または割り込み可能なデータベース操作を実行中のユーザーだけです。

以下のタイプのユーザーおよびアプリケーションは、強制終了できません。

- データベースを作成しているユーザー
- システム・アプリケーション

これらのタイプのユーザーおよびアプリケーションを正常に強制終了するには、データベースの非活動化とインスタンスの再始動のいずれかまたは両方を行う必要があります。

FORCE APPLICATION が出された後も、データベースはまだ接続要求を受諾します。すべてのユーザーを完全に強制終了するためには、追加の FORCE が必要になる場合があります。

コマンドの実行状況は、CALL ステートメントからの結果である SQLCA で戻されます。

ADMIN_CMD プロシージャーを使用する GET STMM TUNING DBPARTITIONNUM コマンド

ユーザー設定のセルフチューニング・メモリー・マネージャー (STMM) の調整データベース・パーティション番号、および現在の STMM 調整データベース・パーティション番号を報告するカタログ表の読み取りに使用します。

許可

ステートメントの許可 ID によって保持されている特権には、少なくとも以下のいずれかの権限または特権が含まれていなければなりません。

- DBADM
- SECADM
- SQLADM
- ACCESSCTRL
- DATAACCESS
- SYSIBM.SYSTUNINGINFO に対する SELECT

必要な接続

データベース

コマンド構文

例

```
CALL SYSPROC.ADMIN_CMD( 'get stmm tuning dbpartitionnum' )
```

以下はこの照会の出力例です。

```
Result set 1
  --------------

  USER_PREFERRED_NUMBER CURRENT_NUMBER
  --------------------- --------------
                      2              2

  1 record(s) selected.

  Return Status = 0
```

使用上の注意

ユーザー設定のセルフチューニング・メモリー・マネージャー (STMM) の調整データベース・パーティション番号 (USER_PREFERRED_NUMBER) は、ユーザーにより設定され、メモリー・チューナーの実行対象にするデータベース・パーティションを指定します。データベースの稼働中に、調整パーティションは 1 時間に数度、非同期に更新されます。結果として、戻される CURRENT_NUMBER および USER_PREFERRED_NUMBER は、ユーザー設定の STMM パーティション番号の更新後にも同期がとれていない可能性があります。これを解決するために、CURRENT_NUMBER が非同期に更新されるのを待機するか、またはデータベース

をいったん停止してから開始し、CURRENT_NUMBER の更新を強制します。

結果セット情報

コマンドの実行状況は、CALL ステートメントからの結果である SQLCA で戻されます。実行が成功した場合、コマンドは以下の結果セットで追加情報を戻します。

表 41. GET STMM TUNING DBPARTITIONNUM コマンドで戻される結果セット

列名	データ・タイプ	説明
USER_PREFERRED_NUMBER	INTEGER	ユーザー設定のセルフチューニング・メモリー・マネージャー (STMM) の調整データベース・パーティション番号。値 -1 は、デフォルトのデータベース・パーティションの使用を示します。
CURRENT_NUMBER	INTEGER	現在の STMM 調整データベース・パーティション番号。値 -1 は、デフォルトのデータベース・パーティションの使用を示します。

IMPORT コマンド (ADMIN_CMD プロシージャーを使用)

外部ファイルのデータを、サポートされているファイル・フォーマットで表、階層、ビュー、またはニックネームに挿入します。 LOAD はより高速な代替方法です。しかしロード・ユーティリティーでは、階層レベルのデータのロードはサポートされていません。

105 ページの『インポート・ユーティリティー用のファイル・タイプ修飾子』へのクイック・リンク。

許可

- INSERT オプションを使用して **IMPORT** する場合、以下のいずれかが必要です。
  - DATAACCESS 権限
  - 関係するそれぞれの表、ビュー、またはニックネームに対する CONTROL 特権
  - 関係するそれぞれの表またはビューに対する INSERT および SELECT 特権
- INSERT_UPDATE オプションを使用して既存の表に **IMPORT** するには、以下のいずれかが必要です。
  - DATAACCESS 権限
  - 関係するそれぞれの表、ビュー、またはニックネームに対する CONTROL 特権
  - 関係するそれぞれの表またはビューに対する INSERT、 SELECT、 UPDATE、および DELETE 特権
- REPLACE または **REPLACE_CREATE** オプションを使用して既存の表に **IMPORT** するには、以下のいずれかが必要です。
  - DATAACCESS 権限

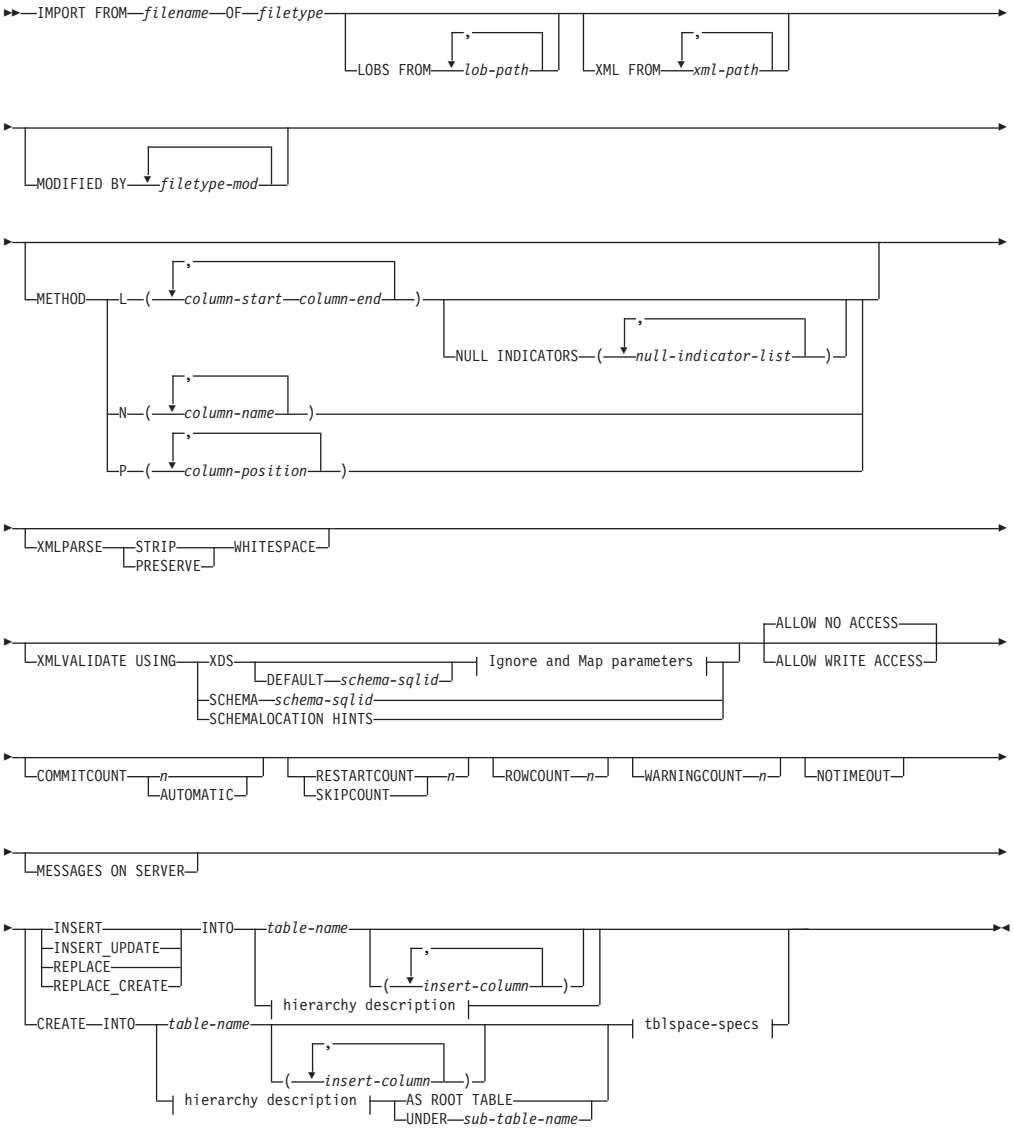
  - 表またはビューに対する CONTROL 特権
  - 表またはビューに対する INSERT、SELECT、および DELETE 特権
- CREATE または **REPLACE_CREATE** オプションを使用して新規の表に **IMPORT** するには、以下のいずれかが必要です。
  - DBADM 権限
  - データベースに対する CREATETAB 権限および表スペースに対する USE 特権に加えて、以下のいずれかが必要です。
    - データベースに対する IMPLICIT_SCHEMA 権限 (表の暗黙的または明示的スキーマ名が存在しない場合)
    - スキーマに対する CREATEIN 特権 (表のスキーマ名が既存のスキーマを指す場合)
- CREATE または **REPLACE_CREATE** オプションを使って、存在しない階層に **IMPORT** するには、以下のいずれかが必要です。
  - DBADM 権限
  - データベースに対する CREATETAB 権限および表スペースに対する USE 特権と、以下のいずれかが必要です。
    - データベースに対する IMPLICIT_SCHEMA 権限 (表のスキーマ名が存在しない場合)
    - スキーマに対する CREATEIN 特権 (表のスキーマが存在する場合)
    - 階層全体に対して **REPLACE_CREATE** オプションが使用されている場合は、階層内のすべての副表に対する CONTROL 特権
- REPLACE オプションを使用して既存の階層に **IMPORT** するには、以下のどれかが必要です。
  - DATAACCESS 権限
  - 階層内のすべての副表に対する CONTROL 特権
- 保護列のある表にデータをインポートするには、セッション許可 ID に、その表内のすべての保護列への書き込みアクセスを許可する LBAC 信用証明情報が必要です。そうでない場合、インポートは失敗し、エラー (SQLSTATE 42512) が戻されます。
- 保護されている行のある表にデータをインポートするには、セッション許可 ID に、以下の基準を満たす LBAC 信用証明情報が必要です。
  - 表を保護しているセキュリティー・ポリシーの一部である
  - 書き込みアクセスに関して、セッション許可 ID に付与された

  挿入する行のラベル、ユーザーの LBAC 信用証明情報、セキュリティー・ポリシー定義、および LBAC 規則が行のラベルを決定します。
- **REPLACE** または **REPLACE_CREATE** オプションが指定された場合、セッション許可 ID には、その表をドロップするための権限が付与されていなければなりません。
- ニックネームにデータをインポートするには、セッション許可 ID には指定したデータ・ソースにパススルー・モードでアクセスおよび使用する特権がなければなりません。

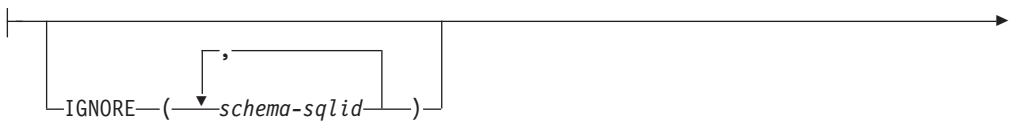
必要な接続

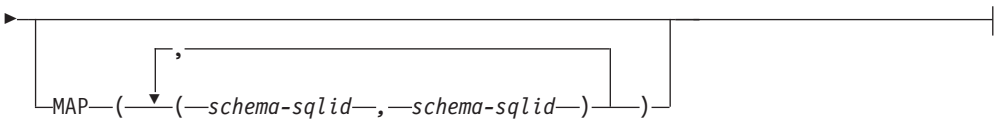
データベース。 Linux、UNIX、または Windows クライアントから Linux、UNIX、または Windows データベース・サーバーへのユーティリティー・アクセスは、DB2 Connect ゲートウェイまたはループバック環境を経由してではなく、エンジンを使用した直接接続でなければなりません。

コマンド構文

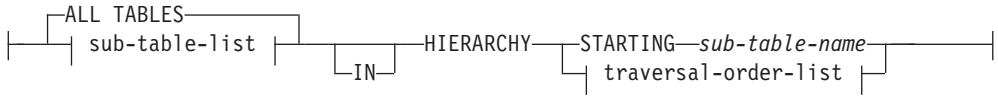


Ignore and Map parameters:

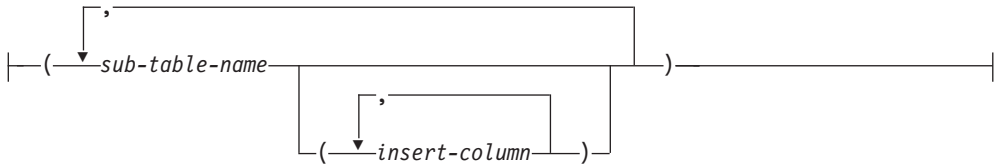




hierarchy description:

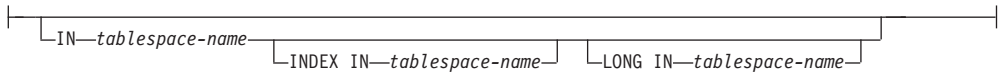


sub-table-list:



traversal-order-list:

tblspace-specs:



コマンド・パラメーター

ALL TABLES
階層専用の暗黙キーワード。階層をインポートする場合、走査順序で指定されるすべての表をインポートすることがデフォルトです。

ALLOW NO ACCESS
オフライン・モードでインポートを実行します。行の挿入の前には常に、ターゲット表に排他 (X) ロックがかけられます。これで、同時アプリケーションは表データにアクセスできなくなります。これがデフォルトのインポート動作です。

ALLOW WRITE ACCESS
オンライン・モードでインポートを実行します。最初の行の挿入時には、ターゲット表に意図的排他 (IX) ロックがかけられます。これで、表データへの同時の読み取りおよび書き出しアクセスが可能になります。オンライン・モードには、**REPLACE**、**CREATE**、または **REPLACE_CREATE** インポート・オプションとの互換性はありません。オンライン・モードとバッファー挿入との連携はサポートされません。インポート操作によって挿入後のデータが定期的にコミットされるので、表ロックへのロック・エスカレーショ

ンが削減され、アクティブなログ・スペースが使い果たされることはなくなります。このようなコミットは、**COMMITCOUNT** オプションを使わなくても実行されます。各コミットごとに、インポートでは IX 表ロックが外されるので、コミットの完了後に再びロックの設定が試みられます。ニックネームにインポートするときにはこのパラメーターが必要で、有効な数値を使って **COMMITCOUNT** を指定する必要があります (AUTOMATIC は有効なオプションとは見なされません)。

AS ROOT TABLE
1 つ以上の副表を、独立した表階層として作成します。

COMMITCOUNT *n* | **AUTOMATIC**
n 個のレコードがインポートされるたびに COMMIT を実行します。数 *n* を指定すると、インポートでは *n* 個のレコードのインポートの後にそのつど COMMIT が実行されます。コンパウンド挿入を使用した場合、ユーザー指定のコミット頻度 *n* は、そのコンパウンド・カウント値に最も近い整数の倍数に切り上げられます。 AUTOMATIC を指定すると、コミットの必要時期はインポート操作で内部的に判別されます。次の 2 つのうちのいずれかの理由で、このユーティリティーはコミットを行います。

- アクティブ・ログ・スペースを使いきらないようにするため。
- ロックが行レベルから表レベルにエスカレーションしないようにするため。

ALLOW WRITE ACCESS オプションを指定した場合に **COMMITCOUNT** オプションを指定しないと、インポート・ユーティリティーは、**COMMITCOUNT** AUTOMATIC が指定されたものとしてコミットを実行します。

アクティブ・ログ・スペースを使いきらないようにする点でのインポート操作の能力は、以下のように DB2 レジストリー変数 **DB2_FORCE_APP_ON_MAX_LOG** の影響を受けます。

- **DB2_FORCE_APP_ON_MAX_LOG** を FALSE にして、**COMMITCOUNT** AUTOMATIC コマンド・オプションを指定した場合、インポート・ユーティリティーは自動的にアクティブ・ログ・スペースを使い切らないようにすることができます。
- **DB2_FORCE_APP_ON_MAX_LOG** を FALSE にして、**COMMITCOUNT** *n* コマンド・オプションを指定した場合、インポート・ユーティリティーはレコードを挿入または更新しようとして SQL0964C (トランザクション・ログがフル) を検出したら、ログ・フル状態が解決しようとします。無条件のコミットを実行してから、レコードの挿入または更新を再試行します。これで問題が解決しない場合 (ログ・フルがデータベース上の他のアクティビティーに起因する場合など)、予期されるように IMPORT コマンドは失敗しますが、コミットされる行数は **COMMITCOUNT** *n* 値の倍数にならないことがあります。インポート操作を再試行する際に既にコミットされた行を処理しないようにするには、 **RESTARTCOUNT** または **SKIPCOUNT** コマンド・パラメーターを使用します。
- **DB2_FORCE_APP_ON_MAX_LOG** が TRUE (これがデフォルトです) の場合、レコードの挿入または更新中に SQL0964C を検出すると、イン

ポート操作は失敗します。これは、**COMMITCOUNT** AUTOMATIC と **COMMITCOUNT** *n* のどちらを指定したかに関係なく発生します。

アプリケーションは強制的にデータベースから切断され、現在の作業単位はロールバックされます。インポート操作を再試行する際に既にコミットされた行を処理しないようにするには、 **RESTARTCOUNT** または **SKIPCOUNT** コマンド・パラメーターを使用します。

CREATE

注: CREATE パラメーターは推奨されておらず、今後のリリースで除去される可能性があります。さらに詳しくは、『IMPORT コマンドの推奨されなくなったオプション **CREATE** および **REPLACE_CREATE**』を参照してください。

データベースのコード・ページで表の定義と行の内容を作成します。 DB2 の表、副表、または階層からエクスポートされたデータの場合、索引も作成されます。このオプションが階層に対するものである場合に、DB2 からデータがエクスポートされると、タイプ階層も作成されます。このオプションは、IXF ファイルの場合にのみ使用することができます。

ニックネームにインポートするときには、このパラメーターは無効です。

注: データが MVS™ ホスト・データベースからエクスポートされたもので、ページ・サイズで計算した長さが 254 より大きい LONGVAR フィールドを含んでいる場合、**CREATE** は行が長過ぎるために失敗します。制約事項のリストは、『インポート済みの表の再作成』 を参照してください。この場合、その表は手動で作成します。そして、IMPORT に **INSERT** を指定して呼び出すか、または LOAD コマンドを使用してください。

DEFAULT *schema-sqlid*

このオプションは、**USING XDS** パラメーターを指定した場合にのみ使用できます。**DEFAULT** 節で指定されたスキーマは、インポート対象 XML 文書の XML Data Specifier (XDS) に XML スキーマを指定する SCH 属性が含まれていない場合に、妥当性検査のために使用するスキーマとなります。

DEFAULT 節は、**IGNORE** 節および **MAP** 節よりも優先されます。 XDS が **DEFAULT** 節を満たすなら、**IGNORE** と **MAP** の指定は無視されます。

FROM *filename*

インポートするデータの入ったファイルの名前を指定します。これは完全修飾パスでなければならず、データベース・サーバー上にそのファイルが存在していなければなりません。

HIERARCHY

階層データをインポートするよう指定します。

IGNORE *schema-sqlid*

このオプションは、**USING XDS** パラメーターを指定した場合にのみ使用できます。**IGNORE** 節は、SCH 属性によって指定されていても無視するスキーマとして、1 つ以上のスキーマのリストを指定します。インポートする XML 文書の XML Data Specifier の中に SCH 属性が存在し、その SCH

属性によって指定されるスキーマが無視するスキーマ・リストに含まれている場合には、インポートするその XML 文書についてスキーマ妥当性検査は実行されません。

あるスキーマが **IGNORE** 節の中で指定されている場合、**MAP** 節のスキーマ・ペアの左辺にそれを含めることはできません。

IGNORE 節は XDS にのみ適用されます。あるスキーマが **IGNORE** 節によって指定されていても、それが **MAP** 節によってマップされているなら、それ以降そのスキーマが無視されることはありません。

IN *tablespace-name*
表を作成する表スペースを指定します。表スペースは存在している必要があり、REGULAR 表スペースでなければなりません。他の表スペースを指定しない場合、すべての表パーツはこの表スペースに保管されます。この節を指定しない場合、表は許可 ID によって作成された表スペース中に作成されます。何も検出されない場合、その表はデフォルト表スペースの USERSPACE1 に入れられます。 USERSPACE1 がドロップされていた場合、表作成は失敗します。

INDEX IN *tablespace-name*
表の索引を作成する表スペースを指定します。このオプションは、**IN** 節で指定される PRIMARY 表スペースが DMS 表スペースである場合のみ使用できます。指定した表スペースは存在している必要があり、かつ REGULAR または LARGE DMS 表スペースでなければなりません。

注: どの表スペースに索引を配置するかは、表を作成するときにのみ指定できます。

insert-column
データの挿入先となる表またはビュー内の列名を指定します。

INSERT
既存の表データを変更することなく、インポートされたデータを表に追加します。

INSERT_UPDATE
インポートしたデータ行をターゲット表に追加するか、または主キーが一致するものがあればターゲット表の既存行を更新します。

INTO *table-name*
データのインポート先となるデータベース表を指定します。この表として、システム表、作成済み一時表、宣言済み一時表、またはサマリー表は指定できません。

以前のサーバーの場合を除き、**INSERT**、**INSERT_UPDATE**、または **REPLACE** オプションには、完全修飾または非修飾の表名を使用しなければならないようなときでも、別名を使用することができます。修飾された表名は、*schema.tablename* の形式になります。 *schema* には、表作成時のユーザー名が入ります。

LOBS FROM *lob-path*
LOB ファイルを保管する 1 つ以上の完全修飾パスを指定します。それらのパスは、データベース・サーバーのコーディネーター・パーティション上に存在していなければなりません。 LOB データ・ファイルの名前は、メイ

ン・データ・ファイル (ASC、DEL、または IXF) の、LOB 列にロードされる列内に保管されます。指定できるパスの最大数は 999 です。これによって、LOBSINFILE 動作が暗黙的にアクティブ化されます。

ニックネームにインポートするときには、このパラメーターは無効です。

LONG IN *tablespace-name*

ロング列の値 (LONG VARCHAR、LONG VARGRAPHIC、LOB データ・タイプ、またはソース・タイプとしてこれらが指定されている特殊タイプ) を保管する表スペースを指定します。このオプションは、**IN** 節で指定した PRIMARY 表スペースが DMS 表スペースである場合のみ使用できます。指定した表スペースは存在している必要があり、LARGE DMS 表スペースでなければなりません。

MAP *schema-sqlid*

このオプションは、**USING XDS** パラメーターを指定した場合にのみ使用できます。**MAP** 節は、インポートする各 XML 文書について XML Data Specifier (XDS) の SCH 属性によって指定されるスキーマの代わりに使用する代替スキーマを指定するのに使用します。**MAP** 節には、それぞれがあるスキーマから別のスキーマへのマッピングを表すスキーマ・ペアを 1 つ以上列挙したリストを指定します。ペア中の最初のスキーマは、XDS 内の SCH 属性によって参照されるスキーマを表します。ペア中の 2 番目のスキーマは、スキーマ検証の実行で使用する必要のあるスキーマを表します。

あるスキーマが **MAP** 節のスキーマ・ペアの左辺で指定されている場合、**IGNORE** 節でさらにそれを指定することはできません。

スキーマ・ペアのマッピングが適用されたなら、その結果は最終的なものです。マッピング操作は推移的ではないため、選択されたスキーマが、それ以降に別のスキーマ・ペアのマッピングに適用されることはありません。

スキーマを複数回マップすることはできません。つまり、複数のペアの左辺に指定することはできません。

MESSAGES ON SERVER

IMPORT コマンドによってサーバー上に作成されるメッセージ・ファイルを保存することを指定します。戻される結果セットには以下の 2 つの列が含まれます。 1 つは MSG_RETRIEVAL で、これはこの操作中に発生したすべての警告メッセージおよびエラー・メッセージを取り出すのに必要な SQL ステートメントです。もう 1 つは MSG_REMOVAL で、これはメッセージをクリーンアップするために必要な SQL ステートメントです。

この節が指定されていない場合は、ADMIN_CMD プロシージャーから呼び出し元に戻る時点でメッセージ・ファイルが削除されます。結果セット中の MSG_RETRIEVAL および MSG_REMOVAL 列の内容は NULL 値になります。

この節が指定されているかどうかに関係なく、fenced ユーザー ID に、**DB2_UTIL_MSGPATH** レジストリー変数の示すディレクトリーおよびデータのエクスポート先ディレクトリーの下にファイルを作成するための権限が付与されていることが必要です。

METHOD

L データのインポートを開始する列および終了する列の番号を指定し

ます。列の番号は、データの行の先頭からのバイト単位のオフセットです。この番号は 1 から始まります。

注: このメソッドは、ASC ファイルの場合にのみ使用することができ、そのファイル・タイプに対してのみ有効なオプションです。

N インポートするデータ・ファイルの中の列の名前を指定します。これらの列名の大文字小文字の区別は、システム・カタログ内の対応する名前の大文字小文字の区別と一致しなければなりません。NULL 可能ではない各表の列には、**METHOD N** リスト内に対応する項目が必要です。例えば、データ・フィールドが F1、 F2、 F3、 F4、 F5、および F6 であり、表の列が C1 INT、C2 INT NOT NULL、C3 INT NOT NULL、および C4 INT の場合、method N (F2, F1, F4, F3) は有効な要求ですが、 method N (F2, F1) は無効です。

注: この方式は、IXF ファイルの場合にのみ使用することができます。

P インポートする入力データ・フィールドのフィールド番号を指定します。

注: この方式は、IXF または DEL ファイルの場合にのみ使用でき、DEL ファイル・タイプに対してのみ有効なオプションです。

MODIFIED BY *filetype-mod*

ファイル・タイプ修飾子オプションを指定します。 105 ページの『インポート・ユーティリティー用のファイル・タイプ修飾子』を参照してください。

NOTIMEOUT

インポート・ユーティリティーがロックの待機中にタイムアウトしないことを指定します。このオプションのほうが、 **locktimeout** データベース構成パラメーターより優先されます。他のアプリケーションは影響を受けません。

NULL INDICATORS *null-indicator-list*

このオプションは、**METHOD L** パラメーターを指定した場合にのみ使用できます。つまり、入力ファイルが ASC ファイルの場合です。 NULL 標識リストは、コンマで区切られた正の整数のリストで、各 NULL 標識フィールドの列の番号を指定します。列の番号は、データの行の先頭からのバイト単位の、各 NULL 標識フィールドのオフセットです。 NULL 標識リストには、**METHOD L** パラメーターで定義された各データ・フィールドに対する 1 つの項目がなければなりません。列の番号がゼロであることは、対応するデータ・フィールド内に必ずデータがあることを示します。

NULL 標識列中の `Y` の値は、その列データが NULL であることを指定します。 NULL 標識列に `Y` *以外* の文字を指定した場合は、列データが NULL ではなく、**METHOD L** オプションで指定された列データがインポートされることを指定することになります。

`nullindchar` ファイル・タイプ修飾子を指定した **MODIFIED BY** オプションを使用すれば、NULL 標識文字を変更することができます。

OF *filetype*

入力ファイル内のデータのフォーマットを指定します。

- ASC (区切りなし ASCII フォーマット)
- DEL (区切り文字付き ASCII フォーマット)。さまざまなデータベース・マネージャーやファイル・マネージャー・プログラムで使用します。
- WSF (ワークシート・フォーマット)。以下のプログラムで使用します。
  - Lotus 1-2-3
  - Lotus Symphony
- IXF (統合交換フォーマット、PC バージョン) は、DB2 によって排他的に使用されるバイナリー・フォーマットです。

重要: WSF ファイル・フォーマットのサポートは推奨されておらず、今後のリリースで除去される可能性があります。サポートが除去される前に、WSF ファイルの代わりに、サポートされているファイル・フォーマットの使用を開始することを推奨します。

ニックネームにインポートするときには、WSF ファイル・タイプはサポートされません。

REPLACE

データ・オブジェクトを切り捨てることによって表内の既存のデータすべてを削除してから、インポートしたデータを挿入します。表定義および索引定義は変更されません。表がない場合は、このオプションを使用できません。階層間でデータを移動する際にこのオプションを使用する場合は、階層全体に関係したデータだけが置き換えられます。副表は置き換えられません。

ニックネームにインポートするときには、このパラメーターは無効です。

このオプションでは、CREATE TABLE ステートメントの NOT LOGGED INITIALLY (NLI) 節、あるいは ALTER TABLE ステートメントの ACTIVE NOT LOGGED INITIALLY 節は考慮されません。

NLI 節が呼び出される CREATE TABLE または ALTER TABLE ステートメントと同じトランザクションの中で、**REPLACE** オプションの指定されたインポートが実行された場合、インポートにおいてその NLI 節は考慮されません。すべての挿入操作がログ対象となります。

予備手段 1

DELETE ステートメントを使用して表の内容を削除した後、INSERT ステートメントによりインポートを呼び出す

予備手段 2

表をドロップしてからそれを再作成した後、INSERT ステートメントによってインポートを呼び出す

この制限は、DB2 Universal Database バージョン 7 および DB2 バージョン 8 に適用されます。

REPLACE_CREATE

注: REPLACE_CREATE パラメーターは推奨されておらず、今後のリリースで除去される可能性があります。さらに詳しくは、『IMPORT コマンドの推奨されなくなったオプション CREATE および REPLACE_CREATE』を参照してください。

表が既にある場合には、データ・オブジェクトを切り捨てることによって表内の既存のデータすべてを削除し、表定義や索引定義は変えることなく、インポートしたデータを挿入します。

表がまだない場合には、データベースのコード・ページで、表と索引の定義と行の内容を作成します。制約事項のリストは、*『インポート済みの表の再作成』* を参照してください。

このオプションは、IXF ファイルの場合にのみ使用することができます。階層間でデータを移動する際にこのオプションを使用する場合は、階層全体に関係したデータだけが置き換えられます。副表は置き換えられません。

ニックネームにインポートするときには、このパラメーターは無効です。

RESTARTCOUNT *n*

n + 1 の位置のレコードからインポート操作を開始するよう指定します。最初の *n* 個のレコードはスキップされます。このオプションは機能的には **SKIPCOUNT** と同等です。**RESTARTCOUNT** と **SKIPCOUNT** は相互に排他的です。

ROWCOUNT *n*

インポート (挿入または更新) するファイル内の物理レコードの数 *n* を指定します。ユーザーは、**SKIPCOUNT** または **RESTARTCOUNT** オプションで指示されたレコードから始めて、ファイルの *n* 行だけをインポートすることができます。**SKIPCOUNT** または **RESTARTCOUNT** オプションの指定がないと、最初の *n* 行がインポートされます。**SKIPCOUNT** *m* または **RESTARTCOUNT** *m* を指定すると、行 *m*+1 から *m*+*n* がインポートされます。コンパウンド挿入を使用した場合、ユーザー指定の **ROWCOUNT** *n* は、そのコンパウンド・カウント値に最も近い整数の倍数に切り上げられます。

SKIPCOUNT *n*

n + 1 の位置のレコードからインポート操作を開始するよう指定します。最初の *n* 個のレコードはスキップされます。このオプションは機能的には **RESTARTCOUNT** と同等です。 **SKIPCOUNT** と **RESTARTCOUNT** は相互に排他的です。

STARTING *sub-table-name*

階層専用キーワード。 *sub-table-name* から始まるデフォルト順を要求します。 PC/IXF ファイルの場合、デフォルト順は入力ファイルに保管されている順です。 PC/IXF ファイル・フォーマットの場合、デフォルト順は有効な唯一の順序です。

sub-table-list

型付き表で **INSERT** または **INSERT_UPDATE** オプションを指定した場合、データのインポート先副表を指定するために副表名のリストが使われます。

traversal-order-list
型付き表で **INSERT**、**INSERT_UPDATE**、または **REPLACE** オプションを指定した場合、インポートする階層内の副表の横断順序を指定するために副表名のリストを使います。

UNDER *sub-table-name*
1 つ以上の副表を作成する場合に親表を指定します。

WARNINGCOUNT *n*
n 個の警告後に、インポート操作を停止します。このパラメーターは、警告は予期されないが、正しいファイルと表が使用されていることを確認するのが望ましい場合に設定してください。インポート・ファイルまたはターゲット表が不適切に指定されると、インポート対象の各行ごとにインポート・ユーティリティーによって警告が生成され、このためにインポートが失敗する可能性があります。 *n* をゼロにした場合や、このオプションを指定しない場合、発行された警告の回数に関係なくインポート操作は続行します。

XML FROM *xml-path*
XML ファイルが含まれているパスを 1 つ以上指定します。

XMLPARSE
XML 文書の解析方法を指定します。このオプションが指定されていない場合、XML 文書の解析の動作は、CURRENT XMLPARSE OPTION 特殊レジスターの値によって決まります。

STRIP WHITESPACE
XML 文書の解析時に空白文字を除去することを指定します。

PRESERVE WHITESPACE
XML 文書の解析時に空白文字を除去しないことを指定します。

XMLVALIDATE
該当する場合に、XML 文書がスキーマに準拠しているかどうかの妥当性検査を実行することを指定します。

USING XDS
メイン・データ・ファイルの中で XML Data Specifier (XDS) によって指定されている XML スキーマに準拠しているかどうかについて、XML 文書の妥当性検査が実行されます。デフォルトでは、**USING XDS** 節によって **XMLVALIDATE** オプションが呼び出された場合、妥当性検査実行のために使用されるスキーマは、その XDS の SCH 属性によって決まります。 XDS の中で SCH 属性が指定されていない場合、**DEFAULT** 節によってデフォルト・スキーマが指定されているのでない限り、スキーマ妥当性検査は実行されません。

DEFAULT、**IGNORE**、および **MAP** 節を使用することにより、スキーマ決定の動作を変更することができます。これら 3 つの節はオプションであり、相互に適用されるのではなく XDS の指定に直接適用されます。例えば、**DEFAULT** 節で指定されているためにあるスキーマが選択された場合、それが **IGNORE** 節で指定されていたとしても無視されることはありません。同じように、MAP 節のペアの最初の部分で指定されているためにあるスキーマが選択された

場合、それが別の **MAP** 節のペアの 2 番目の部分で指定されていたとしても再びマップされることはありません。

USING SCHEMA *schema-sqlid*
指定されている SQL ID の XML スキーマに準拠しているかどうかについて、XML 文書の妥当性検査が実行されます。この場合、すべての XML 列について XML Data Specifier (XDS) の SCH 属性は無視されます。

USING SCHEMALOCATION HINTS
ソース XML 文書の中で XML スキーマ・ロケーション・ヒントによって指定されているスキーマに準拠しているかどうかについて、XML 文書の妥当性検査が実行されます。その XML 文書の中に schemaLocation 属性が指定されていない場合、妥当性検査は実行されません。**USING SCHEMALOCATION HINTS** 節が指定されているなら、すべての XML 列について XML Data Specifier (XDS) の SCH 属性は無視されます。

以下に示す **XMLVALIDATE** オプションの例を参照してください。

例

次に示すのは、ファイル `myfile.ixf` から SAMPLE データベースの STAFF 表に情報をインポートする方法の例です。

```
CALL SYSPROC.ADMIN_CMD
   ('IMPORT FROM /home/userid/data/myfile.ixf
   OF IXF MESSAGES ON SERVER INSERT INTO STAFF')
```

使用上の注意

IMPORT コマンドで使用されるすべてのパスは、サーバーのコーディネーター・ノード上の有効な完全修飾パスでなければなりません。

オプションとして **ALLOW WRITE ACCESS** または **COMMITCOUNT** が指定されている場合、インポート・ユーティリティーによってコミットが実行されます。その場合、タイプ 2 の接続では、ADMIN_CMD プロシージャーから、理由コード 1 の SQL30090N エラーが戻されます。

ADMIN_CMD プロシージャーからの結果セットの列に代入される値が、その列のデータ・タイプの最大値より大きい場合、そのデータ・タイプの最大値が代入され、警告メッセージ SQL1155W が戻されます。

インポート操作を開始する前に、すべての表操作が完了し、すべてのロックがペンディング解除になっていることを確認してください。これは、WITH HOLD でオープンされた、すべてのカーソルをクローズした後で COMMIT または ROLLBACK を発行することによって行われます。

インポート・ユーティリティーは、 SQL INSERT ステートメントを使用してターゲット表に行を追加します。このユーティリティーは、入力ファイル中の各行のデータにつき 1 つずつ INSERT ステートメントを発行します。 INSERT ステートメントが失敗した場合、以下の 2 通りの結果のいずれかになります。

- 後続の INSERT ステートメントが成功すると予測される場合には、警告メッセージがメッセージ・ファイルに書き込まれ、処理が継続されます。
- 後続の INSERT ステートメントが失敗すると予測され、データベースが損傷する可能性がある場合には、エラー・メッセージがメッセージ・ファイルに書き込まれ、処理が停止されます。

このユーティリティーは、**REPLACE** または **REPLACE_CREATE** 操作時に以前の行が削除された後、自動 COMMIT を実行します。したがって、表オブジェクトが切り捨てられた後、システムに障害が起こったり、アプリケーションがデータベース・マネージャーに割り込んだりすると、元のデータがすべて失われてしまいます。これらのオプションを使用する前に、元のデータがもはや必要ないことを確認してください。

CREATE、**REPLACE**、または **REPLACE_CREATE** 操作時にログが満杯になると、このユーティリティーは挿入されたレコードに対して自動 COMMIT を実行します。自動 COMMIT の後に、システムに障害が起こるか、またはアプリケーションがデータベース・マネージャーに割り込むと、部分的にデータの挿入された表はデータベース内に残ります。 **REPLACE** または **REPLACE_CREATE** オプションを使用してインポート操作全体をやり直すか、または正常にインポートされる行の数に設定した **RESTARTCOUNT** パラメーターを指定して **INSERT** を使用してください。

IMPORT コマンドからの更新は、常に IMPORT タスクの終わりにコミットされます。IMPORT コマンドの実行時に自動コミットを実行して、ロック・リストのサイズとアクティブ・ログ・スペースを減らすこともできます。IMPORT 処理中にアクティブ・ログがいっぱいになると、IMPORT コマンドはロールバックします。

- デフォルトでは、**INSERT** または **INSERT_UPDATE** オプションについては自動コミットは実行されません。しかし、**COMMITCOUNT** パラメーターがゼロでない場合は実行されます。
- 以下のいずれかの条件が真であると、オフライン・インポートでは自動の COMMIT は実行されません。
  - ターゲットは表ではなくビューである。
  - コンパウンド挿入を使用している。
  - バッファー挿入を使用している。
- デフォルトでは、オンライン・インポートは自動コミットを実行して、アクティブ・ログ・スペースとロック・リストを両方とも解放します。自動コミットが実行されないのは、ゼロの **COMMITCOUNT** 値を指定した場合のみです。

インポート・ユーティリティーが COMMIT を実行するたびに、 2 つのメッセージがメッセージ・ファイルに書き込まれます。一方は、コミットされるレコードの数を示し、もう一方は、 COMMIT の成功後に書き込まれます。障害の後にインポート操作を再開するときには、スキップするレコードの数 (最後の正常なコミットから判別される) を指定してください。

インポート・ユーティリティーでは、多少の非互換性問題がある入力データは受け入れられます (例えば、文字データは埋め込みまたは切り捨てを用いてインポートできます。数値データは異なる数値データ・タイプを用いてインポートできます)。しかし、大きな非互換性問題のあるデータは受け入れられません。

オブジェクト表に何らかの従属 (それ自体への従属は除く) がある場合は、そのオブジェクト表を **REPLACE** または **REPLACE_CREATE** することはできません。また、オブジェクト・ビューの基本表に何らかの従属 (それ自体への従属を含む) がある場合は、そのオブジェクト・ビューを **REPLACE** または **REPLACE_CREATE** することはできません。そのような表またはビューを置換するには、以下のとおりに行ってください。

1. その表が親となっているすべての外部キーをドロップします。
2. インポート・ユーティリティーを実行します。
3. 表を変更して、外部キーを再作成します。

外部キーの再作成中にエラーが発生する場合、参照整合性を保守するためにデータを変更してください。

参照制約および外部キー定義は、PC/IXF ファイルから表を再作成する場合は保存されません。 (主キー定義は、データが前に `SELECT *` を使ってエクスポートされた場合、*保存されます*。)

リモート・データベースへのインポートでは、サーバーに、入力データ・ファイルのコピー、出力メッセージ・ファイル、およびデータベースのサイズ拡大を見込んだ十分なディスク・スペースが必要とされます。

インポート操作がリモート・データベースに対して実行され、出力メッセージ・ファイルが非常に長くなった (60 KB を超過) 場合、クライアントのユーザーに戻されるメッセージ・ファイルで、インポート操作中にメッセージが欠落する可能性があります。メッセージ情報の最初の 30 KB と最後の 30 KB は、常に保持されます。

PC/IXF ファイルのリモート・データベースへのインポートは、 PC/IXF ファイルがディスケットにあるときよりも、ハード・ディスクにあるときの方がより速く行うことができます。

データベース表または階層が存在していないと、**ASC**、**DEL**、または **WSF** ファイル形式のデータをインポートできません。ただし、表が存在していない場合でも、IMPORT **CREATE** または IMPORT **REPLACE_CREATE** は、PC/IXF ファイルからデータをインポートする際に表を作成します。型付き表の場合、IMPORT **CREATE** はタイプ階層と表階層も作成することができます。

PC/IXF インポートは、データベース間でデータ (階層データも含む) を移動する場合に使用します。行区切り文字を含む文字データが区切り文字付き ASCII (DEL) ファイルにエクスポートされ、テキスト転送プログラムによって処理される場合、行区切り文字を含むフィールドは長さが変わることがあります。ソースとターゲットのデータベースが両方とも同じクライアントからアクセス可能である場合、ファイルのコピーというステップは必要ありません。

ASC および DEL ファイルのデータは、インポートを実行するクライアント・アプリケーションのコード・ページであると仮定されます。異なるコード・ページのデータをインポートする場合は、異なるコード・ページを使用することのできる PC/IXF ファイルをお勧めします。 PC/IXF ファイルとインポート・ユーティリティーが同じコード・ページである場合は、通常のアプリケーションの場合のように処理が行われます。それぞれのコード・ページが異なっており、**FORCEIN** オプショ

ンが指定されている場合、インポート・ユーティリティーは、PC/IXF ファイルのデータのコード・ページと、インポートを実行中のアプリケーションのコード・ページが同じであると見なします。この処理は、それら 2 つのコード・ページ用の変換テーブルが存在する場合であっても行われます。それぞれのコード・ページが異なっており、**FORCEIN** オプションが指定されておらず、変換テーブルが存在する場合、PC/IXF ファイルのすべてのデータは、そのファイルのコード・ページからアプリケーションのコード・ページに変換されます。それぞれのコード・ページが異なっており、**FORCEIN** オプションが指定されておらず、変換テーブルが存在しない場合、インポート操作は失敗します。これが該当するのは、AIX オペレーティング・システムの DB2 クライアント上の PC/IXF ファイルの場合だけです。

8 KB ページ上の表オブジェクトの量が 1012 列の制限に近い場合、PC/IXF データ・ファイルをインポートすると、SQL ステートメントの最大サイズを超過するため、DB2 はエラーを戻します。この状態が発生する可能性があるのは、列が CHAR、 VARCHAR、または CLOB タイプの場合だけです。 **DEL** または **ASC** ファイルのインポートには、この制限事項は適用されません。 PC/IXF ファイルを使って新しい表を作成している場合、別の方法として、db2look を使って表を作成した DDL ステートメントをダンプしてから、そのステートメント CLP から発行する、という方法があります。

DB2 Connect を使用して、DB2 for OS/390、DB2 for VM and VSE、および DB2 for OS/400 などの DRDA サーバーにデータをインポートすることができます。サポートされているのは、PC/IXF インポート (**INSERT** オプション) だけです。**RESTARTCOUNT** パラメーターもサポートされていますが、**COMMITCOUNT** パラメーターはサポートされていません。

型付き表で **CREATE** オプションを使用するときは、PC/IXF ファイルで定義されているすべての副表を作成してください。副表の定義は変更できません。型付き表で **CREATE** 以外のオプションを使用するときは、全探索順序リストによって全探索順序を指定できます。このため、全探索順序リストはエクスポート操作時に使用したものと一致する必要があります。PC/IXF ファイル形式の場合は、ターゲット副表の名前を指定して、ファイルに格納されている全探索順序を使用するだけです。

インポート・ユーティリティーは、以前 PC/IXF ファイルにエクスポートされた表をリカバリーする場合に使用できます。その表は、エクスポート時の状態に戻ります。

システム表、作成済み一時表、宣言済み一時表、またはサマリー表にデータをインポートすることはできません。

インポート・ユーティリティーを介してビューを作成することはできません。

それぞれのパーツが Windows システムから AIX システムにコピーされる、複数のパーツからなる PC/IXF ファイルのインポートがサポートされています。最初のファイルの名前だけ、IMPORT コマンドで指定する必要があります。例えば、`IMPORT FROM data.ixf OF IXF INSERT INTO TABLE1` のようにします。ファイル `data.002` などが、`data.ixf` と同じディレクトリーで使用できます。

Windows オペレーティング・システムの場合は、以下のとおりです。

- 論理分割された PC/IXF ファイルのインポートはサポートされていません。

- 不正な形式の PC/IXF または WSF ファイルのインポートは、サポートされていません。

内部形式のセキュリティー・ラベルには、改行文字が含まれている可能性があります。 DEL ファイル形式を使用してファイルをインポートする場合、それらの改行文字が間違って区切りと解釈される可能性があります。この問題が発生する場合は、IMPORT コマンドで delprioritychar ファイル・タイプ修飾子を指定することによって、区切り文字に関して以前に使用されていた古いデフォルト優先順位を使用してください。

フェデレーテッドに関する考慮事項

IMPORT コマンドで **INSERT**、**UPDATE**、または **INSERT_UPDATE** コマンド・パラメーターを使用するときには、関係するニックネームに対する CONTROL 特権があることを確認してください。インポート操作で使用したいニックネームが既に存在することを確認する必要があります。そのほかにも、IMPORT コマンド・パラメーターのセクションに記載されているようないくつかの制約事項に注意する必要があります。

ODBC などの一部のデータ・ソースでは、ニックネームへのインポートはサポートされていません。

結果セット情報

コマンドの実行状況は、CALL ステートメントからの結果である SQLCA で戻されます。実行が成功したなら、以下の結果セットによりコマンドから追加情報が戻されます。

表 42. IMPORT コマンドから戻される結果セット

列名	データ・タイプ	説明
ROWS_READ	BIGINT	インポート中にファイルから読み取られたレコードの数。
ROWS_SKIPPED	BIGINT	挿入または更新を開始する前にスキップしたレコードの数。
ROWS_INSERTED	BIGINT	ターゲット表に挿入された行数。
ROWS_UPDATED	BIGINT	インポートされたレコード (主キーの値が既に表内に存在するレコード) からの情報によって更新された、ターゲット表内の行数。
ROWS_REJECTED	BIGINT	インポートできなかったレコード数。
ROWS_COMMITTED	BIGINT	正常にインポートされ、データベースにコミット済みのレコード数。
MSG_RETRIEVAL	VARCHAR(512)	このユーティリティーによって作成されたメッセージを取り出すために使用する SQL ステートメント。以下に例を示します。 `SELECT SQLCODE, MSG FROM TABLE (SYSPROC.ADMIN_GET_MSGS ('1203498_txu')) AS MSG`

表 42. *IMPORT* コマンドから戻される結果セット (続き)

列名	データ・タイプ	説明
MSG_REMOVAL	VARCHAR(512)	このユーティリティーによって作成されたメッセージをクリーンアップするために使用する SQL ステートメント。以下に例を示します。 `CALL SYSPROC.ADMIN_REMOVE_MSGS ('1203498_txu')`

インポート・ユーティリティー用のファイル・タイプ修飾子

表 43. インポート・ユーティリティーの有効なファイル・タイプ修飾子: すべてのファイル形式

修飾子	説明
compound=*x*	*x* は 1 から 100 の数字です。非アトミック・コンパウンド SQL を使用してデータを挿入します。毎回 *x* 個のステートメントが試行されます。 この修飾子が指定され、トランザクション・ログに十分な大きさがない場合、インポート操作は失敗します。トランザクション・ログは、**COMMITCOUNT** によって指定された行数か、または **COMMITCOUNT** が指定されていない場合はデータ・ファイルの行数を入れる十分な大きさが必要です。したがって、トランザクション・ログのオーバーフローを避けるために、**COMMITCOUNT** オプションを指定することをお勧めします。 この修飾子は、**INSERT_UPDATE** モード、階層表、および修飾子 `usedefaults`、`identitymissing`、`identityignore`、`generatedmissing`、`generatedignore` とは互換性がありません。
generatedignore	この修飾子は、インポート・ユーティリティーに、すべての生成列のデータはデータ・ファイルに存在するが、それらを無視すべきことを知らせます。この結果として、生成列のすべての値は、このユーティリティーによって生成されます。この修飾子は、`generatedmissing` 修飾子と共に使用することはできません。
generatedmissing	この修飾子が指定されている場合、ユーティリティーは、生成列のデータが入力データ・ファイルに入っていない (NULL も入っていない) ものと見なし、行ごとに値を生成します。この修飾子は、`generatedignore` 修飾子と共に使用することはできません。
identityignore	この修飾子は、インポート・ユーティリティーに、ID 列のデータはデータ・ファイルに存在するが、それらを無視すべきことを知らせます。この結果として、すべて ID 値はこのユーティリティーによって生成されます。この動作は、GENERATED ALWAYS および GENERATED BY DEFAULT のどちらの ID 列の場合も同じです。つまり、GENERATED ALWAYS 列の場合には、リジェクトされる行はありません。この修飾子は、`identitymissing` 修飾子とともに使用することはできません。
identitymissing	この修飾子を指定すると、ユーティリティーは、ID 列のデータが入力データ・ファイルに入っていない (NULL も入っていない) ものと見なし、行ごとに値を生成します。この動作は、GENERATED ALWAYS および GENERATED BY DEFAULT のどちらの ID 列の場合も同じです。この修飾子は、`identityignore` 修飾子とともに使用することはできません。

表 43. インポート・ユーティリティーの有効なファイル・タイプ修飾子: すべてのファイル形式 (続き)

修飾子	説明
lobsinfile	*lob-path* には、LOB データの入ったファイルへのパスを指定します。 各パスには、データ・ファイル内で LOB ロケーション指定子 (LLS) によって示される 1 つ以上の LOB の入った、少なくとも 1 つのファイルが組み込まれます。 LLS は、LOB ファイル・パスに保管されるファイル内の LOB のロケーションのストリング表現です。 LLS の形式は、*filename.ext.nnn.mmm/* です。 *filename.ext* は LOB を収めたファイルの名前、 *nnn* はファイル内の LOB のオフセット (バイト単位)、 mmm は LOB の長さ (バイト単位) を表します。例えば、ストリング db2exp.001.123.456/ がデータ・ファイルに保管される場合、LOB はファイル db2exp.001 のオフセット 123 に位置し、456 バイト長です。 **LOBS FROM** 節は、「lobsinfile」修飾子が使用されているときの、LOB ファイルの場所を指定します。**LOBS FROM** 節によって、LOBSINFILE 動作が暗黙的にアクティブ化されます。 **LOBS FROM** 節は、データのインポート中に、IMPORT ユーティリティーに LOB ファイルを検索するためのパスのリストを送ります。 NULL LOB を指定するには、サイズに -1 と入力します。サイズを 0 と指定すると、長さが 0 の LOB として扱われます。長さが -1 の NULL LOB の場合、オフセットとファイル名は無視されます。例えば、NULL LOB の LLS は、db2exp.001.7.-1/ のようになります。
no_type_id	1 つの副表にインポートする場合にのみ有効です。これを使う場合として典型的な例は、REGULAR 表からデータをエクスポートした後、この修飾子を使ってインポート操作を呼び出してそのデータを単一の副表に変換する場合です。
nodefaults	ターゲット表の列に対応するソース列が明示的に指定されていない場合、その表列が NULL 不可能なら、デフォルト値はロードされません。このオプションを指定せず、あるターゲット表列のためのソース列が明示的に指定されていない場合、以下のいずれかになります。 • 列にデフォルト値を指定できる場合、そのデフォルト値がロードされます。 • 列が NULL 可能で、デフォルト値がその列に指定できない場合、NULL がロードされます。 • 列が NULL 不可能で、デフォルト値がその列に指定できない場合、エラーが戻され、ユーティリティーは処理を停止します。
norowwarnings	リジェクトされた行についてのすべての警告を抑止します。
rowchangetimestampignore	この修飾子は、インポート・ユーティリティーに、行変更タイム・スタンプの列のデータがデータ・ファイルに存在するが、それらを無視すべきことを知らせます。この結果、すべての ROW CHANGE TIMESTAMP はユーティリティーによって生成されます。この動作は、GENERATED ALWAYS 列でも GENERATED BY DEFAULT 列でも同じです。つまり、GENERATED ALWAYS 列の場合には、リジェクトされる行はありません。この修飾子は、rowchangetimestampmissing 修飾子とともに使用することはできません。
rowchangetimestampmissing	この修飾子を指定すると、ユーティリティーは、行変更タイム・スタンプ列のデータが入力データ・ファイルに入っていない (NULL も入っていない) ものと見なし、行ごとに値を生成します。この動作は、GENERATED ALWAYS 列でも GENERATED BY DEFAULT 列でも同じです。この修飾子は、rowchangetimestampignore 修飾子と共に使用することはできません。

表 43. インポート・ユーティリティーの有効なファイル・タイプ修飾子: すべてのファイル形式 (続き)

修飾子	説明
seclabelchar	入力ソース・ファイル内のセキュリティー・ラベルが、デフォルトのエンコードされた数値形式ではなく、セキュリティー・ラベル値のストリング形式であることを示します。 IMPORT は、各セキュリティー・ラベルをロードする際に内部形式に変換します。ストリングが適切な形式ではない場合、行はロードされず、警告 (SQLSTATE 01H53) が戻されます。ストリングが表を保護するセキュリティー・ポリシーの一部である有効なセキュリティー・ラベルを表していない場合、行はロードされず、警告 (SQLSTATE 01H53、SQLCODE SQL3243W) が戻されます。 seclabelname 修飾子が指定されている場合には、この修飾子は指定できません。指定すると、インポートは失敗して、エラー (SQLCODE SQL3525N) が戻されます。
seclabelname	入力ソース・ファイル内のセキュリティー・ラベルが、デフォルトのエンコードされた数値形式ではなく、名前によって指定されることを示します。 IMPORT はその名前を適切なセキュリティー・ラベル (存在する場合) に変換します。表を保護するセキュリティー・ポリシーで、指定された名前のセキュリティー・ラベルが存在しない場合、行はロードされず、警告 (SQLSTATE 01H53、SQLCODE SQL3244W) が戻されます。 seclabelchar 修飾子が指定されている場合には、この修飾子は指定できません。指定すると、インポートは失敗して、エラー (SQLCODE SQL3525N) が戻されます。 **注:** ファイル・タイプが ASC の場合、セキュリティー・ラベルの名前の後にスペースがあれば、それも名前の一部として解釈されます。これを回避するには、striptblanks ファイル・タイプ修飾子を使用してスペースが除去されるようにします。
usedefaults	ターゲット表の列のソース列が指定されているが、1 つ以上の行インスタンスのデータが入っていない場合は、デフォルト値がロードされます。欠落データの例を以下に示します。 • DEL ファイルの場合: 列の値として、2 つの連続した列区切り (,,) や、任意の数のスペースで分離した 2 つの連続する列区切り (, ,) が指定されている。 • DEL/ASC/WSF ファイルの場合: 列が不足している行、または元の指定には十分な長さでない行。 **注:** ASC ファイルの場合: NULL 列値は明示的に欠落していると見なされず、NULL 列値にはデフォルトが置換されません。NULL 列値は、数値、日付、およびタイム・スタンプ列の場合は全桁スペース文字、または任意のタイプの列の場合は NULL INDICATOR を使用して表現され、列が NULL であることを示します。 このオプションが指定されていない場合、行インスタンスのソース列にデータがないと、以下のいずれかの処理が行われます。 • DEL/ASC/WSF ファイルの場合: 列が NULL 可能であれば、NULL がロードされます。列が NULL 可能でない場合、ユーティリティーはその行をリジェクトします。

表 44. インポート・ユーティリティーで有効なファイル・タイプ修飾子: ASCII ファイル・フォーマット (ASC/DEL)

修飾子	説明
codepage=*x*	*x* は、ASCII 文字ストリングです。この値は、入力データ・セット内のデータのコード・ページとして解釈されます。インポート操作中に、文字データをこのコード・ページからアプリケーション・コード・ページに変換します。 以下の規則が適用されます。 • 純 DBCS (GRAPHIC)、混合 DBCS、および EUC では、区切り文字は x00 から x3F の範囲に制限されます。 • nullindchar には、標準の ASCII セットに組み込む (コード・ポイント x20 から x7F の範囲の) 記号を指定する必要があります。これは、ASCII 記号およびコード・ポイントを示します。 **注:** 1. codepage 修飾子を lobsinfile 修飾子と一緒に使用することはできません。 2. コード・ページがアプリケーションのコード・ページからデータベースのコード・ページに変換されているときにデータの拡張が発生する場合は、データは切り捨てられ、データの消失が発生する可能性があります。
dateformat="*x*"	*x* はソース・ファイルの日付のフォーマットです。[2] 有効な日付エレメントは以下のとおりです。 YYYY - 年 (0000 から 9999 の範囲の 4 桁の数) M - 月 (1 から 12 の範囲の 1 桁または 2 桁の数) MM - 月 (1 から 12 の範囲の 2 桁の数。 M とは相互に排他的) D - 日 (1 から 31 の範囲の 1 桁または 2 桁の数) DD - 日 (1 から 31 の範囲の 2 桁の数。 D とは相互に排他的) DDD - 元日から数えた日数 (001 から 366 の範囲の 3 桁の数。 他の日または月エレメントとは 相互に排他的) デフォルト値の 1 が、指定されない各エレメントに割り当てられます。日付形式の例を以下に示します。 "D-M-YYYY" "MM.DD.YYYY" "YYYYDDD"
implieddecimal	暗黙指定されている小数点の位置が列定義によって決定され、値の終わりにあるとは見なされなくなります。例えば、値 12345 は、12345.00 ではなく、123.45 として DECIMAL(8,2) 列にロードされます。

表 44. インポート・ユーティリティーで有効なファイル・タイプ修飾子: ASCII ファイル・フォーマット (ASC/DEL) (続き)

修飾子	説明
timeformat="*x*"	*x* はソース・ファイル内の時刻のフォーマットです。[2] 有効な時刻エレメントは以下のとおりです。 H - 時 (12 時間制の場合は 0 から 12、24 時間制では 0 から 24 の範囲の 1 桁または 2 桁の数) HH - 時 (12 時間制の場合は 0 から 12、24 時間制では 0 から 24 の範囲の 2 桁の数; H と相互に排他的) M - 分 (0 から 59 の範囲の 1 桁または 2 桁の数) MM - 分 (0 から 59 の範囲の 2 桁の数。M とは相互に排他的) S - 秒 (0 から 59 の範囲の 1 桁または 2 桁の数) SS - 秒 (0 から 59 の範囲の 2 桁の数。S と相互に排他的) SSSSS - 夜中の 12 時から数えた秒数 (00000 から 86399 の範囲の 5 桁の数。他の時刻エレメントとは相互に排他的) TT - 午前/午後の指定子 (AM または PM) 指定されない各エレメントには、デフォルト値の 0 が割り当てられます。時刻フォーマットの例を以下に示します。 "HH:MM:SS" "HH.MM TT" "SSSSS"

表 44. インポート・ユーティリティーで有効なファイル・タイプ修飾子: ASCII ファイル・フォーマット (ASC/DEL) (続き)

修飾子	説明

timestampformat="*x*"	*x* はソース・ファイル内のタイム・スタンプのフォーマットです。[2] 有効なタイム・スタンプ・エレメントは以下のとおりです。
	
	YYYY - 年 (0000 から 9999 の範囲の 4 桁の数)
	M - 月 (1 から 12 の範囲の 1 桁または 2 桁の数)
	MM - 月 (01 から 12 の 2 桁の数。
	M および MMM とは相互に排他的)
	MMM - 月 (大文字小文字を区別しない月名の 3 文字の省略形。
	M と MM とは相互に排他的)
	D - 日 (1 から 31 の範囲の 1 桁または 2 桁の数)
	DD - 日 (1 から 31 の範囲の 2 桁の数。D とは相互に排他的)
	DDD - 元日から数えた日数 (001 から 366 の範囲の 3 桁の数。
	他の日または月のエレメントとは相互に排他的)
	H - 時 (12 時間制の場合は 0 から 12、24 時間制では 0 から 24 の範囲の 1 桁または 2 桁の数。)
	HH - 時 (12 時間制の場合は 0 から 12、24 時間制では 0 から 24 の範囲の 1 桁または 2 桁の数。)
	H と相互に排他的)
	M - 分 (0 から 59 の範囲の 1 桁または 2 桁の数)
	MM - 分 (0 から 59 の範囲の 2 桁の数。
	M (分) とは相互に排他的)
	S - 秒 (0 から 59 の範囲の 1 桁または 2 桁の数)
	SS - 秒 (0 から 59 の範囲の 2 桁の数。
	S と相互に排他的)
	SSSSS - 夜中の 12 時から数えた秒数
	(00000 から 86399 の範囲の 5 桁の数。
	他の時刻エレメントとは相互に排他的)
	U (1 から 12 時)
	- 小数秒 (U のオカレンス数は、各桁を 0 から 9 の範囲として、桁数を表します)
	TT - 午前/午後の指定子 (AM または PM)
	
	YYYY、M、MM、D、DD、または DDD エレメントが指定されていない場合、デフォルト値の 1 が割り当てられます。 MMM エレメントが指定されていない場合、デフォルト値の「Jan」が割り当てられます。他のエレメントが指定されていない場合には、デフォルト値の 0 が割り当てられます。タイム・スタンプ・フォーマットの例を以下に示します。
	
	`"YYYY/MM/DD HH:MM:SS.UUUUUU"`
	
	MMM エレメントの有効な値は、「jan」、 「feb」、 「mar」、 「apr」、 「may」、 「jun」、 「jul」、 「aug」、 「sep」、 「oct」、 「nov」、および「dec」です。これらの値では、大/小文字は区別されません。
	
	次の例では、ユーザー定義の日時形式を指示するデータを、schedule という表にインポートする方法を示します。
	
	`db2 import from delfile2 of del`
	`modified by timestampformat="yyyy.mm.dd hh:mm tt"`
	`insert into schedule`

表 44. インポート・ユーティリティーで有効なファイル・タイプ修飾子: ASCII ファイル・フォーマット (ASC/DEL) (続き)

修飾子	説明
usegraphiccodepage	usegraphiccodepage が指定された場合、GRAPHIC または 2 バイト文字ラージ・オブジェクト (DBCLOB) データ・フィールドにインポートされるデータは、GRAPHIC コード・ページであると見なされます。データの残りは、文字コード・ページであると見なされます。 GRAPHIC コード・ページは、文字コード・ページと関連付けられます。 IMPORT は、codepage 修飾子 (指定されている場合)、または codepage 修飾子が指定されていない場合はアプリケーションのコード・ページを介して、文字コード・ページを決定します。 この修飾子は、リカバリーされている表に GRAPHIC データがある場合にのみ、ドロップ済み表のリカバリーによって生成された区切りデータ・ファイルとともに使用される必要があります。 **制約事項** EXPORT ユーティリティーで作成された DEL ファイルは、1 つのコード・ページのみでエンコードされたデータを含んでいるため、これらのファイルで usegraphiccodepage 修飾子を指定することはできません。 usegraphiccodepage 修飾子はまた、ファイル内の 2 バイト文字ラージ・オブジェクト (DBCLOB) には無視されます。
xmlchar	XML 文書が文字コード・ページでエンコードされていることを示します。 このオプションは、指定の文字コード・ページでエンコードされていてもエンコード宣言を含まない XML 文書を処理するために役立ちます。 各文書で、宣言タグが存在してエンコード属性が含まれる場合、そのエンコード方式は文字コード・ページと一致する必要があります。一致しない場合、その文書を含む行はリジェクトされます。文字コード・ページは codepage ファイル・タイプ修飾子で指定されている値であるか、または指定がない場合はアプリケーションのコード・ページであることに注意してください。デフォルトでは、文書は Unicode でエンコードされているか、またはエンコード属性のある宣言タグを含んでいます。
xmlgraphic	XML 文書が指定された GRAPHIC コード・ページでエンコードされていることを示します。 このオプションは、特定の GRAPHIC コード・ページでエンコードされていてもエンコード宣言を含まない XML 文書を処理するために役立ちます。 各文書で、宣言タグが存在してエンコード属性が含まれる場合、そのエンコード方式は GRAPHIC コード・ページと一致する必要があります。一致しない場合、その文書を含む行はリジェクトされます。 GRAPHIC コード・ページは codepage ファイル・タイプ修飾子で指定されている値のグラフィック・コンポーネントであるか、または指定がない場合はアプリケーションのコード・ページのグラフィック・コンポーネントであることに注意してください。デフォルトでは、文書は Unicode でエンコードされているか、またはエンコード属性のある宣言タグを含んでいます。 **注:** xmlgraphic 修飾子が IMPORT コマンドで指定されている場合、インポートされる XML 文書は、UTF-16 コード・ページでエンコードされていなければなりません。そうでない場合、XML 文書は構文解析エラーで拒否されるか、またはデータ破損のある状態で表にインポートされる場合があります。

表 45. インポート・ユーティリティーで有効なファイル・タイプ修飾子: ASC (区切り文字で区切られていない ASCII) ファイル・フォーマット

修飾子	説明
nochecklengths	nochecklengths が指定されていると、ソース・データの列定義がターゲット表の列のサイズを超えるものであっても、各行のインポートが試行されます。このような行が正常にインポートされるのは、コード・ページ変換でソース・データが縮小する場合です。例えば、ソースにある 4 バイトの EUC データがターゲットで 2 バイトの DBCS データに縮小すれば、 必要スペースは半分になります。このオプションが特に役立つのは、列の定義は不一致であるがソース・データが常に適合することが分かっている場合です。
nullindchar=*x*	*x* は、単一文字です。NULL 値を示す文字を *x* に変更します。*x* のデフォルト値は Y です。[3] 文字が 1 つの英字である場合を除いて、この修飾子は EBCDIC データ・ファイルで大文字小文字を区別します。例えば、NULL 標識文字を文字 N に指定した場合、 n も NULL 標識と認識されます。
reclen=*x*	*x* は、32 767 以下の整数です。各行ごとに *x* 個の文字が読み取られ、行の終わりを示すのに改行文字は使用されません。

striptblanks	データを可変長フィールドにロードする際に、後書きブランク・スペースを切り捨てます。このオプションを指定しない場合、ブランク・スペースはそのまま保持されます。
	
	次の例の場合、インポート・ユーティリティーは、 striptblanks によって後書きブランク・スペースを切り捨てます。
	
	`db2 import from myfile.asc of asc`
	`  modified by striptblanks`
	`  method l (1 10, 12 15) messages msgs.txt`
	`  insert into staff`
	
	このオプションは、 striptnulls と一緒に指定することはできません。これらは、相互に排他的なオプションです。このオプションは、廃止された t オプション (下位互換性のためだけにサポートされる) に代わるものです。

striptnulls	データを可変長フィールドにロードする際に、後書き NULL (0x00 文字) を切り捨てます。このオプションを指定しない場合、NULL はそのまま保持されます。 このオプションは、striptblanks と一緒に指定することはできません。これらは、相互に排他的なオプションです。このオプションは、廃止された padwithzero オプション (下位互換性のためだけにサポートされる) に代わるものです。

表 46. インポート・ユーティリティーで有効なファイル・タイプ修飾子: DEL (区切り文字で区切られている ASCII) ファイル・フォーマット

修飾子	説明

chardel*x*	*x* は単一文字のストリング区切り文字です。デフォルト値は二重引用符 (") です。指定した文字は、文字ストリングを囲むために、二重引用符の代わりに使用されます。 [34] 文字ストリング区切り文字として明示的に二重引用符を指定する場合、次のように指定します。
	`modified by chardel""`
	単一引用符 (') も、文字ストリングの区切り文字として指定できます。以下の例では、`chardel''` が指定されており、インポート・ユーティリティーは検出するすべての単一引用符 (') を文字ストリングの区切り文字として解釈します。
	`db2 "import from myfile.del of del`
	`modified by chardel''`
	`method p (1, 4) insert into staff (id, years)"`

coldel*x*	*x* は単一文字の列区切り文字です。デフォルト値はコンマ (,) です。指定した文字は、列の終わりを表すために、コンマの代わりに使用されます。[34]
	以下の例では、`coldel;` が指定されており、インポート・ユーティリティーは検出するすべてのセミコロン (;) を列の区切り文字として解釈します。
	`db2 import from myfile.del of del`
	`modified by coldel;`
	`messages msgs.txt insert into staff`

decplusblank	正符号文字。正の 10 進値の接頭部として、正符号 (+) ではなくブランク・スペースを使用します。デフォルトのアクションでは、正の 10 進数の前に正符号 (+) が付けられます。

decpt*x*	*x* は、小数点文字としてピリオドの代わりに使用される単一の置換文字です。デフォルト値はピリオド (.) です。指定した文字は、小数点文字としてピリオドの代わりに使用されます。[34]
	以下の例では、`decpt;` が指定されており、インポート・ユーティリティーは検出するすべてのセミコロン (;) を小数点として解釈します。
	`db2 "import from myfile.del of del`
	`modified by chardel''`
	`decpt; messages msgs.txt insert into staff"`

delprioritychar	区切り文字の現在のデフォルト優先順位は、(1) レコード区切り文字、(2) 区切り文字、(3) 列区切り文字です。この修飾子を使用すると、区切り文字の優先順位が (1) 区切り文字、(2) レコード区切り文字、(3) 列区切り文字に戻り、以前の優先順位に依存している既存のアプリケーションが保護されます。構文は以下のとおりです。
	`db2 import ... modified by delprioritychar ...`
	例えば、以下のような DEL データ・ファイルがあるとします。
	`"Smith, Joshua",4000,34.98<row delimiter>`
	`"Vincent,<row delimiter>, is a manager", ...`
	`... 4005,44.37<row delimiter>`
	`delprioritychar` 修飾子を指定しているので、このデータ・ファイルは、2 行だけになります。2 番目の <row delimiter> は 2 番目の行の最初のデータ列の一部と解釈されますが、1 番目と 3 番目の <row delimiter> は実レコードの区切り文字と解釈されます。この修飾子が指定されていない 場合、このデータ・ファイルでは 3 行になり、各行は <row delimiter> によって区切られます。

表 46. インポート・ユーティリティーで有効なファイル・タイプ修飾子: DEL (区切り文字で区切られている ASCII) ファイル・フォーマット (続き)

修飾子	説明
keepblanks	タイプが CHAR、VARCHAR、LONG VARCHAR、または CLOB の各フィールドの前後のブランクを保持します。このオプションを指定しないと、区切り文字で囲まれていないすべての前後のブランクは除去され、表のすべてのブランク・フィールドに NULL が挿入されます。
nochardel	インポート・ユーティリティーは、列区切り文字の間にあるすべてのバイトを列のデータの一部であると見なします。文字区切り文字は、列データの一部として構文解析されます。データが DB2 を使用してエクスポートされている場合は、このオプションを指定しないでください (エクスポート時に nochardel が指定されない限り)。これは、区切り文字を持たないベンダー・データ・ファイルをサポートするために用意されています。不適切に使用すると、データが損失または破壊される場合があります。 このオプションを `chardelx`、 `delprioritychar` または `nodoubledel` と一緒に指定することはできません。これらは、相互に排他的なオプションです。
nodoubledel	二重文字区切りの認識を抑止します。

表 47. インポート・ユーティリティーで有効なファイル・タイプ修飾子: IXF ファイル・フォーマット

修飾子	説明
forcein	コード・ページが不一致でもデータを受け入れ、コード・ページ間の変換を抑止するようにユーティリティーに指示します。 固定長ターゲット・フィールドに、そのデータが入るだけの十分な大きさがあるかどうかがチェックされます。 `nochecklengths` が指定されていると、チェックは実行されず、各行のインポートが試行されます。
indexixf	既存の表に現在定義されている索引をすべてドロップし、 PC/IXF ファイルの索引定義に基づいて新しい索引を作成するようにユーティリティーに指示します。このオプションを使用できるのは、表の内容を置換する場合だけです。ビューでは使用できません。また、*insert-column* が指定されている場合にも使用できません。
indexschema=*schema*	指定した *schema* を、索引作成時の索引名として使用します。 *schema* を指定しなかった場合 (しかしキーワード `indexschema` は指定した 場合) には、接続ユーザー ID が使用されます。このキーワードを指定しない場合、IXF ファイルのスキーマが使用されます。
nochecklengths	`nochecklengths` が指定されていると、ソース・データの列定義がターゲット表の列のサイズを超えるものであっても、各行のインポートが試行されます。このような行が正常にインポートされるのは、コード・ページ変換でソース・データが縮小する場合です。例えば、ソースにある 4 バイトの EUC データがターゲットで 2 バイトの DBCS データに縮小すれば、 必要スペースは半分になります。このオプションが特に役立つのは、列の定義は不一致であるがソース・データが常に適合することが分かっている場合です。
forcecreate	インポート操作中に SQL3311N が戻された後、欠落している可能性のある、または限られた情報で表が作成されることを指定します。

表 48. *codepage* および *usegraphiccodepage* 使用時の *IMPORT* 動作

codepage=N	usegraphiccodepage	IMPORT 動作
なし	なし	ファイル内のすべてのデータは、アプリケーション・コード・ページであると見なされます。
あり	なし	ファイル内のすべてのデータは、コード・ページ N であると見なされます。 **警告:** N が 1 バイト・コード・ページの場合、GRAPHIC データをデータベースにインポートすると、壊れます。
なし	あり	ファイル内の文字データは、アプリケーション・コード・ページであると見なされます。 GRAPHIC データは、アプリケーション GRAPHIC データのコード・ページであると見なされます。 アプリケーション・コード・ページが 1 バイトの場合は、すべてのデータはアプリケーション・コード・ページであると見なされます。 **警告:** アプリケーション・コード・ページが 1 バイトの場合、 GRAPHIC データは、データベースにたとえ GRAPHIC 列が収められていても、データベースにインポートされると壊れます。
あり	あり	文字データは、コード・ページ N であると見なされます。 GRAPHIC データは、N の GRAPHIC コード・ページであると見なされます。 N が 1 バイトまたは 2 バイト・コード・ページの場合は、すべてのデータは、コード・ページ N であると見なされます。 **警告:** N が 1 バイト・コード・ページの場合、GRAPHIC データをデータベースにインポートすると、壊れます。

注:

1. サポートされていないファイル・タイプを **MODIFIED BY** オプションで使用しようとしても、インポート・ユーティリティーは警告を出しません。この場合、インポート操作が失敗し、エラー・コードが戻されます。
2. 日付形式ストリングは必ず二重引用符で囲まなければなりません。フィールド区切り文字には、 a から z、A から Z、および 0 から 9 を使用することはできません。フィールド区切り文字は、区切り文字、または DEL ファイル・フォーマットのフィールド区切り文字と同じであってはなりません。エレメントの開始および終了位置が明らかな場合、フィールド区切り文字は任意指定です。あいまいさが生じうるのは、項目の長さが一定でない D、H、M、または S などのエレメントが使用されている場合です (修飾の仕方によって異なります)。

   タイム・スタンプ・フォーマットの場合、月の記述子と分の記述子のどちらも文字 M を使用するため、区別があいまいにならないように注意する必要があります。月のフィールドは、他の日付フィールドと隣接していなければなりま

せん。分フィールドは、他の時刻フィールドに隣接していなければなりません。あいまいなタイム・スタンプ形式の例を以下に示します。

```
"M" (could be a month, or a minute)
"M:M" (Which is which?)
"M:YYYY:M" (Both are interpreted as month.)
"S:M:YYYY" (adjacent to both a time value and a date value)
```

あいまいな場合、ユーティリティーはエラー・メッセージを報告し、操作は失敗します。

以下に、明確なタイム・スタンプ・フォーマットを示します。

```
"M:YYYY" (Month)
"S:M" (Minute)
"M:YYYY:S:M" (Month....Minute)
"M:H:YYYY:M:D" (Minute....Month)
```

二重引用符や円記号などの文字の前には、エスケープ文字 (例えば、¥) を付けなければなりません。

3. chardel、coldel、または decpt ファイル・タイプ修飾子に提供される文字値は、ソース・データのコード・ページで指定する必要があります。

   文字コード・ポイント (文字記号ではない) は、xJJ または 0xJJ という構文で指定することができます (JJ はコード・ポイントの 16 進表記)。例えば、列区切りとして # 文字を指定するには、以下のいずれかを使用します。

   ```
   ... modified by coldel# ...
   ... modified by coldel0x23 ...
   ... modified by coldelX23 ...
   ```

4. *データ移動のための区切り文字の制約事項*に、区切り文字の指定変更として使用できる文字に適用される制限のリストが示されています。
5. 以下のファイル・タイプ修飾子は、ニックネームにインポートするときには使用できません。
   - indexixf
   - indexschema
   - dldelfiletype
   - nodefaults
   - usedefaults
   - no_type_idfiletype
   - generatedignore
   - generatedmissing
   - identityignore
   - identitymissing
   - lobsinfile
6. XML 列では、WSF ファイル形式はサポートされていません。また、このファイル・フォーマットのサポートは推奨されておらず、今後のリリースで除去される可能性があります。サポートが除去される前に、WSF ファイルの代わりに、サポートされているファイル・フォーマットの使用を開始することを推奨します。
7. **CREATE** モードは、XML 列ではサポートされません。

8. すべての XML データは、メイン・データ・ファイルとは別の XML ファイル内に存在する必要があります。 XML Data Specifier (XDS) (または NULL 値) が、メイン・データ・ファイル内の XML 列ごとに存在する必要があります。
9. XMLCHAR または XMLGRAPHIC ファイル・タイプ修飾子が指定されていなければ、 XML 文書は Unicode 形式であるか、またはエンコード属性のある宣言タグを含むと想定されます。
10. 整形式でない文書が含まれている行はリジェクトされます。
11. **XMLVALIDATE** オプションが指定されている場合、対応するスキーマに対して正常に検証された文書は、挿入される際にスキーマ情報がアノテーションとして付加されます。対応するスキーマに対する妥当性検査が失敗した文書を含む行は、リジェクトされます。妥当性検査を正常に行うためには、インポートを起動するユーザーの保持する特権に、次の 1 つ以上が含まれている必要があります。
    - DBADM 権限
    - 妥当性検査に使用する XML スキーマに対する USAGE 特権
12. 暗黙的に隠された row change timestamp 列を含む表にインポートする場合、その列の暗黙的に隠されたプロパティーは反映されません。そのため、列のデータがインポートするデータに含まれておらず、明示的な列リストが存在しない場合には、rowchangetimestampmissing ファイル・タイプ修飾子を import コマンドで指定する必要があります。

INITIALIZE TAPE コマンド (ADMIN_CMD プロシージャーを使用)

ストリーミング磁気テープ装置へのバックアップおよびリストア操作のためにテープを初期化します。このコマンドは Windows オペレーティング・システムでのみサポートされています。

許可

以下のいずれかです。

- SYSADM
- SYSCTRL
- SYSMAINT

必要な接続

データベース

コマンド構文

コマンド・パラメーター

ON *device*

有効なテープ装置名を指定します。デフォルト値は ¥¥.¥TAPE0 です。装置の指定は、サーバーに対する相対指定でなければなりません。

USING *blksize*

装置のブロック・サイズを指定します (バイト単位)。値が装置のブロック・サイズとしてサポートされている範囲内にあれば、装置は指定されたそのブロック・サイズを使用するよう初期化されます。

BACKUP DATABASE コマンドおよび RESTORE DATABASE コマンドで指定されるバッファー・サイズは、ここで指定されるブロック・サイズで割り切れなければなりません。

このパラメーターに値を指定しなかった場合、装置はデフォルトのブロック・サイズを使用するよう初期化されます。値ゼロを指定した場合は、装置は可変長のブロック・サイズを使用するよう初期化されます。装置が可変長のブロック・モードをサポートしていない場合は、エラーが戻されます。

テープへのバックアップの場合、現在、可変ブロック・サイズの使用はサポートされていません。そのオプションを使用する必要がある場合は、リカバリーが正常に実行されるように十分にテストしたプロシージャーが使用できるようになっていることを確認し、また可変ブロック・サイズを指定して作成されたバックアップ・イメージを使用してください。

可変ブロック・サイズを使用する場合、使用している磁気テープ装置の最大限度以下のバックアップ・バッファー・サイズを指定する必要があります。パフォーマンスを最適化するには、使用している装置のブロック・サイズの最大限度と等しい値をバッファー・サイズとして使用しなければなりません。

例

2048 バイトの値がその装置でサポートされているブロック・サイズの範囲内であれば、ブロック・サイズとして 2048 バイトを使用するよう、磁気テープ装置を初期化します。

```
CALL SYSPROC.ADMIN_CMD( 'initialize tape using 2048' )
```

使用上の注意

コマンドの実行状況は、CALL ステートメントからの結果である SQLCA で戻されます。

LOAD コマンド (ADMIN_CMD プロシージャーを使用)

データを DB2 表にロードします。

サーバー上に存在するデータは、ファイル、テープ、または名前付きパイプの形式にすることができます。 また、データは、現在接続されているデータベースまたは同一インスタンス下の別のデータベースに対して実行される照会から定義されるカーソルから、あるいはユーザー作成のスクリプトまたはアプリケーションを使用してロードできます。表の COMPRESS 属性が YES に設定されている場合、ロードされ

るデータは、表内にディクショナリーが既に存在するデータおよびデータベース・パーティションごとに圧縮の対象となります。これには表の XML ストレージ・オブジェクト内のデータが含まれます。

148 ページの『ロード・ユーティリティー用のファイル・タイプ修飾子』へのクイック・リンク。

制約事項

ロード・ユーティリティーでは、階層レベルのデータのロードはサポートされていません。ロード・ユーティリティーには、範囲クラスター表との互換性はありません。ロード・ユーティリティーでは、NOT LOGGED INITIALLY パラメーターを CREATE TABLE や ALTER TABLE ステートメントでサポートしていません。

有効範囲

このコマンドは、一度の要求で複数のデータベース・パーティションに対して発行できます。

許可

以下のいずれかです。

- DATAACCESS
- データベースに対する LOAD 権限と以下のもの
  - 表の INSERT 特権 (ロード・ユーティリティーが INSERT モード、TERMINATE モード、または RESTART モードで呼び出される場合 (TERMINATE モードは直前のロード挿入操作を終了するためのもので、RESTART モードは直前のロード挿入操作を再開するためのものです))
  - 表の INSERT および DELETE 特権 (ロード・ユーティリティーが REPLACE モード、 TERMINATE モード、または RESTART モードで呼び出される場合 (TERMINATE モードは直前のロード置換操作を終了するためのもので、RESTART モードは直前のロード置換操作を再開するためのものです))
  - 例外表の INSERT 特権 (例外表をロード操作の一部として使用する場合)。
- 保護された列を持つ表にデータをロードするには、セッション許可 ID が、表内のすべての保護列への書き込みアクセスを許可する LBAC 信用証明情報を持っていなければなりません。そうでない場合は、ロードが失敗してエラー (SQLSTATE 5U014) が戻されます。
- 保護された行を持つ表にデータをロードするには、セッション許可 ID が、以下の基準を満たすセキュリティー・ラベルを保持していなければなりません。
  - 表を保護しているセキュリティー・ポリシーの一部である
  - 書き込みアクセス、またはすべてのアクセスに関して、セッション許可 ID に付与された

  こうしたセキュリティー・ラベルをセッション許可 ID が保持していない場合は、ロードが失敗してエラー (SQLSTATE 5U014) が戻されます。このセキュリティー・ラベルは、セッション許可 ID の LBAC 信用証明情報が、データ内のロードされる行を保護するセキュリティー・ラベルにその許可 ID が書き込むことを許可しない場合に、その行を保護するために使用されます。ただし、表を保護しているセキュリティー・ポリシーが CREATE SECURITY POLICY ステート

メントの RESTRICT NOT AUTHORIZED WRITE SECURITY LABEL オプションを使用して作成されている場合は、その状況にはなりません。その場合は、ロードが失敗してエラー (SQLSTATE 42519) が戻されます。

- REPLACE オプションを指定する場合、セッション許可 ID は表をドロップできる権限を持っていなければなりません。
- **LOCK WITH FORCE** オプションが指定される場合、 SYSADM 権限が必要です。

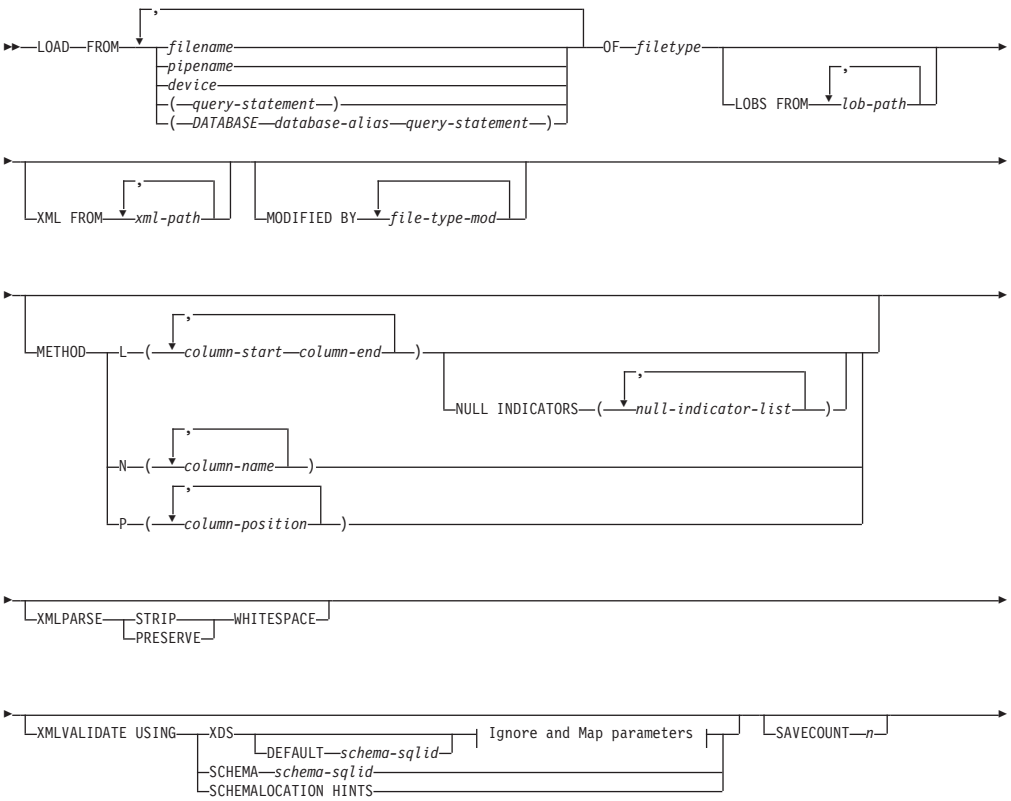
すべてのロード・プロセス (および一般にすべての DB2 サーバー・プロセス) はインスタンス所有者によって所有されており、それらのプロセスすべてにおいて、必要なファイルにアクセスするためにそのインスタンス所有者の ID を使用するため、インスタンス所有者には入力データ・ファイルに対する読み取りアクセス権が必要です。このコマンドをだれが呼び出すかには関係なく、それらの入力データ・ファイルをインスタンス所有者から読むことができなければなりません。

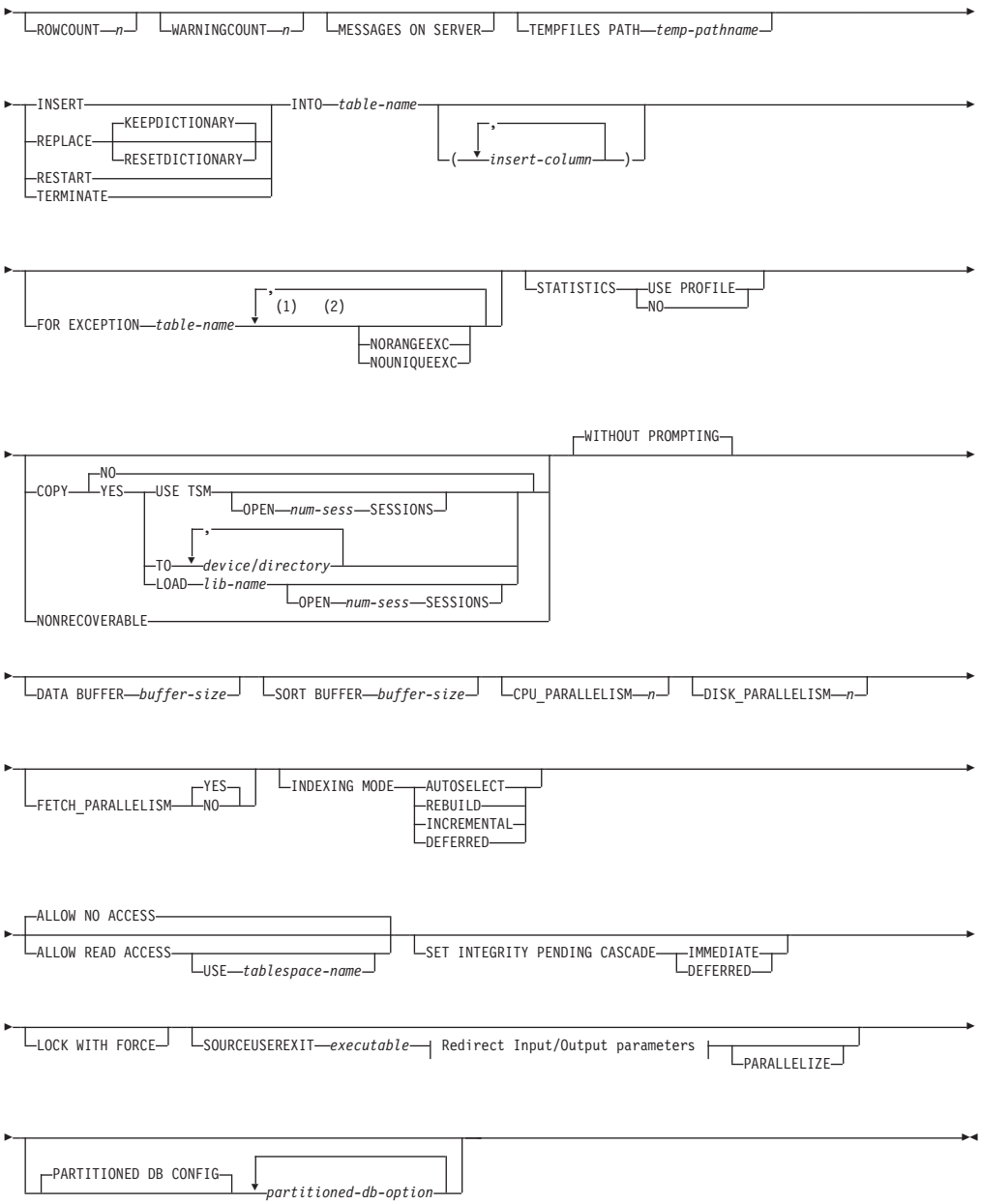
必要な接続

データベース。

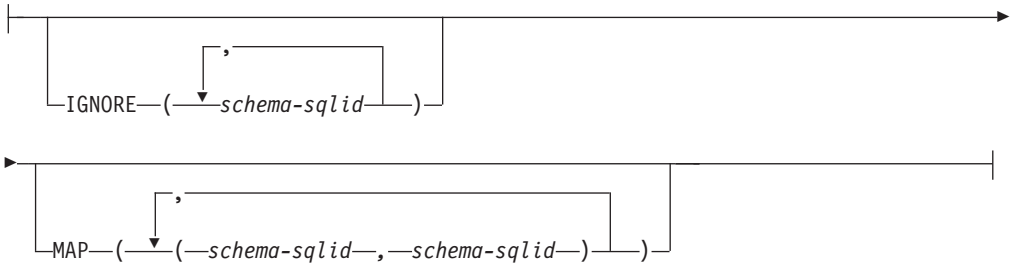
インスタンス。明示的なアタッチは必要ありません。データベースへの接続が確立されている場合には、ローカル・インスタンスへの暗黙的な接続が試みられます。

コマンド構文

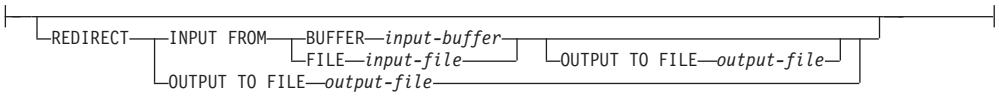




Ignore and Map parameters:



Redirect Input/Output parameters:



注:

1 これらのキーワードの出現順序は任意です。

2 これらのキーワードは、それぞれ 1 回だけ出現できます。

コマンド・パラメーター

FROM *filename* | *pipename* | *device(query-statement)* | (**DATABASE** *database-alias query-statement*)

ロードされるデータを含んだ SQL ステートメントを参照するファイル、パイプ、または装置、あるいは、SQL ステートメントそのものと、カーソルからロードする任意指定のソース・データベースを指定します。

query-statement オプションは、カーソルから LOAD する場合に使用します。これは、括弧で囲んだ照会ステートメントを 1 つだけ含み、VALUES、SELECT、または WITH で始めることができます。例えば、次のようになります。

```
LOAD FROM (SELECT * FROM T1) OF CURSOR INSERT INTO T2
```

DATABASE *database-alias* 節を括弧で囲まれた照会ステートメントの前に組み込むと、LOAD コマンドは *query-statement* を使用して、サーバーで定義されている *database-alias* 名で示される所定のデータベースからのデータのロードを試みます。これはサーバー上に存在するデータベースを指していなければなりません。また、これは、アプリケーションが現在接続されている別のデータベースです。 LOAD は、現在接続されているデータベースの明示的に指定されたユーザー ID とパスワードを使用して実行されることに注意してください (暗黙接続は、LOAD の失敗を引き起こします)。

入力ソースがファイル、パイプ、または装置である場合、サーバー上のコーディネーター・パーティションからそれにアクセスできなければなりません。

複数の名前を指定すると、それらは順番に処理されます。最後に指定した項目がテープ装置であり、ユーザーに対してテープを要求するプロンプトが出された場合、LOAD は失敗して ADMIN_CMD プロシージャーはエラーを戻します。

注:

1. 完全修飾パスのファイル名を使用し、それがサーバーに存在しなければなりません。
2. ADMIN_CMD プロシージャーを使用した EXPORT コマンドを使用してデータをファイルにエクスポートした場合、そのデータ・ファイルは fenced ユーザー ID によって所有されます。このファイルは通常、インスタンス所有者がアクセスすることはできません。 LOAD を CLP または ADMIN_CMD プロシージャーから実行するには、インスタンス所有

者 ID はデータ・ファイルにアクセスできなければならないので、データ・ファイルに対する読み取りアクセス権限をインスタンス所有者に付与する必要があります。

3. ファイルが物理的には分割されてはいるが論理的には 1 つのファイルである場合には、複数の IXF ファイルからのデータのロードがサポートされています。ファイルが論理的にも物理的にも分割されている場合は、サポートされていません。 (複数の物理ファイルがすべて 1 度の EXPORT コマンドの呼び出しで作成された場合、それらは論理的には 1 つであると見なされます。)
4. XML データをファイルからパーティション・データベース環境にある表にロードする場合、XML データ・ファイルはロードが実行されるすべてのデータベース・パーティションから読み取りアクセス可能でなければなりません。

OF *filetype*

データのフォーマットを指定します。

- ASC (区切りなし ASCII フォーマット)
- DEL (区切り付き ASCII フォーマット)
- IXF (統合交換フォーマット、PC バージョン) は、DB2 データベースによってのみ使用されるバイナリー・フォーマットです。
- CURSOR (SELECT または VALUES ステートメントに対して宣言されたカーソル)。

注: CURSOR ファイル・タイプを使用して XML データを分散データベース環境にある表にロードする場合、PARTITION_ONLY および LOAD_ONLY モードはサポートされません。

LOBS FROM *lob-path*

ロードする LOB 値が収められているデータ・ファイルへのパス。パスの最後はスラッシュでなければなりません。パスは完全修飾パスでなければならず、サーバー上のコーディネーター・パーティションからアクセスできなければなりません。 LOB データ・ファイルの名前は、メイン・データ・ファイル (ASC、DEL、または IXF) の、LOB 列にロードされる列内に保管されます。指定できるパスの最大数は 999 です。これによって、**LOBSINFILE** 動作が暗黙的にアクティブ化されます。

CURSOR ファイル・タイプと併せて指定された場合、このオプションは無視されます。

MODIFIED BY *file-type-mod*

ファイル・タイプ修飾子オプションを指定します。 148 ページの『ロード・ユーティリティー用のファイル・タイプ修飾子』を参照してください。

METHOD

L データのロードを開始する列および終了する列の番号を指定します。列の番号は、データの行の先頭からのバイト単位のオフセットです。この番号は 1 から始まります。この方式は、ASC ファイルの場合にのみ使用することができ、そのファイル・タイプでは唯一の有効な方式です。

NULL INDICATORS *null-indicator-list*

このオプションは、**METHOD L** パラメーターを指定した場合だけ使用できます (つまり、入力ファイルが ASC ファイルの場合)。 NULL 標識リストは、コンマで区切られた正の整数のリストで、各 NULL 標識フィールドの列の番号を指定します。列の番号は、データの行の先頭からのバイト単位の、各 NULL 標識フィールドのオフセットです。 NULL 標識リストには、**METHOD L** パラメーターで定義された各データ・フィールドに対する 1 つの項目がなければなりません。列の番号がゼロであることは、対応するデータ・フィールド内に必ずデータがあることを示します。

NULL 標識列中の `Y` の値は、その列データが NULL であることを指定します。 NULL 標識列に `Y` *以外* の文字を指定した場合は、列データが NULL ではなく、**METHOD L** オプションで指定された列データがロードされることを指定することになります。

NULL 標識文字は、**MODIFIED BY** オプションを使用して変更できます。

N ロードするデータ・ファイルの中の列の名前を指定します。これらの列名の大文字小文字の区別は、システム・カタログ内の対応する名前の大文字小文字の区別と一致しなければなりません。 NULL 可能ではない各表の列には、**METHOD N** リスト内に対応する項目が必要です。例えば、データ・フィールドが F1、 F2、 F3、 F4、 F5、および F6 であり、表の列が C1 INT、C2 INT NOT NULL、C3 INT NOT NULL、および C4 INT の場合、 `method N (F2, F1, F4, F3)` は有効な要求ですが、 `method N (F2, F1)` は無効です。この方式は、ファイル・タイプ IXF または CURSOR の場合にのみ使用することができます。

P ロードする入力データ・フィールドのフィールド番号 (1 から始まる) を指定します。 NULL 可能ではない各表の列には、**METHOD P** リスト内に対応する項目が必要です。例えば、データ・フィールドが F1、 F2、 F3、 F4、 F5、および F6 であり、表の列が C1 INT、C2 INT NOT NULL、C3 INT NOT NULL、および C4 INT の場合、 `method P (2, 1, 4, 3)` は有効な要求ですが、 `method P (2, 1)` は無効です。この方式は、ファイル・タイプ IXF、DEL、または CURSOR の場合にのみ使用でき、 DEL ファイル・タイプに対して有効な唯一の方式です。

XML FROM *xml-path*

XML ファイルが含まれているパスを 1 つ以上指定します。 XDS は、メイン・データ・ファイル (ASC、DEL、または IXF) の、XML 列にロードされる列内に入れられます。

XMLPARSE

XML 文書の解析方法を指定します。このオプションが指定されていない場合、XML 文書の解析の動作は、CURRENT XMLPARSE OPTION 特殊レジスターの値によって決まります。

STRIP WHITESPACE
XML 文書の解析時に空白文字を除去することを指定します。

PRESERVE WHITESPACE
XML 文書の解析時に空白文字を除去しないことを指定します。

XMLVALIDATE
該当する場合に、XML 文書がスキーマに準拠しているかどうかの妥当性検査を実行することを指定します。

USING XDS
メイン・データ・ファイルの中で XML Data Specifier (XDS) によって指定されている XML スキーマに準拠しているかどうかについて、XML 文書の妥当性検査が実行されます。デフォルトでは、**USING XDS** 節によって **XMLVALIDATE** オプションが呼び出された場合、妥当性検査実行のために使用されるスキーマは、その XDS の SCH 属性によって決まります。 XDS の中で SCH 属性が指定されていない場合、**DEFAULT** 節によってデフォルト・スキーマが指定されているのでない限り、スキーマ妥当性検査は実行されません。

DEFAULT、**IGNORE**、および **MAP** 節を使用することにより、スキーマ決定の動作を変更することができます。これら 3 つの節はオプションであり、相互に適用されるのではなく XDS の指定に直接適用されます。例えば、**DEFAULT** 節で指定されているためにあるスキーマが選択された場合、それが **IGNORE** 節で指定されていたとしても無視されることはありません。同じように、**MAP** 節のペアの最初の部分で指定されているためにあるスキーマが選択された場合、それが別の **MAP** 節のペアの 2 番目の部分で指定されていたとしても再びマップされることはありません。

USING SCHEMA *schema-sqlid*
指定されている SQL ID の XML スキーマに準拠しているかどうかについて、XML 文書の妥当性検査が実行されます。この場合、すべての XML 列について XML Data Specifier (XDS) の SCH 属性は無視されます。

USING SCHEMALOCATION HINTS
ソース XML 文書の中で XML スキーマ・ロケーション・ヒントによって指定されているスキーマに準拠しているかどうかについて、XML 文書の妥当性検査が実行されます。その XML 文書の中に schemaLocation 属性が指定されていない場合、妥当性検査は実行されません。**USING SCHEMALOCATION HINTS** 節が指定されているなら、すべての XML 列について XML Data Specifier (XDS) の SCH 属性は無視されます。

以下に示す **XMLVALIDATE** オプションの例を参照してください。

IGNORE *schema-sqlid*
このオプションは、**USING XDS** パラメーターを指定した場合にのみ使用できます。**IGNORE** 節は、SCH 属性によって指定されていても無視するスキーマとして、1 つ以上のスキーマのリストを指定します。ロードする XML 文書の XML Data Specifier の中に SCH 属性が存在し、その SCH 属性に

よって指定されるスキーマが無視するスキーマ・リストに含まれている場合には、ロードするその XML 文書についてスキーマ妥当性検査は実行されません。

注:

あるスキーマが **IGNORE** 節の中で指定されている場合、**MAP** 節のスキーマ・ペアの左辺にそれを含めることはできません。

IGNORE 節は XDS にのみ適用されます。あるスキーマが **IGNORE** 節によって指定されていても、それが **MAP** 節によってマップされているなら、それ以降そのスキーマが無視されることはありません。

DEFAULT *schema-sqlid*

このオプションは、**USING XDS** パラメーターを指定した場合にのみ使用できます。**DEFAULT** 節で指定されたスキーマは、ロード対象 XML 文書の XML Data Specifier (XDS) に XML スキーマを指定する SCH 属性が含まれていない場合に、妥当性検査のために使用するスキーマとなります。

DEFAULT 節は、**IGNORE** 節および **MAP** 節よりも優先されます。 XDS が **DEFAULT** 節を満たすなら、**IGNORE** と **MAP** の指定は無視されます。

MAP *schema-sqlid*

このオプションは、**USING XDS** パラメーターを指定した場合にのみ使用できます。**MAP** 節は、ロードする各 XML 文書について XML Data Specifier (XDS) の SCH 属性によって指定されるスキーマの代わりに使用する代替スキーマを指定するのに使用します。 **MAP** 節には、それぞれがあるスキーマから別のスキーマへのマッピングを表すスキーマ・ペアを 1 つ以上列挙したリストを指定します。ペア中の最初のスキーマは、XDS 内の SCH 属性によって参照されるスキーマを表します。ペア中の 2 番目のスキーマは、スキーマ検証の実行で使用する必要のあるスキーマを表します。

あるスキーマが **MAP** 節のスキーマ・ペアの左辺で指定されている場合、**IGNORE** 節でさらにそれを指定することはできません。

スキーマ・ペアのマッピングが適用されたなら、その結果は最終的なものです。マッピング操作は推移的ではないため、選択されたスキーマが、それ以降に別のスキーマ・ペアのマッピングに適用されることはありません。

スキーマを複数回マップすることはできません。つまり、複数のペアの左辺に指定することはできません。

SAVECOUNT *n*

ロード・ユーティリティーが *n* 行ごとに整合点を取ることを指定します。この値はページ・カウントに変換され、エクステント・サイズのインターバルに切り上げられます。メッセージは整合点ごとに発行されるので、LOAD QUERY を使用してロード操作をモニターする場合には、このオプションを選択する必要があります。 *n* の値が十分な大きさでない場合、各整合点で実行される活動の同期化によってパフォーマンスに影響してしまいます。

デフォルト値はゼロですが、それは、必要がなければ整合点は確立されないことを意味します。

CURSOR ファイル・タイプと併せて指定された場合、または XML 列を含む表をロードする場合、このオプションは無視されます。

ROWCOUNT *n*
ロードするファイル内の物理レコードの数 *n* を指定します。ユーザーはファイル内の最初の *n* 個の行だけをロードできます。

WARNINGCOUNT *n*
n 個の警告後に、ロード操作を停止します。このパラメーターは、警告は予期されないが、正しいファイルと表が使用されていることを確認するのが望ましい場合に設定してください。ロード・ファイルまたはターゲット表が不適切に指定されると、ロード対象の行ごとにロード・ユーティリティーによって警告が生成され、このためにロードが失敗する可能性があります。 *n* がゼロの場合、またはこのオプションが指定されていない場合、何度警告が出されてもロード操作は続行します。警告のしきい値に達したためにロード操作が停止された場合でも、あらためて RESTART モードでロード操作を開始できます。ロード操作は、最後の整合点から自動的に続行します。または、入力ファイルの先頭から REPLACE モードであらためてロード操作を開始できます。

MESSAGES ON SERVER
LOAD コマンドによってサーバー上に作成されるメッセージ・ファイルを保存することを指定します。戻される結果セットには以下の 2 つの列が含まれます。 1 つは MSG_RETRIEVAL で、これはこの操作中に発生したすべての警告メッセージおよびエラー・メッセージを取り出すのに必要な SQL ステートメントです。もう 1 つは MSG_REMOVAL で、これはメッセージをクリーンアップするために必要な SQL ステートメントです。

この節が指定されていない場合は、ADMIN_CMD プロシージャーから呼び出し元に戻る時点でメッセージ・ファイルが削除されます。結果セット中の MSG_RETRIEVAL および MSG_REMOVAL 列の内容は NULL 値になります。

この節を使用するしないにかかわらず、fenced ユーザー ID は、**DB2_UTIL_MSGPATH** レジストリー変数で示されるディレクトリー下にファイルを作成できる権限を持っていなければなりません。

TEMPFILES PATH *temp-pathname*
ロード操作時に一時ファイルを作成する場合に使用するパスの名前を指定します。これはサーバー・データベース・パーティションに従って完全に修飾しなければなりません。

一時ファイルは、ファイル・システムのスペースを使用します。場合によっては、このスペースが相当必要になります。以下に示すのは、すべての一時ファイルにどの程度のファイル・システム・スペースを割り振るべきかの見積もりです。

- ロード・ユーティリティーが生成するメッセージごとに 136 バイト
- データ・ファイルに長フィールド・データまたは LOB が入っている場合は、15 KB のオーバーヘッド。 **INSERT** オプションを指定した場合で、

表の中に多量の長フィールドまたは LOB データが既にある場合には、この数値はこれよりもかなり大きくなる場合があります。

INSERT

ロード・ユーティリティーを実行できる 4 つのモードのうちの 1 つ。既存の表データを変更することなく、ロードされたデータを表に追加します。

REPLACE

ロード・ユーティリティーを実行できる 4 つのモードのうちの 1 つ。表から既存データをすべて削除し、ロードされたデータを挿入します。表定義および索引定義は変更されません。階層間でデータを移動する際にこのオプションを使用する場合は、階層全体に関係したデータだけが置き換えられます。副表は置き換えられません。

KEEPDICTIONARY

LOAD REPLACE 操作の後も、既存のコンプレッション・ディクショナリーを保持します。表の COMPRESS 属性が YES の場合、新規に置換されるデータは、ロードの呼び出し前に存在していたディクショナリーを使用して圧縮されるという影響を受けます。ディクショナリーが表の中に以前に存在していない場合、表の COMPRESS 属性が YES である限り、置換されて表に入れられるデータを使用して新規ディクショナリーが作成されます。この場合のコンプレッション・ディクショナリーを作成するために必要なデータ量は、ADC のポリシーの影響を受けます。このデータは圧縮解除された状態で表に取り込まれます。ディクショナリーが表にいったん挿入されると、ロード対象の残りのデータは、このディクショナリーを使用して圧縮されるという影響を受けます。これはデフォルト・パラメーターです。要約を以下の表 1 に示します。

次の例では、現在、古いディクショナリーが表の中にある場合にはそれを保持します。

```
CALL SYSPROC.ADMIN_CMD('load from staff.del of del replace
 keepdictionary into SAMPLE.STAFF statistics use profile
 data buffer 8')
```

表 49. LOAD REPLACE KEEPDICTIONARY

圧縮	表の行データ・ディクショナリーが存在する	XML ストレージ・オブジェクト・ディクショナリーが存在する[1]	コンプレッション・ディクショナリー	データ圧縮
YES	YES	YES	表の行データ・ディクショナリーおよび XML ディクショナリーを保存します。	ロードされるデータは圧縮の対象になります。
YES	YES	NO	表の行データ・ディクショナリーを保存し、新規 XML ディクショナリーを作成します。	ロードされる表の行データは圧縮の対象になります。 XML ディクショナリーが作成された後に、ロードされる残りの XML データは圧縮の対象になります。

表 49. LOAD REPLACE KEEPDICTIONARY (続き)

圧縮	表の行データ・ディクショナリーが存在する	XML ストレージ・オブジェクト・ディクショナリーが存在する[1]	コンプレッション・ディクショナリー	データ圧縮
YES	NO	YES	表の行データ・ディクショナリーを作成し、XML ディクショナリーを保存します。	表の行データ・ディクショナリーが作成された後に、ロードされる残りの表の行データは圧縮の対象になります。ロードされる XML データは圧縮の対象になります。
YES	NO	NO	新規の表の行データおよび XML ディクショナリーを作成します。	ディクショナリーが作成された後に、ロードされる残りのデータは圧縮の対象になります。
NO	YES	YES	表の行データ・ディクショナリーおよび XML ディクショナリーを保存します。	ロードされるデータは、圧縮されません。
NO	YES	NO	表の行データ・ディクショナリーを保存します。	ロードされるデータは、圧縮されません。
NO	NO	YES	表の行ディクショナリーには影響はありません。XML ディクショナリーを保存します。	ロードされるデータは、圧縮されません。
NO	NO	NO	影響なし。	ロードされるデータは、圧縮されません。

注:

1. コンプレッション・ディクショナリーは、XML 列が DB2 バージョン 9.7 以降の表に追加された場合または表が Online Table Move を使用してマイグレーションされた場合にのみ、表の XML ストレージ・オブジェクトに対して作成できます。

RESETDICTIONARY

このディレクティブは、LOAD REPLACE 処理に、表の COMPRESS 属性が YES の場合には表データ・オブジェクトの新規ディクショナリーを作成するように指示します。 COMPRESS 属性が NO で、ディクショナリーがすでに表にある場合、それは除去され、新規ディクショナリーは表に挿入されません。コンプレッション・ディクショナリーは 1 つのユーザー・レコードのみを使用して作成できます。ロードするデータ・セットのサイズがゼロの場合は、既存のディクショナリーが存在していても、そのディクショナリーは保持されません。このディレクティブを使用してディクショナリーを作成するために必要なデータ量は、ADC のポリシーの影響を受けません。要約については、以下の表 2 を参照してください。

次の例は現在のディクショナリーをリセットし、新規のディクショナリーを作成します。

```
CALL SYSPROC.ADMIN_CMD('load from staff.del of del replace
 resetdictionary into SAMPLE.STAFF statistics use profile
 data buffer 8')
```

表 50. LOAD REPLACE RESETDICTIONARY

圧縮	表の行データ・ディクショナリーが存在する	XML ストレージ・オブジェクト・ディクショナリーが存在する[1]	コンプレッション・ディクショナリー	データ圧縮
YES	YES	YES	新規ディクショナリーの作成[2]。 DATA CAPTURE CHANGES オプションが CREATE TABLE または ALTER TABLE ステートメントで有効になっていると、現在の表の行データ・ディクショナリーが保持されます (これは*履歴コンプレッション・ディクショナリー* と呼ばれます)。	ディクショナリーが作成された後に、ロードされる残りのデータは圧縮の対象になります。
YES	YES	NO	新規ディクショナリーの作成[2]。 DATA CAPTURE CHANGES オプションが CREATE TABLE または ALTER TABLE ステートメントで有効になっていると、現在の表の行データ・ディクショナリーが保持されます (これは*履歴コンプレッション・ディクショナリー* と呼ばれます)。	ディクショナリーが作成された後に、ロードされる残りのデータは圧縮の対象になります。
YES	NO	YES	新規ディクショナリーの作成。	ディクショナリーが作成された後に、ロードされる残りのデータは圧縮の対象になります。
YES	NO	NO	新規ディクショナリーの作成。	ディクショナリーが作成された後に、ロードされる残りのデータは圧縮の対象になります。
NO	YES	YES	ディクショナリーを除去します。	ロードされるデータは、圧縮されません。
NO	YES	NO	表の行データ・ディクショナリーを除去します。	ロードされるデータは、圧縮されません。

表 50. LOAD REPLACE RESETDICTIONARY (続き)

圧縮	表の行データ・ディクショナリーが存在する	XML ストレージ・オブジェクト・ディクショナリーが存在する[1]	コンプレッション・ディクショナリー	データ圧縮
NO	NO	YES	XML ストレージ・オブジェクト・ディクショナリーを除去します。	ロードされるデータは、圧縮されません。
NO	NO	NO	影響なし。	すべての表データが圧縮されるわけではありません。

注:

1. コンプレッション・ディクショナリーは、XML 列が DB2 バージョン 9.7 以降の表に追加された場合または表が Online Table Move を使用してマイグレーションされた場合にのみ、表の XML ストレージ・オブジェクトに対して作成できます。
2. ディクショナリーが存在していて圧縮属性が有効であるものの、表パーティションにロードするレコードがない場合、新規ディクショナリーを作成することはできず、**RESETDICTIONARY** 操作では既存のディクショナリーは保持されません。

TERMINATE

ロード・ユーティリティーを実行できる 4 つのモードのうちの 1 つ。以前に割り込みを受けたロード操作を終了し、ロード操作が開始された時点まで操作をロールバックします。途中に整合点があっても通過します。その操作に関係する表スペースの状態は通常に戻され、すべての表オブジェクトの整合性が保たれます (索引オブジェクトが無効とマークされる場合がありますが、そのような場合には、次回のアクセス時に索引の再作成が自動的に行われます)。終了するロード操作が LOAD REPLACE の場合、その表は LOAD TERMINATE 操作完了後に空の表まで切り捨てられます。終了するロード操作が LOAD INSERT の場合、その表は LOAD TERMINATE 操作完了後も元のレコードをすべて保持します。ディクショナリー管理の要約については、以下の表 3 を参照してください。

LOAD **TERMINATE** オプションでは、表スペースの BACKUP PENDING 状態は解除されません。

RESTART

ロード・ユーティリティーを実行できる 4 つのモードのうちの 1 つ。以前に割り込みを受けたロード操作を再開します。ロード操作は、ロード、作成、または削除フェーズの最後の整合点から自動的に続行されます。ディクショナリー管理の要約を以下の表 4 に示します。

INTO *table-name*

データのロード先となるデータベース表を指定します。この表として、システム表、宣言済み一時表、または作成済み一時表は指定できません。別名、完全修飾、または非修飾の表名を指定できます。修飾された表名は、*schema.tablename* の形式になります。非修飾の表名を指定すると、その表は CURRENT SCHEMA で修飾されます。

insert-column
データの挿入先となる表の列を指定します。

ロード・ユーティリティーは、1 つ以上のスペースを使った名前の列を解析できません。例えば、次のようになります。

```
CALL SYSPROC.ADMIN_CMD('load from delfile1 of del noheader
   method P (1, 2, 3, 4, 5, 6, 7, 8, 9)
   insert into table1 (BLOB1, S2, I3, Int 4, I5, I6, DT7, I8, TM9)')
```

は、Int 4 列があるためエラーになります。これは、次のようにして二重引用符で列名を囲むことによって解決できます。

```
CALL SYSPROC.ADMIN_CMD('load from delfile1 of del noheader
   method P (1, 2, 3, 4, 5, 6, 7, 8, 9)
   insert into table1 (BLOB1, S2, I3, "Int 4", I5, I6, DT7, I8, TM9)')
```

FOR EXCEPTION *table-name*
エラーが発生した行のコピー先となる例外表を指定します。ユニーク索引または主キー索引に違反した行がすべてコピーされます。非修飾の表名を指定すると、その表は CURRENT SCHEMA で修飾されます。

例外表に書き込まれる情報は、ダンプ・ファイルには書き込まれません 。パーティション・データベース環境では、ロードする表を定義されたデータベース・パーティションの例外表を定義する必要があります。一方、ダンプ・ファイルには、無効であるか構文エラーであるためにロードできない行が入ります。

XML データのロード時に、**FOR EXCEPTION** 節を使用したロード例外表の指定は、以下の状況ではサポートされません。

- ラベル・ベースのアクセス制御 (LBAC) を使用する場合。
- パーティション表にデータをロードする場合。

NORANGEEXC
範囲違反のためにリジェクトされた行は、例外表に挿入しないことを指定します。

NOUNIQUEEXC
ユニーク制約に違反しているためにリジェクトされた行は、例外表に挿入しないことを指定します。

STATISTICS USE PROFILE
この表で定義されているプロファイルに従ってロード中に統計を収集するようロード操作に指示します。そのプロファイルは、ロードの実行前に作成されていなければなりません。そのプロファイルは、RUNSTATS コマンドで作成します。プロファイルが存在しない場合に、プロファイルに従って統計を収集するようロード操作に指示すると、警告メッセージが戻されて統計は収集されません。

ロードの際に、分散統計はタイプ XML の列については収集されません。

STATISTICS NO
統計データを収集せず、したがってカタログ内の統計データも変更しないことを指定します。これはデフォルトです。

COPY NO
順方向リカバリーが有効 (つまり、**logretain** または **userexit** がオン) になっていれば、表が存在している表スペースを BACKUP PENDING 状態にす

るよう指定します。 **COPY NO** オプションを使用する場合も、表スペース状態は LOAD IN PROGRESS になります。これは、一時的な状態であり、ロードが完了するか打ち切られると解除されます。表スペースのバックアップまたはデータベースのフルバックアップを実行しない限り、表スペースのどの表のデータも更新または削除できません。ただし、SELECT ステートメントを使用すれば、どの表のデータにもアクセス可能です。

リカバリー可能データベースでの COPY NO を指定した **LOAD** は、表スペースを BACKUP PENDING 状態のままにします。例えば、COPY NO を指定した **LOAD** および **INDEXING MODE DEFERRED** を実行すると、索引はリフレッシュが必要な状態になります。表での照会には、索引スキャンが必要なものがあり、索引がリフレッシュされるまで、成功しません。BACKUP PENDING 状態にある表スペース内に常駐する場合、索引はリフレッシュできません。この場合、表へのアクセスは、バックアップが行われるまで許可されません。索引リフレッシュは、索引が照会によってアクセスされたときに、データベースによって自動的に行われます。**COPY NO**、**COPY YES**、または **NONRECOVERABLE** のいずれも指定されておらず、データベースがリカバリー可能 (**logretain** または **logarchmeth1** が有効にされている)場合には、**COPY NO** がデフォルトです。

COPY YES
ロードするデータのコピーを保存することを指定します。フォワード・リカバリーが使用不可になっている場合、このオプションは無効です。

USE TSM
Tivoli Storage Manager (TSM) を使ってコピーを保管することを指定します。

OPEN *num-sess* **SESSIONS**
TSM またはベンダー製品とともに使用する入出力セッションの数です。デフォルト値は 1 です。

TO *device/directory*
コピー・イメージを作成する先の装置またはディレクトリーを指定します。

LOAD *lib-name*
使用するバックアップおよびリストア I/O ベンダー関数を含む共有ライブラリー (Windows オペレーティング・システムでは DLL) の名前。絶対パスで指定することができます。絶対パスを指定しない場合、デフォルトでユーザー出口プログラムの存在するパスになります。

NONRECOVERABLE
ロード・トランザクションがリカバリー不能としてマークされており、それ以降のロールフォワード・アクションによってそれをリカバリーさせることは不可能であることを指定します。ロールフォワード・ユーティリティーは、そのトランザクションをスキップし、データのロード先の表に「無効」としてマークします。さらに、ユーティリティーは、その表に対する後続のすべてのトランザクションを無視します。ロールフォワード操作が完了すると、そのような表は、ドロップするか、またはリカバリー不能なロード操作完了後のコミット・ポイントの後に取られたバックアップ (フルバックアップまたは表スペースのバックアップ) からリストアすることのみ可能です。

このオプションを使用すると、表スペースはロード操作後に BACKUP PENDING 状態になりません。また、ロード操作中にロードされたデータのコピーが作成される必要もなくなります。 **COPY NO**、**COPY YES**、または **NONRECOVERABLE** のいずれも指定されておらず、データベースがリカバリー可能でない (**logretain** または **logarchmeth1** が有効にされていない) 場合には、**NONRECOVERABLE** がデフォルトです。

WITHOUT PROMPTING
データ・ファイルのリストにロードするすべてのファイルを含め、しかもリストに入っている装置またはディレクトリーがロード操作全体で十分であるということを指定します。続きの入力ファイルが見つからなかったり、ロード操作が終了する前にコピー先がいっぱいになったりするとロード操作は失敗し、表は LOAD PENDING 状態のままになります。

これはデフォルトです。通常はユーザー介入を必要とするアクションでは、エラー・メッセージが戻されます。

DATA BUFFER *buffer-size*
ユーティリティー内でデータを転送するためのバッファー・スペースとして使用する 4 KB ページの数を指定します (並列処理の度合いには依存しません)。指定する値がアルゴリズム上の最小値より小さい場合、最小限必要なリソースが使用され、警告は戻されません。

このメモリーは、ユーティリティー・ヒープから直接に割り当てられ、そのサイズは **util_heap_sz** データベース構成パラメーターで修正可能です。バージョン 9.5 以降では、システムにさらに使用可能なメモリーがある場合、LOAD コマンドの DATA BUFFER オプションの値は、一時的に **util_heap_sz** を超えることができます。この場合、ユーティリティー・ヒープは、必要に応じて **database_memory** 限度に達するまで動的に増加します。このメモリーは、ロード操作が完了すると解放されます。

値が指定されていない場合、実行時にユーティリティーによって適切なデフォルトが計算されます。デフォルトは、ローダーのインスタンス生成時にユーティリティー・ヒープで使用できるフリー・スペースの割合と、表の一部の特性に基づいて決まります。

SORT BUFFER *buffer-size*
このオプションは、ロード操作時に **sortheap** データベース構成パラメーターをオーバーライドする値を指定します。これは、索引とともに表をロードする場合、また **INDEXING MODE** パラメーターが DEFERRED として指定されていない場合にのみ関係があります。指定された値は **sortheap** の値を超えることはできません。このパラメーターは、**sortheap** の値を変更せずに多くの索引を持つ表をロードする際に使用される、ソート・メモリーのスロットルで役に立ちます。これは、一般的な照会処理にも影響を与えます。

CPU_PARALLELISM *n*
表オブジェクトの作成時に、レコードの解析、変換、およびフォーマット設定のためにロード・ユーティリティーによって作成されるプロセスまたはスレッドの数を指定します。このパラメーターは、データベース・パーティションごとに実行するプロセスの数を活用するために設計されています。これは、事前にソートされたデータをロードする際に役立ちます (ソース・データのレコード順序が保持されるため)。このパラメーターの値が 0 の場合

や、このパラメーターを指定しなかった場合、ロード・ユーティリティーは、実行時に自動的に計算された適切なデフォルト値 (通常は使用できる CPU の数に基づく) を使用します。

注:

1. LOB または LONG VARCHAR フィールドのどちらかの入った表でこのパラメーターを使用する場合、システムの CPU の数またはユーザーが指定した値には関係なく、値は 1 になります。
2. **SAVECOUNT** パラメーターに指定する値が小さいと、データと表のメタデータの両方をフラッシュするために、ローダーがさらに多くの入出力操作を実行することになります。 **CPU_PARALLELISM** が 1 より大きいなら、フラッシュ操作は非同期になり、ローダーは CPU を活用できます。 **CPU_PARALLELISM** が 1 に設定されている場合、ローダーは整合点において入出力を待ちます。 **CPU_PARALLELISM** を 2 に設定し、**SAVECOUNT** を 10 000 に設定したロード操作は、 CPU が 1 つしかなくても、同じ操作で **CPU_PARALLELISM** を 1 に設定した場合より速く完了します。

DISK_PARALLELISM *n*

表スペース・コンテナーにデータを書き込むためにロード・ユーティリティーが作成するプロセスまたはスレッドの数を指定します。値を指定しない場合、ユーティリティーは表スペース・コンテナーの数と表の特性に基づいて、自動的に計算された適切なデフォルトを選択します。

FETCH_PARALLELISM YES | NO

DATABASE キーワードを使用してカーソルが宣言されていてカーソルからのロードを実行するとき、または API の sqlu_remotefetch_entry メディア項目を使用するとき、このオプションが YES に設定されていると、ロード・ユーティリティーは、リモート・データ・ソースからのフェッチの並列化を試みます (可能な場合)。 NO に設定されている場合、並列フェッチは行われません。デフォルト値は、YES です。詳しくは、『CURSOR ファイル・タイプを使用したデータの移動』を参照してください。

INDEXING MODE

ロード・ユーティリティーが索引を再作成するのか、それとも索引を増分で拡張するのかを指定します。有効な値は以下のとおりです。

AUTOSELECT

REBUILD モードと INCREMENTAL モードのいずれにするかを、ロード・ユーティリティーが自動的に決定します。決定は、ロードされるデータ量と索引ツリーの深さに基づいて行われます。索引ツリーの深さに関連する情報は索引オブジェクトに保管されています。この情報を設定するために、RUNSTATS は不要です。AUTOSELECT がデフォルトの索引付けモードです。

REBUILD

すべての索引が再作成されます。古い表データの索引キー部分も、追加される新しい表データの索引キー部分もすべてソートできるようにするため、ロード・ユーティリティーには十分なリソースが必要となります。

INCREMENTAL

索引に新しいデータが取り込まれて拡張します。このアプローチでは、索引のフリー・スペースが消費されます。このアプローチでは、新たに挿入されるレコードの索引キーを追加するためのソート・スペースだけがあれば十分です。この方式がサポートされるのは、索引オブジェクトが有効で、かつロード操作の開始時にアクセス可能な場合だけです (例えば、 DEFERRED モードが指定されたロード操作の直後では、この方式は無効です)。このモードを指定したものの、索引の状態などの理由でサポートされない場合は、警告が戻され、REBUILD モードでロード操作が続行されます。同様に、ロード作成フェーズでロード再開操作を開始した場合も、INCREMENTAL モードはサポートされません。

DEFERRED

このモードが指定されている場合、ロード・ユーティリティーは索引の作成を試みません。リフレッシュが必要であることを示すマークが索引に付けられます。ロード操作とは関係のないこのような索引に最初にアクセスするときは、再作成が強制的に実行されたり、データベースの再始動時に索引が再作成されたりする場合があります。 このアプローチでは、最も大きい索引のキー部分をすべて処理できるだけのソート・スペースが必要です。索引を作成するためにその後かかる合計時間は、REBUILD モードの場合よりも長くなります。したがって、この索引作成据え置きモードで複数のロード操作を実行する場合、最初の非ロード・アクセス時に索引を再作成できるようにしておくよりも、順序列内の最後のロード操作で索引の再作成を実行できるようにした方が (パフォーマンスの観点から) 賢明であるといえます。

据え置き索引作成がサポートされるのは、非ユニーク索引がある表だけです。そのため、ロード・フェーズで挿入される重複キーがロード操作後は永続的ではなくなります。

ALLOW NO ACCESS

ロードを使用すると、ロード中に、排他的アクセスのためにターゲット表がロックされます。ロード中、表の状態は LOAD IN PROGRESS に設定されます。 **ALLOW NO ACCESS** はデフォルトの動作です。これは、LOAD REPLACE で唯一有効なオプションです。

表に制約があると、表の状態は、LOAD IN PROGRESS の他に、SET INTEGRITY PENDING に設定されます。表の SET INTEGRITY PENDING 状態を解除するには、SET INTEGRITY ステートメントを使用する必要があります。

ALLOW READ ACCESS

ロードを使用すると、ターゲット表は共用モードでロックされます。表の状態は、LOAD IN PROGRESS および READ ACCESS の両方に設定されます。表のロード中、データの非デルタ部分にアクセスすることができます。つまり、表を読み取る側はロードの開始前に存在していたデータにはアクセスができ、ロード中のデータはロードが完了するまで利用できない、ということです。 ALLOW READ ACCESS ロードの LOAD TERMINATE または **LOAD RESTART** はこのオプションを使用できますが、ALLOW NO

ACCESS ロードの LOAD TERMINATE または **LOAD RESTART** はこのオプションを使用できません。また、ターゲット表上の索引が要再作成のマークが付けられると、このオプションは無効になります。

表に制約があると、表の状態は、LOAD IN PROGRESS、および READ ACCESS の他に、SET INTEGRITY PENDING に設定されます。ロードの終了時に、表の状態 LOAD IN PROGRESS は解除されますが、SET INTEGRITY PENDING および READ ACCESS はそのまま残ります。表の SET INTEGRITY PENDING を解除するには、SET INTEGRITY ステートメントを使用する必要があります。表が SET INTEGRITY PENDING および READ ACCESS の状態にある間、データの非デルタ部分には引き続き読み取りアクセスできますが、データの新しい (デルタ) 部分には、 SET INTEGRITY ステートメントが完了するまでアクセス不能のままになります。ユーザーは、SET INTEGRITY ステートメントを発行しないで、同じ表上で複数のロードを実行できます。ただし、元の (チェック済み) データは、 SET INTEGRITY ステートメントが発行されるまで可視のままです。

ALLOW READ ACCESS は、以下の修飾子もサポートします。

USE *tablespace-name*

索引が再作成される場合、表スペース *tablespace-name* に索引のシャドー・コピーが作成され、ロード終了時の INDEX COPY PHASE で、元の表スペース上にコピーされます。 SYSTEM TEMPORARY 表スペースのみ、このオプションを使用できます。指定されない場合、シャドー索引が、索引オブジェクトと同じ表スペース内に作成されます。シャドー・コピーが索引オブジェクトと同じ表スペース内に作成される場合、古い索引オブジェクトを介したシャドー索引オブジェクトのコピーは瞬時に終了します。シャドー・コピーが索引オブジェクトとは異なる表スペースにある場合、物理コピーが実行されます。これにはかなりの入出力および時間を要します。コピーは、表がオフラインの間、ロード終了時の INDEX COPY PHASE で行われます。

このオプションを使用しないと、シャドー索引は元の索引と同じ表スペースに作成されます。デフォルトでは、元の索引とシャドー索引の両方が同時に同じ表スペースに常駐するため、 1 つの表スペース内に両方の索引を保持するためのスペースが不足する場合があります。このオプションを使用すれば、索引用の十分な表スペースを保持できます。

ユーザーが **INDEXING MODE REBUILD** または **INDEXING MODE AUTOSELECT** を指定しない場合、このオプションは無視されます。このオプションは **INDEXING MODE AUTOSELECT** が選択され、ロードが索引を徐々に更新することを選択した場合にも無視されます。

SET INTEGRITY PENDING CASCADE

LOAD によって表が SET INTEGRITY ペンディング状態になる場合、**SET INTEGRITY PENDING CASCADE** オプションを使用することによって、ユーザーはロードされる表の SET INTEGRITY ペンディング状態を即

時にすべての下層 (下層外部キー表、下層即時マテリアライズ照会表、および下層即時ステージング表を含む) にカスケードするかどうか指定することができます。

IMMEDIATE
外部キー制約の SET INTEGRITY PENDING 状態が即時にすべての下層外部キー表、下層即時マテリアライズ照会表、および下層ステージング表に拡張されることを示します。 LOAD INSERT 操作の場合、**IMMEDIATE** オプションが指定されている場合でも、 SET INTEGRITY PENDING 状態は下層外部キー表に拡張されません。

後で (SET INTEGRITY ステートメントの IMMEDIATE CHECKED オプションを使用して) ロードされる表の制約違反をチェックする際、SET INTEGRITY PENDING READ ACCESS 状態だった下層外部キー表は、SET INTEGRITY PENDING NO ACCESS 状態になります。

DEFERRED
ロードされる表だけが、SET INTEGRITY PENDING 状態になることを示します。下層外部キー表、下層即時マテリアライズ照会表、および下層即時ステージング表は、未変更のままになります。

下層外部キー表は、(SET INTEGRITY ステートメントの IMMEDIATE CHECKED オプションを使用して) その親表の制約違反がチェックされるとき、後で暗黙的に SET INTEGRITY PENDING 状態になる場合があります。下層即時マテリアライズ照会表および下層即時ステージング表は、その基礎表のいずれかの保全性違反がチェックされる際、暗黙的に SET INTEGRITY PENDING 状態になります。照会のアクセス先が、指定された表ではなく、SET INTEGRITY PENDING 状態にない適格なマテリアライズ照会表である場合は、SET INTEGRITY PENDING 状態の表の照会が成功することがあります。下層表が SET INTEGRITY ペンディング状態になったことを示す警告 (SQLSTATE 01586) が出されます。この下層表がいつ SET INTEGRITY PENDING 状態になるかについては、「SQL リファレンス」にある SET INTEGRITY ステートメントの『注』の項を参照してください。

SET INTEGRITY PENDING CASCADE オプションが指定されない場合、次のようになります。

- ロードされる表だけが、SET INTEGRITY PENDING 状態になります。下層外部キー表、下層即時マテリアライズ照会表、および下層即時ステージング表の状態は未変更のままになり、後にロードされた表の制約違反がチェックされる際に、暗黙的に SET INTEGRITY PENDING 状態になる場合があります。

LOAD によってターゲット表が SET INTEGRITY PENDING 状態にならない場合、**SET INTEGRITY PENDING CASCADE** オプションは無視されます。

LOCK WITH FORCE
ユーティリティーはロード・プロセス中に、表ロックなどの様々なロックを獲得します。ロックを獲得する際、このオプションを使用すると、ロードは

待機することなく、またタイムアウトになることなく、ターゲット表に競合するロックを持つ他のアプリケーションを強制的にオフにします。システム・カタログ表に対する競合するロックを保持するアプリケーションは、ロード・ユーティリティーによって強制的にオフにされることはありません。強制されたアプリケーションは、ロールバックし、ロード・ユーティリティーが必要とするロックをリリースします。その後、ロード・ユーティリティーを続行できます。このオプションは、FORCE APPLICATIONS コマンドと同じ権限 (SYSADM または SYSCTRL) を必要とします。

ALLOW NO ACCESS ロードは、ロード操作の開始時に競合するロックを持つアプリケーションを強制的にロールバックさせる場合があります。ロードの開始時に、ユーティリティーは、表の照会または変更を試みているアプリケーションを強制的にロールバックさせる場合があります。

ALLOW READ ACCESS ロードは、ロード操作の開始時および終了時に競合するロックを持つアプリケーションを強制的にロールバックさせる場合があります。ロードの開始時に、ロード・ユーティリティーは、表の変更を試みているアプリケーションを強制的にロールバックさせる場合があります。ロード操作の終了時に、ロード・ユーティリティーは、表の照会または変更を試みているアプリケーションを強制的にロールバックさせる場合があります。

SOURCEUSEREXIT *executable*
このユーティリティーにデータを送るために呼び出される実行可能ファイル名を指定します。

REDIRECT

INPUT FROM

BUFFER *input-buffer*
input-buffer で指定されたバイトのストリームが、所定の実行可能ファイルを実行するプロセスの STDIN ファイル記述子に渡されます。

FILE *input-file*
このクライアント・サイドのファイルの内容が、所定の実行可能ファイルを実行するプロセスの STDIN ファイル記述子に渡されます。

OUTPUT TO

FILE *output-file*
STDOUT および STDERR ファイル記述子が、指定した完全修飾されたサーバー・サイドのファイルに取り込まれます。

PARALLELIZE
複数のユーザー出口プロセスを同時に呼び出すことによって、ロード・ユーティリティーへのデータ入力のスループットを高めます。このオプションは、複数パーティション・データベース環境でのみ適用でき、単一パーティション・データベース環境では無視されます。

詳細については、『カスタマイズしたアプリケーション (ユーザー出口) を使用したデータの移動』を参照してください。

PARTITIONED DB CONFIG *partitioned-db-option*
複数のデータベース・パーティションに分散した表へのロードの実行を可能にします。 **PARTITIONED DB CONFIG** パラメーターを使用すると、パーティション・データベース固有の構成オプションを指定することができます。 *partitioned-db-option* の値は、以下のいずれかになります。

```
PART_FILE_LOCATION x
OUTPUT_DBPARTNUMS x
PARTITIONING_DBPARTNUMS x
MODE x
MAX_NUM_PART_AGENTS x
ISOLATE_PART_ERRS x
STATUS_INTERVAL x
PORT_RANGE x
CHECK_TRUNCATION
MAP_FILE_INPUT x
MAP_FILE_OUTPUT x
TRACE x
NEWLINE
DISTFILE x
OMIT_HEADER
RUN_STAT_DBPARTNUM x
```

これらのオプションの詳しい説明は、『パーティション・データベース環境のロード構成オプション』で扱われています。

RESTARTCOUNT
予約済み。

USING *directory*
予約済み。

例

ファイルにある従業員表のデータに対して置換オプションを指定してロードを発行します。

```
CALL SYSPROC.ADMIN_CMD('LOAD FROM /home/theresax/tmp/emp_exp.dat
   OF DEL METHOD P (1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14)
   MESSAGES /home/theresax/tmp/emp_load.msg
   REPLACE INTO THERESAX.EMPLOYEE (EMPNO, FIRSTNME, MIDINIT, LASTNAME,
   WORKDEPT, PHONENO, HIREDATE, JOB, EDLEVEL, SEX, BIRTHDATE, SALARY,
   BONUS, COMM) COPY NO INDEXING MODE AUTOSELECT ISOLATE_PART_ERRS
   LOAD_ERRS_ONLY MODE PARTITION_AND_LOAD' )
```

以下は、単一パーティション・データベースからの出力例です。

```
Result set 1
--------------

ROWS_READ     ROWS_SKIPPED     ROWS_LOADED     ROWS_REJECTED     ...
---------...- ------------...- -----------...- -------------...- ...
          32                0              32                 0 ...

  1 record(s) selected.

Return Status = 0
```

単一パーティション・データベースからの出力 (続き)。

```
... ROWS_DELETED     ROWS_COMMITTED     MSG_RETRIEVAL
... -------------...- --------------...- ------------------------------...-
...                0                 32 SELECT SQLCODE, MSG_TEXT FROM
...                                        TABLE(SYSPROC.ADMIN_GET_MSGS(
...                                        '2203498_thx')) AS MSG
```

単一パーティション・データベースからの出力 (続き)。

```
... MSG_REMOVAL
... ----------------------------------------...-
... CALL SYSPROC.ADMIN_REMOVE_MSGS('2203498_thx')
...
```

注: この結果セットでは ROWS_PARTITIONED および NUM_AGENTINFO_ENTRIES 列も戻されますが、これらの列は、複数パーティション・データベースへのロード時のみデータが取り込まれるので、NULL に設定されます。

以下は、複数パーティション・データベースからの出力例です。

```
Result set 1
--------------

ROWS_READ     ROWS_REJECTED     ROWS_PARTITIONED     NUM_AGENTINFO_ENTRIES ...
----------...- -------------...- -----------------...- --------------------- ...
           32                 0                   32                     5 ...
                                                                           ...
                                                                           ...


  1 record(s) selected.
```

複数パーティション・データベースからの出力 (続き)。

```
... MSG_RETRIEVAL                          MSG_REMOVAL
... -----------------------------------...- ----------------------------...-
... SELECT DBPARTITIONNUM, AGENT_TYPE,     CALL SYSPROC.ADMIN_REMOVE_MSGS
...   SQLCODE, MSG_TEXT FROM TABLE             ('2203498_thx')
...   (SYSPROC.ADMIN_GET_MSGS
...   ('2203498_thx')) AS MSG
```

注: この結果セットでは ROWS_SKIPPED、ROWS_LOADED、ROWS_DELETED、および ROWS_COMMITTED 列も戻されますが、これらの列は、単一パーティション・データベースへのロード時のみデータが取り込まれるので、NULL に設定されます。

複数パーティション・データベースからの出力 (続き)。

```
Result set 2
--------------

DBPARTITIONNUM     SQLCODE     TABSTATE     AGENTTYPE
--------------...- -------...- --------...- ------------...-
                10           0 NORMAL       LOAD
                20           0 NORMAL       LOAD
                30           0 NORMAL       LOAD
                20           0 NORMAL       PARTITION
                10           0 NORMAL       PRE_PARTITION

1 record(s) selected.

Return Status = 0
```

XML 文書からデータをロードする例

XML データのロード

例 1

ユーザーは、表に挿入される文書を記述するために、XDS フィールドを持つデータ・ファイルを構成しました。内容は以下のとおりです。

```
1, "<XDS FIL=""file1.xml"" />"
2, "<XDS FIL='file2.xml' OFF='23' LEN='45' />"
```

最初の行では、XML 文書は file1.xml というファイルで示されます。区切り文字は二重引用符であり、二重引用符は XDS の内側にあるため、XDS 内にある二重引用符は二重になります。 2 番目の行では、XML 文書は file2.xml というファイルで示され、バイト・オフセット 23 で始まります。長さは 45 バイトです。

例 2

ユーザーは、XML 列の構文解析または妥当性検査オプションを指定しないでロード・コマンドを発行し、データは正常にロードされました。

```
LOAD
FROM data.del of DEL INSERT INTO mytable
```

CURSOR からの XML データのロード

データをカーソルからロードすることは、正規のリレーショナル列タイプを使用する場合と同じです。ユーザーは 2 つの表、T1 および T2 を持っており、それぞれは C1 という単一の XML 列で構成されます。 T1 から T2 にロードするには、ユーザーは最初に次のようにカーソルを宣言します。

```
DECLARE
X1 CURSOR FOR SELECT C1 FROM T1;
```

その後、ユーザーは次のようにカーソル・タイプを使用して LOAD を発行できます。

```
LOAD FROM X1 of
CURSOR INSERT INTO T2
```

カーソル・タイプに XML 固有の LOAD オプションを適用する操作は、ファイルからロードする場合と同じです。

使用上の注意

- データは、入力ファイル内に並んでいる順序でロードされます。特定の順序を希望する場合には、ロードが試行される前にデータをソートしてください。ソース・データの順序を保持する必要がなければ、**ANYORDER** ファイル・タイプ修飾子を使用することを考慮してください。この修飾子については、以下の『ロード・ユーティリティーのファイル・タイプ修飾子』セクションを参照してください。
- ロード・ユーティリティーは、既存の定義に基づいて索引を作成します。ユニーク・キーの重複を処理するのに、例外表が使用されます。ユーティリティーは、参照整合性を強制したり、制約検査を実行したり、ロードする表に従属するマテリアライズ照会表を更新したりすることはありません。参照制約またはチェック

制約を組み込まれた表は、SET INTEGRITY PENDING 状態になります。REFRESH IMMEDIATE として定義されているサマリー表、およびロードする表に依存するサマリー表もまた、 SET INTEGRITY PENDING 状態になります。表の SET INTEGRITY PENDING 状態を解除するには、SET INTEGRITY ステートメントを発行してください。ロード操作は、複製されたマテリアライズ照会表では実行できません。

- クラスタリング索引が表に存在する場合、ロード前にクラスタリング索引でデータをソートしてください。ただし、データはマルチディメンション・クラスタリング (MDC) 表にロードする前にソートする必要はありません。
- 保護された表へのロード時に例外表を指定すると、無効なセキュリティー・ラベルで保護されている行がその表に送られます。そのため、例外表にアクセスできるユーザーは、通常はアクセス権限のないデータにアクセスできてしまう可能性があります。セキュリティー・レベルを上げるために、誰に例外表アクセス権限を付与するかに注意し、行が修復されてロードする表にコピーされたら直ちにそれぞれの行を削除するとともに、使い終えた例外表は直ちにドロップしてください。
- 内部形式のセキュリティー・ラベルには、改行文字が含まれている可能性があります。 DEL ファイル形式を使用するファイルをロードする場合、この改行文字が区切り文字と間違えられることがあります。この問題が起きた場合は、LOAD コマンドで **delprioritychar** ファイル・タイプ修飾子を指定することによって、区切り文字に以前のデフォルト優先順位を使用してください。
- LOAD ユーティリティーは、操作の開始時に COMMIT ステートメントを発行しますが、これによってタイプ 2 接続の場合に、プロシージャーは SQL30090N、理由コード 1 を戻します。
- LOAD コマンドで使用されるすべてのパスは、サーバー・コーディネーター・パーティション上の有効な完全修飾パスでなければなりません。
- DECLARE CURSOR ステートメントの実行中に指定した DATABASE キーワードが CURSOR ファイル・タイプを使用してロードを実行する場合、現在接続されているデータベース (ロード用) の認証に使用されるユーザー ID およびパスワードが (DECLARE CURSOR ステートメントの DATABASE オプションによって指定された) ソース・データベースの認証に使用されます。ユーザー ID またはパスワードがロード・データベースの接続に指定されない場合、ソース・データベースのユーザー ID とパスワードは DECLARE CURSOR ステートメントの実行中に指定する必要があります。
- 個々のパートが Windows システムから AIX システムにコピーされる、複数パーツの PC/IXF ファイルのロードがサポートされます。すべてのファイルの名前は、LOAD コマンドで指定する必要があります。例えば、`LOAD FROM DATA.IXF, DATA.002 OF IXF INSERT INTO TABLE1` となります。論理分割された PC/IXF ファイルから Windows オペレーティング・システムにロードすることはサポートされていません。
- 失敗した LOAD を再始動する場合、その動作は、BUILD フェーズで索引用に REBUILD モードの使用が強制されるという点で、既存の動作に従います。
- データベース間での XML 文書のロードはサポートされておらず、エラー・メッセージ SQL1407N が戻されます。
- LOAD ユーティリティーでは、fenced プロシージャーを参照する列を含む表へのロードはサポートされていません。このような表に対して LOAD コマンドを発

行すると、エラー・メッセージ SQL1376N を受け取ります。この制約事項に対処するには、ルーチンを unfenced に再定義するか、インポート・ユーティリティーを使用することができます。

LOAD TERMINATE および LOAD RESTART ディクショナリー管理の要約

次の図表は、TERMINATE ディレクティブの下での **LOAD** 処理に関するコンプレッション・ディクショナリー管理の動作を要約しています。

表 51. LOAD TERMINATE ディクショナリー管理

表の COMPRESS 属性	LOAD 前に表の行データ・ディクショナリーが存在するかどうか	LOAD 前に XML ストレージ・オブジェクト・ディクショナリーが存在する[1]	TERMINATE: LOAD REPLACE KEEPDICTIONARY または LOAD INSERT	TERMINATE: LOAD REPLACE RESETDICTIONARY
YES	YES	YES	既存のディクショナリーを保持します。	どちらのディクショナリーも保持しません。[2]
YES	YES	NO	既存のディクショナリーを保持します。	何も保持しません。[2]
YES	NO	YES	既存のディクショナリーを保持します。	何も保持しません。
YES	NO	NO	何も保持しません。	何も保持しません。
NO	YES	YES	既存のディクショナリーを保持します。	何も保持しません。
NO	YES	NO	既存のディクショナリーを保持します。	何も保持しません。
NO	NO	YES	既存のディクショナリーを保持します。	何も保持しません。
NO	NO	NO	何も行われません。	何も行われません。

注:

1. コンプレッション・ディクショナリーは、XML 列が DB2 バージョン 9.7 以降の表に追加された場合または表が Online Table Move を使用してマイグレーションされた場合にのみ、表の XML ストレージ・オブジェクトに対して作成できます。
2. 表でデータ・キャプチャーが使用可能にされているという特殊な場合には、表の行データ・ディクショナリーが保持されます。

LOAD RESTART は、到達した最後の整合点まで表を切り捨てます。 LOAD RESTART 処理の一部として、最後の LOAD 整合点が取られたときにコンプレッション・ディクショナリーが表にあった場合、そのディクショナリーが表に存在します。その場合、LOAD RESTART では新規ディクショナリーは作成されません。考えられる条件の要約については、以下の表 4 を参照してください。

表 52. LOAD RESTART ディクショナリー管理

表の COMPRESS 属性	LOAD 整合点の前に表の行データ・ディクショナリーが存在するか[1]	最後の LOAD 前に XML ストレージ・オブジェクト・ディクショナリーが存在したか[2]	RESTART: LOAD REPLACE KEEPDICTIONARY または LOAD INSERT	RESTART: LOAD REPLACE RESETDICTIONARY
YES	YES	YES	既存のディクショナリーを保持します。	既存のディクショナリーを保持します。
YES	YES	NO	既存の表の行データ・ディクショナリーを保存し、ADC の制約を受ける XML ディクショナリーを作成します。	既存の表の行データ・ディクショナリーを保存し、XML ディクショナリーを作成します。
YES	NO	YES	ADC の制約を受ける表の行データ・ディクショナリーを作成します。既存の XML ディクショナリーを保持します。	表の行データ・ディクショナリーを作成します。既存の XML ディクショナリーを保持します。
YES	NO	NO	ADC の制約を受ける表の行データ・ディクショナリーおよび XML ディクショナリーを作成します。	表の行データ・ディクショナリーおよび XML ディクショナリーを作成します。
NO	YES	YES	既存のディクショナリーを保持します。	既存のディクショナリーを除去します。
NO	YES	NO	既存の表の行データ・ディクショナリーを保存します。	既存の表の行データ・ディクショナリーを除去します。
NO	NO	YES	既存の XML ディクショナリーを保持します。	既存の XML ディクショナリーを除去します。
NO	NO	NO	何も行われません。	何も行われません。

注:

1. XML データのロード時には **SAVECOUNT** オプションは無視されます。ロード・フェーズ中に失敗したロード操作は、操作の最初から再開されます。
2. コンプレッション・ディクショナリーは、XML 列が DB2 バージョン 9.7 以降の表に追加された場合または表が Online Table Move を使用してマイグレーションされた場合にのみ、表の XML ストレージ・オブジェクトに対して作成できます。

結果セット情報

コマンドの実行状況は、CALL ステートメントからの結果である SQLCA で戻されます。実行が成功すると、コマンドは追加情報を戻します。単一パーティション・データベースは 1 つの結果セットを戻し、複数パーティション・データベースは 2 つの結果セットを戻します。

• 146 ページの表 53: ロード操作の結果セット。

- 147 ページの表 54: 結果セット 2 には、複数パーティションでのロード操作における、データベース・パーティションごとの情報が含まれます。

表 53. LOAD コマンドによって戻される結果セット

列名	データ・タイプ	説明
ROWS_READ	BIGINT	ロード操作時の読み取り行数。
ROWS_SKIPPED	BIGINT	ロード操作開始前のスキップ行数。この情報は、単一パーティション・データベースの場合のみ戻されます。
ROWS_LOADED	BIGINT	ターゲット表にロードされた行の数。この情報は、単一パーティション・データベースの場合のみ戻されます。
ROWS_REJECTED	BIGINT	ターゲット表にロードできなかった行数。
ROWS_DELETED	BIGINT	ターゲット表にロードされなかった重複行の数。この情報は、単一パーティション・データベースの場合のみ戻されます。
ROWS_COMMITTED	BIGINT	処理された行の総数。この数は、ターゲット表に正常にロードされた行数と、スキップおよびリジェクトされた行数の合計です。この情報は、単一パーティション・データベースの場合のみ戻されます。
ROWS_PARTITIONED	BIGINT	すべてのデータベース分散エージェントによって分散された行数。この情報は、複数パーティション・データベースの場合のみ戻されます。
NUM_AGENTINFO_ENTRIES	BIGINT	複数パーティション・データベースの 2 番目の結果セットで戻される項目数。これは、ロード操作によって生成されたエージェント情報項目の数です。この情報は、複数パーティション・データベースの場合のみ戻されます。

MSG_RETRIEVAL	VARCHAR(512)	このユーティリティーによって作成されたメッセージを取り出すために使用する SQL ステートメント。例えば、次のようになります。
		`SELECT SQLCODE, MSG`
		`FROM TABLE`
		`(SYSPROC.ADMIN_GET_MSGS`
		`('2203498_thx')) AS MSG`
		
		この情報は、**MESSAGES ON SERVER** 節が指定された場合のみ戻されます。

表 53. LOAD コマンドによって戻される結果セット (続き)

列名	データ・タイプ	説明
MSG_REMOVAL	VARCHAR(512)	このユーティリティーによって作成されたメッセージをクリーンアップするために使用する SQL ステートメント。以下に例を示します。 `CALL SYSPROC.ADMIN_REMOVE_MSGS ('2203498_thx')` この情報は、**MESSAGES ON SERVER** 節が指定された場合のみ戻されます。

表 54. LOAD コマンドによって戻される結果セット 2 (複数パーティション・データベースのデータベース・パーティションごと)

列名	データ・タイプ	説明
DBPARTITIONNUM	SMALLINT	エージェントがロード操作を実行したデータベース・パーティション番号。
SQLCODE	INTEGER	ロード処理の結果の最終 SQLCODE。
TABSTATE	VARCHAR(20)	ロード操作完了後の表の状態。以下のいずれかです。 • LOADPENDING: ロードは完了していないが、パーティション上の表は LOAD PENDING 状態のままになっていることを示します。このデータベース・パーティションに対して、ロードの再始動または終了操作を行う必要があります。 • NORMAL: このデータベース・パーティションでのロードが正常に完了し、表の LOAD IN PROGRESS (または LOAD PENDING) 状態が解除されたことを示します。さらに制約処理が必要な場合に表はまだ SET INTEGRITY PENDING 状態である可能性がありますが、このインターフェースではその状態は報告されないので、注意してください。 • UNCHANGED: エラーのためにロードは完了していないが、表の状態はまだ変わっていないことを示します。このデータベース・パーティションに対してロードの再始動または終了操作を行う必要はありません。 **注:** 起こりうるすべての表の状態がこのインターフェースで戻されるわけではありません。

表 54. LOAD コマンドによって戻される結果セット 2 (複数パーティション・データベースのデータベース・パーティションごと) (続き)

列名	データ・タイプ	説明
AGENTTYPE	VARCHAR(20)	エージェント・タイプで、以下のいずれかです。 • FILE_TRANSFER • LOAD • LOAD_TO_FILE • PARTITIONING • PRE_PARTITIONING

ロード・ユーティリティー用のファイル・タイプ修飾子

表 55. ロード・ユーティリティーで有効なファイル・タイプ修飾子: すべてのファイル・フォーマット

修飾子	説明
anyorder	この修飾子は、 **cpu_parallelism** パラメーターと共に使用されます。ソース・データの順序を保持する必要がないことを指定します。そのため、SMP システムでは、パフォーマンスがかなり向上します。 **cpu_parallelism** の値が 1 になっていると、このオプションは無視されます。このオプションは、**SAVECOUNT** > 0 の場合はサポートされません。整合点の後のクラッシュ・リカバリーでは、順序のとおりにデータをロードする必要があるからです。
generatedignore	この修飾子を指定すると、ロード・ユーティリティーは、データ・ファイルに入っている、すべての生成済み列のデータを無視するようになります。この結果、すべての生成列の値はユーティリティーによって生成されます。この修飾子は、**generatedmissing** または **generatedoverride** 修飾子とともに使用することはできません。
generatedmissing	この修飾子が指定されている場合、ユーティリティーは、生成列のデータが入力データ・ファイルに入っていない (NULL も入っていない) ものと見なします。この結果、すべての生成列の値はユーティリティーによって生成されます。この修飾子は、**generatedignore** または **generatedoverride** 修飾子とともに使用することはできません。

表 55. ロード・ユーティリティーで有効なファイル・タイプ修飾子: すべてのファイル・フォーマット (続き)

修飾子	説明

generatedoverride	この修飾子は、(こうした列のタイプの通常の規則に反して) 表内のすべての生成列で、ユーザーのデータを受け入れるようにロード・ユーティリティーに指示します。別のデータベース・システムからデータをマイグレーションする場合や、ROLLFORWARD DATABASE コマンドの **RECOVER DROPPED TABLE** オプションを使用してリカバリーしたデータから表をロードする場合は、この修飾子を使用すると便利です。この修飾子を使用した場合、 NULL 不可の生成列でデータまたは NULL データの入っていない行はリジェクトされます (SQL3116W)。この修飾子が使用される場合、表は SET INTEGRITY PENDING 状態になります。ユーザー提供の値をチェックせずに表を SET INTEGRITY PENDING 状態から解放するには、ロード操作後に以下のコマンドを発行します。
	`SET INTEGRITY FOR table-name GENERATED COLUMN IMMEDIATE UNCHECKED`
	
	表の SET INTEGRITY PENDING 状態を解除し、ユーザー定義の値の検査を強制するには、ロード操作の後以下のコマンドを発行してください。
	`SET INTEGRITY FOR table-name IMMEDIATE CHECKED.`
	
	この修飾子が指定され、パーティション・キー、ディメンション・キー、または分散キーのいずれかに生成された列がある場合、LOAD コマンドが修飾子を **generatedignore** 自動的に変換し、ロードを進めます。この影響で、生成された列の値すべてが再生成されます。
	
	この修飾子は、**generatedmissing** または **generatedignore** 修飾子と共に使用することはできません。

identityignore	この修飾子はロード・ユーティリティーに対して、 ID 列のデータがデータ・ファイル内に存在するが、それらのデータは無視するべきものであることを通知します。この結果として、すべて ID 値はこのユーティリティーによって生成されます。この動作は、GENERATED ALWAYS および GENERATED BY DEFAULT のどちらの ID 列の場合も同じです。つまり、GENERATED ALWAYS 列の場合には、リジェクトされる行はありません。この修飾子は、**identitymissing** または **identityoverride** 修飾子とともに使用することはできません。
identitymissing	この修飾子を指定すると、ユーティリティーは、ID 列のデータが入力データ・ファイルに入っていない (NULL も入っていない) ものと見なし、行ごとに値を生成します。この動作は、GENERATED ALWAYS および GENERATED BY DEFAULT のどちらの ID 列の場合も同じです。この修飾子は、 **identityignore** または **identityoverride** 修飾子とともに使用することはできません。
identityoverride	この修飾子は、GENERATED ALWAYS として定義した ID 列が、ロードする表に存在している場合にのみ使用するべきです。この修飾子はユーティリティーに対し、そのような列に関して、明示的な非 NULL データを受け入れる (これらのタイプの ID 列に関する通常の規則に反する) ように指示します。表を GENERATED ALWAYS として定義しなければならない状況で別のデータベース・システムからデータをマイグレーションする場合や、 ROLLFORWARD DATABASE コマンドの **DROPPED TABLE RECOVERY** オプションを使用してリカバリーしたデータから表をロードする場合は、この修飾子を使用すると便利です。この修飾子を使用すると、ID 列にデータのない行や NULL データが入っている行は、リジェクトされます (SQL3116W)。この修飾子は、**identitymissing** または **identityignore** 修飾子とともに使用することはできません。このオプションが使用されていると、ロード・ユーティリティーは、表の ID 列内の値の固有性の保守または検証を行いません。

表 55. ロード・ユーティリティーで有効なファイル・タイプ修飾子: すべてのファイル・フォーマット (続き)

修飾子	説明
indexfreespace=*x*	*x* は、0 から 99 までの整数です。この値は、ロード操作で索引を再作成するときに各索引ページに残すフリー・スペースのパーセンテージとして解釈されます。**INDEXING MODE INCREMENTAL** を指定したロード操作では、このオプションが無視されます。ページの最初の項目は、制限なしで追加されます。それより後の項目は、フリー・スペースのパーセントしきい値内である場合に追加されます。デフォルト値は、CREATE INDEX の実行時に使用した値です。 この値は、CREATE INDEX ステートメントに指定された PCTFREE 値よりも優先して使用されます。 **indexfreespace** オプションの対象になるのは、索引のリーフ・ページだけです。
lobsinfile	*lob-path* には、LOB データの入ったファイルへのパスを指定します。ASC、DEL、または IXF ロード入力ファイルには、 LOB 列に LOB データが入っているファイルの名前が入っています。 ファイル・タイプが CURSOR の場合、このオプションはサポートされていません。 **lobsinfile** 修飾子を使用するときには、LOB ファイルの配置場所を **LOBS FROM** 節で指定します。**LOBS FROM** 節を指定すると、**lobsinfile** の動作が暗黙的にアクティブになります。**LOAD** ユーティリティーは、データをロードするときに、LOB ファイルを検索するためのパスのリストを LOBS FROM 節から受け取ります。 各パスには、データ・ファイル内で LOB ロケーション指定子 (LLS) によって示される 1 つ以上の LOB の入った、少なくとも 1 つのファイルが組み込まれます。 LLS は、LOB ファイル・パスに保管されるファイル内の LOB のロケーションのストリング表現です。 LLS の形式は、*filename*.ext.*nnn*.*mmm*/ です。ここで、*filename*.ext は、LOB が含まれているファイルの名前、*nnn* は、そのファイルに入っている LOB のオフセット (バイト単位)、*mmm* は、その LOB の長さ (バイト単位) です。例えば、ストリング db2exp.001.123.456/ がデータ・ファイルに保管される場合、 LOB はファイル db2exp.001 のオフセット 123 に位置し、456 バイト長です。 NULL LOB を指定するには、サイズに -1 と入力します。サイズを 0 と指定すると、長さが 0 の LOB として扱われます。長さが -1 の NULL LOB の場合、オフセットとファイル名は無視されます。例えば、NULL LOB の LLS は、db2exp.001.7.-1/ のようになります。
noheader	ヘッダー検査コードをスキップします (単一パーティション・データベースのパーティション・グループに存在する表へのロード操作にのみ適用します)。 単一パーティションのデータベース・パーティション・グループに存在する表に対してデフォルトの MPP ロード (モード PARTITION_AND_LOAD) が使用される場合、ファイルにはヘッダーが組み込まれないと想定されます。したがって、**noheader** 修飾子を指定する必要はありません。LOAD_ONLY モードが使用される場合、ファイルにはヘッダーが付いていると想定されます。 **noheader** 修飾子が必要になるのは、ヘッダーのないファイルを使用して LOAD_ONLY 操作を実行する場合に限られます。
norowwarnings	リジェクトされた行についてのすべての警告を抑止します。

表 55. ロード・ユーティリティーで有効なファイル・タイプ修飾子: すべてのファイル・フォーマット (続き)

修飾子	説明
pagefreespace=*x*	*x* は、0 から 100 までの整数です。この値は、各データ・ページ内でフリー・スペースとして残される部分のパーセンテージとして解釈されます。最小の行サイズのために指定値が無効な場合 (例えば、最低でも 3,000 バイトの長さが必要な行で、*x* 値が 50 になっている場合など) は、行が新しいページに配置されます。値として 100 を指定すると、各行が新しいページに配置されます。表の PCTFREE 値は、ページごとに指定されたフリー・スペースの量を決定します。ロード操作の **pagefreespace** 値または表の PCTFREE 値が設定されていないと、ユーティリティーはそれぞれのページで可能なかぎり多くのスペースを満たします。**pagefreespace** に設定されている値は、表で指定されている PCTFREE 値をオーバーライドします。
rowchangetimestampignore	この修飾子はロード・ユーティリティーに対して、ROW CHANGE TIMESTAMP 列のデータがデータ・ファイル内に存在するが、それらのデータは無視するべきものであることを通知します。この結果、すべての ROW CHANGE TIMESTAMP 列がユーティリティーによって生成されます。この動作は、GENERATED ALWAYS 列でも GENERATED BY DEFAULT 列でも同じです。つまり、GENERATED ALWAYS 列の場合には、リジェクトされる行はありません。この修飾子は、**rowchangetimestampmissing** または **rowchangetimestampoverride** 修飾子とともに使用することはできません。
rowchangetimestampmissing	この修飾子を指定すると、ユーティリティーは、行変更タイム・スタンプ列のデータが入力データ・ファイルに入っていない (NULL も入っていない) ものと見なし、行ごとに値を生成します。この動作は、GENERATED ALWAYS 列でも GENERATED BY DEFAULT 列でも同じです。この修飾子は、**rowchangetimestampignore** または **rowchangetimestampoverride** 修飾子とともに使用することはできません。
rowchangetimestampoverride	この修飾子は、GENERATED ALWAYS として定義した ROW CHANGE TIMESTAMP 列が、ロードする表に存在している場合にのみ使用するべきです。この修飾子はユーティリティーに対し、そのような列に関して、明示的な非 NULL データを受け入れる (これらのタイプの ROW CHANGE TIMESTAMP 列に関する通常の規則に反する) ように指示します。表を GENERATED ALWAYS として定義しなければならない状況で別のデータベース・システムからデータをマイグレーションする場合や、 ROLLFORWARD DATABASE コマンドの **DROPPED TABLE RECOVERY** オプションを使用してリカバリーしたデータから表をロードする場合は、この修飾子を使用すると便利です。この修飾子を使用すると、データが入っていない行や ROW CHANGE TIMESTAMP 列に対する NULL データはすべてリジェクトされます (SQL3116W)。この修飾子は、**rowchangetimestampmissing** または **rowchangetimestampignore** 修飾子とともに使用することはできません。このオプションが使用されていると、ロード・ユーティリティーは、表の ROW CHANGE TIMESTAMP 列内の値の固有性の保守または検証を行いません。

表 55. ロード・ユーティリティーで有効なファイル・タイプ修飾子: すべてのファイル・フォーマット (続き)

修飾子	説明

seclabelchar	入力ソース・ファイル内のセキュリティー・ラベルが、デフォルトのエンコードされた数値形式ではなく、セキュリティー・ラベル値のストリング形式であることを示します。 LOAD は、ロード時に各セキュリティー・ラベルを内部形式に変換します。ストリングが適切な形式ではない場合、行はロードされず、警告 (SQLSTATE 01H53、SQLCODE SQL3242W) が戻されます。ストリングが表を保護するセキュリティー・ポリシーの一部である有効なセキュリティー・ラベルを表していない場合、行はロードされず、警告 (SQLSTATE 01H53、SQLCODE SQL3243W) が戻されます。
	
	seclabelname 修飾子を指定した場合は、この修飾子を指定できません。同時に指定すると、ロードは失敗し、エラー (SQLCODE SQL3525N) が戻されます。
	
	単一の DB2SECURITYLABEL 列で構成される表がある場合、データ・ファイルは例えば次のようになります。
	
	`"CONFIDENTIAL:ALPHA:G2"`
	`"CONFIDENTIAL;SIGMA:G2"`
	`"TOP SECRET:ALPHA:G2"`
	
	このデータのロードまたはインポートでは、以下のように **seclabelchar** ファイル・タイプ修飾子を使用する必要があります。
	
	`LOAD FROM input.del OF DEL MODIFIED BY SECLABELCHAR INSERT INTO t1`

seclabelname	入力ソース・ファイル内のセキュリティー・ラベルが、デフォルトのエンコードされた数値形式ではなく、名前によって指定されることを示します。 LOAD は、その名前に対応する適切なセキュリティー・ラベルがあれば、その名前をそのセキュリティー・ラベルに変換します。表を保護するセキュリティー・ポリシーで、指定された名前のセキュリティー・ラベルが存在しない場合、行はロードされず、警告 (SQLSTATE 01H53、SQLCODE SQL3244W) が戻されます。
	
	seclabelchar 修飾子を指定した場合は、この修飾子を指定できません。同時に指定すると、ロードは失敗し、エラー (SQLCODE SQL3525N) が戻されます。
	
	単一の DB2SECURITYLABEL 列で構成される表がある場合、データ・ファイルは以下のようなセキュリティー・ラベル名で構成される可能性があります。
	
	`"LABEL1"`
	`"LABEL1"`
	`"LABEL2"`
	
	このデータのロードまたはインポートでは、以下のように **seclabelname** ファイル・タイプ修飾子を使用する必要があります。
	
	`LOAD FROM input.del OF DEL MODIFIED BY SECLABELNAME INSERT INTO t1`
	
	注: ファイル・タイプが ASC の場合、セキュリティー・ラベルの名前の後にスペースがあれば、それも名前の一部として解釈されます。これを回避するには、**striptblanks** ファイル・タイプ修飾子を使用してスペースが除去されるようにします。

表 55. ロード・ユーティリティーで有効なファイル・タイプ修飾子: すべてのファイル・フォーマット (続き)

修飾子	説明
totalfreespace=*x*	*x* は 0 以上の整数です。この値は表内の合計ページのうち、表の終わりにフリー・スペースとして追加される部分のパーセンテージと解釈されます。例えば、*x* が 20 で、データのロード後に表に 100 個のデータ・ページがある場合は、20 個の空ページが追加されます。その表のデータ・ページの合計数は 120 になります。データ・ページの総数は、表の索引ページの数には影響を与えません。このオプションは、索引オブジェクトには影響を与えません。このオプションを指定して 2 つのロードが行われる場合、 2 番目のロードは、最初のロードによって最後に付加された余分のスペースを再利用しません。
usedefaults	ターゲット表の列のソース列が指定されているが、1 つ以上の行インスタンスのデータが入っていない場合は、デフォルト値がロードされます。欠落データの例を以下に示します。 • DEL ファイルの場合: 列の値として、2 つの連続した列区切り (,,) や、任意の数のスペースで分離した 2 つの連続する列区切り (, ,) が指定されている。 • DEL/ASC/WSF ファイルの場合: 列が不足している行、または元の指定には十分な長さでない行。 ASC ファイルの場合: NULL 列値は明示的に欠落していると見なされず、NULL 列値にはデフォルトが置換されません。数値、日付、時刻、タイム・スタンプの列では、全桁スペース文字で NULL 列値を表記します。また、どのタイプの列でも、NULL INDICATOR を使用すれば、その列が NULL であることを示せます。 このオプションが指定されていない場合、行インスタンスのソース列にデータがないと、以下のいずれかの処理が行われます。 • DEL/ASC/WSF ファイルの場合: 列が NULL 可能であれば、NULL がロードされます。列が NULL 可能でない場合、ユーティリティーはその行をリジェクトします。

表 56. ロード・ユーティリティーで有効なファイル・タイプ修飾子: ASCII ファイル・フォーマット (ASC/DEL)

修飾子	説明
codepage=*x*	*x* は、ASCII 文字ストリングです。この値は、入力データ・セット内のデータのコード・ページとして解釈されます。ロード操作時に、文字データ (および文字内で指定された数値データ) は、このコード・ページからデータベースのコード・ページへ変換されます。 以下の規則が適用されます。 • DBCS のみ (GRAPHIC)、混合 DBCS、および EUC の場合、区切り文字の範囲は x00 から x3F に制限されます。 • EBCDIC コード・ページで指定された DEL データの場合、区切り文字は DBCS のシフトイン文字およびシフトアウト文字と一致しない場合があります。 • `nullindchar` では、標準の ASCII セットのコード・ポイント x20 から x7F の範囲に含まれているシンボルを指定する必要があります。これは、ASCII 記号およびコード・ポイントを示します。 EBCDIC データでは、コード・ポイントが異なるとしても、対応する記号を使用できます。 ファイル・タイプが CURSOR の場合、このオプションはサポートされていません。

表 56. ロード・ユーティリティーで有効なファイル・タイプ修飾子: ASCII ファイル・フォーマット (ASC/DEL) (続き)

修飾子	説明

dateformat=*"x"*	*x* はソース・ファイルの日付のフォーマットです。[1] 有効な日付エレメントは次のとおりです。
	
	`  YYYY   - 年 (0000 から 9999 の範囲の 4 桁の数)`
	`M      - 月 (1 から 12 の範囲の 1 桁または 2 桁の数)`
	`MM     - 月 (1 から 12 の範囲の 2 桁の数。`
	`  M とは相互に排他的)`
	`D      - 日 (1 から 31 の範囲の 1 桁または 2 桁の数)`
	`DD     - 日 (1 から 31 の範囲の 2 桁の数。`
	`  D とは相互に排他的)`
	`DDD    - 元日から数えた日数 (001 から 366 の範囲の 3 桁の数。`
	`  他の日または月エレメントとは`
	`  相互に排他的)`
	
	デフォルト値の 1 が、指定されない各エレメントに割り当てられます。日付形式の例を以下に示します。
	
	`"D-M-YYYY"`
	`"MM.DD.YYYY"`
	`"YYYYDDD"`

dumpfile = *x*	*x* は、リジェクトされた行を書き込む例外ファイルの (サーバー・データベース・パーティションによる) 完全修飾名です。 1 レコードにつき、最大で 32 KB のデータが書き込まれます。以下に、ダンプ・ファイルの指定方法の例を示します。
	
	`db2 load from data of del`
	`  modified by dumpfile = /u/user/filename`
	`  insert into table_name`
	
	ファイルは、インスタンスの所有者によって作成されて所有されます。デフォルトのファイル許可をオーバーライドするには、 **dumpfileaccessall** ファイル・タイプ修飾子を使用します。
	注:
	1. パーティション・データベース環境の場合、パスはロードを実行するデータベース・パーティションにローカルなものでなければなりません。それによって、並行して実行される複数のロード操作が同じファイルに書き込むことを防ぐことができます。
	2. ファイルの内容は、非同期バッファー・モードでディスクに書き込まれます。ロード操作が失敗した場合や割り込みが発生した場合は、ディスクにコミットされたレコードの数を確実に把握する方法がありません。LOAD RESTART 後の整合性も保証できません。ファイルが完全であるとされるのは、 1 回のパスの中で開始して完了するロード操作の場合だけです。
	3. 指定されたファイルが既に存在する場合は、再作成されずに切り捨てられます。

dumpfileaccessall	ダンプ・ファイルの作成時に、読み取りアクセスを OTHERS に付与します。 このファイル・タイプ修飾子が有効なのは、以下の場合のみです。 1. **dumpfile** ファイル・タイプ修飾子と一緒に使用された場合。 2. ロード・ターゲット表に対してユーザーが SELECT 特権をもっている場合。 3. UNIX オペレーティング・システムに置かれている DB2 サーバー・データベース・パーティション上で発行された場合。 指定されたファイルが既に存在する場合、その許可は変更されません。

表 56. ロード・ユーティリティーで有効なファイル・タイプ修飾子: ASCII ファイル・フォーマット (ASC/DEL) (続き)

修飾子	説明
fastparse	使用に際しては、注意が必要です。ユーザー指定の列値の構文検査が削減されるので、パフォーマンスは向上します。表は、体系的な正確さが確保されます (セグメント化違反またはトラップを防ぐための十分なデータ・チェックがユーティリティーで実行されます) が、データの一貫性の妥当性検査は行われません。データに一貫性があり、正確であることが確実な場合にのみ、このオプションを使用してください。例えば、ユーザー指定のデータに無効なタイム・スタンプ列値 `:1>0-00-20-07.11.12.000000` が含まれている場合でも、**fastparse** が指定されていれば、その値は表に挿入されてしまいますが、**fastparse** が指定されていなければ、その値はリジェクトされます。
implieddecimal	暗黙指定されている小数点の位置が列定義によって決定され、値の終わりにあるとは見なされなくなります。例えば、値 `12345` は、`12345.00` ではなく、`123.45` として DECIMAL(8,2) 列にロードされます。 この修飾子は、 **packeddecimal** 修飾子と共に使用することはできません。

timeformat=*"x"*	*x* はソース・ファイル内の時刻のフォーマットです。[1] 有効な時刻エレメントは以下のとおりです。
	
	`H     - 時 (12 時間制の場合は 0 から 12、`
	`  24 時間制では 0 から 24 の範囲の`
	`  1 桁または 2 桁の数)`
	`HH    - 時 (12 時間制の場合は 0 から 12、`
	`  24 時間制では 0 から 24 の範囲の`
	`  2 桁の数;`
	`  H と相互に排他的)`
	`M     - 分 (0 から 59 の範囲の`
	`  1 桁または 2 桁の数)`
	`MM    - 分 (0 から 59 の範囲の 2 桁の数。`
	`  M とは相互に排他的)`
	`S     - 秒 (0 から 59 の範囲の`
	`  1 桁または 2 桁の数)`
	`SS    - 秒 (0 から 59 の範囲の 2 桁の数。`
	`  S と相互に排他的)`
	`SSSSS - 夜中の 12 時から数えた秒数`
	`  (00000 から 86399 の範囲の 5 桁の数。`
	`  他の時刻エレメントとは相互に排他的)`
	`TT    - 午前/午後の指定子 (AM または PM)`
	
	指定されない各エレメントには、デフォルト値の 0 が割り当てられます。時刻フォーマットの例を以下に示します。
	
	`"HH:MM:SS"`
	`"HH.MM TT"`
	`"SSSSS"`

表 56. ロード・ユーティリティーで有効なファイル・タイプ修飾子: ASCII ファイル・フォーマット (ASC/DEL) (続き)

修飾子	説明
timestampformat=*"x"*	*x* はソース・ファイルのタイム・スタンプのフォーマットです。[1] 有効なタイム・スタンプ・エレメントは以下のとおりです。 YYYY - 年 (0000 から 9999 の範囲の 4 桁の数) M - 月 (1 から 12 の範囲の 1 桁または 2 桁の数) MM - 月 (01 から 12 の 2 桁の数。 M および MMM とは相互に排他的) MMM - 月 (大文字小文字を区別しない月名の 3 文字の省略形。 M と MM とは相互に排他的) D - 日 (1 から 31 の範囲の 1 桁または 2 桁の数) DD - 日 (1 から 31 の範囲の 2 桁の数。D とは相互に排他的) DDD - 元日から数えた日数 (001 から 366 の範囲の 3 桁の数。 他の日または月のエレメントとは相互に排他的) H - 時 (12 時間制の場合は 0 から 12、24 時間制では 0 から 24 の 範囲の 1 桁または 2 桁の数。) HH - 時 (12 時間制の場合は 0 から 12、24 時間制では 0 から 24 の 範囲の 2 桁の数。 H と相互に排他的) M - 分 (0 から 59 の範囲の 1 桁または 2 桁の数) MM - 分 (0 から 59 の範囲の 2 桁の数。 M (分) とは相互に排他的) S - 秒 (0 から 59 の範囲の 1 桁または 2 桁の数) SS - 秒 (0 から 59 の範囲の 2 桁の数。 S と相互に排他的) SSSSS - 夜中の 12 時から数えた秒数 (00000 から 86399 の範囲の 5 桁の数。 他の時刻エレメントとは相互に排他的) U (1 から 12 時) - 小数秒 (U のオカレンス数は、各桁を 0 から 9 の範囲として、 桁数を表します) TT - 午前/午後の指定子 (AM または PM)

表 56. ロード・ユーティリティーで有効なファイル・タイプ修飾子: ASCII ファイル・フォーマット (ASC/DEL) (続き)

修飾子	説明

timestampformat=*"x"* (Continued)	YYYY、M、MM、D、DD、または DDD エレメントが指定されていない場合、デフォルト値の 1 が割り当てられます。 MMM エレメントが指定されていない場合、デフォルト値の「Jan」が割り当てられます。他のエレメントが指定されていない場合には、デフォルト値の 0 が割り当てられます。タイム・スタンプ・フォーマットの例を以下に示します。
	`"YYYY/MM/DD HH:MM:SS.UUUUUU"`
	MMM エレメントの有効な値は、「jan」、 「feb」、 「mar」、「apr」、「may」、 「jun」、 「jul」、 「aug」、 「sep」、 「oct」、 「nov」、および「dec」です。これらの値で大/小文字は区別されません。
	timestampformat 修飾子を指定しなかった場合、ロード・ユーティリティーは、タイム・スタンプ・フィールドで以下の 2 つの有効な形式のいずれかを使用します。
	`YYYY-MM-DD-HH.MM.SS`
	`YYYY-MM-DD HH:MM:SS`
	ロード・ユーティリティーは、DD と HH の間の区切り記号を調べてフォーマットを選択します。ダッシュ '-' になっていれば、ロード・ユーティリティーは、通常のダッシュとドットの形式 (YYYY-MM-DD-HH.MM.SS) を使用します。区切り文字がブランク・スペースの場合、ロード・ユーティリティーはコロン「:」を使用して、HH、MM、および SS を区切ります。
	どちらのフォーマットでも、マイクロ秒フィールド (UUUUUU) が含まれる場合、ロード・ユーティリティーは区切り文字としてドット「.」を使用します。YYYY-MM-DD-HH.MM.SS.UUUUUU も YYYY-MM-DD HH:MM:SS.UUUUUU も有効です。
	次の例では、ユーザー定義の日時形式を指示するデータを、schedule という表にロードする方法を示します。
	`db2 load from delfile2 of del`
	`modified by timestampformat="yyyy.mm.dd hh:mm tt"`
	`insert into schedule`

表 56. ロード・ユーティリティーで有効なファイル・タイプ修飾子: ASCII ファイル・フォーマット (ASC/DEL) (続き)

修飾子	説明
usegraphiccodepage	**usegraphiccodepage** が指定された場合、GRAPHIC または 2 バイト文字ラージ・オブジェクト (DBCLOB) データ・フィールドにロードされるデータは、GRAPHIC コード・ページであると見なされます。データの残りは、文字コード・ページであると見なされます。 GRAPHIC コード・ページは、文字コード・ページと関連付けられます。 LOAD は、**codepage** 修飾子が指定されている場合はその修飾子、または **codepage** 修飾子が指定されていない場合は、データベースのコード・ページを通じて、文字コード・ページを決定します。 この修飾子は、リカバリーされている表に GRAPHIC データがある場合にのみ、ドロップ済み表のリカバリーによって生成された区切りデータ・ファイルとともに使用される必要があります。 **制約事項** EXPORT ユーティリティーで作成された DEL ファイルは、1 つのコード・ページのみでエンコードされたデータを含んでいるため、これらのファイルで **usegraphiccodepage** 修飾子を指定することはできません。 **usegraphiccodepage** 修飾子はまた、ファイル内の 2 バイト文字ラージ・オブジェクト (DBCLOB) には無視されます。
xmlchar	XML 文書が文字コード・ページでエンコードされていることを示します。 このオプションは、指定の文字コード・ページでエンコードされていてもエンコード宣言を含まない XML 文書を処理するために役立ちます。 各文書で、宣言タグが存在してエンコード属性が含まれる場合、そのエンコード方式は文字コード・ページと一致する必要があります。一致しない場合、その文書を含む行はリジェクトされます。文字コード・ページは **codepage** ファイル・タイプ修飾子で指定されている値であるか、または指定がない場合はアプリケーションのコード・ページであることに注意してください。デフォルトでは、文書は Unicode でエンコードされているか、またはエンコード属性のある宣言タグを含んでいます。
xmlgraphic	XML 文書が指定された GRAPHIC コード・ページでエンコードされていることを示します。 このオプションは、特定の GRAPHIC コード・ページでエンコードされていてもエンコード宣言を含まない XML 文書を処理するために役立ちます。 各文書で、宣言タグが存在してエンコード属性が含まれる場合、そのエンコード方式は GRAPHIC コード・ページと一致する必要があります。一致しない場合、その文書を含む行はリジェクトされます。 GRAPHIC コード・ページは **codepage** ファイル・タイプ修飾子で指定されている値のグラフィック・コンポーネントであるか、または指定がない場合はアプリケーションのコード・ページのグラフィック・コンポーネントであることに注意してください。デフォルトでは、文書は Unicode でエンコードされているか、またはエンコード属性のある宣言タグを含んでいます。

表 57. ロード・ユーティリティーで有効なファイル・タイプ修飾子: ASC ファイル・フォーマット (区切り文字で区切られていない ASCII)

修飾子	説明
binarynumerics	数値データ (DECIMAL 以外) は、文字表記ではなく、バイナリー形式でなければなりません。これによって、コストの大きい変換操作を避けることができます。 このオプションがサポートされるのは、定位置 ASC において、 **reclen** オプションによって固定長レコードが指定されている場合だけです。 以下の規則が適用されます。 • BIGINT、INTEGER、および SMALLINT を除き、データ・タイプ間の変換は実行されません。 • データ長は、それぞれのターゲット列定義と一致している必要があります。 • FLOAT は、IEEE 浮動小数点フォーマットでなければなりません。 • ロード・ソース・ファイル中のバイナリー・データは、ロード操作を実行するプラットフォームに関係なく、ビッグ・エンディアンであると見なされます。 この修飾子の影響を受ける列のデータに NULL があってはなりません。この修飾子を使用すると、ブランク (通常は NULL と解釈される) は、バイナリー値であると解釈されます。
nochecklengths	**nochecklengths** を指定した場合は、ソース・データの中にターゲット表の列のサイズを超える列定義がある場合であっても、各行のロードが試みられます。コード・ページ変換によってソース・データが縮小されれば、そのような行であったとしても正常にロードすることができます。例えば、ソースに 4 バイトの EUC データがある場合、それがターゲットで 2 バイトの DBCS データに縮小されれば、必要なスペースは半分で済みます。このオプションが特に役立つのは、列の定義は不一致であるがソース・データが常に適合することが分かっている場合です。
nullindchar=*x*	*x* は、単一文字です。NULL 値を示す文字を *x* に変更します。*x* のデフォルト値は `Y` です。[2] 文字が 1 つの英字である場合を除いて、この修飾子は EBCDIC データ・ファイルで大文字小文字を区別します。例えば、NULL 標識文字を文字 `N` に指定した場合、 `n` も NULL 標識と認識されます。

表 57. ロード・ユーティリティーで有効なファイル・タイプ修飾子: ASC ファイル・フォーマット (区切り文字で区切られていない ASCII) (続き)

修飾子	説明

packeddecimal	**binarynumerics** 修飾子は DECIMAL フィールド・タイプで構成されないため、パック 10 進数データを直接ロードします。
	
	このオプションがサポートされるのは、定位置 ASC において、 **reclen** オプションによって固定長レコードが指定されている場合だけです。
	
	符号ニブル用にサポートされる値は以下のとおりです。
	`+ = 0xC 0xA 0xE 0xF`
	`- = 0xD 0xB`
	
	この修飾子の影響を受ける列のデータに NULL があってはなりません。この修飾子を使用すると、ブランク (通常は NULL と解釈される) は、バイナリー値であると解釈されます。
	
	サーバーのプラットフォームには関係なく、ロードのソース・ファイルに入っているバイナリー・データのバイト順はビッグ・エンディアンであることが前提となっています。つまり、この修飾子を Windows オペレーティング・システムで使用する場合も、バイト順を逆にしてはなりません。
	
	この修飾子は、 **implieddecimal** 修飾子と共に使用することはできません。

reclen=*x*	*x* は、最大値 32767 の整数です。各行では *x* 個の文字が読み取られ、行の終わりを示す改行文字は使用されません。
striptblanks	データを可変長フィールドにロードする際に、後書きブランク・スペースを切り捨てます。このオプションを指定しない場合、ブランク・スペースはそのまま保持されます。 このオプションは、 **striptnulls** と一緒に指定することはできません。これらは、相互に排他的なオプションです。このオプションは、廃止された **t** オプション (下位互換性のためだけにサポートされる) に代わるものです。
striptnulls	データを可変長フィールドにロードする際に、後書き NULL (0x00 文字) を切り捨てます。このオプションを指定しない場合、NULL はそのまま保持されます。 このオプションは、**striptblanks** と一緒に指定することはできません。これらは、相互に排他的なオプションです。このオプションは、廃止された **padwithzero** オプション (下位互換性のためだけにサポートされる) に代わるものです。

zoneddecimal	ゾーン 10 進数データをロードします。DECIMAL フィールド・タイプは、**binarynumerics** 修飾子の対象に含まれていません。このオプションがサポートされるのは、定位置 ASC において、 **reclen** オプションによって固定長レコードが指定されている場合だけです。
	
	ハーフバイト符号値は、以下のいずれかになります。
	`+ = 0xC 0xA 0xE 0xF`
	`- = 0xD 0xB`
	
	数字としてサポートされている値は、`0x0` から `0x9` です。
	
	ゾーンとしてサポートされている値は、`0x3` および `0xF` です。

表 58. ロード・ユーティリティーで有効なファイル・タイプ修飾子: DEL ファイル・フォーマット (区切り文字で区切られている ASCII)

修飾子	説明

chardel*x*	*x* は単一文字のストリング区切り文字です。デフォルト値は、二重引用符 (") です。指定した文字は、文字ストリングを囲むために、二重引用符の代わりに使用されます。[23] 文字ストリング区切り文字として明示的に二重引用符 (") を指定したい場合、以下のように指定します。
	`modified by chardel""`
	以下のように、文字ストリング区切りとして単一引用符 (') を指定することもできます。
	`modified by chardel''`

coldel*x*	*x* は単一文字の列区切り文字です。デフォルト値は、コンマ (,) です。指定した文字は、列の終わりを表すために、コンマの代わりに使用されます。 [23]
decplusblank	正符号文字。正の 10 進値の接頭部として、正符号 (+) ではなくブランク・スペースを使用します。デフォルトのアクションでは、正の 10 進数の前に正符号 (+) が付けられます。
decpt*x*	*x* は、小数点文字としてピリオドの代わりに使用される単一文字です。デフォルト値は、ピリオド (.) です。指定した文字は、小数点文字としてピリオドの代わりに使用されます。 [23]

delprioritychar	区切り文字の現在のデフォルト優先順位は、(1) レコード区切り文字、(2) 区切り文字、(3) 列区切り文字です。この修飾子を使用すると、区切り文字の優先順位が (1) 区切り文字、(2) レコード区切り文字、(3) 列区切り文字に戻り、以前の優先順位に依存している既存のアプリケーションが保護されます。構文:
	`db2 load ... modified by delprioritychar ...`
	例えば、以下のような DEL データ・ファイルがあるとします。
	`"Smith, Joshua",4000,34.98<row delimiter>`
	`"Vincent,<row delimiter>, is a manager", ...`
	`... 4005,44.37<row delimiter>`
	delprioritychar 修飾子を指定しているので、このデータ・ファイルは、2 行だけになります。2 番目の <row delimiter> は 2 番目の行の最初のデータ列の一部と解釈されますが、1 番目と 3 番目の <row delimiter> は実レコードの区切り文字と解釈されます。この修飾子が指定されていない 場合、このデータ・ファイルでは 3 行になり、各行は <row delimiter> によって区切られます。

keepblanks	タイプが CHAR、VARCHAR、LONG VARCHAR、または CLOB の各フィールドの前後のブランクを保持します。このオプションを指定しないと、区切り文字で囲まれていないすべての前後のブランクは除去され、表のすべてのブランク・フィールドに NULL が挿入されます。
	以下の例では、データ・ファイルにある前後のブランクを保存しながら、 TABLE1 という表にデータをロードする方法を示します。
	`db2 load from delfile3 of del`
	`  modified by keepblanks`
	`  insert into table1`

表 58. ロード・ユーティリティーで有効なファイル・タイプ修飾子: DEL ファイル・フォーマット (区切り文字で区切られている ASCII) (続き)

修飾子	説明
nochardel	ロード・ユーティリティーは、列区切り文字と列区切り文字の間にあるすべてのバイトが列データの一部であると見なします。文字区切り文字は、列データの一部として構文解析されます。 DB2 データベース・システムを使用してエクスポートしたデータについては、このオプションを指定しないでください (ただし、エクスポート時に **nochardel** を指定していた場合は例外です)。これは、区切り文字を持たないベンダー・データ・ファイルをサポートするために用意されています。不適切に使用すると、データが損失または破壊される場合があります。 このオプションを **chardelx**、 **delprioritychar** または **nodoubledel** と一緒に指定することはできません。これらは、相互に排他的なオプションです。
nodoubledel	二重文字区切りの認識を抑止します。

表 59. ロード・ユーティリティーで有効なファイル・タイプ修飾子: IXF ファイル・フォーマット

修飾子	説明
forcein	コード・ページが不一致でもデータを受け入れ、コード・ページ間の変換を抑止するようにユーティリティーに指示します。 固定長ターゲット・フィールドに、そのデータが入るだけの十分な大きさがあるかどうかがチェックされます。 **nochecklengths** を指定した場合、そのような検査は実行されず、各行のロードが試みられます。
nochecklengths	**nochecklengths** を指定した場合は、ソース・データの中にターゲット表の列のサイズを超える列定義がある場合であっても、各行のロードが試みられます。コード・ページ変換によってソース・データが縮小されれば、そのような行であったとしても正常にロードすることができます。例えば、ソースに 4 バイトの EUC データがある場合、それがターゲットで 2 バイトの DBCS データに縮小されれば、必要なスペースは半分で済みます。このオプションが特に役立つのは、列の定義は不一致であるがソース・データが常に適合することが分かっている場合です。

注:

1. 日付形式ストリングは必ず二重引用符で囲まなければなりません。フィールド区切り文字には、 a から z、A から Z、および 0 から 9 を使用することはできません。フィールド区切り文字は、 DEL ファイル・フォーマットの区切り文字またはフィールド区切り文字と同じであってはなりません。エレメントの開始および終了位置が明らかな場合、フィールド区切り文字は任意指定です。あいまいさが生じうるのは、項目の長さが一定でない D、H、M、または S などのエレメントが使用されている場合です (修飾の仕方によって異なります)。

   タイム・スタンプ・フォーマットの場合、月の記述子と分の記述子のどちらも文字 M を使用するため、区別があいまいにならないように注意する必要があります。月のフィールドは、他の日付フィールドと隣接していなければなりません。分フィールドは、他の時刻フィールドに隣接していなければなりません。あいまいなタイム・スタンプ形式の例を以下に示します。

   ```
   "M" (月または分のどちらにもとれる)
   "M:M" (月と分の区別がつかない)
   "M:YYYY:M" (両方とも月と解釈される)
   "S:M:YYYY" (時刻値と日付値の両方に隣接している)
   ```

あいまいな場合、ユーティリティーはエラー・メッセージを報告し、操作は失敗します。

以下に、明確なタイム・スタンプ・フォーマットを示します。

```
"M:YYYY" (M (月))
"S:M" (M (分))
"M:YYYY:S:M" (M (月)....M (分))
"M:H:YYYY:M:D" (M (分)....M (月))
```

二重引用符や円記号などの文字の前には、エスケープ文字 (例えば、¥) を付けなければなりません。

2. ファイル・タイプ修飾子 **chardel**、**coldel**、**decpt** に指定する文字値は、ソース・データのコード・ページに指定されている文字値でなければなりません。

   文字コード・ポイント (文字記号ではない) は、xJJ または 0xJJ という構文で指定することができます (JJ はコード・ポイントの 16 進表記)。例えば、列区切りとして # 文字を指定するには、以下のいずれかを使用します。

   ```
   ... modified by coldel# ...
   ... modified by coldel0x23 ...
   ... modified by coldelX23 ...
   ```

3. 『データ移動のための区切り文字の制約事項』に、区切り文字の指定変更として使用できる文字に適用される制限のリストが示されています。
4. サポートされていないファイル・タイプを **MODIFIED BY** オプションで使用しようとしても、ロード・ユーティリティーは警告を出しません。この場合、ロード操作が失敗し、エラー・コードが戻されます。
5. 暗黙的に隠された ROW CHANGE TIMESTAMP 列を含む表にインポートする場合、その列の暗黙的に隠されたプロパティーは反映されません。したがって、インポートするデータに列のデータが含まれていない場合に、明示的な列リストも存在しなければ、IMPORT コマンドで **rowchangetimestampmissing** ファイル・タイプ修飾子を指定することが必要です。

表 60. codepage および usegraphiccodepage 使用時の LOAD 動作

codepage=N	**usegraphiccodepage**	**LOAD の動作**
なし	なし	**CLIENT** オプションの指定があっても、ファイル内のすべてのデータは、アプリケーション・コード・ページではなく、データベース・コード・ページであると見なされます
あり	なし	ファイル内のすべてのデータは、コード・ページ N であると見なされます。 **警告:** N が 1 バイト・コード・ページの場合、GRAPHIC データをデータベースにロードすると壊れます。

表 60. codepage および usegraphiccodepage 使用時の LOAD 動作 (続き)

codepage=N	usegraphiccodepage	LOAD の動作
なし	あり	**CLIENT** オプションの指定があっても、ファイル内の文字データは、データベース・コード・ページであると見なされます。 **CLIENT** オプションの指定があっても、GRAPHIC データは、データベース GRAPHIC データのコード・ページであると見なされます。 データベース・コード・ページが 1 バイトの場合は、すべてのデータはデータベース・コード・ページであると見なされます。 **警告:** 1 バイト・データベースに GRAPHIC データをロードすると、壊れます。
あり	あり	文字データは、コード・ページ N であると見なされます。 GRAPHIC データは、N の GRAPHIC コード・ページであると見なされます。 N が 1 バイトまたは 2 バイト・コード・ページの場合は、すべてのデータは、コード・ページ N であると見なされます。 **警告:** N が 1 バイト・コード・ページの場合、GRAPHIC データをデータベースにロードすると壊れます。

PRUNE HISTORY/LOGFILE コマンド (ADMIN_CMD プロシージャーを使用)

リカバリー履歴ファイルから項目を削除したり、現在接続されているデータベース・パーティションのアクティブ・ログ・ファイル・パスからログ・ファイルを削除したりするのに使用します。リカバリー履歴ファイルからの項目の削除は、ファイルが非常に大きくなったり保持期間が長くなっている場合に必要になることがあります。

区画に分割された環境では、PRUNE HISTORY コマンドは、発行されたデータベース・パーティションに関してのみ実行されます。複数のパーティションに関して履歴を整理するには、PRUNE HISTORY コマンドをそれぞれ個々のデータベース・パーティションから発行するか、または db2_all 接頭部を使用して PRUNE HISTORY コマンドをすべてのデータベース・パーティションに対して実行します。

許可

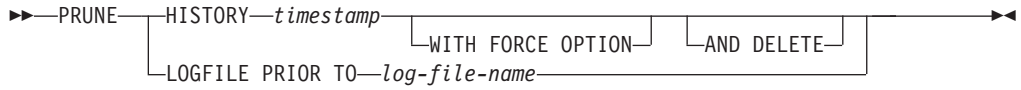
以下のいずれか。

- SYSADM
- SYSCTRL
- SYSMAINT
- DBADM

必要な接続

データベース

コマンド構文



コマンド・パラメーター

HISTORY *timestamp*
削除される、リカバリー履歴ファイルにある項目範囲を識別します。完全なタイム・スタンプ (書式 *yyyymmddhhmmss*)、または最初の接頭部 (最小値 *yyyy*) を指定できます。提供されているそのタイム・スタンプ以下のタイム・スタンプ付きのすべての項目は、リカバリー履歴ファイルから削除されます。

WITH FORCE OPTION
最新のリストア・セットのいくつかの項目がファイルから削除されるとしても、指定したタイム・スタンプに従って項目を整理することを指定します。リストア・セットは、バックアップ・イメージのすべてのリストアを含む、最新の全データベース・バックアップです。このパラメーターを指定しない場合、バックアップ・イメージ転送からのすべての項目は履歴の中で保守されます。

AND DELETE
履歴ファイルの項目を削除する際に、関連するログ・アーカイブを (ロケーション情報に基づいて) 物理的に削除することを指定します。このオプションは、ログ・アーカイブが不要になった場合に、アーカイブ・ストレージ・スペースがリカバリーされるようにする上で、特に有用です。ユーザー出口プログラムによりログをアーカイブしている場合は、このオプションを使用してそれらのログを削除することはできません。

auto_del_rec_obj データベース構成パラメーターを ON に設定している場合、AND DELETE パラメーターを指定して **PRUNE HISTORY** を呼び出すと、バックアップ・イメージも物理的に削除され、その履歴ファイル項目が整理される場合にはコピー・イメージがロードされます。

LOGFILE PRIOR TO *log-file-name*
ログ・ファイル名を表すストリング (例: S0000100.LOG) を指定します。指定したログ・ファイルより前のすべてのログ・ファイルは削除されます。指定したログ・ファイルそのものは削除されません。 **logretain** データベース構成パラメーターは、RECOVERY または CAPTURE に設定する必要があります。

例

例 1 リカバリー履歴ファイルから、2003 年 12 月 31 日以前に書き込まれたすべての項目を除去します。

```
CALL SYSPROC.ADMIN_CMD ('prune history 20031231')
```

例 2: アクティブ・ログ・ファイル・パスから、S0000100.LOG より前のすべてのログ・ファイルを削除します (S0000100.LOG は削除しません)。

```
CALL SYSPROC.ADMIN_CMD('prune logfile prior to S0000100.LOG')
```

使用上の注意

WITH FORCE OPTION が使用されている場合、データベースの自動リストアに必要な項目を削除してしまう可能性があります。その場合でも手動リストアは正常に動作します。また、このコマンドを使用すると、db2ckrst ユーティリティーが、必要なバックアップ・イメージの完全なチェーンを正しく分析できなくなる可能性もあります。 PRUNE HISTORY コマンドを **WITH FORCE OPTION** なしで使用した場合、必要な項目が削除されることはありません。

状況が DB2HISTORY_STATUS_DO_NOT_DELETE の項目は整理されません。**WITH FORCE OPTION** が使用されている場合、DB2HISTORY_STATUS_DO_NOT_DELETE というマークが付いたオブジェクトは、やはり整理または削除されます。 UPDATE HISTORY コマンド、UPDATE_HISTORY を指定した ADMIN_CMD、または db2HistoryUpdate API を使用して、リカバリー履歴ファイルの項目の状況を DB2HISTORY_STATUS_DO_NOT_DELETE に設定できます。DB2HISTORY_STATUS_DO_NOT_DELETE 状況を使用して、キー・リカバリー履歴ファイルの項目が整理されないようにしたり、関連するリカバリー・オブジェクトが削除されないようにすることができます。

PRUNE HISTORY コマンドを使用してスナップショット・データベース・バックアップ履歴ファイルの項目を整理できますが、**AND DELETE** パラメーターを使用して関連する物理リカバリー・オブジェクトを削除することはできません。スナップショット・バックアップ・オブジェクトを削除する唯一の方法は、db2acsutil コマンドを使用することです。

このコマンドは、アプリケーションが現在接続しているデータベース・パーティションだけに作用します。

QUIESCE DATABASE コマンド (ADMIN_CMD プロシージャーを使用)

指定したインスタンスおよびデータベースからすべてのユーザーを強制的に切断して、静止モードにします。

データベースが静止モードにある間、それに対して管理タスクを実行できます。管理タスクの完了後、UNQUIESCE コマンドを使用してデータベースをアクティブ化して、シャットダウンしたり他のデータベースの開始を実行したりすることなく、他のユーザーがデータベースに接続できるようにします。

このモードでは、この制限モード中に権限を持つユーザーだけがデータベースに接続することができます。 SYSADM および DBADM 権限を持つユーザーは、データベースの静止中に常にそのデータベースにアクセスできます。

有効範囲

QUIESCE DATABASE を使用すると、データベース内のすべてのオブジェクトは静止モードに入ります。許可が与えられたユーザー/グループ、および SYSADM、SYSMAINT、DBADM、または SYSCTRL だけが、データベースまたはそのオブジェクトにアクセスできます。

許可

以下のいずれかです。

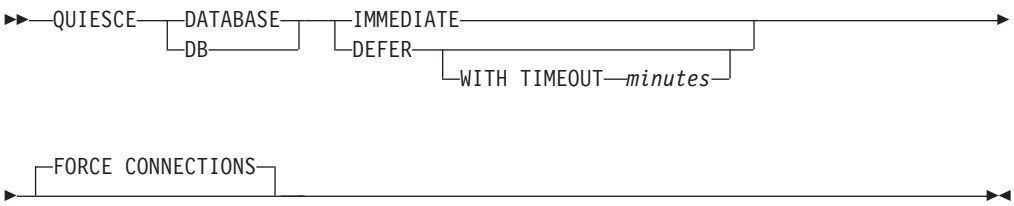
データベース・レベルの静止の場合:

- SYSADM
- DBADM

必要な接続

データベース

コマンド構文



コマンド・パラメーター

DEFER
QUISCE の実行をアプリケーションが現行の作業単位をコミットするまで待ちます。

WITH TIMEOUT *minutes*
アプリケーションが現在の作業単位をコミットするのを待機する時間を分単位で指定します。値を指定しない場合、単一パーティション・データベース環境では、デフォルト値が 10 分になります。パーティション・データベース環境では、データベース・マネージャー構成パラメーター **start_stop_time** によって指定された値が使用されます。

IMMEDIATE
トランザクションがコミットされるのを待たず、即時にトランザクションをロールバックします。

FORCE CONNECTIONS
接続を強制的にオフにします。

DATABASE
データベースを静止します。データベース内のすべてのオブジェクトを静止モードにします。指定したグループのユーザーと、

SYSADM、SYSMAINT、および SYSCTRL 権限を持つユーザーだけが、データベースまたはそのオブジェクトにアクセスすることができます。

例

データベースと接続しているすべてのユーザーを強制的に切断します。

```
CALL SYSPROC.ADMIN_CMD( 'quiesce db immediate' )
```

- このコマンドとともに **FORCE CONNECTIONS** オプションを指定すると、データベースからすべてのユーザーを強制的に切断します。 **FORCE CONNECTIONS** はデフォルトの動作です。コマンドのパラメーターは、互換性の理由により許容されています。
- コマンドは **FORCE CONNECTIONS** と同期され、**FORCE CONNECTIONS** が完了しないと完了しません。

使用上の注意

- QUIESCE DATABASE の後、SYSADM、 SYSMAINT、SYSCTRL、または DBADM 権限、および GRANT または REVOKE 特権を持つユーザーは、接続可能なユーザーを指定できます。この情報は永続的にデータベース・カタログ表に保管されます。

  以下に例を示します。

  ```
  grant quiesce_connect on database to username/groupname
  revoke quiesce_connect on database from username/groupname
  ```

- コマンドの実行状況は、CALL ステートメントからの結果である SQLCA で戻されます。

QUIESCE TABLESPACES FOR TABLE コマンド (ADMIN_CMD プロシージャーを使用)

特定の表の表スペースを静止させます。有効な静止モードは、共用、更新意図、および排他の 3 つです。

静止機能の結果として生じる状態には、次の 3 つの状態があります。

- 静止: SHARE
- 静止: UPDATE
- 静止: EXCLUSIVE

有効範囲

単一パーティション環境では、ロード操作中に排他モードのロード操作を起動すると、このコマンドは表スペースをすべて静止します。パーティション・データベース環境では、このコマンドはデータベース・パーティションでローカルに活動します。このコマンドは、ロード操作を実行しているデータベース・パーティションに属する表スペースの部分のみを静止します。パーティション表の場合、表に関連付けられている SYSDATAPARTITIONS.TBSPACEID および SYSDATAPARTITIONS.LONG_TBSPACEID の中のリストに含まれているすべての表スペースのうち、状況が「正常」、「アタッチ」、または「デタッチ」であるもの (例えば SYSDATAPARTITIONS.STATUS がそれぞれ ""、'A'、または 'D') が静止されます。

許可

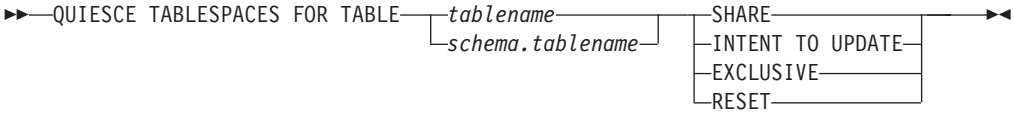
以下のいずれか。

- SYSADM
- SYSCTRL
- SYSMAINT
- DBADM
- LOAD

必要な接続

データベース

コマンド構文



コマンド・パラメーター

TABLE

tablename
非修飾表名を指定します。システム・カタログ表を指定することはできません。

schema.tablename
修飾表名を指定します。 *schema* が指定されない場合には、CURRENT SCHEMA が使用されます。システム・カタログ表を指定することはできません。

SHARE
静止が共用モードであることを指定します。

「静止モードでの共用」要求を行うと、トランザクションは、表スペースに対して意図的共有ロックを、および表に対して共有ロックを要求します。トランザクションがロックを獲得すると、表スペースの状態が QUIESCED SHARE に変更されます。この状態は、他のユーザーがこれと矛盾する状態を保持していない場合に限り、静止者に付与されます。表スペースの状態は、その状態が持続されるように、許可 ID およびそのユーザーのデータベース・エージェント ID とともに、表スペースにある表に記録されます。ある表の表スペースが QUIESCED SHARE 状態である間は、その表を変更できません。表および表スペースに要求するその他の共用モードは、認められます。トランザクションがコミットまたはロールバックされる際、ロックは解除されますが、その表の表スペースはその状態が明示的にリセットされるまで、 QUIESCED SHARE 状態のまま残ります。

INTENT TO UPDATE
静止モードが更新意図モードであることを指定します。

「静止モードでの更新意図」要求を行うと、表スペースは意図的排他 (IX) モードでロックされ、表は更新 (U) モードでロックされます。表スペースの状態は、表スペースの表に記録されます。

EXCLUSIVE
静止が排他モードであることを指定します。

「静止モードでの排他」要求を行うと、トランザクションは、表スペースに対する特別な排他ロックと、表に対する特別な排他ロックを要求します。トランザクションがロックを獲得すると、表スペースの状態が QUIESCED EXCLUSIVE に変更されます。表スペースの状態は、静止者の許可 ID およびデータベース・エージェント ID とともに、表スペース表に記録されます。表スペースは、スーパー排他モードで保留されているため、表スペースへの他のアクセスは認められません。静止プログラム機能を呼び出すユーザー (静止プログラム) は、その表と表スペースへの排他的アクセスを行うことができます。

RESET
表スペースの状態が、「正常」にリセットされることを指定します。静止要求を発行した接続がまだアクティブである場合、静止状態をリセットすることはできません。

例

社員 (staff) 表が含まれている表スペースを静止します。

```
CALL SYSPROC.ADMIN_CMD( 'quiesce tablespaces for table staff share' )
```

使用上の注意

このコマンドは、宣言一時表に対してはサポートされていません。

静止は持続ロックです。その利点は、それがトランザクション障害、接続障害、およびシステム障害 (電源障害や、リブートなど) が生じても持続することです。

静止は接続によって所有されます。接続が失われた場合、静止は残りますが、非所有の状態に移り、*ファントム静止* と呼ばれます。例えば、削除フェーズ中に停電によってロード操作が割り込まれると、ロードされていた表の表スペースは削除ペンディング、静止排他状態で残ります。データベースの再始動時に、この静止は非所有 (ファントム) の状態になります。ファントム静止を取り除くには、静止モードが設定されたときに使用されたのと同じユーザー ID による接続が必要です。

ファントム静止を取り除くには、次のようにします。

1. 静止モードが設定されたときに使用されたのと同じユーザー ID でデータベースと接続する。
2. LIST TABLESPACES コマンドを使用して、静止させる表スペースを決定する。
3. 現行の静止状態を使用して、表スペースを再静止させる。以下に例を示します。

   ```
   CALL SYSPROC.ADMIN_CMD('quiesce tablespaces for table mytable exclusive' )
   ```

完了すると、新しい接続が静止を所有するようになり、ロード操作を再開できるようになります。

1 つの表スペースに対する静止者の限度は、常に 5 人です。

静止プログラムは表スペースの状態を、あまり制限的でない状態から、より制限的な状態へ (例えば、S から U へ、または U から X へ) アップグレードさせることができます。しかし、ユーザーが既に保持している状態より低い状態を要求しても、元の状態が戻されます。つまり、状態がダウングレードされることはありません。

コマンドの実行状況は、CALL ステートメントからの結果である SQLCA で戻されます。

REDISTRIBUTE DATABASE PARTITION GROUP コマンド (ADMIN_CMD プロシージャーを使用)

すべてのパーティション・グループ内のデータベース・パーティション間でデータを再配分します。このコマンドはデータベース・パーティション・グループにあるすべてのオブジェクトに影響を及ぼし、1 つのオブジェクトだけに限定することはできません。

有効範囲

このコマンドは、データベース・パーティション・グループ内のすべてのデータベース・パーティションに影響を与えます。

許可

以下のいずれかの権限が必要です。

- SYSADM
- SYSCTRL
- DBADM

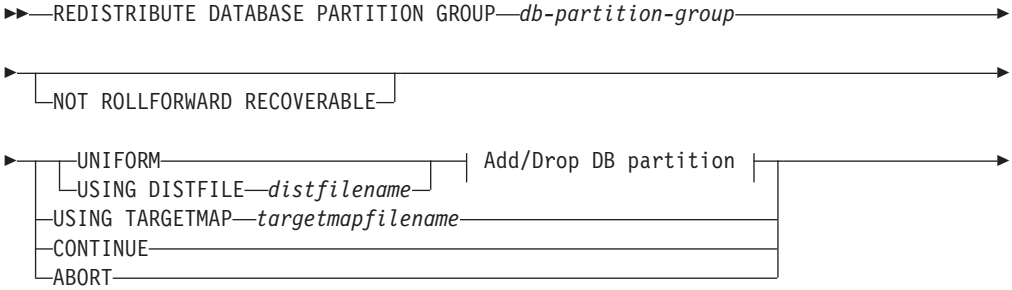
さらに、以下の権限のグループのいずれかも必要です。

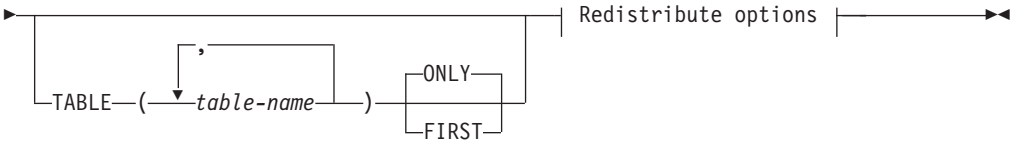
- 再配分されるデータベース・パーティション・グループ内のすべての表に対する DELETE、INSERT、および SELECT 特権。
- DATAACCESS 権限

必要な接続

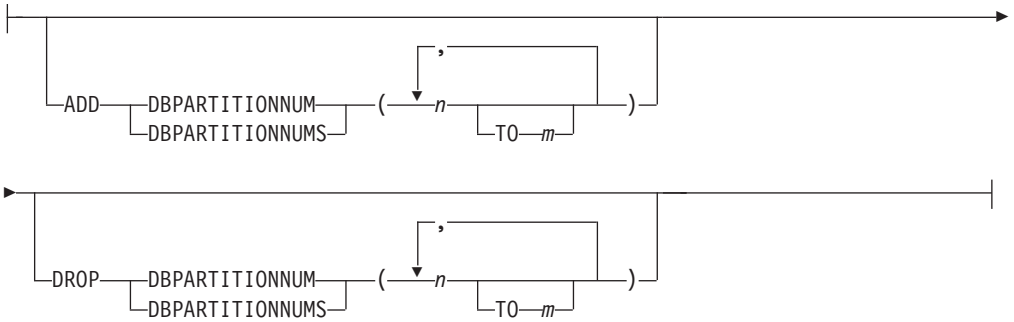
カタログ・パーティションとの接続。

コマンド構文

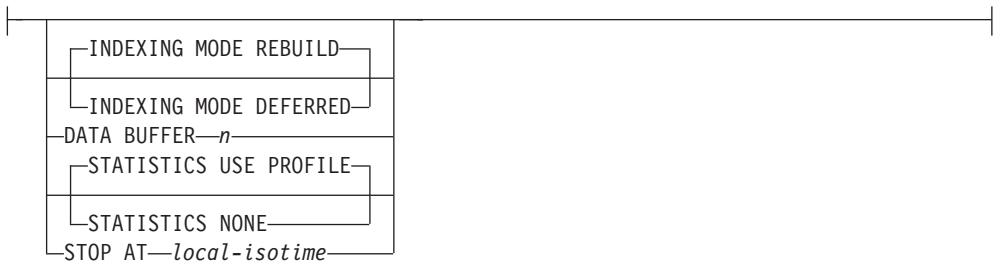




Add/Drop DB partition:



再配分オプション:



コマンド・パラメーター

DATABASE PARTITION GROUP *db-partition-group*

データベース・パーティション・グループの名前。この 1 部構成の名前は、SYSCAT.DBPARTITIONGROUPS カタログ表に記述されたデータベース・パーティション・グループを識別します。データベース・パーティション・グループは、現在再配分を受けることはできません。

注: IBMCATGROUP および IBMTEMPGROUP データベース・パーティション・グループ内の表を再配分することはできません。

NOT ROLLFORWARD RECOVERABLE

このオプションを使用すると、REDISTRIBUTE DATABASE PARTITION GROUP コマンドはロールフォワード・リカバリー可能ではありません。

- データは、内部での挿入および削除操作によってではなく、一括して移動されます。これにより、表のスキャンおよびアクセスの回数が減り、パフォーマンスが向上します。
- 挿入および削除操作それぞれに対するログ・レコードは必要ではなくなりました。このため、データの再配分を実行するときに、システム内で大容量のアクティブ・ログ・スペースおよびログ・アーカイブ・スペースを管理する必要がなくなりました。過去に、大容量のアクティブ・ログ・スペ

ースとストレージの要件のために、単一のデータ再配分操作を複数の小規模の再配分タスクに分割することを強制され、その結果、エンドツーエンドのデータ再配分操作にさらに時間が要求されていた場合には、これは特に有益です。

- REDISTRIBUTE DATABASE PARTITION GROUP コマンドを **NOT ROLLFORWARD RECOVERABLE** オプションとともに使用する場合、再配分操作は XML 列の入った表に対して **INDEXING MODE DEFERRED** オプションを使用します。表に XML 列が含まれていない場合、再配分操作はコマンドの発行時に指定された索引付けモードを使用します。

このオプションが使用されない 場合には、すべての行移動に関する詳細なロギングが実行されるため、中断やエラーが起きた場合、またはほかにビジネス上の必要が生じた場合に、データベースを後からリカバリーすることができます。

UNIFORM
データがハッシュ・パーティション間で均等に配分されることを指定します (つまり、それぞれのハッシュ・パーティションが同じ数の行を持つことが想定されます)。しかし、それぞれのデータベース・パーティションに同じ数のハッシュ・パーティションはマップされません。再配分後、データベース・パーティション・グループのすべてのデータベース・パーティションは、ほぼ同じ数のハッシュ・パーティションを持っています。

USING DISTFILE *distfilename*
分散キー値の分散に偏りがある場合、このオプションを使用して、データベース・パーティション・グループのデータベース・パーティション全体にわたるデータの均一な再分散を行います。

distfilename を使用して、32 768 個のハッシュ・パーティションにわたる現行のデータの配分を指示します。

行カウント、バイト・ボリューム、または他の任意の尺度を使用して、各ハッシュ・パーティションで表示されたデータ量を示します。ユーティリティーは、パーティションに関連する整数値をそのパーティションの重みとして読み取ります。 *distfilename* を指定した場合、ユーティリティーはターゲット分散マップを生成します。このマップは、データベース・パーティション・グループのデータベース・パーティション全体においてデータをできるだけ均一に再配分するために使用されます。再配分した後は、データベース・パーティション・グループ内の各データベース・パーティションの重みが、ほぼ同じになります (データベース・パーティションの重みは、そのデータベース・パーティションにマップするすべてのハッシュ・パーティションの重みの合計です)。

例えば、入力配布ファイルに以下の項目があるとします。

```
10223
1345
112000
0
100
...
```

例の中で、ハッシュ・パーティション 2 は 112000 の重みを持ち、パーティション 3 (重さは 0) には、マッピングするデータがまったくありません。

distfilename には、32 768 の正整数値が文字形式で入っていなければなりません。値の合計は、4 294 967 295 以下である必要があります。

distfilename は、その完全なパス名を含めることが必要であり、また *distfilename* はサーバー上に存在していなければならず、接続されているパーティションからアクセス可能でなければなりません。

USING TARGETMAP *targetmapfilename*

targetmapfilename で指定されたファイルは、ターゲット分散マップとして使用されます。データの再配分はこのファイルに従って行われます。*targetmapfilename* は、その完全なパス名を含めることが必要であり、また *targetmapfilename* はサーバー上に存在していなければならず、接続されているパーティションからアクセス可能でなければなりません。

ターゲット・マップに含まれるデータベース・パーティションがデータベース・パーティション・グループ中に存在しないと、エラーが戻されます。REDISTRIBUTE DATABASE PARTITION GROUP コマンドを実行する前に、ALTER DATABASE PARTITION GROUP ADD DBPARTITIONNUM ステートメントを実行してください。

ターゲット・マップから除外されたデータベース・パーティションが、データベース・パーティション・グループにある 場合、そのデータベース・パーティションはパーティションの中に含まれていません。このようなデータベース・パーティションは、 REDISTRIBUTE DATABASE PARTITION GROUP コマンドの前か後に ALTER DATABASE PARTITION GROUP DROP DBPARTITIONNUM ステートメントを使用することによってドロップできます。

CONTINUE

直前に失敗または停止した REDISTRIBUTE DATABASE PARTITION GROUP 操作を継続します。何も起こらなければ、エラーが戻されます。

ABORT

直前に失敗または停止した REDISTRIBUTE DATABASE PARTITION GROUP 操作をアボートします。何も起こらなければ、エラーが戻されます。

ADD

DBPARTITIONNUM *n*

TO *m*

n または *n* **TO** *m* では、データベース・パーティション・グループに追加するデータベース・パーティション番号のリストを指定します。指定するパーティションは、データベース・パーティション・グループにすでに定義済みであってはなりません (SQLSTATE 42728)。ADD DBPARTITIONNUM 節を指定して ALTER DATABASE PARTITION GROUP ステートメントを実行する操作と等価です。

DBPARTITIONNUMS *n*

TO *m*

n または *n* **TO** *m* では、データベース・パーティション・グループに追加するデータベース・パーティション番号のリストを指定します。指定するパーティションは、データベース・パーティション・グループにすでに定義済みであってはなりません (SQLSTATE 42728)。ADD DBPARTITIONNUM 節を指定して ALTER DATABASE PARTITION GROUP ステートメントを実行する操作と等価です。

注: このオプションを使用してデータベース・パーティションを追加すると、表スペースのコンテナーは、データベース・パーティション・グループ内で最も小さい番号を持つ既存のパーティション内の、対応する表スペースのコンテナーに基づくことになります。その結果、コンテナーの名前の競合が発生する場合は、このオプションを使用しないでください (新しいパーティションが既存のコンテナーと同じ物理マシンにあると、そのような名前の競合が発生する可能性があります)。そのような場合は、WITHOUT TABLESPACES オプションを指定して ALTER DATABASE PARTITION GROUP ステートメントを使用してから、REDISTRIBUTE DATABASE PARTITION GROUP コマンドを実行してください。その後、適切な名前を指定して、表スペース・コンテナーを手動で作成できます。

DROP

DBPARTITIONNUM *n*

TO *m*

n または *n* **TO** *m* では、データベース・パーティション・グループからドロップするデータベース・パーティション番号のリストを指定します。指定するパーティションは、データベース・パーティション・グループにすでに定義されている必要があります (SQLSTATE 42729)。DROP DBPARTITIONNUM 節を指定して ALTER DATABASE PARTITION GROUP ステートメントを実行する操作と等価です。

DBPARTITIONNUMS *n*

TO *m*

n または *n* **TO** *m* では、データベース・パーティション・グループからドロップするデータベース・パーティション番号のリストを指定します。指定するパーティションは、データベース・パーティション・グループにすでに定義されている必要があります (SQLSTATE 42729)。DROP DBPARTITIONNUM 節を指定して ALTER DATABASE PARTITION GROUP ステートメントを実行する操作と等価です。

TABLE *tablename*
再配分処理する表の順番を指定します。

ONLY 表の順序の後に **ONLY** キーワード (デフォルト) を使用すると、指定した表だけが再配分の対象になります。残りの表は、後から REDISTRIBUTE CONTINUE コマンドによって処理できます。これはデフォルトです。

FIRST 表の順序の後に **FIRST** キーワードを使用すると、指定した表が指定の順序で再配分処理を受け、データベース・パーティション・グループ内の残りの表は、ランダムな順序で再配分処理を受けることになります。

INDEXING MODE

このパラメーターでは、**NOT ROLLFORWARD RECOVERABLE** オプションが指定されている場合の、再配分時における索引の保守方法を指定します。有効な値は以下のとおりです。

REBUILD

索引が最初から再作成されます。このオプションを使用する場合は、索引が有効である必要はありません。このオプションを使用する結果として、「索引」ページがディスク上で一緒にクラスター化されます。

DEFERRED

再配分で索引の維持を試行しません。リフレッシュが必要であることを示すマークが索引に付けられます。そのような索引に最初にアクセスした時点で再作成が強制実行されるか、データベースの再始動時に索引が再作成されることになります。

注: MDC 表以外の場合は、INDEXING MODE DEFERRED を指定しなくても、表に無効な索引があれば、**REDISTRIBUTE DATABASE PARTITION GROUP** コマンドによって、そのような索引の再作成が自動的に実行されます。MDC 表の場合は、**INDEXING MODE DEFERRED** を指定しても、表の再配分の開始前に無効な複合索引が再作成されます。ユーティリティーは、MDC 表を処理するために複合索引を必要とするからです。

DATA BUFFER *n*

ユーティリティー内でデータを転送するためのバッファー・スペースとして使用する 4 KB ページの数を指定します。指定された値がサポートされている最小値よりも小さい場合には、最小値が使用され、警告は戻されません。 DATA BUFFER 値を指定しないと、実行時に各表の処理を開始する時点で、ユーティリティーによって適切なデフォルトが計算されます。具体的には、表の再配分の開始時点でユーティリティー・ヒープで使用可能になっているメモリーの 50% を使用することを基本にしながら、さまざまな表プロパティーを考慮に入れることによって、デフォルトを計算することになります。

このメモリーは、ユーティリティー・ヒープから直接に割り当てられ、そのサイズは **util_heap_sz** データベース構成パラメーターで修正可能です。バージョン 9.5 以降では、システムにさらに使用可能なメモリーがある場合、 REDISTRIBUTE DATABASE PARTITION GROUP コマンドの DATA BUFFER オプションの値は、一時的に **util_heap_sz** を超えることができます。

STOP AT *local-isotime*

このオプションを指定すると、各表のデータ再配分を開始する前に、*local-isotime* と現在のローカル・タイム・スタンプが比較されます。指定した *local-isotime* が現在のローカル・タイム・スタンプと同じか、それよりも早いと、ユーティリティーは処理を停止して、警告メッセージを生成します。停止時に進行中であった表のデータ再配分の処理は中断されずに完了します。表の新規のデータ再配分の処理は開始されません。未処理の表の再配分を実行するには、**CONTINUE** オプションを使用します。この *local-isotime* 値は、日付と時刻の組み合わせを識別する 7 部構成の文字ストリングのタイム・スタンプとして指定します。形式は、現地時間の *yyyy-mm-dd-hh.mm.ss.nnnnnn* (年、月、日、時、分、秒、マイクロ秒) です。

STATISTICS

このオプションでは、ユーティリティーが、統計プロファイルのある表の統計を収集するかどうかを指定します。このオプションを指定するほうが、データ再配分の完了後に RUNSTATS コマンドを別途実行するよりも効率的です。

USE PROFILE

統計プロファイルのある表の統計を収集します。統計プロファイルのない表については、何も実行されません。これはデフォルトです。

NONE 表の統計を収集しません。

例: 再配分の手順

ノード・グループからノードを追加またはドロップするとします。以下は、ノード・グループにノードを新規追加し、データを再配分する手順です。追加されるデータベース・パーティションは分散マップにありませんが、データベース・パーティション・グループ内の表スペースのコンテナーが作成されます。REDISTRIBUTE DATABASE PARTITION GROUP 操作が正常に完了すると、データベース・パーティションが分散マップに追加されます。

1. 再配分が必要となるノード・グループを識別します。この文書では、再配分する必要があるノード・グループは sampleNodegrp です。
2. 再配分前に無効化または除去しておく必要があるオブジェクトを特定します。
   a. 複製 MQT: このタイプの MQT は、REDISTRIBUTE ユーティリティーの一部としてサポートされていません。これらを再配分の実行前にドロップし、後で再作成する必要があります。

      ```
      SELECT tabschema, tabname
        FROM syscat.tables
        WHERE partition_mode = 'R'
      ```

   b. 表書き込みイベント・モニター: 表書き込みイベント・モニターでは、データベース・パーティション・グループ内の表が再配分されるため、自動的にアクティブにされる表書き込みイベント・モニターを使用不可にする必要があります。

```
SELECT distinct evmonname
  FROM syscat.eventtables E
  JOIN syscat.tables T on T.tabname = E.tabname
    AND T.tabschema = E.tabschema
  JOIN syscat.tablespaces S on S.tbspace = T.tbspace
    AND S.ngname = 'sampleNodegrp'
```

c. Explain 表: Explain 表は、単一パーティション・ノード・グループに作成するよう勧められています。しかし、再配分を必要とするノード・グループ内にそれらの表が定義されている場合は、今までに生成されたデータを維持する必要がないなら、再配分する前にそれらをドロップして、再配分が完了したときに再定義することも可能です。

d. 表アクセス・モードおよびロード状態: 再配分するノード・グループ内のすべての表がフル・アクセス・モードであること、および、ロード・ペンディングまたはロード進行中の状態でないことを確認してください。

```
SELECT DISTINCT TRIM(T.OWNER) || ¥'.¥' || TRIM(T.TABNAME)
  AS NAME, T.ACCESS_MODE, A.LOAD_STATUS
 FROM SYSCAT.TABLES T, SYSCAT.DBPARTITIONGROUPS
  N, SYSIBMADM.ADMINTABINFO A
 WHERE T.PMAP_ID = N.PMAP_ID
  AND A.TABSCHEMA = T.OWNER
  AND A.TABNAME = T.TABNAME
  AND N.DBPGNAME = 'sampleNodegrp'
  AND (T.ACCESS_MODE <> 'F' OR A.LOAD_STATUS IS NOT NULL)
```

e. 統計プロファイル: 統計プロファイルが表に定義されている場合、表統計を再配分処理の一環として更新できます。REDISTRIBUTE ユーティリティーで表の統計を更新するなら、入出力を削減できます。なぜなら、再配分のために全データがスキャンされ、RUNSTATS でさらにデータをスキャンすることが不要となるためです。

```
RUNSTATS on table schema.table
  USE PROFILE runstats_profile
  SET PROFILE ONLY
```

3. データベース構成を検討します。**util_heap_sz** は、データベース・パーティション間でのデータ移動処理をする上で重要です。したがって、再配分時に備えて、できるだけ多くのメモリーを **util_heap_sz** に割り振ってください。索引の再作成が再配分の一環として行われる場合、十分な **sortheap** が必要です。必要に応じて、**util_heap_sz** および **sortheap** を増加して、再配分のパフォーマンスを向上させます。

4. 新規データベース・パーティションで使用するデータベース構成設定を取得します。データベース・パーティションの追加時、デフォルトのデータベース構成が使用されます。そのため、REDISTRIBUTE コマンドを発行する前に新規ノード上でデータベース構成を更新することにより、ウェアハウス全体が均衡のとれた構成になるようにすることが重要です。

```
SELECT name,
  CASE WHEN deferred_value_flags = 'AUTOMATIC'
    THEN deferred_value_flags
    ELSE substr(deferred_value,1,20)
    END
  AS deferred_value
  FROM sysibmadm.dbcfg
  WHERE dbpartitionnum = existing-node
    AND deferred_value != ''
    AND name NOT IN ('hadr_local_host','hadr_local_svc','hadr_peer_window',
                     'hadr_remote_host','hadr_remote_inst','hadr_remote_svc',
                     'hadr_syncmode','hadr_timeout','backup_pending','codepage',
```

```
'codeset','collate_info','country','database_consistent',
'database_level','hadr_db_role','log_retain_status',
'loghead','logpath','multipage_alloc','numsegs','pagesize',
'release','restore_pending','restrict_access',
'rollfwd_pending','territory','user_exit_status',
'number_compat','varchar2_compat','database_memory')
```

5. リカバリー・ポイントを最新なものとするためには、再配分プロセスを開始する前にデータベース (または再配分するノード・グループ内の表スペース) をバックアップします。
6. db2nodes.cfg ファイルを更新し、新規のデータ BCU データベース・パーティション仕様を追加することにより、DB2 に新規データ BCU を定義します。また、ADD NODE WITHOUT TABLESPACES コマンドを使用して、新規データベース・パーティションを DB2 に定義します。

   ```
   db2start nodenum x export DB2NODE=x
   db2 add node without tablespaces
   db2stop nodenum x
   ```

   注: データ BCU 上で 1 番目の論理ポートでない場合は、後続する論理ポートのために、上述した一連のコマンドの前後で、1 番目の論理ポート番号の開始および停止を実行してください。
7. 新たに定義されたデータベース・パーティション上で、SYSTEM TEMPORARY 表スペース・コンテナーを定義します。

   ```
   ALTER TABLESPACE tablespace_name
     ADD container_information
     ON dbpartitionnums (x to y)
   ```

8. 複数のデータ BCU にまたがるデータベース・パーティション・グループに対し、新規論理データベース・パーティションを追加します。

   ```
   ALTER DATABASE PARTITION GROUP partition_group_name
     ADD dbpartitionnums (x to y)
     WITHOUT TABLESPACES
   ```

9. 新たに定義されたデータベース・パーティション上で、永続データ表スペース・コンテナーを定義します。

   ```
   ALTER TABLESPACE tablespace_name
     ADD container_information
     ON dbpartitionnums (x to y)
   ```

10. ステップ 4 で取得したデータベース構成設定を、新しいデータベース・パーティションに適用します (または、構成サポートの新しい DB2 9.5 単一ビューを使用して、全データベース・パーティションに対して、単一の UPDATE DB CFG コマンドを発行します)。
11. 再配分されるデータベース・パーティション・グループ内に存在する複製 MQT すべての定義を収集してから、ドロップします。

    ```
    db2look -d dbname -e -z
      schema -t replicated_MQT_table_names
      -o repMQTs.clp
    ```

12. 再配分されるデータベース・パーティション・グループに存在する、表書き込みイベント・モニターをすべて無効にします。

    ```
    SET EVENT MONITOR monitor_name STATE 0
    ```

13. REDISTRIBUTE ユーティリティーを実行して、すべてのデータベース・パーティションにわたって均一に再配分します。以下に、単純な再配分コマンドを示します。

```
REDISTRIBUTE DATABASE PARTITION GROUP sampleNodegrp
  NOT ROLLFORWARD RECOVERABLE uniform;
```

さらにユーザーは、REDISTRIBUTE コマンドへの入力データとして表リストを指定し、表を処理する順序を設定することを考慮する必要があります。REDISTRIBUTE ユーティリティーにより、データ (圧縮およびコンパクト) が移動されます。オプションとして、統計プロファイルが定義されている場合に、インデックスの再作成および統計の更新が行われます。それで、前述のコマンドの代わりに、以下のスクリプトを実行できます。

```
REDISTRIBUTE DATABASE PARTITION GROUP sampleNodegrp
  NOT ROLLFORWARD RECOVERABLE uniform
  TABLE (tab1, tab2,...) FIRST;
```

NOT ROLLFORWARD RECOVERABLE オプション使用の影響

REDISTRIBUTE DATABASE PARTITION GROUP コマンドが発行されて **NOT ROLLFORWARD RECOVERABLE** オプションが指定された場合、移動される各行のログ・レコードの書き込みを最小にする、最小限のロギング方式が使用されます。再配分操作のユーザビリティーという観点からすれば、このタイプのロギングは重要です。大規模なシステムですべてのデータ移動を完全にログに記録する方式を採用すれば、アクティブな永続ログ・スペースが大量に必要になり、通常はパフォーマンス特性が悪化してしまいます。しかし、この最小限のロギング・モデルの結果として、REDISTRIBUTE DATABASE PARTITION GROUP コマンドがロールフォワード・リカバリー可能ではない ことをユーザーが認識しておくことは重要です。これは、再配分操作の結果としてデータベースがロールフォワードすることになる操作では、再配分操作の影響を受けるすべての表が UNAVAILABLE 状態のままになるという意味です。そうした表はドロップのみが可能です。つまり、それらの表の中のデータをリカバリーする方法はないということです。したがって、リカバリー可能なデータベースについては、REDISTRIBUTE DATABASE PARTITION GROUP ユーティリティーが、NOT ROLLFORWARD RECOVERABLE オプションを指定して発行されると、対象になるすべての表スペースを BACKUP PENDING 状態に設定し、成功した再配分操作の最後に、再配分を受けたすべての表スペースのバックアップをユーザーが実行するように強制しています。再配分操作の後に作成したバックアップがあれば、ユーザーが再配分操作の途中でロールフォワードを実行する必要はなくなるからです。

再配分ユーティリティーがロールフォワード・リカバリー可能でないことの結果として、ユーザーが注意しなければならない重要な点が 1 つあります。再配分操作の実行中 (再配分の最後に再配分の対象になった表スペースのバックアップをユーザーが作成している時間も含める) に、ユーザーがそのデータベースの表に対する更新を認めると、その表が再配分対象のデータベース・パーティション・グループに含まれていない場合でも、データベース・コンテナーの破壊などの重大な障害が発生すれば、その更新内容が失われてしまう可能性があります。そうした更新が失われる理由は、再配分操作がロールフォワード・リカバリー可能でないからです。再配分操作の前に作成したバックアップからデータベースをリストアしなければならない場合、再配分操作の実行中に更新された内容を再生するために、ログに基づくロールフォワードを実行することは不可能です。すでに触れたとおり、再配分のロールフォワードが発生すると、再配分の対象になった表は UNAVAILABLE 状態になるからです。したがって、その状況で実行できるのは、ロールフォワードをせずに再配分の前に作成したバックアップからデータベースをリストアする操作だけに

なります。その後、再配分操作を再び実行することは可能です。しかし、元の再配分操作の実行中に更新された内容はすべて失われてしまいます。

この点の重要性は、いくら強調しても強調しすぎるということはありません。再配分操作の実行中に更新内容を失わないようにするために、以下のいずれかの条件を満たすようにする必要があります。

- REDISTRIBUTE DATABASE PARTITION GROUP コマンドの操作の実行中は、コマンドの完了後に操作対象の表スペースのバックアップを作成する時間も含めて、ユーザーが更新を実行しないようにすること。
- 再配分操作中に適用する更新内容を反復可能なソースに入れておくこと。その場合は、後からいつでも更新を再適用できます。例えば、ファイルに格納したデータを更新のソースとして使用し、バッチ処理でその更新を適用するようにすれば、データベースのリストアを必要とするような障害が発生したとしても、その更新内容は失われません。その更新内容を後から再び適用すればよいからです。

再配分操作中のデータベース更新を認めるかどうかについては、データベースのリストアが必要になった場合に更新処理を繰り返せるかどうかに基づいて、それぞれのシナリオでそのような更新が適切かどうかを判断する必要があります。

注: REDISTRIBUTE DATABASE PARTITION GROUP コマンドの操作中に発生するあらゆる障害がこのような問題につながるわけではありません。実際のところ、ほとんどの障害では、このような問題は発生しません。REDISTRIBUTE DATABASE PARTITION GROUP コマンドは完全に再始動可能であり、ユーティリティーの処理が途中で失敗したとしても、**CONTINUE** または **ABORT** オプションで簡単に続行/アボートできます。ここで取り上げてきた障害は、あくまでも、再配分操作の前に作成したバックアップからユーザーがデータベースをリストアしなければならなくなるような障害です。

例

データ配分ファイル distfile_for_dbpg_1 によって現在のデータ配分を提供し、データを新しいデータベース・パーティション 6 および 7 に移動することによって、データベース・パーティション・グループ DBPG_1 を再配分します。

```
CALL SYSPROC.ADMIN_CMD('REDISTRIBUTE DATABASE PARTITION GROUP DBPG_1
   USING DISTFILE /home/user1/data/distfile_for_dbpg_1
   ADD DATABASE PARTITION (6 TO 7) ')
```

使用上の注意

- 再配分操作を開始する前に、表が通常状態にあり、ロード・ペンディング状態や REORG ペンディング状態でないことを確認してください。表状態は LOAD QUERY コマンドを使って調べることができます。
- **NOT ROLLFORWARD RECOVERABLE** オプションが指定され、データベースがリカバリー可能である場合、ユーティリティーが最初に表スペースにアクセスした時点で、表スペースは BACKUP PENDING 状態になります。表スペースのバックアップが作成されるまで、その表スペースに含まれているすべての表は読み取り専用になります。表スペースのバックアップは、その表スペース内のすべての表の再配分処理が完了したときにのみ可能になります。

- 再配分操作の実行中に、その再配分操作に関する一般情報、および各表の処理開始時刻と処理終了時刻などの情報を含むイベント・ログ・ファイルが作成されます。このイベント・ログ・ファイルは、以下のようにサーバーに書き込まれます。
  - Linux および UNIX オペレーティング・システムの場合は、homeinst/sqllib/redist ディレクトリー。サブディレクトリーとファイル名の形式は、*database-name.database-partition-group-name.timestamp*.log になります。
  - Windows オペレーティング・システムの場合は、**DB2INSTPROF**¥*instance*¥redist ディレクトリー (**DB2INSTPROF** は、**DB2INSTPROF** レジストリー変数の値です)。サブディレクトリーとファイル名の形式は、*database-name.database-partition-group-name.timestamp*.log になります。
  - タイム・スタンプ値は、コマンドが発行された時の時刻です。
- このユーティリティーは、処理中に断続的な COMMIT を実行します。そのため、タイプ 2 接続では SQL30090N エラーを受け取ることになります。
- 再配分を受けた表と従属関係があるすべてのパッケージは無効になります。データベース・パーティション・グループの再配分操作が完了した後で、そのようなパッケージを明示的に再バインドすることをお勧めします。明示的な再バインドにより、無効パッケージに対する最初の SQL 要求の実行での初期遅延がなくなります。再配分メッセージ・ファイルには、再配分を受けたすべての表のリストが入ります。
- 統計プロファイルがある表については、再配分ユーティリティーの実行時に、デフォルトで統計が更新されます。統計プロファイルがない表の場合は、表や索引の統計を別途更新することをお勧めします。そのためには、再配分操作の完了後に、db2Runstats API を呼び出すか、RUNSTATS コマンドを実行できます。
- 複製されたマテリアライズ照会表や DATA CAPTURE CHANGES を用いて定義された表を含むデータベース・パーティション・グループは、再配分することができません。
- データベース・パーティション・グループに、既存の宣言済み一時表または作成済み一時表を含む USER TEMPORARY 表スペースがある場合、再配分を行うことはできません。
- **INDEXING MODE** などのオプションは、適用対象にならない表では無視されます (警告も生成されません)。例えば、**INDEXING MODE** は、索引のない表では無視されます。
- コマンドの実行状況は、CALL ステートメントからの結果である SQLCA で戻されます。
- **USING DISTFILE** *distfilename* または **USING TARGETMAP** *targetmapfilename* で参照するファイルでは、サーバー上のファイルを参照する必要があります。
- REDISTRIBUTE DATABASE PARTITION GROUP コマンドは、データベース・パーティションの追加サーバー要求が保留中または進行中の場合は、失敗する可能性があります (SQLSTATE 55071)。さらに、新規データベース・パーティション・サーバーがオンラインでインスタンスに追加された場合、およびすべてのア

プリケーションが新規データベース・パーティション・サーバーを認識しているわけではない場合、このコマンドは失敗する可能性があります (SQLSTATE 55077)。

互換性

DB2 バージョン 9.5 以前の XML レコード・フォーマットを使用する XML 列が入った表は、再配分できません。そうした表を新しいフォーマットに移行するには、ADMIN_MOVE_TABLE ストアード・プロシージャーを使用します。

バージョン 8 より前のバージョンとの互換性:

- キーワード **NODEGROUP** は、**DATABASE PARTITION GROUP** に置き換えられます。

REORG INDEXES/TABLE コマンド (ADMIN_CMD プロシージャーを使用)

索引または表を再編成します。

フラグメント化されていない物理的に連続したページに索引データを再構築することによって、表に定義されたすべての索引を再編成することができます。データ・パーティション表では、パーティション表で特定の非パーティション索引を再編成すること、もしくは特定のデータ・パーティションですべてのパーティション索引を再編成することができます。

索引節の **CLEANUP ONLY** オプションを指定すると、索引を再作成しないでクリーンアップが実行されます。このコマンドを宣言済み一時表または作成済み一時表の索引に対して使用することはできません (SQLSTATE 42995)。

表オプションは、フラグメント化されたデータを消去するために行を再作成、および情報を縮小化することによって、表を再編成します。パーティション表では、単一パーティションを再編成できます。

有効範囲

このコマンドは、データベース・パーティション・グループ内のすべてのデータベース・パーティションに影響を与えます。

許可

以下のいずれか。

- SYSADM
- SYSCTRL
- SYSMAINT
- DBADM
- SQLADM
- 表に対する CONTROL 特権

必要な接続

データベース

コマンド構文

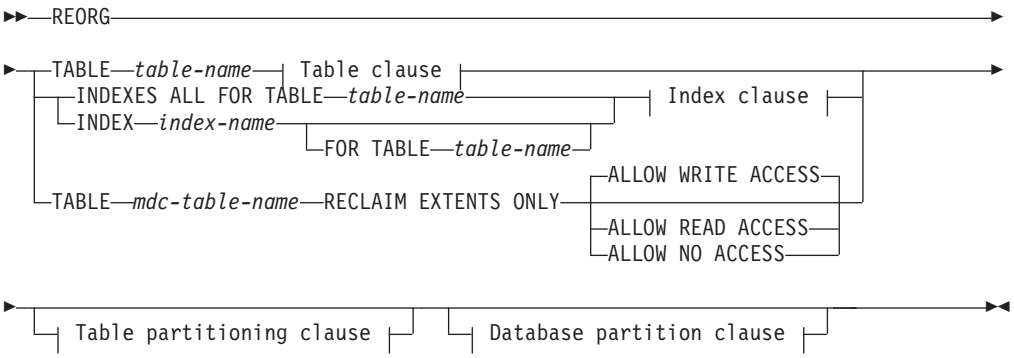
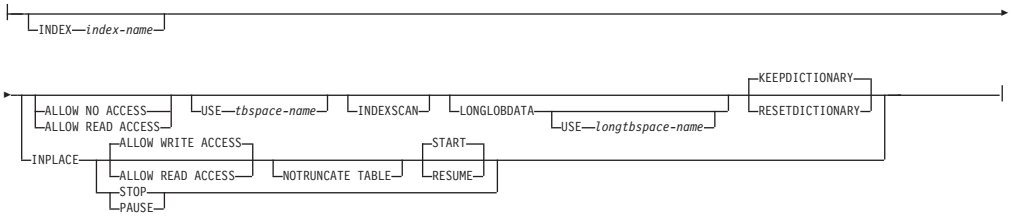


Table clause:



Index clause:

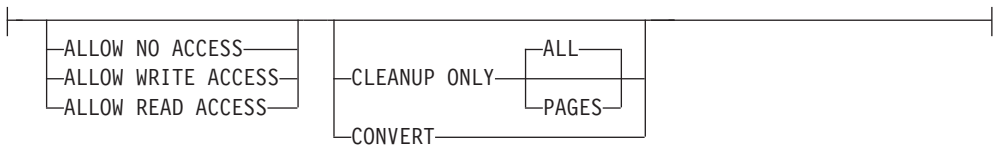
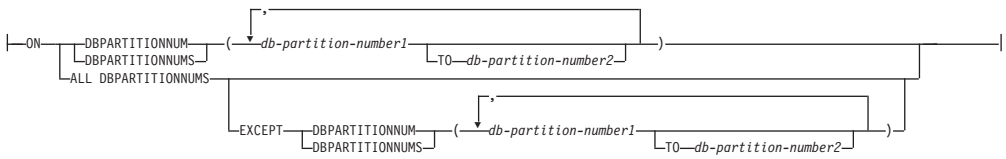


Table partitioning clause:

Database partition clause:



コマンド・パラメーター

INDEXES ALL FOR TABLE *table-name*
索引を再編成する表を指定します。表は、ローカルまたはリモート・データベースにあるものです。

INDEX *index-name*
データ・パーティション表上の再編成する個別索引を指定します。個別索引の再編成は、パーティション表上の非パーティション索引についてのみ サポートされています。このパラメーターは、ブロック索引に関してはサポートされていません。

FOR TABLE *table-name*
非パーティション索引 *index-name* が作成される表の名前を指定します。索引名がデータベースを通じて固有であるなら、このパラメーターはオプションです。

ALLOW NO ACCESS
REORG INDEXES の場合、索引が再編成される間に、他のユーザーがその表にアクセスできないことを指定します。パーティション表に対して **ON DATA PARTITION** 節が指定されている場合、その指定されたパーティションのみがこのアクセス・モード・レベルに制限されます。

REORG INDEX の場合、非パーティション索引が再編成される間に、他のユーザーがその表にアクセスできないことを指定します。

ALLOW READ ACCESS
REORG INDEXES の場合、索引が再編成される間に、他のユーザーがその表に対して、読み取り専用でアクセスできることを指定します。 **CLEANUP ONLY** オプションまたは ON DATA PARTITION 節が指定されていない限り、パーティション表の **REORG INDEXES** に対して **ALLOW READ ACCESS** モードはサポートされません。パーティション表に対して **ON DATA PARTITION** 節が指定されている場合、その指定されたパーティションのみがこのアクセス・モード・レベルに制限されます。

REORG INDEX の場合、非パーティション索引が再編成される間に、その表に読み取り専用でアクセスできることを指定します。

ALLOW WRITE ACCESS
REORG INDEXES の場合、索引が再編成される間に、他のユーザーがその表から読み取ったりそこに書き込んだりできることを指定します。 **CLEANUP ONLY** オプションまたは **ON DATA PARTITION** 節が指定されていない限り、パーティション表の **ALLOW WRITE ACCESS** モードはサポートされません。パーティション表に対して **ON DATA PARTITION** 節が指定されている場合、その指定されたパーティションのみがこのアクセス・モード・レベルに制限されます。

REORG INDEX の場合、非パーティション索引が再編成される間に、その表から読み取ったりそこに書き込んだりできることを指定します。

CLEANUP ONLY オプションが指定されていない限り、**ALLOW WRITE ACCESS** モードはマルチディメンション・クラスタリング (MDC) 表または拡張索引ではサポートされません。

ON DATA PARTITION 節が REORG INDEXES ALL コマンドと共に指定されている場合、以下の項目がデータ・パーティション表に適用されます。

- 指定されたデータ・パーティションのみがこのアクセス・モード・レベルに制限されます。指定されたパーティションのパーティション索引が再編成される間に、ユーザーは表の他のパーティションから読み取ったりそこに書き込んだりできます。

  以下の表に、**ON DATA PARTITION** 節が指定された時にサポートされるアクセス・モードおよび表の他のパーティションで許可される並行アクセスを示します。

表 61. REORG INDEXES ALL と共に ON DATA PARTITION 節が指定された時にサポートされるアクセス・モードおよび許可される並行アクセス

アクセス・モード	指定されたパーティションで許可される並行アクセス	他のパーティションで許可される並行アクセス
ALLOW NO ACCESS	アクセスできません	読み取りおよび書き込みアクセス
ALLOW READ ACCESS	索引が更新されるまでのパーティションにおける読み取り	読み取りおよび書き込みアクセス
ALLOW WRITE ACCESS	索引が更新されるまでのパーティションにおける読み取りおよび書き込みアクセス	読み取りおよび書き込みアクセス

- 指定されたパーティションのパーティション索引のみが再編成されます。非パーティション索引は再編成されません。

  「無効」または「再作成」とマークされた表に非パーティション索引がある場合、再編成の前に「無効」または「再作成」とマークされたすべての索引は再作成されます。それ以外の場合、索引オブジェクトが「無効」または「再作成」とマークされているなら、指定されたパーティションのパーティション索引のみが再編成されるか、または再作成されます。
- **CLEANUP ONLY** オプションも指定されている場合、指定されたパーティションのパーティション索引のみがクリーンにされます。

以下の表は、パーティションおよび非パーティション表における索引の再編成に対してサポートされるアクセス・モードを示します。

表 62. パーティションおよび非パーティション表における索引の再編成に対してサポートされるアクセス・モード

コマンド	表タイプ	Table partitioning clause	index (索引) 節に指定されたその他のパラメーター	サポートされるアクセス・モード
REORG INDEXES	非パーティション表	該当なし	任意	**ALLOW NO ACCESS**, **ALLOW READ ACCESS**[1], **ALLOW WRITE ACCESS**
REORG INDEX	パーティション表	該当なし	任意	**ALLOW READ ACCESS**[1]

表 62. パーティションおよび非パーティション表における索引の再編成に対してサポートされるアクセス・モード (続き)

コマンド	表タイプ	Table partitioning clause	index (索引) 節に指定されたその他のパラメーター	サポートされるアクセス・モード
REORG INDEXES	パーティション表	なし	指定なし	**ALLOW NO ACCESS** [1]
REORG INDEXES	パーティション表	**ON DATA PARTITION**	指定なし	**ALLOW NO ACCESS**, **ALLOW READ ACCESS**[1], **ALLOW WRITE ACCESS**
REORG INDEXES	パーティション表	**ON DATA PARTITION** 節ありまたはなし	**CLEANUP ONLY** を指定	**ALLOW NO ACCESS**, **ALLOW READ ACCESS**[1], **ALLOW WRITE ACCESS**

注:

1. アクセス節が指定されていない場合のデフォルト・モード。

CLEANUP ONLY

CLEANUP ONLY が要求されると、完全な再編成ではなくクリーンアップが実行されます。索引は再作成されません。解放されたページはこの表に定義された索引だけが再使用できます。

CLEANUP ONLY PAGES オプションは、コミット済み疑似空白ページを検索して解放します。コミット済み疑似空白ページとは、ページ上のすべてのキーに削除済みのマークが付いていて、それらすべての削除がコミット済みとして知られているページのことです。索引内の疑似空白ページの数は、RUNSTATS を実行して SYSCAT.INDEXES の NUM EMPTY LEAFS 列を調べることにより判別できます。 **PAGES** オプションは、コミット済みと判別された場合に NUM EMPTY LEAFS を消去します。

CLEANUP ONLY ALL オプションはコミット済み疑似空白ページを解放して、コミット済み疑似削除済みキーを疑似空白ではないページから除去します。このオプションは、隣接する複数のリーフ・ページをマージすると少なくとも PCTFREE のフリー・スペースを持つマージ済みリーフ・ページが生じる場合に、そのマージを試行します。 PCTFREE は、索引作成時に索引に定義されたフリー・スペースのパーセントです。デフォルトの PCTFREE は 10% です。2 つのページがマージ可能な場合、そのうちの 1 つのページが解放されます。疑似空白ページにあるものを除く、索引内の疑似削除済みキーの数は、 RUNSTATS を実行してから NUMRIDS DELETED を SYSCAT.INDEXES から選択することによって判別できます。 **ALL** オプションは、コミット済みと判別された場合に NUMRIDS DELETED および NUM EMPTY LEAFS を消去します。

ALL コミット済み疑似削除済みキーおよびコミット済み疑似空白キーを除去することにより、索引をクリーンアップすることを指定します。

PAGES

コミット済み疑似空白ページを索引ツリーから除去することを指定

します。これは、疑似空白ではないページ上の疑似削除済みキーはクリーンアップしません。これは疑似空白リーフ・ページだけをチェックするので、ほとんどの場合に **ALL** オプションを使用するよりも相当速くなります。

CONVERT
タイプ 1 索引をタイプ 2 索引に変換します。索引が既にタイプ 2 であれば、このオプションは何も行いません。

バージョン 9.7 では、タイプ 1 索引は使用されなくなり、作成されるすべての索引はタイプ 2 索引になりました。そのため、**CONVERT** オプションは推奨されていません。

バージョン 8 よりも前に作成されたすべての索引はタイプ 1 の索引です。バージョン 9.7 より前には、バージョン 8 以降により作成されたすべての索引はタイプ 2 の索引です。ただし、既にタイプ 1 の索引を持つ表に作成した索引は例外です。この場合、新規索引もタイプ 1 でした。バージョン 9.7 では作成されるすべての索引はタイプ 2 であるので、これは当てはまりません。

ALLOW READ ACCESS または **ALLOW WRITE ACCESS** オプションを使用して、索引が再編成されている間に、他のトランザクションに表に対する読み取り専用または読み取り/書き込みのいずれかのアクセス権限を許可することができます。 **ALLOW READ ACCESS** および **ALLOW WRITE ACCESS** は表へのアクセスを許可しますが、索引の再編成済みコピーを使用できる間は、表へのアクセスが許可されません。

TABLE *mdc-table-name* **RECLAIM EXTENTS ONLY**
使用されていないエクステントを再利用するために、再編成するマルチディメンション・クラスタリング (MDC) 表を指定します。 *schema.table-name* 形式の名前あるいは別名を使用することができます。 *schema* には、表作成時のユーザー名が入ります。スキーマ名を省略した場合、デフォルトのスキーマが想定されます。

REORG TABLE RECLAIM EXTENTS ONLY の場合、**ON DATA PARTITION** 節が指定されるときに、アクセス節は指定されたパーティションに対してのみ適用されます。指定されたパーティションのエクステントが再利用される間に、ユーザーは表の残りの部分から読み取ったりそこに書き込んだりできます。この状態も、デフォルトのアクセス・レベルに適用されます。

ALLOW NO ACCESS
REORG TABLE RECLAIM EXTENTS ONLY の場合、エクステントが再利用される間に、他のユーザーがその表にアクセスできないことを指定します。

ALLOW READ ACCESS
REORG TABLE RECLAIM EXTENTS ONLY の場合、エクステントが再利用される間に、他のユーザーがその表に読み取り専用アクセスができることを指定します。

ALLOW WRITE ACCESS

REORG TABLE RECLAIM EXTENTS ONLY の場合、エクステントが再利用される間に、他のユーザーがその表で読み取り/書き込みができることを指定します。

TABLE *table-name*

再編成する表を指定します。表は、ローカルまたはリモート・データベースにあるものです。 *schema.table-name* 形式の名前あるいは別名を使用することができます。 *schema* には、表作成時のユーザー名が入ります。スキーマ名を省略した場合、デフォルトのスキーマが想定されます。

型付き表の場合、指定する表名は階層のルート表の名前でなければなりません。

マルチディメンション・クラスタリング (MDC) 表の再編成に対して索引を指定することはできません。表のインプレース再編成を MDC 表に対して使用することはできません。

データ・パーティション表における表の再編成に対して **ON DATA PARTITION** 節が指定されている場合、指定されたデータ・パーティションのみが再編成されます。

- 表に非パーティション索引が定義されていない場合 (システム生成された XML パス索引を除く)、アクセス・モードは指定されたパーティションにのみ適用され、ユーザーは表の他のパーティションから読み取ったりそこに書き込んだりできます。
- 表に非パーティション索引が定義されている場合 (システム生成された XML パス索引を除く)、**ALLOW NO ACCESS** モードがデフォルトとなり、サポートされる唯一のアクセス・モードになります。この場合、その表は **ALLOW NO ACCESS** モードになります。**ALLOW READ ACCESS** が指定されている場合、SQL1548N が戻されます (SQLSTATE 5U047)。

表 63. 非パーティションおよびパーティション表における表の再編成に対してサポートされるアクセス・モード

コマンド	表タイプ	Table partitioning clause	サポートされるアクセス・モード
REORG TABLE	非パーティション表	該当なし	**ALLOW NO ACCESS**, **ALLOW READ ACCESS**[1]
REORG TABLE	パーティション表	指定なし	**ALLOW NO ACCESS**[1]
REORG TABLE (索引がない、または表にパーティション索引のみが定義されている)	パーティション表	**ON DATA PARTITION**	**ALLOW NO ACCESS**, **ALLOW READ ACCESS**[1]
REORG TABLE (システム生成された XML パス索引を除き、表に非パーティション索引が定義されていない)	パーティション表	**ON DATA PARTITION**	**ALLOW NO ACCESS**[1]

注:

1. アクセス節が指定されていない場合のデフォルト・モード。

データ・パーティション表では、表を再編成すると、表の再編成後にその表の非パーティション索引およびパーティション索引が再作成されます。 **ON DATA PARTITION** 節を使用して、データ・パーティション表の特定のデータ・パーティションを再編成する場合、表の再編成によって、指定したパーティションのみで非パーティション索引およびパーティション索引が再作成されます。

INDEX *index-name*
表を再編成する際に使用する索引を指定します。 *schema.index-name* 形式の完全修飾名を指定しない場合、デフォルトのスキーマが想定されます。 *schema* は、その索引が作成された時のユーザー名です。データベース・マネージャーは、再編成している表のレコードを物理的に再配列する索引を使用します。

表のインプレース再編成では、クラスタリング索引が表に定義されて、索引が指定されている場合、それはクラスタリング索引でなければなりません。インプレース・オプションが指定されない場合、指定された任意の索引が使用されます。索引名を指定しない場合には、そのレコードは順番に関係なく再編成されます。しかし、表にクラスタリング索引が定義されている場合、索引が指定されていなければ、クラスタリング索引が使用されて表がクラスタリングされます。 MDC 表を再編成しているときには、索引を指定できません。

INDEX および **ON DATA PARTITION** 節の両方を使用して表が再編成される場合、指定したパーティションのみが、索引 *index-name* を使用して再編成されます。

ALLOW NO ACCESS
表が再編成される間に、他のユーザーがその表にアクセスできないことを指定します。

ON DATA PARTITION 節を使用せずにパーティション表を再編成する場合、**ALLOW NO ACCESS** モードがデフォルトになり、唯一のサポートされるアクセス・モードになります。

データ・パーティション表に対して **ON DATA PARTITION** 節が指定されている場合、指定されたデータ・パーティションのみが再編成されます。

- 表に非パーティション索引が定義されていない場合 (システム生成された XML パス索引を除く)、指定されたパーティションのみが **ALLOW NO ACCESS** モードに制限されます。ユーザーは表の他のパーティションから読み取ったりそこに書き込んだりできます。
- 表に非パーティション索引が定義されている場合 (システム生成された XML パス索引を除く)、**ALLOW NO ACCESS** モードがデフォルトとなり、サポートされる唯一のアクセス・モードになります。この場合、その表は **ALLOW NO ACCESS** モードになります。

ALLOW READ ACCESS
再編成の際に表に対する読み取りアクセスだけを許可します。

ALLOW READ ACCESS モードが、非パーティション表のデフォルト・モードになります。

データ・パーティション表に対して **ON DATA PARTITION** 節が指定されている場合、指定されたデータ・パーティションのみが再編成されます。

- 表に非パーティション索引が定義されていない場合 (システム生成された XML パス索引を除く)、**ALLOW READ ACCESS** モードがデフォルト・モードとなり、指定されたパーティションのみがそのアクセス・モード・レベルに制限されます。ユーザーは表の他のパーティションから読み取ったりそこに書き込んだりできます。
- 表に非パーティション索引が定義されている場合 (システム生成された XML パス索引を除く)、**ALLOW READ ACCESS** モードはサポートされません。この場合に **ALLOW READ ACCESS** が指定されると、SQL1548N が戻されます (SQLSTATE 5U047)。

INPLACE

ユーザー・アクセスを許可しながら、表を再編成します。

表のインプレース再編成が可能なのは、非パーティションかつ非 MDC の表で、タイプ 2 の索引があり、拡張索引はなく、表内の XML 列に対して索引が定義されていない場合のみです。表のインプレース再編成は、サイズが少なくとも 3 ページである表に対してのみ実行できます。

表のインプレース再編成は非同期に発生するので、即時に有効にならないことがあります。

ALLOW READ ACCESS

再編成の際に表に対する読み取りアクセスだけを許可します。

ALLOW WRITE ACCESS

再編成の際に表に対する書き込みアクセスを許可します。これがデフォルトの動作です。

NOTRUNCATE TABLE

インプレース再編成の後に表を切り捨てないでください。切り捨ての際に、表は S ロックされます。

START

インプレース REORG 処理を開始します。これがデフォルトなので、このキーワードはオプションです。

STOP インプレース REORG 処理を現時点で停止します。

PAUSE

インプレース REORG を当面の間、中断または一時停止します。

RESUME

以前に一時停止した、表のインプレース再編成を継続または再開します。オンライン再編成が再開された時点で、再編成

の一時停止時と同じオプションを指定したい場合は、再開時にそれらのオプションを再び指定する必要があります。

USE *tbspace-name*

再編成されている表の一時コピーを保管する SYSTEM TEMPORARY 表スペースの名前を指定します。表スペースの名前を指定しない場合、データベース・マネージャーは、再編成しようとする表を含む表スペースにその表の作業用コピーを保管します。

8 KB、16 KB、または 32 KB の表オブジェクトの場合、指定した SYSTEM TEMPORARY 表スペースのページ・サイズが、表データの存在する表スペースのページ・サイズと一致していなければ、DB2 データベース製品は、LONG/LOB オブジェクトのサイズが正しい TEMPORARY 表スペースを検出しようとします。再編成が正常に実行されるためには、そのような表スペースが存在していなければなりません。

パーティション表の場合、表に含まれるデータ・パーティションの再編成において、TEMPORARY 表スペースが一時ストレージとして使用されます。パーティション表全体の再編成では、一度に 1 つのデータ・パーティションが再編成されます。 TEMPORARY 表スペースは、表全体ではなく、表に含まれる最大のデータ・パーティションを保持できる必要があります。 **ON DATA PARTITION** 節が指定される場合、TEMPORARY 表スペースは指定されたパーティションを保持できる必要があります。

パーティション表の表スペース名を指定しない場合、各データ・パーティションの存在する表スペースが、そのデータ・パーティションの一時ストレージとして使用されます。各データ・パーティションの表スペースには、そのデータ・パーティションのコピーが入るだけの十分なフリー・スペースがなければなりません。

INDEXSCAN

クラスタリング REORG では、索引スキャンが使用されて表レコードが再配列されます。索引を介して表にアクセスすることにより、表の行を再編成します。デフォルトの方法は、必要に応じて TEMPORARY 表スペースを使用しながら、表をスキャンして結果をソートし表を再編成することです。索引キーはソートの順序に配列していますが、スキャンおよびソートはまず索引から行 ID を読み取って行をフェッチするよりも通常は高速です。

LONGLOBDATA

長いフィールドおよび LOB データが再編成されます。

表に長い列または LOB 列が含まれる場合でも、これは必要ではありません。これは時間がかかり、クラスタリングを改善しないために、デフォルトではこれらのオブジェクトを再編成しません。しかし、XML 列を持つ表に対して **LONGLOBDATA** オプションを指定して再編成を実行すると、未使用のスペースが再利用されるため、XML ストレージ・オブジェクトのサイズが削減されます。

このパラメーターは、既存の LOB データをインライン化された LOB データに変換するときに必要です。

USE *longtbspace-name*

これはオプション・パラメーターであり、LONG データを再作成するために使用する TEMPORARY 表スペースの名前を指定するために使用できます。表オブジェクトについても LONG オブジェクトについても TEMPORARY 表スペースが指定されていない場合、現在それらのオブジェクトが存在している表スペース中にそれらのオブジェクトが構成されることになります。表の TEMPORARY 表スペースが指定されているが、このパラメーターが指定されていない場合には、ページ・サイズが異なるのでない限り、基本再編成データのために使用される表スペースが使用されることになります。ページ・サイズが異なる場合、DB2 データベース・システムは、LONG オブジェクトを作成するために適切なページ・サイズの一時コンテナーを選択することを試みます。

USE *longtbspace-name* が指定されている場合、**USE** *tbspace-name* も指定する必要があります。そうでない場合、*longtbspace-name* 引数は無視されます。

KEEPDICTIONARY

表の COMPRESS 属性が YES であり、表にコンプレッション・ディクショナリーがある場合、新しいディクショナリーは作成されません。再編成中に処理されるすべての行は、既存のディクショナリーを使用して圧縮されます。 COMPRESS 属性が YES であり、表にコンプレッション・ディクショナリーが存在しない場合、ディクショナリーは、表が特定のサイズ (およそ 1 から 2 MB) で、この表に十分のデータがある場合のみ、このシナリオで作成されます (さらに表が圧縮されます)。代わりに明示的に REORG RESETDICTIONARY を指定した場合には、表に少なくとも 1 行あれば、ディクショナリーが作成されます。表の COMPRESS 属性が NO であり、表にコンプレッション・ディクショナリーがある場合、再編成処理によりそのディクショナリーは保存され、新たに編成された表のすべての行は非圧縮形式になります。基本表の行に保管されない LOB などの一部のデータは圧縮できません。

LONGLOBDATA オプションを指定しない場合、表の行データだけが再編成されます。以下の表は、**LONGLOBDATA** オプションが指定されていないときの、REORG コマンドの **KEEPDICTIONARY** 構文の動作を説明しています。

表 64. REORG KEEPDICTIONARY

圧縮	ディクショナリーが存在するかどうか	結果と効果
Y	Y	ディクショナリーはそのまま、行圧縮
Y	N	ディクショナリーを作成、行圧縮
N	Y	ディクショナリーはそのまま、全行圧縮解除
N	N	影響なし、全行圧縮解除

以下の表は、**LONGLOBDATA** オプションが指定されているときの、REORG コマンドの **KEEPDICTIONARY** 構文の動作を説明しています。

表 65. *LONGLOBDATA* オプションが指定されているときの *REORG KEEPDICTIONARY*。

圧縮	表の行データ・ディクショナリーが存在する	XML ストレージ・オブジェクト・ディクショナリーが存在する[1]	コンプレッション・ディクショナリー	データ圧縮
Y	Y	Y	ディクショナリーを保存します。	既存のデータは、圧縮されます。新規データは、圧縮されます。
Y	Y	N	表の行ディクショナリーを保存し、XML ストレージ・オブジェクト・ディクショナリーを作成します。	既存のデータは、圧縮されます。新規データは、圧縮されます。
Y	N	Y	表の行ディクショナリーを作成し、XML ディクショナリーを保存します。	既存のデータは、圧縮されます。新規データは、圧縮されます。
Y	N	N	表の行ディクショナリーおよび XML ディクショナリーを作成します。	既存のデータは、圧縮されます。新規データは、圧縮されます。
N	Y	Y	表の行ディクショナリーおよび XML ディクショナリーを保存します。	表データは圧縮解除されます。新規データは、圧縮されません。
N	Y	N	表の行ディクショナリーを保存します。	表データは圧縮解除されます。新規データは、圧縮されません。
N	N	Y	XML ディクショナリーを保存します。	表データは圧縮解除されます。新規データは、圧縮されません。
N	N	N	影響なし。	表データは圧縮解除されます。新規データは、圧縮されません。

注:

1. コンプレッション・ディクショナリーは、XML 列が DB2 V9.7 以降の表に追加された場合または表が ONLINE_TABLE_MOVE ストアード・プロシージャーを使用してマイグレーションされた場合にのみ、表の XML ストレージ・オブジェクトに対して作成できます。

表の圧縮属性が NO の場合、(置換操作などで) 表の再初期設定または切り捨てが発生したなら、ディクショナリーが存在していればそれは廃棄されます。逆に、表の圧縮属性が YES の場合にディクショナリーが存在しているなら、切り捨てではディクショナリーが保たれ、廃棄は実行されません。リカバリーのことを考慮して、ま

た将来データ・キャプチャーの変更 (つまりレプリケーション) をサポートすることを考慮して、ディクショナリーは全体としてログ記録されます。

RESETDICTIONARY

表の COMPRESS 属性が YES の場合、新しい行コンプレッション・ディクショナリーが作成されます。再編成の際に処理されるすべての行は、この新しいディクショナリーによる圧縮の対象になります。前のすべてのディクショナリーは、このディクショナリーに置き換わります。表の COMPRESS 属性が NO であり、表にコンプレッション・ディクショナリーが存在している場合、再編成処理によりそのディクショナリーは除去され、新たに編成された表のすべての行は非圧縮形式になります。基本表の行に保管されない LOB などの一部のデータは圧縮できません。

LONGLOBDATA オプションを指定しない場合、表の行データだけが再編成されます。以下の表は、**LONGLOBDATA** オプションが指定されていないときの、REORG コマンドの **RESETDICTIONARY** 構文の動作を説明しています。

表 66. REORG RESETDICTIONARY

圧縮	ディクショナリーが存在するかどうか	結果と効果
Y	Y	ディクショナリー新規作成*、行圧縮。DATA CAPTURE CHANGES オプションが CREATE TABLE または ALTER TABLE ステートメントで指定されていると、現在のディクショナリーが保持されます (これは*履歴コンプレッション・ディクショナリー*と呼ばれます)。
Y	N	ディクショナリー新規作成、行圧縮
N	Y	ディクショナリー除去、全行圧縮解除。DATA CAPTURE NONE オプションが CREATE TABLE または ALTER TABLE ステートメントで指定されていると、指定された表の*履歴コンプレッション・ディクショナリー* も除去されます。
N	N	影響なし、全行圧縮解除

* - ディクショナリーが存在し、圧縮属性が有効になっていても、現在は表にデータがない場合は、**RESETDICTIONARY** 操作でも既存のディクショナリーが維持されます。内部の最小レコード長よりサイズの小さい行と、圧縮が試みられたときにレコード長の節減が図られない行は、このような場合は「十分でない」とみなされます。

以下の表は、**LONGLOBDATA** オプションが指定されているときの、REORG コマンドの **RESETDICTIONARY** 構文の動作を説明しています。

表 67. *LONGLOBDATA* オプションが指定されているときの *REORG RESETDICTIONARY*。

圧縮	表の行データ・ディクショナリーが存在する	XML ストレージ・オブジェクト・ディクショナリーが存在する[1]	データ・ディクショナリー	データ圧縮
Y	Y	Y	ディクショナリーの作成[2] [3]。	既存のデータは、圧縮されます。新規データは、圧縮されます。
Y	Y	N	新規の表の行ディクショナリーを作成し、新規の XML ディクショナリーを作成します[3]。	既存のデータは、圧縮されます。新規データは、圧縮されます。
Y	N	Y	表の行データ・ディクショナリーを作成し、新規 XML ディクショナリーを作成します。	既存のデータは、圧縮されます。新規データは、圧縮されます。
Y	N	N	ディクショナリーを作成します。	既存のデータは、圧縮されます。新規データは、圧縮されます。
N	Y	Y	ディクショナリーを除去します。既存および新規のデータは圧縮されません。	既存の表データは圧縮解除されます。新規データは、圧縮されません。
N	Y	N	表の行ディクショナリーを除去します。すべてのデータは圧縮解除されます。	既存の表データは圧縮解除されます。新規データは、圧縮されません。
N	N	Y	XML ストレージ・オブジェクト・ディクショナリーを除去します。	既存の表データは圧縮解除されます。新規データは、圧縮されません。
N	N	N	影響なし。	既存の表データは圧縮解除されます。新規データは、圧縮されません。

注:

1. コンプレッション・ディクショナリーは、XML 列が DB2 V9.7 以降の表に追加された場合または表が Online Table Move を使用してマイグレーションされた場合にのみ、表の XML ストレージ・オブジェクトに対して作成できます。
2. ディクショナリーが存在していて圧縮属性が有効であるが、現在のところ表中にデータが存在しない場合、**RESETDICTIONARY** 操作では既存のディクショナリーがそのまま保たれます。この場合、不適切とみなされるのは、内部最小レコード長よりもサイズが小さい行、および圧縮してもレコード長の節約にならない行です。

3. DATA CAPTURE CHANGES オプションが CREATE TABLE または ALTER TABLE ステートメントで指定されていると、現在のデータ・ディクショナリーが保持されます (これは*履歴コンプレッション・ディクショナリー*と呼ばれます)。

ON DATA PARTITION *partition-name*
データ・パーティション表では、再編成するデータ・パーティションを指定します。

DB2 V9.7 フィックスパック 1 以降のリリースでは、この節を REORG INDEXES ALL コマンドで使用して、特定のパーティションのパーティション索引を再編成することができます。また、REORG TABLE コマンドで使用して、特定のパーティションのデータを再編成することもできます。

パーティション表で REORG TABLE または REORG INDEXES ALL コマンドでこの節を使用するときに、パーティション *partition-name* が指定した表にない場合、再編成は失敗し、理由コード 1 と共に SQL2222N を戻します。パーティション *partition-name* がアタッチまたはデタッチ状態の場合、再編成は失敗し、理由コード 3 と共に SQL2222N を戻します。

ON DATA PARTITION 節を指定して REORG INDEX コマンドが発行された場合、再編成は失敗し、理由コード 2 と共に SQL2222N を戻します。

パーティション表が REORG ペンディング状態にあり、かつ表に非パーティション索引が定義されている場合、REORG TABLE コマンドは失敗し、SQL1549N (SQLSTATE 5U047) を戻します。

ALL DBPARTITIONNUMS
db2nodes.cfg ファイルに指定されているすべてのデータベース・パーティションで操作が実行されることを指定します。ノード節が指定されていない場合、これがデフォルトです。

EXCEPT
ノード・リストに指定されているものを除き、db2nodes.cfg ファイルに指定されているすべてのデータベース・パーティションで操作が実行されることを指定します。

ON DBPARTITIONNUM | ON DBPARTITIONNUMS
データベース・パーティションのセットに対して操作を実行します。

db-partition-number1
データベース・パーティション・リスト内のデータベース・パーティション番号を指定します。

db-partition-number2
2 番目のデータベース・パーティション番号を指定し、*db-partition-number1* から *db-partition-number2* までのすべてのデータベース・パーティションがデータベース・パーティション・リストに含まれるようにします。

例

データベース・パーティション 1、3、および 4 で構成されるデータベース・パーティション・グループ内の表を再編成します。

```
CALL SYSPROC.ADMIN_CMD ('REORG TABLE employee
   INDEX empid ON DBPARTITIONNUM (1,3,4)')
```

使用上の注意

制約事項:

- コマンドの実行状況は、CALL ステートメントからの結果である SQLCA で戻されます。
- REORG ユーティリティーは、操作の開始時に COMMIT ステートメントを発行しますが、これによってタイプ 2 接続の場合に、プロシージャーは SQL30090N、理由コード 2 を戻します。
- REORG ユーティリティーでは、ニックネームの使用はサポートされません。
- REORG TABLE コマンドは、宣言済み一時表または作成済み一時表に対してサポートされていません。
- REORG TABLE コマンドは、ビューに対しては使用できません。
- 表の再編成は、範囲クラスター化表との互換性がありません。表の範囲領域は常に、クラスター化されているからです。
- DMS 表スペース中のパーティション表が属している表スペース (LOB や索引を含む) のオンライン・バックアップが実行されている間は、REORG TABLE をその表に対して使用することはできません。
- REORG TABLE は、索引拡張子に基づく索引を使用できません。
- 表が REORG ペンディング状態なら、その表に対してインプレース再編成を実行することはできません。
- 同じ TEMPORARY DMS 表スペースの共有による表の再編成の同時実行はサポートされていません。
- データ・パーティション表では、以下のようになります。
  - SYSCAT.TABLES の中でその表の ACCESS_MODE はフル・アクセス権限でなければなりません。
  - 再編成では、アタッチまたはデタッチ操作のため、制限状態にあるデータ・パーティションがスキップされます。**ON DATA PARTITION** 節が指定されている場合、そのパーティションは完全にアクセス可能でなければなりません。
  - 表の再編成中にエラーが発生した場合、一部の索引または索引パーティションは無効のままとなる可能性があります。表の非パーティション索引は、再編成が最初のデータ・パーティションの置換フェーズに達するかそれをパスすると、無効のマークが付きます。置換フェーズに既に達したかまたはそれを通過したデータ・パーティションの索引パーティションには、無効のマークが付きます。索引は次回表またはデータ・パーティションにアクセスしたときに再作成されます。
  - ALLOW NO ACCESS モードが使用されている場合、索引の再編成中にエラーが発生すると、表の一部の索引が無効のままになる場合があります。表の非パーティション RID 索引については、障害発生時点で再編成中の索引のみが無効のままになります。非パーティション・ブロック索引がある MDC 表については、エラーが発生した場合、1 つ以上のブロック索引が無効のままになる可能性があります。パーティション索引の場合、再編成中のデータ・パーティシ

ョンの索引オブジェクトだけが無効のままになります。無効のマークが付いたすべての索引は、次回表またはデータ・パーティションにアクセスしたときに再作成されます。

– 表に定義されたパーティション索引のみを使用するデータ・パーティション表が REORG ペンディング状態にある場合、ON DATA PARTITION 節を指定して **REORG TABLE** コマンドを発行すると、指定されたデータ・パーティションのみ REORG ペンディング状態ではなくなります。その表の残りのパーティションを REORG ペンディング状態から戻すためには、表全体に対して REORG TABLE コマンド (**ON DATA PARTITION** 節を使用しないで) を発行するか、または残っているパーティションそれぞれに対して ON DATA PARTITION 節を指定して **REORG TABLE** コマンドを発行します。

表の再編成の現在の進行状態に関する情報は、データベース活動の履歴ファイルに書き込まれます。履歴ファイルには、再編成イベントごとの記録が含まれています。このファイルを表示するには、再編成している表を含むデータベースに対して LIST HISTORY コマンドを実行します。

さらに、表スナップショットを使用して表の再編成の進行状況をモニターすることもできます。表の再編成のモニター・データは、「データベース・モニター表スイッチ (Database Monitor Table Switch)」の設定値に関係なく記録されます。

エラーが生じた場合、SQLCA ダンプが履歴ファイルに書き込まれます。表のインプレース再編成の場合、状況が PAUSED として記録されます。

索引付き表が何回も変更されると、索引内のデータがフラグメント化されることがあります。表が索引に関してクラスター化されている場合、表および索引をクラスターの順序から取り出すことができます。これら両方の要素は索引を使用するスキャンのパフォーマンスを低下させ、索引ページの事前取り出しの効果に影響を与えることがあります。 REORG INDEX または REORG INDEXES を使用して、表の索引の 1 つまたはすべてを再編成できます。索引を再編成すると、フラグメントが除去され、物理クラスタリングがリーフ・ページにリストアされます。
REORGCHK を使用すると、索引に再編成が必要かどうかを判別するために役立ちます。すべてのデータベース操作が完了し、すべてのロックを解放したことを確かめてから、索引の再編成を呼び出してください。これは、WITH HOLD でオープンされた、すべてのカーソルをクローズした後で COMMIT または ROLLBACK を発行することによって行われます。

従来の表の再編成 (オフライン再編成) では、再編成の最後のフェーズで索引が再作成されます。 TEMPORARY 表スペースが複数存在する場合は、表の再編成処理に伴って追加のソートを行う際に、REORG TABLE コマンドで指定した TEMPORARY 表スペースに加えて、それ以外の TEMPORARY 表スペースが 1 つ使用される可能性もあります。ただし、表のインプレース再編成 (オンライン再編成) では索引は再作成されません。インプレース表再編成の完了後に、REORG INDEXES コマンドを発行することをお勧めします。インプレース表再編成は非同期であるため、REORG INDEXES コマンドを発行する前にインプレース表再編成が完了していることを確認するように注意しなければなりません。インプレース表再編成が完了する前に REORG INDEXES コマンドを発行すると、再編成が失敗するおそれがあります (SQLCODE -2219)。

何回も修正されてデータがフラグメント化して、アクセス・パフォーマンスが大幅に低下した表も REORG TABLE コマンドの対象になります。構造化タイプ列のインラインの長さを変更後、このユーティリティーもまた呼び出して、変更の有用性を確認してください。 REORGCHK コマンドを使用して、表の再編成が必要であるかどうか判別してください。すべてのデータベース操作が完了し、すべてのロックが解放されていることを確かめてから、REORG TABLE を呼び出してください。これは、WITH HOLD でオープンされた、すべてのカーソルをクローズした後で COMMIT または ROLLBACK を発行することによって行われます。表の再編成の後で、RUNSTATS を使用して表統計を更新し、 REBIND を使用してこの表を使用するパッケージを再バインドします。再編成ユーティリティーは、暗黙的にすべてのカーソルをクローズします。

DB2 V9.7 フィックスパック 1 以降では、1 つのデータ・パーティション表に対して REORG TABLE コマンドおよび REORG INDEXES ALL コマンドを発行し、同時に複数のデータ・パーティション、または 1 つのパーティション上の複数のパーティション索引を再編成することができます。複数のデータ・パーティション、または 1 つのパーティション上の複数のパーティション索引を並行して再編成する場合、ユーザーは影響を受けないパーティションにはアクセスできますが、影響を受けるパーティションにはアクセスできません。同一表に対して同時に操作する複数の REORG コマンドを発行するには、以下の基準をすべて満たす必要があります。

- 各 REORG コマンドは、**ON DATA PARTITION** 節を指定して、それぞれ別個のパーティションを指定しなければならない。
- 各 REORG コマンドは、**ALLOW NO ACCESS** モードを使用して、データ・パーティションに対するアクセスを制限しなければならない。
- REORG TABLE コマンドを発行する場合、パーティション表にはパーティション索引のみが含まれていなければならない。非パーティション索引が表に定義されていてはなりません (システム生成された XML パス索引を除く)。

非パーティション索引がなく (システムが生成した XML パス索引を除く)、パーティション P1、P2、P3、P4 を使用するパーティション表 T1 の場合、以下の複数の REORG コマンドを並行して実行することができます。

```
REORG INDEXES ALL ALLOW NO ACCESS ON DATA PARTITION P1
REORG TABLE ALLOW NO ACCESS ON DATA PARTITION P2
REORG INDEXES ALL ALLOW NO ACCESS ON DATA PARTITION P3
```

並行 REORG コマンドを使用する場合、以下のような操作はサポートされません。

- **ON DATA PARTITION** 節を指定せずに、表に対して REORG コマンドを使用する。
- データ・パーティションを追加、アタッチ、またはデタッチするのに、表で ALTER TABLE ステートメントを使用する。
- データを表にロードする。
- 表を含むオンライン・バックアップを実行する。

表の値圧縮をアクティブ化または非アクティブ化したために表に混合した行形式が含まれている場合、オフラインで表を再編成することによって、既存の行すべてをターゲットの行形式に変換することができます。

表がいくつかのデータベース・パーティションに分散している場合、影響を受けるデータベース・パーティションのいずれかで表または索引の再編成が失敗すると、失敗したデータベース・パーティションでのみ表または索引の再編成がロールバックされます。

再編成が成功しなかった場合には、一時ファイルを削除すべきではありません。データベース・マネージャーは、これらのファイルを使用し、データベースをリカバリーします。

索引の名前が指定されると、データベース・マネージャーはその索引の順番に従って、データを再編成します。パフォーマンスを最大にするため、SQL 照会で頻繁に使用される索引を指定してください。索引の名前が指定されておらず、クラスタリング索引が存在する場合、データはクラスタリング索引に従って配列されます。

表の PCTFREE 値は、ページごとに指定されたフリー・スペースの量を決定します。この値が設定されていない場合、ユーティリティーは、それぞれのページにできるだけ大きなスペースを割り当てます。

表の再編成の後に表スペースのロールフォワード・リカバリーを完了させるには、通常の表スペースと LARGE 表スペースの両方で、ロールフォワード・リカバリーが有効になっていなければなりません。

その表が、**COMPACT** オプションを使用しない LOB 列を含む場合、 LOB DATA ストレージ・オブジェクトは、表の再編成に従いかなり大きくなることができます。これは、行が再編成された順序、および使用される (SMS または DMS) 表スペースのタイプの結果になります。

REORG INDEXES/TABLE コマンドにより XML データに対する索引が再作成される場合があります。詳細については、『XML データに対する索引の再作成』を参照してください。

RESET ALERT CONFIGURATION コマンド (ADMIN_CMD プロシージャーを使用)

特定のオブジェクトのヘルス・インディケーター設定を、そのオブジェクト・タイプの現行デフォルトにリセットするか、またはオブジェクト・タイプの現行のデフォルトのヘルス・インディケーター設定を、インストール時のデフォルトにリセットします。

重要: バージョン 9.7 でヘルス・モニターが非推奨になったため、このコマンドまたは API は推奨されておらず、将来のリリースで除去される可能性があります。詳しくは、「*DB2 バージョン 9.7 の新機能*」のトピック『ヘルス・モニターが推奨されなくなった』を参照してください。

許可

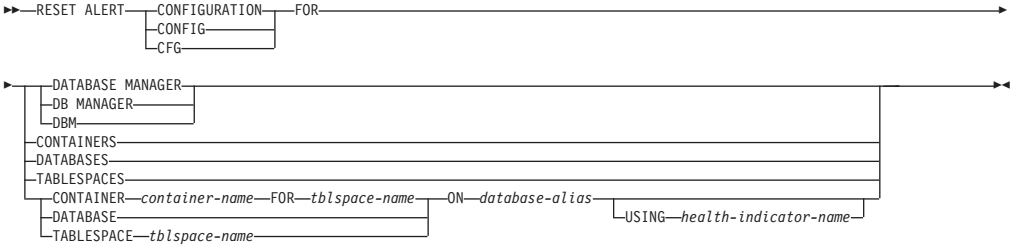
以下のいずれか。

- SYSADM
- SYSMAINT
- SYSCTRL

必要な接続

データベース

コマンド構文



コマンド・パラメーター

DATABASE MANAGER | DB MANAGER | DBM
データベース・マネージャーでアラート設定をリセットします。

CONTAINERS
データベース・マネージャーが管理するすべての表スペース・コンテナーのデフォルトのアラート設定を、インストール時のデフォルトにリセットします。これは、カスタム設定を持たないすべての表スペース・コンテナーに適用される設定です。カスタム設定は、**CONTAINER** *container-name* **FOR** *tblspace-name* **ON** *database-alias* 節を使って定義されます。

DATABASES
データベース・マネージャーが管理するすべてのデータベースのアラート設定をリセットします。これは、カスタム設定を持たないすべてのデータベースに適用される設定です。カスタム設定は、**DATABASE ON** *database-alias* 節を使って定義されます。

TABLESPACES
データベース・マネージャーが管理するすべての表スペースのデフォルトのアラート設定を、インストール時のデフォルトにリセットします。これは、カスタム設定を持たないすべての表スペースに適用される設定です。カスタム設定は、**TABLESPACE** *tblspace-name* **ON** *database-alias* 節を使って定義されます。

CONTAINER *container-name* **FOR** *tblspace-name* **ON** *database-alias*
ON *database-alias* 節を使って指定したデータベース上で、**FOR** *tblspace-name* 節を使って指定した表スペースの、*container-name* という名前の表スペース・コンテナーのアラート設定をリセットします。この表スペース・コンテナーにカスタム設定がある場合、これらの設定は除去され、現行の表スペース・コンテナーのデフォルトが使用されます。

DATABASE ON *database-alias*
ON *database-alias* 節を使って指定したデータベースのアラート設定をリセットします。このデータベースにカスタム設定がある場合、これらの設定は除去され、インストール時のデフォルトが使用されます。

TABLESPACE *tblspace-name* **ON** *database-alias*
ON *database-alias* 節を使って指定したデータベース上で、*tblspace-name* と

いう名前の表スペースのアラート設定をリセットします。この表スペースにカスタム設定がある場合、これらの設定は除去され、インストール時のデフォルトが使用されます。

USING *health-indicator-name*
アラート構成がリセットされるヘルス・インディケーターのセットを指定します。ヘルス・インディケーター名は 2 文字のオブジェクト ID で構成され、その後にインディケーターの測定対象を説明する名前が続きます。以下に例を示します。

```
db.sort_privmem_util
```

このオプションを指定しない場合は、指定したオブジェクトまたはオブジェクト・タイプのすべてのヘルス・インディケーターがリセットされます。

例

ADMIN_CMD プロシージャーが含まれているデータベースを所有するデータベース・マネージャーのアラートの設定値をリセットします。

```
CALL SYSPROC.ADMIN_CMD( 'reset alert cfg for dbm' )
```

使用上の注意

コマンドの実行状況は、CALL ステートメントからの結果である SQLCA で戻されます。

ADMIN_CMD プロシージャーはサーバーでのみ実行されるため、*database-alias* は、サーバー上のカタログの中で定義されているローカル・データベースでなければなりません。

RESET DATABASE CONFIGURATION コマンド (ADMIN_CMD プロシージャーを使用)

特定データベースの構成をシステム・デフォルトにリセットします。

有効範囲

このコマンドは、アプリケーションの接続先であるデータベース・パーティションにのみ影響を与えます。

許可

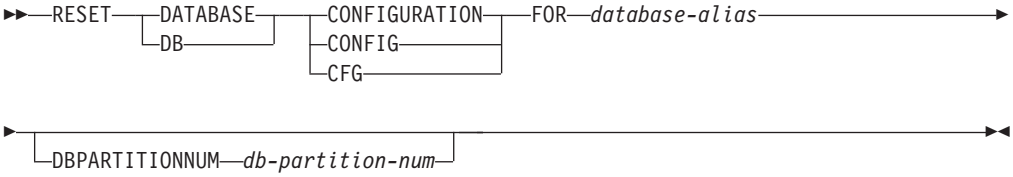
以下のいずれか。

- SYSADM
- SYSCTRL
- SYSMAINT

必要な接続

データベース

コマンド構文



コマンド・パラメーター

FOR *database-alias*

構成がシステム・デフォルトにリセットされるそのデータベースの別名を指定します。データベース別名は、サーバー上のカタログの中で定義されているものでなければならず、サーバー上のローカル・データベースを指していなければなりません。

DBPARTITIONNUM *db-partition-num*

データベース構成のリセットが特定のデータベース・パーティションに適用される場合、このパラメーターが使用されることがあります。このパラメーターが指定されていない場合、リセットはすべてのデータベース・パーティションに対して有効です。

例

サーバー上で別名 SAMPLE でカタログされているデータベースの構成をリセットします。

```
CALL SYSPROC.ADMIN_CMD( 'reset db cfg for SAMPLE' )
```

使用上の注意

データベース構成パラメーターのリストを表示または印刷するには、SYSIBMADM.DBCFG 管理ビューを使用してください。

構成可能なパラメーターの値を変更するには、UPDATE DATABASE CONFIGURATION コマンドを使用してください。

データベース構成ファイルへの変更は、ファイルがメモリーにロードされた後にのみ有効になります。これを行う前にすべてのアプリケーションはデータベースから切断されている必要があります。

エラーが発生した場合、データベース構成ファイルは変更されません。

チェックサムが無効である場合には、データベースの構成ファイルは、リセットできません。適当なコマンドを使用しないでデータベース構成ファイルを変更するとこれが発生することがあります。これが発生する場合、データベースをリストアしてデータベース構成ファイルをリセットする必要があります。

RESET DATABASE CONFIGURATION コマンドを実行すると、データベース構成パラメーターが文書化されているデフォルトの構成値にリセットされ、

auto_runstats は ON になります。 **Self_tuning_mem** は、非パーティション・データベース環境の場合は ON にリセットされ、パーティション・データベース環境の場合は OFF にリセットされます。

コマンドの実行状況は、CALL ステートメントからの結果である SQLCA で戻されます。

ADMIN_CMD プロシージャーはサーバーでのみ実行されるため、*database-alias* は、サーバー上のカタログの中で定義されているローカル・データベースでなければなりません。

RESET DATABASE MANAGER CONFIGURATION コマンド (ADMIN_CMD プロシージャーを使用)

現在接続されているデータベースを含むインスタンスについて、データベース・マネージャーの構成ファイルのパラメーターをシステム・デフォルトにリセットします。この値は、ノード・タイプに基づいたデフォルト値にリセットされます。

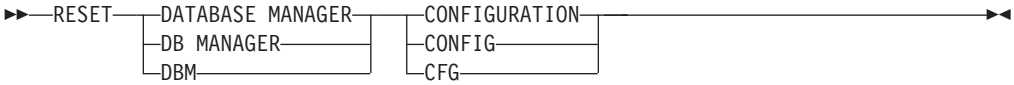
許可

SYSADM

必要な接続

データベース

コマンド構文



コマンド・パラメーター

なし

例

ADMIN_CMD ストアード・プロシージャーの属するデータベースを含むインスタンスの構成をリセットします。

```
CALL SYSPROC.ADMIN_CMD( 'reset dbm cfg' )
```

使用上の注意

このコマンドは、インストール・プログラムによって設定されたすべてのパラメーターをリセットします。パラメーターがリセットされると、DB2 を再始動するときにエラー・メッセージが戻される原因となる場合があります。例えば、**svcename** パラメーターがリセットされると、 DB2 を再始動しようとすると、ユーザーは SQL5043N エラー・メッセージを受け取ります。

このコマンドを実行する前に、既存の設定値を参照できるようにするために、SYSIBMADM.DBMCFG 管理ビューからの出力をファイルに保存します。個々の設

定値は、ADMIN_CMD プロシージャーを通じて UPDATE DATABASE MANAGER CONFIGURATION コマンドを使用して更新できます。

インストール・プログラムで設定される **svcename** パラメーターの、ユーザーによる修正は推奨されません。

データベース・マネージャー構成パラメーターのリストを表示または印刷するには、SYSIBMADM.DBMCFG 管理ビューを使用してください。構成可能なパラメーターの値を変更するには、ADMIN_CMD プロシージャーを通じて UPDATE DATABASE MANAGER CONFIGURATION コマンドを使用してください。

これらのパラメーターについての詳細は、構成パラメーターおよび個々のパラメーターについてのサマリー・リストを参照してください。

データベース・マネージャー構成ファイルへの変更の一部は、ファイルがメモリーにロードされた後にのみ有効になります。オンラインで構成できるパラメーターと構成できないパラメーターについては、構成パラメーターの一覧をご覧ください。即時にリセットされないサーバー構成パラメーターは、 db2start の実行中にリセットされます。クライアント構成パラメーターの場合、パラメーターは次にアプリケーションを開始するときにリセットされます。クライアントがコマンド行プロセッサーである場合は、 TERMINATE を呼び出すことが必要です。

エラーが生じた場合には、データベース・マネージャー構成ファイルは変更されません。

データベース・マネージャー構成ファイルは、そのチェックサムが無効であると、リセットすることができません。このような状況は、適切なコマンドを使用せずに手動で構成ファイルが編集された場合などに発生します。チェックサムが無効な場合は、インスタンスを再作成する必要があります。

REWIND TAPE コマンド (ADMIN_CMD プロシージャーを使用)

ストリーミング磁気テープ装置へのバックアップおよびリストア操作のためにテープを巻き戻します。このコマンドは Windows オペレーティング・システムでのみサポートされています。

許可

以下のいずれかです。

- SYSADM
- SYSCTRL
- SYSMAINT

必要な接続

データベース

コマンド構文

コマンド・パラメーター

ON *device*

有効なテープ装置名を指定します。デフォルト値は ¥¥.¥TAPE0 です。装置の指定は、サーバーに対する相対指定でなければなりません。

例

'¥¥.¥TAPE1' という装置のテープを巻き戻します。

```
CALL SYSPROC.ADMIN_CMD( 'rewind tape on ¥¥.¥TAPE1' )
```

使用上の注意

コマンドの実行状況は、CALL ステートメントからの結果である SQLCA で戻されます。

RUNSTATS コマンド (ADMIN_CMD プロシージャーを使用)

表やそれに関連した索引の特性、あるいは統計ビューの特性に関する統計情報を更新します。これらの特性には、レコード数、ページ数、および平均レコード長が含まれます。オプティマイザーは、データへのアクセス・パスを判別するとき、これらの統計を使用します。

表の場合、このユーティリティーは、表が数多く更新されるとき、または表を再編成した後で、呼び出してください。統計ビューにおいては、基礎表に対する変更が、ビューによって戻される行に対して実質的に影響を与える場合に、このユーティリティーを呼び出す必要があります。そのビューは、それ以前に ALTER VIEW ステートメントを使用して、照会最適化で使用できる状態になっていなければなりません。

有効範囲

このコマンドは、db2nodes.cfg ファイル中のどのデータベース・パーティションからでも発行できます。この API は、カタログ・データベース・パーティション上のカタログを更新するために使用できます。

表の場合、このコマンドは、呼び出し元のデータベース・パーティションの表の統計を収集します。表がそのデータベース・パーティションに存在しない場合、データベース・パーティション・グループの最初のデータベース・パーティションが選択されます。

ビューの場合、このコマンドは、関連するすべてのデータベース・パーティションに含まれる表のデータを使用して、統計情報を収集します。

許可

表の場合、以下のいずれか 1 つです。

- SYSADM
- SYSCTRL
- SYSMAINT
- DBADM

- SQLADM
- 表に対する CONTROL 特権
- LOAD authority

このコマンドを使用する際には、接続内に存在する宣言された一時表のいずれにおいても明示特権は必要ありません。

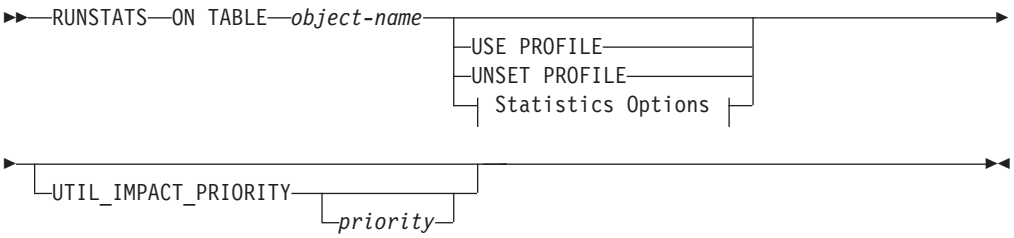
統計ビューの場合、以下のいずれか 1 つです。

- SYSADM
- SYSCTRL
- SYSMAINT
- DBADM
- SQLADM
- 統計ビューに対する CONTROL 特権

必要な接続

データベース

コマンド構文



Statistics Options:

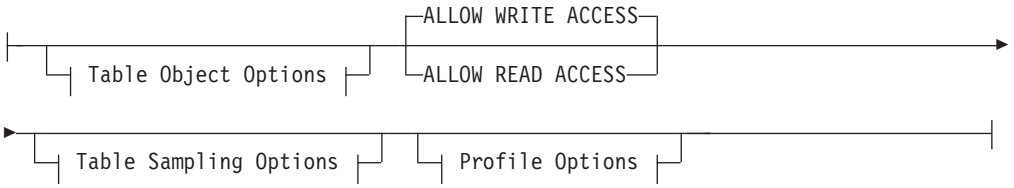


Table Object Options:

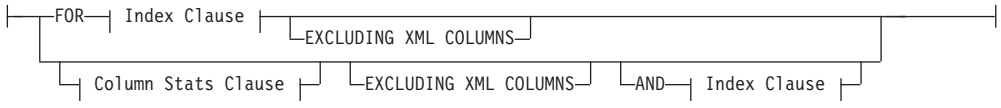
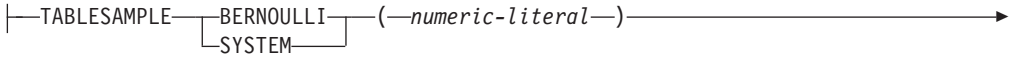
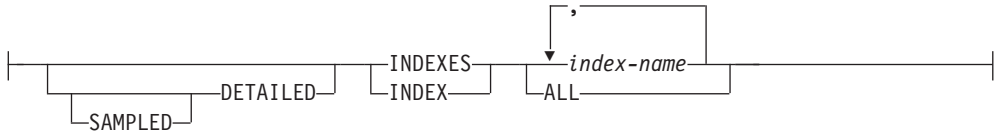


Table Sampling Options:

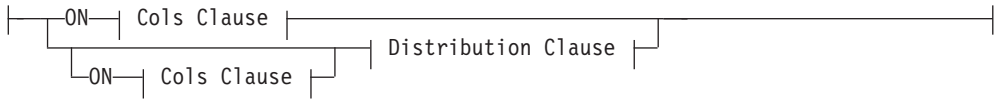


Profile Options:

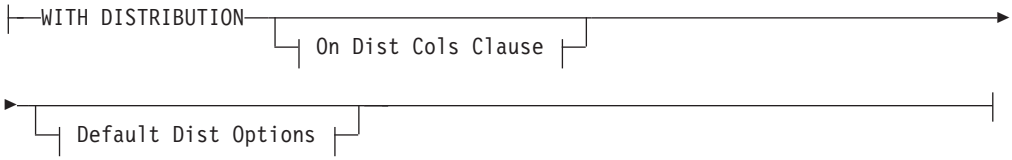
Index Clause:



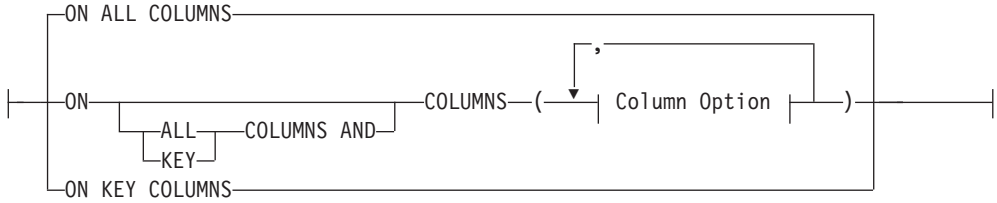
Column Stats Clause:



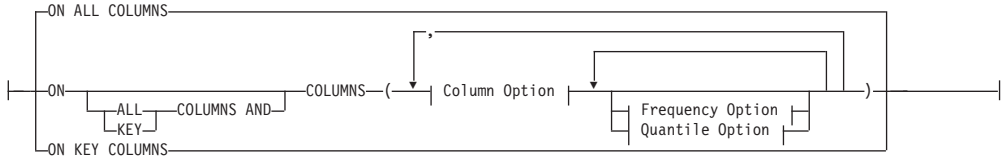
Distribution Clause:



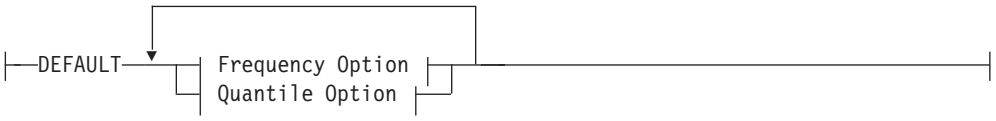
On Cols Clause:



On Dist Cols Clause:



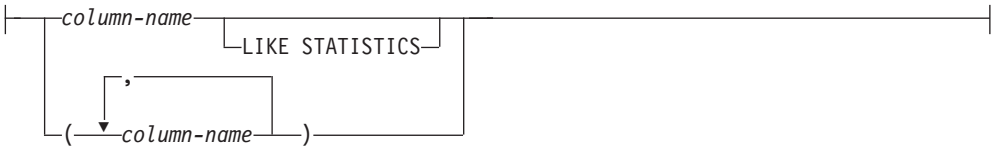
Default Dist Option:



Frequency Option:

Quantile Option:

Column Option:



コマンド・パラメーター

object-name
統計が収集される表または統計ビューを識別します。それは、階層表であってはなりません。型付き表の場合は、*object-name* は表階層のルート表の名前でなければなりません。 *schema.object-name* という形式の完全修飾名または別名を使用しなければなりません。schema には、表作成時のユーザー名が入ります。

index-name
表で定義されている既存の索引を識別します。 *schema.index-name* 形式の完全修飾名を使用しなければなりません。ビューの場合、このオプションは使用できません。

USE PROFILE
このオプションを使用した RUNSTATS は、以前に保管した統計プロファイルを使用して表または統計ビューの統計を収集します。統計プロファイルの作成には、**SET PROFILE** オプションを使用し、更新には **UPDATE PROFILE** オプションを使用します。

UNSET PROFILE
既存の統計プロファイルを除去する場合は、このオプションを指定します。以下に例を示します。

```
RUNSTATS ON tablemyschema.mytable UNSET PROFILE
```

FOR INDEXES
索引のみの統計を収集および更新します。表に、以前に収集された表統計が

ない場合は、基本表統計も収集されます。これらの基本統計には、分散統計は一切含まれません。ビューの場合、このオプションは使用できません。

AND INDEXES
表と索引両方の統計を収集および更新します。ビューの場合、このオプションは使用できません。

DETAILED
拡張された索引統計を計算します。これは CLUSTERFACTOR および PAGE_FETCH_PAIRS 統計で、比較的大規模な索引の場合に収集されます。ビューの場合、このオプションは使用できません。

SAMPLED
このオプションは、**DETAILED** オプションと合わせて使用することにより、拡張された索引統計をコンパイルする際に RUNSTATS で CPU のサンプリング方式を使用できるようにします。オプションが指定されていないときは、拡張された索引統計を計算するために、索引内のすべての項目が調べられます。ビューの場合、このオプションは使用できません。

ON ALL COLUMNS
適格なすべての列で統計を収集するには、**ON ALL COLUMNS** 節を使用します。列は、基本統計の収集に指定するか (On Cols 節)、または **WITH DISTRIBUTION** 節と組み合わせて指定 (On Dist Cols 節) できます。これら列固有の節がどちらも指定されない場合は、デフォルト・オプションとして **ON ALL COLUMNS** が指定されます。

なお、On Cols 節でこれが指定された場合は、特定の列が **WITH DISTRIBUTION** 節の一部として選択されない限り、すべての列で基本列統計だけが収集されます。 **WITH DISTRIBUTION** 節の一部として指定された列では、基本統計も分散統計も収集されます。

WITH DISTRIBUTION ON ALL COLUMNS が指定されている場合は、収集が行えるすべての列で基本統計と分散統計の両方が収集されます。 On Cols 節での指定はすべて重複になるため、必要ありません。

ON COLUMNS
この節では、統計の収集を行う列のリストを任意に指定できます。列のグループを指定した場合は、そのグループについて重複のない異なる値が収集されます。索引統計情報を収集することなく表に対して RUNSTATS を実行し、統計情報収集の対象として列のサブセットを指定した場合、

1. RUNSTATS コマンドに指定されていないが、索引の最初の列である列の統計情報は、リセットされません。
2. RUNSTATS コマンドで指定されていない他のすべての列の統計情報は、リセットされます。

この節は、On Cols 節と On Dist Cols 節の中で使用できます。列のグループに関する分散統計の収集は、現在サポートされていません。

列グループの中に XML タイプの列が指定されている場合は、そのグループについて重複のない異なる値を収集するため、XML タイプのそれらの列は無視されます。しかし、その列グループ中の XML タイプ列について、XML 列の基本統計情報は収集されます。

EXCLUDING XML COLUMNS
この節を使用すると、XML タイプのすべての列を、統計情報収集から除外することができます。 XML データを含めた場合、必要となるシステム・リソースが増加することがあるため、この節を使用すれば、XML 以外の列に関する統計情報の収集が容易になります。 **EXCLUDING XML COLUMNS** 節は、統計情報収集の対象として XML 列を指定する他の節よりも優先されます。例えば、**EXCLUDING XML COLUMNS** 節を使用する場合、**ON COLUMNS** 節で XML タイプの列を指定したり **ON ALL COLUMNS** 節を使用したりしても、統計情報収集において XML タイプの列はすべて無視されます。 DB2 V9.7 フィックスパック 1 以降のリリースでは、この節が指定される場合、XML タイプの列に対する分散統計は収集されません。

ON KEY COLUMNS
特定の列をリストする代わりに、表で定義されたすべての索引を構成する列の統計を収集することもできます。ここでは、照会に含まれる重要な列が、表での索引の作成にも使用されることが前提となっています。表に索引がない場合は、列がリストされず、列統計が収集されない場合と同様になります。これは、`On Cols` 節または `On Dist Cols` 節の中で使用できます。ただし、その両方の節で指定すると、**WITH DISTRIBUTION** 節で基本統計と分散統計の両方の収集が指定されているため、`On Cols` 節で重複が生じます。定義により XML タイプの列はキー列ではなく、**ON KEY COLUMNS** 節による統計情報収集には含められません。ビューの場合、このオプションは使用できません。

column-name
表または統計ビューの中の列の名前。存在しない列が指定された場合や列名の入力を誤った場合など、統計収集を行えない列の名前が指定された場合は、エラー (-205) が戻されます。一方は配分なし、一方は配分ありで、2 つの列のリストを指定できます。 **WITH DISTRIBUTION** 節が関連付けられていないリストで列を指定する場合は、基本列統計だけが収集されます。列が両方のリストに含まれている場合は、分散統計が収集されます (**NUM_FREQVALUES** および **NUM_QUANTILES** がゼロに設定されていない限り)。

NUM_FREQVALUES
収集の頻度を示す値の最大値を定義します。これは、**ON COLUMNS** 節の中で、個々の列ごとに指定できます。個々の列に対して値が指定されない場合は、**DEFAULT** 節で指定されている頻度のしきい値が選出されます。どちらも指定されていない場合は、**num_freqvalues** データベース構成パラメーターで設定されている値が、収集の頻度を指定する値の最大値になります。

NUM_QUANTILES
収集する分散変位値の最大値を定義します。これは、**ON COLUMNS** 節の中で、個々の列ごとに指定できます。個々の列に対して値が指定されない場合は、**DEFAULT** 節で指定されている変位値のしきい値が選出されます。どちらも指定されていない場合は、**num_quantiles** データベース構成パラメーターで設定されている値が、収集する分位値の最大個数になります。

DB2 V9.7 フィックスパック 1 以降のリリースでは、 XML データに対する各索引の分散統計は、デフォルトで最大 250 の分位数を使用します。このデフォルトは、**ON COLUMNS** 節または **DEFAULT** 節の **NUM_QUANTILES** パラメーターを指定して変更することができます。XML 分散統計を収集している間、**num_quantiles** データベース構成パラメーターは無視されます。

WITH DISTRIBUTION

この節は、指定された列で基本統計と分散統計の両方を収集することを指定します。 **ON COLUMNS** 節が指定されていない場合は、表または統計ビューの中のすべての列 (CLOB や LONG VARCHAR といった、収集用には選択できない列を除く) で分散統計が収集されます。一方 **ON COLUMNS** 節が指定されている場合は、提供された列リストに対してのみ (統計収集用には選択できない列を除く) 分散統計が収集されます。なお、節が指定されなければ、基本統計だけが収集されます。

列のグループに関する分散統計の収集は、現在サポートされていません。**WITH DISTRIBUTION ON COLUMNS** 節で列のグループが指定された場合は、分散統計は収集されません。

DEFAULT

NUM_FREQVALUES または **NUM_QUANTILES** を指定した場合、これらの値を **ON COLUMNS** 節で個々の列に対して指定していなければ、これらの値によって、すべての列に関して収集する度数と分位の統計の最大個数が決まります。 **DEFAULT** 節が指定されない場合は、対応するデータベース構成パラメーターにある値が使用されます。

LIKE STATISTICS

このオプションを指定すると、付加的な列統計が収集されます。収集されるのは、SYSSTAT.COLUMNS の SUB_COUNT および SUB_DELIM_LENGTH 統計です。これらの統計は、1 バイト文字セット (SBCS)、FOR BIT DATA、または UTF-8 のコード・ページ属性を持つタイプ CHAR および VARCHAR の列について収集されます。タイプ "column LIKE '%xyz'" および "column LIKE '%xyz%'" の述部に関する選択度の評価を上げるために、照会オプティマイザーで使用されます。

ALLOW WRITE ACCESS

統計が計算される間に、他のユーザーが表から読み込んだりそこに書き込んだりできることを指定します。統計ビューの場合は、ビュー定義の中で参照されている基本表です。

多くの挿入、更新、または削除が同時に実行される表の場合、**ALLOW WRITE ACCESS** オプションは勧められていません。 RUNSTATS コマンドは、まず表統計を実行した後、索引統計を実行します。表および索引の統計情報を収集している間に表の状態が変化した場合、不整合が発生することがあります。照会の最適化のために最新の統計を収集することは重要ですが、整合性のある統計を収集することも重要です。したがって統計情報の収集は、挿入、更新、または削除の操作が最小になる時間に実行してください。

ALLOW READ ACCESS

統計が計算される間に、他のユーザーが表に対して読み取り専用のアクセスを行えることを指定します。統計ビューの場合は、ビュー定義の中で参照されている基本表です。

TABLESAMPLE BERNOULLI

このオプションを使用した RUNSTATS は、表または統計ビューから取られた行のサンプルに関する統計を収集します。 *BERNOULLI (ベルヌーイ) サンプリング* では各行が個別に処理され、その際に *P*/100 (*P* は数値リテラル値) の確率で行が含められ、1-*P*/100 の確率で行が除外されます。例えば、数値リテラルが値 10 (つまり、10 % のサンプル) と評価された場合は、0.1 の確率で各行が含められ、0.9 の確率で行が除外されます。オプションの **REPEATABLE** 節を指定しない限り、RUNSTATS を実行するたびに、通常は異なった表のサンプルが作成されます。すべてのデータ・ページが表スキャンによって検索されますが、数値リテラル・パラメーターによって指定したパーセンテージの行だけが、統計収集に使用されます。

TABLESAMPLE SYSTEM

このオプションを使用した RUNSTATS は、表から取られたデータ・ページのサンプルに関する統計を収集します。 *SYSTEM (システム) サンプリング* では各ページが個別に処理され、その際に *P*/100 (*P* は数値リテラル値) の確率でページが含められ、1-*P*/100 の確率でページが除外されます。オプションの **REPEATABLE** 節を指定しない限り、RUNSTATS を実行するたびに、通常は異なった表のサンプルが作成されます。サンプルのサイズは、括弧内の数値リテラル・パラメーターによって制御し、表の約 P % を戻すように指定します。数値リテラル・パラメーターによって指定したパーセンテージのデータ・ページだけが、検索されて統計収集に使用されます。

統計ビューでは、SYSTEM サンプリングは、定義が単一の基本表に対する選択であるビューに制限されます。ビューに複数の表が含まれている場合、以下の条件をすべて満たしていれば SYSTEM サンプリングも可能です。

- 表が、表間で定義されている参照整合性制約に組み込まれている、すべての主キー列および外部キー列上の等価述部を使用して結合されている。
- リレーションシップ内のどの親表にも検索条件フィルター行がない。
- 親表ではない、単一の子表が、すべての表の中から識別できる。

統計ビューがこれらの条件を満たしていない場合、BERNOULLI サンプリングが代わりに使用され、警告が戻されます (SQL2317W)。

REPEATABLE (*integer-literal***)**

REPEATABLE 節を **TABLESAMPLE** 節に追加すれば、RUNSTATS の反復実行時に必ず同じサンプルが戻されるようになります。 *integer-literal* パラメーターは、サンプリングで使用するシードを表す負以外の整数です。負のシードを渡した場合、エラー (SQL1197N) が発生します。
TABLESAMPLE REPEATABLE の最後の実行以降に行われた表または統計ビューに対する活動によって表または統計ビューのデータが変更された場合には、反復可能な **RUNSTATS** 呼び出しでサンプル・セットが変化する可能性があります。また、必ず整合した結果を得るためには、**BERNOULLI** (ベルヌーイ) または **SYSTEM** (システム) キーワードによって指定するサンプルの入手方法が同じでなければなりません。

numeric-literal

numeric-literal パラメーターでは、入手するサンプルのサイズを指定します (*P* %)。この値は 100 以下の正数でなければならず、1 と 0 の間の数を指定することもできます。例えば、値 0.01 は、1 % の 100 分の 1 を表します。この場合は、平均して 10,000 行のうちの 1 行がサンプルとして取られます。 0 または 100 の値を指定した場合、DB2 データベース・システムでは、**TABLESAMPLE BERNOULLI** と **TABLESAMPLE SYSTEM** のどちらが指定されているかにかかわらず、サンプリングが指定されていない場合と同じように処理されます。 100 より大きい値または 0 より小さい値は、DB2 データベース・システムではエラー (SQL1197N) として処理されます。

SET PROFILE NONE

この RUNSTATS 呼び出しには統計プロファイルを設定しないことを指定します。

SET PROFILE

RUNSTATS は、特定の統計プロファイルを生成してシステム・カタログ表に保管し、RUNSTATS コマンド・オプションを実行して統計を収集します。

SET PROFILE ONLY

RUNSTATS は、特定の統計プロファイルを生成してシステム・カタログ表に保管しますが、RUNSTATS のコマンド・オプションを実行しません。

UPDATE PROFILE

RUNSTATS は、システム・カタログ表内の既存の統計プロファイルを変更し、その更新済みの統計プロファイルの RUNSTATS コマンド・オプションを実行して統計を収集します。UPDATE PROFILE オプションを使用して、統計プロファイルにある節を削除することはできません。

UPDATE PROFILE ONLY

RUNSTATS は、システム・カタログ表内の既存の統計プロファイルを変更しますが、その更新済みの統計プロファイルのRUNSTATS コマンド・オプションを実行しません。 UPDATE PROFILE ONLY オプションを使用して、統計プロファイルにある節を削除することはできません。

UTIL_IMPACT_PRIORITY *priority*

priority に指定されているレベルで、RUNSTATS をスロットルすることを指定します。 *priority* は 1 から 100 の範囲の数であり、 100 が最高の優先順位、1 が最低の優先順位を表します。優先順位によって、ユーティリティーのスロットルの量が決まります。優先順位が同じユーティリティーはすべて同じ量のスロットルになり、優先順位の低いユーティリティーは、優先順位の高いユーティリティーよりも量が絞られます。 *priority* を指定しない場合、RUNSTATS はデフォルトの優先順位 50 を使用します。

UTIL_IMPACT_PRIORITY キーワードを省略すると、スロットルのサポートなしで RUNSTATS ユーティリティーが呼び出されます。

UTIL_IMPACT_PRIORITY キーワードを指定した場合でも、**util_impact_lim** 構成パラメーターが 100 に設定されていれば、スロットルなしでユーティリティーが実行されます。ビューの場合、このオプションは使用できません。

パーティション・データベースでは、RUNSTATS コマンドが表に対して使用された場合、1 つのデータベース・パーティションでしか統計を収集できません。RUNSTATS コマンドが実行されたデータベース・パーティションに表のパーティションがある場合、コマンドは、そのデータベース・パーティションで実行されます。それ以外の場合は、表がパーティションに分けられているデータベース・パーティション・グループの最初のデータベース・パーティションで実行されます。

例

索引で使用されるすべての列と、すべての索引の統計を収集します。

```
CALL SYSPROC.ADMIN_CMD ('RUNSTATS ON TABLE db2user.employee
  ON KEY COLUMNS and INDEXES ALL')
```

使用上の注意

1. パーティション表にデタッチされたパーティションが存在する場合、デタッチされたデータ・パーティションでクリーンアップの必要なものにまだ属している索引キーは、統計においてキーの一部としてカウントされません。それらのキーは、不可視であり表の一部ではなくなっているため、カウントされません。そのような索引キーは、最終的に非同期の索引クリーンアップによって索引から除去されます。その結果、非同期の索引クリーンアップを実行する前に収集された統計は誤ったものとなります。非同期索引クリーンアップの完了前に RUNSTATS コマンドが発行された場合、不正確な統計情報を基に、索引再編成または索引クリーンアップに対する誤ったアラームが生成されることがしばしばあります。非同期の索引クリーンアップの実行を開始すると、クリーンアップを必要とするデタッチされたデータ・パーティションにまだ属しているすべての索引キーが除去されるので、索引の再編成の必要がなくなることもあります。

   パーティション表の場合、非同期索引クリーンアップ完了後に RUNSTATS コマンドを発行することをお勧めします。それは、デタッチされているデータ・パーティションの存在に関して正確な索引統計情報を生成するためです。表の中にデタッチされているデータ・パーティションがあるかどうかを調べるには、 SYSCAT.DATAPARTITIONS カタログ・ビューの状況フィールドを確認して、値 L (論理的にデタッチ済み)、I (索引クリーンアップ)、または D (デタッチ済みで従属 MQT 付き) を探してください。

   RUNSTATS コマンドは、パーティション索引のすべての索引パーティションの統計を収集します。パーティション索引の SYSTAT.INDEXES ビュー内の統計は、索引パーティションを表します。ただし FIRSTKEYCARD、FIRST2KEYCARD、 FIRST3KEYCARD、 FIRST4KEYCARD、および FULLKEYCARD 統計は除きます。これらの統計はカーディナリティーの見積もりで使用されるので、それらは索引全体に対するものであり、1 つの索引パーティションに対するものではありません。分散統計 (頻度および変位値) は、パーティション索引では収集されませんが、RUNSTATS が表上で実行される場合は収集されます。パーティション索引の先行列の統計は、非パーティション索引の先行列の統計ほど正確でない場合があります。
2. コマンドの実行状況は、CALL ステートメントからの結果である SQLCA で戻されます。
3. RUNSTATS コマンドは、次のような場合に実行することが勧められています。

- 表が大幅に変更されている場合 (例えば、多くの変更が行われている場合や、大量のデータが挿入または削除されている場合、あるいは、LOAD 時に統計オプションを指定しないで LOADが行われた場合など)。
- 表が再編成されている場合 (REORG、REDISTRIBUTE DATABASE PARTITION GROUP を使用)。
- 行圧縮が実行された表の場合。
- 新しい索引が作成されている場合
- パフォーマンスが重要な意味を持つアプリケーションのバインドの前。
- プリフェッチ・サイズが変更された場合。
- 基礎表に実質的な変更が加えられたことにより、ビューによって戻される行が変更された統計ビューの場合。
- STATISTICS オプションを指定した **LOAD** が実行された後には、RUNSTATS ユーティリティーを使用して、XML 列に関する統計情報を収集してください。 LOAD 実行時には、LOAD に **STATISTICS** オプションを指定して実行された場合であっても、XML 列に関する統計情報が収集されることは決してありません。 RUNSTATS を使用することにより、XML 列についてのみの統計情報を収集する場合、XML 以外の列に関して LOAD またはそれ以前に実行された RUNSTATS ユーティリティーによって収集された既存の統計情報は、そのまま保持されます。以前に一部の XML 列に関する統計情報が収集されていた場合、ある XML 列に関する統計情報が現在のコマンドでは収集されないのであれば、その XML 列に関して以前に収集された統計情報はドロップされます。あるいは、その XML 列に関する統計が現在のコマンドで収集されるのであれば、置き換えられます。

4. オプションの選択は、特定の表やアプリケーションに合わせて行う必要があります。一般的なヒントとして、以下の点を考慮してください。
   - 重要な照会に使用される非常に重要な表、比較的小規模な表、またはあまり変化がなく、システムそのものでの活動があまりない表には、可能な限り詳細に統計を収集する努力を費やす価値があります。
   - 統計を収集する時間が限られている場合、表が比較的大規模な場合、または表が頻繁に更新される場合には、述部で使用される列セットに限って RUNSTATS を実行するのも良い方法かもしれません。このような方法を使用する場合には、より頻繁に RUNSTATS コマンドを実行できるでしょう。
   - 統計を収集する時間が極めて限られており、表ごとに表の RUNSTATS コマンドを調整するのが時間の面で大きな問題となっている場合は、"KEY" 列だけの統計を収集することも考慮してください。索引に含まれている列セットは、表にとって重要で、述部に使用される確立が最も高いと考えられます。
   - 統計を収集する時間が非常に限られている状況で表統計を収集する場合は、**TABLESAMPLE** オプションを使用して、表データのサブセットに関する統計を収集することを検討してください。
   - 表に多くの索引があり、それらの索引に含まれる **DETAILED** (拡張) 情報がアクセス・プランを向上させる可能性がある場合は、統計の収集にかかる時間を減らすために、**SAMPLED** オプションを考慮してください。
   - 特定の列にスキューがあり、述部のタイプが `"column = constant"` である場合、その列にはより大きな **NUM_FREQVALUES** 値を指定するほうが良い可能性があります。

- 等式の述部で使用される列や、値の分散がスキューされる可能性のある列では、必ず分散統計を収集してください。
- 範囲の述部を持つ列 (例えば、"column >= constant"、"column BETWEEN constant1 AND constant2" など) や、タイプ "column LIKE '%xyz'" の列では、より大きな **NUM_QUANTILES** 値を指定したほうが有益な場合があります。
- ストレージ・スペースが関係している場合で、統計の収集にあまり時間をかけられない場合は、述部で使用されない列の **NUM_FREQVALUES** 値や **NUM_QUANTILES** 値をあまり高くしないでください。
- 索引統計を要求したときに、索引を含む表についての統計がそれまで実行されていなかった場合、表と索引の両方に関する統計が計算されます。
- 表に含まれる XML 列に関する統計情報が必要ない場合は、**EXCLUDING XML COLUMNS** オプションを使用することによって、XML 列をすべて除外することができます。このオプションは、統計情報収集の対象として XML 列を指定する他のどの節よりも優先されます。

5. コマンドを実行した後は、以下の点に注意してください。
   - ロックを解除するには、COMMIT を発行する必要があります。
   - 新しいアクセス・プランを生成できるようにするには、ターゲット表を参照するパッケージを再バインドする必要があります。
   - 表で部分的にコマンドを実行すると、コマンドが最後に実行されてからの表での活動の結果として、不整合が生じる可能性があります。このような場合には、警告メッセージが戻されます。表でだけ RUNSTATS が実行されると、表レベルの統計と索引レベルの統計に不整合が生じます。例えば、ある表に関して索引レベルの統計を収集した後で、その表からかなりの数の行を削除してしまったとします。このような場合に、その表でだけ RUNSTATS を発行すると、表のカーディナリティーが FIRSTKEYCARD よりも小さくなってしまう可能性があります。これは不整合です。これと同様に、作成した新しい索引で統計を収集した場合にも、表レベルの統計に不整合が生じることがあります。
6. RUNSTATS コマンドは、表統計が要求したときに、以前に収集された分散統計をドロップします。例えば、RUNSTATS ON TABLE または RUNSTATS ON TABLE ... AND INDEXES ALL は、以前に収集された分散統計がドロップされる原因になります。コマンドが索引でのみ実行される場合、以前に収集された分散統計は保持されます。例えば、RUNSTATS ON TABLE ... FOR INDEXES ALL は、以前に収集された分散統計が保持される原因になります。RUNSTATS コマンドが XML 列に対してのみ実行される場合、それ以前に収集された基本列統計および分散統計はそのまま保持されます。以前に一部の XML 列に関する統計情報が収集されていた場合、ある XML 列に関する統計情報が現在のコマンドでは収集されないのであれば、その XML 列に関して以前に収集された統計情報はドロップされます。あるいは、その XML 列に関する統計が現在のコマンドで収集されるのであれば、置き換えられます。
7. DB2 バージョン 9.7 フィックスパック 1 以降のリリースでは、XML 列に定義された XML データに対する索引で、分散統計が収集されます。表に対して、RUNSTATS コマンドを **WITH DISTRIBUTION** 節付きで実行する場合、タイプ XML の列に関する分散統計の収集は以下のようになります。

- 分散統計は、XML 列で指定されている XML データに対する各索引に関して収集されます。
- RUNSTATS コマンドでは、XML 列で定義された XML データに対する索引に関して分散統計を収集するため、分散統計と表統計の両方を収集する必要があります。XML 分散統計は表統計と共に保管されるので、分散統計を収集するためには表統計を収集しなければなりません。

  XML 分散統計を収集するために、索引節は必要ありません。索引節だけを指定しても、XML 分散統計は収集されません。

  XML 分散統計はデフォルトで、XML データに対する各索引に関して最大 250 の分位数を使用します。 XML 列に対する分散統計を収集する場合、**ON COLUMNS** 節または **DEFAULT** 節の **NUM_QUANTILES** パラメーターに値を指定することによって分位の最大数を変更することができます。
- 分散統計は、タイプ VARCHAR、DOUBLE、TIMESTAMP、および DATE の XML データの索引に関して収集されます。タイプ VARCHAR HASHED の索引に関して、分散統計は収集されません。
- 分散統計は、パーティション表で定義された XML データに対するパーティション索引に関しては収集されません。

8. 範囲クラスター表の場合、範囲クラスター表の範囲配列プロパティーを表す特殊なシステム生成索引がカタログ表内に存在します。この種の表の統計を収集するときに、統計収集の一部として表を組み込む場合は、システム生成索引用の統計も収集されます。この統計は、基本データ表と同じページ数を持つ 2 レベルの索引として索引を表現し、索引の順序に沿って完全に基本データをクラスター化することによって、範囲検索の高速アクセスを反映することになっています。
9. コマンド構文の `On Dist Cols` 節では、列 `GROUPS` に対する Frequency Option および Quantile Option のパラメーターの使用は、現在サポートされていません。これらのオプションは、単一の列でのみサポートされています。
10. DMS モードでの作業中に、計算できない 3 つのプリフェッチ統計があります。索引カタログ内の索引統計で、以下の統計の値は `-1` になります。
    - `AVERAGE_SEQUENCE_FETCH_PAGES`
    - `AVERAGE_SEQUENCE_FETCH_GAP`
    - `AVERAGE_RANDOM_FETCH_PAGES`
11. **TABLESAMPLE** による RUNSTATS サンプリングは、索引ページではなく表データ・ページでのみ行われます。索引統計とサンプリングが要求された場合は、統計収集のためにすべての索引ページがスキャンされます。そのようになるのは、**TABLESAMPLE** が適用される表統計の収集においてのみです。しかし、**SAMPLED DETAILED** オプションを使用すれば、詳細な索引統計をより効率的に収集できます。これは、**TABLESAMPLE** の場合とは別のサンプリング方法であり、索引統計の詳細なセットにのみ適用されます。
12. プロファイルの設定またはプロファイルの更新のオプションを使用することによって、RUNSTATS コマンドで指定する表または統計ビューの統計プロファイルを設定または更新できます。統計プロファイルは、`STATISTICS_PROFILE` システム・カタログ表の `SYSCAT.TABLES` 列に、可視ストリングのフォーマットで保管されます。これが RUNSTATS コマンドに相当します。

13. XML タイプの列に関する統計情報収集は、**DB2_XML_RUNSTATS_PATHID_K** および **DB2_XML_RUNSTATS_PATHVALUE_K** の 2 つの DB2 データベース・システム・レジストリー値によって制御されます。これらの 2 つのパラメーターは、収集する度数の数を指定するという点において **NUM_FREQVALUES** パラメーターに似ています。設定されていない場合、どちらのパラメーターについてもデフォルトとして 200 が使用されます。
14. RUNSTATS は、開始時に、SYSTABLES に対して IX 表ロックを取得し、統計収集の対象となる表の行に対して U ロックを取得します。 U ロックのかかった行を含め、SYSTABLES からの読み取り操作は可能です。また、U ロックのかかった行でない限り、書き込み操作も可能です。しかし、RUNSTATS が IX ロックを取得しているため、別の読み取りプログラムまたは書き込みプログラムが SYSTABLES に対する S ロックを取得することはできません。
15. 統計は、構造化タイプの列については収集されません。それらが指定されている場合、そのデータ・タイプの列は無視されます。
16. LOB または LONG データ・タイプの列では、AVGCOLLEN および NUMNULLS のみが収集されます。
17. AVGCOLLEN は、列がデータベース・メモリーまたは一時表に保管される場合の、バイト単位の平均スペースを表します。この値は、LOB または LONG データ・タイプのデータ記述子の長さを表します。ただし LOB データがデータ・ページ上でインライン化されている場合を除きます。

    注: ディスク上で列を保管するための平均スペース所要量は、この統計により表される値とは異なる場合があります。

SET TAPE POSITION コマンド (ADMIN_CMD プロシージャーを使用)

ストリーミング磁気テープ装置へのバックアップおよびリストア操作のためにテープの位置を設定します。このコマンドは Windows オペレーティング・システムでのみサポートされています。

許可

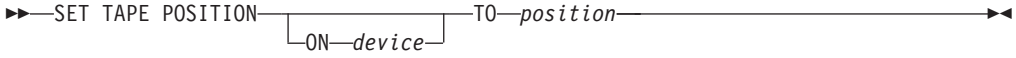
以下のいずれかです。

- SYSADM
- SYSCTRL
- SYSMAINT

必要な接続

データベース

コマンド構文



コマンド・パラメーター

ON *device*

有効なテープ装置名を指定します。デフォルト値は ¥¥.¥TAPE0 です。装置の指定は、サーバーに対する相対指定でなければなりません。

TO *position*

テープ位置のマークを指定します。 DB2 (Windows 版) は、バックアップ・イメージの度にテープ・マークを書き込みます。値 1 は 1 番目の位置、 2 は 2 番目の位置、以下同じ手順で指定します。テープがテープ・マーク 1 に位置している場合、例えば、アーカイブ 2 がリストアされる位置に置かれます。

例

DB2 データベースは各バックアップ・イメージの後にテープ・マークを書き込むため、位置として 1 を指定すると、テープ上の 2 番目のアーカイブの開始位置にテープが移動します。

```
CALL SYSPROC.ADMIN_CMD( 'set tape position to 1' )
```

使用上の注意

コマンドの実行状況は、CALL ステートメントからの結果である SQLCA で戻されます。

UNQUIESCE DATABASE コマンド (ADMIN_CMD プロシージャーを使用)

保守などの理由で静止状態になっていたデータベースに対するユーザー・アクセスを回復します。UNQUIESCE は、シャットダウンとデータベース再始動の必要なしにユーザー・アクセスを回復します。

特に指定がない限り、SYSADM、SYSMAINT、または SYSCTRL 以外のユーザーは、静止中のデータベースにアクセスできません。そのため、静止データベースの一般アクセスを回復するには、UNQUIESCE が必要です。

有効範囲

UNQUIESCE DB は、静止データベース内のすべてのオブジェクトに対するユーザー・アクセスを回復します。

インスタンスを停止した後、そのインスタンスとそのすべてのデータベースの静止を解除するには、 db2stop コマンドを発行します。 DB2 を停止し、再開すると、すべてのインスタンスとデータベースの静止が解除されます。

許可

以下のいずれかです。

データベース・レベルの静止解除の場合:

- SYSADM
- DBADM

コマンド構文

必要な接続

データベース

コマンド・パラメーター

DB データベースの静止解除。データベース内のすべてのオブジェクトに対するユーザー・アクセスが回復されます。

例

データベースの静止解除

```
CALL SYSPROC.ADMIN_CMD( 'unquiesce db' )
```

このコマンドは、以前に静止されていたデータベースの静止を解除します。

使用上の注意

コマンドの実行状況は、CALL ステートメントからの結果である SQLCA で戻されます。

UPDATE ALERT CONFIGURATION コマンド (ADMIN_CMD プロシージャーを使用)

ヘルス・インディケーターのアラート構成設定を更新します。

重要: バージョン 9.7 でヘルス・モニターが非推奨になったため、このコマンドまたは API は推奨されておらず、将来のリリースで除去される可能性があります。詳しくは、「*DB2 バージョン 9.7 の新機能*」のトピック『ヘルス・モニターが推奨されなくなった』を参照してください。

許可

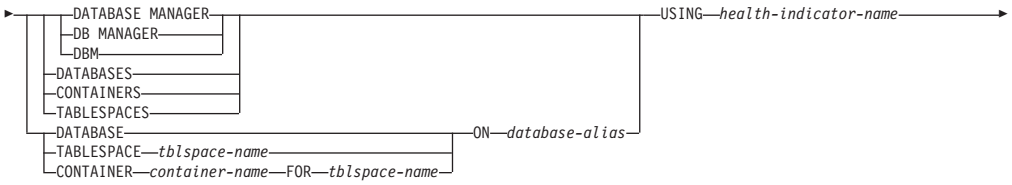
以下のいずれか。

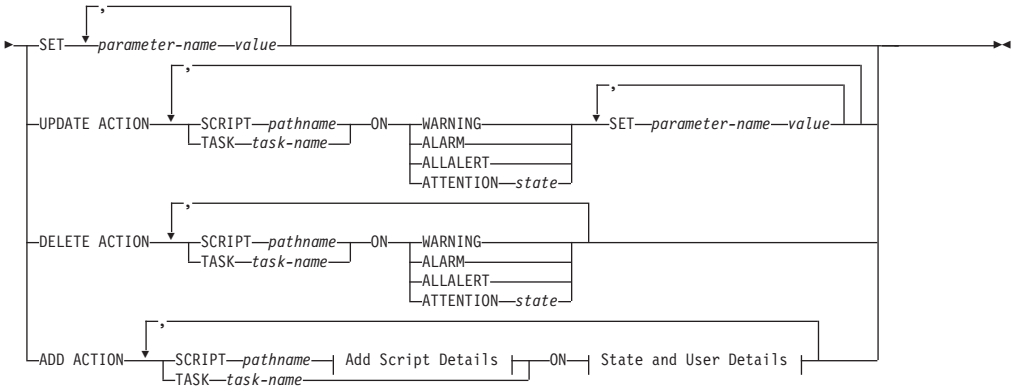
- SYSADM
- SYSMAINT
- SYSCTRL

必要な接続

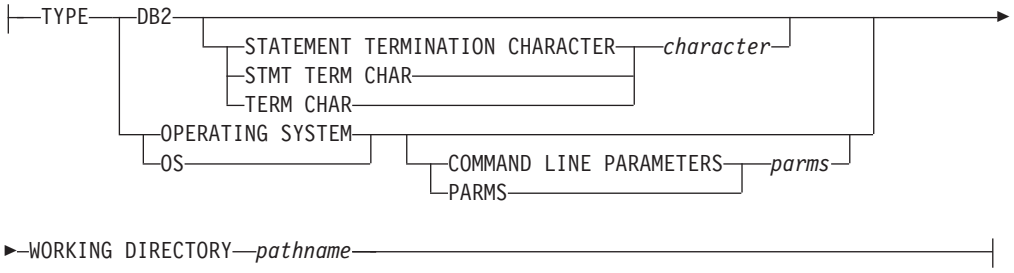
データベース

コマンド構文

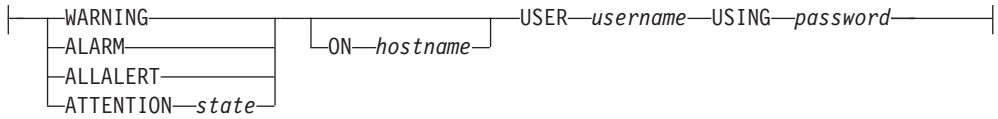




Add Script Details:



State and User Details:



コマンド・パラメーター

DATABASE MANAGER
データベース・マネージャーのアラート設定を更新します。

DATABASES
データベース・マネージャーによって管理されるすべてのデータベースのアラート設定を更新します。これは、カスタム設定を持たないすべてのデータベースに適用される設定です。カスタム設定は、**DATABASE ON** *database-alias* 節を使って定義されます。

CONTAINERS
データベース・マネージャーによって管理されるすべての表スペース・コンテナーのアラート設定を更新します。これは、カスタム設定を持たないすべ

ての表スペース・コンテナーに適用される設定です。カスタム設定は、**CONTAINER** *container-name* **ON** *database-alias* 節を使って定義されます。

TABLESPACES
データベース・マネージャーによって管理されるすべての表スペースのアラート設定を更新します。これは、カスタム設定を持たないすべての表スペースに適用される設定です。カスタム設定は、**TABLESPACE** *tblspace-name* **ON** *database-alias* 節を使って定義されます。

DATABASE ON *database-alias*
ON *database-alias* 節を使って指定したデータベースのアラート設定を更新します。このデータベースがカスタム設定を持つ場合、インスタンスの全データベースの設定をオーバーライドします。これは、**DATABASES** パラメーターを使って指定されます。

CONTAINER *container-name* **FOR** *tblspace-name* **ON** *database-alias*
ON *database-alias* 節を使って指定したデータベース上で、**FOR** *tblspace-name* 節を使って指定した表スペースの、*container-name* という名前の表スペース・コンテナーのアラート設定を更新します。この表スペース・コンテナーがカスタム設定を持つ場合、データベースの全表スペース・コンテナーの設定をオーバーライドします。これは、**CONTAINERS** パラメーターを使って指定されます。

TABLESPACE *tblspace-name* **ON** *database-alias*
ON *database-alias* 節を使って指定したデータベース上で、*name* という名前の表スペースのアラート設定を更新します。この表スペースがカスタム設定を持つ場合、データベースの全表スペースの設定をオーバーライドします。これは、**TABLESPACES** パラメーターを使って指定されます。

USING *health-indicator-name*
アラート構成が更新されるヘルス・インディケーターのセットを指定します。ヘルス・インディケーター名は 2 文字のオブジェクト ID で構成され、その後にインディケーターの測定対象を説明する名前が続きます。以下に例を示します。

```
db.sort_privmem_util
```

SET *parameter-name value*
ヘルス・インディケーターのアラート構成エレメント *parameter-name* を、指定した値に更新します。 *parameter-name* は以下のどれかになります。

- ALARM: *value* はヘルス・インディケーター・ユニット。
- WARNING: *value* はヘルス・インディケーター・ユニット。
- SENSITIVITY: *value* は秒単位。
- ACTIONSENABLED: *value* は YES または NO。
- THRESHOLDSCHECKED: *value* は YES または NO。

特定の DB2 バージョンについて可能性のあるヘルス・インディケーター・ユニットのリストは、以下の照会を実行することにより収集できます。

```
SELECT SUBSTR(UNIT,1,80) AS UNIT
   FROM TABLE(HEALTH_GET_IND_DEFINITION('')) AS T GROUP BY UNIT
```

UPDATE ACTION SCRIPT *pathname* **ON [WARNING | ALARM | ALLALERT | ATTENTION** *state***]**

絶対パス名 *pathname* を持つ定義済みスクリプトのスクリプト属性が以下の節に従って更新されるように指定します。

SET *parameter-name value*

スクリプト属性 *parameter-name* を、指定した値に更新します。*parameter-name* は以下のどれかになります。

- SCRIPTTYPE

  有効なタイプは OS または DB2。
- WORKINGDIR
- TERMCHAR
- CMDLINEPARMS

  オペレーティング・システム・スクリプトに対して指定するコマンド行パラメーターが、デフォルトで指定されるパラメーターに先行します。オペレーティング・システム・スクリプトに送られるパラメーターは、以下のとおりです。
  - ユーザーの指定するパラメーターのリスト
  - ヘルス・インディケーターの短縮名
  - 完全修飾オブジェクト名
  - ヘルス・インディケーターの値
  - アラート状態
- USERID
- PASSWORD
- SYSTEM

UPDATE ACTION TASK *task-name* **ON [WARNING | ALARM | ALLALERT | ATTENTION** *state***]**

名前 *name* を持つタスクのタスク属性が以下の節に従って更新されるように指定します。

SET *parameter-name value*

タスク属性 *parameter-name* を、指定した値に更新します。*parameter-name* は以下のどれかになります。

- USERID
- PASSWORD
- SYSTEM

DELETE ACTION SCRIPT *pathname* **ON [WARNING | ALARM | ALLALERT | ATTENTION** *state***]**

アラート・アクション・スクリプトから、絶対パス名 *pathname* を持つアクション・スクリプトを除去します。

DELETE ACTION TASK *task-name* **ON [WARNING | ALARM | ALLALERT | ATTENTION** *state***]**

アラート・アクション・タスクのリストから *name* という名前のアクション・タスクを除去します。

ADD ACTION SCRIPT *pathname* **ON [WARNING | ALARM | ALLALERT | ATTENTION** *state*]

絶対パス名 *pathname* を持つ新規アクション・スクリプトが追加されるように指定します。その属性は、以下のように指定されます。

TYPE アクション・スクリプトは、 DB2 コマンド・スクリプトか、オペレーティング・システム・スクリプトのいずれかでなければなりません。

- DB2
- OPERATING SYSTEM

DB2 コマンド・スクリプトの場合、以下の節を使用することにより、オプションで文字 *character* を指定することができます。この文字は、ステートメントを終了するのにスクリプト内で使用されます。

```
STATEMENT TERMINATION CHARACTER ;
```

オペレーティング・システム・スクリプトの場合、以下の節を使用することにより、オプションでコマンド行パラメーター *parms* を指定することができます。これは、呼び出しの際にスクリプトに渡されます。 **COMMAND LINE PARAMETERS** *parms*

WORKING DIRECTORY *pathname*
スクリプトが実行されるディレクトリーの絶対パス名 *pathname* を指定します。

USER *username* **USING** *password*
スクリプトが実行される際のユーザー・アカウント *username*、およびそれに関連したパスワード *password* を指定します。 **ADD ACTION** オプションが使用されている場合、*username* と *password* が、ネットワークの中 (*username* と *password* は暗号化されないで送られる)、db2diag ログ・ファイル、トレース・ファイル、ダンプ・ファイル、スナップショット・モニター (動的 SQL スナップショット)、システム・モニター・スナップショット、いくつかのイベント・モニター (ステートメントやデッドロックなど)、Query Patroller、Explain 表、db2diag 出力 (パッケージ・キャッシュやロック・タイムアウト・メカニズムなど)、および DB2 監査レコードの中で露出する可能性があります。

ADD ACTION TASK *name* **ON [WARNING | ALARM | ALLALERT | ATTENTION** *state*]

指定した条件を **ON** にして *name* という新規タスクを追加し、実行することを指定します。

ON [WARNING | ALARM | ALLALERT | ATTENTION *state*]

アクションまたはタスクが実行される条件を指定します。しきい値ベースのヘルス・インディケーター (HI) の場合、これは **WARNING** または **ALARM** になります。状態ベースの HI の場合、これは、各状態ベースの HI ごとに記されている数値状態 (例えば、ts.ts_op_status ヘルス・インディケーターの場合は、表スペース状態の **tablespace_state** モニター・エレメントを参照してください) か、この状態を表すテキストの ID となります。

ALLALERTS は、しきい値ベースの HI と状態ベースの HI に関する状態の変更を処理します (例えば、警告から正常への状態変更)。

ATTENTION *state*

データベース・ヘルス・インディケーターの一部の状態の有効な数値は、ADD ACTION SCRIPT CLP コマンド・オプションの例として以下に示されています。

- 0 - アクティブ、正常 (ACTIVE)
- 1 - 静止ペンディング (QUIESCE_PEND)
- 2 - 静止済み (QUIESCED)
- 3 - ロールフォワード (ROLLFWD)

その他の状態ベースのヘルス・インディケーターは、ヘッダー・ファイル `sqlmon.h` と `sqlutil.h` に定義されています。

ADMIN_CMD ストアード・プロシージャーにより呼び出される UPDATE ALERT CFG コマンドは、*state* において数値またはテキスト ID をサポートします。表スペース操作状況のヘルス・インディケーター (ts.ts_op_status) の例として、ヘルス・インディケーターの一部の追加の状態の有効な数値およびテキスト ID は、次のようになります。

- 0x1 - QUIESCED_SHARE
- 0x2 - QUIESCED_UPDATE
- 0x4 - QUIESCED_EXCLUSIVE

UPDATE ALERT CFG コマンドと上記のヘルス・インディケーター値を使用して、以下のコマンド行を入力するとします。

```
ADD ACTION SCRIPT ... ON ATTENTION 2
```

これは、以下と同じ意味になります。

```
ADD ACTION SCRIPT ... ON ATTENTION QUIESCED_UPDATE
```

さらに、表スペース操作状況のヘルス・インディケーター (ts.ts_op_status) では、複数の状態を OR 操作することにより、単一の数値を使用して複数の状態を指定できます。例えば、状態 7 (= 0x1 + 0x2 + 0x4) を指定すると、表スペースが、静止: SHARE、静止: UPDATE、または静止: EXCLUSIVE のいずれかの状態になったときにアクションが実行されます。その代わりに、3 つの別個の UPDATE ALERT CFG コマンド実行で、QUIESCED_SHARE、QUIESCED_UPDATE、および QUIESCED_EXCLUSIVE を指定することもできます。

例

ホスト名 'plato' のシステムでアラームがあった場合に、スクリプト `/home/test/scripts/logfsutilact` を実行するという、db.log_fs_util インディケーターのアクションを追加します。

```
CALL SYSPROC.ADMIN_CMD( 'update alert cfg for databases using
   db.log_fs_util add action script /home/test/scripts/logfsutilact
   type os command line parameters "param1 param2" working
   directory /tmp on alarm on plato user dricard using mypasswdv' )
```

アラート構成が設定された後、それをチェックするには、以下のようにして HEALTH_GET_IND_DEFINITION および HEALTH_GET_ALERT_ACTION_CFG の表関数を使用することができます。

```
SELECT OBJECTTYPE, ID, CONDITION, ACTIONTYPE,
   SUBSTR(ACTIONNAME,1,50) AS ACTION_NAME
   FROM TABLE(SYSPROC.HEALTH_GET_ALERT_ACTION_CFG('DB','G','',''))
   AS ALERT_ACTION_CFG
```

以下はこの照会の出力例です。

```
OBJECTTYPE ID     CONDITION  ACTIONTYPE ACTION_NAME
---------- ------ ---------- ---------- --------------------------------
DB           1006 ALARM      S          /home/dricard/scripts/logfsutilact

   1 record(s) selected.
```

使用上の注意

ADD ACTION オプションでは、入力した *username* および *password* は、SQL ステートメント・テキストがキャプチャーされる以下のさまざまな場所で公開される場合があります。

- ネットワーク (username と password は暗号化されずにワイヤーを経由して渡されます)
- db2diag ログ・ファイル
- トレース・ファイル
- ダンプ・ファイル
- スナップショット・モニター (動的 SQL スナップショット)
- システム・モニター・スナップショット
- いくつかのイベント・モニター (ステートメント、デッドロック)
- Query Patroller
- Explain 表
- db2pd 出力 (パッケージ・キャッシュおよびロック・タイムアウト機構など)
- DB2 監査レコード

コマンドの実行状況は、CALL ステートメントからの結果である SQLCA で戻されます。

database-alias は、サーバー上のカタログの中で定義されていなければならず、サーバーに対してローカルなものでなければなりません。

pathname は、完全修飾サーバー・パス名を使用して指定する必要があります。

UPDATE CONTACT コマンド (ADMIN_CMD プロシージャーを使用)

ローカル・システムで定義される連絡先の属性を更新します。連絡先とは、スケジューラーおよびヘルス・モニターがメッセージを送信する先のユーザーです。

連絡先を作成するには、ADD CONTACT コマンドを使用します。 Database Administration Server (DAS) の **contact_host** 構成パラメーターの設定により、リストがローカルかグローバルかが決まります。

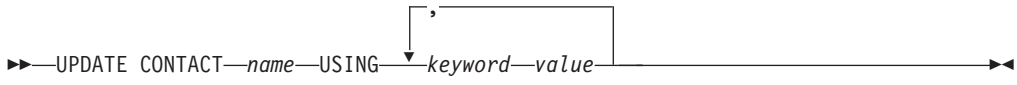
許可

なし

必要な接続

データベース。 DAS が実行中でなければなりません。

コマンド構文



コマンド・パラメーター

UPDATE CONTACT *name*
更新される連絡先の名前。

USING *keyword value*
更新される連絡先パラメーター (*keyword*) および設定される値 (*value*) を指定します。有効なキーワードのセットは次のとおりです。

ADDRESS
SMTP サーバーが通知を送信するのに使用する E メール・アドレス。

TYPE アドレスが E メール・アドレスか、ページャーかを指定します。

MAXPAGELEN
ページャーが受信できる最大文字数。

DESCRIPTION
連絡先のテキスト記述。長さは、最大 128 文字です。

例

ユーザー 'test' のアドレスを 'newaddress@test.com' に更新します。

```
CALL SYSPROC.ADMIN_CMD( 'update contact test using address newaddress@test.com' )
```

使用上の注意

DAS が作成されていて実行中でなければなりません。

コマンドの実行状況は、CALL ステートメントからの結果である SQLCA で戻されます。

UPDATE CONTACTGROUP コマンド (ADMIN_CMD プロシージャーを使用)

ローカル・システムで定義される連絡先グループの属性を更新します。連絡先グループは、スケジューラーおよびヘルス・モニターから通知を受け取るユーザーのリストです。

Database Administration Server (DAS) の **contact_host** 構成パラメーターの設定により、リストがローカルかグローバルかが決まります。

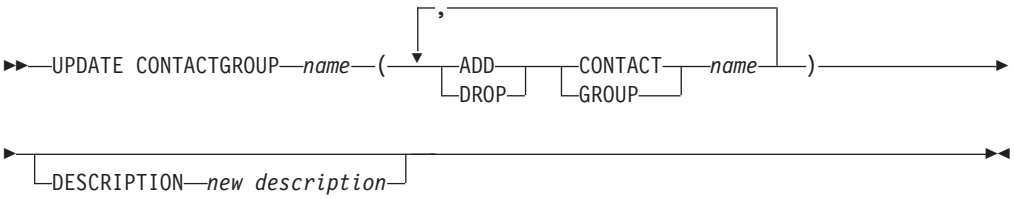
許可

なし

必要な接続

データベース。 DAS が実行中でなければなりません。

コマンド構文



コマンド・パラメーター

CONTACTGROUP *name*
更新される連絡先グループの名前。

ADD CONTACT *name*
グループに追加される新しい連絡先の名前を指定します。グループに追加された後、ADD CONTACT コマンドを使用して連絡先を定義できます。

DROP CONTACT *name*
グループからドロップされる、グループ中の連絡先の名前を指定します。

ADD GROUP *name*
グループに追加される新しい連絡先グループの名前を指定します。

DROP GROUP *name*
グループからドロップされる、連絡先グループの名前を指定します。

DESCRIPTION *new description*
オプション。連絡先グループの新しいテキスト記述。

例

gname1 という連絡先グループに cname2 という連絡先を追加します。

```
CALL SYSPROC.ADMIN_CMD( 'update contactgroup gname1 add contact cname2' )
```

使用上の注意

DAS が作成されていて実行中でなければなりません。

コマンドの実行状況は、CALL ステートメントからの結果である SQLCA で戻されます。

UPDATE DATABASE CONFIGURATION コマンド (ADMIN_CMD プロシージャーを使用)

特定のデータベース構成ファイルの中の個々の項目を修正します。データベース構成ファイルは、データベースが作成されたデータベース・パーティションすべてに存在しています。

有効範囲

このコマンドはデフォルトですべてのデータベース・パーティションを更新します。ただし、DBPARTITIONNUM が 1 つのデータベース・パーティションのみ更新するように指定されている場合は除きます。

許可

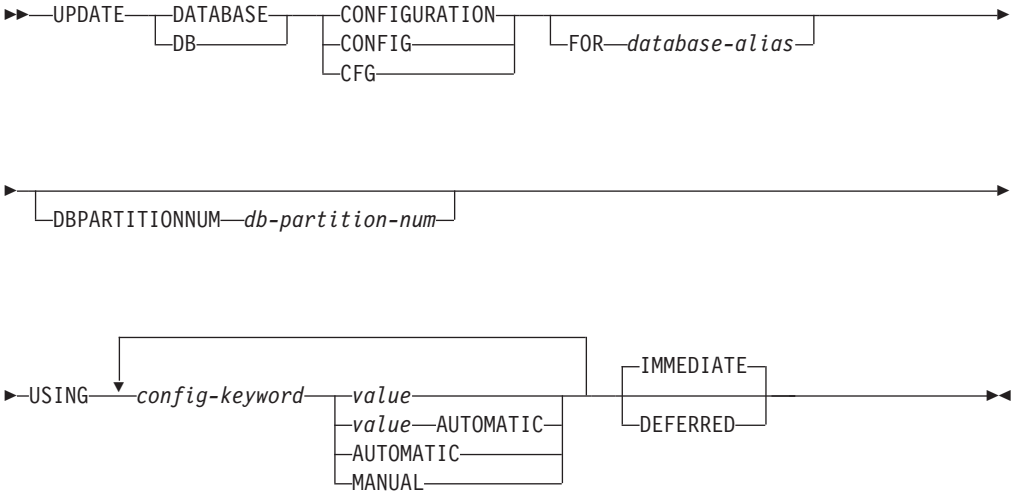
以下のいずれか。

- SYSADM
- SYSCTRL
- SYSMAINT

必要な接続

データベース。データベース接続は、接続されているデータベースが含まれているインスタンスに対してローカルなものでなければなりません。

コマンド構文



コマンド・パラメーター

AUTOMATIC

いくつかの構成パラメーターは AUTOMATIC に設定できます。それにより、DB2 が自動的にこれらのパラメーターを調整し、現行のリソース要件を反映します。 **AUTOMATIC** キーワードをサポートする構成パラメーターのリストは、構成パラメーターのサマリーを参照してください。

AUTOMATIC キーワードと一緒に値を指定すると、自動計算に影響する可能性があります。この動作の具体的な詳細については、構成パラメーターの資料を参照してください。

注: appl_memory、**logindexbuild**、**max_log**、**num_log_span** のデータベース構成パラメーターを AUTOMATIC に設定できるのは、コマンド行プロセッサーを使用する場合に限られます。

DEFERRED
構成ファイルでのみ変更を行います。したがって、加えられた変更は、次にデータベースが再活動化されるときに有効になります。

FOR *database-alias*
構成を更新するデータベースの別名を指定します。データベース接続が既に確立されている場合は、データベース別名を指定する必要はありません。データベース別名は、サーバー上でローカルに定義されていなければなりません。同じデータベース・インスタンスにある別のデータベースについては、その構成ファイルを更新できます。例えば、データベース `db11` にのみ接続されている場合に `update db config for alias db22 using .... immediate` を発行すると、

- `db22` 上にアクティブな接続がない場合、更新の必要なのは構成ファイルだけであるため、更新は成功します。新しい接続 (それによりデータベースがアクティブになる) により、メモリー内で新しい変更が認識されるようになります。
- `db22` 上に他のアプリケーションからのアクティブな接続があるなら、更新はディスク上で動作しますが、メモリー内では動作しません。データベースを再始動する必要があることを示す警告を受け取ることになります。

DBPARTITIONNUM *db-partition-num*
データベース構成の更新が特定のデータベース・パーティションに適用される場合、このパラメーターが使用されることがあります。このパラメーターが指定されていない場合、更新はすべてのデータベース・パーティションに対して有効です。

IMMEDIATE
データベースが稼働している場合に、即時に変更を行います。
IMMEDIATE はデフォルトのアクションです。 ADMIN_CMD プロシージャーにはデータベース接続が必要であるため、接続されているデータベースの動的構成可能パラメーターについては、即時に変更が有効になります。

これは、CLPPlus インターフェースで操作する場合のデフォルトの節でもあります。 **IMMEDIATE** は、CLPPlus プロセッサーの使用時に呼び出す必要はありません。

MANUAL
構成パラメーターの自動チューニングを使用不可にします。パラメーターはその現行の内部値に設定され、自動的な更新は行われなくなります。

USING *config-keyword value*
config-keyword は、更新するデータベース構成パラメーターを指定します。*value* は、パラメーターに割り当てる値を指定します。

例

データベース構成パラメーター **sortheap** を、アプリケーションが現在接続しているデータベース・パーティションに対して 1000 の値に設定します。

```
CALL SYSPROC.ADMIN_CMD ('UPDATE DB CFG USING sortheap 1000')
```

使用上の注意

コマンドの実行状況は、CALL ステートメントからの結果である SQLCA で戻されます。

database-alias は、サーバー上で定義されている別名でなければなりません。

このコマンドは、**DBPARTITIONNUM** が指定されていなければ、すべてのデータベース・パーティションに影響します。

データベース構成パラメーターのリストを表示または印刷するには、SYSIBMADM.DBCFG 管理ビューを使用してください。

すべてのデータベース構成パラメーターを推奨されているデフォルトにリセットするには、ADMIN_CMD プロシージャーを使用し、RESET DATABASE CONFIGURATION コマンドを使用してください。

データベース構成パラメーターを変更するには、ADMIN_CMD プロシージャーを通じて UPDATE DATABASE CONFIGURATION コマンドを使用してください。例えば、ロギング・モードを 『archival logging』 に、ZELLMART というデータベースを含む単一パーティション・データベース環境で変更する場合は、次を使用します。

```
CALL SYSPROC.ADMIN_CMD ('update db cfg for zellmart using logretain recovery')
```

logretain 構成パラメーターが変更されたことをチェックするには、次を使用します。

```
SELECT * FROM SYSIBMADM.DBCFG WHERE NAME='logretain'
```

特定のデータベース・パーティションのデータベース構成パラメーターを更新するには、以下を行います。

1. **DB2NODE** 変数にデータベース・パーティション番号を設定します。
2. データベース・パーティションに接続する。
3. ADMIN_CMD プロシージャーで UPDATE DATABASE CONFIGURATION コマンドを使用してデータベース構成パラメーターを更新する。
4. データベース・パーティションから切断する。

あるいは、**DBPARTITIONNUM** を使用することもできます。例えば、DBPARTITIONNUM を使用してロギング・モードを 1 つだけの特定のパーティション (30) に更新するには、次を使用します。

```
CALL SYSPROC.ADMIN_CMD ('update db cfg for zellmart dbpartitionnum 30 using
 logretain recovery')
```

DB2 構成パラメーターと、各種データベース・ノードに使用できる値についての詳細は、個々の構成パラメーターの説明を参照してください。これらのパラメーター

の値は、構成するデータベース・ノードの各タイプ (サーバー、クライアント、またはリモート・クライアントを持つサーバー) によって異なります。

すべてのパラメーターを更新できるわけではありません。

データベース構成ファイルへの変更の一部は、ファイルがメモリーにロードされた後にのみ有効になります。これを行う前にすべてのアプリケーションはデータベースから切断されている必要があります。オンラインで構成できるパラメーターと構成できないパラメーターについては、構成パラメーターの一覧をご覧ください。

エラーが発生した場合、データベース構成ファイルは変更されません。チェックサムが無効な場合、データベース構成ファイルは更新できません。適当なコマンドを使用しないでデータベース構成ファイルを変更するとこれが発生することがあります。これが発生する場合、データベースをリストアしてデータベース構成ファイルをリセットする必要があります。

UPDATE DATABASE MANAGER CONFIGURATION コマンド (ADMIN_CMD プロシージャーを使用)

現在接続されているデータベースを含むインスタンスについて、データベース・マネージャーの構成ファイルの中の個々の項目を修正します。

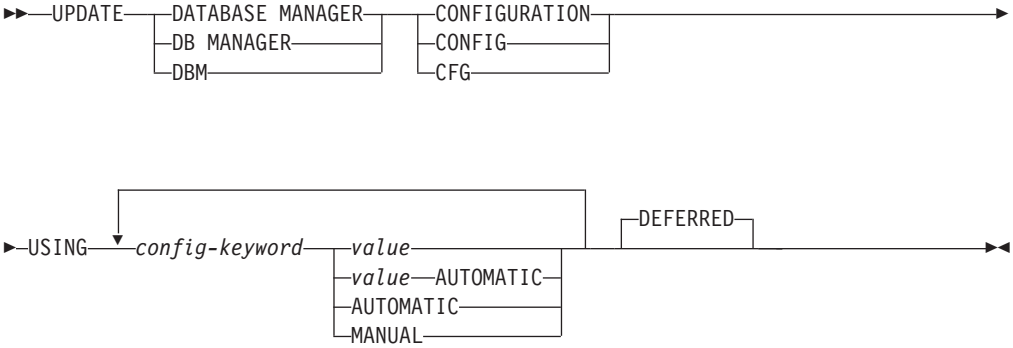
許可

SYSADM

必要な接続

データベース

コマンド構文



コマンド・パラメーター

AUTOMATIC

いくつかの構成パラメーターは AUTOMATIC に設定できます。それにより、DB2 が自動的にこれらのパラメーターを調整し、現行のリソース要件を反映します。 **AUTOMATIC** キーワードをサポートする構成パラメーターのリストは、構成パラメーターのサマリーを参照してください。

AUTOMATIC キーワードと一緒に値を指定すると、自動計算に影響する可能性があります。この動作の具体的な詳細については、構成パラメーターの資料を参照してください。

注: federated_async データベース・マネージャー構成パラメーターは、コマンド行プロセッサーを使用することによってのみ AUTOMATIC に設定できます。

DEFERRED
構成ファイルでのみ変更を行います。したがって、加えられた変更は、インスタンスの再始動時に有効になります。これはデフォルトです。

これは、CLPPlus インターフェースで操作する場合のデフォルトの節です。**DEFERRED** は、CLPPlus プロセッサーの使用時に呼び出す必要はありません。

MANUAL
構成パラメーターの自動チューニングを使用不可にします。パラメーターはその現行の内部値に設定され、自動的な更新は行われなくなります。

USING *config-keyword value*
更新するデータベース・マネージャー構成パラメーターを指定します。構成パラメーターのリストは、構成パラメーターのサマリーを参照してください。 *value* は、パラメーターに割り当てる値を指定します。

例

データベース・マネージャー構成の診断レベルを 1 に更新します。

```
CALL SYSPROC.ADMIN_CMD('db2 update dbm cfg using DIAGLEVEL 1')
```

使用上の注意

データベース・マネージャー構成パラメーターのリストを表示または印刷するには、SYSIBMADM.DBMCFG 管理ビューを使用してください。データベース・マネージャー構成パラメーターを推奨されているデータベース・マネージャーのデフォルトにリセットするには、ADMIN_CMD プロシージャーを通じて RESET DATABASE MANAGER CONFIGURATION コマンドを使用してください。データベース・マネージャーの構成パラメーターと、構成されている各種データベース・ノード (サーバー、クライアント、またはリモート・クライアントを持つサーバー) に適したこれらのパラメーターの値については、個々の構成パラメーターの説明を参照してください。

すべてのパラメーターを更新できるわけではありません。

データベース・マネージャー構成ファイルへの変更の一部は、ファイルがメモリーにロードされた後にのみ有効になります。オンラインで構成できるパラメーターと構成できないパラメーターについては、構成パラメーターの一覧をご覧ください。即時にリセットされないサーバー構成パラメーターは、 db2start の実行中にリセットされます。クライアント構成パラメーターの場合、パラメーターは次にアプリケーションを開始するときにリセットされます。クライアントがコマンド行プロセッサーである場合は、TERMINATE を呼び出すことが必要です。

エラーが生じた場合には、データベース・マネージャー構成ファイルは変更されません。

データベース・マネージャー構成ファイルは、そのチェックサムが無効であると、更新することができません。このような状況は、データベース・マネージャー構成ファイルが変更されて、適切なコマンドが使用されていない場合に起こります。チェックサムが無効な場合は、データベース・マネージャーを再インストールして、データベース・マネージャー構成ファイルをリセットする必要があります。

現行のインスタンスの **SVCENAME**、または **TPNAME** データベース・マネージャー構成パラメーターを更新するとき、LDAP サポートを使用することができて、このインスタンスに LDAP サーバーが登録されている場合は、LDAP サーバーが新しい値に更新されます。

コマンドの実行状況は、CALL ステートメントからの結果である SQLCA で戻されます。

更新は、接続されているデータベースが含まれているデータベース・インスタンスに対してのみ可能です。

動的更新がサポートされているパラメーターの場合、**IMMEDIATE** キーワードが指定されていなくても、それを動的に更新することが試みられます。使用される許可は現在の SYSTEM_USER ID です。

UPDATE HEALTH NOTIFICATION CONTACT LIST コマンド (ADMIN_CMD プロシージャーを使用)

インスタンスによって発行されるヘルス・アラートについての通知の連絡先リストを更新します。

許可

以下のいずれか。

- SYSADM
- SYSCTRL
- SYSMAINT

必要な接続

データベース

コマンド構文

コマンド・パラメーター

ADD GROUP *name*
インスタンスのヘルスの通知を受ける新しい連絡先グループを追加します。

ADD CONTACT *name*
インスタンスのヘルスの通知を受ける新しい連絡先を追加します。

DROP GROUP *name*
インスタンスのヘルスの通知を受ける連絡先のリストから、連絡先グループを除去します。

DROP CONTACT *name*
インスタンスのヘルスの通知を受ける連絡先のリストから、連絡先を除去します。

例

ヘルス通知連絡先リストに連絡先グループ gname1 を追加します。

```
CALL SYSPROC.ADMIN_CMD( 'update notification list add group gname1' )
```

使用上の注意

コマンドの実行状況は、CALL ステートメントからの結果である SQLCA で戻されます。

UPDATE HISTORY コマンド (ADMIN_CMD プロシージャーを使用)

現在接続されているデータベース・パーティションのデータベース履歴レコード項目にあるロケーション、装置タイプ、コメント、または状況を更新します。

許可

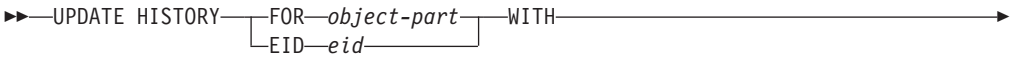
以下のいずれか。

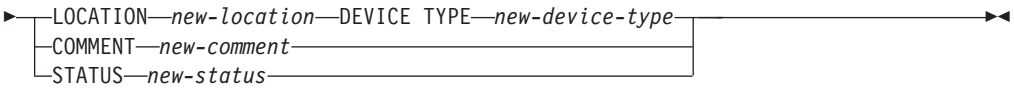
- SYSADM
- SYSCTRL
- SYSMAINT
- DBADM

必要な接続

データベース

コマンド構文





コマンド・パラメーター

FOR *object-part*

更新される履歴項目の ID を指定します。この ID は、タイム・スタンプと 001 から 999 までのオプションのシーケンス番号で構成されます。項目の状況を更新するためにこのパラメーターを使用することはできません。項目の状況を更新するには、代わりに EID を指定してください。

EID *eid*

履歴項目 ID を指定します。

LOCATION *new-location*

バックアップ・イメージの新しい物理ロケーションを指定します。このパラメーターの解釈は装置タイプに依存します。

DEVICE TYPE *new-device-type*

バックアップ・イメージを保管する新しい装置タイプを指定します。有効な装置タイプは次のとおりです。

D ディスク

K ディスケット

T テープ

A Tivoli Storage Manager

F スナップショット・バックアップ

U ユーザー出口

P パイプ

N Null 装置

X XBSA

Q SQL ステートメント

O その他

COMMENT *new-comment*

項目を記述する新しい注釈を指定します。

STATUS *new-status*

項目の新しい状況を指定します。バックアップ項目のみがその状況を更新できます。有効な値は以下のとおりです。

A アクティブ。バックアップ・イメージはアクティブ・ログ・チェーン上にあります。ほとんどの項目はアクティブです。

I 非アクティブ。現行のログ・シーケンス (現行のログ・チェーンとも言う) に対応しなくなったバックアップ・イメージには、非アクティブのフラグが立てられます。

E 期限切れ。アクティブ・イメージの数が NUM_DB_BACKUPS を超えたために不要になったバックアップ・イメージは、期限切れのフラグが立てられます。

D 削除済み。リカバリーに使用可能でなくなったバックアップ・イメージは、削除済みとしてマークされることになります。

X 削除しません。DB2HISTORY_STATUS_DO_NOT_DELETE のマークが付いたリカバリー・データベース履歴レコード項目は、PRUNE HISTORY コマンドの呼び出し、PRUNE HISTORY を指定した ADMIN_CMD プロシージャーの実行、db2Prune API の呼び出し、またはリカバリー・データベース履歴レコードファイルの自動プルーニングにより整理されません。
DB2HISTORY_STATUS_DO_NOT_DELETE 状況を使用すると、キー・リカバリー・ファイルの項目が整理されたり、それらに関連付けられたリカバリー・オブジェクトが削除されたりしないように保護できます。ログ・ファイル、バックアップ・イメージ、およびロード・コピー・イメージのみに
DB2HISTORY_STATUS_DO_NOT_DELETE のマークを付けることができます。

例

1997 年 4 月 13 日午前 10 時 00 分に取った全データベース・バックアップのデータベース履歴レコード項目を更新するには、次のように入力します。

```
CALL SYSPROC.ADMIN_CMD('update history
   for 19970413100000001 with location
   /backup/dbbackup.1 device type d')
```

使用上の注意

データベース履歴レコードの主な用途は情報を記録することですが、履歴に含まれるデータは、自動リストア操作で直接に使用されます。 **AUTOMATIC** オプションを指定したリストアにおいては、リストア・ユーティリティーによりバックアップ・イメージとそのロケーションの履歴が参照および使用されることにより、自動リストア要求が処理されます。自動リストア機能を使用する場合に、バックアップ・イメージが作成されてから再配置されているなら、現在のロケーションを反映するよう、それらのイメージのデータベース履歴レコードを更新することをお勧めします。データベース履歴の中のバックアップ・イメージのロケーションが更新されない場合、自動リストア処理においてはバックアップ・イメージを見つけることができなくなりますが、手動リストア・コマンドは正常に使用できます。

コマンドの実行状況は、CALL ステートメントからの結果である SQLCA で戻されます。

object-part または *eid* が、接続されているデータベース・パーティションのログ履歴項目を指していなければなりません。

ADMIN_CMD プロシージャーを使用する UPDATE STMM TUNING DBPARTITIONNUM コマンド

ユーザー設定のセルフチューニング・メモリー・マネージャー (STMM) の調整データベース・パーティションを更新します。

許可

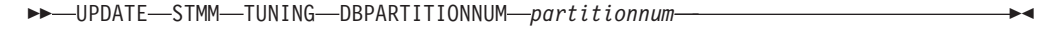
ステートメントの許可 ID によって保持されている特権には、少なくとも以下のいずれかの権限が含まれていなければなりません。

- DBADM
- DATAACCESS
- SQLADM

必要な接続

データベース

コマンド構文



コマンド・パラメーター

partitionnum

partitionnum は整数です。-1 または存在しないデータベース・パーティション番号が使用される場合、DB2 は STMM メモリー・チューナーを実行する適切なデータベース・パーティションを自動的に選択します。

例

ユーザー設定のセルフチューニング・メモリー・マネージャー (STMM) の調整データベース・パーティションを更新して、データベース・パーティション 3 にします。

```
CALL SYSPROC.ADMIN_CMD( 'update stmm tuning dbpartitionnum 3' )
```

使用上の注意

STMM 調整プロセスは、ユーザー設定の STMM 調整データベース・パーティション番号の値の変更を定期的に検査します。 STMM 調整プロセスは、*partitionnum* が存在しており、それがアクティブなデータベース・パーティションであれば、ユーザー設定の STMM 調整データベース・パーティションに移ります。 このコマンドが STMM 調整データベース・パーティション番号を変更すると、現在の STMM 調整データベース・パーティション番号は即時に変更されます。

コマンドの実行状況は、CALL ステートメントからの結果である SQLCA で戻されます。

このコマンドは、その変更内容を ADMIN_CMD プロシージャーでコミットします。

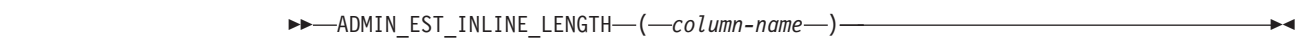
ADMIN_EST_INLINE_LENGTH 関数 - データのインライン化に必要な長さを見積もる

ADMIN_EST_INLINE_LENGTH 関数は、XML 列、BLOB 列、CLOB 列、または DBCLOB 列に保管されているデータをインライン化するために必要なインライン長の見積もりを戻します。

データをインライン化できない場合、関数は負の値を戻します。

データがすでにインライン化されている場合、関数はインライン化データの実際の長さを戻します。

構文



スキーマは SYSIBM です。

戻り値

この関数は、データの見積インライン長 (バイト) を表す INTEGER 値か、あるいは以下のいずれかの値を戻します。

NULL 入力が NULL であることを示します。

-1 列値をインライン化するのを可能にする有効なインライン長がないため、データをインライン化できないことを示します。

-2 文書が DB2 for Linux, UNIX, and Windows バージョン 9.7 より前のリリースに挿入および保管されたため、文書の見積インライン長を判別できないことを示します。

関数のパラメーター

column-name
データ・タイプ XML、BLOB、CLOB、または DBCLOB を持つ基本表の列を識別します (SQLSTATE 42884)。この列は、式に基づいて生成されていない基本表の列を直接的または間接的に参照する必要があります (SQLSTATE 42815)。

例

例 1: 以下の例は、TAB1 表の XML 列 xml_doc1 に含まれている 3 つの XML 文書の見積インライン長を戻します。

```
db2 => SELECT PK, ADMIN_IS_INLINED(xml_doc1) as IS_INLINED,
              ADMIN_EST_INLINE_LENGTH(xml_doc1) as EST_INLINE_LENGTH
       from TAB1
```

この照会では、以下の出力が結果として戻ります。

```
PK          IS_INLINED EST_INLINE_LENGTH
----------- ---------- -----------------
          1          1               292
```

```
          2          0               450
          3          0               454

  3 record(s) selected.
```

この例では、ADMIN_IS_INLINED 関数は最初の文書がインライン化されていることを示しています。このため、ADMIN_EST_INLINE_LENGTH 関数はインライン化 XML 文書の実際の長さを戻します。 2 番目の文書はインライン化されていないため、ADMIN_EST_INLINE_LENGTH 関数は、2 番目の XML 文書をインライン化するために必要な見積インライン長を戻します。

例 2: 以下の例は、TAB1 表の XML 列 xml_doc1 に含まれている 1 つの XML 文書の見積インライン長を戻します。この例には、述部が含まれます。

```
db2 => SELECT PK, ADMIN_IS_INLINED(xml_doc1) as IS_INLINED,
              ADMIN_EST_INLINE_LENGTH(xml_doc1) as EST_INLINE_LENGTH
       from TAB1 where PK=2
```

この照会では、以下の出力が結果として戻ります。

```
PK          IS_INLINED EST_INLINE_LENGTH
----------- ---------- -----------------
          2          0               450

  1 record(s) selected.
```

例 3: 以下の例は、TAB1 表の CLOB 列 clob_1 に含まれている 3 つの CLOB データの見積インライン長を戻します。

```
db2 => SELECT PK, ADMIN_IS_INLINED(clob_1) as IS_INLINED,
              ADMIN_EST_INLINE_LENGTH(clob_1) as EST_INLINE_LENGTH
       from TAB1
```

この照会では、以下の出力が結果として戻ります。

```
PK          IS_INLINED EST_INLINE_LENGTH
----------- ---------- -----------------
          1          1                68
          2          0              3665
          3          0                -1

   3 record(s) selected.
```

使用上の注意

- XML 列がサポートされるのは、XML 文書が DB2 for Linux, UNIX, and Windows バージョン 9.7 以降を使用して挿入された場合のみです。これより前のリリースで挿入された XML 文書の場合、保管形式は異なります。正しくない保管形式を検出すると、ADMIN_EST_INLINE_LENGTH 関数は値 -2 を戻します。
- 列のインライン長を増やすことを計画している場合、この長さは減らすことができないことに注意してください。
- インライン長を増やすと、行サイズの合計も増加し、バッファー・プールのパフォーマンスにも影響を与える場合があります。行サイズの合計には、以下の制限があります。

表 68. 行サイズの制限

ページ・サイズ	行サイズの制限	インライン長の制限
4K	4005	4001

表 68. 行サイズの制限 (続き)

ページ・サイズ	行サイズの制限	インライン長の制限
8K	8101	8097
16K	16 293	16 289
32K	32 677	32 673

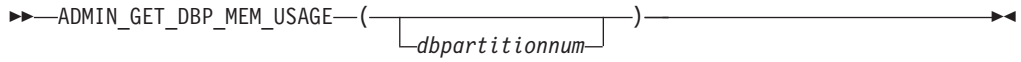
- XML ストレージ・オブジェクトのページ・サイズが基本表のページ・サイズと同じでない場合は、見積インライン長は正確でない可能性があります。

ADMIN_GET_DBP_MEM_USAGE 表関数 - インスタンスの合計メモリー消費量の取得

ADMIN_GET_DBP_MEM_USAGE 表関数は、指定されたインスタンスの合計メモリー消費量を取得します。

ADMIN_GET_DBP_MEM_USAGE 表関数は、オプション入力引数 *dbpartitionnum* (INTEGER タイプ) を取ります。これは、有効なデータベース・パーティション番号を指定し、その単一のデータベース・パーティションの統計のみ戻します。引数が省略される場合、すべてのアクティブなデータベース・パーティションの統計が戻されます。パーティション・データベース環境では、*dbpartitionnum* に -1 または NULL 値を指定すると、現在接続されているパーティションからデータが戻されます。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbpartitionnum
メモリー使用量統計が取得されるデータベース・パーティションを指定する、タイプ integer のオプション入力引数。 -1 または NULL 値が指定される場合、データは現在接続中のパーティションから戻されます。

許可

ADMIN_GET_DBP_MEM_USAGE 関数に対する EXECUTE 特権。

戻される情報

表 69. *ADMIN_GET_DBP_MEM_USAGE* の結果セット

列名	データ・タイプ	説明
DBPARTITIONNUM	SMALLINT	メモリー使用量統計が取得されるデータベース・パーティション番号。

表 69. ADMIN_GET_DBP_MEM_USAGE の結果セット (続き)

列名	データ・タイプ	説明
MAX_PARTITION_MEM	BIGINT	インスタンス・メモリー限度が実施される場合にデータベース・パーティションで消費することが許可されるインスタンス・メモリーの最大量 (バイト)。
CURRENT_PARTITION_MEM	BIGINT	データベース・パーティションで現在消費されているインスタンス・メモリーの量 (バイト)。
PEAK_PARTITION_MEM	BIGINT	データベース・パーティション内のインスタンス・メモリーの消費量のピーク水準点または最高水準点 (バイト)。

例

例 1: メモリー使用量統計をデータベース・パーティション 3 から取得します。

```
SELECT * FROM TABLE (SYSPROC.ADMIN_GET_DBP_MEM_USAGE(3)) AS T

DBPARTITIONNUM   MAX_PARTITION_MEM   CURRENT_PARTITION_MEM   PEAK_PARTITION_MEM
--------------   ----------------   --------------------   ------------------
             3          500000000              381000000            481000000

  1 record(s) selected.
```

例 2: 現在接続中のパーティションからメモリー使用量統計を取得します (ユーザーがパーティション 2 でデータベースに接続されていると想定)。

```
SELECT * FROM TABLE (SYSPROC.ADMIN_GET_DBP_MEM_USAGE(-1)) AS T

DBPARTITIONNUM   MAX_PARTITION_MEM   CURRENT_PARTITION_MEM   PEAK_PARTITION_MEM
--------------   ----------------   --------------------   ------------------
             2          500000000              381000000            481000000

  1 record(s) selected.
```

例 3: メモリー使用量統計をすべてのパーティションから取得します。

```
SELECT * FROM TABLE (SYSPROC.ADMIN_GET_DBP_MEM_USAGE()) AS T

DBPARTITIONNUM   MAX_PARTITION_MEM   CURRENT_PARTITION_MEM   PEAK_PARTITION_MEM
--------------   ----------------   --------------------   ------------------
             0          500000000              381000000            481000000
             1          500000000              381000000            481000000
             2          500000000              381000000            481000000
             3          500000000              381000000            481000000

  4 record(s) selected.
```

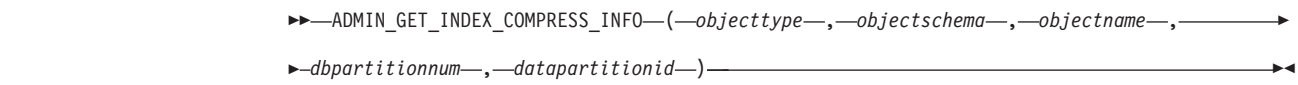
例 4: メモリー使用量統計をメガバイト (MB) 単位の値で取得します。

```
SELECT DBPARTITIONNUM, MAX_PARTITION_MEM/1048576 AS MAX_MEM_MB,
   CURRENT_PARTITION_MEM/1048576 AS CURRENT_MEM_MB, PEAK_PARTITION_MEM/1048576
   AS PEAK_MEM_MB FROM TABLE (SYSPROC.ADMIN_GET_DBP_MEM_USAGE()) AS T

DBPARTITIONNUM  MAX_MEM_MB  CURRENT_MEM_MB  PEAK_MEM_MB
--------------  ----------  --------------  -----------
             0        4590            1107         1107
             1        4590            1108         1108
             2        4590            1106         1106

  3 record(s) selected.
```

ADMIN_GET_INDEX_COMPRESS_INFO 表関数 - 圧縮索引情報を戻す

ADMIN_GET_INDEX_COMPRESS_INFO 表関数は、非圧縮索引について索引圧縮による節約の可能性を戻すか、カタログからの索引圧縮統計をレポートします。

構文



スキーマは SYSPROC です。

表関数パラメーター

objecttype
オブジェクト・タイプを示す、タイプ VARCHAR(1) の入力引数。値は以下のいずれかでなければならず、大/小文字を区別します。

- 'T'、NULL、または空ストリング (表を示す場合)
- 'I' (索引の場合)

objectschema
オブジェクト・スキーマを指定する、タイプ VARCHAR (128) の大/小文字を区別する入力パラメーター。

objecttype が 'T'、NULL、または空ストリング (") である場合、*objectschema* は表スキーマを示します。

- *objectschema* が指定され、*objectname* が NULL または空ストリング (") の場合、指定されたスキーマのすべての表に関する情報が戻されます。
- *objectschema* と *objectname* の両方が指定される場合、指定された表のすべての索引に関する情報が戻されます。

objecttype が 'I' である場合、*objectschema* は索引スキーマを示します。

- *objectschema* が指定され、*objectname* が NULL または空ストリング (") の場合、指定されたスキーマのすべての索引に関する情報が戻されます。
- *objectschema* と *objectname* の両方が指定される場合、指定された索引に関する情報が戻されます。
- *objectschema* と *objectname* のどちらも指定されない場合、すべてのスキーマのすべての索引に関する情報が戻されます。

objectname が指定され、*objectschema* が指定されない場合、関数が SQL エラーを戻します。値が NULL または空ストリング (") である場合、パラメーター値は未指定であるといいます。

objectname
オブジェクト名を指定する、タイプ VARCHAR (128) の大/小文字を区別する入力パラメーター。 *objectschema* パラメーターの説明を参照してください。

dbpartitionnum
データベース・パーティション番号を指定するタイプ INTEGER の入力パラメーター。指定されている場合、指定されたデータベース・パーティションに存在する索引に関する情報のみが戻されます。すべてのアクティブ・データベース・

パーティションに関してデータが戻されるように指定するには、*dbpartitionnum* パラメーター値を -2 または NULL に設定します。非パーティション・データベース環境では、-2 または NULL を指定します。

datapartitionid
データ・パーティション ID を指定するタイプ INTEGER の入力パラメーター。指定されている場合、指定されたデータ・パーティションで定義された索引パーティションに関する情報のみが戻されます。データ・パーティション ID は、SYSCAT.DATAPARTITIONS ビューにある DATAPARTITIONID に対応します。すべてのデータ・パーティションに関してデータが戻されるように指定するには、*datapartitionid* パラメーター値を -2 または NULL に設定します。非パーティション索引の場合は、-2、0、または NULL を指定します。

許可

ADMIN_GET_INDEX_COMPRESS_INFO 表関数に対する EXECUTE 特権。

例

データベースのマイグレーション後は、すべての既存の索引は圧縮解除されます。データベース・パーティション番号 2 に置かれている、データ・パーティション ID が 3 の表「S.T1」について、既存の索引の索引圧縮による節約の可能性を見積もることもできます。この例では、S はスキーマ名、T1 は表名であり、T1 は圧縮されていません。

```
SELECT compress_attr, iid, dbpartitionnum, index_compressed,
    pct_pages_saved, num_leaf_pages_saved
  FROM TABLE(sysproc.admin_get_index_compress_info('', 'S', 'T1', 2, 3))
  AS t
```

以下は、このステートメントからの出力例です。

```
COMPRESS_ATTR       IID DBPARTITIONNUM INDEX_COMPRESSED ...
------------- --------- -------------- ---------------- ...
N                     1              2 N                ...
N                     2              2 N                ...

... PCT_PAGES_SAVED NUM_LEAF_PAGES_SAVED
... --------------- --------------------
...              50                  200
...              45                  150
```

圧縮による節約を行う価値があると判断し、索引圧縮を使用可能にしたいと考えます。

```
ALTER INDEX INDEX1 compress yes
ALTER INDEX INDEX2 compress yes
REORG INDEXES all FOR table S.T1
```

時間の経過とともに、表の新規索引を作成する必要について判断し、圧縮する前に索引圧縮によってこれらの索引についてどれほど節約されるかを見積もることもできます。さらに、すでに圧縮済みの索引から圧縮統計を参照することもできます。

```
SELECT compress_attr, iid, dbpartitionnum, index_compressed,
    pct_pages_saved,  num_leaf_pages_saved
  FROM TABLE(sysproc.admin_get_index_compress_info('', 'S', 'T1', 2, 3))
  AS t
```

以下は、このステートメントからの出力例です。

```
COMPRESS_ATTR       IID DBPARTITIONNUM INDEX_COMPRESSED ...
------------- --------- -------------- ---------------- ...
Y                     1              2 Y                ...
Y                     2              2 Y                ...
N                     3              2 N                ...
N                     4              2 N                ...

... PCT_PAGES_SAVED NUM_LEAF_PAGES_SAVED
... --------------- --------------------
...              -1                   -1
...              -1                   -1
...              58                  230
...              49                  140
```

index_compressed 列で示されているように、最初の 2 つの索引がすでに圧縮されているため、ステートメントはシステム・カタログから値が戻されます。この場合、カタログからの値は収集されませんでした。

表に対して RUNSTATS を実行した後、索引関数を次回実行するときに、訂正された結果が生成されます。

```
RUNSTATS ON TABLE S.T1 FOR INDEXES ALL
SELECT compress_attr, iid, dbpartitionnum, index_compressed,
    pct_pages_saved, num_leaf_pages_saved
  FROM TABLE(sysproc.admin_get_index_compress_info('', 'S', 'T1', 2, 3))
  AS t
```

以下は、このステートメントからの出力例です。

```
COMPRESS_ATTR       IID DBPARTITIONNUM INDEX_COMPRESSED ...
------------- --------- -------------- ---------------- ...
Y                     1              2 Y                ...
Y                     2              2 Y                ...
N                     3              2 N                ...
N                     4              2 N                ...

... PCT_PAGES_SAVED NUM_LEAF_PAGES_SAVED
... --------------- --------------------
...              50                  200
...              45                  150
...              58                  230
...              49                  140
```

ADMIN_GET_INDEX_COMPRESS_INFO 表関数のメタデータ

表 70. ADMIN_GET_INDEX_COMPRESS_INFO 表関数のメタデータ

列名	データ・タイプ	説明
INDSCHEMA	VARCHAR(128)	索引が定義されているスキーマの名前。
INDNAME	VARCHAR(128)	索引名。
TABSCHEMA	VARCHAR(128)	表が定義されているスキーマの名前。
TABNAME	VARCHAR(128)	表名。
DBPARTITIONNUM	SMALLINT	データベース・パーティション番号。
IID	SMALLINT	索引の ID。
DATAPARTITIONID	INTEGER	データ・パーティション ID。
COMPRESS_ATTR	CHAR(1)	索引の COMPRESSION 属性の状態。 • Y = 索引圧縮は使用可能 • N = 索引圧縮は使用不可

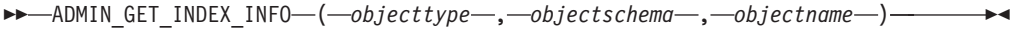
表 70. ADMIN_GET_INDEX_COMPRESS_INFO 表関数のメタデータ (続き)

列名	データ・タイプ	説明
INDEX_COMPRESSED	CHAR(1)	物理索引形式。 • Y = 索引は圧縮形式である • N = 索引は非圧縮形式である 物理索引形式が圧縮属性と一致しない場合、索引を定義済み形式に変換するには索引の再編成が必要です。この関数を実行したときに表または索引でエラーが発生した場合、この値は NULL です。
PCT_PAGES_SAVED	SMALLINT	索引が物理的に圧縮されていない (INDEX_COMPRESSED が N である) 場合、この値は、索引が実際に圧縮されているかのように、節約されるリーフ・ページの見積パーセントを表します。索引が物理的に圧縮されている (INDEX_COMPRESSED が Y である) 場合、この値は、システム・カタログ・ビュー (SYSCAT.INDEXES または SYSCAT.INDEXPARTITIONS のいずれか) からの PCTPAGESSAVED 値をレポートします。 **注:** この値は、パーティション・データベース環境のデータベース・パーティションごとの索引または索引パーティションの各項目で同じになります。この関数を実行したときに表または索引でエラーが発生した場合、この値は NULL です。
NUM_LEAF_PAGES_SAVED	BIGINT	索引が物理的に圧縮されていない (INDEX_COMPRESSED が N である) 場合、この値は、索引が実際に圧縮されているかのように、節約されるリーフ・ページ数の見積もりを表します。索引が物理的に圧縮されている (INDEX_COMPRESSED が Y である) 場合、この値は、システム・カタログ・ビュー (SYSCAT.INDEXES または SYSCAT.INDEXPARTITIONS のいずれか) から PCTPAGESSAVED および NLEAF 値に基づいて、節約される計算済みリーフ・ページ数をレポートします。PCTPAGESSAVED または NLEAF のいずれかが無効値 (-1) である場合、この値も -1 に設定されます。 **注:** この値は、パーティション・データベース環境のデータベース・パーティションごとの索引または索引パーティションの各項目で同じになります。この関数を実行したときに表または索引でエラーが発生した場合、この値は NULL です。

ADMIN_GET_INDEX_INFO 表関数 - 索引情報を戻す

ADMIN_GET_INDEX_INFO 表関数は、圧縮情報および索引の論理および物理サイズなど、カタログ・ビューで使用できない索引情報を戻します。

構文



スキーマは SYSPROC です。

表関数パラメーター

objecttype
オブジェクト・タイプを示す、タイプ VARCHAR(1) の入力引数。値は以下のいずれかでなければならず、大/小文字を区別します。

- 'T'、NULL、または空ストリング (') (表を示す場合)
- 'I' (索引の場合)

objectschema
オブジェクト・スキーマを指定する、タイプ VARCHAR (128) の大/小文字を区別する入力パラメーター。

objecttype が 'T'、NULL、または空ストリング (') である場合、*objectschema* は表スキーマを示します。

- *objectschema* が指定され、*objectname* が NULL または空ストリング (') の場合、指定されたスキーマのすべての表に関する情報が戻されます。
- *objectschema* と *objectname* の両方が指定される場合、指定された表のすべての索引に関する情報が戻されます。

objecttype が 'I' である場合、*objectschema* は索引スキーマを示します。

- *objectschema* が指定され、*objectname* が NULL または空ストリング (') の場合、指定されたスキーマのすべての索引に関する情報が戻されます。
- *objectschema* と *objectname* の両方が指定される場合、指定された索引に関する情報が戻されます。
- *objectschema* と *objectname* のどちらも指定されない場合、すべてのスキーマのすべての索引に関する情報が戻されます。

objectname が指定され、*objectschema* が指定されない場合、関数が SQL エラーを戻します。値が NULL または空ストリング (') である場合、パラメーター値は未指定であるといいます。

objectname
オブジェクト名を指定する、タイプ VARCHAR (128) の大/小文字を区別する入力パラメーター。 *objectschema* パラメーターの説明を参照してください。

許可

ADMIN_GET_INDEX_INFO 表関数に対する EXECUTE 特権。

例

表のいくつかの索引に対して索引圧縮を有効にした後で、どの索引が圧縮されるのか、および圧縮するためにはどの索引を再作成する必要があるかを知りたいとします。この例では、S はスキーマ名、T1 は表名です。

```
db2 SELECT iid, compress_attr, index_compressed
    FROM TABLE(sysproc.admin_get_index_info('','S','T1')) AS t
```

以下はこの照会の出力例です。

```
IID       COMPRESS_ATTR INDEX_COMPRESSED
--------- ------------- ----------------
        1 Y             Y
        2 Y             Y
        3 Y             N
        4 N             N
```

さらに、スキーマ S2 のすべての索引の他の索引情報を参照するとします。この例では、それぞれ次の意味を持つとします。

- T2 = 2 つのデータ・パーティションを持つパーティション表
- T3 = 非パーティション表
- IND_1 = T2 上の非パーティション索引
- IND_2 = T2 上のパーティション索引
- IND_3 = T2 上のパーティション索引
- IND_4 = T3 上の索引
- IND_5 = T3 上の索引

```
db2 SELECT tabname, indname, iid,index_partitioning, datapartitionid,
   index_object_l_size, index_object_p_size, index_requires_rebuild,
   large_rids, FROM TABLE(sysproc.admin_get_index_info('I','S2','')) AS t
```

以下はこの照会の出力例です。

```
TABNAME INDNAME IID       INDEX_PARTITIONING DATAPARTITIONID
------- ------- --------- ------------------ ---------------
T2      IND_1           1 N                              0
T2      IND_2           2 P                              1
T2      IND_2           2 P                              2
T2      IND_3           3 P                              1
T2      IND_3           3 P                              2
T3      IND_4           4                                0
T3      IND_5           5                                0
```

この手順からの出力 (続き):

```
INDEX_OBJECT_L_SIZE INDEX_OBJECT_P_SIZE INDEX_REQUIRES_REBUILD LARGE_RIDS
------------------- ------------------- ---------------------- ----------
                 50                  51                      N          Y
                 40                  40                      N          Y
                 45                  45                      N          Y
                 40                  40                      N          Y
                 45                  45                      N          Y
                 20                  20                      N          Y
                 20                  20                      N          Y
```

ADMIN_GET_INDEX_INFO 表関数のメタデータ

表 71. ADMIN_GET_INDEX_INFO 表関数のメタデータ

列名	データ・タイプ	説明
INDSCHEMA	VARCHAR(128)	索引が定義されているスキーマの名前。
INDNAME	VARCHAR(128)	索引名。
TABSCHEMA	VARCHAR(128)	表が定義されているスキーマの名前。
TABNAME	VARCHAR(128)	表名。
DBPARTITIONNUM	SMALLINT	データベース・パーティション番号。
IID	SMALLINT	索引の ID。
DATAPARTITIONID	INTEGER	データ・パーティション ID。

表 71. ADMIN_GET_INDEX_INFO 表関数のメタデータ (続き)

列名	データ・タイプ	説明
COMPRESS_ATTR	CHAR(1)	索引の COMPRESSION 属性の状態。 • Y = 索引圧縮は使用可能 • N = 索引圧縮は使用不可
INDEX_COMPRESSED	CHAR(1)	物理索引形式。 • Y = 索引は圧縮形式である • N = 索引は非圧縮形式である 物理索引形式が圧縮属性と一致しない場合、索引を定義済み形式に変換するには索引の再編成が必要です。この関数を実行したときに表または索引でエラーが発生した場合、この値は NULL です。
INDEX_PARTITIONING	CHAR(1)	索引のパーティション特性を示します。 • N = 非パーティション索引 • P = パーティション索引 • ブランク = 索引はパーティション表にはない
INDEX_OBJECT_L_SIZE	BIGINT	索引オブジェクトの論理サイズ。非パーティション表の場合、これは、表に定義されるすべての索引に対して論理的に割り振られるディスク・スペースの量になります。パーティション表の非パーティション索引の場合、これは、索引に対して論理的に割り振られるディスク・スペースの量になります。パーティション表のパーティション索引の場合、これは、データ・パーティションに定義されるすべての索引パーティションに対して論理的に割り振られるディスク・スペースの量になります。すべてのサイズはキロバイト (KB) 単位でレポートされます。 論理サイズとは、表またはデータ・パーティションが認識するスペースの量のことです。このサイズは表またはデータ・パーティションの索引データを保持するために物理的に割り振られるスペースの量より小さくなることもあります (たとえば論理表の切り捨ての場合)。戻されるサイズは、索引に対して論理的に割り振られるすべてのエクステントを考慮に入れ、DMS 表スペースで作成される索引の場合は EMP エクステントの見積もりを考慮に入れます。この関数を実行したときに表または索引でエラーが発生した場合、この値は NULL です。

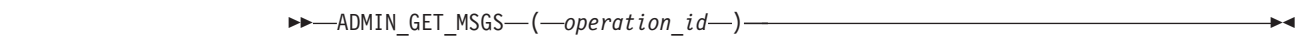
表 71. ADMIN_GET_INDEX_INFO 表関数のメタデータ (続き)

列名	データ・タイプ	説明
INDEX_OBJECT_P_SIZE	BIGINT	索引オブジェクトの物理サイズ。非パーティション表の場合、これは、表に定義されるすべての索引に対して物理的に割り振られるディスク・スペースの量になります。パーティション表の非パーティション索引の場合、これは、索引に対して物理的に割り振られるディスク・スペースの量になります。パーティション表のパーティション索引の場合、これは、データ・パーティションに定義されるすべての索引パーティションに対して物理的に割り振られるディスク・スペースの量になります。すべてのサイズはキロバイト (KB) 単位でレポートされます。 戻されるサイズは、索引に割り振られるすべてのエクステントを考慮に入れ、DMS 表スペースで作成される索引の EMP エクステントが含まれます。この関数を実行したときに表または索引でエラーが発生した場合、この値は NULL です。
INDEX_REQUIRES_REBUILD	CHAR(1)	索引の再作成状況。 • 表またはデータ・パーティションで定義される索引で再編成が必要な場合は Y • それ以外の場合は N この関数を実行したときに表でエラーが発生した場合、この値は NULL です。
LARGE_RIDS	CHAR(1)	索引がラージ行 ID (RID) を使用しているかどうかを示します (4 バイトのページ番号と 2 バイトのスロット番号)。 • 「Y」は、索引がラージ RID を使用していることを示します。 • 「N」は、索引がラージ RID を使用していないことを示します。 • 「P」(保留) は、索引が定義されている表がラージ RID をサポートしている (つまり表が LARGE 表スペースにある) ものの、表またはデータ・パーティションの索引がまだ再編成されていないかまたは再作成されていないことを示します。したがって、表はまだ 4 バイト RID を使用しているので、表または索引をラージ RID に変換するアクションを取る必要があります。 この関数を実行する表に誤りがある場合、この値は NULL になります。

ADMIN_GET_MSGS 表関数 - ADMIN_CMD プロシージャーを通して実行するデータ移動ユーティリティーによって生成されたメッセージの検索

ADMIN_GET_MSGS 表関数は、ADMIN_CMD プロシージャーによるデータ移動ユーティリティー・コマンドの単一の実行によって生成されたメッセージを検索するために使用します。入力パラメーター *operation_id* によってこの操作が識別されます。

構文



スキーマは SYSPROC です。

表関数パラメーター

operation_id
ADMIN_CMD プロシージャーを通して実行したデータ移動ユーティリティーによって作成された、メッセージ・ファイルの操作 ID を指定する、タイプ VARCHAR(139) の入力引数。操作 ID は、ADMIN_CMD プロシージャーによって生成されます。

許可

ADMIN_GET_MSGS 表関数に対する EXECUTE 特権。fenced ユーザー ID には、レジストリー変数 DB2_UTIL_MSGPATH によって示されるディレクトリーの下にあるファイルへの読み取りアクセスが必要です。レジストリー変数が設定されていない場合、fenced ユーザー ID には、instance ディレクトリーの tmp サブディレクトリーにあるファイルへの読み取りアクセスが必要です。

例

ADMIN_CMD プロシージャーを通して実行した EXPORT ユーティリティーによって戻された、操作 ID '24523_THERESAX' のすべてのメッセージを確認します。

```
SELECT * FROM TABLE(SYSPROC.ADMIN_GET_MSGS('24523_THERESAX')) AS MSG
```

以下はこの照会の出力例です。

```
DBPARTITIONNUM AGENTTYPE SQLCODE   MSG
-------------- --------- --------- ----------------------------------------...-
-              -         SQL3104N  The Export utility is beginning to
                                    export data to file
                                    "/home/theresax/rtest/data/ac_load03.del".
-              -         SQL3105N  The Export utility has finished
                                    exporting "8" rows.

2 record(s) selected.
```

使用上の注意

適切な *operation_id* を指定してこの表関数を呼び出す照会ステートメントは、ADMIN_CMD プロシージャーが戻す最初の結果セットの MSG_RETRIEVAL 列に示されます。

戻される情報

表 72. ADMIN_GET_MSGS 表関数によって戻される情報

列名	データ・タイプ	説明
DBPARTITIONNUM	INTEGER	データベース・パーティション番号。この値は分散ロードの場合にのみ戻され、対応するメッセージがどのデータベース・パーティションのものかを示します。
AGENTTYPE	CHAR(4)	エージェント・タイプ。この値は、分散ロードの場合にのみ戻されます。可能な値は次のとおりです。 • 'LOAD': ロード・エージェント • 'PART': パーティション・エージェント • 'PREP': 事前パーティション・エージェント • NULL: 使用可能なエージェント・タイプ情報なし
SQLCODE	VARCHAR(9)	戻されるメッセージの SQLCODE。
MSG	VARCHAR(1024)	SQLCODE に対応する簡略エラー・メッセージ。

ADMIN_IS_INLINED 関数 - データがインラインかどうかを判別する

ADMIN_IS_INLINED 関数は、XML 列、BLOB 列、CLOB 列、または DBCLOB 列のインライン・データに関する状態情報を取得します。

構文

スキーマは SYSIBM です。

戻り値

この関数は、タイプ SMALLINT の以下のいずれかの値、あるいは NULL 値を戻します。

1 データがインライン化されることを示します。

0 データがインライン化されないことを示します。

NULL 入力が NULL であることを示します。

関数のパラメーター

column-name
データ・タイプ XML、BLOB、CLOB、または DBCLOB を持つ基本表の列を識別します (SQLSTATE 42884)。この列は、式に基づいて生成されていない基本表の列を直接的または間接的に参照する必要があります (SQLSTATE 42815)。

例

例 1: 以下の例は、TAB1 表の XML 列 xml_doc1 の 3 つの XML 文書がインライン化されるかどうか示します。

```
db2 => SELECT PK, ADMIN_IS_INLINED(xml_doc1) as IS_INLINED
       from TAB1
```

この照会では、以下の出力が結果として戻ります。

```
PK          IS_INLINED
----------- -------------------
          1                   1
          2                   0
          3                   0

  3 record(s) selected.
```

例 2: 以下の例は、TAB1 表の XML 列 xml_doc1 の XML 文書のいずれかがインライン化されるかどうかを示します。

```
db2 => SELECT PK, ADMIN_IS_INLINED(xml_doc1) as IS_INLINED
       from TAB1 where PK=1
```

この照会では、以下の出力が結果として戻ります。

```
PK          IS_INLINED
----------- -------------------
          1                   1

  1 record(s) selected.
```

例 3: TAB1 表の CLOB 列に含まれている 3 つの CLOB データがインライン化されるかどうかを示します。

```
db2 => SELECT PK, ADMIN_IS_INLINED(clob_1) as IS_INLINED
       from TAB1
```

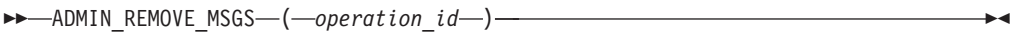
この照会では、以下の出力が結果として戻ります。

```
PK          IS_INLINED
----------- -------------------
          1                   0
          2                   0
          3                   1

  3 record(s) selected.
```

ADMIN_REMOVE_MSGS プロシージャー - ADMIN_CMD プロシージャーを通して実行するデータ移動ユーティリティーによって生成されたメッセージのクリーンアップ

ADMIN_REMOVE_MSGS プロシージャーは、ADMIN_CMD プロシージャーによるデータ移動ユーティリティー・コマンドの単一の実行によって生成されたメッセージをクリーンアップするために使用します。入力パラメーター *operation_id* によって操作が識別されます。

構文



スキーマは SYSPROC です。

プロシージャー・パラメーター

operation_id
ADMIN_CMD プロシージャーを通して実行したデータ移動ユーティリティーによって作成された、メッセージ・ファイルの操作 ID を指定する、タイプ VARCHAR(139) の入力引数。操作 ID は、ADMIN_CMD プロシージャーによって生成されます。

許可

ADMIN_REMOVE_MSGS プロシージャーに対する EXECUTE 特権。fenced ユーザー ID は、レジストリー変数 DB2_UTIL_MSGPATH によって示されるディレクトリーの下にあるファイルを削除できなければなりません。レジストリー変数が設定されていない場合、fenced ユーザー ID には、instance ディレクトリーの `tmp` サブディレクトリーにあるファイルを削除できなければなりません。

例

操作 ID '24523_THERESAX' のメッセージをクリーンアップします。

```
CALL SYSPROC.ADMIN_REMOVE_MSGS('24523_THERESAX')
```

使用上の注意

適切な *operation_id* を指定してこのプロシージャーを呼び出す CALL ステートメントは、ADMIN_CMD プロシージャーが戻す最初の結果セットの MSG_REMOVAL 列に示されます。

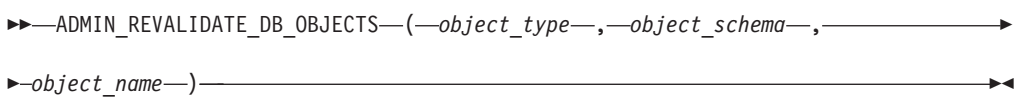
ADMIN_REVALIDATE_DB_OBJECTS プロシージャー - 無効なデータベース・オブジェクトを再検証する

ADMIN_REVALIDATE_DB_OBJECTS プロシージャーは、無効なデータベース・オブジェクトを再検証します。

このプロシージャーでは、3 つの入力パラメーター、*object_type*、*object_schema*、および *object_name* を指定します。これらは、実行される再検証のレベルを制御します。

- データベースの無効なオブジェクトをすべて再検証するには、すべてのパラメーターに NULL を指定するか、パラメーターを指定せずにプロシージャーを呼び出します。
- 特定のスキーマの下の無効なデータベース・オブジェクトをすべて再検証するには、*object_schema* に値を指定し、*object_name* および *object_type* に NULL を指定します。
- 特定の無効なデータベース・オブジェクトを再検証するには、すべてのパラメーターに有効な値を指定します。

構文



スキーマは SYSPROC です。

プロシージャー・パラメーター

object_type

データベース・オブジェクトのタイプを識別する、タイプ VARCHAR (30) の入力引数。以下のタイプが有効です。

- FUNCTION
- GLOBAL VARIABLE
- METHOD
- MODULE
- PROCEDURE
- SPECIFIC
- TABLE
- TRIGGER
- TYPE
- VIEW

この値は大/小文字が区別されません。この値を NULL にすることができます。

これらのタイプのいずれかが指定された場合は、プロシージャーは、MODULE に属するオブジェクトを除いて、そのタイプの無効なオブジェクトをすべて再検証します。モジュール内のオブジェクトを再検証する場合は、特定のモジュールの名前を指定して MODULE タイプを使用すると、そのモジュール内のすべての無効なオブジェクトが再検証されます。

複数のパラメーター・シグニチャーを持つルーチンがあり、それらのうちの 1 つのみを再検証する場合は、再検証するルーチンの名前を指定して SPECIFIC タイプを使用します。

TABLE タイプを使用すると、指定された表が再編成され、その統計が収集されます。プロシージャーは、REORG ペンディング状態にある正規またはマテリアライズ照会表に対して、REORG ユーティリティー、RUNSTATS ユーティリティーの順に呼び出します。プロシージャーは、RUNSTATS のユーザー・プロファイルが存在する場合は、これを使用することを試みます。存在しない場合は、デフォルトの RUNSTATS 操作が呼び出されます。

object_schema
データベース・オブジェクト・リファレンスの修飾に使用されるスキーマ名を識別する、タイプ VARCHAR (128) の入力引数。この名前は大文字小文字が区別されます。この値を NULL にすることができます。

object_name
データベース・オブジェクトを識別する、タイプ VARCHAR(128) の入力引数。 この名前は大文字小文字が区別されます。この値は、型付き表または行関数の値とすることはできません。これは、プロシージャーがこれらのタイプのオブジェクトをサポートしていないからです。そのようなオブジェクトの名前が指定されると、エラーが戻されます。この値を NULL にすることができます。

許可

ADMIN_REVALIDATE_DB_OBJECTS プロシージャーに対する EXECUTE 特権。

例

例 1: 現在のデータベースのすべてを再検証します。

```
CALL SYSPROC.ADMIN_REVALIDATE_DB_OBJECTS(NULL, NULL, NULL)
```

または、パラメーターを指定せずにプロシージャーを呼び出します。

```
CALL SYSPROC.ADMIN_REVALIDATE_DB_OBJECTS()
```

例 2: スキーマ MY_SCHEMA によって修飾されるオブジェクトをすべて再検証します。

```
CALL SYSPROC.ADMIN_REVALIDATE_DB_OBJECTS(NULL, 'MY_SCHEMA', NULL)
```

例 3: データベース内のトリガー・オブジェクトをすべて再検証します。

```
CALL SYSPROC.ADMIN_REVALIDATE_DB_OBJECTS('trigger', NULL, NULL)
```

例 4: 特定のビュー・オブジェクトを再検証します。

```
CALL SYSPROC.ADMIN_REVALIDATE_DB_OBJECTS('view', 'MY_SCHEMA', 'MY_VIEW')
```

例 5: MY_SCHEMA の下のプロシージャーをすべて再検証します。この例では、このスキーマの下に 3 つのプロシージャー (proc1、proc2、および proc3) があります。 proc1 によって使用された参照されたオブジェクトは存在しません。以下の呼び出しによって、proc2 および proc3 を再検証できますが、proc1 は無効のままです。この状態の場合、呼び出しによって警告が戻されます。

```
CALL SYSPROC.ADMIN_REVALIDATE_DB_OBJECTS('procedure', 'MY_SCHEMA', NULL)
```

例 6: 存在しないオブジェクトを再検証します。この例では、エラーが戻されます。

```
CALL SYSPROC.ADMIN_REVALIDATE_DB_OBJECTS('procedure', 'MY_SCHEMA', 'MY_VIEW')
```

例 7: 名前付きパラメーター表記を使用して、MY_SCHEMA の下のプロシージャーをすべて再検証します。

```
CALL SYSPROC.ADMIN_REVALIDATE_DB_OBJECTS(
  object_type=>'PROCEDURE',object_schema=>'MY_SCHEMA')
```

使用上の注意

ADMIN_REVALIDATE_DB_OBJECTS プロシージャーに渡される NULL 以外のパラメーター値は、すべて指定される必要があります。指定されない場合は、プロシージャーは再検証が必要なオブジェクトを識別できません。例えば、ビュー名をトリガー・タイプで指定すると、タイプが一致しないので、プロシージャーはビューを再検証しません。

このプロシージャーは、無効なオブジェクト、および REORG ペンディング状態の正規またはマテリアライズ照会表のみを再検証します。 すべての無効なオブジェクトは SYSCAT.INVALIDOBJECTS で検出できます。どの表が REORG ペンディング状態であるかを判別するには、ADMIN_GET_TAB_INFO 表関数を使用します。

有効なオブジェクトが入力として指定された場合は、プロシージャーはあらゆる操作を実行せず、成功コードを戻します。表の再検証中に障害が発生すると、プロシージャーも失敗します。他のオブジェクトの再検証中に障害が発生すると、プロシージャーは、障害を無視し、他のオブジェクトの再検証を続行します。1 つでも障害が発生すると、プロシージャーは、警告 (SQLSTATE 0168B) を戻します。すべてのオブジェクトの再検証が失敗すると、プロシージャーはエラーを戻します (SQLSTATE 429C4)。表を除くオブジェクトのすべての再検証の失敗について詳しくは、SYSCAT.INVALIDOBJECTS を参照してください。

グローバル変数が再検証されると、現行セッションに対してもインスタンス化されます。

表の再検証の進行をモニターするには、関連する表の REORG 操作の進行をモニターできます。他のすべてのオブジェクトについては、SYSCAT.INVALIDOBJECTS カタログ・ビューを照会します。オブジェクトは、正常に再検証されるとこのビューから削除され、再検証が失敗すると、項目は更新されます。

ADMINTABCOMPRESSINFO 管理ビューおよび ADMIN_GET_TAB_COMPRESS_INFO_V97 表関数 - 圧縮された情報を返す

ADMINTABCOMPRESSINFO 管理ビューおよび ADMIN_GET_TAB_COMPRESS_INFO_V97 表関数は、表、マテリアライズ照会表 (MQT)、階層表の圧縮された情報を戻します。

ADMINTABCOMPRESSINFO 管理ビュー

ADMINTABCOMPRESSINFO 管理ビューは、表、マテリアライズ照会表 (MQT)、および階層表だけの圧縮された情報を戻します。これらの表タイプは SYSCAT.TABLES カタログ・ビューで、T (表)、S (マテリアライズ照会表)、およ

び H (階層表) として報告されます。情報は表のデータ・パーティション・レベルとデータベース・パーティション・レベルの両方のものが戻されます。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、¥ADMINTABCOMPRESSINFO 管理ビューおよび ADMIN_GET_TAB_COMPRESS_INFO_V97 表関数のメタデータ表を参照してください。

許可

以下のいずれかの権限が必要です。

- ADMINTABCOMPRESSINFO 管理ビューに対する SELECT 特権
- ADMINTABCOMPRESSINFO 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- ADMIN_GET_TAB_COMPRESS_INFO_V97 表関数に対する EXECUTE 特権
- DATAACCESS 権限

例

例 1: すべての表に関するすべての圧縮された情報を検索します。

```
SELECT * FROM SYSIBMADM.ADMINTABCOMPRESSINFO
```

以下はこの照会の出力例です。

```
TABSCHEMA TABNAME   DBPARTITIONNUM DATA_PARTITION_ID COMPRESS_ATTR DICT_BUILDER ...
--------- --------- -------------- ----------------- ------------- ------------ ...
SYSIBM    SYSTABLES              0                 0 N             NOT BUILT    ...
SYSIBM    SYSTABLES              0                 0 N             NOT BUILT    ...
...
SIMAP2    STAFF                  0                 4 Y             REORG        ...
SIMAP2    STAFF                  0                 4 Y             REORG        ...
...
  156 record(s) selected.
```

この照会からの出力 (続き):

```
DICT_BUILD_TIMESTAMP       COMPRESS_DICT_SIZE EXPAND_DICT_SIZE ROWS_SAMPLED ...
-------------------------- ------------------ ---------------- ------------ ...
-                                           0                0            0 ...
-                                           0                0            0 ...
...
2009-03-31-11.08.18.000000               3968             3000            6 ...
2009-03-31-11.08.18.000000              13312            10944            6 ...
...
```

この照会からの出力 (続き):

```
PAGES_SAVED_PERCENT BYTES_SAVED_PERCENT AVG_COMPRESS_REC_LENGTH OBJECT_TYPE
------------------- ------------------- ----------------------- -----------
                  0                   0                       0 DATA
                  0                   0                       0 XML
...
                 70                  70                      31 DATA
                 66                  66                     235 XML
...
```

例 2: すべての表に関して、ディクショナリーが作成されたオブジェクト、ディクショナリー作成アクション、およびディクショナリー作成時刻を判別します。

```
SELECT TABSCHEMA, TABNAME, DBPARTITIONNUM, DATA_PARTITION_ID,
    OBJECT_TYPE, DICT_BUILDER, DICT_BUILD_TIMESTAMP
  FROM SYSIBMADM.ADMINTABCOMPRESSINFO
```

以下はこの照会の出力例です。

```
TABSCHEMA       TABNAME         DBPARTITIONNUM DATA_PARTITION_ID ...
--------------- --------------- -------------- ----------------- ...
SYSIBM          SYSTABLES                   0                 0 ...
SYSIBM          SYSTABLES                   0                 0 ...
...
SIMAP2          STAFF                       0                 4 ...
SIMAP2          STAFF                       0                 4 ...
SYSTOOLS        HMON_COLLECTION             0                 0 ...
SYSTOOLS        HMON_COLLECTION             0                 0 ...

  156 record(s) selected.
```

この照会からの出力 (続き):

```
OBJECT_TYPE DICT_BUILDER               DICT_BUILD_TIMESTAMP
----------- -------------------------- --------------------------
DATA        NOT BUILT                  -
XML         NOT BUILT                  -
...
DATA        REORG                      2009-03-31-11.08.18.000000
XML         REORG                      2009-03-31-11.08.18.000000
DATA        REDISTRIBUTE               2009-03-29-06.44.32.000000
XML         REDISTRIBUTE               2009-03-29-06.44.32.000000
```

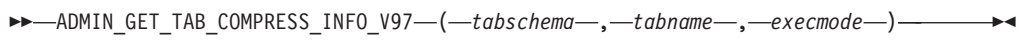
ADMIN_GET_TAB_COMPRESS_INFO_V97 表関数

ADMIN_GET_TAB_COMPRESS_INFO_V97 表関数は ADMINTABCOMPRESSINFO 管理ビューと同じ情報を戻しますが、この関数ではスキーマ、表名、および実行モードを指定することが可能です。

戻される可能性のある情報の完全なリストは、ADMINTABCOMPRESSINFO 管理ビューおよび ADMIN_GET_TAB_COMPRESS_INFO_V97 表関数のメタデータ表を参照してください。

この関数は、各表について 2 つの行を返します。 1 つの行では、OBJECT_TYPE 列に DATA の値があり、別の行では、その列に XML の値があります。 DATA としてマークが付いた行は、推奨されない 1134 ページの『ADMINTABCOMPRESSINFO ビューおよび ADMIN_GET_TAB_COMPRESS_INFO』 表関数からの戻り値に相当します。 XML としてマークが付いた行は、XML コンプレッション・ディクショナリーを記述します。

構文



スキーマは SYSPROC です。

表関数パラメーター

tabschema
スキーマ名を指定する、タイプ VARCHAR(128) の入力引数。

tabname
表、マテリアライズ照会表、または階層表それぞれの名前を指定する、タイプ VARCHAR(128) の入力引数。

execmode
実行モードを指定する、タイプ VARCHAR(30) の入力引数。実行モードは以下のいずれかになります。

- 'REPORT' -- 最後の生成の時点での圧縮情報を報告します。これはデフォルト値です。
- 'ESTIMATE' -- 現在の表に基づいて新規の圧縮情報を生成します。

許可

ADMIN_GET_TAB_COMPRESS_INFO_V97 関数に対する EXECUTE 特権。

例

例 1: 表 SIMAP2.STAFF に関する既存の圧縮情報を検索します。

```
SELECT *
  FROM TABLE (
    SYSPROC.ADMIN_GET_TAB_COMPRESS_INFO_V97('SIMAP2', 'STAFF', 'REPORT'))
  AS T
```

以下はこの照会の出力からの例です。

```
TABSCHEMA TABNAME DBPARTITIONNUM DATA_PARTITION_ID COMPRESS_ATTR DICT_BUILDER ...
--------- ------- -------------- ----------------- ------------- ------------ ...
SIMAP2    STAFF                0                 4 Y             REORG        ...
SIMAP2    STAFF                0                 4 Y             NOT BUILT    ...

2 record(s) selected.
```

この照会からの出力 (続き):

```
DICT_BUILD_TIMESTAMP       COMPRESS_DICT_SIZE EXPAND_DICT_SIZE ROWS_SAMPLED ...
-------------------------- ------------------ ---------------- ------------ ...
2009-03-31-12.19.30.000000              13312             5296           35 ...
-                                           0                0            0 ...
```

この照会からの出力 (続き):

```
PAGES_SAVED_PERCENT BYTES_SAVED_PERCENT AVG_COMPRESS_REC_LENGTH OBJECT_TYPE
------------------- ------------------- ------------------------ -----------
                 38                  38                       27 DATA
                  0                   0                        0 XML
```

例 2: 現時点での表 SIMAP2.STAFF に関する推定圧縮圧縮情報を検索します。

```
SELECT *
  FROM TABLE (
    SYSPROC.ADMIN_GET_TAB_COMPRESS_INFO_V97('SIMAP2', 'STAFF', 'ESTIMATE'))
  AS T
```

以下はこの照会の出力からの例です。

```
TABSCHEMA TABNAME DBPARTITIONNUM DATA_PARTITION_ID COMPRESS_ATTR DICT_BUILDER   ...
--------- ------- -------------- ----------------- ------------- -------------- ...
SIMAP2    STAFF                0                 4 Y             TABLE FUNCTION ...
SIMAP2    STAFF                0                 4 Y             TABLE FUNCTION ...

2 record(s) selected.
```

この照会からの出力 (続き):

```
DICT_BUILD_TIMESTAMP       COMPRESS_DICT_SIZE EXPAND_DICT_SIZE ROWS_SAMPLED ...
-------------------------- ------------------ ---------------- ------------ ...
2009-03-31-12.27.06.000000              13312             5296           35 ...
2009-03-31-12.27.06.000000              13312             9544            8 ...
```

この照会からの出力 (続き):

```
PAGES_SAVED_PERCENT BYTES_SAVED_PERCENT AVG_COMPRESS_REC_LENGTH OBJECT_TYPE
------------------- ------------------- ------------------------ -----------
                 38                  38                       27 DATA
                 75                  75                       95 XML
```

例 3: スキーマ SIMAP2 でのすべての表のオブジェクトに関する合計ディクショナリー・サイズを決定します。

```
SELECT TABSCHEMA, TABNAME, OBJECT_TYPE, DICT_BUILDER, (
     COMPRESS_DICT_SIZE+EXPAND_DICT_SIZE)
     AS TOTAL_DICT_SIZE, DBPARTITIONNUM, DATA_PARTITION_ID
FROM TABLE (
  SYSPROC.ADMIN_GET_TAB_COMPRESS_INFO_V97('SIMAP2', '', 'REPORT'))
AS T
```

この照会からの出力:

```
TABSCHEMA    TABNAME      OBJECT_TYPE  DICT_BUILDER ...
------------ ---------- -------------- ------------ ...
SIMAP2       ACT          DATA         NOT BUILT    ...
SIMAP2       ACT          XML          NOT BUILT    ...
SIMAP2       ADEFUSR      DATA         INSPECT      ...
SIMAP2       ADEFUSR      XML          NOT BUILT    ...
...
SIMAP2       CUSTOMER     DATA         REORG        ...
SIMAP2       CUSTOMER     XML          REORG        ...
SIMAP2       DEPARTMENT   DATA         NOT BUILT    ...
SIMAP2       DEPARTMENT   XML          NOT BUILT    ...
...
SIMAP2       STAFF        DATA         REORG        ...
SIMAP2       STAFF        XML          NOT BUILT    ...
SIMAP2       SUPPLIERS    DATA         TABLE GROWTH ...
SIMAP2       SUPPLIERS    XML          NOT BUILT    ...

44 record(s) selected.
```

この照会からの出力 (続き):

```
TOTAL_DICT_SIZE DBPARTITIONNUM DATA_PARTITION_ID
--------------- -------------- -----------------
              0              0                 0
              0              0                 0
           1890              0                 0
              0              0                 0
...
           6968              0                 1
          24256              0                 1
              0              1                 0
              0              1                 0
...
          18608              0                 4
```

```
            0             0               4
         6960             0               2
            0             0               2
```

例 4: SIMAP2 スキーマの表のディクショナリー情報のレポートを表示します。

```
SELECT * FROM TABLE (
    SYSPROC.ADMIN_GET_TAB_COMPRESS_INFO_V97('SIMAP2', '', 'REPORT'))
  AS T
```

この照会からの出力:

```
TABSCHEMA TABNAME   DBPARTITIONNUM DATA_PARTITION_ID COMPRESS_ATTR DICT_BUILDER ...
--------- --------- -------------- ----------------- ------------- ------------ ...
SIMAP2    ACT                    0                 0 N             NOT BUILT    ...
SIMAP2    ACT                    0                 0 N             NOT BUILT    ...
SIMAP2    ADEFUSR                0                 0 N             INSPECT      ...
SIMAP2    ADEFUSR                0                 0 N             NOT BUILT    ...
...
SIMAP2    CUSTOMER               0                 1 Y             REORG        ...
SIMAP2    CUSTOMER               0                 1 Y             REORG        ...
...
SIMAP2    STAFF                  0                 4 Y             REORG        ...
SIMAP2    STAFF                  0                 4 Y             NOT BUILT    ...
SIMAP2    SUPPLIERS              0                 2 N             NOT BUILT    ...
SIMAP2    SUPPLIERS              0                 2 N             NOT BUILT    ...

44 record(s) selected.
```

この照会からの出力 (続き):

```
DICT_BUILD_TIMESTAMP       COMPRESS_DICT_SIZE EXPAND_DICT_SIZE ROWS_SAMPLED ...
-------------------------- ------------------ ---------------- ------------ ...
-                                           0                0            0 ...
-                                           0                0            0 ...
2009-03-31-12.11.02.000000                290             1890           22 ...
-                                           0                0            0 ...
...
2009-03-31-11.08.18.000000                3968             3000            6 ...
2009-03-31-11.08.18.000000               13312            10944            6 ...
...
2009-03-31-12.19.30.000000               13312             5296           35 ...
-                                           0                0            0 ...
-                                           0                0            0 ...
-                                           0                0            0 ...
```

この照会からの出力 (続き):

```
PAGES_SAVED_PERCENT BYTES_SAVED_PERCENT AVG_COMPRESS_REC_LENGTH OBJECT_TYPE
------------------- ------------------- ------------------------ -----------
                  0                   0                        0 DATA
                  0                   0                        0 XML
                 20                  25                       21 DATA
                  0                   0                        0 XML
...
                 70                  70                       31 DATA
                 66                  66                      235 XML
...
                 38                  38                       27 DATA
                  0                   0                        0 XML
                  0                   0                        0 DATA
                  0                   0                        0 XML
```

例 5: SIMAP2 スキーマの表の DATA オブジェクトのディクショナリー情報のレポートを表示します。

```
  SELECT * FROM TABLE (
      SYSPROC.ADMIN_GET_TAB_COMPRESS_INFO_V97('SIMAP2','','REPORT'))
    WHERE OBJECT_TYPE='DATA'
```

この照会からの出力:

```
TABSCHEMA TABNAME   DBPARTITIONNUM DATA_PARTITION_ID COMPRESS_ATTR DICT_BUILDER ...
--------- --------- -------------- ----------------- ------------- ------------ ...
SIMAP2    ACT                   0                 0 N             NOT BUILT    ...
SIMAP2    ADEFUSR               0                 0 N             INSPECT      ...
...
SIMAP2    CUSTOMER              0                 1 Y             REORG        ...
SIMAP2    DEPARTMENT            1                 0 N             NOT BUILT    ...
...
SIMAP2    STAFF                 0                 4 Y             REORG        ...
SIMAP2    SUPPLIERS             0                 2 N             NOT BUILT    ...

  22 record(s) selected.
```

この照会からの出力 (続き):

```
DICT_BUILD_TIMESTAMP       COMPRESS_DICT_SIZE EXPAND_DICT_SIZE ROWS_SAMPLED...
-------------------------- ------------------ ---------------- ------------ ...
-                                           0                0            0 ...
2009-03-31-12.11.02.000000                290             1890           22 ...
...
2009-03-31-11.08.18.000000               3968             3000            6 ...
-                                           0                0            0 ...
...
2009-03-31-12.19.30.000000              13312             5296           35 ...
-                                           0                0            0 ...
```

この照会からの出力 (続き):

```
PAGES_SAVED_PERCENT BYTES_SAVED_PERCENT AVG_COMPRESS_REC_LENGTH OBJECT_TYPE
------------------- ------------------- ----------------------- -----------
                  0                   0                       0 DATA
                 20                  25                      21 DATA

                 70                  70                      31 DATA
                  0                   0                       0 DATA

                 38                  38                      27 DATA
                  0                   0                       0 DATA
```

例 6: SIMAP2 スキーマの CUSTOMER 表の XML オブジェクトのディクショナリー情報のレポートを表示します。

```
  SELECT * FROM TABLE (
      SYSPROC.ADMIN_GET_TAB_COMPRESS_INFO_V97('SIMAP2', 'CUSTOMER',  'REPORT'))
    WHERE OBJECT_TYPE='XML'
```

この照会からの出力:

```
TABSCHEMA TABNAME   DBPARTITIONNUM DATA_PARTITION_ID COMPRESS_ATTR DICT_BUILDER ...
--------- --------- -------------- ----------------- ------------- ------------ ...
SIMAP2    CUSTOMER              0                 1 Y             REORG        ...
```

この照会からの出力 (続き):

```
DICT_BUILD_TIMESTAMP       COMPRESS_DICT_SIZE EXPAND_DICT_SIZE ROWS_SAMPLED ...
-------------------------- ------------------ ---------------- ------------ ...
2009-03-31-11.08.18.000000              13312            10944            6 ...
```

この照会からの出力 (続き):

```
PAGES_SAVED_PERCENT BYTES_SAVED_PERCENT AVG_COMPRESS_REC_LENGTH OBJECT_TYPE
------------------- ------------------- ----------------------- -----------
                 66                  66                     235 XML
```

使用上の注意

- *tabschema* と *tabname* の両方が指定される場合、その特定の表の情報のみが戻されます。
- *tabschema* が指定され、*tabname* が空 (") または NULL の場合、指定したスキーマのすべての表に関する情報が戻されます。
- *tabschema* が空 (") または NULL で、*tabname* が指定される場合、エラーは戻されます。 特定の表の情報を取り出すには、その表がスキーマと表名の両方によって識別されることが必要です。
- *tabschema* と *tabname* の両方が空 (") または NULL の場合、すべての表の情報が戻されます。
- *tabschema* または *tabname* が存在しないか、あるいは *tabname* が表名 (タイプ T)、マテリアライズ照会表名 (タイプ S)、または階層表名 (タイプ H) と一致しない場合、空の結果セットが戻されます。
- ADMIN_GET_TAB_COMPRESS_INFO_V97 表関数が指定の表のデータを検索するとき、この表関数は SYSTABLES の対応する行に対する共用ロックを獲得します。これは、戻されるデータの整合性を確保するための動作です (たとえば、情報の検索中に、検索されている表が変更されないようにするなど)。ロックが保持されるのは、表関数の呼び出し期間中ではなく、表の圧縮情報を検索する間だけです。
- 照会された表が非 XML 表である場合は、XML ストレージ・オブジェクト (XDA) についての行が戻されます。

ADMINTABCOMPRESSINFO 管理ビューおよび ADMIN_GET_TAB_COMPRESS_INFO_V97 表関数のメタデータ

表 73. ADMINTABCOMPRESSINFO 管理ビューおよび ADMIN_GET_TAB_COMPRESS_INFO_V97 表関数のメタデータ

列名	データ・タイプ	説明
TABSCHEMA	VARCHAR(128)	スキーマ名
TABNAME	VARCHAR(128)	表名
DBPARTITIONNUM	SMALLINT	データベース・パーティション番号
DATA_PARTITION_ID	INTEGER	データ・パーティション番号
COMPRESS_ATTR	CHAR(1)	表の COMPRESS 属性の状態。以下のいずれかになります。 • 「Y」 = 行圧縮は yes に設定されます。 • 「N」 = 行圧縮は no に設定されます。

表 73. ADMINTABCOMPRESSINFO 管理ビューおよび ADMIN_GET_TAB_COMPRESS_INFO_V97 表関数のメタデータ (続き)

列名	データ・タイプ	説明
DICT_BUILDER	VARCHAR(30)	ディクショナリーを作成するために取られるコード・パス。以下のいずれかになります。 • 'INSPECT' = INSPECT ROWCOMPESTIMATE • 'LOAD' = LOAD INSERT/REPLACE • 'NOT BUILT' = 使用可能なディクショナリーがありません • 'REDISTRIBUTE' = REDISTRIBUTE • 'REORG' = REORG RESETDICTIONARY • 'TABLE GROWTH' = INSERT
DICT_BUILD_TIMESTAMP	TIMESTAMP	ディクショナリーが作成された時刻のタイム・スタンプ。タイム・スタンプの細分性は秒単位です。使用可能なディクショナリーがない場合、タイム・スタンプは NULL です。
COMPRESS_DICT_SIZE	BIGINT	バイト単位で測定されたコンプレッション・ディクショナリーのサイズ。
EXPAND_DICT_SIZE	BIGINT	バイト単位で測定されたエクスパンション・ディクショナリーのサイズ。履歴ディクショナリーが存在する場合、この値は現在のディクショナリー・サイズと履歴ディクショナリー・サイズの合計になります。
ROWS_SAMPLED	INTEGER	ディクショナリーの作成に役立つレコードの数。コンプレッション・ディクショナリーを含むマイグレーション済みの表はこの列で NULL を戻します。
PAGES_SAVED_PERCENT	SMALLINT	圧縮により削減されるページのパーセンテージこの情報は、サンプル・バッファー内のレコード・データのみに基づきます。コンプレッション・ディクショナリーを含むマイグレーション済みの表はこの列で NULL を戻します。
BYTES_SAVED_PERCENT	SMALLINT	圧縮により削減されるバイトのパーセンテージこの情報は、サンプル・バッファー内のレコード・データのみに基づきます。コンプレッション・ディクショナリーを含むマイグレーション済みの表はこの列で NULL を戻します。
AVG_COMPRESS_REC_LENGTH	SMALLINT	ディクショナリーの作成に役立っているレコードの平均圧縮レコード長。コンプレッション・ディクショナリーを含むマイグレーション済みの表はこの列で NULL を戻します。
OBJECT_TYPE	VARCHAR(4)	オブジェクトのタイプ。この行には、指定されたオブジェクトに関する、タイプに応じた値が入ります。値は以下のいずれかになります。 • 'XML' • 'DATA'

ADMINTABINFO 管理ビューおよび ADMIN_GET_TAB_INFO_V97 表関数 - 表のサイズおよび状態に関する情報の検索

ADMINTABINFO 管理ビューおよびADMIN_GET_TAB_INFO_V97 表関数は、現在カタログ・ビューで使用できない表のサイズおよび状態に関する情報を検索するメソッドを提供します。

ADMINTABINFO 管理ビュー

ADMINTABINFO 管理ビューは、表、マテリアライズ照会表 (MQT)、および階層表だけのサイズおよび状態に関する情報を戻します。これらの表タイプは SYSCAT.TABLES カタログ・ビューで、T (表)、S (マテリアライズ照会表)、および H (階層表) として報告されます。情報は表のデータ・パーティション・レベルとデータベース・パーティション・レベルの両方のものが戻されます。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、ADMINTABINFO 管理ビューおよび ADMIN_GET_TAB_INFO_V97 表関数のメタデータ表を参照してください。

許可

以下のいずれかの権限が必要です。

- ADMINTABINFO 管理ビューに対する SELECT 特権
- ADMINTABINFO 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- ADMIN_GET_TAB_INFO_V97 表関数に対する EXECUTE 特権
- DATAACCESS 権限

例

例 1: すべての表のサイズおよび状態に関する情報を検索します。

```
SELECT * FROM SYSIBMADM.ADMINTABINFO
```

例 2: データ密度の低い多数の表が使用する物理スペースの量を判別します。

```
SELECT TABSCHEMA, TABNAME, SUM(DATA_OBJECT_P_SIZE),
   SUM(INDEX_OBJECT_P_SIZE), SUM(LONG_OBJECT_P_SIZE),
   SUM(LOB_OBJECT_P_SIZE), SUM(XML_OBJECT_P_SIZE)
   FROM SYSIBMADM.ADMINTABINFO GROUP BY TABSCHEMA, TABNAME
```

例 3: ラージ RID の使用に適してはいるものの、現在使用可能になっていない表を識別します。

```
SELECT TABSCHEMA, TABNAME FROM SYSIBMADM.ADMINTABINFO
   WHERE LARGE_RIDS = 'P'
```

例 4: どの表がタイプ 1 索引を使用しており、タイプ 2 索引への変換のための再編成が必要かを識別します。

```
SELECT TABSCHEMA, TABNAME FROM SYSIBMADM.ADMINTABINFO
   WHERE INDEX_TYPE = 1
```

例 5: タイプ 1 形式の XML データがあり、Online Table Move でタイプ 2 形式に変換する必要がある表を識別します。

```
SELECT TABSCHEMA, TABNAME FROM SYSIBMADM.ADMINTABINFO
   WHERE XML_RECORD_TYPE=1
```

例 4: 表 T1 について収集された統計情報の現在のタイプを調べます。

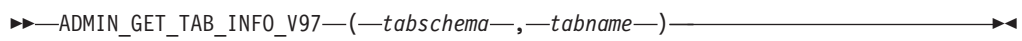
```
SELECT SUBSTR(TABSCHEMA, 1, 10) AS TBSCHEMA, SUBSTR(TABNAME, 1, 10)
   AS TBNAME, STATSTYPE FROM SYSIBMADM.ADMINTABINFO WHERE TABNAME = 'T1';

TBSCHEMA   TBNAME     STATSTYPE
---------- ---------- ---------
DB2USER1   T1         U

  1 record(s) selected.
```

ADMIN_GET_TAB_INFO_V97 表関数

ADMIN_GET_TAB_INFO_V97 表関数は ADMINTABINFO 管理ビューと同じ情報を戻しますが、この関数ではスキーマおよび表名を指定することが可能です。

戻される可能性のある情報の完全なリストは、ADMINTABINFO 管理ビューおよび ADMIN_GET_TAB_INFO_V97 表関数のメタデータ表を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

tabschema
スキーマ名を指定する、タイプ VARCHAR(128) の入力引数。

tabname
表、マテリアライズ照会表、または階層表それぞれの名前を指定する、タイプ VARCHAR(128) の入力引数。

許可

ADMIN_GET_TAB_INFO_V97 表関数に対する EXECUTE 特権。

例

例 1: 表 DBUSER1.EMPLOYEE のサイズおよび状態に関する情報を検索します。

```
SELECT * FROM TABLE (SYSPROC.ADMIN_GET_TAB_INFO_V97('DBUSER1', 'EMPLOYEE'))
   AS T
```

例 2: 非パーティション表 (DBUSER1.EMPLOYEE) が存在し、関連オブジェクト (たとえば索引や LOB など) がすべて 1 つの表スペースに保管されていると仮定します。 表が表スペース内のどのくらいの物理スペースを使用しているかを計算します。

```
SELECT (data_object_p_size + index_object_p_size + long_object_p_size +
   lob_object_p_size + xml_object_p_size) as total_p_size
   FROM TABLE( SYSPROC.ADMIN_GET_TAB_INFO_V97( 'DBUSER1', 'EMPLOYEE' )) AS T
```

表が別の表スペースに移動されるときにどのくらいのスペースが必要になるかを計算します。ここでは、新規の表スペースには元の表スペースと同じページ・サイズおよびエクステント・サイズがあるとします。

```
SELECT (data_object_l_size + index_object_l_size + long_object_l_size +
   lob_object_l_size + xml_object_l_size) as total_l_size
   FROM TABLE( SYSPROC.ADMIN_GET_TAB_INFO_V97( 'DBUSER1', 'EMPLOYEE' )) AS T
```

例 3: 表 DBUSER1.EMPLOYEE 用のコンプレッション・ディクショナリーの合計サイズを判別します。

```
SELECT SUBSTR(TABSCHEMA,1,10) AS TBSCHEMA, SUBSTR(TABNAME,1,10) AS TBNAME,
   DICTIONARY_SIZE + XML_DICTIONARY_SIZE AS TOTAL_DICTIONARY_SIZE
   FROM TABLE(SYSPROC.ADMIN_GET_TAB_INFO_V97('DBUSER1','EMPLOYEE'))
```

例 4: マルチディメンション・クラスタリング表 SAMPLE.STAFF から再利用可能なスペースの量を判別します。

```
SELECT RECLAIMABLE_SPACE
   FROM TABLE(SYSPROC.ADMIN_GET_TAB_INFO_V97('SAMPLE','STAFF'))
```

使用上の注意

- *tabschema* と *tabname* の両方が指定される場合、その特定の表の情報のみが戻されます。
- *tabschema* が指定され、*tabname* が NULL または空ストリング (") の場合、指定したスキーマのすべての表に関する情報が戻されます。
- *tabschema* が NULL または空ストリング (") で、*tabname* が指定される場合、エラーは戻されます。特定の表の情報を取り出すには、その表がスキーマと表名の両方によって識別されることが必要です。
- *tabschema* と *tabname* の両方が NULL または空ストリング (") の場合、すべての表の情報が戻されます。
- *tabschema* または *tabname* が存在しないか、あるいは *tabname* が表名 (タイプ T)、マテリアライズ照会表名 (タイプ S)、または階層表名 (タイプ H) と一致しない場合、空の結果セットが戻されます。
- ADMIN_GET_TAB_INFO_V97 表関数が指定の表のデータを検索するとき、この表関数は SYSTABLES の対応する行に対する共用ロックを獲得します。これは、戻されるデータの整合性を確保するための動作です (たとえば、情報の検索中に、検索されている表がドロップされないようにするなど)。ロックが保持されるのは、表関数の呼び出し期間中ではなく、表のサイズおよび状態に関する情報を検索する間だけです。
- SMS 表スペースの表の物理サイズが報告されますが、このサイズは論理サイズと同じです。
- 表で INPLACE の REORG がアクティブになっていると、データ・オブジェクトの物理サイズ (DATA_OBJECT_P_SIZE) は計算されません。論理サイズだけが戻されます。 INPLACE の REORG が表でアクティブになっているかどうかは、INPLACE_REORG_STATUS 出力の列を見ると分かります。
- DB2 UDB バージョン 8 より前に作成された LOB オブジェクトの論理サイズが報告されますが、オブジェクトの再編成が行われていないと、この論理サイズは物理サイズよりも大きくなっている場合があります。

REDISTRIBUTING_PENDING

1. 指定された表について再配分は実行されていません N
2. 再配分の実行がデータベース・パーティション・グループで開始されましたが、表では開始されませんでした N
3. データを移動する前のフェーズで再配分が失敗しました N
4. データの移動のフェーズで再配分が失敗しました Y
5. 表について再配分が正常に完了し、コミットされました N

ADMINTABINFO 管理ビューおよび ADMIN_GET_TAB_INFO_V97 表関数のメタデータ

表 74. ADMINTABINFO 管理ビューおよび ADMIN_GET_TAB_INFO_V97 表関数のメタデータ

列名	データ・タイプ	説明
TABSCHEMA	VARCHAR(128)	スキーマ名。
TABNAME	VARCHAR(128)	表名。
TABTYPE	CHAR(1)	表タイプ: • 「H」= 階層表 • 「S」= マテリアライズ照会表 • 「T」= 表
DBPARTITIONNUM	SMALLINT	データベース・パーティション番号。
DATA_PARTITION_ID	INTEGER	データ・パーティション番号。
AVAILABLE	CHAR(1)	表の状態: • 「N」= 表は使用不可。表を使用できない場合、サイズおよび状態に関係する他の出力列はすべて NULL になります。 • 「Y」= 表は使用可能。 **注:** リカバリー不能ロードでのロールフォワードを行うと、表の状態が使用不可になります。
DATA_OBJECT_L_SIZE	BIGINT	データ・オブジェクトの論理サイズ。表に対して論理的に割り振られるディスク・スペースの量 (KB 単位で報告)。論理サイズとは、表が認識するスペースの量のことです。このサイズは表に対して物理的に割り振られるスペースの量より小さくなることもあります (たとえば論理表の切り捨ての場合)。多次元クラスタリング (MDC) 表の場合、このサイズにはブロック・マップ・オブジェクトの論理サイズが含まれます。戻されるサイズは、表に対して論理的に割り振られるすべてのエクステントを考慮に入れ、DMS 表スペースで作成されるオブジェクトの場合は Extent Map Page (EMP) エクステントの見積もりを考慮に入れます。このサイズは、基本表のみの論理サイズを表します。 LOB データ、長形式データ、索引、および XML オブジェクトが消費するスペースは別の列で報告されます。

表 74. ADMINTABINFO 管理ビューおよび ADMIN_GET_TAB_INFO_V97 表関数のメタデータ (続き)

列名	データ・タイプ	説明
DATA_OBJECT_P_SIZE	BIGINT	データ・オブジェクトの物理サイズ。表に対して物理的に割り振られるディスク・スペースの量 (KB 単位で報告)。MDC 表の場合、このサイズにはブロック・マップ・オブジェクトのサイズが含まれます。戻されるサイズは、表に割り振られるすべてのエクステントを考慮に入れ、DMS 表スペースで作成されるオブジェクトの EMP エクステントが含まれます。このサイズは、基本表のみの物理サイズを表します。 LOB データ、長形式データ、索引、および XML オブジェクトが消費するスペースは別の列で報告されます。
INDEX_OBJECT_L_SIZE	BIGINT	索引オブジェクトの論理サイズ。表で定義される索引に対して論理的に割り振られるディスク・スペースの量 (KB 単位で報告)。論理サイズとは、表が認識するスペースの量のことです。このサイズは表の索引データを保持するために物理的に割り振られるスペースの量より小さくなることもあります (たとえば論理表の切り捨ての場合)。戻されるサイズは、索引に対して論理的に割り振られるすべてのエクステントを考慮に入れ、DMS 表スペースで作成される索引の場合は EMP エクステントの見積もりを考慮に入れます。 パーティション表のパーティション索引の場合、これは DATA_PARTITION_ID によって識別されるデータ・パーティションの索引パーティションを含む索引オブジェクトの論理サイズです。この値には、パーティション表の非パーティション索引は考慮に入れられません。パーティション索引と非パーティション索引の両方の情報については、ADMIN_GET_INDEX_INFO 関数を使用できます。
INDEX_OBJECT_P_SIZE	BIGINT	索引オブジェクトの物理サイズ。表で定義される索引に対して物理的に割り振られるディスク・スペースの量 (KB 単位で報告)。戻されるサイズは、索引に割り振られるすべてのエクステントを考慮に入れ、DMS 表スペースで作成される索引の EMP エクステントが含まれます。 パーティション表のパーティション索引の場合、これは DATA_PARTITION_ID によって識別されるデータ・パーティションの索引パーティションを含む索引オブジェクトの物理サイズです。この値には、パーティション表の非パーティション索引は考慮に入れられません。パーティション索引と非パーティション索引の両方の情報については、ADMIN_GET_INDEX_INFO 関数を使用できます。

表 74. *ADMINTABINFO 管理ビューおよび ADMIN_GET_TAB_INFO_V97 表関数のメタデータ (続き)*

列名	データ・タイプ	説明
LONG_OBJECT_L_SIZE	BIGINT	長形式オブジェクトの論理サイズ。表の長形式フィールド・データに対して論理的に割り振られるディスク・スペースの量 (KB 単位で報告)。論理サイズとは、表が認識するスペースの量のことです。このサイズは表の長形式フィールド・データを保持するために物理的に割り振られるスペースの量より小さくなることもあります (たとえば論理表の切り捨ての場合)。戻されるサイズは、長形式フィールド・データに対して論理的に割り振られるすべてのエクステントを考慮に入れ、DMS 表スペースで作成される長形式フィールド・データの場合は EMP エクステントの見積もりを考慮に入れます。
LONG_OBJECT_P_SIZE	BIGINT	長形式オブジェクトの物理サイズ。表の長形式フィールド・データに対して物理的に割り振られるディスク・スペースの量 (KB 単位で報告)。戻されるサイズは、長形式フィールド・データに割り振られるすべてのエクステントを考慮に入れ、DMS 表スペースで作成される長形式フィールド・データの EMP エクステントが含まれます。
LOB_OBJECT_L_SIZE	BIGINT	LOB オブジェクトの論理サイズ。表の LOB データに対して論理的に割り振られるディスク・スペースの量 (KB 単位で報告)。論理サイズとは、表が認識するスペースの量のことです。このサイズは表の LOB データを保持するために物理的に割り振られるスペースの量より小さくなることもあります (たとえば論理表の切り捨ての場合)。サイズには LOB 割り振りオブジェクトに対して論理的に割り振られるスペースが含まれます。戻されるサイズは、LOB データに対して論理的に割り振られるすべてのエクステントを考慮に入れ、DMS 表スペースで作成される LOB データの場合は EMP エクステントの見積もりを考慮に入れます。
LOB_OBJECT_P_SIZE	BIGINT	LOB オブジェクトの物理サイズ。表の LOB データに対して物理的に割り振られるディスク・スペースの量 (KB 単位で報告)。サイズには LOB 割り振りオブジェクトに対して割り振られるスペースが含まれます。戻されるサイズは、LOB データに割り振られるすべてのエクステントを考慮に入れ、DMS 表スペースで作成される LOB データの EMP エクステントが含まれます。
XML_OBJECT_L_SIZE	BIGINT	XML オブジェクトの論理サイズ。表の XML データに対して論理的に割り振られるディスク・スペースの量 (KB 単位で報告)。論理サイズとは、表が認識するスペースの量のことです。このサイズは表の XML データを保持するために物理的に割り振られるスペースの量より小さくなることもあります (たとえば論理表の切り捨ての場合)。戻されるサイズは、XML データに対して論理的に割り振られるすべてのエクステントを考慮に入れ、DMS 表スペースで作成される XML データの場合は EMP エクステントの見積もりを考慮に入れます。

表 74. ADMINTABINFO 管理ビューおよび ADMIN_GET_TAB_INFO_V97 表関数のメタデータ (続き)

列名	データ・タイプ	説明
XML_OBJECT_P_SIZE	BIGINT	XML オブジェクトの物理サイズ。表の XML データに対して物理的に割り振られるディスク・スペースの量 (KB 単位で報告)。戻されるサイズは、XML データに割り振られるすべてのエクステントを考慮に入れ、DMS 表スペースで作成される XML データの EMP エクステントが含まれます。
INDEX_TYPE	SMALLINT	現在表で使用されている索引のタイプを示します。これは、以下のものを戻します。 • タイプ 1 索引が使用されている場合は 1。 • タイプ 2 索引が使用されている場合は 2。
REORG_PENDING	CHAR(1)	「Y」は、REORG 推奨の変更がすでに表に適用されており、クラシック (オフライン) REORG が必要であることを示しています。それ以外の場合は「N」が戻されます。
INPLACE_REORG_STATUS	VARCHAR(10)	表に対する表のインプレース再編成の現在の状況。状況値は以下のいずれかになります。 • ABORTED (PAUSED 状態にあるが、RESUME は不可。STOP は必須) • EXECUTING • NULL (表に対して INPLACE の REORG が実行されていない場合) • PAUSED
LOAD_STATUS	VARCHAR(12)	表に対するロード操作の現在の状況。状況値は以下のいずれかになります。 • IN_PROGRESS • NULL (表でロードが進行しておらず、表がロード・ペンディング状態になっていない場合) • PENDING
READ_ACCESS_ONLY	CHAR(1)	表が「読み取りアクセス専用」の状態になっていれば「Y」、そうでなければ「N」になります。「N」の値を、表が完全にアクセス可能であるという意味に解釈するべきではありません。ロードが進行中またはペンディング状態の場合、「Y」の値は表データが読み取りアクセス可能であることを意味し、「N」の値は表がアクセス不能であることを意味します。同様に、表の状況が SET INTEGRITY ペンディングである場合 (SYSCAT.TABLES STATUS 列を参照)、「N」の値は表がアクセス不能であることを意味します。
NO_LOAD_RESTART	CHAR(1)	「Y」の値は、表が部分的にロードされている状態になっていることを示します。この場合、ロードを再始動することができません。この状態になっていなければ「N」の値が戻されます。
NUM_REORG_REC_ALTERS	SMALLINT	最後に再編成が行われてからこの表に対して実行された REORG 推奨の変更操作 (たとえば直後に再編成を必要とする変更操作) の回数。

表 74. *ADMINTABINFO 管理ビューおよび ADMIN_GET_TAB_INFO_V97 表関数のメタデータ (続き)*

列名	データ・タイプ	説明
INDEXES_REQUIRE_REBUILD	CHAR(1)	非パーティション表において、表に定義されている索引のいずれかが再ビルドを必要とする場合は「Y」、必要としない場合は「N」。パーティション表において、DATA_PARTITION_ID によって識別されるデータ・パーティションの索引パーティションのいずれかが再ビルドを必要とする場合は「Y」、必要としない場合は「N」。
LARGE_RIDS	CHAR(1)	表がラージ行 ID (RID) を使用しているかどうかを示します (4 バイトのページ番号と 2 バイトのスロット番号)。「Y」の値は表がラージ RID を使用していることを示し、「N」は使用していないことを示します。表がラージ RID をサポートしている (つまり表が LARGE 表スペースにある) ものの、少なくとも表の索引の 1 つがまだ再編成されていないかまたは再ビルドされていない場合、「P」(保留) の値が戻されます。これは表が 4 バイトの RID を使用しているためです (これは表または索引を変換するためのアクションを取る必要があることを意味します)。
LARGE_SLOTS	CHAR(1)	表がラージ・スロット (これは 1 ページにつき 255 を超える行が可能です) を使用しているかどうかを示します。「Y」の値は表がラージ・スロットを使用していることを示し、「N」は使用していないことを示します。表がラージ・スロットをサポートしている (つまり表が LARGE 表スペースにある) ものの、表に対してまだオフラインの表の再編成または表の切り捨て操作が実行されていない場合、「P」(保留) の値が戻されます。これは、表が 1 ページにつき最大 255 行のラージ・スロットを使用しているためです。
DICTIONARY_SIZE	BIGINT	表ディクショナリーのサイズ (バイト)。表に行コンプレッション・ディクショナリーが存在する場合に行の圧縮で使用されます。履歴ディクショナリーが存在する場合、この値は現在のディクショナリー・サイズと履歴ディクショナリー・サイズの合計になります。
BLOCKS_PENDING_CLEANUP	BIGINT	MDC 表の場合、クリーンアップを保留にしているブロックの数。 MDC 以外の表の場合、この値は常にゼロです。

表 74. *ADMINTABINFO 管理ビューおよび ADMIN_GET_TAB_INFO_V97 表関数のメタデータ (続き)*

列名	データ・タイプ	説明
STATSTYPE	CHAR(1)	• 「F」= 表または索引のスキャンを行わずにシステムが作り上げた統計。これらの統計はメモリーに保管され、システム・カタログに保管されているものとは異なります。これは一時状態であり、最終的に完全な統計が DB2 によって収集され、システム・カタログに保管されます。 • 「A」= システムが非同期に収集した統計。統計は DB2 によりバックグラウンド・プロセスで自動的に収集され、システム・カタログに保管されました。 • 「S」= システムが同期に収集した統計。統計は DB2 によって SQL ステートメントのコンパイル中に自動的に収集されました。これらの統計はメモリーに保管され、システム・カタログに保管されているものとは異なります。これは一時状態であり、最終的に DB2 が統計をシステム・カタログに保管します。 • 「U」= ユーザーが収集した統計。統計の収集は、RUNSTATS、CREATE INDEX、LOAD、REDISTRIBUTE などのユーティリティーを使用するか、システム・カタログ統計を手動で更新することによって、ユーザーが開始しました。 • NULL = 不明なタイプ
XML_RECORD_TYPE	SMALLINT	現在表で使用されている XML レコードのタイプを示します。 • タイプ 1 (シングル・ノード) XML レコード・フォーマットが使用されている場合には 1。 • タイプ 2 (マルチノード) XML レコード・フォーマットが使用されている場合には 2。 • 表に XML 列がない場合には NULL。
RECLAIMABLE_SPACE	BIGINT	DMS 表スペースの MDC 表の場合、この値は、RECLAIM オプションを指定して REORG コマンドを実行することで再利用可能なディスク・スペース量を示します。ディスク・スペースは、K バイト (KB) で報告されます。その他の表の場合、この値はゼロです。
XML_DICTIONARY_SIZE	BIGINT	XML ストレージ・オブジェクトにデータ・コンプレッション・ディクショナリーが存在する場合、データ圧縮に使用される XML ディクショナリーのサイズ (バイト単位)。表に XML 列がない場合、またはコンプレッション・ディクショナリーが作成されていない場合、値は 0 になります。

ADMINTEMPCOLUMNS 管理ビューおよび ADMIN_GET_TEMP_TABLES 表関数 - 一時表の列情報を取得する

ADMINTEMPCOLUMNS 管理ビューおよび ADMIN_GET_TEMP_COLUMNS 表関数は、作成済み一時表および宣言済み一時表の列属性に関する情報を取得する方法を提供します。

カタログ・ビューには作成済み一時表のインスタンスの列属性情報が含まれますが、宣言済み一時表の表属性情報は含まれません。

ADMINTEMPCOLUMNS 管理ビュー

ADMINTEMPCOLUMNS 管理ビューは、作成済み一時表および宣言済み一時表のインスタンスの列属性に関する情報を戻します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、ADMINTEMPCOLUMNS 管理ビューおよび ADMIN_GET_TEMP_COLUMNS 表関数のメタデータ表を参照してください。

許可

以下のいずれかの権限が必要です。

- ADMINTEMPCOLUMNS 管理ビューに対する SELECT 特権
- ADMINTEMPCOLUMNS 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- ADMIN_GET_TEMP_COLUMNS 表関数に対する EXECUTE 特権
- DATAACCESS 権限

例

例 1: データベースに現在存在する作成済み一時表およびすべての宣言済み一時表のすべてのインスタンスの列属性に関する情報を戻します。

```
SELECT * FROM SYSIBMADM.ADMINTEMPCOLUMNS
```

例 2: データベースでアクティブになっているどの一時表がユーザー定義のデータ・タイプ USMONEY を使用しているかを判別します。

```
SELECT APPLICATION_HANDLE, TABSCHEMA, TABNAME
   FROM SYSIBMADM.ADMINTEMPCOLUMNS
   WHERE TYPENAME = 'USMONEY'
```

例 3: SYSTEM_USER で宣言されたすべての宣言済み一時表の表スキーマ、表名、および列名を取得します。

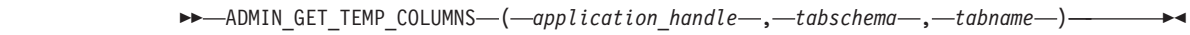
```
SELECT T.TABSCHEMA, T.TABNAME, C.COLNAME
   FROM SYSIBMADM.ADMINTEMPCOLUMNS C, SYSIBMADM.ADMINTEMPTABLES T
   WHERE T.TEMPTABTYPE = 'D'
   AND T.INSTANTIATOR = SYSTEM_USER
   AND T.TABSCHEMA = C.TABSCHEMA
   AND T.TABNAME = C.TABNAME
```

ADMIN_GET_TEMP_COLUMNS 表関数

ADMIN_GET_TEMP_TABLES 表関数は ADMINTEMPCOLUMNS 管理ビューと同じ情報を戻しますが、この関数ではスキーマ名および表名を指定することが可能です。

戻される可能性のある情報の完全なリストは、ADMINTEMPCOLUMNS 管理ビューおよび ADMIN_GET_TEMP_COLUMNS 表関数のメタデータ表を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

application_handle
アプリケーション・ハンドルを指定する、タイプ BIGINT の入力引数。*application_handle* が指定されている場合、指定された接続に関するデータのみが戻されます。*application_handle* が NULL である場合、すべての接続に関するデータが戻されます。

tabschema
スキーマ名を指定する、タイプ VARCHAR(128) の入力引数。

tabname
作成済み一時表の名前または宣言済み一時表の名前を指定する、タイプ VARCHAR(128) の入力引数。

許可

ADMIN_GET_TEMP_COLUMNS 表関数に対する EXECUTE 特権。

例

例 1: 現行接続の宣言済み一時表 TEMPEMPLOYEE の列情報を取得します。

```
SELECT *
   FROM TABLE (
      SYSPROC.ADMIN_GET_TEMP_COLUMNS(
         APPLICATION_ID(), 'SESSION', 'TEMPEMPLOYEE'))
   AS T
```

使用上の注意

- *tabschema* と *tabname* の両方が指定される場合、その特定の一時表の情報のみが戻されます。
- *tabschema* が指定され、*tabname* が NULL または空ストリング (") の場合、指定したスキーマのすべての表に関する情報が戻されます。
- *tabschema* が NULL または空ストリング (") で、*tabname* が指定される場合、エラーは戻されます。特定の一時表の情報を取り出すには、その表がスキーマと表名の両方によって識別されることが必要です。

- *tabschema* と *tabname* の両方が NULL または空ストリング ('') の場合、*application_handle* の値に応じて、特定の接続 (またはすべての接続) のすべての一時表に関する情報が戻されます。
- *tabschema* または *tabname* が存在しないか、*tabname* が一時表名と一致しないか、あるいは識別された一時表のインスタンスがデータベースに存在しない場合、空の結果セットが戻されます。

ADMINTEMPCOLUMNS 管理ビューおよび ADMIN_GET_TEMP_COLUMNS 表関数のメタデータ

表 75. ADMINTEMPCOLUMNS 管理ビューおよび ADMIN_GET_TEMP_COLUMNS 表関数のメタデータ

列名	データ・タイプ	説明
APPLICATION_HANDLE	BIGINT	システム全体におけるアプリケーションのユニーク ID。単一パーティション・データベースの場合、この ID は 16 ビットのカウンターで構成されます。複数パーティション・データベースの場合、この ID はコーディネーター・パーティション番号と 16 ビットのカウンターを連結したもので構成されます。さらに、この ID はアプリケーションが 2 次接続を行うすべてのパーティションにおいて同じです。
APPLICATION_NAME	VARCHAR(256)	アプリケーションの名前。
TABSCHEMA	VARCHAR(128)	この列のある一時表のスキーマ名。
TABNAME	VARCHAR(128)	この列のある一時表の表名。
COLNAME	VARCHAR(128)	列の名前。
COLNO	SMALLINT	表の中のこの列の番号 (0 から始まる)。
TYPESCHEMA	VARCHAR(128)	列のデータ・タイプのスキーマ名。
TYPENAME	VARCHAR(128)	列のデータ・タイプの非修飾名。
LENGTH	INTEGER	データの最大長。 特殊タイプの場合は 0。 LENGTH 列は、DECIMAL フィールドの精度を示し、10 進浮動小数点列に必要なストレージのバイト数を示します。DECFLOAT(16) と DECFLOAT(34) ではそれぞれ 8 と 16 です。
SCALE	SMALLINT	列タイプが DECIMAL の場合は位取りで、列タイプが TIMESTAMP の場合には秒の小数部分の桁数。その他の場合には 0。
DEFAULT	VARCHAR(254)	列のデータ・タイプに適した定数、特殊レジスター、または cast 関数で表された表の列のデフォルト値。キーワード NULL の場合もあります。値は、デフォルト値として指定された値から変換されることがあります。例えば、日時の定数は ISO フォーマットで表示され、cast 関数名はスキーマ名で修飾され、ID は区切り文字で区切られます。 DEFAULT 節が指定されていないか、または列がビューの列の場合は、NULL 値になります。

表 75. ADMINTEMPCOLUMNS 管理ビューおよび ADMIN_GET_TEMP_COLUMNS 表関数のメタデータ (続き)

列名	データ・タイプ	説明
NULLS	CHAR(1)	列の NULL 可能性属性。 • Y = 列は NULL 可能。 • N = 列は NULL 不可。 式または関数から派生したビューの列の場合、値は N である可能性があります。それでも、このビューを使用するステートメントが処理され、算術計算エラーの警告を出された場合は、この列に NULL 値が入ります。
CODEPAGE	SMALLINT	この列のデータに使用されるコード・ページ。列が FOR BIT DATA として定義されている場合、またはストリング・タイプではない場合は 0。
LOGGED	CHAR(1)	LOB タイプまたは LOB に基づく特殊タイプの列だけに適用されます。それ以外はブランクです。 • Y = 列のログ記録を取る。 • N = 列のログ記録を取らない。
COMPACT	CHAR(1)	LOB タイプまたは LOB に基づく特殊タイプの列だけに適用されます。それ以外はブランクです。 • Y = ストレージ内で列を圧縮する。 • N = 列を圧縮しない。
INLINE_LENGTH	INTEGER	基本表に保管できる XML 文書または構造化タイプのインスタンスの内部表記の最大サイズ (バイト数)。適用されない場合には 0。
IDENTITY	CHAR(1)	• Y = ID 列。 • N = ID 列ではない。
GENERATED	CHAR(1)	生成される列の型。 • A = 列の値が常に生成される • D = 列の値がデフォルトで生成される • ブランク = 列は生成されない

ADMINTEMPTABLES 管理ビューおよび ADMIN_GET_TEMP_TABLES 表関数 - 一時表の情報を取得する

ADMINTEMPTABLES 管理ビューおよび ADMIN_GET_TEMP_TABLES 表関数は、作成済み一時表および宣言済み一時表のインスタンスの表属性およびインスタンス生成時間に関する情報を検索する方法を提供します。

カタログ・ビューには作成済み一時表の表属性情報が含まれますが、宣言済み一時表の表属性情報は含まれません。さらに、カタログ・ビューには、作成済み一時表または宣言済み一時表の表インスタンス生成時間に関する情報は含まれません。

ADMINTEMPTABLES 管理ビュー

ADMINTEMPTABLES 管理ビューは、作成済み一時表および宣言済み一時表のインスタンスの表属性およびインスタンス生成時間に関する情報を戻します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、ADMINTEMPTABLES 管理ビューおよび ADMIN_GET_TEMP_TABLES 表関数のメタデータ表を参照してください。

許可

以下のいずれかの権限が必要です。

- ADMINTEMPTABLES 管理ビューに対する SELECT 特権
- ADMINTEMPTABLES 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- ADMIN_GET_TEMP_TABLES 表関数に対する EXECUTE 特権
- DATAACCESS 権限

例

例 1: データベースに現在存在する作成済み一時表および宣言済み一時表のすべてのインスタンスの表属性およびインスタンス生成時間に関する情報を取得します。

```
SELECT * FROM SYSIBMADM.ADMINTEMPTABLES
```

例 2: どの接続に作成済み一時表のインスタンスが含まれているかを判別します。

```
SELECT APPLICATION_HANDLE, TABSCHEMA, TABNAME
   FROM SYSIBMADM.ADMINTEMPTABLES
   WHERE TEMPTABTYPE = 'C'
```

例 3: データベースに接続したユーザーによってインスタンス生成されたすべての表に宣言された、すべての宣言済み一時表の表属性およびインスタンス生成時間に関する情報を検索します。

```
SELECT TABSCHEMA, TABNAME, ONCOMMIT, ONROLLBACK,
      INSTANTIATION_TIME
   FROM SYSIBMADM.ADMINTEMPTABLES
   WHERE TEMPTABTYPE = 'D' AND INSTANTIATOR = SYSTEM_USER
```

ADMIN_GET_TEMP_TABLES 表関数

ADMIN_GET_TEMP_TABLES 表関数は ADMINTABINFO 管理ビューと同じ情報を戻しますが、この関数ではスキーマ名および表名を指定することが可能です。

戻される可能性のある情報の完全なリストは、ADMINTABINFO 管理ビューおよび ADMIN_GET_TEMP_TABLES 表関数のメタデータ表を参照してください。

構文

スキーマは SYSPROC です。

表関数パラメーター

application_handle
アプリケーション・ハンドルを指定する、タイプ BIGINT の入力引数。*application_handle* が指定されている場合、指定された接続に関するデータのみが戻されます。*application_handle* が NULL である場合、すべての接続に関するデータが戻されます。

tabschema
スキーマ名を指定する、タイプ VARCHAR(128) の入力引数。

tabname
作成済み一時表の名前または宣言済み一時表の名前を指定する、タイプ VARCHAR(128) の入力引数。

許可

ADMIN_GET_TEMP_TABLES 表関数に対する EXECUTE 特権。

例

例 1: すべての接続の作成済み一時表 DBUSER1.EMPLOYEE のすべてのインスタンスの表属性およびインスタンス生成時間に関する情報を取得します。

```
SELECT TABSCHEMA, TABNAME, ONCOMMIT, ONROLLBACK, INSTANTIATION_TIME
   FROM TABLE (SYSPROC.ADMIN_GET_TEMP_TABLES(NULL, 'DBUSER1', 'EMPLOYEE'))
   AS T
```

例 2: 現行接続でのユーザー一時表のすべてのインスタンスのインスタンス生成時間および表スペース ID を取得します。

```
SELECT TABSCHEMA, TABNAME, INSTANTIATION_TIME, TBSP_ID
   FROM TABLE (SYSPROC.ADMIN_GET_TEMP_TABLES(APPLICATION_ID(), '', ''))
   AS T
```

使用上の注意

-
- *tabschema* と *tabname* の両方が指定される場合、その特定の一時表の情報のみが戻されます。
- *tabschema* が指定され、*tabname* が NULL または空ストリング (") の場合、指定したスキーマのすべての表に関する情報が戻されます。
- *tabschema* が NULL または空ストリング (") で、*tabname* が指定される場合、エラーは戻されます。特定の一時表の情報を取り出すには、その表がスキーマと表名の両方によって識別されることが必要です。
- *tabschema* と *tabname* の両方が NULL または空ストリング (") の場合、*application_handle* の値に応じて、特定の接続 (またはすべての接続) のすべての一時表に関する情報が戻されます。
- *tabschema* または *tabname* が存在しないか、*tabname* が一時表名と一致しないか、あるいは識別された一時表のインスタンスがデータベースに存在しない場合、空の結果セットが戻されます。

ADMINTEMPTABLES 管理ビューおよび ADMIN_GET_TEMP_TABLES 表関数のメタデータ

表 76. ADMINTEMPTABLES 管理ビューおよび ADMIN_GET_TEMP_TABLES 表関数のメタデータ

列名	データ・タイプ	説明
APPLICATION_HANDLE	BIGINT	システム全体におけるアプリケーションのユニーク ID。単一パーティション・データベースの場合、この ID は 16 ビットのカウンターで構成されます。複数パーティション・データベースの場合、この ID はコーディネーター・パーティション番号と 16 ビットのカウンターを連結したもので構成されます。さらに、この ID はアプリケーションが 2 次接続を行うすべてのパーティションにおいて同じです。
APPLICATION_NAME	VARCHAR(256)	アプリケーションの名前。
TABSCHEMA	VARCHAR(128)	スキーマ名。
TABNAME	VARCHAR(128)	表名。
INSTANTIATOR	VARCHAR(128)	作成済み一時表がインスタンス生成されたか、または宣言済み一時表が宣言された許可 ID。
INSTANTIATORTYPE	CHAR(1)	• U = インスタンス生成者は個々のユーザー
TEMPTABTYPE	CHAR(1)	一時表タイプ: • C = 作成済み一時表 • D = 宣言済み一時表
INSTANTIATION_TIME	TIMESTAMP	作成済み一時表がインスタンス生成されたか、または宣言済み一時表が宣言された時刻。
COLCOUNT	SMALLINT	列の数。継承された列がある場合は、それも含む。
TAB_FILE_ID	BIGINT	table_file_id - 表のファイル ID (FID)。
TBSP_ID	BIGINT	tablespace_id - 現行データベースで使用する表スペースを表す固有の整数。
PMAP_ID	SMALLINT	この表によって現在使用されている分散マップの ID。
PARTITION_MODE	CHAR(1)	パーティション・データベース・システム内のデータベース・パーティション間でどのようにデータが配分されるかを示します。 • H = ハッシュ • ブランク = データベース・パーティションなし
CODEPAGE	SMALLINT	オブジェクトのコード・ページ。これが、すべての文字の列および式で生成された列のデフォルトのコード・ページです。
ONCOMMIT	CHAR(1)	COMMIT 操作の実行時にこの表で行うアクションを指定します。 • D = 行の削除 • P = 行の保存
ONROLLBACK	CHAR(1)	ROLLBACK 操作の実行時にこの表で行うアクションを指定します。 • D = 行の削除 • P = 行の保存

表 76. ADMINTEMPTABLES 管理ビューおよび ADMIN_GET_TEMP_TABLES 表関数のメタデータ (続き)

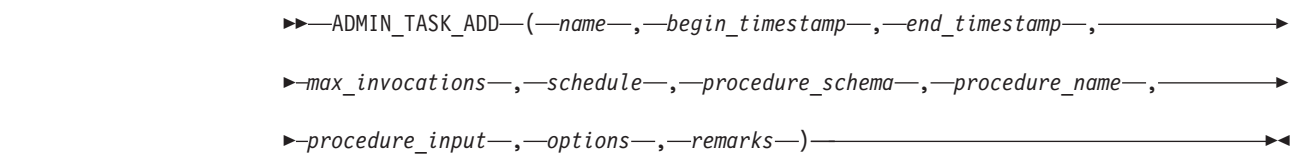
列名	データ・タイプ	説明
LOGGED	CHAR(1)	この表をログ対象にするかどうかを指定します。 • N = ログ対象にしません • Y = ログ対象です

第 5 章 管理タスク・スケジューラー・ルーチンおよびビュー

ADMIN_TASK_ADD プロシージャー - 新規タスクのスケジュール

ADMIN_TASK_ADD プロシージャーは、管理タスク、つまりプロシージャーの内部にカプセル化できる処理をスケジュールに入れます。

構文



スキーマは SYSPROC です。

プロシージャー・パラメーター

name
タスクの名前を指定する、タイプ VARCHAR (128) の入力引数。この引数は NULL にできません。

begin_timestamp
タスクの実行を開始できる最も早い時刻を指定する、タイプ TIMESTAMP の入力引数。この引数の値は、過去の日付にしたり、*end_timestamp* より後の日付にしたりすることはできません。

タスク実行が開始されるタイミングは、この引数と *schedule* 引数が定義されている方法によって異なります。

- *begin_timestamp*引数が NULL 以外の場合:
  - *schedule* 引数が NULL の場合、タスク実行は *begin_timestamp* に開始されます。
  - *schedule* 引数が NULL 以外の場合、タスク実行は、スケジュールされている次の時刻または *begin_timestamp* よりも後に開始されます。
- *begin_timestamp* 引数が NULL の場合:
  - *schedule* 引数が NULL の場合、タスク実行は即時に開始されます。
  - *schedule* 引数が NULL 以外の場合、タスク実行は、スケジュールされている次の時刻に開始されます。

end_timestamp
タスクの実行を開始できる最後の時刻を指定する、タイプ TIMESTAMP の入力引数。この引数の値は、過去の日付にしたり、*begin_timestamp* より前の日付にしたりすることはできません。引数が NULL の場合、タスクは、スケジュールされたとおりに無期限に実行されます。

実行中のタスクは、その *end_timestamp* で中断されることはありません。

© Copyright IBM Corp. 2006, 2010

max_invocations
タスクで許容される最大実行数を指定する、タイプ INTEGER の入力引数。引数が NULL の場合、タスクを実行できる回数に制限はありません。引数が 0 の場合、タスクは実行されません。

schedule が NULL 以外の場合、この値がスケジュールに適用されます。

end_timestamp と *max_invocations* の両方が指定されている場合、*end_timestamp* が優先されます。つまり、*end_timestamp* タイム・スタンプに達すると、タスクの回数がその時点で *max_invocations* の値に達していなくても、そのタスクが再び実行されることはありません。

schedule
特定の時間にタスクが実行されるようにスケジュールを指定する、タイプ VARCHAR(1024) の入力引数。引数が NULL の場合、タスクは、特定の時間にスケジュールされません。

schedule ストリングは、UNIX cron 形式を使用して指定する必要があります。

複数のスケジュールはサポートされていません。

procedure_schema
このタスクが実行するプロシージャーのスキーマを指定する、タイプ VARCHAR(128) の入力引数。この引数は NULL にできません。

procedure_name
このタスクが実行するプロシージャーの名前を指定する、タイプ VARCHAR(128) の入力引数。この引数は NULL にできません。

procedure_input
このタスクが実行するプロシージャーの入力引数を指定する、タイプ CLOB(2M) の入力引数。この引数には、1 行のデータを戻す SQL ステートメントが含まれている必要があります。戻り値は、引数としてプロシージャーに渡されます。この引数が NULL の場合、引数はプロシージャーに渡されません。

SQL ステートメントによって戻される列数は、プロシージャーの引数の総数 (およびタイプ) と一致していなければならず、単一行を含んでいる必要があります。出力引数の場合、値自体は無視されますが、プロシージャーが要求するのと同じ SQL データ・タイプでなければなりません。

この SQL ステートメントは、タスクが実行されるたびに実行されます。 SQL ステートメントが失敗すると、タスクの状況は NOTRUN に設定され、特定の SQLCODE 情報が記録されます。ステートメントが結果セットや行を戻さず、複数の行または結果セットを戻す場合、タスクは実行されません。タスクの状況は NOTRUN に設定され、この引数が無効であることを示す SQLCODE SQL1465N が設定されます。

ステートメントの結果に直列化 XML パラメーターが含まれる場合、結合されるすべての XML パラメーターの合計サイズは 256 キロバイトに制限されます。結果がこのしきい値を超えた場合、タスクの状況は NOTRUN に設定されます。データ切り捨てが発生したことを示す、SQLCODE -302 および SQLSTATE 22001 が設定されます。

タスクの状況を表示するには、SYSTOOL.ADMIN_TASK_STATUS ビューを使用します。

options
タイプ VARCHAR(512) の入力引数。この引数は NULL でなければなりません。

remarks
タスクの説明を指定する、タイプ VARCHAR(254) の入力引数。この引数はオプションであり、NULL にすることができます。

許可

ADMIN_TASK_ADD プロシージャーに対する EXECUTE 特権。データベースが **RESTRICTIVE** オプションで作成された場合を除き、デフォルトで EXECUTE 特権が PUBLIC に付与されます。

使用上の注意

ADMIN_TASK_ADD プロシージャーを呼び出す前に、SYSTOOLSPACE 表スペースが存在している必要があります。存在しない場合、プロシージャーは SQL0204N エラー・メッセージを戻します。

タスクがスケジュールされている場合は、現行セッション・ユーザーの許可 ID が記録されます。スケジューラーは、タスクを実行するときにこのセッション許可 ID に切り替えます。

管理タスク・スケジューラーは、指定したユーザー ID およびパスワードを使用せずにデータベース接続を実行するプロシージャーの実行をサポートしていません。例えば、ADMIN_CMD プロシージャーを使用して、データベースから LOAD を実行できます。ソース・データベースへの接続は、現在接続中のデータベースに提供されているユーザー ID およびパスワードを使用して確立されます。このタイプの LOAD 操作は、タスク・スケジューラーでは実行できません。

無効な引数がプロシージャーに渡されると、SQL0171N が戻されます。メッセージのトークンは、無効な引数およびプロシージャーの名前を示します。

タスクは、作業単位がコミットされて、スケジューラーがタスク定義をフェッチするまでは、実行をスケジュールすることはできません。

スケジューラーは、新規および更新されたタスクがないかどうか 5 分おきに調べます。予期されるとおりにタスクが実行されるようにするには、*begin_timestamp*、*end_timestamp*、および *schedule* 引数で定義される最も早い開始時刻は、作業単位がコミットされてから少なくとも 5 分後でなければなりません。

タスクがスケジューラーによって実行できるようにするには、データベースはすべてのデータベース・パーティションでアクティブである必要があります。

パーティション・データベース環境では、ADMIN_TASK_ADD プロシージャーは任意のデータベース・パーティションから呼び出すことができます。ただし、スケジューラーはカタログ・データベース・パーティションからすべてのタスクを実行します。

begin_timestamp、*end_timestamp*、および *schedule* はサーバーの時間帯に基づいています。夏時間調整 (DST) の移行期間にタスクをスケジュールする場合には、特に注

意が必要です。タスクが午前 2 時 1 分に実行されるようスケジュールされている場合に、それが時間を早めるときに当たると、時間は午前 2 時から午前 3 時までスキップされるため、そのタスクは実行されません。一方、時間が 1 時間戻されるときには、午前 2 時から 3 時の間にスケジュールされているタスクは 2 回実行されることになります。希望する動作が確実に実行されるようにするには、夏時間調整に関する設定をユーザーの責任で行う必要があります。

スケジューラーは常に、*procedure_schema* および *procedure_name* で指定されたプロシージャーを呼び出した後にコミットされます。トランザクションのロールバックが必要な場合、ロールバックはプロシージャーの内部で行われる必要があります。

タスク名が固有のものでない場合、プロシージャーは SQL0601N で失敗します。

例

例 1: 2008 年 2 月 4 日から毎日午前 12 時にオンライン TSM バックアップを実行するタスクの作成:

```
CALL SYSPROC.ADMIN_TASK_ADD
 ( 'DAILY TSM BACKUP',
     '2007-02-04-00.00.00.000000',
      NULL,
       NULL,
       '0 0 * * *',
       'SYSPROC',
       'ADMIN_CMD',
       'VALUES(''BACKUP DATABASE SALES ONLINE USE TSM WITHOUT PROMPTING'')',
       NULL,
       NULL )
```

例 2: 1 時間ごとにイベント・モニターをフラッシュするタスクをスケジュールする:

1. "em" というイベント・モニターをフラッシュする SQL プロシージャーを PROD スキーマに作成します。

   ```
   CREATE PROCEDURE FLUSH_EVENT_MONITOR()
    SPECIFIC FLUSH_EVENT_MONITOR
      LANGUAGE SQL
      BEGIN
      DECLARE stmt VARCHAR(100) ;
      SET stmt = 'FLUSH EVENT MONITOR em' ;
      EXECUTE IMMEDIATE stmt ;
      END
   ```

   注: FLUSH EVENT MONITOR SQL ステートメントをプロシージャー内で直接呼び出すことはできません。ただし、EXECUTE IMMEDIATE を使用できます。

2. ADMIN_TASK_ADD を呼び出してタスクをスケジュールする:

   ```
   CALL SYSPROC.ADMIN_TASK_ADD
    ('FLUSH EVENT MONITOR EVERY HOUR',
      NULL,
       NULL,
      NULL,
      '0 0-23 * * *',
      'PROD',
      'FLUSH_EVENT_MONITOR',
      NULL,
      NULL,
      NULL )
   ```

UNIX cron 形式

UNIX cron 形式は、ADMIN_TASK_ADD および ADMIN_TASK_UPDATE プロシージャーの *schedule* パラメーターで時刻を指定するために使用されます。

cron 形式には、少なくとも 1 つのブランクで区切られた 5 つの日時フィールドがあります。フィールド値をブランクにすることはできません。スケジュールされたタスクは、分、*時間*、および *月* フィールドが現在の日時と一致し、2 つの日付フィールド (*日*、あるいは *曜日*) のうち少なくとも 1 つが現在日付と一致する場合に実行されます。

表 1 では、日時フィールドとその許容値が cron 形式でリストされています。

表 77. UNIX cron 形式のフィールド名および値

フィールド名	許容値
分	0-59
時	0-23
日	1-31
月	• 1-12。1 は 1 月、2 は 2 月、以下 3 月、4 月と続きます。 • 英語の月名に基づいた大文字、小文字、および大/小文字混合の 3 文字のストリング。例: jan、 feb、 mar、 apr、 may、 jun 、 jul、 aug、 sep、 oct、 nov、または dec。
曜日	• 0-7。0 または 7 は日曜日、1 は月曜日、以降同様です。 • 英語の曜日名に基づいた大文字、小文字、または大/小文字混合の 3 文字のストリング: mon、tue、wed、thu、fri、sat、または sun。

範囲およびリスト

数値の範囲を使用できます。範囲は、ハイフンで区切られた 2 つの数値で指定されます。指定した範囲は包括的です。例えば、時間項目の範囲 8-11 は、8、9、10 および 11 時に実行されることを示します。

リストを使用できます。リストは、コンマで区切られた数値または範囲のセットです。以下に例を示します。

```
1,2,5,9
0-4,8-12
```

無制限の範囲

フィールドには、アスタリスク (*) を含めることができます。これは、フィールド内のすべての可能な値を表します。

コマンドを実行する日は、*日*および*曜日*の 2 つのフィールドで指定できます。両方のフィールドがアスタリスク以外の値の使用で制限されている場合、コマンドはいずれかのフィールドが現在時刻と一致したときに実行されます。例えば、値 `30 4 1,15 * 5` の場合、コマンドは各月の 1 日、15 日、および毎週金曜日の午前 4 時 30 分に実行されます。

ステップ値

ステップ値は、範囲とともに使用することができます。構文 *range*`/`*step* は、範囲および実行間隔を定義します。

first`-`*last*`/`*step* を指定した場合、実行は *first* で行われ、次に *first* から *step* ずつ離れた連続するすべての値で (*last* まで) 行われます。

例えば、2 時間おきにコマンド実行を指定するには、`0-23/2` を使用します。この式は、値 `0,2,4,6,8,10,12,14,16,18,20,22` と同等です。

`*/`*step* を指定した場合、実行は *step* の間隔ごとに無制限の範囲で行われます。例えば、2 時間おきに実行する場合には、`0-23/2` の代わりに `*/2` を使用します。

例

表 2 には、さまざまなスケジュール・シナリオで ADMIN_TASK_ADD または ADMIN_TASK_UPDATE プロシージャーの *schedule* 引数に使用できる値がリストされています。

表 78. タスク・スケジュールおよび該当する schedule 引数値の例

希望するタスク・スケジュール	*schedule* **値**
毎月曜日の午後 2 時 10 分	`10 14 * * 1`
毎日午前 0 時	`0 0 * * *`
毎平日の午前 0 時	`0 0 * * 1-5`
月の 1 日および 15 日の午前 0 時	`0 0 1,15 * *`
各年の 11 月 17 日、21 日および 29 日と、11 月の毎月曜日および毎水曜日の午後 6 時 32 分	`32 18 17,21,29 11 mon,wed`

ADMIN_TASK_LIST 管理ビュー - スケジューラーのタスクに関する情報の取得

ADMIN_TASK_LIST 管理ビューは、管理タスク・スケジューラーで定義されている各タスクに関する情報を取得します。

スキーマは SYSTOOLS です。

このビューは、最初に ADMIN_TASK_ADD プロシージャーが呼び出されるときに作成されます。

許可

ADMIN_TASK_LIST 管理ビューに対する SELECT または CONTROL 特権。データベースが **RESTRICTIVE** オプションで作成された場合を除き、デフォルトで SELECT 特権が PUBLIC に付与されます。

ADMIN_TASK_LIST ビューを照会した場合、セッション許可 ID を使用して作成されたタスクのみが戻されます。 SYSADM、SYSCTRL、SYSMAINT、または DBADM の各権限を持っている場合、すべてのタスクが戻されます。

例

スケジューラー内のタスクのリストの要求:

```
SELECT * from SYSTOOLS.ADMIN_TASK_LIST
```

戻される情報

表 79. ADMIN_TASK_LIST 管理ビューによって戻される情報

列名	データ・タイプ	説明
NAME	VARCHAR(128)	タスクの名前。
TASKID	INTEGER	タスク ID。
OWNER	VARCHAR(128)	タスクを作成したユーザーのセッション許可 ID。
OWNERTYPE	VARCHAR(1)	許可 ID タイプ。有効な値は以下のとおりです。 • U - ユーザー
BEGIN_TIME	TIMESTAMP	タスクが最初に実行可能になるときのタイム・スタンプ。[1]
END_TIME	TIMESTAMP	タスクが最後に実行可能になるときのタイム・スタンプ。[1] この列が NULL である場合、MAX_INVOCATIONS が指定されていない限り、タスクは無限に実行できます。
MAX_INVOCATIONS	INTEGER	タスクで許容される最大実行数。この列が NULL である場合、END_TIME が指定されていない限り、タスクは無限に実行できます。
SCHEDULE	VARCHAR(1024)	タスクのスケジュール (UNIX cron 形式)
PROCEDURE_SCHEMA	VARCHAR(128)	このタスクが実行するプロシージャーのスキーマ。
PROCEDURE_NAME	VARCHAR(128)	このタスクが実行するプロシージャーの名前。
PROCEDURE_INPUT	CLOB(2M)	このタスクが実行するプロシージャーの入力パラメーター。この列が NULL である場合、入力パラメーターは存在しません。

表 79. *ADMIN_TASK_LIST* 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明
OPTIONS	VARCHAR(512)	タスクの動作に影響するオプション。
UPDATE_TIME	TIMESTAMP	タスクが最後に更新されたときのタイム・スタンプ。
REMARKS	VARCHAR(254)	タスクの説明。

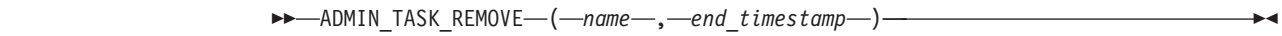
注:

- [1] BEGIN_TIME および END_TIME はデータベース・サーバーの時間帯に基づいています。夏時間調整時間 (DST) の調整はユーザーの責任で行う必要があります。

ADMIN_TASK_REMOVE プロシージャー - スケジュールされたタスクまたはタスク状況レコードの除去

ADMIN_TASK_REMOVE プロシージャーは、スケジュールされた管理タスク、つまりプロシージャーの内部にカプセル化できる作業を除去します。また、タスク状況レコードも除去します。

構文



スキーマは SYSPROC です。

プロシージャー・パラメーター

name
タスクの名前を指定する、タイプ VARCHAR (128) の入力引数。

end_timestamp
状況レコードの *end_timestamp* タイム・スタンプを指定するタイプ TIMESTAMP の出力引数。

許可

ADMIN_TASK_REMOVE プロシージャーに対する EXECUTE 特権。データベースが **RESTRICTIVE** オプションで作成された場合を除き、デフォルトで EXECUTE 特権が PUBLIC に付与されます。

ステートメント許可 ID を使用してプロシージャーを実行することが可能な場合でも、タスクおよび状況レコードが正常に除去されるかどうかは現行のセッション許可 ID の値に依存します。現行のセッション許可 ID は、タスクの作成時に記録されたセッション許可 ID と一致する必要があります。タスクまたは状況レコードを除去できるのは、SYSADM、SYSCTRL、SYSMAINT、または DBADM の各権限を持ったユーザーです。許可を持たないユーザーがタスクまたは状況レコードを除去しようとすると、SQL0551N が戻されます。

使用上の注意

作業単位がコミットされるまでタスクは除去されません。

タスク除去の動作は、*name* および *end_timestamp* 引数が定義される方法によって異なります。

- *end_timestamp* 引数が NULL の場合:
  - *name* 引数が NULL の場合、すべてのタスクおよび状況レコードが除去されます。 1 つ以上のタスクが現在実行中の場合、タスクおよび関連した状況レコードは除去されません。この場合、SQL1464W が戻されます。
  - *name* 引数が NULL 以外の場合、*name* と一致するタスク・レコードは除去されます。指定されたタスクが現在実行中の場合、タスクは除去されず、SQL20453N が戻されます。指定されたタスクが除去されると、関連したすべての状況レコードが除去されます。
- *end_timestamp* 引数が NULL 以外の場合:
  - *name* 引数が NULL の場合、*end_timestamp* 以前の *end_timestamp* タイム・スタンプを持つすべてのタスクおよび状況レコードが除去されます。タスク・レコードは除去されません。プロシージャーは、状況値 RUNNING を持つ状況レコードを除去しません。
  - *name* 引数が NULL 以外の場合、タスクの *end_timestamp* タイム・スタンプが *end_timestamp* 以前であれば、*name* と一致するタスクの状況レコードが除去されます。タスク・レコードは除去されません。プロシージャーは、状況値 RUNNING を持つ状況レコードを除去しません。

ユーザーが存在しないタスクを除去しようとすると、SQL0204N が戻されます。

例

「DAILY TSM BACKUP」というバックアップ・タスクを除去する場合:

```
CALL SYSPROC.ADMIN_TASK_REMOVE('DAILY TSM BACKUP', NULL)
```

ADMIN_TASK_STATUS 管理ビュー - タスク状況情報の取得

ADMIN_TASK_STATUS 管理ビューは、管理タスク・スケジューラーでのタスク実行の状況に関する情報を取得します。

スキーマは SYSTOOLS です。

このビューは、最初に ADMIN_TASK_ADD プロシージャーが呼び出されるときに作成されます。

許可

ADMIN_TASK_STATUS 管理ビューに対する SELECT または CONTROL 特権。データベースが **RESTRICTIVE** オプションで作成された場合を除き、デフォルトで SELECT 特権が PUBLIC に付与されます。

ADMIN_TASK_STATUS ビューを照会した場合、セッション許可 ID によって作成されたタスク状況レコードのみが戻されます。

例

例 1: スケジューラー内のタスクの状況の要求:

```
SELECT * from SYSTOOLS.ADMIN_TASK_STATUS
```

例 2: SQLERRM 関数を使用した SQLERRMC 列のデータのフォーマット:

```
SELECT TASKID, STATUS, SQLCODE, SQLSTATE, RC,
   VARCHAR( SQLERRM( 'SQL' || CHAR( ABS(SQLCODE) ),
   SQLERRMC, x'FF', 'en_US', 1 ), 256) AS MSG_TXT
   FROM SYSTOOLS.ADMIN_TASK_STATUS
```

戻される情報

表 80. ADMIN_TASK_STATUS 管理ビューによって戻される情報

列名	データ・タイプ	説明
NAME	VARCHAR(128)	タスクの名前。
TASKID	INTEGER	タスク ID。
STATUS	VARCHAR(10)	タスクの状況。有効な値は以下のとおりです。 • RUNNING - タスクは現在実行中です。 • COMPLETED - タスクは実行を完了しました。 • NOTRUN - エラーが原因で、スケジューラーはタスクのプロシージャーを呼び出すことができません。 • UNKNOWN - タスクの実行が開始されましたが、予期しない状態が原因で、スケジューラーはタスクの結果を記録できませんでした。これは、タスクの実行中にシステムが異常終了したり、電源障害が発生したりした場合に生じる可能性があります。
INVOCATION	INTEGER	現在の呼び出しカウント。
BEGIN_TIME	TIMESTAMP	タスクが開始された時刻。[1] STATUS が RUNNING、COMPLETED、または UNKNOWN である場合、この値はタスクの実行が開始された時刻を示します。 STATUS が NOTRUN である場合、これはタスクが開始されるはずだった時刻を示します。
END_TIME	TIMESTAMP	タスクの実行が完了した時刻。[1] STATUS が RUNNING である場合、この値は NULL になります。 STATUS が UNKNOWN である場合、この値は、タスク・スケジューラーがタスクはもはや実行されていないと判断し、状況表を更新した時刻になります。
AGENT_ID	BIGINT	タスクが実行するアプリケーションのエージェント ID。エージェント ID はアプリケーション・ハンドルと同義語です。この値は、タスクが実行中である場合にのみ有効です。

表 80. ADMIN_TASK_STATUS 管理ビューによって戻される情報 (続き)

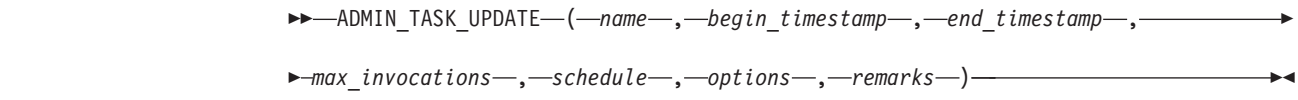
列名	データ・タイプ	説明
SQLCODE	INTEGER	STATUS が COMPLETED である場合、この値は、プロシージャーの呼び出しによって戻される SQLCODE を示します。 STATUS が NOTRUN である場合、この値は、タスクを実行できない原因となった SQLCODE のエラーを示します。 状況が RUNNING または UNKNOWN である場合、この値は NULL になります。
SQLSTATE	CHAR(5)	STATUS が COMPLETED である場合、この値は、プロシージャーの呼び出しによって戻される SQLSTATE を示します。 STATUS が NOTRUN である場合、この値は、タスクを実行できない原因となった SQLSTATE のエラーを示します。 状況が RUNNING または UNKNOWN である場合、この値は NULL になります。
SQLERRMC	VARCHAR(70) FOR BIT DATA	SQLCA の SQLERRMC フィールドに現れるとおり、X'FF' で区切られた 1 つ以上のトークンが入ります。これらのトークンは、エラー条件の説明の中の変数を置き換えます。 STATUS が COMPLETED である場合、この値は、プロシージャーの呼び出しによって戻される SQLERRMC を示します。 STATUS が NOTRUN である場合、この値は、タスクを実行できない原因となった SQLERRMC のエラーを示します。 状況が RUNNING または UNKNOWN である場合、この値は NULL になります。
RC	INTEGER	STATUS が COMPLETED である場合、プロシージャーに戻りコードが含まれていれば、これにはプロシージャーの呼び出しからの戻りコードが含まれます。それ以外の場合、これは NULL になります。

- [1] BEGIN_TIME および END_TIME はデータベース・サーバーの時間帯に基づいています。夏時間調整時間 (DST) の調整はユーザーの責任で行う必要があります。

ADMIN_TASK_UPDATE プロシージャー - 既存のタスクの更新

ADMIN_TASK_UPDATE プロシージャーは、管理タスク、つまりプロシージャーの内部にカプセル化できる処理を更新します。

構文



スキーマは SYSPROC です。

プロシージャー・パラメーター

name
既存のタスクの名前を指定する、タイプ VARCHAR (128) の入力引数。この引数は NULL にできません。

begin_timestamp
タスクの実行を開始できる最も早い時刻を指定する、タイプ TIMESTAMP の入力引数。この引数の値は、過去の日付にしたり、*end_timestamp* より後の日付にしたりすることはできません。

タスク実行が開始されるタイミングは、このパラメーターと *schedule* パラメーターが定義されている方法によって異なります。

- *begin_timestamp*引数が NULL 以外の場合:
  - *schedule* 引数が NULL の場合、タスク実行は *begin_timestamp* に開始されます。
  - *schedule* 引数が NULL 以外の場合、タスク実行は、スケジュールされている次の時刻または *begin_timestamp* よりも後に開始されます。
- *begin_timestamp* 引数が NULL の場合:
  - *schedule* 引数が NULL の場合、タスク実行は即時に開始されます。
  - *schedule* 引数が NULL 以外の場合、タスク実行は、スケジュールされている次の時刻に開始されます。

end_timestamp
タスクの実行を開始できる最後の時刻を指定する、タイプ TIMESTAMP の入力引数。この引数の値は、過去の日付にしたり、*begin_timestamp* より前の日付にしたりすることはできません。引数が NULL の場合、タスクは、スケジュールされたとおりに無期限に実行されます。

実行中のタスクは、その *end_timestamp* で中断されることはありません。

max_invocations
タスクで許容される最大実行数を指定する、タイプ INTEGER の入力引数。引数が NULL の場合、タスクを実行できる回数に制限はありません。引数が 0 の場合、タスクは実行されません。

schedule が NULL 以外の場合、この値がスケジュールに適用されます。

end_timestamp と *max_invocations* の両方が指定されている場合、*end_timestamp* が優先されます。つまり、*end_timestamp* タイム・スタンプに達すると、タスクの回数がその時点で *max_invocations* の値に達していなくても、そのタスクが再び実行されることはありません。

schedule
特定の時間にタスクが実行されるようにスケジュールを指定する、タイプ VARCHAR(1024) の入力引数。引数が NULL の場合、タスクは、特定の時間にスケジュールされません。

schedule ストリングは、UNIX cron 形式を使用して指定する必要があります。

複数のスケジュールはサポートされていません。

options
タイプ VARCHAR(512) の入力引数。この引数は NULL でなければなりません。

remarks
タスクの説明を指定する、タイプ VARCHAR(254) の入力引数。これは、NULL に設定できるオプションの引数です。

許可

ADMIN_TASK_UPDATE プロシージャーに対する EXECUTE 特権。データベースが **RESTRICTIVE** オプションで作成された場合を除き、デフォルトで EXECUTE 特権が PUBLIC に付与されます。

ステートメント許可 ID を使用してプロシージャーを実行することが可能な場合でも、現行のセッション許可 ID がタスクの作成時に記録されたセッション許可 ID と一致しない限り、タスクを更新することはできません。既存のタスクを更新できるのは、SYSADM、SYSCTRL、SYSMAINT、または DBADM の各権限を持ったユーザーです。別のユーザーによって追加されたタスクを更新しようとすると、SQL0551N が戻されます。

使用上の注意

無効な引数がプロシージャーに渡されると、SQL0171N が戻されます。メッセージのトークンは、無効な引数およびプロシージャーの名前を示します。

タスクに対する変更は、作業単位がコミットされて、スケジューラーが更新されたタスク定義をフェッチするまでは有効になりません。作業単位を非コミットのままにしておくと、既存のタスクの実行が遅延したり、実行されなかったりする場合があります。

スケジューラーは、更新されたタスクがないかどうか 5 分おきに調べます。想定どおりにタスクが実行されるようにするには、*begin_timestamp*、*end_timestamp*、および *schedule* パラメーターで定義される最も早い開始時刻は、作業単位がコミットされてから少なくとも 5 分後でなければなりません。

タスクがスケジューラーによって実行できるようにするには、データベースはすべてのデータベース・パーティションでアクティブである必要があります。

begin_timestamp、*end_timestamp*、および *schedule* はデータベース・サーバーの時間帯に基づいています。夏時間調整 (DST) の移行期間にタスクをスケジュールする場合には、特に注意が必要です。タスクが午前 2 時 1 分に実行されるようスケジュールされている場合に、それが時間を早めるときに当たると、時間は午前 2 時から午前 3 時までスキップされるため、そのタスクは実行されません。一方、時間が 1 時間戻されるときには、午前 2 時から 3 時の間にスケジュールされているタスクは 2 回実行されることになります。希望する動作が確実に実行されるようにするには、夏時間調整に関する設定をユーザーの責任で行う必要があります。

タスクが更新されると、タスクの内部呼び出しカウンターはリセットされます。一例として、*max_invocations* 値が 10 の循環タスクについて考えてみましょう。タスクが 3 回実行されると、ADMIN_TASK_STATUS 出力には 3 つの対応する状況レコードが存在することになります。エントリーの INVOCATION 値はそれぞれ、1、

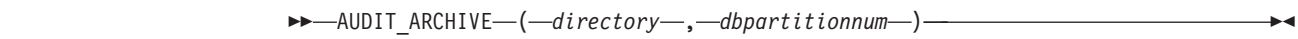
2、および 3 となります。次に、タスク作成者がこのタスクを更新すると仮定します。この更新によって、内部呼び出しカウンターはリセットされます。元の状況レコードはそのまま残されます。時間の経過とともに、1、2、3 といった INVOCATION 値を持つ新しい状況レコードが作成されます。BEGIN_TIME を使用して、元のタスク実行と更新されたタスク実行を区別することができます。

第 6 章 監査ルーチンおよびプロシージャー

AUDIT_ARCHIVE プロシージャーおよび表関数 - 監査ログ・ファイルのアーカイブ

AUDIT_ARCHIVE プロシージャーおよび表関数はどちらも、接続中のデータベースの監査ログ・ファイルをアーカイブします。

構文



スキーマは SYSPROC です。

構文は、プロシージャーと表関数のどちらでも同じです。

プロシージャーおよび表関数パラメーター

directory
アーカイブ対象監査ファイルが書き込まれるディレクトリーを指定する、タイプ VARCHAR(1024) の入力引数。ディレクトリーがサーバー上に存在しており、インスタンス所有者がそのディレクトリーにファイルを作成できなければなりません。引数が NULL または空ストリングである場合、デフォルト・ディレクトリーが使用されます。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションには -1、すべてのデータベース・パーティションの集約には NULL または -2 を指定します。

許可

AUDIT_ARCHIVE プロシージャーまたは表関数に対する Execute 特権。

例

例 1: プロシージャーを使用して、すべてのデータベース・パーティションの監査ログをデフォルト・ディレクトリーにアーカイブします。

```
CALL SYSPROC.AUDIT_ARCHIVE(NULL, NULL)
```

例 2: 表関数を使用して、すべてのデータベース・パーティションの監査ログをデフォルト・ディレクトリーにアーカイブします。

```
SELECT * FROM TABLE(SYSPROC.AUDIT_ARCHIVE('', -2)) AS T1
```

© Copyright IBM Corp. 2006, 2010

戻される情報

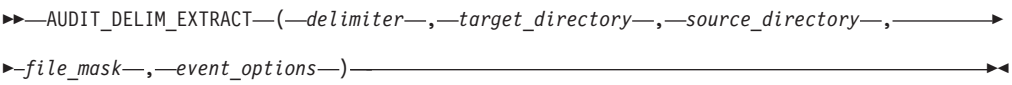
表 81. AUDIT_ARCHIVE 表関数によって戻される情報

列名	データ・タイプ	説明
DBPARTITIONNUM	INTEGER	アーカイブ対象ファイルのパーティション番号。
PATH	VARCHAR(1024)	アーカイブ対象ファイルのディレクトリー位置。
FILE	VARCHAR(1024)	アーカイブ対象ファイルの名前。
SQLCODE	INTEGER	ファイルのアーカイブ試行中に受信した SQLCODE。
SQLSTATE	VARCHAR(5)	ファイルのアーカイブ試行中に受信した SQLSTATE。 SQLSTATE が NULL の場合、値はゼロです。
SQLERRMC	VARCHAR(70) FOR BIT DATA	ファイルのアーカイブ試行中に受信した sqlerrmc。 SQLSTATE が NULL の場合、値はゼロです。

AUDIT_DELIM_EXTRACT - 区切り文字付きファイルへの抽出の実行

AUDIT_DELIM_EXTRACT ストアード・プロシージャーは、接続中のデータベースのアーカイブ対象監査ファイル上で区切り文字付きファイルへの抽出を実行します。特に、指定されたマスク・パターンに一致するファイル名を持つアーカイブ済み監査ファイルへの抽出を実行します。

構文



スキーマは SYSPROC です。

プロシージャー・パラメーター

delimiter
区切り文字付きファイルで使用される区切り文字を指定する、タイプ VARCHAR(1) のオプション入力引数。引数が NULL または空ストリングである場合、二重引用符が区切り文字として使用されます。

target_directory
区切り文字付きファイルが保管されるディレクトリーを指定する、タイプ VARCHAR(1024) のオプション入力引数。引数が NULL または空ストリングである場合、*source_directory* と同じディレクトリーが使用されます。

source_directory
アーカイブ対象監査ログ・ファイルが保管されるディレクトリーを指定する、タイプ VARCHAR(1024) のオプション入力引数。引数が NULL または空ストリングである場合、監査デフォルトが使用されます。

file_mask
どのファイルを抽出するかについてのマスクである、タイプ VARCHAR(1024) のオプション入力引数。引数が NULL または空ストリングである場合、ソース・ディレクトリーのすべての監査ログ・ファイルから抽出されます。

event_options
どのイベントを抽出するかを定義するストリングを指定する、タイプ VARCHAR(1024) のオプション入力引数。これは、db2audit ユーティリティーの同じストリングと一致します。引数が NULL または空ストリングである場合、すべてのイベントが抽出されます。

許可

AUDIT_DELIM_EXTRACT 関数に対する実行特権。

例

注: 監査ログ・ファイルには、その命名規則の一部としてタイム・スタンプが含まれます。

*例 1:*2007 年 6 月 18 日にデフォルト・アーカイブ・ディレクトリーにアーカイブされたすべての監査ログ・ファイルについて区切り付きの抽出を実行します。この例では、実行イベントのみを抽出し、二重引用符 (") 区切り文字を使用し、さらに生成された抽出ファイル (<category>.del) を $HOME/audit_delim_extract ディレクトリーで作成または追加します。

```
CALL SYSPROC.AUDIT_DELIM_EXTRACT(NULL, '$HOME/AUDIT_DELIM_EXTRACT', NULL,
'%20070618%', 'CATEGORIES EXECUTE STATUS BOTH')
```

AUDIT_LIST_LOGS 表関数 - アーカイブ対象監査ログ・ファイルのリスト

AUDIT_LIST_LOGS 表関数は、指定されたディレクトリーにある、データベースのアーカイブ対象監査ログ・ファイルをリストします。

構文

スキーマは SYSPROC です。

プロシージャー・パラメーター

directory
アーカイブ対象監査ファイルが書き込まれるディレクトリーを指定する、タイプ VARCHAR(1024) のオプション入力引数。ディレクトリーがサーバー上に存在しており、インスタンス所有者がそのディレクトリーにファイルを作成できなければなりません。引数が NULL または空ストリングである場合、検索デフォルト・ディレクトリーが使用されます。

許可

AUDIT_LIST_LOGS 表関数に対する EXECUTE 特権。

例

例 1: デフォルトの監査アーカイブ・ディレクトリーにあるアーカイブ対象監査ログをすべてリストします。

```
SELECT * FROM TABLE(SYSPROC.AUDIT_LIST_LOGS('')) AS T1
```

注: これは、照会が実行されるデータベースのディレクトリーにあるログのみリストします。アーカイブ対象ファイルのフォーマットは db2audit.db.<dbname>.log.<timestamp> です。

戻される情報

表 82. AUDIT_LIST_LOGS について戻される情報

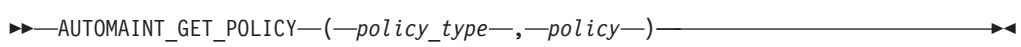
列名	データ・タイプ	説明
PATH	VARCHAR(1024)	アーカイブ対象ファイルのパス位置。
FILE	VARCHAR(1024)	アーカイブ対象ファイルのファイル名。
SIZE	BIGINT	アーカイブ対象ファイルのファイル・サイズ。

第 7 章 自動保守ルーチン

AUTOMAINT_GET_POLICY プロシージャー - 自動保守ポリシーの取得

AUTOMAINT_GET_POLICY システム・ストアード・プロシージャーは、データベースの自動保守構成を取得します。このプロシージャーは 2 つのパラメーターを取ります。1 つはどの情報を収集するかに関する自動保守のタイプで、もう 1 つは構成情報を戻す BLOB へのポインターです。構成情報は XML フォーマットで戻されます。

構文



スキーマは SYSPROC です。

プロシージャー・パラメーター

policy_type
取得する自動保守ポリシーのタイプを指定する、タイプ VARCHAR(128) の入力引数。値は以下のいずれかになります。

AUTO_BACKUP
自動バックアップ

AUTO_REORG
表および索引の自動再編成

AUTO_RUNSTATS
表の自動 RUNSTATS 操作

MAINTENANCE_WINDOW
保守ウィンドウ

policy
指定のポリシー・タイプの自動保守設定を XML フォーマットで指定する、タイプ BLOB(2M) の出力引数。

許可

AUTOMAINT_GET_POLICY プロシージャーに対する EXECUTE 特権。

例

以下の例は、組み込み SQL C ソース・コード内からの AUTOMAINT_GET_POLICY プロシージャーの呼び出しを示しています。

- プロシージャー出力パラメーター用に BLOB 変数が宣言されます。

© Copyright IBM Corp. 2006, 2010

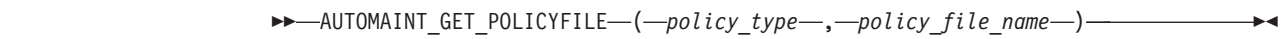
- プロシージャーが呼び出されます。このとき、自動保守ポリシーのタイプとして自動バックアップを指定し、このプロシージャーが現在接続中のデータベースのバックアップ・ポリシーを戻す出力パラメーターとして BLOB 変数を指定します。

```
EXEC SQL BEGIN DECLARE SECTION;
SQL TYPE IS BLOB(2M) backupPolicy;
EXEC SQL END DECLARE SECTION;

EXEC SQL CALL AUTOMAINT_GET_POLICY( 'AUTO_BACKUP', :backupPolicy );
```

AUTOMAINT_GET_POLICYFILE プロシージャー - 自動保守ポリシーの取得

AUTOMAINT_GET_POLICYFILE システム・ストアード・プロシージャーは、データベースの自動保守構成を取得します。このプロシージャーは 2 つのパラメーターを取ります。1 つはどの情報を収集するかに関する自動保守のタイプで、もう 1 つは構成情報を戻すファイルの名前です。構成情報は XML フォーマットで戻されます。

構文



スキーマは SYSPROC です。

プロシージャー・パラメーター

policy_type
取得する自動保守ポリシーのタイプを指定する、タイプ VARCHAR(128) の入力引数。値は以下のいずれかになります。

AUTO_BACKUP
自動バックアップ

AUTO_REORG
表および索引の自動再編成

AUTO_RUNSTATS
表の自動 RUNSTATS 操作

MAINTENANCE_WINDOW
保守ウィンドウ

policy_file_name
DB2 インスタンス・ディレクトリーの tmp サブディレクトリーに作成されるファイルの名前を指定する、タイプ VARCHAR(2048) の入力引数。

注: ファイル名には接頭部として tmp への相対パスが付けられることがあります。その場合、ディレクトリーが存在しており、ファイルを作成/上書きする許可を持っていなければなりません。また、DB2 サーバーの正しいパス区切り記号を使用する必要があります。

以下に例を示します。

インスタンス・ディレクトリーが $HOME/sqllib で定義されている場合は、UNIX。 'policy.xml' という名前のポリシー・ファイルの場合、ファイル名は '$HOME/sqllib/tmp/policy.xml' です。

Windows の場合、インスタンス・ディレクトリー名は **DB2INSTPROF** レジストリー変数および **DB2INSTANCE** 環境変数の値から決定できます。'policy.xml' という名前のポリシー・ファイルの場合、db2set が DB2INSTPROF=C:¥DB2PROF および %DB2INSTANCE%=db2 を指定すると、ファイル名は C:¥DB2PROF¥db2¥tmp¥policy.xml です。

許可

AUTOMAINT_GET_POLICYFILE プロシージャーに対する EXECUTE 特権。

例

バックアップ操作の現在の自動保守設定を取得するには、次のようにします。

```
call sysproc.automaint_get_policyfile( 'AUTO_BACKUP', 'AutoBackup.xml' )
```

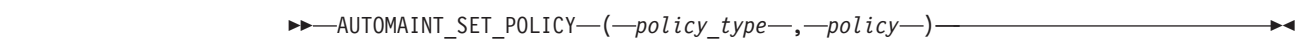
これで、AutoBackup.xml という名前の XML ファイルが DB2 インスタンス・ディレクトリーの下の tmp サブディレクトリーに作成されます。

AUTOMAINT_SET_POLICY プロシージャー - 自動保守ポリシーの構成

AUTOMAINT_SET_POLICY システム・ストアード・プロシージャーを使用して、データベースの自動保守を構成できます。このプロシージャーは 2 つのパラメーターを取ります。1 つは構成する自動保守のタイプで、もう 1 つは構成を指定する XML を含む BLOB です。

マルチディメンション・クラスタリング (MDC) 表での自動再編成操作中にエクステントの再利用を有効にするには、XML 入力ファイルで、ReorgOptions エレメントに「reclaimExtentsSizeForMDCTables」属性としきい値を指定する必要があります。このしきい値は、再利用可能な表内のフリー・スペースの最小サイズ (キロバイト単位) です。この値は、0 以上にする必要があります。例えば、しきい値として 1024 KB という値を指定する場合、エクステントを再利用するための自動再編成で考慮対象となるのは、1 MB 以上のフリー・スペースがある表だけです。

構文



スキーマは SYSPROC です。

表関数パラメーター

policy_type
構成する自動保守ポリシーのタイプを指定する、タイプ VARCHAR(128) の入力引数。値は以下のいずれかになります。

AUTO_BACKUP
自動バックアップ

AUTO_REORG
表および索引の自動再編成

AUTO_RUNSTATS
表の自動 RUNSTATS 操作

MAINTENANCE_WINDOW
保守ウィンドウ

policy
自動保守ポリシーを XML フォーマットで指定する、タイプ BLOB(2M) の入力引数。

許可

SYSPROC.AUTOMAINT_SET_POLICY プロシージャーに対する EXECUTE 特権。

例

例 1: RUNSTATS 操作の現在の自動保守設定を設定するには、次のようにします。

```
CALL SYSPROC.AUTOMAINT_SET_POLICY
  ( 'AUTO_RUNSTATS',
    BLOB(' <?xml version=¥"1.0¥" encoding=¥"UTF-8¥"?>
      <DB2AutoRunstatsPolicy
          xmlns=¥"http://www.ibm.com/xmlns/prod/db2/autonomic/config¥">
        <RunstatsTableScope><FilterCondition/></RunstatsTableScope>
      </DB2AutoRunstatsPolicy>')
  )
```

これにより、現在の自動統計収集構成が、XML 文書に含まれる新規構成と置換され、それは 2 番目のパラメーターとしてプロシージャーに渡されます。

例 2: DB2 の自動再編成フィーチャーは、新しい「RECLAIM EXTENTS ONLY」オプションを使用して、マルチディメンション・クラスタリング (MDC) 表を再編成することができます。このフィーチャーを使用可能にするには、以下に示すように AUTO_REORG ポリシー内に「reclaimExtentsSizeForMDCTables」値を設定します。

```
CALL SYSPROC.AUTOMAINT_SET_POLICY
  ('AUTO_REORG',
  BLOB(' <?xml version=¥"1.0¥" encoding=¥"UTF-8¥"?>
    <DB2AutoReorgPolicy
        xmlns=¥"http://www.ibm.com/xmlns/prod/db2/autonomic/config¥">
      <ReorgOptions dictionaryOption="Keep" indexReorgMode="Online"
          useSystemTempTableSpace="false" reclaimExtentsSizeForMDCTables ="1024" >
      <ReorgTableScope>
        <FilterClause>TABSCHEMA NOT LIKE 'EMP%'</FilterClause>
      </ReorgTableScope>
    </DB2AutoReorgPolicy>')
  )
```

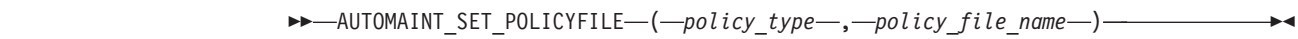
サンプル XML 入力ファイルが SQLLIB/samples/automaintcfg ディレクトリーにあります。これをご使用の要件に合わせて変更し、この例で示されるように、XML の内容を BLOB() スカラー関数に渡すことができます。

AUTOMAINT_SET_POLICYFILE プロシージャー - 自動保守ポリシーの構成

AUTOMAINT_SET_POLICYFILE システム・ストアード・プロシージャーを使用して、データベースの自動保守を構成できます。このプロシージャーは 2 つのパラメーターを取ります。1 つは構成する自動保守のタイプで、もう 1 つは構成を指定する XML 文書の名前です。

このプロシージャーは、SQL 成功または SQL エラー・コードを戻します。

構文



スキーマは SYSPROC です。

表関数パラメーター

policy_type
構成する自動保守ポリシーのタイプを指定する、タイプ VARCHAR(128) の入力引数。値は以下のいずれかになります。

AUTO_BACKUP
自動バックアップ

AUTO_REORG
表および索引の自動再編成

AUTO_RUNSTATS
表の自動 RUNSTATS 操作

MAINTENANCE_WINDOW
保守ウィンドウ

policy_file_name
DB2 インスタンス・ディレクトリーの tmp サブディレクトリーで使用可能なファイルの名前を指定する、タイプ VARCHAR(2048) の入力引数。

注: ファイル名が相対パスで指定される場合、DB2 サーバーの正しいパス区切り記号を使用する必要があり、ディレクトリーおよびファイルが読み取り許可付きで存在しなければなりません。

以下に例を示します。

インスタンス・ディレクトリーが $HOME/sqllib で定義されている場合は、UNIX。 'automaint/policy.xml' という名前のポリシー・ファイルの場合、ファイル名は '$HOME/sqllib/tmp/automaint/policy.xml' です。

Windows の場合、インスタンス・ディレクトリー名は **DB2INSTPROF** レジストリー変数および **DB2INSTANCE** 環境変数の値から決定できます。'automaint¥policy.xml' という名前のポリシー・ファイルの場合、db2set が DB2INSTPROF=C:¥DB2PROF および %DB2INSTANCE%=db2 を指定すると、ファイル名は C:¥DB2PROF¥db2¥tmp¥automaint¥policy.xml です。

許可

SYSPROC.AUTOMAINT_SET_POLICYFILE プロシージャーに対する EXECUTE 特権。

例

自動バックアップの現在の自動保守設定を変更するには、次のようにします。

```
call sysproc.automaint_set_policyfile( 'AUTO_BACKUP', 'AutoBackup.xml' )
```

これにより、現在の自動バックアップ構成設定が、DB2 インスタンス・ディレクトリーの下の tmp ディレクトリーにある AutoBackup.xml ファイルに含まれる新規構成と置換されます。

サンプル XML 入力ファイルが SQLLIB/samples/automaintcfg ディレクトリーにあります。これをポリシー xml ファイルを作成するための参照として使用することができます。

第 8 章 共通 SQL API プロシージャー

共通 SQL API は、複数の IBM データ・サーバーで共通に使用できる、共通シグニチャーの集合およびシグニチャー安定型のストアード・プロシージャーを提供します。これらのストアード・プロシージャーを使用して、構成パラメーターの取得および設定、システム情報の取得などのさまざまな管理機能を実行するアプリケーションを作成できます。

ストアード・プロシージャーは、あらゆるデータ・サーバーの上で共通に使用できる、構文的に同一の XML パラメーターとエラー処理を提供することにより、データ・サーバーのバージョン独立性を保ちます。シグニチャーの安定性と共通性は、共通 DTD を使用する簡単な XML 文書をパラメーターとして使用することにより実現されます。バージョン、プラットフォーム、およびテクノロジーの相違は、階層プロパティー・リスト内の異なるキーと値のペアで表現されます。

共通入出力パラメーター

共通 SQL API ストアード・プロシージャーは、入出力パラメーターのセットを共有します。

次の表では、これらのパラメーターの要旨を示しています。さらに詳しい情報は、共通 SQL API ストアード・プロシージャーについてのリファレンス・トピックを参照してください。

表 83. 共通 SQL API の共有入出力パラメーター

パラメーター	説明
major_version	プロシージャーでパラメーターとして渡される XML 文書に対して呼び出し元がサポートする文書タイプのメジャー・バージョンを示します。
minor_version	プロシージャーでパラメーターとして渡される XML 文書に対して呼び出し元がサポートする文書タイプのマイナー・バージョンを示します。 パラメーター *major_version* と *minor_version* は、呼び出し元が誤ったバージョンの XML 入力文書を使用しないように同時に使用されます。プロシージャーは、指定した *major_version* および *minor_version* のすべての XML 文書を処理するか、またはバージョンが無効な場合はエラーを戻します。この設計では、新しい文書タイプのバージョンの追加を既存のアプリケーションに影響を与えることなく行えるため、将来のリリースにおける拡張性がサポートされます。

© Copyright IBM Corp. 2006, 2010

表 83. 共通 SQL API の共有入出力パラメーター (続き)

パラメーター	説明
requested_locale	*xml_output* および *xml_message* パラメーターで戻された XML 文書の変換された内容を戻すために使用するロケールを指定します。値のみが変換され、キー名は変換されません。
xml_input	プロシージャーの入力値が含まれる XML 入力文書を指定します。
xml_filter	出力パラメーター文書から単一の値を検索するのに使用する有効な XPath 照会ストリングを指定します。
xml_output	UTF-8 でエンコードされた完全な XML 出力文書を戻します。呼び出されるプロシージャーに応じて、この文書には構成パラメーターとその値、システム情報、またはメッセージ・テキストが含まれる場合があります。プロシージャーが *コンプリート・モード* で動作する場合、このパラメーターによって戻される XML 文書は、変更することができ、*xml_input* パラメーターとしてプロシージャーに再び渡すことができます。このアプローチにより、有効な XML 入力文書を作成するためのプログラム的な方法が提供されます。
xml_message	SQL 警告状態についての詳細情報を提供する Data Server Message タイプの完全な XML 出力文書を UTF-8 で戻します。

XML 文書のバージョン管理

将来のリリースにおける拡張性をサポートするために、共通 SQL API ストアード・プロシージャーは、バージョン情報を含む XML 出力文書を戻します。

XML 出力文書の構造が変更される場合は常に (例えば、エレメントが追加または除去される場合)、バージョン・レベルが増分されます。したがって、プロシージャーは複数バージョンの XML 出力文書をサポートする場合があります。

XML 文書のバージョン情報は、文書タイプのメジャー・バージョンおよび文書タイプのマイナー・バージョンに関するキーと値のペアで表現されます。例えば、XML 出力文書はディクショナリーのエレメントに次のキーと値を定義する場合があります。

```
<key>Document Type Name</key><string>Data Server Configuration Output</string>
<key>Document Type Major Version</key><integer>2</integer>
<key>Document Type Minor Version</key><integer>0</integer>
```

プロシージャーを呼び出す際に、戻したい XML 文書のメジャー・バージョンとマイナー・バージョンを指定します。XML 出力文書の内容は、指定する値によって異なります。

例えば、GET_CONFIG プロシージャーは特定のインスタンスに対して設定されたデータベースおよびデータベース・マネージャー構成パラメーターを取得します。*major_version* が 2 に、*minor_version* が 0 に設定された状態でこのプロシージャーが呼び出されると、カテゴリー別にグループ化された構成パラメーターを含む XML 文書が戻されます。しかし、*major_version* が 1 に、*minor_version* が 0 に設定された状態で同じプロシージャーが呼び出されると、構成パラメーターを含む XML 文書が戻されますが、これらのパラメーターはカテゴリー別にグループ化されてはいません。

同様に、GET_MESSAGE プロシージャーは指定した SQLCODE のメッセージ・テキストと SQLSTATE を取得します。*major_version* を 2 に、*minor_version* を 0 に設定してこのプロシージャーが呼び出されると、対応する SQLCODE の短いテキスト・メッセージ、長いテキスト・メッセージ、および SQLSTATE を含む XML 文書が戻されます。ただし、 *major_version* を 1 に、 *minor_version* を 0 に設定して同じプロシージャーが呼び出されると、短いテキスト・メッセージと SQLSTATE のみを含む XML 文書が戻されます。長いテキスト・メッセージは文書のバージョン 1 では使用できません。

プロシージャーに対してサポートされる文書の最も高いバージョンを判別するには、*major_version*、*minor_version*、およびその他すべての入力パラメーターに NULL を指定します。プロシージャーは、*major_version* および *minor_version* 出力パラメーターの値として、サポートされる文書の最も高いバージョンを戻し、*xml_output* および *xml_message* 出力パラメーターを NULL に設定します。

major_version および *minor_version* に NULL 以外の値を指定する場合、サポートされる文書のバージョンを指定しなければなりません。そうしない場合、サポートされていないバージョンがプロシージャーによって検出されたことを示すエラー (-20457) がプロシージャーによって起こされます。

XML 入力文書には、オプションで文書タイプのメジャー・バージョンとマイナー・バージョンの値を組み込むことができます。これらの値が XML 入力文書に指定された場合、プロシージャー呼び出しで *major_version* および *minor_version* に渡された値は、XML 文書に指定されている値と完全に一致していなければなりません。そうでない場合にプロシージャーはエラー (+20458) を起こします。この動作により、サポートされていない XML 入力文書のバージョンを呼び出し元が指定することが防止されます。

XML 入力文書

共通 SQL API ストアード・プロシージャーへの入力として渡される XML 文書は、共通の DTD に基づく簡単な XML 形式を共有します。

XML 入力文書は、すべてのストアード・プロシージャーに共通する項目のセット、およびそれぞれのストアード・プロシージャーに固有の項目のセットから構成されています。XML 入力文書の一般的な構造は以下のとおりです。

```
<?xml version="1.0" encoding="UTF-8"?>
<plist version="1.0">
<dict>
   <key>Document Type Name</key><string>Data Server Message Input</string>
   <key>Document Type Major Version</key><integer>1</integer>
   <key>Document Type Minor Version</key><integer>0</integer>
```

```
   <key>Document Locale</key><string>en_US</string>
   <key>Complete</key><false/>
   <dict>
      <!-- Document type specific data appears here. -->
   </dict>
</dict>
</plist>
```

重要: XML 入力文書は UTF-8 でエンコードされている必要があり、含めることができるのは英語の文字のみです。

有効な XML 入力文書を戻すためのコンプリート・モード

コンプリート・モード を使用して、入力を受け入れる共通 SQL API ストアード・プロシージャーに対して有効な XML 文書を作成できます。文書をカスタマイズしてプロシージャーに戻すことができます。

コンプリート・モードでプロシージャーを実行するには、入力 XML 文書の Complete キーに true を指定し、次の最小限の内容を渡します。

```
<?xml version="1.0" encoding="UTF-8"?>
<plist version="1.0">
<dict>
   <key>Complete</key><true/>
</dict>
</plist>
```

必須でない XML エレメントはいずれも無視され、出力文書で戻されません。

プロシージャーを実行したときに、完全な XML 入力文書がストアード・プロシージャーの *xml_output* パラメーターで戻されます。戻された XML 文書には、指定可能なすべての必須パラメーターとオプション・パラメーターについての文書タイプおよびセクションが含まれます。また、戻された XML 文書には必須ではない他の項目 (表示名、ヒント、および文書ロケールなど) が含まれますが、通常、これらはクライアント・アプリケーションで文書をレンダリングする場合に必要になります。

XML 文書をレンダリングし、プラットフォームに依存しない方式でそれを変更してから、同じストアード・プロシージャーを実行し、変更した XML 文書を入力として渡すことができます。

XML 出力文書

共通 SQL API ストアード・プロシージャーから出力として戻される XML 文書には共通の項目のセットがあります。

xml_output パラメーターで戻される XML 文書には、少なくとも次の必須のキーと値のペアが含まれます。

```
<?xml version="1.0" encoding="UTF-8"?>
<plist version="1.0">
<dict>
   <key>Document Type Name</key>
   <string>Data Server Configuration Output</string>
   <key>Document Type Major Version</key><integer>1</integer>
   <key>Document Type Minor Version</key><integer>0</integer>
   <key>Data Server Product Name</key><string>DSN</string>
   <key>Data Server Product Version</key><string>8.1.0.356</string>
```

```
   <key>Data Server Major Version</key><integer>8</integer>
   <key>Data Server Minor Version</key><integer>1</integer>
   <key>Data Server Platform</key><string>z/OS</string>
   <key>Document Locale</key><string>en_US</string>

   <!-- Document type specific data appears here. -->
</dict>
</plist>
```

XML 出力文書内の項目はネストされたディクショナリーを使用してグループ化されることがあります。XML 出力文書内の各項目は、単一の情報について記述します。項目は、値、表示名、およびヒントから構成されます。オプションで、表示単位が指定される場合があります。表示名、ヒント、および表示単位は言語によって異なり、*requested_locale* パラメーターの値に指定された言語 (または要求したロケールがサポートされていない場合はデフォルト) に変換されます。通常、項目には次に示されている例のような構造があります。

```
<key>Real Storage Size</key>
<dict>
     <key>Display Name</key><string>Real Storage Size</string>
     <key>Value</key><integer>2048</integer>
     <key>Display Unit</key><string>MB</string>
     <key>Hint</key><string>Size of actual real storage online</string>
</dict>
```

IBM データ・サーバーには、すべてのデータ・サーバーに適用されるキーワード、およびデータ・サーバーに固有のキーワードを含む共通のパラメーター文書があります。データ・サーバーが新たなキーワードを追加したり除去したりする場合は常に、(すべてのデータ・サーバーの) バージョン番号が増分されます。変更内容により、メジャー・バージョン番号が増加してマイナー・バージョン番号が 0 (ゼロ) に設定されるか、またはマイナー・バージョン番号のみが増分されることがあります。

XML 出力文書は UTF-8 で生成され、英語の文字のみが含まれます。

出力のフィルタリングのための XPath 式

XPath 式を使用して、共通 SQL API ストアード・プロシージャーで戻される XML 出力をフィルタリングできます。

出力をフィルタリングするには、有効な XPath 照会ストリングをプロシージャーの *xml_filter* パラメーターに指定します。指定する XPath 式には次の制約事項が適用されます。

- XPath 式は単独の値を参照しなければなりません。
- XPath 式は常にルート・ノードからの絶対パスでなければなりません。例えば、`/`、 *`nodename`*、 `.`、および `..` のようなパス式は使用できますが、`//` および `@` のようなパス式は使用できません。
- 使用できる述部は [*`path`*='*`value`*'] および [*`n`*] のみです。
- 使用できる軸は `following-sibling` のみです。
- XPath 式の末尾は次のいずれかでなければならず、必要な場合には述部 [1] を追加しなければなりません。`following-sibling::string`、`following-sibling::data`、`following-sibling::date`、`following-sibling::real`、または `following-sibling::integer`

- 軸が XPath 式の末尾にない場合、軸の後に ::dict、::string、::data、::date、::real、または ::integer を付け、必要な場合には述部 [1] を追加しなければなりません。
- サポートされる XPath 演算子は = のみです。
- XPath 式には、関数、名前空間、処理命令、またはコメントを含めることができません。

ヒント: ストアード・プロシージャーが コンプリート・モード で動作する場合、フィルタリングを行わないでください。これを行うと、SQLCODE (+20458) が発生します。

xml_output パラメーターで戻された XML 文書の処理を細かく制御するために、DB2 pureXML® で使用可能な XMLPARSE 関数を使用できます。

例

次の XPath 式は、XML 出力文書から Data Server Product Version キーの値を選択します。

```
/plist/dict/key[.='Data Server Product Version']following-sibling::string[1]
```

プロシージャーは、*xml_output* パラメーターでストリング 8.1.0.356 を戻します。したがって、プロシージャー呼び出しは、 XML 文書ではなく単独の値を戻します。

XML メッセージ文書

共通 SQL API ストアード・プロシージャーで内部処理エラーまたは無効なパラメーターが検出された場合、データ・サーバーは SQLCODE と対応する SQL メッセージを呼び出し元に戻します。この場合、プロシージャーは、*xml_message* パラメーターで警告状態についてのより詳細な情報を含む XML メッセージ文書を戻します。

XML メッセージ文書の一般的な構造は次のとおりです。

```
<?xml version="1.0" encoding="UTF-8"?>
<plist version="1.0">
<dict>
   <key>Document Type Name</key><string>Data Server Message</string>
   <key>Document Type Major Version</key><integer>1</integer>
   <key>Document Type Minor Version</key><integer>0</integer>
   <key>Data Server Product Name</key><string>QDB2/AIX64</string>
   <key>Data Server Product Version</key><string>9.5.0.3</string>
   <key>Data Server Major Version</key><integer>9</integer>
   <key>Data Server Minor Version</key><integer>5</integer>
   <key>Data Server Platform</key><string>AIX 64BIT</string>
   <key>Document Locale</key><string>en_US</string>
   <key>Short Message Text</key>
   <dict>
      <key>Value</key><string>
      <!-- Additional description of warning appears here. --></string>
      <key>Hint</key><string></string>
   </dict>
</dict>
</plist>
```

XML メッセージ文書は UTF-8 で生成され、英語の文字のみが含まれます。

例

次の例では、GET_MESSAGE プロシージャーへの呼び出しにより SQL 警告が発生しています。

```
db2 "CALL SYSPROC.GET_MESSAGE(NULL,NULL,'en_US',NULL,NULL,?,?)"

SQL20458W プロシージャー SYSPROC.GET_MESSAGE は、
パラメーター 3 の内部パラメーター処理エラーを検出しました。
パラメーター 7 の値には、エラーに関するさらに詳しい情報が含まれています。
SQLSTATE=01H54
```

パラメーター 7 (*xml_message*) で戻された XML 文書には次の内容が含まれます。

```
<?xml version="1.0" encoding="UTF-8"?>
<plist version="1.0">
<dict>
   <key>Document Type Name</key><string>Data Server Message</string>
   <key>Document Type Major Version</key><integer>1</integer>
   <key>Document Type Minor Version</key><integer>0</integer>
   <key>Data Server Product Name</key><string>QDB2/AIX64</string>
   <key>Data Server Product Version</key><string>9.5.0.3</string>
   <key>Data Server Major Version</key><integer>9</integer>
   <key>Data Server Minor Version</key><integer>5</integer>
   <key>Data Server Platform</key><string>AIX 64BIT</string>
   <key>Document Locale</key><string>en_US</string>
   <key>Short Message Text</key>
   <dict>
      <key>Value</key><string>If parameters 1 and 2 are set to NULL, all
      other input parameters must be set to NULL as well, but the value
      of parameter "3" is not NULL. </string>
      <key>Hint</key><string></string>
   </dict>
</dict>
</plist>
```

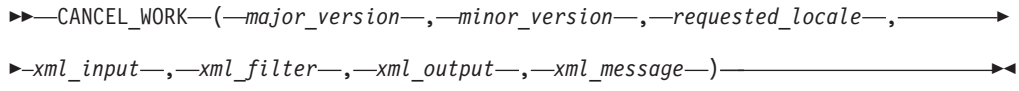
Short Message Text キーの値は、警告についての追加の情報を提供します。

CANCEL_WORK プロシージャー - 作業の取り消し

CANCEL_WORK ストアード・プロシージャーは、特定のアクティビティー (例えば SQL ステートメントなど)、または接続されたアプリケーションに対するすべてのアクティビティーを取り消します。

特定のアクティビティーを取り消すには、取り消したいアクティビティーの、アプリケーション・ハンドル、作業単位 ID、およびアクティビティー ID を渡します。接続したアプリケーションに対するすべてのアクティビティーを取り消す場合は、アプリケーション・ハンドルを渡します。取り消したアクティビティーに関連付けられているすべての変更は、ロールバックされます。

構文



スキーマは SYSPROC です。

プロシージャー・パラメーター

major_version
メジャー文書バージョンを指す、タイプ INTEGER の入出力引数。入力において、この引数は、プロシージャーでパラメーターとして渡される XML 文書に対して呼び出し元がサポートする、メジャー文書バージョンを示します (*xml_input*、*xml_output*、および *xml_message* のパラメーターの説明を参照してください)。プロシージャーは、指定したバージョンのすべての XML 文書を処理するか、またはバージョンが無効な場合はエラー (+20458) を戻します。出力において、このパラメーターは、プロシージャーによってサポートされている最も高いメジャー文書バージョンを指定します。サポートされる文書の最も高いバージョンを判別するには、この入力パラメーターおよびその他すべての必須パラメーターに NULL を指定します。

サポートされているバージョン: 1

minor_version
マイナー文書バージョンを指す、タイプ INTEGER の入出力引数。入力において、この引数は、プロシージャーでパラメーターとして渡される XML 文書に対して呼び出し元がサポートする、マイナー文書バージョンを示します (*xml_input*、*xml_output*、および *xml_message* のパラメーターの説明を参照してください)。プロシージャーは、指定したバージョンのすべての XML 文書を処理するか、またはバージョンが無効な場合はエラーを戻します。出力において、このパラメーターは、サポートされている最も高いメジャー・バージョンでサポートされている、最も高いマイナー文書バージョンを指します。サポートされる文書の最も高いバージョンを判別するには、この入力パラメーターおよびその他すべての必須パラメーターに NULL を指定します。

サポートされているバージョン: 0

requested_locale
ロケールを指定する、タイプ VARCHAR(33) の入力引数。指定した言語がサーバーでサポートされている場合は、変換された内容が *xml_output* および *xml_message* パラメーターに戻されます。サポートされていなければ、内容はデフォルトの言語で戻されます。ロケールから使用されるのは、言語 (および場合によっては地域情報) のみです。ロケールは数字のフォーマット設定に使用されたり、文書エンコードに影響を与えたりすることはありません。例えば、キー名と値は変換されません。XML 出力および XML メッセージ文書で変換される部分は、各項目のヒントのテキスト、表示名、および表示単位のみです。呼び出し元は常に、要求された言語と XML 出力文書で使用されている言語を比較する必要があります (XML 出力文書の文書ロケール項目を参照してください)。

現時点でサポートされている *requested_locale* の値は en_US のみです。

xml_input
プロシージャーの入力値が含まれる XML 入力文書 (UTF-8 でエンコードされている) を指定する、タイプ BLOB(32MB) の入力引数。

このプロシージャーでは、XML 入力文書でアプリケーション・ハンドルを指定する必要があります。特定のアクティビティーを取り消す場合は、XML 入力文書で、作業単位 ID およびアクティビティー ID を識別するオプション・パラ

メーターも指定する必要があります。このストアード・プロシージャーの完全な XML 入力文書は、次の文書のようなものになります。

```
<?xml version="1.0" encoding="UTF-8"?>
<plist version="1.0">
<dict>
   <key>Document Type Name</key><string>Data Server Cancel Work Input</string>
   <key>Document Type Major Version</key><integer>1</integer>
   <key>Document Type Minor Version</key><integer>0</integer>
   <key>Required Parameters</key>
   <dict>
      <key>Application Handle</key>
      <dict>
         <key>Display name</key><string>Application Handle</string>
         <key>Value</key><integer>10</integer>
         <key>Hint</key>
         <string>
         Numeric value equivalent to the application handle to be cancelled
         </string>
      </dict>
   </dict>
   <key>Optional Parameters</key>
   <dict>
      <key>Unit Of Work Id</key>
      <dict>
         <key>Display Name</key><string>Unit Of Work Id</string>
         <key>Value</key><integer>20</integer>
         <key>Hint</key>
         <string>
         Numeric value that specifies the unit of work id of the activity
         that is to be cancelled
         </string>
      </dict>
      <key>Activity Id</key>
      <dict>
         <key>Display Name</key><string>Activity Id</string>
         <key>Value</key><integer>10</integer>
         <key>Hint</key>
         <string>
         Numeric value equivalent to the activity id to be cancelled
         </string>
      </dict>
   </dict>
</dict>
</plist>
```

ストアード・プロシージャーが実行されているアプリケーションのアプリケーション・ハンドルを指定した場合、プロシージャーは警告 (SQL20458) を戻します。

xml_filter
有効な XPath 照会ストリングを指定する、タイプ BLOB(4K) の入力引数。XML 出力文書から単一値を検索する場合、フィルターを使用します。詳しくは、XPath フィルター操作について記述しているトピックを参照してください。

次の例では、XML 出力文書から、データ・サーバー製品バージョンの値を選択しています (`/plist/dict/key[.='Data Server Product Version']/following-sibling::string`)。キーの後に兄弟が指定されていないと、エラーが戻されます。

xml_output
UTF-8 の完全な XML 出力文書を戻す、タイプ BLOB(32MB) の出力パラメーターを戻します。フィルターが指定されている場合、このパラメーターはストリ

ング値を戻します。ストアード・プロシージャーが完全な出力文書を戻すことができない場合 (例えば、処理エラーが発生して SQL 警告やエラーが出される場合など)、このパラメーターは NULL に設定されます。

XML 出力は、以下のように *major_version* および *minor_version* で指定した値によって決まります。

メジャー・バージョン	マイナー・バージョン	*xml_output* の値
NULL	NULL	NULL
1	0	プロシージャーが取り消そうとしているアクティビティーの状況。

プロシージャーが コンプリート・モード で動作する場合、このパラメーターによって戻される XML 文書は、変更して *xml_input* パラメーターとしてプロシージャーに再び渡すことができます。このアプローチにより、有効な XML 入力文書を作成するためのプログラム的な方法が提供されます。詳しくは、コンプリート・モードについてのトピックを参照してください。

xml_message

SQL 警告状態についての詳細情報を提供する Data Server Message タイプの完全な XML 出力文書を UTF-8 で戻す、タイプ BLOB(64K) の出力パラメーター。プロシージャーの呼び出しにより SQL 警告が出され、XML メッセージ出力文書に追加情報が戻されることを警告メッセージが示す場合に、この文書が戻されます。追加情報が戻されることを警告メッセージが示していない場合は、このパラメーターは NULL に設定されます。

許可

- SYSADM または DBADM 権限
- CANCEL_WORK プロシージャーに対する EXECUTE 特権

例

例 1: プロシージャーの、サポートされる最も高いバージョンを戻します。

```
db2 "call sysproc.cancel_work(null,null,null,null,null,?,?)"
```

以下はこの照会の出力例です。

```
Value of output parameters
--------------------------
Parameter Name : MAJOR_VERSION
Parameter Value : 1

Parameter Name : MINOR_VERSION
Parameter Value : 0

Parameter Name : XML_OUTPUT
Parameter Value : -

Parameter Name : XML_MESSAGE
Parameter Value : -

Return Status = 0
```

例 2: 特定のアクティビティーを取り消します。

```
db2 "call sysproc.cancel_work(1,0,'en_US',blob(
<?xml version="1.0" encoding="UTF-8"?>
<plist version="1.0">
<dict>
   <key>Document Type Name</key><string>Data Server Cancel Work Input</string>
   <key>Document Type Major Version</key><integer>1</integer>
   <key>Document Type Minor Version</key><integer>0</integer>
   <key>Required Parameters</key>
   <dict>
      <key>Application Handle</key>
      <dict>
          <key>Display name</key><string>Application Handle</string>
          <key>Value</key><integer>1</integer>
          <key>Hint</key>
          <string>
          Numeric value equivalent to the application handle to be cancelled
          </string>
      </dict>
   </dict>
   <key>Optional Parameters</key>
   <dict>
      <key>Unit Of Work Id</key>
      <dict>
          <key>Display Name</key><string>Unit Of Work Id</string>
          <key>Value</key><integer>2</integer>
          <key>Hint</key>
          <string>
          Numeric value that specifies the unit of work id of the activity
          that is to be cancelled
          </string>
      </dict>
      <key>Activity Id</key>
      <dict>
         <key>Display Name</key><string>Activity Id</string>
         <key>Value</key><integer>3</integer>
         <key>Hint</key>
         <string>
         Numeric value equivalent to the activity id to be cancelled
         </string>
      </dict>
   </dict>
</dict>
</plist> ) ,null,?,?)"
```

以下はこの照会の出力例です。

```
Value of output parameters
--------------------------
Parameter Name : MAJOR_VERSION
Parameter Value : 1

Parameter Name : MINOR_VERSION
Parameter Value : 0

Parameter Name : XML_OUTPUT
Parameter Value : x'3C3F78...'

Parameter Name : XML_MESSAGE
Parameter Value : -

Return Status = 0
```

CANCEL_WORK プロシージャーでアクティビティーを取り消せる場合、 XML 出力文書には次の内容が含まれます。

```
<?xml version="1.0" encoding="UTF-8"?>
<plist version="1.0">
<dict><key>Document Type Name</key><string>Data Server Cancel Work Output</string>
   <key>Document Type Major Version</key><integer>1</integer>
   <key>Document Type Minor Version</key><integer>0</integer>
   <key>Data Server Product Name</key><string>QDB2/AIX64</string>
   <key>Data Server Product Version</key><string>9.7.0.0</string>
   <key>Data Server Major Version</key><integer>9</integer>
   <key>Data Server Minor Version</key><integer>7</integer>
   <key>Data Server Platform</key><string>AIX 64BIT</string>
   <key>Document Locale</key><string>en_US</string>
   <key>Successful Cancel Work Message</key>
   <dict>
      <key>Display Name</key><string>Successful Cancel Work Message</string>
      <key>Value</key><string>The activity has been cancelled successfully</string>
      <key>Hint</key><string></string>
   </dict>
</dict>
</plist>
```

例 2: アプリケーションを取り消します。

```
db2 "call sysproc.cancel_work(1,0,'en_US,blob(
<?xml version="1.0" encoding="UTF-8"?>
<plist version="1.0">
<dict>
   <key>Document Type Name</key><string>Data Server Cancel Work Input</string>
   <key>Document Type Major Version</key><integer>1</integer>
   <key>Document Type Minor Version</key><integer>0</integer>
   <key>Required Parameters</key>
   <dict>
      <key>Application Handle</key>
      <dict>
          <key>Display name</key><string>Application Handle</string>
          <key>Value</key><integer>101</integer>
          <key>Hint</key>
          <string>
          Numeric value equivalent to the application handle to be cancelled
          </string>
      </dict>
   </dict>
</dict>
</plist> ),null,?,?)"
```

以下はこの照会の出力例です。

```
Value of output parameters
--------------------------
Parameter Name : MAJOR_VERSION
Parameter Value : 1

Parameter Name : MINOR_VERSION
Parameter Value : 0

Parameter Name : XML_OUTPUT
Parameter Value : x'3C3F78...'

Parameter Name : XML_MESSAGE
Parameter Value : -

Return Status = 0
```

CANCEL_WORK プロシージャーでアプリケーションを取り消せる場合、 XML 出力文書には次の内容が含まれます。

```
<?xml version="1.0" encoding="UTF-8"?>
<plist version="1.0">
<dict>
   <key>Document Type Name</key><string>Data Server Cancel Work Output</string>
   <key>Document Type Major Version</key><integer>1</integer>
   <key>Document Type Minor Version</key><integer>0</integer>
   <key>Data Server Product Name</key><string>QDB2/AIX64</string>
   <key>Data Server Product Version</key><string>9.7.0.0</string>
   <key>Data Server Major Version</key><integer>9</integer>
   <key>Data Server Minor Version</key><integer>7</integer>
   <key>Data Server Platform</key><string>AIX 64BIT</string>
   <key>Document Locale</key><string>en_US</string>
   <key>Successful Cancel Work Message</key>
   <dict>
      <key>Display Name</key><string>Successful Cancel Work Message</string>
      <key>Value</key>
      <string>The application has been cancelled successfully</string>
      <key>Hint</key><string></string>
   </dict>
</dict>
</plist>
```

例 3: フィルターを指定して、成功した取り消し処理メッセージの値を戻します。

```
db2 "call sysproc.cancel_work(1,0,'en_US,blob(
<?xml version="1.0" encoding="UTF-8"?>
<plist version="1.0">
<dict>
   <key>Document Type Name</key><string>Data Server Cancel Work Input</string>
   <key>Document Type Major Version</key><integer>1</integer>
   <key>Document Type Minor Version</key><integer>0</integer>
   <key>Required Parameters</key>
   <dict>
      <key>Application Handle</key>
      <dict>
          <key>Display name</key><string>Application Handle</string>
          <key>Value</key><integer>101</integer>
          <key>Hint</key>
          <string>
          Numeric value equivalent to the application handle to be cancelled
          </string>
      </dict>
   </dict>
</dict>
</plist> ),blob('/plist/dict/key[.="Successful Cancel Work Message"]
/following-sibling::dict[1]/key[.="Value"]
/following-sibling::string[1]'),?,?)"
```

以下はこの照会の出力例です。

```
Value of output parameters
--------------------------
Parameter Name : MAJOR_VERSION
Parameter Value : 1

Parameter Name : MINOR_VERSION
Parameter Value : 0

Parameter Name : XML_OUTPUT
Parameter Value : x'3C3F78...'

Parameter Name : XML_MESSAGE
Parameter Value : -

Return Status = 0
```

xml_output に次の値が戻されます。

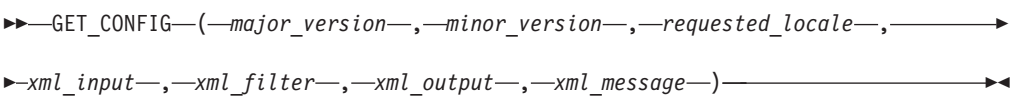
```
"The application has been cancelled successfully"
```

GET_CONFIG プロシージャー - 構成データの取得

GET_CONFIG ストアード・プロシージャーは、特定のインスタンスに対して設定されたデータベース構成、データベース・マネージャー構成、およびレジストリー変数を検索します。

パーティション・データベース環境では、このプロシージャーはすべてのパーティションから、データベース構成およびレジストリー変数設定値を検索します。

構文



スキーマは SYSPROC です。

プロシージャー・パラメーター

major_version
メジャー文書バージョンを指す、タイプ INTEGER の入出力引数。入力において、この引数は、プロシージャーでパラメーターとして渡される XML 文書に対して呼び出し元がサポートする、メジャー文書バージョンを示します (*xml_input*、*xml_output*、および *xml_message* のパラメーターの説明を参照してください)。プロシージャーは、指定したバージョンのすべての XML 文書を処理するか、またはバージョンが無効な場合はエラー (+20458) を戻します。出力において、このパラメーターは、プロシージャーによってサポートされている最も高いメジャー文書バージョンを指定します。サポートされる文書の最も高いバージョンを判別するには、この入力パラメーターおよびその他すべての必須パラメーターに NULL を指定します。

サポートされているバージョン: 1 および 2

minor_version
マイナー文書バージョンを指す、タイプ INTEGER の入出力引数。入力において、この引数は、プロシージャーでパラメーターとして渡される XML 文書に対して呼び出し元がサポートする、マイナー文書バージョンを示します (*xml_input*、*xml_output*、および *xml_message* のパラメーターの説明を参照してください)。プロシージャーは、指定したバージョンのすべての XML 文書を処理するか、またはバージョンが無効な場合はエラーを戻します。出力において、このパラメーターは、サポートされている最も高いメジャー・バージョンでサポートされている、最も高いマイナー文書バージョンを指します。サポートされる文書の最も高いバージョンを判別するには、この入力パラメーターおよびその他すべての必須パラメーターに NULL を指定します。

サポートされているバージョン: 0

requested_locale
ロケールを指定する、タイプ VARCHAR(33) の入力引数。指定した言語がサーバーでサポートされている場合は、変換された内容が *xml_output* および

xml_message パラメーターに戻されます。サポートされていなければ、内容はデフォルトの言語で戻されます。ロケールから使用されるのは、言語 (および場合によっては地域情報) のみです。ロケールは数字のフォーマット設定に使用されたり、文書エンコードに影響を与えたりすることはありません。例えば、キー名と値は変換されません。XML 出力および XML メッセージ文書で変換される部分は、各項目のヒントのテキスト、表示名、および表示単位のみです。呼び出し元は常に、要求された言語と XML 出力文書で使用されている言語を比較する必要があります (XML 出力文書の文書ロケール項目を参照してください)。

現時点でサポートされている *requested_locale* の値は en_US のみです。

xml_input
現時点で、このプロシージャーは入力を受け入れません。このパラメーターに NULL を指定してください。NULL を指定しないとエラー (+20458) が発生し、入力が無効であることが示されます。

xml_filter
有効な XPath 照会ストリングを指定する、タイプ BLOB(4K) の入力引数。XML 出力文書から単一値を検索する場合、フィルターを使用します。詳しくは、XPath フィルター操作について記述しているトピックを参照してください。

次の例では、XML 出力文書から、データ・サーバー製品バージョンの値を選択しています (`/plist/dict/key[.='Data Server Product Version']/following-sibling::string`)。キーの後に兄弟が指定されていないと、エラーが戻されます。

xml_output
UTF-8 の完全な XML 出力文書を戻す、タイプ BLOB(32MB) の出力パラメーターを戻します。フィルターが指定されている場合、このパラメーターはストリング値を戻します。ストアード・プロシージャーが完全な出力文書を戻すことができない場合 (例えば、処理エラーが発生して SQL 警告やエラーが出される場合など)、このパラメーターは NULL に設定されます。

XML 出力は、以下のように *major_version* および *minor_version* で指定した値によって決まります。

メジャー・バージョン	**マイナー・バージョン**	***xml_output* の値**
NULL	NULL	NULL
1	0	データベース・マネージャー構成パラメーター、データベース構成パラメーター、およびレジストリー変数とそれぞれの値

メジャー・バージョン	マイナー・バージョン	*xml_output* の値
2	0	カテゴリーにグループ化された、データベース・マネージャーおよびデータベース・マネージャー構成パラメーター各パラメーターに関して、そのパラメーターが更新可能かどうかを示します。さらに、インスタンスに設定された、レジストリー変数および値も戻します。

プロシージャーが コンプリート・モード で動作する場合、このパラメーターによって戻される XML 文書は、変更して *xml_input* パラメーターとしてプロシージャーに再び渡すことができます。このアプローチにより、有効な XML 入力文書を作成するためのプログラム的な方法が提供されます。詳しくは、コンプリート・モードについてのトピックを参照してください。

xml_message

SQL 警告状態についての詳細情報を提供する Data Server Message タイプの完全な XML 出力文書を UTF-8 で戻す、タイプ BLOB(64K) の出力パラメーター。プロシージャーの呼び出しにより SQL 警告が出され、XML メッセージ出力文書に追加情報が戻されることを警告メッセージが示す場合に、この文書が戻されます。追加情報が戻されることを警告メッセージが示していない場合は、このパラメーターは NULL に設定されます。

許可

- SYSADM または DBADM 権限
- GET_CONFIG プロシージャーに対する EXECUTE 特権

例

例 1: プロシージャーの最新バージョンを戻します。

```
db2 "call sysproc.get_config(null,null,null,null,null,?,?)"
```

以下はこの照会の出力例です。

```
Value of output parameters
--------------------------
Parameter Name : MAJOR_VERSION
Parameter Value : 2

Parameter Name : MINOR_VERSION
Parameter Value : 0

Parameter Name : XML_OUTPUT
Parameter Value : -

Parameter Name : XML_MESSAGE
Parameter Value : -

Return Status = 0
```

例 2: カテゴリーにグループ化された、データベースおよびデータベース・マネージャー構成パラメーターを戻します。

```
db2 "call sysproc.get_config(2,0,'en_US',null, null, ?,?)"
```

以下はこの照会の出力例です。

```
 Value of output parameters
--------------------------
Parameter Name : MAJOR_VERSION
Parameter Value : 2

Parameter Name : MINOR_VERSION
Parameter Value : 0

Parameter Name : XML_OUTPUT
Parameter Value : x'3C3F78.......'

Parameter Name : XML_MESSAGE
Parameter Value : -

Return Status = 0
```

XML 出力文書には次の内容が含まれます。

```
<plist version="1.0">
<dict>
   <key>Document Type Name</key><string>Data Server Configuration Output</string>
   <key>Document Type Major Version</key><integer>2</integer>
   <key>Document Type Minor Version</key><integer>0</integer>
   <key>Data Server Product Name</key><string>QDB2/AIX64</string>
   <key>Data Server Product Version</key><string>9.7.0.0</string>
   <key>Data Server Major Version</key><integer>9</integer>
   <key>Data Server Minor Version</key><integer>7</integer>
   <key>Data Server Platform</key><string>AIX 64BIT</string>
   <key>Document Locale</key><string>en_US</string>
   <key>Database Manager Configuration Parameter Settings</key>
   <dict>
      <key>Display Name</key>
      <string>Database Manager Configuration Parameter Settings</string>
      <key>Application</key>
      <dict>
         <key>Display Name</key><string>Application</string>
         <key>agentpri</key>
         <dict>
            <key>Display Name</key><string>agentpri</string>
            <key>Parameter Value</key>
            <dict>
               <key>Display Name</key><string>Parameter Value</string>
               <key>Value</key><string>-1</string>
               <key>Updatable</key><string>No</string>
               <key>Hint</key><string></string>
            </dict>
            <key>Value Flags</key>
            <dict>
               <key>Display Name</key><string>Value Flags</string>
               <key>Value</key><string>NONE</string>
               <key>Updatable</key><string>No</string>
               <key>Hint</key><string></string>
            </dict>
            <key>Deferred Value</key>
            <dict>
               <key>Display Name</key><string>Deferred Value</string>
               <key>Value</key><string>-1</string>
               <key>Updatable</key><string>Yes</string>
               <key>Hint</key><string></string>
            </dict>
            <key>Deferred Value Flags</key>
            <dict>
               <key>Display Name</key><string>Deferred Value Flags</string>
```

```
            <key>Value</key><string>INTEGER</string>
            <key>Updatable</key><string>Yes</string>
            <key>Hint</key><string></string>
         </dict>
         <key>Data Type</key>
         <dict>
            <key>Display Name</key><string>Data Type</string>
            <key>Value</key><string>NONE</string>
            <key>Hint</key><string></string>
         </dict>
         <key>Hint</key>
         <string>
         Specifies the priority given to an agent and other database manager
         instance processes and threads by the operating system scheduler.
         Consider rebinding applications after changing this parameter.
         </string>
      </dict>
      <key>Hint</key><string></string>
   </dict>
</dict>
<key>Administration</key>
.
.
.
<key>Communication</key>
.
.
.
<key>Diagnostics</key>
.
.
.
<key>Environment</key>
.
.
.
<key>Miscellaneous</key>
.
.
.
<key>Monitor</key>
.
.
.
<key>Parallel</key>
.
.
.
<key>Performance</key>
.
.
.
</dict>
<key>Database Partition</key>
<dict>
   <key>Display Name</key><string>Database Partition</string>
   <key>0</key>
   <dict>
      <key>Display Name</key><string>0</string>
      <key>Database Configuration Parameter Settings</key>
      <dict>
         <key>Display Name</key>
         <string>Database Configuration Parameter Settings</string>
         <key>Application</key>
         .
         .
         .
```

```
   <key>Environment</key>
<dict>
   <key>Display Name</key><string>Environment</string>
   <key>alt_collate</key>
   <dict>
      <key>Display Name</key><string>alt_collate</string>
      <key>Parameter Value</key>
      <dict>
         <key>Display Name</key><string>Parameter Value</string>
         <key>Value</key><string></string>
         <key>Updatable</key><string>No</string>
         <key>Hint</key><string></string>
      </dict>
      <key>Value Flags</key>
      <dict>
         <key>Display Name</key><string>Value Flags</string>
         <key>Value</key><string>NONE</string>
         <key>Updatable</key><string>No</string>
         <key>Hint</key><string></string>
      </dict>
      <key>Deferred Value</key>
      <dict>
         <key>Display Name</key><string>Deferred Value</string>
         <key>Value</key><string></string>
         <key>Updatable</key><string>Yes</string>
         <key>Hint</key><string></string>
      </dict>
      <key>Deferred Value Flags</key>
      <dict>
         <key>Display Name</key><string>Deferred Value Flags</string>
         <key>Value</key><string>INTEGER</string>
         <key>Updatable</key><string>Yes</string>
         <key>Hint</key><string></string>
      </dict>
      <key>Data Type</key>
      <dict>
         <key>Display Name</key><string>Data Type</string>
         <key>Value</key><string>NONE</string>
         <key>Hint</key><string></string>
      </dict>
      <key>Hint</key>
      <string>
      Specifies the collating sequence to be used for Unicode tables in a
      non-Unicode database. Until this parameter is set, Unicode tables and
      routines cannot be created in a non-Unicode database. When set, this
      parameter cannot be changed or reset. Default [range] :
      Null [IDENTITY_16BIT].
      </string>
   </dict>
   .
   .
   .
</dict>
<key>Logs</key>
.
.
.
<key>Maintenance</key>
.
.
.
<key>Performance</key>
.
.
.
<key>Recovery</key>
.
```

```
            .
            .
            <key>Status</key>
            .
            .
            .
         </dict>
         <key>Registry Variables Settings</key>
         <dict>
            <key>Display Name</key><string>Registry Variables Settings</string>
            <key>DB2CODEPAGE</key>
            <dict>
               <key>Display Name</key><string>DB2CODEPAGE</string>
               <key>Parameter Value</key>
               <dict>
                  <key>Display Name</key><string>Parameter Value</string>
                  <key>Value</key><string>1208</string>
                  <key>Hint</key><string></string>
               </dict>
               <key>Is Aggregate</key>
               <dict>
                  <key>Display Name</key><string>Is Aggregate</string>
                  <key>Value</key><integer>0</integer>
                  <key>Hint</key><string></string>
               </dict>
               <key>Aggregate Name</key>
               <dict>
                  <key>Display Name</key><string>Aggregate Name</string>
                  <key>Value</key><string></string>
                  <key>Hint</key><string></string>
               </dict>
               <key>Level</key>
               <dict>
                  <key>Display Name</key><string>Level</string>
                  <key>Value</key><string>I</string>
                  <key>Hint</key><string></string>
               </dict>
               <key>Hint</key><string></string>
            </dict>
            .
            .
            .
         </dict>
         <key>Hint</key><string></string>
      </dict>
</dict>
</plist>
```

例 3: データベースおよびデータベース・マネージャー構成パラメーターを戻します。

```
db2 "call sysproc.get_config(1,0,'en_US',null, null, ?,?)"
```

以下はこの照会の出力例です。

```
Value of output parameters
--------------------------
Parameter Name : MAJOR_VERSION
Parameter Value : 1

Parameter Name : MINOR_VERSION
Parameter Value : 0

Parameter Name : XML_OUTPUT
Parameter Value : x'3C3F78.......'
```

```
Parameter Name : XML_MESSAGE
Parameter Value : -

Return Status = 0
```

XML 出力文書には、例 2 のような内容が含まれますが、構成パラメーターはカテゴリーにグループ化されません。

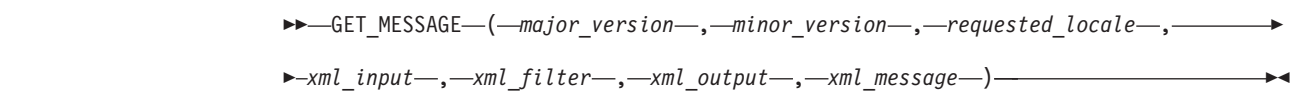
例 4: 関数からプロシージャーを呼び出します。

```
EXEC SQL BEGIN DECLARE SECTION;
  sqlint16  getconfigMaj;
  sqlint16  getconfigMin;

  SQL TYPE IS BLOB(2M) xmlOutput;
  SQL TYPE IS BLOB(2K) xmlOutMessage;
EXEC SQL END DECLARE SECTION;
  getconfigMaj = 2;
  getconfigMin = 0;

  EXEC SQL CALL SYSPROC.GET_CONFIG(
                :getconfigMaj,
                :getconfigMin,
                'en_US',
                null,
                null,
                :xmlOutput,
                :xmlOutMessage );
```

GET_MESSAGE プロシージャー - メッセージ・テキストの取得

GET_MESSAGE プロシージャーは、SQLCODE の簡略メッセージ・テキスト、詳細メッセージ・テキスト、および SQLSTATE を戻します。

構文



スキーマは SYSPROC です。

プロシージャー・パラメーター

major_version

メジャー文書バージョンを指す、タイプ INTEGER の入出力引数。入力において、この引数は、プロシージャーでパラメーターとして渡される XML 文書に対して呼び出し元がサポートする、メジャー文書バージョンを示します (*xml_input*、*xml_output*、および *xml_message* のパラメーターの説明を参照してください)。プロシージャーは、指定したバージョンのすべての XML 文書を処理するか、またはバージョンが無効な場合はエラー (+20458) を戻します。出力において、このパラメーターは、プロシージャーによってサポートされている最も高いメジャー文書バージョンを指定します。サポートされる文書の最も高いバージョンを判別するには、この入力パラメーターおよびその他すべての必須パラメーターに NULL を指定します。

xml_input パラメーターの XML 文書で、文書タイプのメジャー・バージョンのキーが指定されている場合は、そのキーの値は *major_version* パラメーターで指定される値と同じでなければなりません。同じでない場合、エラー (+20458) が発生します。

サポートされているバージョン: 1 および 2

minor_version
マイナー文書バージョンを指す、タイプ INTEGER の入出力引数。入力において、この引数は、プロシージャーでパラメーターとして渡される XML 文書に対して呼び出し元がサポートする、マイナー文書バージョンを示します (*xml_input*、*xml_output*、および *xml_message* のパラメーターの説明を参照してください)。プロシージャーは、指定したバージョンのすべての XML 文書を処理するか、またはバージョンが無効な場合はエラーを戻します。出力において、このパラメーターは、サポートされている最も高いメジャー・バージョンでサポートされている、最も高いマイナー文書バージョンを指します。サポートされる文書の最も高いバージョンを判別するには、この入力パラメーターおよびその他すべての必須パラメーターに NULL を指定します。

xml_input パラメーターの XML 文書で、文書タイプのマイナー・バージョンのキーが指定されている場合は、そのキーの値は *minor_version* パラメーターで指定される値と同じでなければなりません。同じでない場合、エラー (+20458) が発生します。

サポートされているバージョン: 0

requested_locale
ロケールを指定する、タイプ VARCHAR(33) の入力引数。指定した言語がサーバーでサポートされている場合は、変換された内容が *xml_output* および *xml_message* パラメーターに戻されます。サポートされていなければ、内容はデフォルトの言語で戻されます。ロケールから使用されるのは、言語 (および場合によっては地域情報) のみです。ロケールは数字のフォーマット設定に使用されたり、文書エンコードに影響を与えたりすることはありません。例えば、キー名と値は変換されません。XML 出力および XML メッセージ文書で変換される部分は、各項目のヒントのテキスト、表示名、および表示単位のみです。呼び出し元は常に、要求された言語と XML 出力文書で使用されている言語を比較する必要があります (XML 出力文書の文書ロケール項目を参照してください)。

現時点でサポートされている *requested_locale* の値は en_US のみです。

xml_input
プロシージャーの入力値が含まれる XML 入力文書 (UTF-8 でエンコードされている) を指定する、タイプ BLOB(32MB) の入力引数。

このプロシージャーでは、XML 入力文書に SQLCODE が含まれ、次のフォーマットを使用します。

```
<?xml version="1.0" encoding="UTF-8"?>
<plist version="1.0">
<dict>
    <key>Document Type Name</key><string>Data Server Message Input</string>
    <key>Required Parameters</key>
    <!-- Specify either SQLCODE or message identifier and message tokens
    for the key values below. -->
    <dict>
        <key>SQL Code</key><integer></integer>
```

```
        <key>Message Identifier</key><integer></integer>
        <key>Message Tokens</key><array><string>...</string></array>
    </dict>
    <key>Optional Parameters</key>
    <dict>
        <key>Message Token Delimiter<key><string>;</string>
    </key></key></dict>
</dict>
</plist>
```

xml_filter
有効な XPath 照会ストリングを指定する、タイプ BLOB(4K) の入力引数。XML 出力文書から単一値を検索する場合、フィルターを使用します。詳しくは、XPath フィルター操作について記述しているトピックを参照してください。

次の例では、XML 出力文書から SQLSTATE の値を選択しています (`/plist/dict/key[.="SQLSTATE"]/following-sibling::dict[1]/key[.="Value"]/following-sibling::string[1]`)。キーの後に兄弟が指定されていないと、エラーが戻されます。

xml_output
UTF-8 の完全な XML 出力文書を戻す、タイプ BLOB(32MB) の出力パラメーターを戻します。フィルターが指定されている場合、このパラメーターはストリング値を戻します。ストアード・プロシージャーが完全な出力文書を戻すことができない場合 (例えば、処理エラーが発生して SQL 警告やエラーが出される場合など)、このパラメーターは NULL に設定されます。

XML 出力は、以下のように *major_version* および *minor_version* で指定した値によって決まります。

メジャー・バージョン	**マイナー・バージョン**	***xml_output* の値**
NULL	NULL	NULL
1	0	*xml_input* に渡された対応する SQLCODE の、簡略テキスト・メッセージおよび SQLSTATE を戻します。
2	0	*xml_input* に渡された対応する SQLCODE の、簡略テキスト・メッセージ、詳細テキスト・メッセージ、および SQLSTATE を戻します。

プロシージャーが コンプリート・モード で動作する場合、このパラメーターによって戻される XML 文書は、変更して *xml_input* パラメーターとしてプロシージャーに再び渡すことができます。このアプローチにより、有効な XML 入力文書を作成するためのプログラム的な方法が提供されます。詳しくは、コンプリート・モードについてのトピックを参照してください。

xml_message
SQL 警告状態についての詳細情報を提供する Data Server Message タイプの完全な XML 出力文書を UTF-8 で戻す、タイプ BLOB(64K) の出力パラメーター。プロシージャーの呼び出しにより SQL 警告が出され、XML メッセージ出力文書に追加情報が戻されることを警告メッセージが示す場合に、この文書が戻

されます。追加情報が戻されることを警告メッセージが示していない場合は、このパラメーターは NULL に設定されます。

許可

- SYSADM または DBADM 権限
- GET_MESSAGE プロシージャーに対する EXECUTE 特権

例

例 1: プロシージャーの、サポートされる最も高いバージョンを戻します。

```
db2 "call sysproc.get_message(null,null,null,null,null,?,?)"
```

以下はこの照会の出力例です。

```
Value of output parameters
--------------------------
Parameter Name : MAJOR_VERSION
Parameter Value : 2

Parameter Name : MINOR_VERSION
Parameter Value : 0

Parameter Name : XML_OUTPUT
Parameter Value : -

Parameter Name : XML_MESSAGE
Parameter Value : -

Return Status = 0
```

例 2: getmsglong.sql というスクリプトを実行し、 SQL1034 の簡略テキスト・メッセージおよび詳細テキスト・メッセージを戻します。

```
getmsglong.sql:

call sysproc.get_message(2,0, 'en_US', blob('
<?xml version="1.0" encoding="UTF-8"?>
<plist version="1.0">
<dict>
  <key>Document Type Name</key><string>Data Server Message Input</string>
  <key>Document Type Major Version</key><integer>2</integer>
  <key>Document Type Minor Version</key><integer>0</integer>
  <key>Required Parameters</key>
  <dict>
    <key>SQLCODE</key><string>SQL1034</string>
  </dict>
</dict>
</plist>'), null, ? , ?)@
```

以下はこの照会の出力例です。

```
Value of output parameters
--------------------------
Parameter Name : MAJOR_VERSION
Parameter Value : 2

Parameter Name : MINOR_VERSION
Parameter Value : 0

Parameter Name : XML_OUTPUT
Parameter Value : x'3C3F786D6C20766572.....................'
```

```
Parameter Name : XML_MESSAGE
Parameter Value : -

Return Status = 0
```

出力 XML 文書には次の内容が含まれます。

```
<plist version="1.0">
<dict>
  <key>Document Type Name</key>
  <string>Data Server Message Output</string>
  <key>Document Type Major Version</key>
  <integer>2</integer>
  <key>Document Type Minor Version</key>
  <integer>0</integer>
  <key>Data Server Product Name</key>
  <string>QDB2/AIX64</string>
  <key>Data Server Product Version</key>
  <string>9.7.0.0</string>
  <key>Data Server Major Version</key>
  <integer>9</integer>
  <key>Data Server Minor Version</key>
  <integer>7</integer>
  <key>Data Server Platform</key>
  <string>AIX 64BIT</string>
  <key>Document Locale</key>
  <string>en_US</string>
  <key>Short Message Text</key>
  <dict>
  <key>Display Name</key><string>Short Message Text</string>
  <key>Value</key>
  <string>
  SQL1034C  The database is damaged.  All applications processing the database
  have been stopped.
  </string>
  <key>Hint</key><string></string>
</dict>
  <key>SQLSTATE</key>
  <dict>
    <key>Display Name</key><string>SQLSTATE</string>
    <key>Value</key><string> 58031</string>
    <key>Hint</key><string></string>
  </dict>
  <key>Long Message Text</key>
  <dict>
    <key>Display Name</key><string>Long Message Text</string>
    <key>Value</key>
    <array>
      <string>
      SQL1034C  The database is damaged. All applications
      processing the
      </string>
      <string>      database have been stopped.</string>
      <string></string>
      <string>Explanation: </string>
      <string></string>
      <string>
      Damage has occurred to the database. It cannot be used until it is
      </string>
      <string>
      recovered. All applications connected to the database have been
      </string>
      <string>
      disconnected and all processes running applications on the
      database have
      </string>
      <string>been stopped.</string>
```

```
      <string></string>
      <string>The command cannot be processed.</string>
      <string></string>
      <string>User response: </string>
      <string></string>
      <string>
      Issue a RESTART DATABASE command to recover the database. If the RESTART
      </string>
      <string>
      command consistently fails, you may want to restore the database from a
      </string>
      <string>
      backup. In a partitioned database server environment, check the syslog
      </string>
      <string>
      to find out if the RESTART command fails because of node or
      </string>
      <string>
      communication failures before restoring the database from a backup. If
      </string>
      <string>
      so, ensure the database manager is up and running and communication is
      </string>
      <string>
      available among all the nodes, then resubmit the restart command.
      </string>
      <string></string>
      <string>
      If you encountered this error during roll-forward processing, you must
      </string>
      <string>
      restore the database from a backup and perform roll-forward again.
      </string>
      <string></string>
      <string>
      Note that in a partitioned database environment, the RESTART database
      </string>
      <string>
      command is run on a per-node basis. To ensure that the database is
      </string>
      <string>restarted on all nodes, use the command: </string>
      <string></string>
      <string>db2_all db2 restart database</string>
      <string><database_name></string>
      <string></string>
      <string>
      This command may have to be run several times to ensure that all
      </string>
      <string>in-doubt transactions have been resolved.</string>
      <string></string>
      <string>
      If you are installing the sample database, drop it and install the
      </string>
      <string>sample database again.</string>
      <string></string>
      <string> sqlcode: -1034</string>
      <string></string>
      <string> sqlstate: 58031</string>
      <string></string>
      <string></string>
      <string></string>
      </array>
    <key>Hint</key><string></string>
  </dict>
</dict>
</plist>
```

例 3: getmsgshort.sql というスクリプトを実行し、 SQL1034 の簡略テキスト・メッセージのみを戻します。

```
getmsgshort.sql:

call sysproc.get_message(1,0,'en_US', blob('
<?xml version="1.0" encoding="UTF-8"?>
<plist version="1.0">
<dict>
  <key>Document Type Name</key><string>Data Server Message Input</string>
  <key>Document Type Major Version</key><integer>1</integer>
  <key>Document Type Minor Version</key><integer>0</integer>
  <key>Required Parameters</key>
  <dict>
    <key>SQLCODE</key><string>SQL1034</string>
  </dict>
</dict>
</plist>'), null, ? , ?)@
```

以下はこの照会の出力例です。

```
Value of output parameters
--------------------------
Parameter Name : MAJOR_VERSION
Parameter Value : 2

Parameter Name : MINOR_VERSION
Parameter Value : 0

Parameter Name : XML_OUTPUT
Parameter Value : x'3C3F786D6C20766572.....................'

Parameter Name : XML_MESSAGE
Parameter Value : -

Return Status = 0

SQL20460W The procedure "SYSPROC.GET_MESSAGE" supports a higher version, "2",
than the specified version, "1", for parameter "1".
```

XML 出力文書には次の内容が含まれます。

```
<plist version="1.0">
<dict><key>Document Type Name</key><string>Data Server Message Output</string>
  <key>Document Type Major Version</key><integer>1</integer>
  <key>Document Type Minor Version</key><integer>0</integer>
  <key>Data Server Product Name</key><string>QDB2/AIX64</string>
  <key>Data Server Product Version</key><string>9.7.0.0</string>
  <key>Data Server Major Version</key><integer>9</integer>
  <key>Data Server Minor Version</key><integer>7</integer>
  <key>Data Server Platform</key><string>AIX 64BIT</string>
  <key>Document Locale</key><string>en_US</string>
  <key>Short Message Text</key>
  <dict>
    <key>Display Name</key><string>Short Message Text</string>
    <key>Value</key>
    <string>
    SQL1034C  The database is damaged.  All applications processing the database
    have been stopped.
    </string>
    <key>Hint</key><string></string>
  </dict>
  <key>SQLSTATE</key>
  <dict>
    <key>Display Name</key><string>SQLSTATE</string>
    <key>Value</key><string> 58031</string>
```

```
    <key>Hint</key><string></string>
  </dict>
</dict>
</plist>
```

例 4: フィルターを指定して、 SQL1034 の SQLSTATE を戻します。

```
db2 "call sysproc.get_message(2,0, 'en_US', blob('
<plist version="1.0">
<dict>
   <key>Document Type Name</key>
   <string>Data Server Message Input</string>
   <key>Required Parameters</key>
   <dict>
      <key>SQLCODE</key><string>SQL1034</string>
   </dict>
</dict>
</plist>'),
blob('/plist/dict/key[.="SQLSTATE"]/following-sibling::dict[1]/
key[.="Value"]/following-sibling::string[1]'), ? , ?)"
```

以下はこの照会の出力例です。

```
Value of output parameters
--------------------------
Parameter Name : MAJOR_VERSION
Parameter Value : 2

Parameter Name : MINOR_VERSION
Parameter Value : 0

Parameter Name : XML_OUTPUT
Parameter Value : x'203538303331'

Parameter Name : XML_MESSAGE
Parameter Value : -

Return Status = 0
```

xml_output に次の値が戻されます。

```
58031
```

例 5: 関数からプロシージャーを呼び出します。

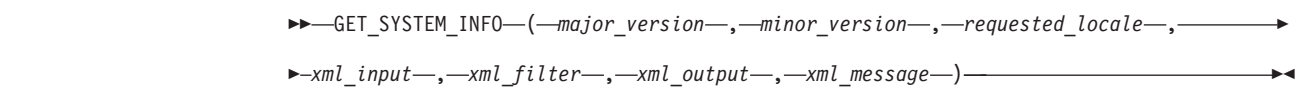
```
EXEC SQL BEGIN DECLARE SECTION;
  sqlint16  getMsgMaj;
  sqlint16  getMsgMin;

  SQL TYPE IS BLOB(2M) xmlOutput;
  SQL TYPE IS BLOB(2K) xmlOutMessage;
 EXEC SQL END DECLARE SECTION;
   getMsgMaj = 2;
   getMsgMin = 0;

  EXEC SQL CALL SYSPROC.GET_MESSAGE(
                :getMsgMaj,
                :getMsgMin,
                'en_US',
                BLOB('
                <?xml version="1.0" encoding="UTF-8"?>
                <plist version="1.0">
                <dict>
                   <key>Document Type Name</key>
                   <string>
                   Data Server Message Input
                   </string>
```

```
    <key>Document Type Major Version</key><integer>2</integer>
    <key>Document Type Minor Version</key><integer>0</integer>
    <key>Required Parameters</key>
    <dict>
       <key>SQLCODE</key><string>SQL1034</string>
    </dict>
 </dict>
 </plist>'),
 null,
:xmlOutput,
:xmlOutMessage );
```

GET_SYSTEM_INFO プロシージャー - システム情報の取得

GET_SYSTEM_INFO プロシージャーは、データ・サーバーについての情報を戻します。その情報には、システムに関する情報、現行インスタンス、インストール済みデータ・サーバー製品、環境変数、使用可能な CPU、および他のシステム情報が含まれます。

構文



スキーマは SYSPROC です。

プロシージャー・パラメーター

major_version

メジャー文書バージョンを指す、タイプ INTEGER の入出力引数。入力において、この引数は、プロシージャーでパラメーターとして渡される XML 文書に対して呼び出し元がサポートする、メジャー文書バージョンを示します (*xml_input*、*xml_output*、および *xml_message* のパラメーターの説明を参照してください)。プロシージャーは、指定したバージョンのすべての XML 文書を処理するか、またはバージョンが無効な場合はエラー (+20458) を戻します。出力において、このパラメーターは、プロシージャーによってサポートされている最も高いメジャー文書バージョンを指定します。サポートされる文書の最も高いバージョンを判別するには、この入力パラメーターおよびその他すべての必須パラメーターに NULL を指定します。

xml_input パラメーターの XML 文書で、文書タイプのメジャー・バージョンのキーが指定されている場合は、そのキーの値は *major_version* パラメーターで指定される値と同じでなければなりません。同じでない場合、エラー (+20458) が発生します。

サポートされているバージョン: 1

minor_version

マイナー文書バージョンを指す、タイプ INTEGER の入出力引数。入力において、この引数は、プロシージャーでパラメーターとして渡される XML 文書に対して呼び出し元がサポートする、マイナー文書バージョンを示します (*xml_input*、*xml_output*、および *xml_message* のパラメーターの説明を参照してください)。プロシージャーは、指定したバージョンのすべての XML 文書を処

理するか、またはバージョンが無効な場合はエラーを戻します。出力において、このパラメーターは、サポートされている最も高いメジャー・バージョンでサポートされている、最も高いマイナー文書バージョンを指します。サポートされる文書の最も高いバージョンを判別するには、この入力パラメーターおよびその他すべての必須パラメーターに NULL を指定します。

サポートされているバージョン: 0

requested_locale
ロケールを指定する、タイプ VARCHAR(33) の入力引数。指定した言語がサーバーでサポートされている場合は、変換された内容が *xml_output* および *xml_message* パラメーターに戻されます。サポートされていなければ、内容はデフォルトの言語で戻されます。ロケールから使用されるのは、言語 (および場合によっては地域情報) のみです。ロケールは数字のフォーマット設定に使用されたり、文書エンコードに影響を与えたりすることはありません。例えば、キー名と値は変換されません。XML 出力および XML メッセージ文書で変換される部分は、各項目のヒントのテキスト、表示名、および表示単位のみです。呼び出し元は常に、要求された言語と XML 出力文書で使用されている言語を比較する必要があります (XML 出力文書の文書ロケール項目を参照してください)。

現時点でサポートされている *requested_locale* の値は en_US のみです。

xml_input
現時点で、このプロシージャーは入力を受け入れません。このパラメーターに NULL を指定してください。NULL を指定しないとエラー (+20458) が発生し、入力が無効であることが示されます。

xml_filter
有効な XPath 照会ストリングを指定する、タイプ BLOB(4K) の入力引数。XML 出力文書から単一値を検索する場合、フィルターを使用します。詳しくは、XPath フィルター操作について記述しているトピックを参照してください。

次の例では、XML 出力文書から、データ・サーバー製品バージョンの値を選択しています (`/plist/dict/key[.='Data Server Product Version']/following-sibling::string`)。キーの後に兄弟が指定されていないと、エラーが戻されます。

xml_output
UTF-8 の完全な XML 出力文書を戻す、タイプ BLOB(32MB) の出力パラメーターを戻します。フィルターが指定されている場合、このパラメーターはストリング値を戻します。ストアード・プロシージャーが完全な出力文書を戻すことができない場合 (例えば、処理エラーが発生して SQL 警告やエラーが出される場合など)、このパラメーターは NULL に設定されます。

XML 出力文書には、フィックスパック・レベル、リリース、システム情報、および環境変数に関する情報などの、インスタンス情報が含まれます。

xml_message
SQL 警告状態についての詳細情報を提供する Data Server Message タイプの完全な XML 出力文書を UTF-8 で戻す、タイプ BLOB(64K) の出力パラメーター。プロシージャーの呼び出しにより SQL 警告が出され、XML メッセージ出力文書に追加情報が戻されることを警告メッセージが示す場合に、この文書が戻

されます。追加情報が戻されることを警告メッセージが示していない場合は、このパラメーターは NULL に設定されます。

許可

- SYSADM または DBADM 権限
- GET_SYSTEM_INFO プロシージャーに対する EXECUTE 特権。

例

例 1: プロシージャーの最も高いバージョンを戻します。

```
db2 "call sysproc.get_system_info(null,null,null,null,null,?,?)"
```

以下はこの照会の出力例です。

```
Value of output parameters
  --------------------------
  Parameter Name  : MAJOR_VERSION
  Parameter Value : 1

  Parameter Name  : MINOR_VERSION
  Parameter Value : 0

  Parameter Name  : XML_OUTPUT
  Parameter Value : -

  Parameter Name  : XML_MESSAGE
  Parameter Value : -

  Return Status = 0
```

例 2: システム情報を戻します。

```
 db2 "call sysproc.get_system_info(1,0,'en_US',null,null,?,?)"
```

以下はこの照会の出力例です。

```
  Value of output parameters
  --------------------------
  Parameter Name  : MAJOR_VERSION
  Parameter Value : 1

  Parameter Name  : MINOR_VERSION
  Parameter Value : 0

  Parameter Name  : XML_OUTPUT
  Parameter Value : x'3C3F786D6C20766572.....

  Parameter Name  : XML_MESSAGE
  Parameter Value : -

  Return Status = 0
```

XML 出力文書には次のような内容が含まれます。

```
<plist version="1.0">
<dict><key>Document Type Name</key><string>Data Server System Output</string>
   <key>Document Type Major Version</key><integer>1</integer>
   <key>Document Type Minor Version</key><integer>0</integer>
   <key>Data Server Product Name</key><string>QDB2/AIX64</string>
   <key>Data Server Product Version</key><string>9.7.0.0</string>
   <key>Data Server Major Version</key><integer>9</integer>
   <key>Data Server Minor Version</key><integer>7</integer>
   <key>Data Server Platform</key><string>AIX 64BIT</string>
```

```
<key>Document Locale</key><string>en_US</string>
<key>Instance Information</key>
<dict>
   <key>Display Name</key><string>Instance Information</string>
   <key>Instance Name</key>
   <dict>
      <key>Display Name</key><string>Instance Name</string>
      <key>Value</key><string>myinstance</string>
      <key>Hint</key><string></string>
   </dict>
   <key>Partitionable State</key>
   <dict>
      <key>Display Name</key><string>Partitionable State</string>
      <key>Value</key><integer>0</integer>
      <key>Hint</key><string></string>
   </dict>
   <key>Number of Database Partitions</key>
   <dict>
      <key>Display Name</key><string>Number of Database Partitions</string>
      <key>Value</key><integer>1</integer>
      <key>Hint</key><string></string>
   </dict>
   .
   .
   .
</dict>
<key>Product Information</key>
<dict>
   <key>Display Name</key><string>Product Information</string>
   .
   .
   .
   <key>DB2_ENTERPRISE_SERVER_EDITION</key>
   <dict>
      <key>Display Name</key><string>DB2_ENTERPRISE_SERVER_EDITION</string>
      <key>Product short name</key>
      <dict>
         <key>Display Name</key><string>Product short name</string>
         <key>Value</key><string>ESE</string>
         <key>Hint</key><string></string>
      </dict>
      <key>Licence</key>
      <dict>
         <key>Display Name</key><string>Licence</string>
         <key>Value</key><string>Y</string>
         <key>Hint</key><string></string>
      </dict>
      <key>Product Release</key>
      <dict>
         <key>Display Name</key><string>Product Release</string>
         <key>Value</key><string>9.7</string>
         <key>Hint</key><string></string>
      </dict>
      <key>Licence type</key>
      <dict>
         <key>Display Name</key><string>Licence type</string>
         <key>Value</key><string>DEVELOPER</string>
         <key>Hint</key><string></string>
      </dict>
      <key>Hint</key><string></string>
   </dict>
   .
   .
   .
<key>Operating System Information</key>
<dict>
   <key>Display Name</key><string>Operating System Information</string>
```

```
   <key>Name</key>
   <dict>
      <key>Display Name</key><string>Name</string>
      <key>Value</key><string>AIX</string>
      <key>Hint</key><string></string>
   </dict>
   <key>Version</key>
   <dict>
      <key>Display Name</key><string>Version</string>
      <key>Value</key><string>5</string>
      <key>Hint</key><string></string>
   </dict>
   <key>Release</key>
   <dict>
      <key>Display Name</key><string>Release</string>
      <key>Value</key><string>3</string>
      <key>Hint</key><string></string>
   </dict>
   <key>Hostname</key>
   <dict>
      <key>Display Name</key><string>Hostname</string>
      <key>Value</key><string>achilles</string>
      <key>Hint</key><string></string>
   </dict>
   .
   .
   .
</dict>
<key>Workload Management Configuration</key>
<dict>
   <key>Display Name</key><string>Workload Management Configuration</string>
   <key>Service Class Information</key>
   <dict>
      <key>Display Name</key><string>Service Class Information</string>
      <key>1</key>
      <dict>
         <key>Display Name</key><string>1</string>
         <key>Service Class Name</key>
         <dict>
            <key>Display Name</key><string>Service Class Name</string>
            <key>Value</key><string>SYSDEFAULTSYSTEMCLASS</string>
            <key>Hint</key><string></string>
      </dict>
      <key>Parent Identifier</key>
      <dict>
         <key>Display Name</key><string>Parent Identifier</string>
         <key>Value</key><integer>0</integer>
         <key>Hint</key><string></string>
      </dict>
      <key>Parent Class Name</key>
      <dict>
         <key>Display Name</key><string>Parent Class Name</string>
         <key>Value</key><string></string>
         <key>Hint</key><string></string>
      </dict>
      <key>Creation Time</key>
      <dict>
         <key>Display Name</key><string>Creation Time</string>
         <key>Value</key><string>2008-04-21-15.14.32.956930</string>
         <key>Hint</key><string></string>
      </dict>
      <key>Alter Time</key>
      <dict>
         <key>Display Name</key><string>Alter Time</string>
         <key>Value</key><string>2008-04-21-15.14.32.956930</string>
         <key>Hint</key><string></string>
      </dict>
```

```
   <key>Enabled</key>
   <dict>
      <key>Display Name</key><string>Enabled</string>
      <key>Value</key><string>Y</string>
      <key>Hint</key><string></string>
   </dict>
   <key>Agent Priority</key>
   <dict>
      <key>Display Name</key><string>Agent Priority</string>
      <key>Value</key><integer>-32768</integer>
      <key>Hint</key><string></string>
   </dict>
   <key>Prefetcher Priority</key>
   <dict>
      <key>Display Name</key><string>Prefetcher Priority</string>
      <key>Value</key><string> </string>
      <key>Hint</key><string></string>
   </dict>
   .
   .
   .
</dict>
.
.
.

<key>Workload Information</key>
<dict>
   <key>Display Name</key><string>Workload Information</string>
   <key>1</key>
   <dict>
      <key>Display Name</key><string>1</string>
      <key>Workload Name</key>
      <dict>
         <key>Display Name</key><string>Workload Name</string>
         <key>Value</key><string>SYSDEFAULTUSERWORKLOAD</string>
         <key>Hint</key><string></string>
      </dict>
      <key>Evaluation Order</key>
      <dict>
         <key>Display Name</key><string>Evaluation Order</string>
         <key>Value</key><integer>1</integer>
         <key>Hint</key><string></string>
      </dict>
      <key>Creation Time</key>
      <dict>
         <key>Display Name</key><string>Creation Time</string>
         <key>Value</key><string>2008-04-21-15.14.32.955296</string>
         <key>Hint</key><string></string>
      </dict>
      <key>Alter Time</key>
      <dict>
         <key>Display Name</key><string>Alter Time</string>
         <key>Value</key><string>2008-04-21-15.14.32.955296</string>
         <key>Hint</key><string></string>
      </dict>
      <key>Enabled</key>
      <dict>
         <key>Display Name</key><string>Enabled</string>
         <key>Value</key><string>Y</string>
         <key>Hint</key><string></string>
      </dict>
      <key>Allow Access</key>
      <dict>
         <key>Display Name</key><string>Allow Access</string>
         <key>Value</key><string>Y</string>
         <key>Hint</key><string></string>
```

```
            </dict>
            <key>Service Class Name</key>
            <dict>
               <key>Display Name</key><string>Service Class Name</string>
               <key>Value</key><string>SYSDEFAULTSUBCLASS</string>
               <key>Hint</key><string></string>
            </dict>
            <key>Parent Service Class Name</key>
            <dict>
               <key>Display Name</key><string>Parent Service Class Name</string>
               <key>Value</key><string>SYSDEFAULTUSERCLASS</string>
               <key>Hint</key><string></string>
            </dict>
            .
            .
            .
         </dict>
         <key>Hint</key><string></string>
      </dict>
   </dict>
</dict></dict></dict></plist>
```

例 3: GET_SYSTEM_INFO プロシージャーを呼び出し、サポートされないロケールを渡します。

```
db2 "call sysproc. get_system_info(1,0,'ja_JP',null,null,?,?)"
```

以下はこの照会の出力例です。

```
  Value of output parameters
  --------------------------
  Parameter Name  : MAJOR_VERSION
  Parameter Value : 1

  Parameter Name  : MINOR_VERSION
  Parameter Value : 0

  Parameter Name  : XML_OUTPUT
  Parameter Value : x'3C3F786D6C20766572.....

  Parameter Name  : XML_MESSAGE
  Parameter Value : -

  Return Status = 0

  SQL20461W  The procedure "SYSPROC.GET_SYSTEM_INFO" returned output in the
alternate locale, "en_US", instead of the locale, "ja_JP", specified in
parameter "3". SQLSTATE=01H57
```

XML 出力文書には、例 2 に示されているのと同じ内容が含まれます。

例 4: 関数からプロシージャーを呼び出します。

```
 EXEC SQL BEGIN DECLARE SECTION;
  sqlint16  getSysInfMaj;
  sqlint16  getSysInfMin;

  SQL TYPE IS BLOB(2M) xmlOutput;
  SQL TYPE IS BLOB(2K) xmlOutMessage;
 EXEC SQL END DECLARE SECTION;
   getSysInfMaj = 1;
   getSysInfMin = 0;

  EXEC SQL CALL SYSPROC.GET_SYSTEM_INFO(
                :getSysInfMaj,
                :getSysInfMin,
```

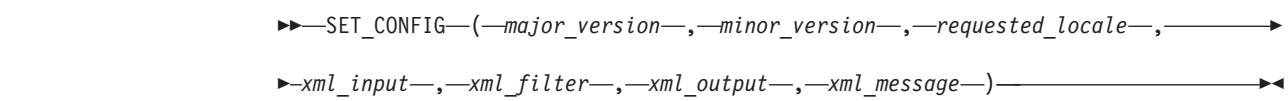
```
'en_US',
 null,
 null,
:xmlOutput,
:xmlOutMessage );
```

SET_CONFIG プロシージャー - 構成パラメーターの設定

SET_CONFIG ストアード・プロシージャーは、GET_CONFIG プロシージャーで戻される、データベースおよびデータベース・マネージャー構成パラメーターを更新します。

SET_CONFIG プロシージャーは、構成パラメーターとその値が含まれる入力 XML 文書を受け入れ、その情報を使用して指定の構成パラメーターを更新します。さらに、各構成パラメーターの更新状況を示す出力 XML 文書を戻します。

構文



スキーマは SYSPROC です。

プロシージャー・パラメーター

major_version
メジャー文書バージョンを指す、タイプ INTEGER の入出力引数。入力において、この引数は、プロシージャーでパラメーターとして渡される XML 文書に対して呼び出し元がサポートする、メジャー文書バージョンを示します (*xml_input*、*xml_output*、および *xml_message* のパラメーターの説明を参照してください)。プロシージャーは、指定したバージョンのすべての XML 文書を処理するか、またはバージョンが無効な場合はエラー (+20458) を戻します。出力において、このパラメーターは、プロシージャーによってサポートされている最も高いメジャー文書バージョンを指定します。サポートされる文書の最も高いバージョンを判別するには、この入力パラメーターおよびその他すべての必須パラメーターに NULL を指定します。

xml_input パラメーターの XML 文書で、文書タイプのメジャー・バージョンのキーが指定されている場合は、そのキーの値は *major_version* パラメーターで指定される値と同じでなければなりません。同じでない場合、エラー (+20458) が発生します。

サポートされているバージョン: 1

minor_version
マイナー文書バージョンを指す、タイプ INTEGER の入出力引数。入力において、この引数は、プロシージャーでパラメーターとして渡される XML 文書に対して呼び出し元がサポートする、マイナー文書バージョンを示します (*xml_input*、*xml_output*、および *xml_message* のパラメーターの説明を参照してください)。プロシージャーは、指定したバージョンのすべての XML 文書を処理するか、またはバージョンが無効な場合はエラーを戻します。出力において、

このパラメーターは、サポートされている最も高いメジャー・バージョンでサポートされている、最も高いマイナー文書バージョンを指します。サポートされる文書の最も高いバージョンを判別するには、この入力パラメーターおよびその他すべての必須パラメーターに NULL を指定します。

xml_input パラメーターの XML 文書で、文書タイプのマイナー・バージョンのキーが指定されている場合は、そのキーの値は *minor_version* パラメーターで指定される値と同じでなければなりません。同じでない場合、エラー (+20458) が発生します。

サポートされているバージョン: 0

requested_locale
ロケールを指定する、タイプ VARCHAR(33) の入力引数。指定した言語がサーバーでサポートされている場合は、変換された内容が *xml_output* および *xml_message* パラメーターに戻されます。サポートされていなければ、内容はデフォルトの言語で戻されます。ロケールから使用されるのは、言語 (および場合によっては地域情報) のみです。ロケールは数字のフォーマット設定に使用されたり、文書エンコードに影響を与えたりすることはありません。例えば、キー名と値は変換されません。XML 出力および XML メッセージ文書で変換される部分は、各項目のヒントのテキスト、表示名、および表示単位のみです。呼び出し元は常に、要求された言語と XML 出力文書で使用されている言語を比較する必要があります (XML 出力文書の文書ロケール項目を参照してください)。

現時点でサポートされている *requested_locale* の値は en_US のみです。

xml_input
プロシージャーの入力値が含まれる XML 入力文書 (UTF-8 でエンコードされている) を指定する、タイプ BLOB(32MB) の入力引数。

このプロシージャーでは、XML 入力文書にデータベースおよびデータベース・マネージャー構成設定が含まれます。

xml_filter
有効な XPath 照会ストリングを指定する、タイプ BLOB(4K) の入力引数。XML 出力文書から単一値を検索する場合、フィルターを使用します。詳しくは、XPath フィルター操作について記述しているトピックを参照してください。

次の例では、XML 出力文書から、特定の構成パラメーター設定の値を選択しています (`/plist/dict/key[.="Database Manager Configuration Parameter Settings"]/following-sibling::dict[1]/key[3]/following-sibling::dict[1]/dict[1]/key[.="Value"]/following-sibling::string[1]`)。キーの後に兄弟が指定されていないと、エラーが戻されます。

xml_output
UTF-8 の完全な XML 出力文書を戻す、タイプ BLOB(32MB) の出力パラメーターを戻します。フィルターが指定されている場合、このパラメーターはストリング値を戻します。ストアード・プロシージャーが完全な出力文書を戻すことができない場合 (例えば、処理エラーが発生して SQL 警告やエラーが出される場合など)、このパラメーターは NULL に設定されます。

このプロシージャーが *complete* モードで動作する場合、このパラメーターによって戻される XML 文書には、サーバーで設定されている現行の構成値が含まれます。この文書を変更して、*xml_input* パラメーターとしてプロシージャーに

戻すことができます。このアプローチにより、有効な XML 入力文書を作成するためのプログラム的な方法が提供されます。

xml_message
SQL 警告状態についての詳細情報を提供する Data Server Message タイプの完全な XML 出力文書を UTF-8 で戻す、タイプ BLOB(64K) の出力パラメーター。プロシージャーの呼び出しにより SQL 警告が出され、XML メッセージ出力文書に追加情報が戻されることを警告メッセージが示す場合に、この文書が戻されます。追加情報が戻されることを警告メッセージが示していない場合は、このパラメーターは NULL に設定されます。

許可

- SYSADM または DBADM 権限
- SET_CONFIG プロシージャーに対する EXECUTE 特権

例

例 1: プロシージャーの最新バージョンを戻します。

```
db2 "call  sysproc.set_config (null,null,null,null,null,?,?)"
```

以下はこの照会の出力例です。

```
Value of output parameters
--------------------------
Parameter Name : MAJOR_VERSION
Parameter Value : 1

Parameter Name : MINOR_VERSION
Parameter Value : 0

Parameter Name : XML_OUTPUT
Parameter Value : -

Parameter Name : XML_MESSAGE
Parameter Value : -

Return Status = 0
```

例 2: いくつかのデータベースおよびデータベース・マネージャー構成パラメーターを更新する `setconfig.sql` というスクリプトを実行します。

```
setconfig.sql:

call sysproc.set_config(1,0,'en_US',blob('
<?xml version="1.0" encoding="UTF-8"?>
<plist version="1.0">
<dict>
  <key>Document Type Name</key><string>Data Server Set Configuration Input</string>
  <key>Document Type Major Version</key><integer>1</integer>
  <key>Document Type Minor Version</key><integer>0</integer>
  <key>Document Locale</key><string>en_US</string>
  <key>Database Manager Configuration Parameter Settings</key>
  <dict>
    <key>diaglevel</key><dict><key>Parameter Value</key>
    <dict>
      <key>Value</key><string>4</string>
    </dict>
  </dict>
  <key>fcm_num_buffers</key>
  <dict>
    <key>Parameter Value</key>
```

```
    <dict>
      <key>Value</key><string>4096</string>
    </dict>
    <key>Value Flags</key>
    <dict>
      <key>Value</key><string>MANUAL</string>
    </dict>
  </dict>
  <key>instance_memory</key>
  <dict>
    <key>Deferred Value</key>
    <dict>
      <key>Value</key><string>7424</string>
    </dict>
    <key>Deferred Value Flags</key>
    <dict>
        <key>Value</key><string>AUTOMATIC</string>
    </dict>
  </dict>
  </dict>
  <key>Database Partition</key>
  <dict>
    <key>All</key>
    <dict>
      <key>Database Configuration Parameter Settings</key>
      <dict>
        <key>avg_appls</key>
        <dict>
          <key>Parameter Value</key>
          <dict>
            <key>Value></key><string>2</string>
          </dict>
          <key>Value Flags</key>
          <dict>
            <key>Value</key><string>AUTOMATIC</string>
          </dict>
        </dict>
        <key>database_memory</key>
        <dict>
          <key>Deferred Value</key>
          <dict>
            <key>Value</key><string>2</string>
          </dict>
          <key>Deferred Value Flags</key>
          <dict>
            <key>Value</key><string>MANUAL</string>
          </dict>
        </dict>
      </dict>
    </dict>
  </dict>
</dict>
</plist>'), null, ?,?)@
```

以下はこの照会の出力例です。

```
Value of output parameters
--------------------------
Parameter Name : MAJOR_VERSION
Parameter Value : 1

Parameter Name : MINOR_VERSION
Parameter Value : 0

Parameter Name : XML_OUTPUT
Parameter Value : x'3C3F78...'
```

```
Parameter Name : XML_MESSAGE
Parameter Value : -

Return Status = 0
```

出力 XML 文書には、次のような内容が含まれます。

```
<?xml version="1.0" encoding="UTF-8"?>
<plist version="1.0">
<dict>
  <key>Document Type Name</key>
  <string>Data Server Set Configuration Output</string>
  <key>Document Type Major Version</key><integer>1</integer>
  <key>Document Type Minor Version</key><integer>0</integer>
  <key>Data Server Product Name</key><string>QDB2/AIX64</string>
  <key>Data Server Product Version</key><string>9.7.0.0</string>
  <key>Data Server Major Version</key><integer>9</integer>
  <key>Data Server Minor Version</key><integer>7</integer>
  <key>Data Server Platform</key><string>AIX 64BIT</string>
  <key>Document Locale</key><string>en_US</string>
  <key>Database Manager Configuration Parameter Settings</key>
  <dict>
    <key>Display Name</key>
    <string>Database Manager Configuration Parameter Settings</string>
    <key>diaglevel</key>
    <dict>
      <key>Display Name</key><string>diaglevel</string>
      <key>Parameter Value</key>
      <dict>
        <key>Display Name</key><string>Parameter Value</string>
        <key>Value</key><string>4</string>
      </dict>
      <key>Parameter Update Status</key>
      <dict>
        <key>Display Name</key><string>Parameter Update Status</string>
        <key>SQLCODE</key>
        <dict
          <key>Display Name</key><string>SQLCODE</string>
          <key>Value</key><integer>0</integer>
        </dict>
      <key>Message Tokens</key>
      <dict>
        <key>Display Name</key><string>Message Tokens</string>
        <key>Value</key><array><string></string></array>
      </dict>
      <key>SQLSTATE</key>
      <dict>
        <key>Display Name</key><string>SQLSTATE</string>
        <key>Value</key><string></string>
      </dict>
    </dict>
  </dict>
  <key>fcm_num_buffers</key>
  <dict>
    <key>Display Name</key><string>fcm_num_buffers</string>
    <key>Parameter Value</key>
      <dict>
        <key>Display Name</key><string>Parameter Value</string>
        <key>Value</key><string>4096</string>
      </dict>
      <key>Value Flags</key>
      <dict>
        <key>Display Name</key><string>Value Flags</string>
        <key>Value</key><string>MANUAL</string>
      </dict>
      <key>Parameter Update Status</key>
      <dict>
```

```
      <key>Display Name</key><string>Parameter Update Status</string>
      <key>SQLCODE</key><dict>
      <key>Display Name</key><string>SQLCODE</string>
      <key>Value</key> <integer>0</integer>
    </dict>
    <key>Message Tokens</key>
    <dict>
      <key>Display Name</key><string>Message Tokens</string>
      <key>Value</key><array><string></string></array>
    </dict>
    <key>SQLSTATE</key>
    <dict>
      <key>Display Name</key><string>SQLSTATE</string>
      <key>Value</key><string></string>
    </dict>
  </dict>
</dict>
<key>instance_memory</key>
<dict>
  <key>Display Name</key><string>instance_memory</string>
  <key>Deferred Value</key>
  <dict>
    <key>Display Name</key><string>Deferred Value</string>
   <key>Value</key><string>7424</string>
  </dict>
  <key>Deferred Value Flags</key>
  <dict>
    <key>Display Name</key><string>Deferred Value Flags</string>
    <key>Value</key><string>AUTOMATIC</string>
  </dict>
  <key>Parameter Update Status</key>
  <dict>
    <key>Display Name</key><string>Parameter Update Status</string>
    <key>SQLCODE</key>
    <dict>
      <key>Display Name</key><string>SQLCODE</string>
      <key>Value</key><integer>0</integer>
    </dict>
      <key>Message Tokens</key>
      <dict>
        <key>Display Name</key><string>Message Tokens</string>
        <key>Value</key><array><string></string></array>
      </dict>
      <key>SQLSTATE</key>
       <dict>
         <key>Display Name</key><string>SQLSTATE</string>
         <key>Value</key><string></string>
       </dict>
     </dict>
   </dict>
  </dict>
  <key>Database Partition</key>
  <dict>
    <key>Display Name</key><string>Database Partition</string>
    <key>All</key>
    <dict>
      <key>Display Name</key><string>All</string>
     <key>Database Configuration Parameter Settings</key>
     <dict>
       <key>Display Name</key>
       <string>Database Configuration Parameter Settings</string>
       <key>avg_appls</key>
       <dict>
         <key>Display Name</key><string>avg_appls</string>
         <key>Parameter Value</key>
         <dict>
           <key>Display Name</key><string>Parameter Value</string>
```

```
    <key>Value</key><string>2</string>
   </dict>
  <key>Value Flags</key>
  <dict>
    <key>Display Name</key><string>Value Flags</string>
    <key>Value</key><string>AUTOMATIC</string>
  </dict>
  <key>Parameter Update Status</key>
  <dict>
    <key>Display Name</key><string>Parameter Update Status</string>
    <key>Update Coverage</key>
    <dict>
      <key>Display Name</key><string>Update Coverage</string>
      <key>Value</key><string>Complete</string>
    </dict>
    <key>SQLCODE</key>
    <dict>
      <key>Display Name</key><string>SQLCODE</string>
      <key>Value</key><integer>0</integer>
    </dict>
    <key>Message Tokens</key>
    <dict>
      <key>Display Name</key><string>Message Tokens</string>
      <key>Value</key><array><string></string> </array>
    </dict>
    <key>SQLSTATE</key>
    <dict>
      <key>Display Name</key><string>SQLSTATE</string>
      <key>Value</key><string></string>
    </dict>
  </dict>
</dict>
<key>database_memory</key>
<dict>
  <key>Display Name</key><string>database_memory</string>
  <key>Deferred Value</key>
  <dict>
    <key>Display Name</key><string>Deferred Value</string>
    <key>Value</key><string>2</string>
  </dict>
  <key>Deferred Value Flags</key>
  <dict>
    <key>Display Name</key><string>Deferred Value Flags</string>
    <key>Value</key><string>MANUAL</string>
  </dict>
  <key>Parameter Update Status</key>
  <dict>
    <key>Display Name</key><string>Parameter Update Status</string>
    <key>Update Coverage</key>
    <dict>
      <key>Display Name</key><string>Update Coverage</string>
      <key>Value</key><string>Complete</string>
    </dict>
    <key>SQLCODE</key>
    <dict>
      <key>Display Name</key><string>SQLCODE</string>
      <key>Value</key><integer>0</integer>
    </dict>
    <key>Message Tokens</key>
    <dict>
      <key>Display Name</key><string>Message Tokens</string>
      <key>Value</key><array><string></string></array>
    </dict>
    <key>SQLSTATE</key>
    <dict>
      <key>Display Name</key><string>SQLSTATE</string>
      <key>Value</key><string></string>
```

```
          </dict>
         </dict>
       </dict>
     </dict>
   </dict>
</dict>
</plist>
```

例 3: フィルターを指定して、特定の構成パラメーターの値を戻します。

```
db2 "call sysproc.set_config(1,0, 'en_US', blob('
<plist version="1.0">
<dict>
   <key>Document Type Name</key><string>Data Server Set Configuration Input</string>
   <key>Document Type Major Version</key><integer>1</integer>
   <key>Document Type Minor Version</key><integer>0</integer>
   <key>Document Locale</key><string>en_US</string>
   <key>Database Manager Configuration Parameter Settings</key>
   <dict>
      <key>diaglevel</key>
      <dict>
         <key>Parameter Value</key>
         <dict>
            <key>Value</key><string>4</string>
         </dict>
      </dict>
      <key>fcm_num_buffers</key>
      <dict>
         <key>Parameter Value</key>
         <dict>
            <key>Value</key><string>4096</string>
         </dict>
        <key>Value Flags</key>
        <dict>
           <key>Value</key><string>MANUAL</string>
        </dict>
      </dict>
      <key>instance_memory</key>
      <dict>
         <key>Deferred Value</key>
         <dict>
            <key>Value</key><string>7424</string>
         </dict>
         <key>Deferred Value Flags</key>
         <dict>
            <key>Value</key><string>AUTOMATIC</string>
         </dict>
      </dict>
   </dict>
   <key>Database Partition</key>
   <dict>
      <key>All</key>
      <dict>
         <key>Database Configuration Parameter Settings</key>
         <dict>
            <key>avg_appls</key>
            <dict>
               <key>Parameter Value</key>
               <dict>
                  <key>Value></key><string>2</string>
               </dict>
               <key>Value Flags</key>
               <dict>
                  <key>Value</key><string>AUTOMATIC</string>
               </dict>
            </dict>
```

```
            <key>database_memory</key>
            <dict>
               <key>Deferred Value</key>
               <dict>
                  <key>Value</key><string>2</string>
               </dict>
               <key>Deferred Value Flags</key>
               <dict>
                  <key>Value</key><string>MANUAL</string>
               </dict>
            </dict>
         </dict>
      </dict>
   </dict>
</dict>
</plist>'),
blob('/plist/dict/key[.="Database Manager Configuration Parameter Settings"]
/following-sibling::dict[1]/key[3]
/following-sibling::dict[1]/dict[1]/key[.="Value"]
/following-sibling::string[1]'),?,?)"
```

以下はこの照会の出力例です。

```
Value of output parameters
--------------------------
Parameter Name : MAJOR_VERSION
Parameter Value : 1

Parameter Name : MINOR_VERSION
Parameter Value : 0

Parameter Name : XML_OUTPUT
Parameter Value : x'34303936'

Parameter Name : XML_MESSAGE
Parameter Value : -

Return Status = 0
```

xml_output に次の値が戻されます。

```
4096
```

例 4: 関数からプロシージャーを呼び出します。

```
  EXEC SQL BEGIN DECLARE SECTION;
  sqlint16  getconfigMaj;
  sqlint16  getconfigMin;

  SQL TYPE IS BLOB(2M) xmlOutput;
  SQL TYPE IS BLOB(2K) xmlOutMessage;
  EXEC SQL END DECLARE SECTION;
   getconfigMaj = 1;
   getconfigMin = 0;

  EXEC SQL CALL SYSPROC.SET_CONFIG(
                :getconfigMaj,
                :getconfigMin,
                'en_US',
                BLOB('blob('
                <?xml version="1.0" encoding="UTF-8"?>
                <plist version="1.0">
                <dict>
                  <key>Document Type Name</key>
                  <string>Data Server Set Configuration Input</string>
                  <key>Document Type Major Version</key><integer>1</integer>
                  <key>Document Type Minor Version</key><integer>0</integer>
```

```
  <key>Document Locale</key><string>en_US</string>
  <key>Database Manager Configuration Parameter Settings</key>
  <dict>
    <key>diaglevel</key><dict><key>Parameter Value</key>
    <dict>
      <key>Value</key><string>4</string>
    </dict>
  </dict>
  <key>fcm_num_buffers</key>
  <dict>
    <key>Parameter Value</key>
    <dict>
      <key>Value</key><string>4096</string>
    </dict>
    <key>Value Flags</key>
    <dict>
      <key>Value</key><string>MANUAL</string>
    </dict>
  </dict>
  <key>instance_memory</key>
  <dict>
    <key>Deferred Value</key>
    <dict>
      <key>Value</key><string>7424</string>
    </dict>
    <key>Deferred Value Flags</key>
    <dict>
        <key>Value</key><string>AUTOMATIC</string>
    </dict>
  </dict>
  </dict>
  <key>Database Partition</key>
  <dict>
    <key>All</key>
    <dict>
      <key>Database Configuration Parameter Settings</key>
      <dict>
        <key>avg_appls</key>
        <dict>
          <key>Parameter Value</key>
          <dict>
            <key>Value></key><string>2</string>
          </dict>
          <key>Value Flags</key>
          <dict>
            <key>Value</key><string>AUTOMATIC</string>
          </dict>
        </dict>
        <key>database_memory</key>
        <dict>
          <key>Deferred Value</key>
          <dict>
            <key>Value</key><string>2</string>
          </dict>
          <key>Deferred Value Flags</key>
          <dict>
            <key>Value</key><string>MANUAL</string>
          </dict>
        </dict>
      </dict>
    </dict>
  </dict>
</dict>
</plist>'),
null,
:xmlOutput,
:xmlOutMessage );
```

第 9 章 構成ルーチンおよびビュー

DB_PARTITIONS

DB_PARTITIONS 表関数は、表形式の db2nodes.cfg ファイルの内容を戻します。

構文

スキーマは SYSPROC です。

許可

DB_PARTITIONS 表関数に対する EXECUTE 特権。

表関数パラメーター

関数には入力パラメーターはありません。

例

3 つのロジカル・パーティションを持つデータベースから情報を取り出します。

```
SELECT * FROM TABLE(DB_PARTITIONS()) AS T
```

以下はこの照会の出力例です。

```
PARTITION_NUMBER HOST_NAME                 PORT_NUMBER SWITCH_NAME
---------------- ---------------------...- ----------- -----------
               0 jessicae.torolab.ibm.com           0 jessicae
               1 jessicae.torolab.ibm.com           1 jessicae
               2 jessicae.torolab.ibm.com           2 jessicae

  3 record(s) selected.
```

戻される情報

表 84. DB_PARTITIONS 表関数によって戻される情報

列名	データ・タイプ	説明
PARTITION_NUMBER	SMALLINT	パーティション・データベース環境内のデータベース・パーティション・サーバーを識別する 0 から 999 までの固有番号。
HOST_NAME	VARCHAR(128)	データベース・パーティション・サーバーの TCP/IP ホスト名。
PORT_NUMBER	SMALLINT	データベース・パーティション・サーバーのポート番号。

© Copyright IBM Corp. 2006, 2010

表 84. DB_PARTITIONS 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明
SWITCH_NAME	VARCHAR(128)	データベース・パーティション通信用の高速相互接続 (スイッチ) の名前。

DBCFG 管理ビュー - データベース構成パラメーター情報の検索

DBCFG 管理ビューは、現在接続中のデータベースのすべてのデータベース・パーティションに関する、データベース構成パラメーター情報を検索します。

スキーマは SYSIBMADM です。

許可

以下のいずれかの権限が必要です。

- DBCFG 管理ビューに対する SELECT 特権
- DBCFG 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- DB_GET_CFG 表関数に対する EXECUTE 特権
- DATAACCESS 権限

例

例 1: メモリーに保管されている、すべてのデータベース・パーティションに関するデータベース構成内の自動保守の設定を検索します。

```
SELECT DBPARTITIONNUM, NAME, VALUE FROM SYSIBMADM.DBCFG WHERE NAME LIKE 'auto_%'
```

以下はこの照会の出力例です。

```
DBPARTITIONNUM NAME                            VALUE
-------------- ------------------------------- --------------
             0 auto_maint                      OFF
             0 auto_db_backup                  OFF
             0 auto_tbl_maint                  OFF
             0 auto_runstats                   OFF
             0 auto_stats_prof                 OFF
             0 auto_prof_upd                   OFF
             0 auto_reorg                      OFF
             0 autorestart                     ON

  8 record(s) selected.
```

例 2: ディスクに保管されている、すべてのデータベース・パーティションのデータベース構成パラメーター値すべてを検索します。

```
SELECT NAME, DEFERRED_VALUE, DBPARTITIONNUM FROM SYSIBMADM.DBCFG
```

以下はこの照会の出力例です。

```
NAME                 DEFERRED_VALUE      DBPARTITIONNUM
-----------------...- ---------------...- --------------
app_ctl_heap_sz      128                              0
appgroup_mem_sz      30000                            0
applheapsz           256                              0
archretrydelay       20                               0
...
autorestart          ON                               0
avg_appls            1                                0
blk_log_dsk_ful      NO                               0
catalogcache_sz      -1                               0
...
```

戻される情報

表 85. DBCFG 管理ビューによって戻される情報

列名	データ・タイプ	説明
NAME	VARCHAR(32)	構成パラメーター名。
VALUE	VARCHAR(1024)	メモリー内に保管されている構成パラメーターの現行値。
VALUE_FLAGS	VARCHAR(10)	構成パラメーターの現行値に固有の情報を指定します。有効な値は以下のとおりです。 • NONE - 追加情報なし • AUTOMATIC - 構成パラメーターが自動に設定されている
DEFERRED_VALUE	VARCHAR(1024)	ディスク上の構成パラメーター値。データベース構成パラメーターによっては、データベースを再活動化しないと変更が有効にならないことがあります。その場合に、すべてのアプリケーションをまずデータベースから切断する必要があります。(データベースがアクティブ化されていた場合には、非アクティブにしてから再アクティブ化しなければなりません。)変更は次にデータベースに接続したときに有効になります。
DEFERRED_VALUE_FLAGS	VARCHAR(10)	構成パラメーターの据え置き値に固有の情報を指定します。有効な値は以下のとおりです。 • NONE - 追加情報なし • AUTOMATIC - 構成パラメーターが自動に設定されている
DATATYPE	VARCHAR(128)	構成パラメーター・データ・タイプ。

表 85. DBCFG 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明
DBPARTITIONNUM	SMALLINT	データベース・パーティション番号。

DBMCFG 管理ビュー - データベース・マネージャー構成パラメーター情報の検索

DBMCFG 管理ビューは、メモリー内の値およびディスクに保管された値を含む、データベース・マネージャー構成パラメーター情報を検索します。

スキーマは SYSIBMADM です。

許可

以下のいずれかの権限が必要です。

- DBMCFG 管理ビューに対する SELECT 特権
- DBMCFG 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- DBM_GET_CFG 表関数に対する EXECUTE 特権
- DATAACCESS 権限

例

例 1: ディスクに保管されているすべてのデータベース・マネージャー構成パラメーターの値を検索します。

```
SELECT NAME, DEFERRED_VALUE FROM SYSIBMADM.DBMCFG
```

以下はこの照会の出力例です。

```
NAME                            DEFERRED_VALUE
------------------------------- -------------------...----
agent_stack_sz                  0
agentpri                        -1
alternate_auth_enc              AES_ONLY
aslheapsz                       15
audit_buf_sz                    0
authentication                  SERVER
catalog_noauth                  YES
clnt_krb_plugin
...
comm_bandwidth                  0.000000e+00
conn_elapse                     0
cpuspeed                        4.000000e-05
dft_account_str
dft_mon_bufpool                 OFF
...
dft_mon_timestamp               ON
dft_mon_uow                     OFF
...
jdk_path                        /wsdb/v91/bldsupp/AIX5L
...
```

```
ssl_svcename                    22711
ssl_svr_keydb                   /GSKit/Keystore/key.kdb
ssl_svr_label
ssl_svr_stash                   /GSKit/Keystore/key.sth
```

例 2: データベース・マネージャー構成パラメーター値すべてを検索します。

```
SELECT * FROM SYSIBMADM.DBMCFG
```

以下はこの照会の出力例です。

```
NAME                VALUE           VALUE_FLAGS ...
------------...------------...- ------------ ...
agent_stack_sz      0               NONE        ...
agentpri            -1              NONE        ...
alternate_auth_enc  NOT_SPECIFIED   NONE
aslheapsz           15              NONE        ...
audit_buf_sz        0               NONE        ...
authentication      SERVER          NONE        ...
catalog_noauth      YES             NONE        ...
clnt_krb_plugin                     NONE        ...
clnt_pw_plugin                      NONE        ...
comm_bandwidth      0.000000e+00    NONE        ...
conn_elapse         0               NONE        ...
cpuspeed            4.000000e-05    NONE        ...
dft_account_str                     NONE        ...
dft_mon_bufpool     OFF             NONE        ...
dft_mon_lock        OFF             NONE        ...
dft_mon_sort        OFF             NONE        ...
dft_mon_stmt        OFF             NONE        ...
dft_mon_table       OFF             NONE        ...
...                                           ...
dir_cache           YES             NONE        ...
discover            SEARCH          NONE        ...
discover_inst       ENABLE          NONE        ...
fcm_num_anchors     0               AUTOMATIC   ...
fcm_num_buffers     0               AUTOMATIC   ...
fcm_num_connect     0               AUTOMATIC   ...
```

この照会の出力 (続き)。

```
... DEFERRED_VALUE     DEFERRED_VALUE_FLAGS      DATATYPE
... -------------...- ------------------------ ---------...-
... 0                  NONE                      INTEGER
... -1                 NONE                      INTEGER
... AES_ONLY           NONE                      VARCHAR(32)
... 15                 NONE                      BIGINT
... 0                  NONE                      BIGINT
... SERVER             NONE                      VARCHAR(32)
... YES                NONE                      VARCHAR(3)
...                    NONE                      VARCHAR(32)
...                    NONE                      VARCHAR(32)
... 0.000000e+00       NONE                      REAL
... 0                  NONE                      INTEGER
... 4.000000e-05       NONE                      REAL
...                    NONE                      VARCHAR(25)
... OFF                NONE                      VARCHAR(3)
... OFF                NONE                      VARCHAR(3)
... OFF                NONE                      VARCHAR(3)
... OFF                NONE                      VARCHAR(3)
... OFF                NONE                      VARCHAR(3)
...
... YES                NONE                      VARCHAR(3)
... SEARCH             NONE                      VARCHAR(8)
... ENABLE             NONE                      VARCHAR(8)
```

```
... 0                AUTOMATIC               BIGINT
... 512              AUTOMATIC               BIGINT
... 0                AUTOMATIC               BIGINT
...
```

戻される情報

表 86. DBMCFG 管理ビューによって戻される情報

列名	データ・タイプ	説明
NAME	VARCHAR(32)	構成パラメーター名。
VALUE	VARCHAR(256)	メモリー内に保管されている構成パラメーターの現行値。
VALUE_FLAGS	VARCHAR(10)	構成パラメーターの現行値に固有の情報を指定します。有効な値は以下のとおりです。 • NONE - 追加情報なし • AUTOMATIC - 構成パラメーターが自動に設定されている
DEFERRED_VALUE	VARCHAR(256)	ディスク上の構成パラメーター値。データベース・マネージャー構成パラメーターによっては、データベース・マネージャーを停止 (db2stop) してから再始動 (db2start) しないと、この値が有効になりません。
DEFERRED_VALUE_FLAGS	VARCHAR(10)	構成パラメーターの据え置き値に固有の情報を指定します。有効な値は以下のとおりです。 • NONE - 追加情報なし • AUTOMATIC - 構成パラメーターが自動に設定されている
DATATYPE	VARCHAR(128)	構成パラメーター・データ・タイプ。

REG_VARIABLES 管理ビュー - 使用中のDB2 レジストリー設定の検索

REG_VARIABLES 管理ビューは、すべてのデータベース・パーティションから DB2 レジストリー設定値を戻します。インスタンス開始後に db2set コマンドを使用して構成された DB2 レジストリー変数がある場合、REG_VARIABLES 管理ビューを照会して戻される DB2 レジストリー変数値と、db2set コマンドによって戻される値が異なることがあります。REG_VARIABLES はインスタンスの開始時に有効だった値を戻すに過ぎないため、この違いが生じます。

スキーマは SYSIBMADM です。

許可

以下のいずれかの権限が必要です。

- REG_VARIABLES 管理ビューに対する SELECT 特権
- REG_VARIABLES 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- REG_LIST_VARIABLES 表関数に対する EXECUTE 特権
- DATAACCESS 権限

例

現在使用されている DB2 レジストリー設定を要求します。

```
SELECT * from SYSIBMADM.REG_VARIABLES
```

以下はこの照会の出力例です。

```
DBPARTITIONNUM REG_VAR_NAME        REG_VAR_VALUE     IS_AGGREGATE AGGREGATE_NAME
-------------- ---------------...- -------------...- ------------ --------------...-
             0 DB2ADMINSERVER     DB2DAS00                     0 -
             0 DB2INSTPROF        D:¥SQLLIB                    0 -
             0 DB2PATH            D:¥SQLLIB                    0 -
             0 DB2SYSTEM          D570                         0 -
             0 DB2TEMPDIR         D:¥SQLLIB¥                   0 -
             0 DB2_EXTSECURITY    YES                          0 -

  6 record(s) selected.
```

戻される情報

表 87. REG_VARIABLES 管理ビューによって戻される情報

列名	データ・タイプ	説明
DBPARTITIONNUM	SMALLINT	関数が操作する各データベース・パーティションのロジカル・パーティションの数。
REG_VAR_NAME	VARCHAR(256)	DB2 レジストリー変数の名前。
REG_VAR_VALUE	VARCHAR(2048)	DB2 レジストリー変数の現行設定。
IS_AGGREGATE	SMALLINT	DB2 レジストリー変数が集約変数であるかどうかを示します。考えられる戻り値は、集約変数でない場合は 0、集約変数である場合は 1 です。

表 87. *REG_VARIABLES* 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明
AGGREGATE_NAME	VARCHAR(256)	現在 DB2 レジストリー変数が、構成済みの集約から値を取得している場合には、集約の名前。レジストリー変数が集約から設定されているのではない場合、または集約から設定されたもののオーバーライドされている場合、AGGREGATE_NAME の値は NULL になります。
LEVEL	CHAR(1)	DB2 レジストリー変数がその値を獲得するレベルを示します。考えられる戻り値と、それが表す対応するレベルは以下のとおりです。 • I = インスタンス • G = グローバル • N = データベース・パーティション • E = 環境

第 10 章 環境ビュー

ENV_FEATURE_INFO 管理ビュー - DB2 フィーチャーのライセンス情報を戻す

ENV_FEATURE_INFO 管理ビューは、ライセンスが必要とされる使用可能なすべてのフィーチャーに関する情報を戻します。フィーチャーごとに、そのフィーチャーの有効なライセンスがインストールされているかどうかに関する情報があります。

スキーマは SYSIBMADM です。

許可

以下のいずれかの権限が必要です。

- ENV_FEATURE_INFO 管理ビューに対する SELECT 特権
- ENV_FEATURE_INFO 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- ENV_GET_FEATURE_INFO 表関数に対する EXECUTE 特権
- DATAACCESS 権限

例

インストール済みの DB2 フィーチャー・ライセンス情報の要求

```
SELECT * FROM SYSIBMADM.ENV_FEATURE_INFO
```

以下はこの照会の出力例です。

```
FEATURE_NAME  FEATURE_FULLNAME                                ...
------------  ----------------------------------------------  ...
DPF           DB2_DATABASE_PARTITIONING_FEATURE               ...
POESE         DB2_PERFORMANCE_OPTIMIZATION_FEATURE_FOR_ESE    ...
SO            DB2_STORAGE_OPTIMIZATION_FEATURE                ...
AAC           DB2_ADVANCED_ACCESS_CONTROL_FEATURE             ...
GEO           DB2_GEODETIC_DATA_MANAGEMENT_FEATURE            ...
HFESE         IBM_HOMOGENEOUS_FEDERATION_FEATURE_FOR_ESE      ...
XMLESE        DB2_PUREXML_FEATURE_FOR_ESE                     ...
```

この照会からの出力 (続き)。

```
... LICENSE_INSTALLED  PRODUCT_NAME  FEATURE_USE_STATUS
... ----------------   ------------  -----------------
... Y                  ESE           IN_COMPLIANCE
... Y                  ESE           IN_COMPLIANCE
... Y                  ESE           IN_COMPLIANCE
... Y                  ESE           NOT_USED
... Y                  ESE           NOT_USED
... Y                  ESE           NOT_USED
... N                  ESE           IN_VIOLATION
```

© Copyright IBM Corp. 2006, 2010

ENV_FEATURE_INFO 管理ビューのメタデータ

表 88. ENV_FEATURE_INFO 管理ビューのメタデータ

列名	データ・タイプ	説明
FEATURE_NAME	VARCHAR(26)	ライセンスがある DB2 サーバー上で使用可能な DB2 フィーチャーの短縮名。
FEATURE_FULLNAME	VARCHAR(100)	DB2 フィーチャーのフルネーム。列値は英語の大文字で表示されます。語はスペース文字ではなく下線文字で区切られます。
LICENSE_INSTALLED	CHAR(1)	フィーチャーにライセンスがあるかどうかを示します。値が 'N' の場合、フィーチャーにはライセンスがありません。値が 'Y' の場合、フィーチャーにはライセンスがあります。
PRODUCT_NAME	VARCHAR(26)	フィーチャーが使用可能な DB2 サーバー製品の ID。使用できる戻り値は次のとおりです。 • ESE - DB2 Enterprise Server Edition • WSE - DB2 Workgroup Server Edition • EXP - DB2 Express® Edition
FEATURE_USE_STATUS	VARCHAR(30)	ライセンス準拠状況を示します。この値はフィーチャーの使用状況を示します。使用できる値は次の 3 つです。 • IN_COMPLIANCE: フィーチャーは少なくとも 1 回使用されており、フィーチャーの有効なライセンスがあります。 • IN_VIOLATION: フィーチャーは少なくとも 1 回使用されていますが、フィーチャーの有効なライセンスがありません。 • NOT_USED: フィーチャーは使用されていません。

ENV_INST_INFO 管理ビュー - 現在のインスタンスに関する情報の検索

ENV_INST_INFO 管理ビューは、現在のインスタンスについての情報を戻します。

スキーマは SYSIBMADM です。

許可

以下のいずれかの権限が必要です。

- ENV_INST_INFO 管理ビューに対する SELECT 特権
- ENV_INST_INFO 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- ENV_GET_INST_INFO 表関数に対する EXECUTE 特権
- DATAACCESS 権限

例

現在のインスタンスについての情報を要求します。

```
SELECT * FROM SYSIBMADM.ENV_INST_INFO
```

以下はこの照会の出力例です。

```
INST_NAME          IS_INST_PARTITIONABLE NUM_DBPARTITIONS INST_PTR_SIZE ...
-----------...---- --------------------- ---------------- ------------- ...
DB2                                    0                1            32 ...

  1 record(s) selected.
```

この照会の出力 (続き)。

```
... RELEASE_NUM       SERVICE_LEVEL        BLD_LEVEL        PTF        FIXPACK_NUM
... -----------...--- --------------...--- ----------...--- ----...--- -----------
... 01010107          DB2 v9.1.0.115       n051106                              0
```

戻される情報

表 89. ENV_INST_INFO 管理ビューによって戻される情報

列名	データ・タイプ	説明
INST_NAME	VARCHAR(128)	現在のインスタンスの名前。
IS_INST_PARTITIONABLE	SMALLINT	現在のインスタンスがパーティション化が可能なデータベース・サーバー・インスタンスであるかどうかを示します。考えられる戻り値は、パーティション化可能データベース・サーバー・インスタンスでない場合は 0、パーティション化可能データベース・サーバー・インスタンスである場合は 1 です。
NUM_DBPARTITIONS	INTEGER	データベース・パーティションの数。パーティション・データベース環境ではない場合、値 1 を戻します。
INST_PTR_SIZE	INTEGER	現在のインスタンスのビット・サイズ (32 または 64)。
RELEASE_NUM	VARCHAR(128)	db2level コマンドによって戻される、内部リリース番号。例えば 03030106 など。
SERVICE_LEVEL	VARCHAR(128)	db2level コマンドによって戻される、サービス・レベル。例えば DB2 v8.1.1.80 など。
BLD_LEVEL	VARCHAR(128)	db2level コマンドによって戻される、ビルド・レベル。例えば n041021 など。
PTF	VARCHAR(128)	db2level コマンドによって戻される、プログラム一時修正 (PTF) ID。例えば U498350 など。
FIXPACK_NUM	INTEGER	db2level コマンドによって戻される、フィックスパック番号。例えば 9 など。

ENV_PROD_INFO 管理ビュー - インストール済みの DB2 製品に関する情報の検索

ENV_PROD_INFO 管理ビューは、インストール済みの DB2 製品についての情報を戻します。

スキーマは SYSIBMADM です。

許可

以下のいずれかの権限が必要です。

- ENV_PROD_INFO 管理ビューに対する SELECT 特権
- ENV_PROD_INFO 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- ENV_GET_PROD_INFO_V95 表関数に対する EXECUTE 特権
- DATAACCESS 権限

例

インストール済みの DB2 製品の情報を要求します。

```
SELECT * FROM SYSIBMADM.ENV_PROD_INFO
```

以下はこの照会の出力例です。

```
INSTALLED_PROD  INSTALLED_PROD_FULLNAME        ...
--------------  ------------------------------ ...
ESE             DB2_ENTERPRISE_SERVER_EDITION  ...
WSE             DB2_WORKGROUP_SERVER_EDITION   ...
EXP             DB2_EXPRESS_EDITION            ...
```

この照会からの出力 (続き)。

```
... LICENSE_INSTALLED  PROD_RELEASE  LICENSE_TYPE
... ----------------  ------------  ---------------------
... Y                 9.5           AUTHORIZED_USER_OPTION
... N                 9.5           LICENSE_NOT_REGISTERED
... Y                 9.5           RESTRICTED
```

ENV_PROD_INFO 管理ビューのメタデータ

表 90. ENV_PROD_INFO 管理ビューのメタデータ

列名	データ・タイプ	説明
INSTALLED_PROD	VARCHAR(26)	システムにインストール済みの DB2 製品の ID。
INSTALLED_PROD_FULLNAME	VARCHAR(100)	インストール済みの DB2 製品のフルネーム。列値は英語の大文字で表示されます。語は下線文字で区切られます。
LICENSE_INSTALLED	CHAR(1)	製品にライセンスがあるかどうかを示します。値が N の場合、製品にはライセンスがありません。値が Y の場合、製品にはライセンスがあります。

表 90. ENV_PROD_INFO 管理ビューのメタデータ (続き)

列名	データ・タイプ	説明
PROD_RELEASE	VARCHAR(26)	製品リリース番号。
LICENSE_TYPE	VARCHAR(50)	製品についてインストールされるライセンスのタイプの名前。使用できる戻り値は次のとおりです。 • 12_MONTHS_LICENSE_AND_SUBSCRIPTION • AUTHORIZED_USER • AUTHORIZED_USER_OPTION • CLIENT_DEVICE • CPU • CPU_OPTION • HOST_SERVER_AND_MSU • LICENSE_NOT_REGISTERED • MANAGED_PROCESSOR • N/A • RESTRICTED • TRIAL • UNWARRANTED • USER

ENV_SYS_INFO 管理ビュー - システムに関する情報の検索

ENV_SYS_INFO 管理ビューは、システムについての情報を戻します。

スキーマは SYSIBMADM です。

許可

以下のいずれかの権限が必要です。

- ENV_SYS_INFO 管理ビューに対する SELECT 特権
- ENV_SYS_INFO 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- ENV_GET_SYS_INFO 表関数に対する EXECUTE 特権
- DATAACCESS 権限

例

システムについての情報を要求します。

```
SELECT * from SYSIBMADM.ENV_SYS_INFO
```

以下はこの照会の出力例です。

```
OS_NAME      OS_VERSION     OS_RELEASE         HOST_NAME
--------...- ----------...- ---------------...- ---------...-
WIN32_NT     5.1            Service Pack 1     D570

  1 record(s) selected.
```

この照会からの出力 (続き)。

```
... TOTAL_CPUS  CONFIGURED_CPUS TOTAL_MEMORY
... ----------- --------------- ------------
...           1               2         1527
```

戻される情報

表 91. ENV_SYS_INFO 管理ビューによって戻される情報

列名	データ・タイプ	説明
OS_NAME	VARCHAR(256)	オペレーティング・システムの名前。
OS_VERSION	VARCHAR(256)	オペレーティング・システムのバージョン番号。
OS_RELEASE	VARCHAR(256)	オペレーティング・システムのリリース番号。
HOST_NAME	VARCHAR(256)	システムの名前。
TOTAL_CPUS	INTEGER	システム上の物理 CPU の総数。
CONFIGURED_CPUS	INTEGER	システム上の構成済み物理 CPU の数。
TOTAL_MEMORY	INTEGER	システム上のメモリーの合計量 (MB 単位)。

ENV_SYS_RESOURCES 管理ビュー - システム情報を戻す

ENV_SYS_RESOURCES 管理ビューは、オペレーティング・システム、CPU、メモリー、およびその他のシステム関連情報を戻します。

スキーマは SYSIBMADM です。

許可

以下のいずれかの権限が必要です。

- ENV_SYS_RESOURCES 管理ビューに対する SELECT 特権
- ENV_SYS_RESOURCES 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- ENV_GET_SYS_RESOURCES 表関数に対する EXECUTE 特権
- DATAACCESS 権限

例

```
SELECT SUBSTR(NAME,1,20) AS NAME, SUBSTR(VALUE,1,10) AS VALUE,
  SUBSTR(DATATYPE,1,10) AS DATATYPE, DBPARTITIONNUM
FROM SYSIBMADM.ENV_SYS_RESOURCES
WHERE SUBSTR(NAME,1,8)='CPU_LOAD' OR NAME='CPU_USAGE_TOTAL'
```

以下はこの照会の出力例です。

```
NAME                 VALUE      DATATYPE   DBPARTITIONNUM
-------------------- ---------- ---------- --------------
CPU_LOAD_SHORT       0.044052   DECIMAL                 0
CPU_LOAD_MEDIUM      0.087250   DECIMAL                 0
CPU_LOAD_LONG        0.142059   DECIMAL                 0
CPU_USAGE_TOTAL      7          SMALLINT                0

  4 record(s) selected.
```

ENV_SYS_RESOURCES 管理ビューのメタデータ

表 92. ENV_SYS_RESOURCES 管理ビューのメタデータ

列名	データ・タイプ	説明
NAME	VARCHAR(128)	属性の名前。考えられる値については、 370 ページの表 93 を参照してください。 **注:** サーバーでのオペレーティング・システムおよびハードウェア構成によっては、一部の属性が使用できない場合があります。
VALUE	VARCHAR(1024)	属性の値。
DATATYPE	VARCHAR(128)	属性データ・タイプ。
UNIT	VARCHAR(128)	VALUE 列に使用される単位 (適用される場合)。適用外の場合は NULL が戻されます。
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

表 93. NAME 列に使用できる値

情報タイプ	名前	データ・タイプ	説明	この情報を戻すプラットフォーム	UNIT
オペレーティング・システム	OS_NAME	VARCHAR(256)	オペレーティング・システム・ソフトウェアの名前。	すべて	NULL
	HOST_NAME	VARCHAR(256)	システムのホスト名。	すべて	NULL
	OS_VERSION	VARCHAR(256)	オペレーティング・システムのバージョン。例えば、AIX: 4.3 version = 4 のようになります。	すべて	NULL
	OS_RELEASE	VARCHAR(256)	オペレーティング・システムのリリース。例えば、AIX: 4.3 release = 3 のようになります。	すべて	NULL
	MACHINE_IDENTIFICATION	VARCHAR(256)	マシン・ハードウェア ID。	すべて	NULL
	OS_LEVEL	VARCHAR(256)	現行バージョンおよびリリースの保守レベル。例えば、LINUX: 2.4.9, level = 9 のようになります。	Linux	NULL

表 93. NAME 列に使用できる値 (続き)

情報タイプ	名前	データ・タイプ	説明	この情報を戻すプラットフォーム	UNIT
CPU	CPU_TOTAL	BIGINT	CPU の総数。	すべて	NULL
	CPU_ONLINE	BIGINT	オンラインの CPU 数。	すべて	NULL
	CPU_CONFIGURED	BIGINT	構成済みの CPU 数。	すべて	NULL
	CPU_SPEED	BIGINT	CPU の速度。	すべて	MHz
	CPU_TIMEBASE	BIGINT	時間基準のレジスター増分の周波数。	Linux PowerPC®	Hz
	CPU_HMT_DEGREE	BIGINT	ハードウェア・マルチスレッド化 (HMT) をサポートするシステムでは、これは物理プロセッサーがオペレーティング・システムに存在すると見なすプロセッサーの数です。非 HMT システムでは、この値は 1 です。HMT システムでは、「合計」は論理 CPU の数を反映します。物理 CPU の数を取得するには、「合計」を「threadingDegree」で除算します。	すべて	NULL
	CPU_CORES_PER_SOCKET	BIGINT	ソケットごとの CPU コアの数。単一コア・システムでは、この値は 1 です。	すべて	NULL
物理メモリー	MEMORY_TOTAL	BIGINT	物理メモリーの合計サイズ。	すべて	MB
	MEMORY_FREE	BIGINT	空き物理メモリーの量。	すべて	MB
	MEMORY_SWAP_TOTAL	BIGINT	スワップ・スペースの合計量。	すべて	MB
	MEMORY_SWAP_FREE	BIGINT	空きスワップ・スペースの量。	すべて	MB

表 93. NAME 列に使用できる値 (続き)

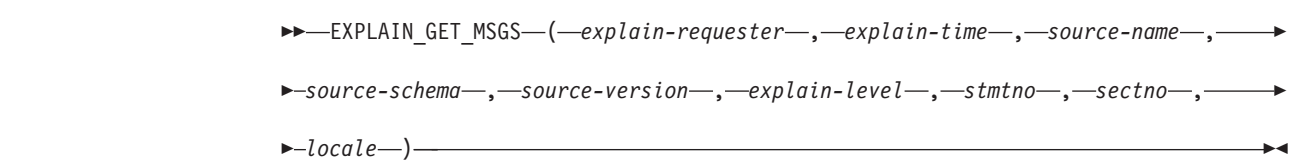
情報タイプ	名前	データ・タイプ	説明	この情報を戻すプラットフォーム	UNIT
仮想メモリー	VIRTUAL_MEM_TOTAL	BIGINT	システム上の仮想メモリーの合計量。	すべて	MB
	VIRTUAL_MEM_RESERVED	BIGINT	予約済み仮想メモリーの量。	すべて	MB
	VIRTUAL_MEM_FREE	BIGINT	空き仮想メモリーの量。	すべて	MB
CPU ロード	CPU_LOAD_SHORT	DECIMAL	最短期間。例えば、最後の 5 分間でサンプルをロードします。	Windows オペレーティング・システムを除くすべて。	NULL
	CPU_LOAD_MEDIUM	DECIMAL	中間の期間。例えば、最後の 10 分間でサンプルをロードします。	Windows オペレーティング・システムを除くすべて。	NULL
	CPU_LOAD_LONG	DECIMAL	長期間。例えば、最後の 15 分間でサンプルをロードします。	Windows オペレーティング・システムを除くすべて。	NULL
	CPU_USAGE_TOTAL	DECIMAL	マシンの全 CPU 使用量のパーセント。	すべて	パーセント

第 11 章 Explain ルーチン

EXPLAIN_GET_MSGS

EXPLAIN_GET_MSGS 表関数は、EXPLAIN_DIAGNOSTIC および EXPLAIN_DIAGNOSTIC_DATA Explain 表を照会し、定様式メッセージを戻します。

構文



スキーマは Explain 表スキーマと同じです。

表関数パラメーター

以下の入力引数はいずれもヌルにすることができます。引数がヌルの場合、その引数は照会を制限するためには使用されません。

explain-requester
この Explain 要求のイニシエーターの許可 ID を指定する、タイプ VARCHAR(128) の入力引数。 NULL 値であれば、このパラメーターは照会の検索条件から除外されます。

explain-time
Explain 要求の開始時刻を指定する、タイプ TIMESTAMP の入力引数。 NULL 値であれば、このパラメーターは照会の検索条件から除外されます。

source-name
動的ステートメントの Explain 実行時に実行されるパッケージの名前、または静的 SQL ステートメントの Explain 実行時のソース・ファイルの名前を指定する、タイプ VARCHAR(128) の入力引数。 NULL 値であれば、このパラメーターは照会の検索条件から除外されます。

source-schema
Explain 要求のソースのスキーマ、または修飾子を指定する、タイプ VARCHAR(128) の入力引数。 NULL 値であれば、このパラメーターは照会の検索条件から除外されます。

source-version
Explain 要求のソースのバージョンを指定する、タイプ VARCHAR(64) の入力引数。 NULL 値であれば、このパラメーターは照会の検索条件から除外されます。

© Copyright IBM Corp. 2006, 2010

explain-level
この行が関係する Explain 情報のレベルを指定する、タイプ CHAR(1) 入力引数。 NULL 値であれば、このパラメーターは照会の検索条件から除外されます。

stmtno
この Explain 情報が関連付けられるパッケージ内のステートメント番号を指定する、タイプ INTEGER の入力引数。 NULL 値であれば、このパラメーターは照会の検索条件から除外されます。

sectno
この Explain 情報が関連付けられるパッケージ内のセクション番号を指定する、タイプ INTEGER の入力引数。 NULL 値であれば、このパラメーターは照会の検索条件から除外されます。

locale
戻されるメッセージのロケールを指定する、タイプ VARCHAR(33) の入力引数。指定したロケールが DB2 サーバーにインストールされていない場合、値は無視されます。

戻される情報

表 94. EXPLAIN_GET_MSGS 表関数によって戻される情報

列名	データ・タイプ	説明
EXPLAIN_REQUESTER	VARCHAR(128)	この Explain 要求のイニシエーターの許可 ID。
EXPLAIN_TIME	TIMESTAMP	Explain 要求の開始時刻。
SOURCE_NAME	VARCHAR(128)	動的ステートメントの Explain 実行時に実行されるパッケージの名前、または静的 SQL ステートメントの Explain 実行時のソース・ファイルの名前。
SOURCE_SCHEMA	VARCHAR(128)	Explain 要求のソースのスキーマまたは修飾子。
SOURCE_VERSION	VARCHAR(64)	Explain 要求のソースのバージョン。
EXPLAIN_LEVEL	CHAR(1)	この行に関連する Explain 情報のレベル。
STMTNO	INTEGER	この Explain 情報が関連付けられるパッケージ内のステートメント番号。
SECTNO	INTEGER	この Explain 情報が関連付けられるパッケージ内のセクション番号。
DIAGNOSTIC_ID	INTEGER	EXPLAIN_STATEMENT 表内の特定のステートメントのインスタンスの診断 ID。
LOCALE	VARCHAR(33)	戻されるメッセージのロケール。このロケールは、指定のロケールが DB2 サーバーにインストールされていない場合、指定のロケールとは一致しません。
MSG	VARCHAR(4096)	定様式メッセージ・テキスト。

例

過去 1 時間に生成された、リクエスター SIMMEN のデフォルト・スキーマの Explain 表から、定様式の英語のメッセージを要求します。 SQLC2E03 のソース名を指定します。

```
SELECT MSG
  FROM TABLE(EXPLAIN_GET_MSGS(
    'SIMMEN',
    CAST(NULL AS TIMESTAMP),
    'SQLC2E03',
    CAST(NULL AS VARCHAR(128)),
    CAST(NULL AS VARCHAR(64)),
    CAST(NULL AS CHAR(1)),
    CAST(NULL AS INTEGER),
    CAST(NULL AS INTEGER),
    'en_US'))
  AS REGISTRYINFO
  WHERE EXPLAIN_TIME >= (CURRENT TIMESTAMP - 1 HOUR)
  ORDER BY DIAGNOSTIC_ID
```

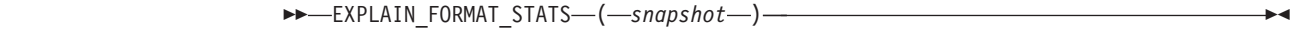
以下はこの照会の出力例です。

```
MSG
------------------------------------------------------------------------
EXP0012W  Invalid access request. The index "index1" could not be found.
  Line number "554", character number "20".
EXP0012W  Invalid access request. The index "index2" could not be found.
  Line number "573", character number "20".
EXP0015W  Invalid join request. Join refers to tables that are not in
  the same FROM clause. Line number "573", character number "20".
```

EXPLAIN_FORMAT_STATS

この新規のスカラー関数は、定様式の統計情報を表示するために使用されます。この情報は構文解析され、特定の照会についてキャプチャーされた Explain スナップショットから抽出されます。結果のデータ・タイプは CLOB(50M) です。

構文



スキーマは SYSPROC です。

関数のパラメーター

snapshot
指定された照会についてキャプチャーされた Explain スナップショットである、タイプ BLOB(10M) の入力引数。これは Explain 表 *EXPLAIN_STATEMENT* のスナップショット列として保管されます。

許可

EXPLAIN_FORMAT_STATS プロシージャーに対する EXECUTE 特権。

例

```
SELECT EXPLAIN_FORMAT_STATS(SNAPSHOT)
  FROM EXPLAIN_STATEMENT
  WHERE EXPLAIN_REQUESTER = 'DB2USER1' AND
      EXPLAIN_TIME = timestamp('2006-05-12-14.38.11.109432') AND
      SOURCE_NAME = 'SQLC2F0A' AND
      SOURCE_SCHEMA = 'NULLID' AND
      SOURCE_VERSION = '' AND
      EXPLAIN_LEVEL = 'O' AND
      STMTNO = 1 AND
      SECTNO = 201
```

以下はこの関数の出力例です。

```
Tablespace Context:
-------------------
Name:                                   USERSPACE1
Overhead:                               7.500000
Transfer Rate:                          0.060000
Prefetch Size:                          32
Extent Size:                            32
Type:                                   Database managed
Partition Group Name:                   NULLP
Buffer Pool Identifier:                 0

Base Table Statistics:
----------------------
Name:                                   T1
Schema:                                 DB2USER2
Number of Columns:                      3
Number of Pages with Rows:              1
Number of Pages:                        1
Number of Rows:                         5
Table Overflow Record Count:            0
Width of Rows:                          26
Time of Creation:                       2006-06-16-11.46.53.041085
Last Statistics Update:                 2006-06-26-12.23.44.814201
Statistics Type:                        Fabrication
Primary Tablespace:                     USERSPACE1
Tablespace for Indexes:                 USERSPACE1
Tablespace for Long Data:               NULLP
Number of Referenced Columns:           2
Number of Indexes:                      1
Volatile Table:                         No
Table Active Blocks:                    1
Number of Column Groups:                0
Number of Data Partitions:              1
Average Row Compression Ratio:          -9.000000
Percent Rows Compressed:                -9.000000
Average Compressed Row Size:            -9
Statistics Type:                        U

Column Information:
-------------------
Number:                                 1
Name:                                   C1
Statistics Available:                   Yes

Column Statistics:
------------------
Schema name of the column type:         SYSIBM
Name of column type:                    INTEGER
Maximum column length:                  4
Scale for decimal column:               0
Number of distinct column values:       4
Average column length:                  5
```

```
Number of most frequent values:          1
Number of quantiles:                     5
Second highest data value:               3
Second lowest data value:                2
Column sequence in partition key:        0
Average number of sub-elements:          -1
Average length of delimiters:            -1

Column Distribution Statistics:
-------------------------------
Frequency Statistics:
     Valcount   Value
     ---------- -----------
     2          1

Quantile Statistics:
     Valcount   Distcount   Value
     ---------- ----------- ----------
     0          1           1
     2          1           1
     3          2           2
     4          3           3
     5          4           4

Column Information:
-------------------
Number:                                  2
Name:                                    C2
Statistics Available:                    Yes

Column Statistics:
------------------
Schema name of the column type:          SYSIBM
Name of column type:                     INTEGER
Maximum column length:                   4
Scale for decimal column:                0
Number of distinct column values:        4
Average column length:                   5
Number of most frequent values:          1
Number of quantiles:                     5
Second highest data value:               3
Second lowest data value:                2
Column sequence in partition key:        0
Average number of sub-elements:          -1
Average length of delimiters:            -1

Column Distribution Statistics:
-------------------------------
Frequency Statistics:
      Valcount   Value
      ---------- -----------
      2          1

Quantile Statistics:
      Valcount   Distcount   Value
      ---------- ----------- -----------
      0          0           1
      2          0           1
      3          0           2
      4          0           4
      5          0           4

Indexes defined on the table:
-----------------------------
Name:                                    IDX_T1C1C2
Schema:                                  DB2USER2
Unique Rule:                             Duplicate index
```

```
Used in Operator:                       Yes
Page Fetch Pairs:                       Not Available
Number of Columns:                      2
Index Leaf Pages:                       1
Index Tree Levels:                      1
Index First Key Cardinality:            4
Index Full Key Cardinality:             4
Index Cluster Ratio:                    100
Index Cluster Factor:                   -1.000000
Time of Creation:                       2006-06-16-11.46.53.596717
Last Statistics Update:                 2006-06-26-12.23.44.814201
Index Sequential Pages:                 0
Index First 2 Keys Cardinality:         4
Index First 3 Keys Cardinality:         -1
Index First 4 Keys Cardinality:         -1
Index Avg Gap between Sequences:        0.000000
Fetch Avg Gap between Sequences:        -1.000000
Index Avg Sequential Pages:             0.000000
Fetch Avg Sequential Pages:             -1.000000
Index Avg Random Pages:                 1.000000
Fetch Avg Random Pages:                 -1.000000
Index RID Count:                        5
Index Deleted RID Count:                0
Index Empty Leaf Pages:                 0
Avg Partition Cluster Ratio:            -1
Avg Partition Cluster Factor:           -1.000000
Data Partition Cluster Factor:          1.000000
Data Partition Page Fetch Pairs:        Not Available

Base Table Statistics:
----------------------
Name:                                   T2
Schema:                                 DB2USER2
Number of Columns:                      3
Number of Pages with Rows:              1
Number of Pages:                        1
Number of Rows:                         2
Table Overflow Record Count:            0
Width of Rows:                          26
Time of Creation:                       2006-06-16-11.46.53.398092
Last Statistics Update:                 2006-06-26-12.23.45.157028
Statistics Type:                        Synchronous
Primary Tablespace:                     USERSPACE1
Tablespace for Indexes:                 USERSPACE1
Tablespace for Long Data:               NULLP
Number of Referenced Columns:           2
Number of Indexes:                      1
Volatile Table:                         No
Table Active Blocks:                    -1
Number of Column Groups:                0
Number of Data Partitions:              1

Column Information:
-------------------
Number:                                 1
Name:                                   C1
Statistics Available:                   Yes

Column Statistics:
------------------
Schema name of the column type:         SYSIBM
Name of column type:                    INTEGER
Maximum column length:                  4
Scale for decimal column:               0
Number of distinct column values:       2
Average column length:                  5
Number of most frequent values:         -1
```

```
Number of quantiles:                     2
Second highest data value:               2
Second lowest data value:                1
Column sequence in partition key:        0
Average number of sub-elements:          -1
Average length of delimiters:            -1

Column Distribution Statistics:
-------------------------------
Quantile Statistics:
      Valcount   Distcount   Value
      ---------- ----------- -----------
      1          1           1
      2          2           2

Column Information:
-------------------
Number:                                  2
Name:                                    C2
Statistics Available:                    Yes

Column Statistics:
------------------
Schema name of the column type:          SYSIBM
Name of column type:                     INTEGER
Maximum column length:                   4
Scale for decimal column:                0
Number of distinct column values:        2
Average column length:                   5
Number of most frequent values:          -1
Number of quantiles:                     2
Second highest data value:               2
Second lowest data value:                1
Column sequence in partition key:        0
Average number of sub-elements:          -1
Average length of delimiters:            -1

Column Distribution Statistics:
-------------------------------
Quantile Statistics:
      Valcount   Distcount   Value
      ---------- ----------- -----------
      1          0           1
      2          0           2

Indexes defined on the table:
-----------------------------
Name  :                                  IDX_T2C1
Schema:                                  DB2USER2
Unique Rule:                             Duplicate index
Used in Operator:                        No
Page Fetch Pairs:                        Not Available
Number of Columns:                       1
Index Leaf Pages:                        1
Index Tree Levels:                       1
Index First Key Cardinality:             2
Index Full Key Cardinality:              2
Index Cluster Ratio:                     100
Index Cluster Factor:                    -1.000000
Time of Creation:                        2006-06-16-11.46.53.857520
Last Statistics Update:                  2006-06-26-12.23.45.157028
Index Sequential Pages:                  0
Index First 2 Keys Cardinality:          -1
Index First 3 Keys Cardinality:          -1
Index First 4 Keys Cardinality:          -1
Index Avg Gap between Sequences:         0.000000
Fetch Avg Gap between Sequences:         -1.000000
```

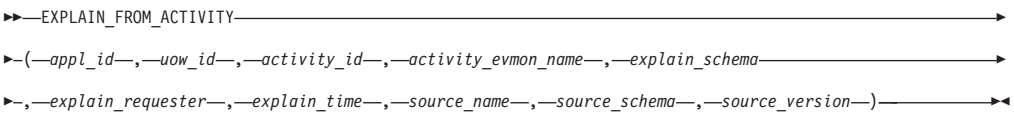
```
Index Avg Sequential Pages:               0.000000
Fetch Avg Sequential Pages:               -1.000000
Index Avg Random Pages:                   1.000000
Fetch Avg Random Pages:                   -1.000000
Index RID Count:                          2
Index Deleted RID Count:                  0
Index Empty Leaf Pages:                   0
Avg Partition Cluster Ratio:              -1
Avg Partition Cluster Factor:             -1.000000
Data Partition Cluster Factor:            1.000000
Data Partition Page Fetch Pairs:          Not Available
```

EXPLAIN_FROM_ACTIVITY プロシージャー - アクティビティー・イベント・モニター情報を使用したステートメントの Explain

EXPLAIN_FROM_ACTIVITY プロシージャーは、アクティビティー・イベント・モニターから得られるセクションの内容を使用して、ステートメントの特定の実行を Explain します。

注: バージョン 9.7 のフィックスパック 1 以前で作成されたデータベースを使用している場合、このルーチンを実行するには、既に db2updv97 コマンドを実行済みでなければなりません。バージョン 9.7 以前に作成されたデータベースを使用している場合は、db2updv97 コマンドを実行する必要はありません (データベースのマイグレーションによって自動的にカタログ更新が実行されるため)。バージョン 9.7 にダウングレードすると、このルーチンは機能しなくなります。

Explain の出力は Explain 表に格納され、任意の既存の Explain ツール (例えば db2exfmt) を使ってこれを処理できます。 Explain の出力には、存在する場合、アクセス・プランとセクション実行時統計(アクセス・プランの演算子に関するランタイム統計) がどちらも含まれます。



スキーマは SYSPROC です。

許可

以下のすべての特権と権限が必要です。

- EXPLAIN_FROM_ACTIVITY プロシージャーに対する EXECUTE 特権
- 指定されたスキーマ内の Explain 表に対する INSERT 特権
- ソース・アクティビティー・イベント・モニターに関するイベント・モニター表に対する SELECT 特権

appl_id
タイプ VARCHAR(64) の入力引数。セクションを Explain する対象となるアクティビティーを発行したアプリケーションを一意的に識別します。 *appl_id* が NULL または空ストリングの場合、SQL2032N が戻されます。

uow_id
タイプ INTEGER の入力引数。セクションを Explain する対象となるアクティ

ビティーの作業単位 ID を指定します。作業単位 ID は、特定のアプリケーション内でのみ固有です。 *uow_id* が NULL である場合、SQL2032N が戻されます。

activity_id
タイプ INTEGER の入力引数。セクションを Explain する対象となるアクティビティーの ID を指定します。アクティビティー ID は、作業単位の中でのみ固有です。 *activity_id* が NULL である場合、SQL2032N が戻されます。

activity_evmon_name
VARCHAR(128) の入力引数。セクションを Explain する対象となるアクティビティーを含む、表への書き込みアクティビティー・イベント・モニターの名前を指定します。イベント・モニターが存在しない場合、またはアクティビティー・イベント・モニターでない場合には、SQL0204N が戻されます。イベント・モニターが表への書き込みイベント・モニターでない場合、SQL20502N が戻されます。 *activity_evmon_name* が指定されない場合、SQL2032N が戻されます。呼び出し元がアクティビティー・イベント・モニター表に対する SELECT 特権を持っていない場合、SQL0551N が戻されます。

explain_schema
タイプ VARCHAR(128) のオプションの入力または出力引数。Explain 情報が書き込まれる Explain 表を含むスキーマを指定します。空ストリングまたは NULL を指定した場合、セッション許可 ID のもとで Explain 表が検索され、その後、SYSTOOLS スキーマで検索されます。 Explain 表が見つからない場合、SQL0219N が戻されます。呼び出し元が Explain 表に対する INSERT 特権を持っていない場合、SQL0551N が戻されます。出力の場合、このパラメーターは、情報が書き込まれた Explain 表を含んでいるスキーマに設定されます。

explain_requester
タイプ VARCHAR(128) の出力引数。このルーチンが呼び出された接続のセッション許可 ID が格納されます。

explain_time
Explain 要求の開始時刻を格納する、タイプ TIMESTAMP の出力引数。

source_name
タイプ VARCHAR (128) の出力引数。ステートメントの準備時またはコンパイル時に実行されていたパッケージの名前を格納します。

source_schema
ソース Explain 要求のスキーマまたは修飾子を格納する、タイプ VARCHAR(128) の出力引数。

source_version
Explain 要求のソースのバージョンを格納する、タイプ VARCHAR(64) の出力引数。

例

以下の例では、一定期間にわたってアクティビティー・イベント・モニターに収集されたデータに対してマイニングを実行していると想定します。以下の照会を使用することで、CPU コストが非常に高い SQL ステートメントの存在に気付きます。

```
SELECT APPL_ID,
       UOW_ID,
       ACTIIVTY_ID,
       USER_CPU_TIME
FROM ACTIVITY_A
ORDER BY USER_CPU_TIME
```

以下は、この照会の出力例を示しています。 N2.DB2INST1.0B5A12222841 という ID を持つアプリケーションが非常に多くの CPU 時間を消費しています。

```
APPL_ID                   UOW_ID     ACTIVITY_ID    USER_CPU_TIME
------------------------  -------- -------------- ---------------
*N2.DB2INST1.0B5A12222841   1              1       92782334234
*N2.DB2INST1.0B5A12725841   2              7               326

  2 record(s) selected.
```

このアクティビティーのアクセス・プランを調べる EXPLAIN_FROM_ACTIVITY プロシージャーを使用して、このアクティビティーのチューニング (例えば索引の追加) に効果があるかどうかを判別できます。

```
CALL EXPLAIN_FROM_ACTIVITY( '*N2.DB2INST1.0B5A12222841', 1, 1, 'A', 'MYSCHEMA',
     ?, ?, ?, ?, ? )
```

使用上の注意

アクティビティーのセクションに対して Explain を実行するには、アクティビティー・データの収集を有効にするときに COLLECT ACTIVITY DATA WITH SECTION 節を指定する必要があります。こうすると、アクティビティー情報と共にセクションが収集されます。識別されたアクティビティー項目と共にセクションが保管されていない場合には、SQL20501 が戻されます。

アクティビティーに関するセクション実行時統計が収集されなかった場合、セクションの Explain は成功しますが、Explain 出力には actuals 情報が含まれません。以下のようなケースでは、セクション実行時統計が収集されません。

- 入力として指定されたアクティビティーが WLM_CAPTURE_ACTIVITY_IN_PROGRESS ストアード・プロシージャーを使ってキャプチャーされた。この場合、アクティビティー論理グループの *partial_record* エレメントの値は 1 です。
- アクティビティー・イベント・モニターの ACTIVITY 表で SECTION_ACTUALS エレメントが欠落している。
- 実行されるセクションが静的セクションであり、DB2 バージョン 9.7 フィックスパック 1 の適用以降、このセクションがまだ再バインドされていない。
- キャプチャー対象のセクションに関するセクション実行時統計が有効になっていなかった。セクション実行時統計を有効にするには、**section_actuals** データベース構成パラメーターを使用します。または特定のアプリケーションを対象に、WLM_SET_CONN_ENV プロシージャーを使用します。デフォルトではセクション実行時統計が無効です。

注: アクティビティーに関するセクション実行時統計が収集されたことを検証するには、ACTIVITY 表の SECTION_ACTUALS エレメントの長さが 0 より大きいかどうか確認してください。

注: アプリケーション用に WLM_SET_CONN_ENV プロシージャーによって指定された **section_actuals** 設定は、直ちに有効になります。セクション実行時統計は、アプリケーションが次に発行するステートメントについて収集されます。

注: パーティション・データベース環境では、アクティビティー・データが収集されるパーティションでのみセクション実行時統計が収集されます。すべてのパーティションで actuals を収集するには、必ず COLLECT ACTIVITY DATA WITH DETAILS, SECTION ON ALL DATABASE PARTITIONS 節を使ってアクティビティーを収集してください。特定のアプリケーションを対象にすべてのパーティションでの収集を有効にするには、WLM_SET_CONN_ENV プロシージャーを呼び出すときに、値を「ALL」に設定した <collect_act_partition> タグを 2 番目の引数に含めます。

入力した *appl_id*、*uow_id*、および *activity_id* に一致するアクティビティーが見つからない場合、SQL20501 が戻されます。実行中に WLM_CAPTURE_ACTIVITY_IN_PROGRESS ストアード・プロシージャーを使ってアクティビティーが複数回にわたって収集された場合には、複数のアクティビティーがこれらに一致する可能性があります。そのような場合、セクションのキャプチャー対象となった最新の項目が Explain に使用されます。

出力パラメーター *explain_requester*、*explain_time*、*source_name*、*source_schema*、および *source_version* はキーを構成し、これを使って Explain 表内のセクションに関する Explain 情報を検索します。セクションから取得された Explain 情報をフォーマット設定するために、既存の任意の Explain ツール (例えば db2exfmt) でこれらのパラメーターを使用できます。

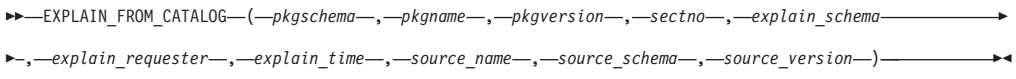
EXPLAIN_FROM_ACTIVITY プロシージャーは、Explain 表への挿入後に COMMIT ステートメントを発行しません。プロシージャーの呼び出し元が COMMIT を発行する必要があります。

ACTIVITYSTMT 論理グループの中には、エレメント STMT_TEXT、ORIGINAL_STMT_TEXT、 SECTION_ENV、 EXECUTABLE_ID、 APPL_ID、ACTIVITY_ID、 UOW_ID が含まれる必要があります。これらのいずれかのエレメントが欠落している場合、ストアード・プロシージャーは SQL206 を戻します。

EXPLAIN_FROM_CATALOG プロシージャー - カタログからのセクション情報を使用したステートメントの Explain

EXPLAIN_FROM_CATALOG プロシージャーは、カタログから得られるセクションの内容を使用してステートメントを Explain します。 Explain の出力は Explain 表に格納され、任意の既存の Explain ツール (例えば db2exfmt) を使ってこれを処理できます。

注: バージョン 9.7 のフィックスパック 1 以前で作成されたデータベースを使用している場合、このルーチンを実行するには、既に db2updv97 コマンドを実行済みでなければなりません。バージョン 9.7 以前に作成されたデータベースを使用している場合は、db2updv97 コマンドを実行する必要はありません (データベースのマイグレーションによって自動的にカタログ更新が実行されるため)。バージョン 9.7 にダウングレードすると、このルーチンは機能しなくなります。



スキーマは SYSPROC です。

許可

以下のすべての特権と権限が必要です。

- EXPLAIN_FROM_CATALOG プロシージャーに対する EXECUTE 特権
- 指定されたスキーマ内の Explain 表に対する INSERT 特権
- EXPLAIN 権限

pkgschema
Explain されるセクションを含むパッケージのスキーマを指定する、タイプ VARCHAR(128) の入力引数。 *pkgschema* が NULL または空ストリングの場合、SQL2032N が戻されます。

pkgname
Explain されるセクションを含むパッケージを指定する、タイプ VARCHAR(128) の入力引数。 *pkgname* が NULL または空ストリングの場合、SQL2032N が戻されます。

pkgversion
Explain されるセクションを含むパッケージのバージョン ID を指定する、タイプ VARCHAR(64) の入力引数。パッケージにバージョンがない場合は、空ストリングを指定します (VARCHAR2 互換モードが有効になっている場合はブランク ' ' 文字)。 *pkgversion* が NULL である場合、SQL2032N が戻されます。

sectno
Explain されるセクションを指定する、タイプ SMALLINT の入力引数。 *sectno* が NULL である場合、SQL2032N が戻されます。

explain_schema
タイプ VARCHAR(128) のオプションの入力または出力引数。Explain 情報が書き込まれる Explain 表を含むスキーマを指定します。空ストリングまたは NULL を指定した場合、セッション許可 ID のもとで Explain 表が検索され、その後、SYSTOOLS スキーマで検索されます。 Explain 表が見つからない場合、SQL0219N が戻されます。呼び出し元が Explain 表に対する INSERT 特権を持っていない場合、SQL0551N が戻されます。出力の場合、このパラメーターは、情報が書き込まれた Explain 表を含んでいるスキーマに設定されます。

explain_requester
タイプ VARCHAR(128) の出力引数。このルーチンが呼び出された接続のセッション許可 ID が格納されます。

explain_time
Explain 要求の開始時刻を格納する、タイプ TIMESTAMP の出力引数。

source_name
タイプ VARCHAR (128) の出力引数。ステートメントの準備時またはコンパイル時に実行されていたパッケージの名前を格納します。

source_schema
ソース Explain 要求のスキーマまたは修飾子を格納する、タイプ VARCHAR(128) の出力引数。

source_version
Explain 要求のソースのバージョンを格納する、タイプ VARCHAR(64) の出力引数。

例

以下の例は、コンパイル済みでカタログ内に存在する静的ステートメントを Explain する方法を示しています。まず、例えば次のようにして SYSCAT.STATEMENTS カタログ・ビューから選択することにより、セクションを識別できます。

```
SELECT pkgschema,
       pkgname,
       version,
       Sectno
FROM SYSCAT.STATEMENTS
WHERE TEXT = 'select count(*) from syscat.tables'
```

この照会により、以下の例のような出力が戻されます。

```
PKGSCHEMA  PKGNAME              PKGVERSION           SECTNO
---------- -------------------- -------------------- ------
NULLID     SQLE2G0S                                  1
NULLID     SQLE2G0S             VERSION1             1

     2 record(s) selected.
```

次に、例えば以下のようにして、識別情報 *pkgschema*、*pkgname*、*pkgversion* および *sectno* を EXPLAIN_FROM_CATALOG プロシージャーに渡します。

```
CALL EXPLAIN_FROM_CATALOG( 'NULLID', 'SQLE2G0S', '', 1, 'MYSCHEMA', ?, ?, ?, ?, ? )
```

使用上の注意

入力パラメーターに一致するセクションが見つからない場合は SQL20501 が戻されます。

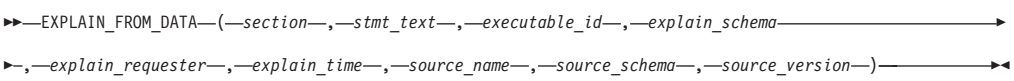
出力パラメーター *explain_requester*、*explain_time*、*source_name*、*source_schema*、*source_version* はキーを構成し、これを使って Explain 表内のセクションに関する Explain 情報を検索します。セクションから取得された Explain 情報をフォーマット設定するために、既存の任意の Explain ツール (例えば db2exfmt) でこれらのパラメーターを使用できます。

このプロシージャーは、Explain 表への挿入後に COMMIT ステートメントを発行しません。プロシージャーの呼び出し元が COMMIT を発行する必要があります。

EXPLAIN_FROM_DATA プロシージャー - 入力セクションを使用したステートメントの Explain

EXPLAIN_FROM_DATA プロシージャーは、入力セクションの内容を使用してステートメントを Explain します。 Explain の出力は Explain 表に格納され、任意の既存の Explain ツール (例えば db2exfmt) を使ってこれを処理できます。

注: バージョン 9.7 のフィックスパック 1 以前で作成されたデータベースを使用している場合、このルーチンを実行するには、既に db2updv97 コマンドを実行済みでなければなりません。バージョン 9.7 以前に作成されたデータベースを使用している場合は、db2updv97 コマンドを実行する必要はありません (データベースのマイグレーションによって自動的にカタログ更新が実行されるため)。バージョン 9.7 にダウングレードすると、このルーチンは機能しなくなります。



スキーマは SYSPROC です。

許可

以下のすべての特権と権限が必要です。

- EXPLAIN_FROM_DATA プロシージャーに対する EXECUTE 特権
- 指定されたスキーマ内の Explain 表に対する INSERT 特権

section
Explain 対象のセクションが含まれる、タイプ BLOB(134M) の入力引数。イベント・モニター表、カタログ表など、さまざまなソースからセクションを取得できます。入力セクションが有効なセクションでない場合、SQL20503N が戻されます。

stmt_text
オプションの、タイプ CLOB(2M) の入力引数。入力セクションに対応するステートメントのテキストが含まれます。 *stmt_text* が NULL の場合、フォーマット済み Explain 出力にはステートメント・テキストが含まれません。

executable_id
オプションの、タイプ VARCHAR(32) FOR BIT DATA の入力引数。セクションの識別に使用される実行可能 ID が含まれます。 *executable_id* が NULL の場合、フォーマット済み Explain 出力には実行可能 ID が含まれません。

explain_schema
タイプ VARCHAR(128) のオプションの入力または出力引数。Explain 情報が書き込まれる Explain 表を含むスキーマを指定します。空ストリングまたは NULL を指定した場合、セッション許可 ID のもとで Explain 表が検索され、その後、SYSTOOLS スキーマで検索されます。 Explain 表が見つからない場合、SQL0219N が戻されます。呼び出し元が Explain 表に対する INSERT 特権を持っていない場合、SQL0551N が戻されます。出力の場合、このパラメーターは、情報が書き込まれた Explain 表を含んでいるスキーマに設定されます。

explain_requester
タイプ VARCHAR(128) の出力引数。このルーチンが呼び出された接続のセッション許可 ID が格納されます。

explain_time
Explain 要求の開始時刻を格納する、タイプ TIMESTAMP の出力引数。

source_name
タイプ VARCHAR (128) の出力引数。ステートメントの準備時またはコンパイル時に実行されていたパッケージの名前を格納します。

source_schema
ソース Explain 要求のスキーマまたは修飾子を格納する、タイプ VARCHAR(128) の出力引数。

source_version
Explain 要求のソースのバージョンを格納する、タイプ VARCHAR(64) の出力引数。

例

パッケージ・キャッシュ・イベント・モニターを使って多数のステートメントをキャプチャーした後、(EVMON_FORMAT_UE_TO_TABLE ストアード・プロシージャーを使って) PKGCACHE という表にイベント・モニター・データを抽出したとします。表のデータを調べてみると、非常にコストが高い、実行可能 ID「x'0100000000000000007000000000000000000000000000020020081126190410369 8'」のステートメントが見つかりました。

このステートメントのアクセス・プランを把握するために EXPLAIN_FROM_DATA プロシージャーを発行し、PKGCACHE 表の項目からのセクションを入力として渡します。 MYSCHEMA スキーマの Explain 表の中に Explain 出力を書き込みます。

```
SET SERVEROUTPUT ON;

BEGIN
  DECLARE EXECUTABLE_ID VARCHAR(32) FOR BIT DATA; --
  DECLARE SECTION BLOB(134M); --
  DECLARE STMT_TEXT CLOB(2M); --
  DECLARE EXPLAIN_SCHEMA VARCHAR(128); --

  DECLARE EXPLAIN_REQUESTER VARCHAR(128); --
  DECLARE EXPLAIN_TIME TIMESTAMP; --
  DECLARE SOURCE_NAME VARCHAR(128); --
  DECLARE SOURCE_SCHEMA VARCHAR(128); --
  DECLARE SOURCE_VERSION VARCHAR(128); --

  SET EXPLAIN_SCHEMA = 'MYSCHEMA'; --

  SELECT P.SECTION, P.STMT_TEXT, P.EXECUTABLE_ID INTO
         SECTION, STMT_TEXT, EXECUTABLE_ID
  FROM PKGCACHE WHERE EXECUTABLE_ID =
   x'0100000000000000007000000000000000000000000000020020081126190410369 8'; --

  CALL EXPLAIN_FROM_DATA( SECTION,
                          STMT_TEXT,
                          EXECUTABLE_ID,
                          EXPLAIN_SCHEMA,
                          EXPLAIN_REQUESTER,
                          EXPLAIN_TIME,
                          SOURCE_NAME,
                          SOURCE_SCHEMA,
                          SOURCE_VERSION ); --

  CALL DBMS_OUTPUT.PUT( 'EXPLAIN_REQUESTER = ' ); --
  CALL DBMS_OUTPUT.PUT_LINE( EXPLAIN_REQUESTER ); --
  CALL DBMS_OUTPUT.PUT( 'EXPLAIN_TIME = ' ); --
  CALL DBMS_OUTPUT.PUT_LINE( EXPLAIN_TIME ); --
```

```
  CALL DBMS_OUTPUT.PUT( 'SOURCE_NAME = ' ); --
  CALL DBMS_OUTPUT.PUT_LINE( SOURCE_NAME ); --
  CALL DBMS_OUTPUT.PUT( 'SOURCE_SCHEMA = ' ); --
  CALL DBMS_OUTPUT.PUT_LINE( SOURCE_SCHEMA ); --
  CALL DBMS_OUTPUT.PUT( 'SOURCE_VERSION = ' ); --
  CALL DBMS_OUTPUT.PUT_LINE( SOURCE_VERSION ); --
END;

SET SERVEROUTPUT OFF;
```

使用上の注意

入力セクションは、以下のようなさまざまなソースから入手可能です。

- アクティビティー・イベント・モニター
- パッケージ・キャッシュ・イベント・モニター
- カタログ表
- 上記のいずれかの場所からセクションをコピーした、任意のユーザー表または入力ソース。

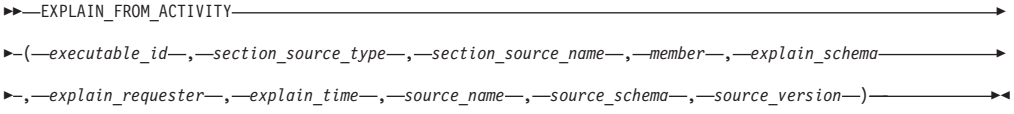
出力パラメーター *explain_requester*、*explain_time*、*source_name*、*source_schema*、*source_version* はキーを構成し、これを使って Explain 表内のセクションに関する Explain 情報を検索します。セクションから取得された Explain 情報をフォーマット設定するために、既存の任意の Explain ツール (例えば db2exfmt) でこれらのパラメーターを使用できます。

このプロシージャーは、Explain 表への挿入後に COMMIT を発行しません。プロシージャーの呼び出し元が COMMIT を発行する必要があります。

EXPLAIN_FROM_SECTION プロシージャー - パッケージ・キャッシュまたはパッケージ・キャッシュ・イベント・モニター情報を使用したステートメントの Explain

EXPLAIN_FROM_SECTION プロシージャーは、パッケージ・キャッシュまたはパッケージ・キャッシュ・イベント・モニターから得られるセクションの内容を使ってステートメントを Explain します。 Explain の出力は Explain 表に格納され、任意の既存の Explain ツール (例えば db2exfmt) を使ってこれを処理できます。

注: バージョン 9.7 のフィックスパック 1 以前で作成されたデータベースを使用している場合、このルーチンを実行するには、既に db2updv97 コマンドを実行済みでなければなりません。バージョン 9.7 以前に作成されたデータベースを使用している場合は、db2updv97 コマンドを実行する必要はありません (データベースのマイグレーションによって自動的にカタログ更新が実行されるため)。バージョン 9.7 にダウングレードすると、このルーチンは機能しなくなります。



スキーマは SYSPROC です。

許可

以下のすべての特権と権限が必要です。

- EXPLAIN_FROM_SECTION プロシージャーに対する EXECUTE 特権
- 指定されたスキーマ内の Explain 表に対する INSERT 特権
- セクション・ソース名がパッケージ・キャッシュ・イベント・モニターを識別する場合には、パッケージ・キャッシュ・イベント・モニター表に対する SELECT 特権

executable_id
Explain されるセクションを一意的に識別する、タイプ VARCHAR(32) FOR BIT DATA の入力引数。この引数が NULL または空ストリングである場合、SQL2032 が戻されます。

section_source_type
Explain されるセクションのソースを指定する、タイプ CHAR(1) の入力引数。有効な値は以下のとおりです。

- M - メモリー内のパッケージ・キャッシュからセクションが取得されます
- P - パッケージ・キャッシュ・イベント・モニターからセクションが取得されます

静的 SQL では、*section_source_type* が M でセクションがパッケージ・キャッシュに存在しない場合、カタログ表の中でセクションが検索されます。

section_source_name
VARCHAR(128) の入力引数。*section_source_type* が P の場合はパッケージ・キャッシュ・イベント・モニターの名前を指定します。*section_source_type* が M の場合は、オプションで、パッケージ・キャッシュ・イベント・モニターの名前を指定できます。パッケージ・キャッシュ内にセクションが見つからない場合 (例えば EXPLAIN_FROM_SECTION ストアード・プロシージャーの呼び出し前にセクションがパッケージ・キャッシュからフラッシュされた場合)、イベント・モニターの中でセクションが検索されます。ソース入力イベント・モニターが、COLLECT DETAILED DATA オプションを使って作成されたパッケージ・キャッシュ・イベント・モニターではない場合、SQL0204N が戻されます。呼び出し元がパッケージ・キャッシュ・イベント・モニター表に対する SELECT 特権を持っていない場合、SQL0551N が戻されます。

member
タイプ INTEGER の入力引数。*section_source_type* が M の場合、Explain 対象のセクションがメモリー内で格納されているメンバーを指定します。 -1 を指定した場合、プロシージャーは現在のコーディネーター・メンバーおよびセクション・コンパイル・メンバーの中でセクションを検索します。 *section_source_type* が M 以外の場合には、この引数は無視されます。

explain_schema
タイプ VARCHAR(128) のオプションの入力または出力引数。Explain 情報が書き込まれる Explain 表を含むスキーマを指定します。空ストリングまたは NULL を指定した場合、セッション許可 ID のもとで Explain 表が検索され、その後、SYSTOOLS スキーマで検索されます。 Explain 表が見つからない場合、SQL0219N が戻されます。呼び出し元が Explain 表に対する INSERT 特権

を持っていない場合、SQL0551N が戻されます。出力の場合、このパラメーターは、情報が書き込まれた Explain 表を含んでいるスキーマに設定されます。

explain_requester
タイプ VARCHAR(128) の出力引数。このルーチンが呼び出された接続のセッション許可 ID が格納されます。

explain_time
Explain 要求の開始時刻を格納する、タイプ TIMESTAMP の出力引数。

source_name
タイプ VARCHAR (128) の出力引数。ステートメントの準備時またはコンパイル時に実行されていたパッケージの名前を格納します。

source_schema
ソース Explain 要求のスキーマまたは修飾子を格納する、タイプ VARCHAR(128) の出力引数。

source_version
Explain 要求のソースのバージョンを格納する、タイプ VARCHAR(64) の出力引数。

例

この例は、セクションごとに入手可能なモニタリング・メトリックを調べることで、パッケージ・キャッシュ内の非常にコストが高いステートメントを識別して分析する方法を示しています。まず、セクションによる CPU 時間の使用状況を判別するために、以下の SELECT ステートメントのような照会を発行します。

```
SELECT executable_id,
       Total_cpu_time,
       Varchar(stmt_text, 100) as stmt_text
FROM TABLE(MON_GET_PKG_CACHE_STMT (NULL, NULL,NULL, -1)) AS T
```

以下は、この照会の出力例を示しています。

```
EXECUTABLE_ID                                TOTAL_CPU_TIME ...
-------------                                -------------- ...
x'010000000000000012...20020081126190410369' 91875622       ...
x'010000000000000007...20020081126190410323'      300       ...

     2 record(s) selected.
```

以下は、この照会の出力例の続きを示しています。

```
...STMT_TEXT
...----------------------
...SELECT * FROM SYSCAT.TABLES
...INSERT INTO T1 VALUES(123)

     2 record(s) selected.
```

高コストの SELECT ステートメントのアクセス・プランを調べるには、その *executable_id* を EXPLAIN_FROM_SECTION プロシージャーに渡します。出力を MYSCHEMA スキーマの Explain 表に入れます。

```
CALL EXPLAIN_FROM_SECTION
     ( x'010000000000000012000000000000000000000000020020081126190410369',
      'M', NULL, 0, 'MYSCHEMA', ?, ?, ?, ?, ? )
```

使用上の注意

入力の実行可能 ID に一致するセクションが見つからない場合、SQL20501 が戻されます。入力 *executable_id* は、以下のソースから入手可能です。

- アクティビティー・イベント・モニター
- パッケージ・キャッシュ・イベント・モニター
- MON_GET_ACTIVITY_DETAILS 表関数
- MON_GET_PKG_CACHE_STMT 表関数
- WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES_V97 表関数
- WLM_GET_SERVICE_CLASS_AGENTS_V97 表関数
- MON_GET_PKG_CACHE_STMT_DETAILS 表関数
- MON_GET_APP_LOCK_WAITS 表関数

出力パラメーター *explain_requester*、*explain_time*、*source_name*、*source_schema*、*source_version* はキーを構成し、これを使って Explain 表内のセクションに関する情報を検索します。セクションから取得された Explain 情報をフォーマット設定するために、既存の任意の Explain ツール (例えば db2exfmt) でこれらのパラメーターを使用できます。

このプロシージャーは、Explain 表への挿入後に COMMIT ステートメントを発行しません。プロシージャーの呼び出し元が COMMIT を発行する必要があります。

第 12 章 モニター・ルーチンおよびビュー

モニター表関数およびビューは、「MON」で始まる名前を持つルーチンです (例えば MON_GET_SERVICE_SUBCLASS)。これらの表関数とビューは、DB2 バージョン 9.7 で導入されたモニター・インフラストラクチャーから使用できるモニター・エレメントへのアクセスを提供します。スナップショット関数など他の特定のルーチンも、モニター情報を戻します。

「MON」ルーチンは戦略的に重要であるため、このルーチンの名前は将来のリリースでも変更されません。ただし、将来のリリースで機能が拡張される場合には、新しい出力列が追加されます。したがって、システム定義ルーチンまたはビューを使って情報を取得するために照会を発行するとき、SELECT * ... という形式のステートメントを使用しないでください。その代わりに、SELECT ステートメントで結果列を指定します。こうすることで、アプリケーションは結果列の数と、それらが戻される順序を制御することができます。

モニター (MON) 表関数

すべての表関数には、共通のモニター・エレメントの集合が組み込まれます。これらのエレメントは、アプリケーション応答時間に影響する可能性のあるさまざまなシステム・パフォーマンス標識の集合に関する情報を提供します。関心あるワークロードのサブセットに関するモニター・データを入手することもできます。

例えば、次のようなモニター表関数は、システム・ワークロード全体のさまざまな局面に関して報告します。

- MON_GET_CONNECTION および MON_GET_CONNECTION_DETAILS
- MON_GET_SERVICE_SUBCLASS および MON_GET_SERVICE_SUBCLASS_DETAILS
- MON_GET_UNIT_OF_WORK および MON_GET_UNIT_OF_WORK_DETAILS
- MON_GET_WORKLOAD および MON_GET_WORKLOAD_DETAILS

これらの表関数には 2 つのバージョンがあります。その 1 つの接尾部は _DETAILS になります。 _DETAILS 接尾部のないバージョンは、最も一般的に使用されるデータを戻すリレーショナル SQL インターフェースを提供します。_DETAILS 接尾部のあるバージョンは、モニター・データに対する XML ベースのアクセスを提供し、より包括的なデータ・セットを戻します。

このほかに、例えば次のような表関数は、特定のタイプのデータ・オブジェクトに関するデータを戻します。

- MON_GET_APPL_LOCKWAIT
- MON_GET_BUFFERPOOL
- MON_GET_CONTAINER
- MON_GET_INDEX
- MON_GET_LOCKS
- MON_GET_TABLE

© Copyright IBM Corp. 2006, 2010

- MON_GET_TABLESPACE
- MON_GET_PKG_CACHE_STMT

これらの表関数は、特定のデータ・オブジェクトに関連したパフォーマンスの問題を調査するのに使用します。

バージョン 9.7 フィックスパック 2 リリースでは、高速コミュニケーション・マネージャー (FCM) に関するデータを戻す以下の表関数が追加されました。

- MON_GET_FCM
- MON_GET_FCM_CONNECTION_LIST

さらに別の表関数は、個々のアクティビティーおよびステートメントの詳細を調べるのに役立ちます。

- MON_GET_ACTIVITY_DETAILS は、システム上で現在実行中の特定のアクティビティーに関する詳細を戻します。これらの詳細には、一般的なアクティビティー情報 (ステートメント・テキストなど) や一連のメトリックが含まれます。

さらに、以下の表関数は、進捗モニターの役割を果たします。

- MON_GET_EXTENT_MOVEMENT_STATUS はエクステント移動操作の状況を戻します。

MON_FORMAT_ で始まる表関数は、読みやすい行ベースの形式で情報を戻します。 MON_FORMAT_LOCK_NAME はロックの内部バイナリー名を入力として、そのロックに関する詳細情報を戻します。 MON_FORMAT_XML_ で始まる表関数は、いずれかの MON_GET_*_DETAILS 表関数によって (あるいは統計、アクティビティー、作業単位、またはパッケージ・キャッシュ・イベント・モニターの出力から) 戻される XML メトリック文書を入力として受け入れ、フォーマット設定された行ベースの出力を戻します。

- MON_FORMAT_XML_COMPONENT_TIMES_BY_ROW はコンポーネント時間に関するフォーマット設定された行ベースの出力を戻します。
- MON_FORMAT_XML_METRICS_BY_ROW はすべてのメトリックに関するフォーマット設定された行ベースの出力を戻します。
- MON_FORMAT_XML_TIMES_BY_ROW は、待機時間と処理時間の結合された階層に関するフォーマット設定された行ベースの出力を戻します。
- MON_FORMAT_XML_WAIT_TIMES_BY_ROW 表関数は、待機時間に関するフォーマット設定された行ベースの出力を戻します。

モニター (MON) 表関数の特性

- モニター表関数によって戻されるメトリックがリセットされることはありません。データベースがアクティブにされる時点で 0 で開始され、データベースが非アクティブ化されるまで継続的に累算されます。
- ほとんどの表関数と同様に、受信データの対象を単一のオブジェクト (サービス・クラス「A」など) にするか、すべてのオブジェクトにするかを選択できます。
- ほとんどの表関数と同様に、パーティション・データベース環境でこれらの表関数を使用する際には、受信データの対象を単一のパーティションにするか、すべてのパーティションにするかを選択できます。すべてのパーティションに関する

データの受信を選択すると、表関数はパーティションごとに 1 行ずつ戻します。パーティション全体の値を加算して、パーティション全体にわたるモニター・エレメントの値を取得できます。

モニター (MON) ビュー

モニター・ビューは、例えば次のような、さまざまなデータベース・アクティビティーに関するメトリックを戻します。

- MON_CURRENT_SQL は、データベースの全メンバーに対してサブミットされた、まだ未完了であるすべてのアクティビティーに関するメトリックを戻します (現在実行中の SQL ステートメントのポイント・イン・タイム・ビューを含む)。
- MON_DB_SUMMARY はすべてのサービス・クラスにわたって集約されたメトリックを戻します。
- MON_LOCKWAITS は、現在接続しているデータベースでのロック取得を待機しているアプリケーションのために機能するエージェントについての情報を戻します。
- MON_SERVICE_SUBCLASS_SUMMARY はすべてのサービス・サブクラスに関するメトリックを戻し、サービス・クラスごとに実行された作業を示します。
- MON_CURRENT_UOW はすべての作業単位に関するメトリックを戻します。
- MON_WORKLOAD_SUMMARY はすべてのワークロードに関するメトリックを戻し、ワークロードごとに受信される作業を示します。

イベント・モニター (EVMON) ルーチン

DB2 バージョン 9.7 リリースは、2 つの新しいルーチンも導入しています。これらのルーチンの目的は、他の「MON」表関数とは多少違います。これらのルーチンは、イベントを未フォーマット・イベント表に書き込むイベント・モニターからデータを抽出してフォーマットします。LOCKING および UNIT OF WORK イベント・モニター・タイプは、不定形式のイベント表を使用します。ルーチン名は、次のとおりです。

- EVMON_FORMAT_UE_TO_XML 表関数
- EVMON_FORMAT_UE_TO_TABLES プロシージャー

これらのルーチンを使用すると、EVMON_FORMAT_UE_TO_XML 表関数を使用して XML 文書を介するか、または EVMON_FORMAT_UE_TO_TABLES プロシージャーを使用してリレーショナル表を介することにより、イベント・モニター・データにアクセスできます。

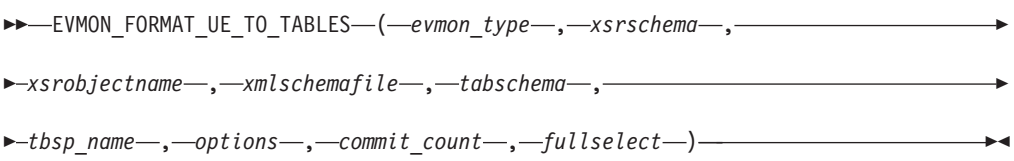
EVMON_FORMAT_UE_TO_TABLES プロシージャー - XML 文書をリレーショナル表へ移動する

EVMON_FORMAT_UE_TO_TABLES プロシージャーは、イベント・モニターによって作成された未フォーマット・イベント (UE) 表に保管されたデータを取り出し、一連のリレーショナル表に変換します。

リレーショナル表の作成処理には 2 つのステップがあります。最初は、EVMON_FORMAT_UE_TO_XML 表関数を使用して、UE 表にあるデータを XML 形式に変換します。この表関数は、EVMON_FORMAT_UE_TO_TABLES プロシージ

ャーの実行の一部として、自動的に実行されます。次に、イベント・モニター・データを含む XML 文書を、XML 分解を使用してリレーショナル表に入れます。

構文



スキーマは SYSPROC です。

表関数パラメーター

evmon_type
フォーマットされていないイベント表に保管されているデータのタイプを表す、タイプ VARCHAR(128) の入力パラメーター。可能な値は、以下のとおりです。

LOCKING
未フォーマット・イベント表に保管されているデータは、ロック・イベント・モニターからのものです。

PKGCACHE
フォーマットされていないイベント表に保管されているデータは、PACKAGE CACHE イベント・モニターからのものです。

UOW フォーマットされていないイベント表に保管されているデータは、UOW イベント・モニターからのものです。

xsrschema
UE ファイルからのデータがどのように表の列に対応するかを示す XSR オブジェクトの名前の最初の部分を指定する VARCHAR (128) タイプの入力パラメーター。XSR オブジェクト名の 2 番目の部分は *xsrobjectname* パラメーターから導出されます。完全な XSR オブジェクト名は、*xsrschema.xsrobjectname* と定義されます。この値が NULL の場合は、現行セッションのユーザーの許可 ID が使用されます。

xsrobjectname
UE ファイルからのデータがどのように表の列に対応するかを示す XSR オブジェクトの名前の 2 番目の部分を指定する VARCHAR (128) タイプの入力パラメーター。XSR オブジェクト名の最初の部分は、*xsrschema* パラメーターから導出されます。完全な XSR オブジェクト名は *xsrschema.xsrobjectname* と定義され、XSR 内のすべてのオブジェクト間で固有です。この値が NULL の場合、*xsrobjectname* は EVMON_<evmon_type>_SCHEMA_<SQL release level> として導出されます。例えば、DB2 バージョン 9.7 のロック・イベント・モニターは、EVMON_LOCKING_SCHEMA_SQL09070 という導出された *xsrname* を持ちます。

XSR オブジェクトは、イベント・モニターの出力を説明する XML スキーマ・ファイルのコピーです。これは、XML スキーマ・リポジトリー (XSR) に格納され、EVMON_FORMAT_UE_TO_TABLES 処理の最初の段階で作成される一時 XML 文書の要素と、最終的にプロシージャーが作成する表および列の間のリレーションシップを定義します。 XSR オブジェクトは、作成されるすべての表と

それらの表を派生させる XML スキーマとの間の相互依存関係の管理を行うためにも使用されます。XSR オブジェクトがドロップされるか、プロシージャーにより作成されたいずれかの表がドロップされたり列が変更されたりした場合、これら 2 つのものの間の依存関係は「壊れた」と言います。
EVMON_FORMAT_UE_TO_TABLES (または EVMON_FORMAT_UE_TO_XML 表関数) が特定のタイプのイベント・モニターの UE ファイルに対してまだ実行されていない場合、イベント・モニター出力を説明する XSR オブジェクトはまだ存在していません。この場合、イベント・モニター用の XML スキーマ・ファイルが使用されて、システム・カタログ表に XSR オブジェクトを作成し登録します。

xmlschemafile
イベント・モニターにより作成された出力を説明するディスク上の XML スキーマ文書の絶対パスを表わす、タイプ VARCHAR (1024) の入力パラメーター。XML スキーマ文書要素には、XML 要素と属性をリレーショナル表とその列にマップする情報でアノテーションが付けられています。

このパラメーターは、XSR オブジェクトを登録するために使用されます。*evmon_type* で指定されているイベント・モニターのタイプに関して登録されていて使用可能になっている XSR オブジェクトが存在しない場合、XSR オブジェクトは以下のように登録されます。

- *xmlschemafile* が NULL の場合、*evmon_type* に指定されている値に応じて、以下のように、プロシージャーはディスク上の XML スキーマ・ファイルを使用します。

  LOCKING
  `sqllib/misc/DB2EvmonLocking.xsd`

  PKGCACHE
  `sqllib/misc/DB2EvmonPkgCache.xsd`

  UOW `sqllib/misc/DB2EvmonUOW.xsd`
- XML スキーマ・ファイル名を指定した場合、分解のための XSR オブジェクトを登録し使用可能にするためにそのファイルが使用されます。
- *xsrschema* および *xsrobjectname* パラメーターの値を指定した場合、これらの名前を使用して XSR オブジェクトが作成されます。指定されていない場合、XSR オブジェクトの名前は、以前 *xsrobjectname* について説明したように、デフォルトを使用して付けられます。

重要: 以前、XSR オブジェクトが分解のために登録されて使用可能になっている場合、このパラメーターは無視されます。別の XML スキーマ・ファイルを使用する XSR オブジェクトを登録した場合、まず、既存の XSR オブジェクトをドロップする必要があります。

tabschema
イベント・モニターのリレーショナル表が作成される SQL スキーマ名を表す、タイプ VARCHAR (128) の入力パラメーター。この値が NULL の場合は、現行セッションのユーザーの許可 ID が使用されます。表が作成される SQL スキーマは、次のように判別されます。

- `<db2-xdb:SQLSchema>` が指定される場合には、このスキーマを使用します。

- <db2-xdb:defaultSchema> が指定される場合には、このスキーマを使用します。
- これらのどちらの値も指定されない場合には、*sqlschema* 入力パラメーターからの値を使用します。

注: XML スキーマが分解用に登録されると、XSR スキーマ・リポジトリーは、スキーマ内で参照される各表とこのスキーマに対応する XSR オブジェクトの従属関係を作成します。これは、データベース内のリレーショナル表の固有のセットに XSR オブジェクト名がリンクされることを意味します。既存の XSR オブジェクトを参照すると、そのデータは必ず分解され、XSR オブジェクトがリンクされた表に挿入されます。

tbsp_name
リレーショナル表が作成される表スペースを示す、タイプ VARCHAR(128) の入力パラメーター。このパラメーターのデフォルト値は NULL です。なお、XML スキーマ・ファイル内の CREATE TABLE ステートメントで指定された表スペース名は、この入力パラメーターよりも優先されることに注意してください。

options
この表関数でサポートされるキーワード・オプションのリストを表す、タイプ VARCHAR (1024) の入力パラメーター。各オプションはセミコロン (;) 文字で区切られている必要があります。使用できる値は次のとおりです。

RECREATE_FORCE
分解の前にリレーショナル表をドロップして再作成します。

RECREATE_ONERROR
以下の状況では、リレーショナル表をドロップして再作成します。

1. XSR オブジェクトは登録されていないが、表が存在する場合。
2. 最初に失敗した分解の試行の際。その後の失敗は戻され、表を再作成する試みは行われません。

例えば表スペースの満杯エラーや許可エラーなど、エラーが発生した場合にプロシージャーは、分解プロシージャーから戻される SQLCODE をフィルターに掛けません。プロシージャーは負の SQLCODE すべてを等しく扱い、表の再作成を試みます。

commit_count
タイプ INTEGER の入力パラメーター。可能な値は、以下のとおりです。

-1 100 文書が正常に分解されるごとにコミットします。-1 がデフォルト値です。

0 コミットなし。

n *n* 個の文書が正常に分解されるごとにコミットします。

fullselect
フォーマットされていないイベント表からの全選択ステートメントを表す、タイプ CLOB(2M) の入力パラメーター。全選択ステートメントは、SELECT ステートメントの規則に準拠する照会です。この照会は、以下の規則に従っていなければなりません。

- この照会は、"*" 節を使用するか、未フォーマット・イベント表のすべての列を指定する必要があります。それ以外の場合、エラーが戻されます。列は、未フォーマット・イベント表の DESCRIBE ステートメントで戻されたのと同じ順序で指定されている必要があります。
- この照会は、フォーマットされていないイベント表からのみ選択を行う必要があります。
- WHERE 節は、フォーマットされていないイベント表の非 LOB 列のいずれかを使用して、イベントをフィルターで除外することができます。

許可

EVMON_FORMAT_UE_TO_TABLES ストアード・プロシージャーに対する EXECUTE 特権。

未フォーマット・イベント表に対する SELECT 特権 (表を作成していない場合)。

指定した SQL スキーマのリレーショナル表を作成するための CREATE 特権。

リレーショナル表に挿入するための INSERT 特権 (それらの表を作成していない場合)。

XDB_DECOMP_XMP_FROM_QUERY プロシージャーが必要とするすべての特権。

使用上の注意

表の作成

分解が行われるためには、一連のリレーショナル表が存在しなければなりません。EVMON_FORMAT_UE_TO_TABLES プロシージャーは、以下のようにリレーショナル表を自動的に作成します。

- プロシージャーは、<db2_mon:createStmt> エレメントを見つけるために、イベント・モニターの XML スキーマ・ファイルを解析します。各エレメントには、完全な CREATE TABLE ステートメントが含まれています。
- プロシージャーは CREATE TABLE ステートメントを抽出して実行します。

<db2_mon:createStmt> は、既存の <db2_xdb:table> エレメントの子エレメントです。この子エレメントを認識して使用できるのは、EVMON_FORMAT_UE_TO_TABLES プロシージャーだけです。 XML スキーマ・ファイルを解析する XSR オブジェクトなどの他のプロシージャーはすべて、このエレメントを無視します。

<db2_mon:createStmt> 内の表名を修飾してはなりません。

各リリースに対応した XML スキーマ・ファイル

イベント・モニターごとに用意されているデフォルトの XML スキーマ・ファイルは、現行リリース用の XML スキーマを常に反映しています。それで、EVMON_FORMAT_UE_TO_TABLES (または EVMON_FORMAT_UE_TO_XML) を実行する際には、出力はそのリリースのイベント・モニター用に定義されているモニター要素を反映します。次のセクションでは、イベント・モニター用のスキーマ・ファイルが時間と共に変更された場合に何が起こるかを説明します。

EVMON_FORMAT_UE_TO_TABLES プロシージャーを使用して表を作成してから、フィックスパックを適用したり新規リリースにアップグレードしたりする場合、これらの変更の影響について理解することが重要です。

EVMON_FORMAT_UE_TO_TABLES により作成される表へのスキーマ更新の影響

将来のフィックスパックまたはリリースでは、新規モニター要素がイベント・モニターに追加されるものと思われます。これらの新規モニター要素により、新しい列また新規表を EVMON_FORMAT_UE_TO_TABLES プロシージャーが作成することになるかもしれません。しかし、フィックスパックを適用したり新規リリースにアップグレードしたりする前に既にこのプロシージャーで作成した表がある場合、以下のことを行って新規リレーショナル列または表を作成する必要があります。

フィックスパック更新の場合

最新のフィックスパックのインストール前に EVMON_FORMAT_UE_TO_TABLES により作成されたリレーショナル表がまだ存在する場合、リレーショナル形式で新規モニター要素を表示したいならば、フィックスパックで出荷された新規スキーマに基づいて一連の新しい表を作成させなければなりません。

EVMON_FORMAT_UE_TO_TABLES プロシージャーに、フィックスパックで出荷された新規スキーマを強制的に使用させて新規表を作成させるには、以下のステップを行います。

1. 現登録バージョンの XML スキーマ (スキーマの登録について詳しくは、EVMON_FORMAT_UE_TO_TABLES プロシージャーの *tabschema* パラメーターの 注を参照してください) と既存の表の間の依存関係を、以下のいずれかを行って切断してください。
   - EVMON_FORMAT_UE_TO_TABLES によって作成された既存の表の 1 つをドロップする
   - DROP XSROBJECT ステートメントを使用して、既存の表に関連する登録済み XML スキーマ・オブジェクトをドロップする。例えば、DB2 V9.7 のロッキング・イベント・モニター用に EVMON_FORMAT_UE_TO_TABLES により作成された表に関連した登録済み XML スキーマ・オブジェクトをドロップするには、次のコマンドを使用します: `DROP XSROBJECT EVMON_LOCKING_SCHEMA_SQL09070`。
   - 現行の登録済み XML スキーマ・オブジェクトのアノテーションの付いたモニター要素に対応する既存の列を変更する。
2. FORCE オプションを使用して、EVMON_FORMAT_UE_TO_TABLES プロシージャーを実行する。このオプションにより、以前の表はドロップされ、一連の新しい表が作成されます。このオプションを指定しないと、SQL0601N エラーが戻ります。

このプロセスは、403 ページの『例 5: フィックスパック更新に含まれる新規エレメントを取り出す』 に例示されています。

上記のステップを実行しない場合、既存の表は以前に登録されたスキーマ・ファイルに基づいて更新されます。フィックスパックに追加されているかもしれない新しい列または表は、EVMON_FORMAT_UE_TO_TABLES プロシージャーの出力に反映されません。

リリース・アップグレードの場合

他の指定をしない場合、EVMON_FORMAT_UE_TO_TABLES プロシージャーを呼び出す際には、現行リリース用の XML スキーマ・ファイルのデフォルト・バージョンが使用されます。それで、DB2 製品の新規リリースにアップグレードする場合は、このプロシージャーを実行する際には、デフォルトで、スキーマ・ファイルの新規バージョンが使用されます。

以前のリリースからの表が存在しない場合、EVMON_FORMAT_UE_TO_TABLES は最新のスキーマ使用して表を作成します。しかし、以前のリリースからの表が存在する場合、FORCE または RECREATE_ONERROR オプションを使用して以前の表を新規表に置き換えなければなりません。こうしないと、SQL0601N エラーが戻ります。 403 ページの『例 6: リリース更新に含まれる新規エレメントを取り出す』は、新規リリースでデフォルト・スキーマを使用して表を再作成する例を示しています。

または、最新のリリースで導入されるかもしれないどの新しい列または表も追加せずに、既存の表の使用を続けることもできます。既存の表を更新するには、表を作成する際に使用された 登録済み XML スキーマ・ファイルの名前を EVMON_FORMAT_UE_TO_TABLES プロシージャーの *xsrobjectname* パラメーターとして指定しなければなりません。 404 ページの『例 7: リリース更新でそれまでのリレーショナル表を使用する』は、以前のリリースからのスキーマを使用する例を示しています。

注: EVMON_FORMAT_UE_TO_TABLES により作成されたリレーショナル表に以前存在していたデータを保持している限り、フィックスパックまたは新規リリースで導入された新しい列または表を取り出すことはできません。どの新しい列を取り出す場合でも、表の再作成が必要です。

例

- 『例 1: デフォルト・パラメーターを使用する』
- 402 ページの『例 2: 異なるスキーマ下の表の使用を試みる』
- 402 ページの『例 3: 異なるスキーマ下の表の使用を試みる』
- 403 ページの『例 4: RECREATE_FORCE オプションを使用する』
- 403 ページの『例 5: フィックスパック更新に含まれる新規エレメントを取り出す』
- 403 ページの『例 6: リリース更新に含まれる新規エレメントを取り出す』
- 404 ページの『例 7: リリース更新でそれまでのリレーショナル表を使用する』

例 1: デフォルト・パラメーターを使用する

Paul という名前のユーザーが、デフォルト・パラメーターを使用して、このプロシージャーを呼び出します。サービス・クラス STUDENTS に含まれるすべてのイベントをリレーショナル表に挿入するためです。

```
EVMON_FORMAT_UE_TO_TABLES (
  'UOW', NULL, NULL, NULL, NULL, NULL, NULL, -1,
  'SELECT * FROM UOWUE
     WHERE service_subclass_name = 'STUDENTS'
     ORDER BY event_id, event_timestamp')
```

呼び出しの結果は、以下のようになります。

1. プロシージャーは、デフォルト XML スキーマ・ファイルである DB2EvmonUOW.xsd ファイルを解析して、作成するリレーショナル表のセットを特定します。
2. SQL スキーマ Paul の下にリレーショナル表が作成されます。
3. XML スキーマが XSR オブジェクト名 PAUL.EVMON_UOW_SCHEMA_SQL09070 で登録されます。
4. XSR オブジェクトが分解可能になります。
5. データが分解されて SQL スキーマ Paul 下の表に挿入されます。

例 2: 異なるスキーマ下の表の使用を試みる

前の例に続いて、Dave という名前のユーザーがこのストアード・プロシージャーを呼び出します。 *tabschema* パラメーターが Paul に設定されています。

```
EVMON_FORMAT_UE_TO_TABLES (
  'UOW', NULL, NULL, NULL, 'Paul', NULL, NULL, -1,
  'SELECT * FROM UOWTBLE
     ORDER BY event_timestamp')
```

呼び出しの結果は、以下のようになります。

1. プロシージャーは、デフォルト XML スキーマ・ファイルである DB2EvmonUOW.xsd ファイルを解析して、作成するリレーショナル表のセットを特定します。
2. プロシージャーは、スキーマ Paul の下に表を作成しようとします。しかし、リレーショナル表は現在、SQL スキーマ Paul の下に存在するため、エラーが戻されます。新規 XSR オブジェクトが登録されているときには、既存の表は使用できません。

例 3: 異なるスキーマ下の表の使用を試みる

前の例に続いて、Greg という名前のユーザーがこのストアード・プロシージャーを呼び出します。入力パラメーター *xsrschema* が Paul に設定されています。

```
EVMON_FORMAT_UE_TO_TABLES (
  'UOW', 'Paul', NULL, NULL, NULL, NULL, NULL, -1,
  'SELECT * FROM UOWTBL
     ORDER BY event_timestamp')
```

呼び出しの結果は、以下のようになります。

1. XSR オブジェクト Paul.EVMON_UOW_SCHEMA_SQL09070 が存在し、分解可能になっています。
2. Greg が表に対する INSERT 特権を持っている場合には、データが分解され、SQL スキーマ Paul 下のリレーショナル表に挿入されます。既存の XSR オブジェクト Paul.EVMON_UOW_SCHEMA_SQL09070 が使用されるので、プロシージャーへの入力パラメーターとして提供されているものではなく XSR オブジェクトから、リレーショナル表用の SQL スキーマが取得されます。

例 4: RECREATE_FORCE オプションを使用する

前の例に続いて、Paul が表を再び作成しますが、今度は表スペース MYSPACE に作成します。 Paul は RECREATE_FORCE オプションと *tbsp_name* パラメーターを指定してプロシージャーを呼び出します。

```
EVMON_FORMAT_UE_TO_TABLES (
  'UOW', NULL, NULL, NULL, NULL, 'MYSPACE', 'RECREATE_FORCE', -1,
  'SELECT * FROM UOWTBL
     ORDER BY event_timestamp')
```

呼び出しの結果は、以下のようになります。

1. XSR オブジェクト Paul.EVMON_UOW_SCHEMA_SQL09070 が存在し、分解可能になっています。
2. RECREATE_FORCE オプションが設定されます。
3. リレーショナル・ファイルのセットを特定するために、XML スキーマ・ファイルがスキーマ・リポジトリーから取り出され、解析されます。
4. 現行の表がドロップされ、MYSPACE 表スペースに再び作成されます。
5. データが分解され、新しい表に挿入されます。

例 5: フィックスパック更新に含まれる新規エレメントを取り出す

最新のフィックスパックで、ロック・イベント・モニターの XML スキーマ・ファイルに、「db2EventNew」という新規 XML エレメントが追加されました。 Paul は、XML ファイルの分解に使用する新規エレメントを取り出したいと考えています。そのために、Paul は以下の手順に従います。

1. Paul は元のリリースで作成された XSR オブジェクトをドロップします。

   ```
   DROP XSROBJECT EVMON_LOCKING_SCHEMA_SQL09070
   ```

2. プロシージャーを RECREATE_ONERROR オプションを指定して呼び出します。

   ```
   EVMON_FORMAT_UE_TO_TABLES (
     'LOCKING', NULL, NULL, NULL, NULL, NULL, 'RECREATE_ONERROR', -1,
     'SELECT * FROM LOCK
        ORDER BY event_timestamp')
   ```

   呼び出しの結果は、以下のようになります。

   a. XSR オブジェクトが存在しないので、リレーショナル表のセットを特定するために、デフォルトの DB2EvmonLocking.xsd スキーマ・ファイルが解析されます。
   b. RECREATE_ONERROR オプションが指定されたので、既存の表がドロップされて再作成されます。

例 6: リリース更新に含まれる新規エレメントを取り出す

Paul は新規 DB2 リリースにアップグレードしており、イベント・モニターの XML スキーマ・ファイルに含まれる新規変更を取り出したいと考えています。 Paul は RECREATE_ONERROR オプションを指定してプロシージャーを呼び出します。

```
EVMON_FORMAT_UE_TO_TABLES (
  'LOCKING', NULL, NULL, NULL, NULL, NULL, 'RECREATE_ONERROR', -1,
  'SELECT * FROM LOCK
     ORDER BY event_timestamp')
```

呼び出しの結果は、以下のようになります。

1. XSR オブジェクトの Paul.EVMON_LOCKING_SCHEMA_SQL1000 が存在しません。
2. RECREATE_ONERROR オプションが指定されたので、表がドロップされて再作成されます。

例 7: リリース更新でそれまでのリレーショナル表を使用する

Greg は新規 DB2 リリースにアップグレードしましたが、イベント・モニターの XML スキーマ・ファイルに含まれる新規変更を取り出したくないと考えています。Greg は、前のリリースからの *xsrobjectname* 値を指定して、プロシージャーを呼び出します。

```
EVMON_FORMAT_UE_TO_TABLES (
  'LOCKING', NULL, 'EVMON_LOCKING_SCHEMA_SQL09070', NULL, NULL, NULL, NULL, -1,
  'SELECT * FROM LOCK
     ORDER BY event_timestamp')
```

戻される情報

SQLCA を除いて、このプロシージャーからの出力はありません。 SQLCA は完了状況を示します。使用される SQLCODES は次のとおりです。

0 すべてのイベントがリレーショナル表に正常に挿入されました。

16278 1 つ以上のイベントがリレーショナル表に挿入されませんでした。 SQLCA 内のトークンに、試みられた文書の合計数と分解に成功した文書の合計数が含まれています。

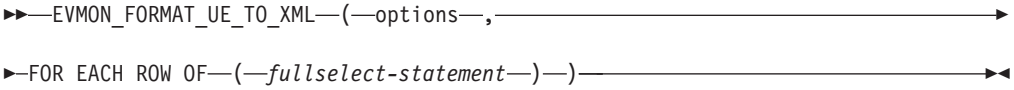
診断ファイルも作成されます。その診断ファイルの名前とロケーションは、DB2 診断パスにある db2diag ログ・ファイルに保管されます。

負の SQLCODE
エラーが発生しました。SQLCODE メッセージを調べることにより、失敗に関する追加の詳細情報が得られます。追加の診断メッセージについては、DB2 診断パスにある db2diag ログ・ファイルを参照してください。

EVMON_FORMAT_UE_TO_XML 表関数 - 不定形式イベントを XML に変換する

EVMON_FORMAT_UE_TO_XML 表関数は、バイナリー・イベントを未フォーマット・イベント表から抽出し、XML 文書にフォーマットします。

構文



スキーマは SYSPROC です。

表関数パラメーター

オプション

この表関数でサポートされるキーワード・オプションのリストを表す、タイプ VARCHAR (1024) の入力引数。

LOG_TO_FILE
: XML 文書が 100 MB より大きい場合、表関数が XML 文書をファイルに書き込むことを示します。この表関数によって行ごとに戻される各文書の最大サイズは 100 MB です。ファイルは、<xml_document_id>.xml ファイルに書き込まれます。ここで、<xml_document_id> は文書ごとに生成されるユニーク ID です。出力ファイルは、DB2 診断パス・ディレクトリーに書き込まれます。

LOG_PARTIAL_EVENTS
: 表関数がすべての部分的な (不完全な) イベントをファイルに書き込むことを示します。特定のファイルを指す診断メッセージが db2diag ログ・ファイルに挿入されます。

NULL オプションが選択されていません。

fullselect-statement
: 全選択ステートメントは、SELECT ステートメントの規則に準拠する照会です。この照会は、以下の規則に従っていなければなりません。

- この照会は、"*" 節を使用するか、未フォーマット・イベント表のすべての列を指定する必要があります。それ以外の場合、エラーが戻されます。列は、未フォーマット・イベント表の DESCRIBE ステートメントで戻されたのと同じ順序で指定されている必要があります。
- この照会は、フォーマットされていないイベント表からのみ選択を行う必要があります。
- WHERE 節は、フォーマットされていないイベント表の非 LOB 列のいずれかを使用して、イベントをフィルターで除外することができます。
- SELECT ステートメントは、大括弧で囲まれたキーワード FOR EACH ROWS OF で指定されている必要があります。

許可

EVMON_FORMAT_UE_TO_XML 関数に対する EXECUTE 特権。

フォーマットされていないイベント表に対する SELECT 特権。

例

例 1: 未フォーマット・イベント表「MYLOCKS」からすべてのイベントを照会します。

```
SELECT evmon.* FROM TABLE (
  EVMON_FORMAT_UE_TO_XML (
    NULL,
    FOR EACH ROW OF (
      select * from MYLOCKS
        order by EVENT_TIMESTAMP )))
  AS evmon;
```

例 2: 未フォーマット・イベント表「LOCK」から、過去 5 時間に発生したタイプ「LOCKWAIT」のすべてのイベントを照会します。

```
SELECT evmon.* FROM TABLE (
  EVMON_FORMAT_UE_TO_XML (
    NULL,
    FOR EACH ROW OF (
      select * from LOCK order by EVENT_TIMESTAMP
        where EVENT_TYPE = 'LOCKWAIT'
        and EVENT_TIMESTAMP >= CURRENT_TIMESTAMP – 5 hours )))
  AS evmon;
```

例 3: 未フォーマット・イベント表「UOW」から、過去 32 時間に発生したワークロード「PAYROLL」に属するすべてのイベントを取得します。任意の文書が 100 MB より大きい場合、結果をファイルに書き込みます。

```
SELECT evmon.* FROM TABLE (
  EVMON_FORMAT_UE_TO_XML(
    'LOG TO FILE',
    FOR EACH ROW OF (
      select * from UOW order by EVENT_TIMESTAMP
        where WORKLOAD_NAME = 'PAYROLL'
        and EVENT_TIMESTAMP = CURRENT_TIMESTAMP - 32 hours )))
  AS evmon;
```

例 4:「UOWEVMON」表からすべての作業単位イベントを照会し、XMLTABLE 表関数を使用して、UOW ID、UOW の開始時刻と停止時刻、および作業単位を発行したユーザーのユーザー ID を提示します。

```
SELECT EVENT.UOW_ID, EVENT.APPLICATION_ID, EVENT.SESSION_AUTHID,
  EVENT.START_TIME, EVENT.STOP_TIME
  FROM TABLE(
    EVMON_FORMAT_UE_TO_XML(
      'LOG TO FILE',
      FOR EACH ROW OF(
        select * from UOWEVMON )))
    AS UEXML,
    XMLTABLE(
      XMLNAMESPACES( DEFAULT 'http://www.ibm.com/xmlns/prod/db2/mon' ),
      '$uowevent/db2_uow_event'
      PASSING XMLPARSE( DOCUMENT UEXML.XMLREPORT ) as "uowevent"
      COLUMNS UOW_ID INTEGER PATH 'uow_id',
      MEMBER SMALLINT PATH '@member',
      APPLICATION_ID VARCHAR(128) PATH 'application_id',
      SESSION_AUTHID VARCHAR(128) PATH 'session_authid',
      START_TIME TIMESTAMP PATH 'start_time',
      STOP_TIME TIMESTAMP PATH 'stop_time'
    )
    AS EVENT
```

使用上の注意

EVMON_FORMAT_UE_TO_XML 表関数をロック・イベント・モニターおよび作業単位イベント・モニターとともに使用することで、未フォーマット・イベント表からデータを抽出することができます。

イベント・モニターのタイプに応じて、表関数は複数のレコードをフォーマットされていないイベント表から単一イベントにマップします。この場合、イベントを構成するすべてのレコードを受け取るまで、レコードはメモリー内にキャッシュされます。表関数に渡されるレコードが、作成されて表に挿入された順序になっていない場合、大量のメモリーが必要になる可能性があります。レコードがそのようにソ

ートされていない場合、表関数は複数のイベントのレコードをキャッシュに入れる必要があります。この問題を回避するには、列
(EVENT_ID、EVENT_TIMESTAMP、EVENT_TYPE、および MEMBER) が含まれる ORDER BY 節で *fullselect-statement* パラメーターを修飾します。表関数は、常に単一イベントからのレコードのみを処理およびキャッシュするため、メモリー消費量は削減されます。

戻される情報

表 95. EVMON_FORMAT_UE_TO_XML について戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
XMLID	VARCHAR(1024)	固有の文書 ID。この ID は、以下のように導出します。<event_header>_<event_id>_<event_type>_<event_timestamp>_<partition>
XMLREPORT	BLOB(100M)	単一の完全イベントを含む XML 文書。各文書の最大サイズは 100 MB です。

MON_BP_UTILIZATION - バッファー・プールに関するメトリックの取得

MON_BP_UTILIZATION 管理ビューは、現在接続されているデータベース内のすべてのバッファー・プールとすべてのデータベース・パーティションに関するヒット率、平均読み取り/書き込み時間などの主要なモニタリング・メトリックを戻します。これはバッファー・プールの使用効率の確認に役立つため、パフォーマンスをモニターするうえで重要な情報が提供されるといえます。

注: バージョン 9.7 のフィックスパック 1 以前で作成されたデータベースを使用している場合、このルーチンを実行するには、既に db2updv97 コマンドを実行済みでなければなりません。バージョン 9.7 以前に作成されたデータベースを使用している場合は、db2updv97 コマンドを実行する必要はありません (データベースのマイグレーションによって自動的にカタログ更新が実行されるため)。バージョン 9.7 にダウングレードすると、このルーチンは機能しなくなります。

許可

以下のいずれかの権限が必要です。

- MON_BP_UTILIZATION 管理ビューに対する SELECT 特権
- MON_BP_UTILIZATION 管理ビューに対する CONTROL 特権

戻される情報

表 96. MON_BP_UTILIZATION 管理ビューによって戻される情報

列名	データ・タイプ	説明またはモニター・エレメント
BP_NAME	VARCHAR(128)	bp_name - バッファー・プール名
MEMBER	SMALLINT	member - データベース・メンバー

表 96. MON_BP_UTILIZATION 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明またはモニター・エレメント
DATA_PHYSICAL_READS	BIGINT	TEMPORARY、REGULAR、および LARGE 表スペースに対して、表スペース・コンテナー (物理) から読み取られたデータ・ページの数を示します。これは (*pool_data_p_reads* + *pool_temp_data_p_reads*) として計算されます。ここで *pool_data_p_reads* と *pool_temp_data_p_reads* は以下のモニター・エレメントを表しています。 • pool_data_p_reads - バッファー・プール・データの物理読み取り • pool_temp_data_p_reads - バッファー・プール一時データの物理読み取り
DATA_HIT_RATIO_PERCENT	DECIMAL(5,2)	データのヒット率。つまり、データ・ページ要求を処理するためにデータベース・マネージャーがディスクからページをロードする必要のなかった時間のパーセンテージ。
INDEX_PHYSICAL_READS	BIGINT	TEMPORARY、REGULAR、および LARGE 表スペースに対して、表スペース・コンテナー (物理) から読み取られた索引ページの数を示します。これは (*pool_index_p_reads* + *pool_temp_index_p_reads*) として計算されます。ここで *pool_index_p_reads* + *pool_temp_index_p_reads* は以下のモニター・エレメントを表しています。 • pool_index_p_reads - バッファー・プール索引の物理読み取り • pool_temp_index_p_reads - バッファー・プール一時索引の物理読み取り
INDEX_HIT_RATIO_PERCENT	DECIMAL(5,2)	索引のヒット率。つまり、索引データ・ページ要求を処理するためにデータベース・マネージャーがディスクからページをロードする必要のなかった時間のパーセンテージ。

表 96. *MON_BP_UTILIZATION* 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明またはモニター・エレメント
XDA_PHYSICAL_READS	BIGINT	TEMPORARY、REGULAR、および LARGE 表スペースに対して、表スペース・コンテナー (物理) から読み取られた XML ストレージ・オブジェクト (XDA) 用データ・ページの数を示します。これは (*pool_xda_p_reads* + *pool_temp_xda_p_reads*) として計算されます。ここで *pool_xda_p_reads* と *pool_temp_xda_p_reads* は以下のモニター・エレメントを表しています。 • pool_xda_p_reads - バッファー・プール XDA データの物理読み取り • pool_temp_xda_p_reads - バッファー・プール一時 XDA データの物理読み取り
XDA_HIT_RATIO_PERCENT	DECIMAL(5,2)	補助ストレージ・オブジェクトのヒット率。つまり、XML ストレージ・オブジェクト (XDA) に関するデータ・ページ要求を処理するためにデータベース・マネージャーがディスクからページをロードする必要のなかった時間のパーセンテージ。

表 96. *MON_BP_UTILIZATION* 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明またはモニター・エレメント
TOTAL_PHYSICAL_READS	BIGINT	TEMPORARY、REGULAR、および LARGE 表スペースに対して、表スペース・コンテナー (物理) から読み取られたデータ・ページ、索引ページ、および XML ストレージ・オブジェクト (XDA) 用データ・ページの数を示します。 これは (*pool_data_p_reads* + *pool_temp_data_p_reads* + *pool_index_p_reads* + *pool_temp_index_p_reads* + *pool_xda_p_reads* + *pool_temp_xda_p_reads*) として計算されます。ここで *pool_data_p_reads*、*pool_temp_data_p_reads*、*pool_index_p_reads*、*pool_temp_index_p_reads*、*pool_xda_p_reads*、および *pool_temp_xda_p_reads* は以下のモニター・エレメントを表しています。 • pool_data_p_reads - バッファー・プール・データの物理読み取り • pool_temp_data_p_reads - バッファー・プール一時データの物理読み取り • pool_index_p_reads - バッファー・プール索引の物理読み取り • pool_temp_index_p_reads - バッファー・プール一時索引の物理読み取り • pool_xda_p_reads - バッファー・プール XDA データの物理読み取り • pool_temp_xda_p_reads - バッファー・プール一時 XDA データの物理読み取り

表 96. *MON_BP_UTILIZATION* 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明またはモニター・エレメント
AVG_PHYSICAL_ READ_TIME	BIGINT	すべてのタイプの表スペースに関して、表スペース・コンテナー (物理) からページを読み取るために費やされた平均時間 (ミリ秒)。 物理的読み取りの合計が 0 より大きい場合、これは *pool_read_time* / (*pool_data_p_reads* + *pool_temp_data_p_reads* + *pool_index_p_reads* + *pool_temp_index_p_reads* + *pool_xda_p_reads* + *pool_temp_xda_p_reads*) として計算されます。 ここで *pool_read_time*、*pool_data_p_reads*、*pool_temp_data_p_reads*、*pool_index_p_reads*、*pool_temp_index_p_reads*、*pool_xda_p_reads*、および *pool_temp_xda_p_reads* は以下のモニター・エレメントを表しています。 • pool_read_time - バッファー・プール物理読み取り時間の合計 • pool_data_p_reads - バッファー・プール・データの物理読み取り • pool_temp_data_p_reads - バッファー・プール一時データの物理読み取り • pool_index_p_reads - バッファー・プール索引の物理読み取り • pool_temp_index_p_reads - バッファー・プール一時索引の物理読み取り • pool_xda_p_reads - バッファー・プール XDA データの物理読み取り • pool_temp_xda_p_reads - バッファー・プール一時 XDA データの物理読み取り 物理的読み取りの合計が 0 より大きくない場合は、NULL が戻されます。

表 96. *MON_BP_UTILIZATION* 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明またはモニター・エレメント
PREFETCH_RATIO_ PERCENT	DECIMAL(5,2)	(プリフェッチを使用して) 非同期的に読み取られたページのパーセンテージ。多数のアプリケーションがプリフェッチなしで同期的にデータを読み取っている場合は、システムが最適に調整されていない可能性があります。
ASYNC_NOT_READ_ PERCENT	DECIMAL(5,2)	ディスクから非同期で読み取られ、照会でアクセスされなかったページのパーセンテージ。ディスクからバッファー・プールに非同期的に読み取られて、照会で 1 度もアクセスされないようなページが非常に多く存在する場合は、プリフェッチによってパフォーマンスが低下する可能性があります。 非同期読み取りの合計が 0 より大きい場合、これは *unread_prefetch_pages* / (*pool_async_data_reads* + *pool_async_index_reads* + *pool_async_xda_reads*) として計算されます。ここで *unread_prefetch_pages*、*pool_async_data_reads*、*pool_async_index_reads*、および *pool_async_xda_reads* は以下のモニター・エレメントを表しています。 • unread_prefetch_pages - 読み取り不能プリフェッチ・ページ • pool_async_data_reads - バッファー・プール非同期データ読み取り • pool_async_index_reads - バッファー・プール非同期索引読み取り • pool_async_xda_reads - バッファー・プール非同期 XDA データ読み取り 非同期読み取りの合計が 0 より大きくない場合は、NULL が戻されます。

表 96. *MON_BP_UTILIZATION* 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明またはモニター・エレメント
TOTAL_WRITES	BIGINT	データ、索引、または XML ストレージ・オブジェクト (XDA) 用データのページが物理的にディスクに書き込まれた回数。 これは (*pool_data_writes* + *pool_index_writes* + *pool_xda_writes*) として計算されます。ここで *pool_data_writes*、*pool_index_writes*、および *pool_xda_writes* は以下のモニター・エレメントを表しています。 • pool_data_writes - バッファー・プールへのデータの書き込み • pool_index_writes - バッファー・プール索引の書き込み • pool_xda_writes - バッファー・プール XDA データの書き込み
AVG_WRITE_TIME	BIGINT	バッファー・プールからディスクにページを物理的に書き込むために費やされた平均時間 (ミリ秒)。 書き込み操作の合計が 0 より大きい場合、これは *pool_write_time* / (*pool_data_writes* + *pool_index_writes* + *pool_xda_writes*) として計算されます。ここで *pool_write_time*、*pool_data_writes*、*pool_index_writes*、および *pool_xda_writes* は以下のモニター・エレメントを表しています。 • pool_write_time - バッファー・プール物理書き込み時間の合計 • pool_data_writes - バッファー・プールへのデータの書き込み • pool_index_writes - バッファー・プール索引の書き込み • pool_xda_writes - バッファー・プール XDA データの書き込み 書き込み操作の合計が 0 より大きくない場合は、NULL が戻されます。
SYNC_WRITES_PERCENT	DECIMAL(5,2)	同期的な書き込み操作のパーセンテージ。

MON_CONNECTION_SUMMARY - すべての接続に関するメトリックの取得

MON_CONNECTION_SUMMARY 管理ビューは、現在接続されているデータベース内のすべての接続に関する主要なメトリックを戻します。これは各接続が受け取る処理を示すため、システムの概要をモニターするのに役立ちます。

注: バージョン 9.7 のフィックスパック 1 以前で作成されたデータベースを使用している場合、このルーチンを実行するには、既に db2updv97 コマンドを実行済みでなければなりません。バージョン 9.7 以前に作成されたデータベースを使用している場合は、db2updv97 コマンドを実行する必要はありません (データベースのマイグレーションによって自動的にカタログ更新が実行されるため)。バージョン 9.7 にダウングレードすると、このルーチンは機能しなくなります。

戻されるメトリックは、データベースの全メンバーにおける、識別された接続によってサブミットされた要求に関するすべてのメトリックの累計を表します。

スキーマは SYSIBMADM です。

許可

以下のいずれかの権限が必要です。

- MON_CONNECTION_SUMMARY 管理ビューに対する SELECT 特権
- MON_CONNECTION_SUMMARY 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

戻される情報

表 97. MON_CONNECTION_SUMMARY 管理ビューによって戻される情報

列名	データ・タイプ	説明またはモニター・エレメント
APPLICATION_HANDLE	BIGINT	application_handle - アプリケーション・ハンドル
APPLICATION_NAME	VARCHAR(128)	appl_name - アプリケーション名
APPLICATION_ID	VARCHAR(128)	appl_id - アプリケーション ID
SESSION_AUTH_ID	VARCHAR(128)	session_auth_id - セッション許可 ID
TOTAL_APP_COMMITS	BIGINT	指定されたサービス・サブクラスに関する、データベースの全メンバーでのアプリケーション・コミットの総数。

表 97. MON_CONNECTION_SUMMARY 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明またはモニター・エレメント
TOTAL_APP_ROLLBACKS	BIGINT	指定されたサービス・サブクラスに関する、データベースの全メンバーでのアプリケーション・ロールバックの総数。
ACT_COMPLETED_TOTAL	BIGINT	指定されたサービス・サブクラスに関する、データベースの全メンバーにおける、正常に完了した任意のネスト・レベルのコーディネーター・アクティビティーの総数。
APP_RQSTS_COMPLETED_TOTAL	BIGINT	指定されたサービス・サブクラスに関する、データベースの全メンバーで正常に完了した外部 (アプリケーション) 要求の総数。
AVG_RQST_CPU_TIME	BIGINT	正常に完了したすべての外部要求によって費やされた平均 CPU 時間 (マイクロ秒)。これは、ユーザーとシステムの両方の CPU 時間の合計を表します。
ROUTINE_TIME_RQST_PERCENT	DECIMAL(5,2)	データベース・サーバーが要求を処理するのに費やした時間のうち、ユーザー・ルーチンの実行に費やした時間のパーセンテージ。
RQST_WAIT_TIME_PERCENT	DECIMAL(5,2)	要求を処理するのに費やされた時間のうち、DB2 データベース・サーバー内部での待機に費やされた時間のパーセンテージ。
ACT_WAIT_TIME_PERCENT	DECIMAL(5,2)	アクティビティーの実行に費やされた時間のうち、DB2 データベース・サーバー内部での待機に費やされた時間のパーセンテージ。

表 97. MON_CONNECTION_SUMMARY 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明またはモニター・エレメント
IO_WAIT_TIME_PERCENT	DECIMAL(5,2)	DB2 データベース・サーバー内部での待機に費やされた時間のうち、入出力操作に費やされた時間のパーセンテージ。これには、直接読み取り/直接書き込みの実行に費やされた時間に加えて、表スペースからバッファー・プールへのデータおよび索引ページの読み取りや元のディスクへの書き込みに費やされた時間が含まれます。
LOCK_WAIT_TIME_PERCENT	DECIMAL(5,2)	DB2 データベース・サーバー内部での待機に費やされた時間のうち、ロック待機に費やされた時間のパーセンテージ。
AGENT_WAIT_TIME_PERCENT	DECIMAL(5,2)	DB2 データベース・サーバー内部での待機に費やされた時間のうち、コンセントレーター構成のもとでエージェントを待機するためにキューに入れられたアプリケーションによって費やされた時間のパーセンテージ。
NETWORK_WAIT_TIME_PERCENT	DECIMAL(5,2)	DB2 データベース・サーバー内部での待機に費やされた時間のうち、クライアント/サーバー通信のために費やされた時間のパーセンテージ。これには、TCP/IP または IPC プロトコルを使ってデータを送受信するのに費やされた時間が含まれます。
SECTION_PROC_TIME_PERCENT	DECIMAL(5,2)	要求をアクティブに処理するデータベース・サーバーが、セクションの実行に費やした時間のパーセンテージ。これには、ソートの実行に費やされた時間が含まれます。
SECTION_SORT_PROC_TIME_PERCENT	DECIMAL(5,2)	要求をアクティブに処理するデータベース・サーバーが、セクションの実行中にソートの実行に費やした時間のパーセンテージ。

表 97. MON_CONNECTION_SUMMARY 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明またはモニター・エレメント
COMPILE_PROC_ TIME_PERCENT	DECIMAL(5,2)	要求をアクティブに処理するデータベース・サーバーが、SQL ステートメントのコンパイルに費やした時間のパーセンテージ。これには、明示的および暗黙的コンパイル時間が含まれます。
TRANSACT_END_PROC _TIME_PERCENT	DECIMAL(5,2)	要求をアクティブに処理するデータベース・サーバーが、コミット処理の実行またはトランザクションのロールバックに費やした時間のパーセンテージ。
UTILS_PROC_ TIME_PERCENT	DECIMAL(5,2)	要求をアクティブに処理するデータベース・サーバーが、ユーティリティーの実行に費やした時間のパーセンテージ。これには、runstats、再編成、およびロード操作の実行が含まれます。
AVG_LOCK_WAITS _PER_ACT	BIGINT	各コーディネーター・アクティビティー (成功およびアボート) に対して、アプリケーションまたは接続がロックを待機した平均回数。
AVG_LOCK_TIMEOUTS _PER_ACT	BIGINT	各コーディネーター・アクティビティー (成功およびアボート) に対して、オブジェクトのロック要求がタイムアウトになった平均回数。
AVG_DEADLOCKS_ PER_ACT	BIGINT	コーディネーター・アクティビティー (成功およびアボート) ごとのデッドロックの平均数。
AVG_LOCK_ESCALS _PER_ACT	BIGINT	各コーディネーター・アクティビティー (成功およびアボート) に対して、いくつかの行ロックから表ロックにロックがエスカレートした平均回数。
ROWS_READ_PER_ ROWS_RETURNED	BIGINT	アプリケーションに戻された各行に対して、表から読み取られた平均行数。

表 97. MON_CONNECTION_SUMMARY 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明またはモニター・エレメント
TOTAL_BP_HIT_RATIO_PERCENT	DECIMAL(5,2)	XML ストレージ・オブジェクト (XDA) の要求を含めて、データまたは索引ページの要求を処理するためにデータベース・マネージャーがディスクからページをロードする必要のなかった時間のパーセンテージ。

MON_CURRENT_SQL - 全メンバーの全アクティビティーに関する主要なメトリックの取得

MON_CURRENT_SQL 管理ビューは、データベースの全メンバーに対してサブミットされた、まだ未完了であるすべてのアクティビティーに関する主要なメトリックを戻します。これには、現在接続されているデータベースで現在実行中の SQL ステートメント (静的および動的) のポイント・イン・タイム・ビューが含まれます。

注: バージョン 9.7 のフィックスパック 1 以前で作成されたデータベースを使用している場合、このルーチンを実行するには、既に db2updv97 コマンドを実行済みでなければなりません。バージョン 9.7 以前に作成されたデータベースを使用している場合は、db2updv97 コマンドを実行する必要はありません (データベースのマイグレーションによって自動的にカタログ更新が実行されるため)。バージョン 9.7 にダウングレードすると、このルーチンは機能しなくなります。

MON_CURRENT_SQL 管理ビューを使用すると、長く実行されているアクティビティーを識別し、パフォーマンス上の問題を防ぐことができます。

このビューは、個々のメンバーではなく、コーディネーターの観点を表します。

スキーマは SYSIBMADM です。

許可

以下のいずれかの権限が必要です。

- MON_CURRENT_SQL 管理ビューに対する SELECT 特権
- MON_CURRENT_SQL 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

戻される情報

表 98. MON_CURRENT_SQL 管理ビューによって戻される情報

列名	データ・タイプ	説明またはモニター・エレメント
COORD_MEMBER	SMALLINT	coord_member - コーディネーター・メンバー

表 98. MON_CURRENT_SQL 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明またはモニター・エレメント
APPLICATION_HANDLE	BIGINT	application_handle - アプリケーション・ハンドル
APPLICATION_NAME	VARCHAR(128)	appl_name - アプリケーション名
SESSION_AUTH_ID	VARCHAR(128)	session_auth_id - セッション許可 ID
CLIENT_APPLNAME	VARCHAR(128)	CURRENT CLIENT_APPLNAME 特殊レジスター
ELAPSED_TIME_SEC	INTEGER	このアクティビティーが開始してから経過した時間 (秒)。アクティビティーがシステムに入ったが、キューに入っているためまだ実行開始済みでない場合は、この列の値が NULL です。
ACTIVITY_STATE	VARCHAR(32)	activity_state - アクティビティー状態
ACTIVITY_TYPE	VARCHAR(32)	activity_type - アクティビティー・タイプ
TOTAL_CPU_TIME	BIGINT	total_cpu_time - 合計 CPU 時間
ROWS_READ	BIGINT	rows_read - 読み取り行数
ROWS_RETURNED	BIGINT	rows_returned - 戻り行数
QUERY_COST_ESTIMATE	BIGINT	query_cost_estimate - 照会コストの見積もり
DIRECT_READS	BIGINT	direct_reads - データベースからの直接読み取り
DIRECT_WRITES	BIGINT	direct_writes - データベースへの直接書き込み
STMT_TEXT	CLOB(2MB)	stmt_text - SQL 動的ステートメント・テキスト

MON_CURRENT_UOW - すべての作業単位に関するメトリックの取得

MON_CURRENT_UOW 管理ビューは、データベースの全メンバーに対してサブミットされたすべての作業単位に関する主要なメトリックを戻します。長く実行されている作業単位が識別されるため、これを使用してパフォーマンス上の問題を防ぐことができます。

注: バージョン 9.7 のフィックスパック 1 以前で作成されたデータベースを使用している場合、このルーチンを実行するには、既に db2updv97 コマンドを実行済みでなければなりません。バージョン 9.7 以前に作成されたデータベースを使用している場合は、db2updv97 コマンドを実行する必要はありません (データベースのマイ

グレーションによって自動的にカタログ更新が実行されるため)。バージョン 9.7 にダウングレードすると、このルーチンは機能しなくなります。

MON_CURRENT_UOW ビューは、個々のメンバーではなく、コーディネーターの観点を表します。

スキーマは SYSIBMADM です。

許可

以下のいずれかの権限が必要です。

- MON_CURRENT_UOW 管理ビューに対する SELECT 特権
- MON_CURRENT_UOW 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

例

以下の例では、1 分を超えて実行されたすべての作業単位に関するアプリケーション・ハンドル、作業単位 ID、経過時間、読み取られた行と戻された行の総数が取得されます。

```
SELECT APPLICATION_HANDLE AS APPL_HANDLE,
       UOW_ID, ELAPSED_TIME_SEC,
       TOTAL_ROWS_MODIFIED AS TOTAL_READ,
       TOTAL_ROWS_MODIFIED AS TOTAL_MODIFIED
  FROM MON_CURRENT_UOW
 WHERE ELAPSED_TIME_SEC > 60
 ORDER BY ELAPSED_TIME_SEC DESC
```

以下はこの照会の出力例です。

```
APPL_HANDLE UOW_ID ELAPSED_TIME_SEC TOTAL_READ TOTAL_MODIFIED
----------- ------ ---------------- ---------- --------------
        254      1              750      87460              0
         61      1              194        108              0
        145      4               82          0             34

  3 record(s) selected.
```

戻される情報

表 99. MON_CURRENT_UOW 管理ビューによって戻される情報

列名	データ・タイプ	説明またはモニター・エレメント
COORD_MEMBER	SMALLINT	coord_member - コーディネーター・メンバー
UOW_ID	INTEGER	uow_id - 作業単位 ID
APPLICATION_HANDLE	BIGINT	application_handle - アプリケーション・ハンドル
APPLICATION_NAME	VARCHAR(128)	appl_name - アプリケーション名
SESSION_AUTH_ID	VARCHAR(128)	session_auth_id - セッション許可 ID

表 99. MON_CURRENT_UOW 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明またはモニター・エレメント
CLIENT_APPLNAME	VARCHAR(255)	CURRENT CLIENT_APPLNAME 特殊レジスター
ELAPSED_TIME_SEC	INTEGER	この作業単位が開始してから経過した時間 (秒)。アクティビティーがシステムに入ったが、キューに入っているためまだ実行開始済みでない場合は、この列の値が NULL です。
WORKLOAD_OCCURRENCE_STATE	VARCHAR(32)	workload_occurrence_state - ワークロード・オカレンスの状態
TOTAL_CPU_TIME	BIGINT	total_cpu_time - 合計 CPU 時間
TOTAL_ROWS_MODIFIED	BIGINT	挿入、更新、または削除された行の総数。
TOTAL_ROWS_READ	BIGINT	表から読み取られた行の総数。
TOTAL_ROWS_RETURNED	BIGINT	選択されてアプリケーションに戻された行の総数。

MON_DB_SUMMARY - データベースの全メンバーにわたる累計メトリックの取得

MON_DB_SUMMARY 管理ビューは、現在接続されているデータベース内のすべてのサービス・クラスにわたって集計された主要なメトリックを戻します。データベースの簡潔な要約が示されるため、システムの概要をモニターするのに役立ちます。

注: バージョン 9.7 のフィックスパック 1 以前で作成されたデータベースを使用している場合、このルーチンを実行するには、既に db2updv97 コマンドを実行済みでなければなりません。バージョン 9.7 以前に作成されたデータベースを使用している場合は、db2updv97 コマンドを実行する必要はありません (データベースのマイグレーションによって自動的にカタログ更新が実行されるため)。バージョン 9.7 にダウングレードすると、このルーチンは機能しなくなります。

戻されるメトリックは、データベースの全メンバーにわたるメトリックの累計を表します。

スキーマは SYSIBMADM です。

許可

以下のいずれかの権限が必要です。

- MON_DB_SUMMARY 管理ビューに対する SELECT 特権
- MON_DB_SUMMARY 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

戻される情報

表 100. MON_DB_SUMMARY 管理ビューによって戻される情報

列名	データ・タイプ	説明またはモニター・エレメント
TOTAL_APP_COMMITS	BIGINT	指定されたサービス・サブクラスに関する、データベースの全メンバーでのアプリケーション・コミットの総数。
TOTAL_APP_ROLLBACKS	BIGINT	指定されたサービス・サブクラスに関する、データベースの全メンバーでのアプリケーション・ロールバックの総数。
ACT_COMPLETED_TOTAL	BIGINT	指定されたサービス・サブクラスに関する、データベースの全メンバーにおける、正常に完了した任意のネスト・レベルのコーディネーター・アクティビティーの総数。
APP_RQSTS_COMPLETED_TOTAL	BIGINT	指定されたサービス・サブクラスに関する、データベースの全メンバーで正常に完了した外部 (アプリケーション) 要求の総数。
AVG_RQST_CPU_TIME	BIGINT	正常に完了したすべての外部要求によって費やされた平均 CPU 時間 (マイクロ秒)。これは、ユーザーとシステムの両方の CPU 時間の合計を表します。
ROUTINE_TIME_RQST_PERCENT	DECIMAL(5,2)	データベース・サーバーが要求を処理するのに費やした時間のうち、ユーザー・ルーチンの実行に費やした時間のパーセンテージ。
RQST_WAIT_TIME_PERCENT	DECIMAL(5,2)	要求を処理するのに費やされた時間のうち、DB2 データベース・サーバー内部での待機に費やされた時間のパーセンテージ。

表 100. MON_DB_SUMMARY 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明またはモニター・エレメント
ACT_WAIT_TIME_PERCENT	DECIMAL(5,2)	アクティビティーの実行に費やされた時間のうち、DB2 データベース・サーバー内部での待機に費やされた時間のパーセンテージ。
IO_WAIT_TIME_PERCENT	DECIMAL(5,2)	DB2 データベース・サーバー内部での待機に費やされた時間のうち、入出力操作に費やされた時間のパーセンテージ。これには、直接読み取り/直接書き込みの実行に費やされた時間に加えて、表スペースからバッファー・プールへのデータおよび索引ページの読み取りや元のディスクへの書き込みに費やされた時間が含まれます。
LOCK_WAIT_TIME_PERCENT	DECIMAL(5,2)	DB2 データベース・サーバー内部での待機に費やされた時間のうち、ロック待機に費やされた時間のパーセンテージ。
AGENT_WAIT_TIME_PERCENT	DECIMAL(5,2)	DB2 データベース・サーバー内部での待機に費やされた時間のうち、コンセントレーター構成のもとでエージェントを待機するためにキューに入れられたアプリケーションによって費やされた時間のパーセンテージ。
NETWORK_WAIT_TIME_PERCENT	DECIMAL(5,2)	DB2 データベース・サーバー内部での待機に費やされた時間のうち、クライアント/サーバー通信のために費やされた時間のパーセンテージ。これには、TCP/IP または IPC プロトコルを使ってデータを送受信するのに費やされた時間が含まれます。
SECTION_PROC_TIME_PERCENT	DECIMAL(5,2)	要求をアクティブに処理するデータベース・サーバーが、セクションの実行に費やした時間のパーセンテージ。これには、ソートの実行に費やされた時間が含まれます。

表 100. MON_DB_SUMMARY 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明またはモニター・エレメント
SECTION_SORT_ PROC_TIME_PERCENT	DECIMAL(5,2)	要求をアクティブに処理するデータベース・サーバーが、セクションの実行中にソートの実行に費やした時間のパーセンテージ。
COMPILE_PROC_ TIME_PERCENT	DECIMAL(5,2)	要求をアクティブに処理するデータベース・サーバーが、SQL ステートメントのコンパイルに費やした時間のパーセンテージ。これには、明示的および暗黙的コンパイル時間が含まれます。
TRANSACT_END_PROC _TIME_PERCENT	DECIMAL(5,2)	要求をアクティブに処理するデータベース・サーバーが、コミット処理の実行またはトランザクションのロールバックに費やした時間のパーセンテージ。
UTILS_PROC_ TIME_PERCENT	DECIMAL(5,2)	要求をアクティブに処理するデータベース・サーバーが、ユーティリティーの実行に費やした時間のパーセンテージ。これには、runstats、再編成、およびロード操作の実行が含まれます。
AVG_LOCK_WAITS _PER_ACT	BIGINT	各コーディネーター・アクティビティー (成功およびアボート) に対して、アプリケーションまたは接続がロックを待機した平均回数。
AVG_LOCK_TIMEOUTS _PER_ACT	BIGINT	各コーディネーター・アクティビティー (成功およびアボート) に対して、オブジェクトのロック要求がタイムアウトになった平均回数。
AVG_DEADLOCKS_ PER_ACT	BIGINT	コーディネーター・アクティビティー (成功およびアボート) ごとのデッドロックの平均数。
AVG_LOCK_ESCALS _PER_ACT	BIGINT	各コーディネーター・アクティビティー (成功およびアボート) に対して、いくつかの行ロックから表ロックにロックがエスカレートした平均回数。

表 100. MON_DB_SUMMARY 管理ビューによって戻される情報 (続き)

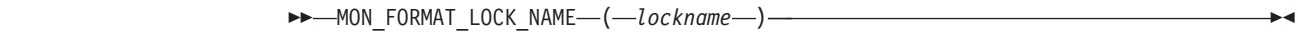
列名	データ・タイプ	説明またはモニター・エレメント
ROWS_READ_PER_ ROWS_RETURNED	BIGINT	アプリケーションに戻された各行に対して、表から読み取られた平均行数。
TOTAL_BP_HIT_ RATIO_PERCENT	DECIMAL(5,2)	XML ストレージ・オブジェクト (XDA) の要求を含めて、データまたは索引ページの要求を処理するためにデータベース・マネージャーがディスクからページをロードする必要のなかった時間のパーセンテージ。

MON_FORMAT_LOCK_NAME - 内部ロック名のフォーマット設定と詳細の出力

MON_FORMAT_LOCK_NAME 表関数は、内部ロック名をフォーマット設定し、ロックについての詳細情報を行ベースの形式で戻します。戻される各行は、その特定のロックに関する *key-value* の組で構成されます。

注: バージョン 9.7 のフィックスパック 1 以前で作成されたデータベースを使用している場合、このルーチンを実行するには、既に db2updv97 コマンドを実行済みでなければなりません。バージョン 9.7 以前に作成されたデータベースを使用している場合は、db2updv97 コマンドを実行する必要はありません (データベースのマイグレーションによって自動的にカタログ更新が実行されるため)。バージョン 9.7 にダウングレードすると、このルーチンは機能しなくなります。

ロックについての情報を取得するには MON_FORMAT_LOCK_NAME、MON_GET_LOCKS、および MON_GET_APPL_LOCKWAIT 表関数を使用してください。SNAPLOCKWAIT 管理ビューと SNAP_GET_LOCKWAIT 表関数、および SNAPLOCK 管理ビューと SNAP_GET_LOCK 表関数はバージョン 9.7 フィックスパック 1 で非推奨になりました。



スキーマは SYSPROC です。

表関数パラメーター

lockname
フォーマット設定されるロックの内部バイナリー名を指定する、タイプ VARCHAR (32) の入力引数。 NULL 値を指定した場合、結果としてエラー SQL0171N が戻されます。

許可

以下の特権が必要です。

- MON_FORMAT_LOCK_NAME 表関数に対する EXECUTE 特権

例

内部ロック名は、db2diag log ログ・ファイルへの書き込み、**lock_name** モニター・エレメントの値などのさまざまな状況で戻されます。以下の例は、MON_FORMAT_LOCK_NAME 表関数を使ってロックについての詳細情報を検出する方法を示しています。この場合はロック名 0000000E00000000000B00C152 を使用します。

```
SELECT SUBSTR(NAME,1,20) AS NAME,
       SUBSTR(VALUE,1,50) AS VALUE
       FROM
       TABLE( MON_FORMAT_LOCK_NAME('0000000E00000000000B00C152')) as LOCK
```

以下の出力が戻されます。

```
NAME                 VALUE
----------------     ----------------------------------------------------------
LOCK_OBJECT_TYPE     ROW
TBSP_NAME            SYSCATSPACE
TABSCHEMA            SYSIBM
TABNAME              SYSPLANDEP
ROWID                0000000B00C1

 4 record(s) selected
```

戻される情報

表 101. MON_FORMAT_LOCK_NAME 表関数によって戻される情報

列名	データ・タイプ	説明
NAME	VARCHAR(256)	ロック名のエレメント。詳しくは、下記の表を参照してください。
VALUE	VARCHAR(1024)	エレメントの値。

指定されたロック名を構成するすべてのエレメントが戻されるわけではありません。関連のある *key-value* の組だけが戻されます。

戻される可能性のあるエレメントは次のとおりです。

表 102. 戻される可能性のあるモニター・エレメント

エレメント名	説明	可能な値またはモニター・エレメント
LOCK_OBJECT_TYPE	ロック対象タイプ	lock_object_type - ロック対象タイプ 可能な値については、『lock_object_type - 待機中のロック対象タイプ』モニター・エレメントを参照してください。

表 102. 戻される可能性のあるモニター・エレメント (続き)

エレメント名	説明	可能な値またはモニター・エレメント
DATA_PARTITION_ID	情報を戻す対象となるデータ・パーティションの ID。このエレメントは、パーティション表およびパーティション化索引にのみ該当します。ロック・レベル情報が戻されるとき、値 -1 は表全体へのアクセスを制御するロックを表します。	data_partition_id - データ・パーティション ID
TBSP_NAME	表スペースの名前	tablespace_name - 表スペース名
TABSCHEMA	表のスキーマ	table_schema - 表スキーマ名
TABNAME	表の名前	table_name - 表名
ROWID	表の行 ID	-
PAGEID	ページ ID	-
WORKLOAD_NAME	ワークロードの名前	workload_name - ワークロード名
STORAGE_GRP_ID	ストレージ・グループ ID	-
BUFFERPOOL_NAME	バッファー・プールの名前	-
FED_SERVER_NAME	フェデレーション・サーバーの名前	-
FED_USER_NAME	フェデレーション・ユーザー・マッピングの名前	-

表 102. 戻される可能性のあるモニター・エレメント (続き)

エレメント名	説明	可能な値またはモニター・エレメント
SEQ_OPERATION	シーケンス・ロックを要求している操作	可能な値は以下のとおりです。 • AUTONOMIC_POLICIES • CATALOG_ARRAY • DESCRIBE • INIT_EVMON • INIT_PACKAGE • INIT_AUDIT • PACKAGE_CREATION • INIT_ROUTINE_ID • INIT_ROLE_ID • TEMP_TBSPACE • AUDIT_DDL • VERSION_TIMES • WLM • TRUSTED_CTX • INIT_TRUSTED_CTX • STATIC_STMT • USER_TEMP_TBSPACE
CONTAINER_ID	コンテナー ID	-
STMT_UID	ステートメント ID	-
PACKAGE_TOKEN	パッケージ・トークン	-
INTERNAL	内部使用のために予約済み	-

MON_FORMAT_XML_COMPONENT_TIMES_BY_ROW - フォーマット設定された行ベースのコンポーネント時間の取得

MON_FORMAT_XML_COMPONENT_TIMES_BY_ROW 表関数は、XML メトリック文書に含まれるコンポーネント時間に関するフォーマット設定された行ベースの出力を戻します。

注: バージョン 9.7 のフィックスパック 1 以前で作成されたデータベースを使用している場合、このルーチンを実行するには、既に db2updv97 コマンドを実行済みでなければなりません。バージョン 9.7 以前に作成されたデータベースを使用している場合は、db2updv97 コマンドを実行する必要はありません (データベースのマイグレーションによって自動的にカタログ更新が実行されるため)。バージョン 9.7 にダウングレードすると、このルーチンは機能しなくなります。

構文

スキーマは SYSPROC です。

表関数パラメーター

xmldoc

system_metrics または activity_metrics エレメントを持つ XML 文書が含まれる、タイプ BLOB(100M) の入力引数。これらのエレメントを持つ XML 文書は、以下のソースから入手可能です。

- いずれかの MON_GET_*_DETAILS 表関数からの戻り。
- 統計およびアクティビティー・イベント・モニターからのメトリック列出力。
- 作業単位またはパッケージ・キャッシュ・イベント・モニターからのフォーマット設定された出力。

許可

MON_FORMAT_XML_COMPONENT_TIMES_BY_ROW 関数に対する EXECUTE 特権。

例

以下の例は、サービス・サブクラスに関する DB2 データベース・マネージャー内のコンポーネント時間の明細を戻します。特定のコンポーネントで費やされた合計時間、およびコンポーネント内で (待機ではなく) 実際の処理に費やされた時間の合計が示されます。

```
SELECT SUBSTR(T.SERVICE_SUPERCLASS_NAME,1,19) AS SUPERCLASS,
  SUBSTR(T.SERVICE_SUBCLASS_NAME,1,19) AS SUBCLASS,
  T.MEMBER,
  SUBSTR(COMP.METRIC_NAME,1,15) AS METRIC_NAME
  SUBSTR(COMP.PARENT_METRIC_NAME,1,15) AS PARENT_NAME
  COMP.TOTAL_TIME_VALUE AS TOTAL_TIME,
  COMP.PROC_TIME_VALUE AS TOTAL_PROC_TIME,
   COMP.COUNT
 FROM TABLE (MON_GET_SERVICE_SUBCLASS_DETAILS(NULL,
              NULL,-2)) AS T,
       TABLE(MON_FORMAT_XML_COMPONENT_TIMES_BY_ROW(
                                            T.DETAILS
                                           )) AS COMP

 WHERE COMP.PARENT_METRIC_NAME IS NOT NULL;
```

以下はこの照会の出力例です。

```
SUPERCLASS          SUBCLASS           MEMBER METRIC_NAME     PARENT_NAME    ...
------------------- ------------------ ------ --------------- ---------------...
MYSC                MYSSC                   0 TOTAL_COMPILE_T TOTAL_RQST_TIME...
MYSC                MYSSC                   0 TOTAL_IMPLICIT_ TOTAL_RQST_TIME...
MYSC                MYSSC                   0 TOTAL_SECTION_T TOTAL_RQST_TIME...
MYSC                MYSSC                   0 TOTAL_COMMIT_TI TOTAL_RQST_TIME...
MYSC                MYSSC                   0 TOTAL_ROLLBACK_ TOTAL_RQST_TIME...
MYSC                MYSSC                   0 TOTAL_RUNSTATS_ TOTAL_RQST_TIME...
MYSC                MYSSC                   0 TOTAL_REORG_TIM TOTAL_RQST_TIME...
```

```
MYSC                MYSSC                    0 TOTAL_LOAD_TIME TOTAL_RQST_TIME...
MYSC                MYSSC                    0 TOTAL_SECTION_S TOTAL_SECTION_T...

  9 record(s) selected.
```

以下は、この照会の出力例の続きです。

```
...TOTAL_TIME           TOTAL_PROC_TIME      COUNT
...-------------------- -------------------- --------------------
...                  100                  100                    1
...                    0                    0                    0
...                 1253                  953                    0
...                  213                  153                    0
...                    0                    0                    0
...                    0                    0                    0
...                    0                    0                    0
...                    0                    0                    0
...                    0                    0                    0

  9 record(s) selected.
```

戻される情報

表 103. MON_FORMAT_XML_COMPONENT_TIMES_BY_ROW で戻される情報

列名	データ・タイプ	説明
METRIC_NAME	VARCHAR(128)	合計時間メトリック値のユニーク ID。
PROC_METRIC_NAME	VARCHAR(128)	処理時間メトリックのユニーク ID。
TOTAL_TIME_VALUE	BIGINT	metric_name に対応する合計時間値 (ミリ秒)。
PROC_TIME_VALUE	BIGINT	proc_metric_name に対応する処理時間値 (ミリ秒)
COUNT	BIGINT	このタイプのインターバルのオカレンス数。
PARENT_METRIC_NAME	VARCHAR(128)	親である合計時間メトリックの ID。そのメトリックの値には total_time_value がサブセットとして含まれます。
PARENT_PROC_ METRIC_NAME	VARCHAR(128)	親である処理時間メトリックの ID。そのメトリックの値には proc_time_value がサブセットとして含まれます。

タイプ *system_metrics* のエレメントを含む XML 文書は、以下のインターフェースから生成されます。

- MON_GET_CONNECTION_DETAILS
- MON_GET_SERVICE_SUBCLASS_DETAILS
- MON_GET_UNIT_OF_WORK_DETAILS
- MON_GET_WORKLOAD_DETAILS
- STATISTICS イベント・モニターからの DETAILS_XML 列
- UNIT OF WORK イベント・モニターに対する EVMON_FORMAT_UE_TO_TABLES によって生成される METRICS 列
- UNIT OF WORK イベント・モニターに対する EVMON_FORMAT_UE_TO_XML の XMLREPORT 列

この場合に XML 文書から戻されるメトリックの種類とその親メトリックについては、431 ページの表 104 を参照してください。

表 104. system_metrics エレメント・タイプを含む XML 文書に対する MON_FORMAT_XML_COMPONENT_TIMES_BY_ROW によって戻されるメトリック名

メトリック名	処理メトリック名	親メトリック名	親処理メトリック名	メトリックについての説明またはモニター・エレメント
TOTAL_RQST_TIME	NULL	NULL	NULL	total_rqst_time - 合計要求時間
TOTAL_COMPILE_TIME	TOTAL_COMPILE_PROC_TIME	TOTAL_RQST_TIME	TOTAL_RQST_TIME	total_compile_time - コンパイル時間の合計
TOTAL_IMPLICIT_COMPILE_TIME	TOTAL_IMPLICIT_COMPILE_PROC_TIME	TOTAL_RQST_TIME	TOTAL_RQST_TIME	total_implicit_compile_time - 暗黙的コンパイル時間の合計
TOTAL_SECTION_TIME	TOTAL_SECTION_PROC_TIME	TOTAL_RQST_TIME	TOTAL_RQST_TIME	total_section_time - セクション時間の合計
TOTAL_COMMIT_TIME	TOTAL_COMMIT_PROC_TIME	TOTAL_RQST_TIME	TOTAL_RQST_TIME	total_commit_time - コミット時間の合計
TOTAL_ROLLBACK_TIME	TOTAL_ROLLBACK_PROC_TIME	TOTAL_RQST_TIME	TOTAL_RQST_TIME	total_rollback_time - ロールバック時間の合計
TOTAL_ROUTINE_USER_CODE_TIME	TOTAL_ROUTINE_USER_CODE_PROC_TIME	TOTAL_RQST_TIME	TOTAL_RQST_TIME	total_routine_user_code_time - ルーチン・ユーザー・コード時間の合計
TOTAL_RUNSTATS_TIME	TOTAL_RUNSTATS_PROC_TIME	TOTAL_RQST_TIME	TOTAL_RQST_TIME	total_runstats_time - ランタイム統計時間の合計
TOTAL_REORG_TIME	TOTAL_REORG_PROC_TIME	TOTAL_RQST_TIME	TOTAL_RQST_TIME	total_reorg_time - 再編成時間の合計
TOTAL_LOAD_TIME	TOTAL_LOAD_PROC_TIME	TOTAL_RQST_TIME	TOTAL_RQST_TIME	total_load_time - ロード時間の合計
TOTAL_SECTION_SORT_TIME	TOTAL_SECTION_SORT_PROC_TIME	TOTAL_SECTION_TIME	TOTAL_SECTION_PROC_TIME	total_section_sort_time - セクション・ソート時間合計

タイプ *activity_metrics* のエレメントを含む XML 文書は、以下のインターフェースから生成されます。

- MON_GET_ACTIVITY_DETAILS
- MON_GET_PKG_CACHE_STMT_DETAILS
- ACTIVITY イベント・モニターからの DETAILS_XML 列
- PACKAGE CACHE イベント・モニターに対する EVMON_FORMAT_UE_TO_TABLES によって生成される METRICS 列

- PACKAGE CACHE イベント・モニターに対する EVMON_FORMAT_UE_TO_XML の XMLREPORT 列

この場合に XML 文書から戻されるメトリックの種類とその親メトリックについては、表 105 を参照してください。

表 105. activity_metrics エレメント・タイプを含む XML 文書に対する MON_FORMAT_XML_COMPONENT_TIMES_BY_ROW によって戻されるメトリック名

メトリック名	処理メトリック名	親メトリック名	親処理メトリック名	説明またはモニター・エレメント
STMT_EXEC_TIME	NULL	NULL	NULL	stmt_exec_time - ステートメントの実行時間
TOTAL_ROUTINE_TIME	NULL	STMT_EXEC_TIME	NULL	total_routine_time - ルーチン時間の合計
TOTAL_ROUTINE_NON_SECT_TIME	TOTAL_ROUTINE_NON_SECT_PROC_TIME	TOTAL_ROUTINE_TIME	STMT_EXEC_TIME	total_routine_non_sect_time - 非セクション・ルーチン実行時間
TOTAL_ROUTINE_USER_CODE_TIME	TOTAL_ROUTINE_USER_CODE_PROC_TIME	TOTAL_ROUTINE_NON_SECT_TIME	TOTAL_ROUTINE_NON_SECT_PROC_TIME	total_routine_user_code_time - ルーチン・ユーザー・コード時間の合計
TOTAL_SECTION_TIME	TOTAL_SECTION_PROC_TIME	STMT_EXEC_TIME	STMT_EXEC_TIME	total_section_time - セクション時間の合計
TOTAL_SECTION_SORT_TIME	TOTAL_SECTION_SORT_PROC_TIME	TOTAL_SECTION_TIME	TOTAL_SECTION_PROC_TIME	total_section_sort_time - セクション・ソート時間合計

MON_FORMAT_XML_METRICS_BY_ROW - すべてのメトリックに関するフォーマット設定された行ベースの出力の取得

MON_FORMAT_XML_METRICS_BY_ROW 表関数は、XML メトリック文書に含まれるすべてのメトリックに関するフォーマット設定された行ベースの出力を戻します。

注: バージョン 9.7 のフィックスパック 1 以前で作成されたデータベースを使用している場合、このルーチンを実行するには、既に db2updv97 コマンドを実行済みでなければなりません。バージョン 9.7 以前に作成されたデータベースを使用している場合は、db2updv97 コマンドを実行する必要はありません (データベースのマイグレーションによって自動的にカタログ更新が実行されるため)。バージョン 9.7 にダウングレードすると、このルーチンは機能しなくなります。

構文

スキーマは SYSPROC です。

表関数パラメーター

xmldoc

system_metrics または activity_metrics エレメントを持つ XML 文書が含まれる、タイプ BLOB(100M) の入力引数。これらのエレメントを持つ XML 文書は、以下のソースから入手可能です。

- いずれかの MON_GET_*_DETAILS 表関数からの戻り。
- 統計およびアクティビティー・イベント・モニターからのメトリック列出力。
- 作業単位またはパッケージ・キャッシュ・イベント・モニターからのフォーマット設定された出力。

許可

MON_FORMAT_XML_METRICS_BY_ROW 関数に対する EXECUTE 特権。

例

この例は、MON_GET_WORKLOAD_DETAILS 表関数によって生成された XML 文書から行ベースのフォーマット設定された情報を戻すために、MON_FORMAT_XML_METRICS _BY_ROW 表関数を呼び出す方法を示しています。

```
SELECT SUBSTR(TFXML.WORKLOAD_NAME, 1, 13) AS WORKLOAD_NAME,
       SUBSTR(METRICS.METRIC_NAME, 1, 25) AS METRIC_NAME,
       METRICS.VALUE
   FROM
     TABLE( MON_GET_WORKLOAD_DETAILS( NULL, -2 ) ) AS TFXML,
     TABLE( MON_FORMAT_XML_METRICS_BY_ROW(
                                           TFXML.DETAILS
                                         )) AS METRICS
ORDER BY METRICS.VALUE DESC
```

以下は、この照会によって出力されるリストの一部です。

```
WORKLOAD_NAME METRIC_NAME               VALUE
------------- ------------------------- ---------------
PAYROLL       ACT_COMPLETED_TOTAL                    15
FINANCE       ACT_COMPLETED_TOTAL                    12
PAYROLL       LOCK_WAITS                              8
FINANCE       LOCK_WAITS                              5
FINANCE       DEADLOCKS                               3
PAYROLL       DEADLOCKS                               0
```

戻される情報

表 106. MON_FORMAT_XML_METRICS_BY_ROW で戻される情報

列名	データ・タイプ	説明
METRIC_NAME	VARCHAR(128)	合計時間メトリック値のユニーク ID。

表 106. MON_FORMAT_XML_METRICS_BY_ROW で戻される情報 (続き)

列名	データ・タイプ	説明
VALUE	BIGINT	メトリックの現行値。

タイプ *system_metrics* のエレメントを含む XML 文書は、以下のインターフェースから生成されます。

- MON_GET_CONNECTION_DETAILS
- MON_GET_SERVICE_SUBCLASS_DETAILS
- MON_GET_UNIT_OF_WORK_DETAILS
- MON_GET_WORKLOAD_DETAILS
- STATISTICS イベント・モニターからの DETAILS_XML 列
- UNIT OF WORK イベント・モニターに対する EVMON_FORMAT_UE_TO_TABLES によって生成される METRICS 列
- UNIT OF WORK イベント・モニターに対する EVMON_FORMAT_UE_TO_XML の XMLREPORT 列

この場合に XML 文書から戻されるメトリックについては、表 107 を参照してください。

表 107. *system_metrics* エレメント・タイプを含む *XML* 文書に対する *MON_FORMAT_XML_METRICS_BY_ROW* によって戻されるメトリック名

メトリック名	メトリックについての説明またはモニター・エレメント
TOTAL_WAIT_TIME	total_wait_time - 合計待ち時間
CLIENT_IDLE_WAIT_TIME	client_idle_wait_time - クライアントのアイドル待機時間
POOL_READ_TIME	pool_read_time - バッファー・プール物理読み取り時間の合計
POOL_WRITE_TIME	pool_write_time - バッファー・プール物理書き込み時間の合計
DIRECT_READ_TIME	direct_read_time - 直接読み取り時間
DIRECT_WRITE_TIME	direct_write_time - 直接書き込み時間
LOCK_WAIT_TIME	lock_wait_time - ロック待機中の時間
AGENT_WAIT_TIME	agent_wait_time - エージェント待機時間
WLM_QUEUE_TIME_TOTAL	wlm_queue_time_total - ワークロード・マネージャー合計キュー時間
FCM_SEND_WAIT_TIME	fcm_send_wait_time - FCM 送信待ち時間
FCM_RECV_WAIT_TIME	fcm_recv_wait_time - FCM 受信待機時間
TCPIP_SEND_WAIT_TIME	tcpip_send_wait_time - TCP/IP 送信待ち時間
TCPIP_RECV_WAIT_TIME	tcpip_recv_wait_time - TCP/IP 受信待ち時間
IPC_SEND_WAIT_TIME	ipc_send_wait_time - プロセス間通信送信待ち時間
IPC_RECV_WAIT_TIME	ipc_recv_wait_time - プロセス間通信受信待ち時間
LOG_BUFFER_WAIT_TIME	log_buffer_wait_time - ログ・バッファー待ち時間
LOG_DISK_WAIT_TIME	log_disk_wait_time - ログ・ディスク待機時間
FCM_MESSAGE_SEND_WAIT_TIME	fcm_message_send_wait_time - FCM メッセージ送信待ち時間
FCM_MESSAGE_RECV_WAIT_TIME	fcm_message_recv_wait_time - FCM メッセージの受信待機時間
FCM_TQ_SEND_WAIT_TIME	fcm_tq_send_wait_time - FCM 表キューの送信待ち時間
FCM_TQ_RECV_WAIT_TIME	fcm_tq_recv_wait_time - FCM 表キューの受信待ち時間

表 107. system_metrics エレメント・タイプを含む XML 文書に対する MON_FORMAT_XML_METRICS_BY_ROW によって戻されるメトリック名 (続き)

メトリック名	メトリックについての説明またはモニター・エレメント
AUDIT_FILE_WRITE_WAIT_TIME	audit_file_write_wait_time - 監査ファイル書き込み待機時間
AUDIT_SUBSYSTEM_WAIT_TIME	audit_subsystem_wait_time - 監査サブシステム待機時間
DIAGLOG_WRITE_WAIT_TIME	diaglog_write_wait_time - 診断ログ・ファイルの書き込み待機時間
TOTAL_RQST_TIME	total_rqst_time - 合計要求時間
TOTAL_COMPILE_TIME	total_compile_time - コンパイル時間の合計
TOTAL_IMPLICIT_COMPILE_TIME	total_implicit_compile_time - 暗黙的コンパイル時間の合計
TOTAL_SECTION_TIME	total_section_time - セクション時間の合計
TOTAL_COMMIT_TIME	total_commit_time - コミット時間の合計
TOTAL_ROLLBACK_TIME	total_rollback_time - ロールバック時間の合計
TOTAL_RUNSTATS_TIME	total_runstats_time - ランタイム統計時間の合計
TOTAL_REORG_TIME	total_reorg_time - 再編成時間の合計
TOTAL_LOAD_TIME	total_load_time - ロード時間の合計
TOTAL_SECTION_SORT_TIME	total_section_sort_time - セクション・ソート時間合計
TOTAL_ROUTINE_USER_CODE_TIME	total_routine_user_code_time - ルーチン・ユーザー・コード時間の合計
TOTAL_COMPILE_PROC_TIME	total_compile_proc_time - コンパイル処理時間の合計
TOTAL_IMPLICIT_ COMPILE_PROC_TIME	total_implicit_compile_proc_time - 暗黙的コンパイルの処理時間の合計
TOTAL_SECTION_PROC_TIME	total_section_proc_time - セクション処理時間の合計
TOTAL_COMMIT_PROC_TIME	total_commit_proc_time - コミット処理時間の合計
TOTAL_ROLLBACK_PROC_TIME	total_rollback_proc_time - ロールバック処理時間の合計
TOTAL_RUNSTATS_PROC_TIME	total_runstats_proc_time - ランタイム統計の処理時間の合計
TOTAL_REORG_PROC_TIME	total_reorg_proc_time - 再編成の処理時間の合計
TOTAL_LOAD_PROC_TIME	total_load_proc_time - ロード処理時間の合計
TOTAL_SECTION_SORT_PROC_TIME	total_section_sort_proc_time - セクション・ソート処理時間合計
TOTAL_ROUTINE_USER_CODE_ PROC_TIME	total_routine_user_code_proc_time - ルーチン・ユーザー・コード処理時間の合計
ACT_ABORTED_TOTAL	act_aborted_total - 打ち切られたアクティビティーの合計数
ACT_COMPLETED_TOTAL	act_completed_total - 完了したアクティビティーの合計数
ACT_REJECTED_TOTAL	act_rejected_total - リジェクトされたアクティビティーの合計数
AGENT_WAITS_TOTAL	agent_waits_total - エージェント待機の合計
POOL_DATA_L_READS	pool_data_l_reads - バッファー・プール・データの論理読み取り
POOL_INDEX_L_READS	pool_index_l_reads - バッファー・プール索引の論理読み取り
POOL_TEMP_DATA_L_READS	pool_temp_data_l_reads - バッファー・プール一時データの論理読み取り
POOL_TEMP_INDEX_L_READS	pool_temp_index_l_reads - バッファー・プール一時索引の論理読み取り
POOL_TEMP_XDA_L_READS	pool_temp_xda_l_reads - バッファー・プール一時 XDA データの論理読み取り
POOL_XDA_L_READS	pool_xda_l_reads - バッファー・プール XDA データの論理読み取り
POOL_DATA_P_READS	pool_data_p_reads - バッファー・プール・データの物理読み取り
POOL_INDEX_P_READS	pool_index_p_reads - バッファー・プール索引の物理読み取り
POOL_TEMP_DATA_P_READS	pool_temp_data_p_reads - バッファー・プール一時データの物理読み取り

表 107. system_metrics エレメント・タイプを含む XML 文書に対する MON_FORMAT_XML_METRICS_BY_ROW によって戻されるメトリック名 (続き)

メトリック名	メトリックについての説明またはモニター・エレメント
POOL_TEMP_INDEX_P_READS	pool_temp_index_p_reads - バッファー・プール一時索引の物理読み取り
POOL_TEMP_XDA_P_READS	pool_temp_xda_p_reads - バッファー・プール一時 XDA データの物理読み取り
POOL_XDA_P_READS	pool_xda_p_reads - バッファー・プール XDA データの物理読み取り
POOL_DATA_WRITES	pool_data_writes - バッファー・プールへのデータの書き込み
POOL_INDEX_WRITES	pool_index_writes - バッファー・プール索引の書き込み
POOL_XDA_WRITES	pool_xda_writes - バッファー・プール XDA データの書き込み
DEADLOCKS	deadlocks - デッドロック検出数
DIRECT_READS	direct_reads - データベースからの直接読み取り
DIRECT_WRITES	direct_writes - データベースへの直接書き込み
DIRECT_READ_REQS	direct_read_reqs - 直接読み取り要求
DIRECT_WRITE_REQS	direct_write_reqs - 直接書き込み要求
FCM_RECV_VOLUME	fcm_recv_volume - FCM 受信ボリューム
FCM_RECVS_TOTAL	fcm_recvs_total - FCM 合計受信数
FCM_SEND_VOLUME	fcm_send_volume - FCM 送信ボリューム
FCM_SENDS_TOTAL	fcm_sends_total - FCM 合計送信数
IPC_RECV_VOLUME	ipc_recv_volume - プロセス間通信受信ボリューム
IPC_RECVS_TOTAL	ipc_recvs_total - プロセス間通信合計受信数
IPC_SEND_VOLUME	ipc_send_volume - プロセス間通信送信ボリューム
IPC_SENDS_TOTAL	ipc_sends_total - プロセス間通信合計送信数
LOCK_ESCALS	lock_escals - ロック・エスカレーション数
LOCK_TIMEOUTS	lock_timeouts - ロック・タイムアウト数
LOCK_WAITS	lock_waits - ロック待機数
NUM_LOG_BUFFER_FULL	num_log_buffer_full - フル・ログ・バッファーの回数
LOG_DISK_WAITS_TOTAL	log_disk_waits_total - ログ・ディスク待機の合計
RQSTS_COMPLETED_TOTAL	rqsts_completed_total - 完了した要求の合計数
ROWS_MODIFIED	rows_modified - 変更された行数
ROWS_READ	rows_read - 読み取り行数
ROWS_RETURNED	rows_returned - 戻り行数
TCPIP_RECV_VOLUME	tcpip_recv_volume - TCP/IP 受信ボリューム
TCPIP_SEND_VOLUME	tcpip_send_volume - TCP/IP 送信ボリューム
TCPIP_RECVS_TOTAL	tcpip_recvs_total - TCP/IP 合計受信数
TCPIP_SENDS_TOTAL	tcpip_sends_total - TCP/IP 合計送信数
WLM_QUEUE_ASSIGNMENTS_TOTAL	wlm_queue_assignments_total - ワークロード・マネージャー合計キュー割り当て数
APP_RQSTS_COMPLETED_TOTAL	app_rqsts_completed_total - 完了したアプリケーション要求の合計数
TOTAL_SECTION_SORTS	total_section_sorts - セクション・ソート合計
TOTAL_SORTS	total_sorts - ソート合計
POST_THRESHOLD_SORTS	post_threshold_sorts - ポストしきい値ソート

表 107. *system_metrics* エレメント・タイプを含む *XML* 文書に対する *MON_FORMAT_XML_METRICS_BY_ROW* によって戻されるメトリック名 *(続き)*

メトリック名	メトリックについての説明またはモニター・エレメント
POST_SHRTHRESHOLD_SORTS	post_shrthreshold_sorts - ポスト共有しきい値ソート
SORT_OVERFLOWS	sort_overflows - ソート・オーバーフロー
ACT_RQSTS_TOTAL	act_rqsts_total - アクティビティー要求の合計
TOTAL_ROUTINE_INVOCATIONS	total_routine_invocations - ルーチン呼び出しの合計回数
TOTAL_COMPILATIONS	total_compilations - コンパイルの合計回数
TOTAL_IMPLICIT_COMPILATIONS	total_implicit_compilations - 暗黙的コンパイルの合計回数
TOTAL_APP_SECTION_EXECUTIONS	total_app_section_executions - セクション実行の合計回数
TOTAL_APP_COMMITS	total_app_commits - アプリケーション・コミットの合計回数
INT_COMMITS	int_commits - 内部コミット数
TOTAL_APP_ROLLBACKS	total_app_rollbacks - アプリケーション・ロールバックの合計回数
INT_ROLLBACKS	int_rollbacks - 内部ロールバック数
TOTAL_RUNSTATS	total_runstats - ランタイム統計の合計回数
TOTAL_REORGS	total_reorgs - 再編成の合計回数
TOTAL_LOADS	total_loads - ロードの合計回数
CAT_CACHE_INSERTS	cat_cache_inserts - カタログ・キャッシュ挿入数
CAT_CACHE_LOOKUPS	cat_cache_lookups - カタログ・キャッシュ参照数
PKG_CACHE_INSERTS	pkg_cache_inserts - パッケージ・キャッシュ挿入
PKG_CACHE_LOOKUPS	pkg_cache_lookups - パッケージ・キャッシュ参照
THRESH_VIOLATIONS	thresh_violations - しきい値違反の回数
NUM_LW_THRESH_EXCEEDED	num_lw_thresh_exceeded - しきい値を超えた回数
AUDIT_EVENTS_TOTAL	audit_events_total - 監査イベント合計数
AUDIT_SUBSYSTEM_WAITS_TOTAL	audit_subsystem_waits_total - 監査サブシステム待機の合計
AUDIT_FILE_WRITES_TOTAL	audit_file_writes_total - 書き込まれた監査ファイルの合計数
DIAGLOG_WRITES_TOTAL	diaglog_writes_total - 診断ログ・ファイル書き込みの合計
FCM_MESSAGE_RECV_VOLUME	fcm_message_recv_volume - FCM メッセージ受信ボリューム
FCM_MESSAGE_RECVS_TOTAL	fcm_message_recvs_total - FCM メッセージ合計受信数
FCM_MESSAGE_SEND_VOLUME	fcm_message_send_volume - FCM メッセージ送信ボリューム
FCM_MESSAGE_SENDS_TOTAL	fcm_message_sends_total - FCM メッセージ合計送信数
FCM_TQ_RECV_VOLUME	fcm_tq_recv_volume - FCM 表キューの受信ボリューム
FCM_TQ_RECVS_TOTAL	fcm_tq_recvs_total - FCM 表キューの合計受信数
FCM_TQ_SEND_VOLUME	fcm_tq_send_volume - FCM 表キューの送信ボリューム
FCM_TQ_SENDS_TOTAL	fcm_tq_sends_total - FCM 表キューの合計送信数
TQ_TOT_SEND_SPILLS	tq_tot_send_spills - オーバーフローした表キュー・バッファーの合計数
TOTAL_ROUTINE_TIME	total_routine_time - ルーチン時間の合計
TOTAL_CPU_TIME	total_cpu_time - 合計 CPU 時間
TOTAL_ACT_TIME	total_act_time - 合計アクティビティー時間
TOTAL_ACT_WAIT_TIME	total_act_wait_time - 合計アクティビティー待機時間
TOTAL_APP_RQST_TIME	total_app_rqst_time - アプリケーション要求合計時間

タイプ *activity_metrics* のエレメントを含む XML 文書は、以下のインターフェースから生成されます。

- MON_GET_ACTIVITY_DETAILS
- MON_GET_PKG_CACHE_STMT_DETAILS
- ACTIVITY イベント・モニターからの DETAILS_XML 列
- PACKAGE CACHE イベント・モニターに対する EVMON_FORMAT_UE_TO_TABLES によって生成される METRICS 列
- PACKAGE CACHE イベント・モニターに対する EVMON_FORMAT_UE_TO_XML の XMLREPORT 列

この場合に XML 文書から戻されるメトリックについては、表 108 を参照してください。

表 108. activity_metrics エレメント・タイプを含む XML 文書に対する MON_FORMAT_XML_METRICS_BY_ROW によって戻されるメトリック名

メトリック名	説明またはモニター・エレメント
TOTAL_ACT_WAIT_TIME	total_act_wait_time - 合計アクティビティー待機時間
POOL_READ_TIME	pool_read_time - バッファー・プール物理読み取り時間の合計
POOL_WRITE_TIME	pool_write_time - バッファー・プール物理書き込み時間の合計
DIRECT_READ_TIME	direct_read_time - 直接読み取り時間
DIRECT_WRITE_TIME	direct_write_time - 直接書き込み時間
WLM_QUEUE_TIME_TOTAL	wlm_queue_time_total - ワークロード・マネージャー合計キュー時間
LOCK_WAIT_TIME	lock_wait_time - ロック待機中の時間
LOG_BUFFER_WAIT_TIME	log_buffer_wait_time - ログ・バッファー待ち時間
LOG_DISK_WAIT_TIME	log_disk_wait_time - ログ・ディスク待機時間
AUDIT_FILE_WRITE_WAIT_TIME	audit_file_write_wait_time - 監査ファイル書き込み待機時間
AUDIT_SUBSYSTEM_WAIT_TIME	audit_subsystem_wait_time - 監査サブシステム待機時間
DIAGLOG_WRITE_WAIT_TIME	diaglog_write_wait_time - 診断ログ・ファイルの書き込み待機時間
FCM_SEND_WAIT_TIME	fcm_send_wait_time - FCM 送信待ち時間
FCM_RECV_WAIT_TIME	fcm_recv_wait_time - FCM 受信待機時間
FCM_MESSAGE_SEND_WAIT_TIME	fcm_message_send_wait_time - FCM メッセージ送信待ち時間
FCM_MESSAGE_RECV_WAIT_TIME	fcm_message_recv_wait_time - FCM メッセージの受信待機時間
FCM_TQ_SEND_WAIT_TIME	fcm_tq_send_wait_time - FCM 表キューの送信待ち時間
FCM_TQ_RECV_WAIT_TIME	fcm_tq_recv_wait_time - FCM 表キューの受信待ち時間
STMT_EXEC_TIME	stmt_exec_time - ステートメントの実行時間
TOTAL_ROUTINE_TIME	total_routine_time - ルーチン時間の合計
TOTAL_ROUTINE_NON_SECT_TIME	total_routine_non_sect_time - 非セクション・ルーチン実行時間
TOTAL_ROUTINE_USER_CODE_TIME	total_routine_user_code_time - ルーチン・ユーザー・コード時間の合計
TOTAL_SECTION_TIME	total_section_time - セクション時間の合計
TOTAL_SECTION_SORT_TIME	total_section_sort_time - セクション・ソート時間合計
TOTAL_ROUTINE_NON_SECT_PROC_TIME	total_routine_non_sect_proc_time - 非セクション処理時間
TOTAL_ROUTINE_USER_CODE_PROC_TIME	total_routine_user_code_proc_time - ルーチン・ユーザー・コード処理時間の合計

表 108. *activity_metrics* エレメント・タイプを含む *XML* 文書に対する *MON_FORMAT_XML_METRICS_BY_ROW* によって戻されるメトリック名 *(続き)*

メトリック名	説明またはモニター・エレメント
TOTAL_SECTION_PROC_TIME	total_section_proc_time - セクション処理時間の合計
TOTAL_SECTION_SORT_PROC_TIME	total_section_sort_proc_time - セクション・ソート処理時間合計
TOTAL_SECTION_SORTS	total_section_sorts - セクション・ソート合計
LOCK_ESCALS	lock_escals - ロック・エスカレーション数
LOCK_WAITS	lock_waits - ロック待機数
ROWS_MODIFIED	rows_modified - 変更された行数
ROWS_READ	rows_read - 読み取り行数
ROWS_RETURNED	rows_returned - 戻り行数
DIRECT_READS	direct_reads - データベースからの直接読み取り
DIRECT_READ_REQS	direct_read_reqs - 直接読み取り要求
DIRECT_WRITES	direct_writes - データベースへの直接書き込み
DIRECT_WRITE_REQS	direct_write_reqs - 直接書き込み要求
POOL_DATA_L_READS	pool_data_l_reads - バッファー・プール・データの論理読み取り
POOL_TEMP_DATA_L_READS	pool_temp_data_l_reads - バッファー・プール一時データの論理読み取り
POOL_XDA_L_READS	pool_xda_l_reads - バッファー・プール XDA データの論理読み取り
POOL_TEMP_XDA_L_READS	pool_temp_xda_l_reads - バッファー・プール一時 XDA データの論理読み取り
POOL_INDEX_L_READS	pool_index_l_reads - バッファー・プール索引の論理読み取り
POOL_TEMP_INDEX_L_READS	pool_temp_index_l_reads - バッファー・プール一時索引の論理読み取り
POOL_DATA_P_READS	pool_data_p_reads - バッファー・プール・データの物理読み取り
POOL_TEMP_DATA_P_READS	pool_temp_data_p_reads - バッファー・プール一時データの物理読み取り
POOL_TEMP_XDA_P_READS	pool_temp_xda_p_reads - バッファー・プール一時 XDA データの物理読み取り
POOL_TEMP_INDEX_P_READS	pool_temp_index_p_reads - バッファー・プール一時索引の物理読み取り
POOL_INDEX_P_READS	pool_index_p_reads - バッファー・プール索引の物理読み取り
POOL_DATA_WRITES	pool_data_writes - バッファー・プールへのデータの書き込み
POOL_XDA_WRITES	pool_xda_writes - バッファー・プール XDA データの書き込み
POOL_INDEX_WRITES	pool_index_writes - バッファー・プール索引の書き込み
TOTAL_SORTS	total_sorts - ソート合計
POST_THRESHOLD_SORTS	post_threshold_sorts - ポストしきい値ソート
POST_SHRTHRESHOLD_SORTS	post_shrthreshold_sorts - ポスト共有しきい値ソート
SORT_OVERFLOWS	sort_overflows - ソート・オーバーフロー
WLM_QUEUE_ASSIGNMENTS_TOTAL	wlm_queue_assignments_total - ワークロード・マネージャー合計キュー割り当て数
DEADLOCKS	deadlocks - デッドロック検出数
FCM_RECV_VOLUME	fcm_recv_volume - FCM 受信ボリューム
FCM_RECVS_TOTAL	fcm_recvs_total - FCM 合計受信数
FCM_SEND_VOLUME	fcm_send_volume - FCM 送信ボリューム
FCM_SENDS_TOTAL	fcm_sends_total - FCM 合計送信数
LOCK_TIMEOUTS	lock_timeouts - ロック・タイムアウト数

表 108. activity_metrics エレメント・タイプを含む XML 文書に対する MON_FORMAT_XML_METRICS_BY_ROW によって戻されるメトリック名 (続き)

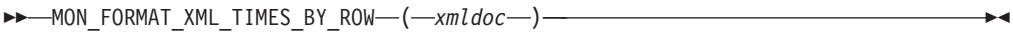
メトリック名	説明またはモニター・エレメント
NUM_LOG_BUFFER_FULL	num_log_buffer_full - フル・ログ・バッファーの回数
LOG_DISK_WAITS_TOTAL	log_disk_waits_total - ログ・ディスク待機の合計
TOTAL_ROUTINE_INVOCATIONS	total_routine_invocations - ルーチン呼び出しの合計回数
AUDIT_EVENTS_TOTAL	audit_events_total - 監査イベント合計数
AUDIT_SUBSYSTEM_WAITS_TOTAL	audit_subsystem_waits_total - 監査サブシステム待機の合計
AUDIT_FILE_WRITES_TOTAL	audit_file_writes_total - 書き込まれた監査ファイルの合計数
DIAGLOG_WRITES_TOTAL	diaglog_writes_total - 診断ログ・ファイル書き込みの合計
FCM_MESSAGE_RECV_VOLUME	fcm_message_recv_volume - FCM メッセージ受信ボリューム
FCM_MESSAGE_RECVS_TOTAL	fcm_message_recvs_total - FCM メッセージ合計受信数
FCM_MESSAGE_SEND_VOLUME	fcm_message_send_volume - FCM メッセージ送信ボリューム
FCM_MESSAGE_SENDS_TOTAL	fcm_message_sends_total - FCM メッセージ合計送信数
FCM_TQ_RECV_VOLUME	fcm_tq_recv_volume - FCM 表キューの受信ボリューム
FCM_TQ_RECVS_TOTAL	fcm_tq_recvs_total - FCM 表キューの合計受信数
FCM_TQ_SEND_VOLUME	fcm_tq_send_volume - FCM 表キューの送信ボリューム
FCM_TQ_SENDS_TOTAL	fcm_tq_sends_total - FCM 表キューの合計送信数
TQ_TOT_SEND_SPILLS	tq_tot_send_spills - オーバーフローした表キュー・バッファーの合計数
THRESH_VIOLATIONS	thresh_violations - しきい値違反の回数
NUM_LW_THRESH_EXCEEDED	num_lw_thresh_exceeded - しきい値を超えた回数
COORD_STMT-EXEC_TIME	coord_stmt_exec_time - コーディネーター・エージェントによるステートメントの実行時間
TOTAL_ACT_TIME	total_act_time - 合計アクティビティー時間
TOTAL_CPU_TIME	total_cpu_time - 合計 CPU 時間

MON_FORMAT_XML_TIMES_BY_ROW - フォーマット設定された行ベースの待機/処理時間の結合された階層を取得する

MON_FORMAT_XML_TIMES_BY_ROW 表関数は、XML メトリック文書に含まれる待機時間と処理時間の結合された階層に関するフォーマット設定された行ベースの出力を戻します。

注: バージョン 9.7 のフィックスパック 1 以前で作成されたデータベースを使用している場合、このルーチンを実行するには、既に db2updv97 コマンドを実行済みでなければなりません。バージョン 9.7 以前に作成されたデータベースを使用している場合は、db2updv97 コマンドを実行する必要はありません (データベースのマイグレーションによって自動的にカタログ更新が実行されるため)。バージョン 9.7 にダウングレードすると、このルーチンは機能しなくなります。

構文



スキーマは SYSPROC です。

表関数パラメーター

xmldoc

system_metrics または activity_metrics エレメントを持つ XML 文書が含まれる、タイプ BLOB(100M) の入力引数。これらのエレメントを持つ XML 文書は、以下のソースから入手可能です。

- いずれかの MON_GET_*_DETAILS 表関数からの戻り。
- 統計およびアクティビティー・イベント・モニターからのメトリック列出力。
- 作業単位またはパッケージ・キャッシュ・イベント・モニターからのフォーマット設定された出力。

許可

MON_FORMAT_XML_TIMES_BY_ROW 関数に対する EXECUTE 特権。

例

DB2 データベース・マネージャーの中で、アプリケーションによって時間が費やされている場所を判別するには、以下の照会を実行することにより、待機/処理時間メトリックの組み合わせをメトリック階層で表示することができます。

```
SELECT SUBSTR(T.SERVICE_SUPERCLASS_NAME,1,15) as SUPERCLASS,
       SUBSTR(T.SERVICE_SUBCLASS_NAME,1,15) as SUBCLASS,
       T.MEMBER,
       SUBSTR(U.METRIC_NAME, 1,15) AS METRIC_NAME,
       SUBSTR(U.PARENT_METRIC_NAME,1,15) AS PARENT_NAME,
       U.TOTAL_TIME_VALUE,
       U.COUNT
FROM
TABLE(MON_GET_SERVICE_SUBCLASS_DETAILS(NULL, NULL, -2)) AS T,
TABLE(MON_FORMAT_XML_TIMES_BY_ROW(T.DETAILS)) AS U
```

以下はこの照会の出力例です。

```
SUPERCLASS SUBCLASS MEMBER METRIC_NAME     PARENT_NAME     T..._VALUE COUNT
---------- -------- ------ --------------- --------------- ---------- -----
MYSC       MYSSC         0 FCM_MESSAGE_REC FCM_RECV_WAIT_T          0     0
MYSC       MYSSC         0 FCM_TQ_RECV_WAI FCM_RECV_WAIT_T          0     0
MYSC       MYSSC         0 FCM_MESSAGE_SEN FCM_SEND_WAIT_T          0     0
MYSC       MYSSC         0 FCM_TQ_SEND_WAI FCM_SEND_WAIT_T          0     0
MYSC       MYSSC         0 TOTAL_COMMIT_PR TOTAL_RQST_TIME        300     1
MYSC       MYSSC         0 TOTAL_COMPILE_P TOTAL_RQST_TIME        700     1
MYSC       MYSSC         0 TOTAL_IMPLICIT_ TOTAL_RQST_TIME          0     0
MYSC       MYSSC         0 TOTAL_LOAD_PROC TOTAL_RQST_TIME          0     0
MYSC       MYSSC         0 TOTAL_REORG_PRO TOTAL_RQST_TIME          0     0
MYSC       MYSSC         0 TOTAL_ROLLBACK_ TOTAL_RQST_TIME          0     0
MYSC       MYSSC         0 TOTAL_RUNSTATS_ TOTAL_RQST_TIME          0     0
MYSC       MYSSC         0 TOTAL_SECTION_P TOTAL_RQST_TIME       7322     1
MYSC       MYSSC         0 TOTAL_WAIT_TIME TOTAL_RQST_TIME          0     0
MYSC       MYSSC         0 TOTAL_SECTION_S TOTAL_SECTION_P          0     0
MYSC       MYSSC         0 AGENT_WAIT_TIME TOTAL_WAIT_TIME          0     0
MYSC       MYSSC         0 AUDIT_FILE_WRIT TOTAL_WAIT_TIME          0     0
MYSC       MYSSC         0 AUDIT_SUBSYSTEM TOTAL_WAIT_TIME          0     0
MYSC       MYSSC         0 DIAGLOG_WRITE_W TOTAL_WAIT_TIME          0     0
MYSC       MYSSC         0 DIRECT_READ_TIM TOTAL_WAIT_TIME       1204    17
MYSC       MYSSC         0 DIRECT_WRITE_TI TOTAL_WAIT_TIME          0     0
MYSC       MYSSC         0 FCM_RECV_WAIT_T TOTAL_WAIT_TIME          0     0
MYSC       MYSSC         0 FCM_SEND_WAIT_T TOTAL_WAIT_TIME          0     0
```

```
MYSC       MYSSC         0 IPC_RECV_WAIT_T TOTAL_WAIT_TIME          0     0
MYSC       MYSSC         0 IPC_SEND_WAIT_T TOTAL_WAIT_TIME          0     0
MYSC       MYSSC         0 LOCK_WAIT_TIME  TOTAL_WAIT_TIME          0     0
MYSC       MYSSC         0 LOG_BUFFER_WAIT TOTAL_WAIT_TIME          0     0
MYSC       MYSSC         0 LOG_DISK_WAIT_T TOTAL_WAIT_TIME        523     2
MYSC       MYSSC         0 POOL_READ_TIME  TOTAL_WAIT_TIME       2432     7
MYSC       MYSSC         0 POOL_WRITE_TIME TOTAL_WAIT_TIME          0     0
MYSC       MYSSC         0 TCPIP_RECV_WAIT TOTAL_WAIT_TIME        523     1
MYSC       MYSSC         0 TCPIP_SEND_WAIT TOTAL_WAIT_TIME        241     1
MYSC       MYSSC         0 WLM_QUEUE_TIME_ TOTAL_WAIT_TIME          0     0
MYSC       MYSSC         0 CLIENT_IDLE_WAI -                      234     -
MYSC       MYSSC         0 TOTAL_RQST_TIME -                    13245     1
  34 record(s) selected.
```

戻される情報

表 109. MON_FORMAT_XML_TIMES_BY_ROW で戻される情報

列名	データ・タイプ	説明
METRIC_NAME	VARCHAR(128)	合計時間メトリック値のユニーク ID。
TOTAL_TIME_VALUE	BIGINT	metric_name に対応する合計時間値 (ミリ秒)。
COUNT	BIGINT	このタイプのインターバルのオカレンス数。
PARENT_METRIC_NAME	VARCHAR(128)	親である合計時間メトリックの ID。そのメトリックの値には total_time_value がサブセットとして含まれます。

タイプ *system_metrics* のエレメントを含む XML 文書は、以下のインターフェースから生成されます。

- MON_GET_CONNECTION_DETAILS
- MON_GET_SERVICE_SUBCLASS_DETAILS
- MON_GET_UNIT_OF_WORK_DETAILS
- MON_GET_WORKLOAD_DETAILS
- STATISTICS イベント・モニターからの DETAILS_XML 列
- UNIT OF WORK イベント・モニターに対する EVMON_FORMAT_UE_TO_TABLES によって生成される METRICS 列
- UNIT OF WORK イベント・モニターに対する EVMON_FORMAT_UE_TO_XML の XMLREPORT 列

この場合に XML 文書から戻されるメトリックの種類とその親メトリックについては、表 110 を参照してください。

表 110. system_metrics エレメント・タイプを含む XML 文書に対する MON_FORMAT_XML_TIMES_BY_ROW によって戻されるメトリック名

メトリック名	親メトリック名	メトリックについての説明またはモニター・エレメント
TOTAL_RQST_TIME	NULL	total_rqst_time - 合計要求時間
TOTAL_COMPILE_ PROC_TIME	TOTAL_RQST_TIME	total_compile_proc_time - コンパイル処理時間の合計
TOTAL_IMPLICIT_ COMPILE_PROC_TIME	TOTAL_RQST_TIME	total_implicit_compile_proc_time - 暗黙的コンパイルの処理時間の合計
TOTAL_SECTION_ PROC_TIME	TOTAL_RQST_TIME	total_section_proc_time - セクション処理時間の合計

表 110. system_metrics エレメント・タイプを含む XML 文書に対する MON_FORMAT_XML_TIMES_BY_ROW によって戻されるメトリック名 (続き)

メトリック名	親メトリック名	メトリックについての説明またはモニター・エレメント
TOTAL_COMMIT_ PROC_TIME	TOTAL_RQST_TIME	total_commit_proc_time - コミット処理時間の合計
TOTAL_ROLLBACK _PROC_TIME	TOTAL_RQST_TIME	total_rollback_proc_time - ロールバック処理時間の合計
TOTAL_ROUTINE_USER _CODE_PROC_TIME	TOTAL_RQST_TIME	total_routine_user_code_proc_time - ルーチン・ユーザー・コード処理時間の合計
TOTAL_RUNSTATS_ PROC_TIME	TOTAL_RQST_TIME	total_runstats_proc_time - ランタイム統計の処理時間の合計
TOTAL_REORG _PROC_TIME	TOTAL_RQST_TIME	total_reorg_proc_time - 再編成の処理時間の合計
TOTAL_LOAD_PROC_TIME	TOTAL_RQST_TIME	total_load_proc_time - ロード処理時間の合計
TOTAL_SECTION_ SORT_PROC_TIME	TOTAL_SECTION_ PROC_TIME	total_section_sort_proc_time - セクション・ソート処理時間合計
TOTAL_WAIT_TIME	TOTAL_RQST_TIME	total_wait_time - 合計待ち時間
CLIENT_IDLE_WAIT_TIME	NULL	client_idle_wait_time - クライアントのアイドル待機時間
POOL_READ_TIME	TOTAL_WAIT_TIME	pool_read_time - バッファー・プール物理読み取り時間の合計
POOL_WRITE_TIME	TOTAL_WAIT_TIME	pool_write_time - バッファー・プール物理書き込み時間の合計
DIRECT_READ_TIME	TOTAL_WAIT_TIME	direct_read_time - 直接読み取り時間
DIRECT_WRITE_TIME	TOTAL_WAIT_TIME	direct_write_time - 直接書き込み時間
LOCK_WAIT_TIME	TOTAL_WAIT_TIME	lock_wait_time - ロック待機中の時間
AGENT_WAIT_TIME	TOTAL_WAIT_TIME	agent_wait_time - エージェント待機時間
WLM_QUEUE_TIME_TOTAL	TOTAL_WAIT_TIME	wlm_queue_time_total - ワークロード・マネージャー合計キュー時間
FCM_SEND_WAIT_TIME	TOTAL_WAIT_TIME	fcm_send_wait_time - FCM 送信待ち時間
FCM_RECV_WAIT_TIME	TOTAL_WAIT_TIME	fcm_recv_wait_time - FCM 受信待機時間
TCPIP_SEND_WAIT_TIME	TOTAL_WAIT_TIME	tcpip_send_wait_time - TCP/IP 送信待ち時間
TCPIP_RECV_WAIT_TIME	TOTAL_WAIT_TIME	tcpip_recv_wait_time - TCP/IP 受信待ち時間
IPC_SEND_WAIT_TIME	TOTAL_WAIT_TIME	ipc_send_wait_time - プロセス間通信送信待ち時間
IPC_RECV_WAIT_TIME	TOTAL_WAIT_TIME	ipc_recv_wait_time - プロセス間通信受信待ち時間
LOG_BUFFER_WAIT_TIME	TOTAL_WAIT_TIME	log_buffer_wait_time - ログ・バッファー待ち時間
LOG_DISK_WAIT_TIME	TOTAL_WAIT_TIME	log_disk_wait_time - ログ・ディスク待機時間
FCM_MESSAGE_ SEND_WAIT_TIME	FCM_SEND_WAIT_TIME	fcm_message_send_wait_time - FCM メッセージ送信待ち時間
FCM_MESSAGE_ RECV_WAIT_TIME	FCM_RECV_WAIT_TIME	fcm_message_recv_wait_time - FCM メッセージの受信待機時間
FCM_TQ_SEND_WAIT_TIME	FCM_SEND_WAIT_TIME	fcm_tq_send_wait_time - FCM 表キューの送信待ち時間
FCM_TQ_RECV_WAIT_TIME	FCM_RECV_WAIT_TIME	fcm_tq_recv_wait_time - FCM 表キューの受信待ち時間
AUDIT_FILE_WRITE _WAIT_TIME	TOTAL_WAIT_TIME	audit_file_write_wait_time - 監査ファイル書き込み待機時間

表 110. system_metrics エレメント・タイプを含む XML 文書に対する MON_FORMAT_XML_TIMES_BY_ROW によって戻されるメトリック名 (続き)

メトリック名	親メトリック名	メトリックについての説明またはモニター・エレメント
AUDIT_SUBSYSTEM _WAIT_TIME	TOTAL_WAIT_TIME	audit_subsystem_wait_time - 監査サブシステム待機時間
DIAGLOG_WRITE_ WAIT_TIME	TOTAL_WAIT_TIME	diaglog_write_wait_time - 診断ログ・ファイルの書き込み待機時間

タイプ *activity_metrics* のエレメントを含む XML 文書は、以下のインターフェースから生成されます。

- MON_GET_ACTIVITY_DETAILS
- MON_GET_PKG_CACHE_STMT_DETAILS
- ACTIVITY イベント・モニターからの DETAILS_XML 列
- PACKAGE CACHE イベント・モニターに対する EVMON_FORMAT_UE_TO_TABLES によって生成される METRICS 列
- PACKAGE CACHE イベント・モニターに対する EVMON_FORMAT_UE_TO_XML の XMLREPORT 列

この場合に XML 文書から戻されるメトリックの種類とその親メトリックについては、表 111 を参照してください。

表 111. activity_metrics エレメント・タイプを含む XML 文書に対する MON_FORMAT_XML_TIMES_BY_ROW によって戻されるメトリック名

メトリック名	親メトリック名	説明またはモニター・エレメント
STMT_EXEC_TIME	NULL	stmt_exec_time - ステートメントの実行時間
TOTAL_ROUTINE_NON _SECT_PROC_TIME	STMT_EXEC_TIME	total_routine_non_sect_proc_time - 非セクション処理時間
TOTAL_ROUTINE_USER _CODE_PROC_TIME	TOTAL_ROUTINE_NON _SECT_PROC_TIME	total_routine_user_code_proc_time - ルーチン・ユーザー・コード処理時間の合計
TOTAL_SECTION_ PROC_TIME	STMT_EXEC_TIME	total_section_proc_time - セクション処理時間の合計
TOTAL_SECTION_SORT PROC_TIME	TOTAL_SECTION_ PROC_TIME	total_section_sort_proc_time - セクション・ソート処理時間合計
TOTAL_ACT_WAIT_TIME	STMT_EXEC_TIME	total_act_wait_time - 合計アクティビティー待機時間
WLM_QUEUE_TIME_TOTAL	NULL	wlm_queue_time_total - ワークロード・マネージャー合計キュー時間
POOL_READ_TIME	TOTAL_ACT_ WAIT_TIME	pool_read_time - バッファー・プール物理読み取り時間の合計
POOL_WRITE_TIME	TOTAL_ACT_ WAIT_TIME	pool_write_time - バッファー・プール物理書き込み時間の合計
DIRECT_READ_TIME	TOTAL_ACT_ WAIT_TIME	direct_read_time - 直接読み取り時間
DIRECT_WRITE_TIME	TOTAL_ACT_ WAIT_TIME	direct_write_time - 直接書き込み時間
LOCK_WAIT_TIME	TOTAL_ACT_ WAIT_TIME	lock_wait_time - ロック待機中の時間

表 111. activity_metrics エレメント・タイプを含む XML 文書に対する MON_FORMAT_XML_TIMES_BY_ROW によって戻されるメトリック名 (続き)

メトリック名	親メトリック名	説明またはモニター・エレメント
LOG_BUFFER_WAIT_TIME	TOTAL_ACT_ WAIT_TIME	log_buffer_wait_time - ログ・バッファー待ち時間
LOG_DISK_WAIT_TIME	TOTAL_ACT_ WAIT_TIME	log_disk_wait_time - ログ・ディスク待機時間
AUDIT_FILE_WRITE _WAIT_TIME	TOTAL_ACT_ WAIT_TIME	audit_file_write_wait_time - 監査ファイル書き込み待機時間
AUDIT_SUBSYSTEM _WAIT_TIME	TOTAL_ACT_ WAIT_TIME	audit_subsystem_wait_time - 監査サブシステム待機時間
DIAGLOG_WRITE_ WAIT_TIME	TOTAL_ACT_ WAIT_TIME	diaglog_write_wait_time - 診断ログ・ファイルの書き込み待機時間
FCM_SEND_WAIT_TIME	TOTAL_ACT_ WAIT_TIME	fcm_send_wait_time - FCM 送信待ち時間
FCM_RECV_WAIT_TIME	TOTAL_ACT_ WAIT_TIME	fcm_recv_wait_time - FCM 受信待機時間
FCM_MESSAGE_ SEND_WAIT_TIME	FCM_SEND_ WAIT_TIME	fcm_message_send_wait_time - FCM メッセージ送信待ち時間
FCM_MESSAGE_ RECV_WAIT_TIME	FCM_RECV_ WAIT_TIME	fcm_message_recv_wait_time - FCM メッセージの受信待機時間
FCM_TQ_SEND_WAIT_TIME	FCM_SEND_ WAIT_TIME	fcm_tq_send_wait_time - FCM 表キューの送信待ち時間
FCM_TQ_RECV_WAIT_TIME	FCM_RECV_ WAIT_TIME	fcm_tq_recv_wait_time - FCM 表キューの受信待ち時間

MON_FORMAT_XML_WAIT_TIMES_BY_ROW - 待機時間に関するフォーマット設定された行ベースの出力の取得

MON_FORMAT_XML_WAIT_TIMES_BY_ROW 表関数は、XML メトリック文書に含まれる待機時間に関するフォーマット設定された行ベースの出力を戻します。

注: バージョン 9.7 のフィックスパック 1 以前で作成されたデータベースを使用している場合、このルーチンを実行するには、既に db2updv97 コマンドを実行済みでなければなりません。バージョン 9.7 以前に作成されたデータベースを使用している場合は、db2updv97 コマンドを実行する必要はありません (データベースのマイグレーションによって自動的にカタログ更新が実行されるため)。バージョン 9.7 にダウングレードすると、このルーチンは機能しなくなります。

構文

スキーマは SYSPROC です。

表関数パラメーター

xmldoc

system_metrics または activity_metrics エレメントを持つ XML 文書が含まれる、タイプ BLOB(100M) の入力引数。これらのエレメントを持つ XML 文書は、以下のソースから入手可能です。

- いずれかの MON_GET_*_DETAILS 表関数からの戻り。
- 統計およびアクティビティー・イベント・モニターからのメトリック列出力。
- 作業単位またはパッケージ・キャッシュ・イベント・モニターからのフォーマット設定された出力。

許可

MON_FORMAT_XML_WAIT_TIMES_BY_ROW 関数に対する EXECUTE 特権。

例

この例は、MON_GET_WORKLOAD_DETAILS 表関数によって生成された XML 文書から行ベースのフォーマット設定された出力を戻すために、MON_FORMAT_XML_WAIT_TIMES_BY_ROW 表関数を呼び出す方法を示しています。各ワークロードのメトリックとその値が出力に示されます。

```
SELECT SUBSTR(TFXML.WORKLOAD_NAME, 1, 13) AS WORKLOAD_NAME,
       SUBSTR(WAITS.METRIC_NAME, 1, 25) AS METRIC_NAME,
       WAITS.TOTAL_TIME_VALUE,
       WAITS.COUNT
   FROM
     TABLE( MON_GET_WORKLOAD_DETAILS( NULL, -2 ) ) AS TFXML,
     TABLE( MON_FORMAT_XML_WAIT_TIMES_BY_ROW(
                                          TFXML.DETAILS
                                        )) AS WAITS
   ORDER BY WAITS.TOTAL_TIME_VALUE DESC
```

以下は、この照会によって出力されるリストの一部です。

```
WORKLOAD_NAME METRIC_NAME               TOTAL_TIME_VALUE    COUNT
------------- ------------------------- ------------------- ------
PAYROLL       CLIENT_IDLE_WAIT_TIME                2193672    174
FINANCE       CLIENT_IDLE_WAIT_TIME                 738290     16
PAYROLL       DIRECT_READ_TIME                       67892     81
FINANCE       DIRECT_READ_TIME                       32343      8
FINANCE       LOCK_WAIT_TIME                          8463      3
PAYROLL       LOCK_WAIT_TIME                            55      1
```

戻される情報

表 112. MON_FORMAT_XML_WAIT_TIMES_BY_ROW で戻される情報

列名	データ・タイプ	説明
METRIC_NAME	VARCHAR(128)	合計時間メトリック値のユニーク ID。
TOTAL_TIME_VALUE	BIGINT	metric_name に対応する合計時間値 (ミリ秒)。
COUNT	BIGINT	このタイプのインターバルのオカレンス数。
PARENT_METRIC_NAME	VARCHAR(128)	親である合計時間メトリックの ID。そのメトリックの値には total_time_value がサブセットとして含まれます。

タイプ *system_metrics* のエレメントを含む XML 文書は、以下のインターフェースから生成されます。

- MON_GET_CONNECTION_DETAILS
- MON_GET_SERVICE_SUBCLASS_DETAILS
- MON_GET_UNIT_OF_WORK_DETAILS
- MON_GET_WORKLOAD_DETAILS
- STATISTICS イベント・モニターからの DETAILS_XML 列
- UNIT OF WORK イベント・モニターに対する EVMON_FORMAT_UE_TO_TABLES によって生成される METRICS 列
- UNIT OF WORK イベント・モニターに対する EVMON_FORMAT_UE_TO_XML の XMLREPORT 列

この場合に XML 文書から戻されるメトリックの種類とその親メトリックについては、表 113 を参照してください。

表 113. system_metrics エレメント・タイプを含む XML 文書に対する MON_FORMAT_XML_WAIT_TIMES_BY_ROW によって戻されるメトリック名

メトリック名	親メトリック名	メトリックについての説明またはモニター・エレメント
TOTAL_WAIT_TIME	TOTAL_RQST_TIME	total_wait_time - 合計待ち時間
CLIENT_IDLE_WAIT_TIME	NULL	client_idle_wait_time - クライアントのアイドル待機時間
POOL_READ_TIME	TOTAL_WAIT_TIME	pool_read_time - バッファー・プール物理読み取り時間の合計
POOL_WRITE_TIME	TOTAL_WAIT_TIME	pool_write_time - バッファー・プール物理書き込み時間の合計
DIRECT_READ_TIME	TOTAL_WAIT_TIME	direct_read_time - 直接読み取り時間
DIRECT_WRITE_TIME	TOTAL_WAIT_TIME	direct_write_time - 直接書き込み時間
LOCK_WAIT_TIME	TOTAL_WAIT_TIME	lock_wait_time - ロック待機中の時間
AGENT_WAIT_TIME	TOTAL_WAIT_TIME	agent_wait_time - エージェント待機時間
WLM_QUEUE_TIME_TOTAL	TOTAL_WAIT_TIME	wlm_queue_time_total - ワークロード・マネージャー合計キュー時間
FCM_SEND_WAIT_TIME	TOTAL_WAIT_TIME	fcm_send_wait_time - FCM 送信待ち時間
FCM_RECV_WAIT_TIME	TOTAL_WAIT_TIME	fcm_recv_wait_time - FCM 受信待機時間
TCPIP_SEND_WAIT_TIME	TOTAL_WAIT_TIME	tcpip_send_wait_time - TCP/IP 送信待ち時間
TCPIP_RECV_WAIT_TIME	TOTAL_WAIT_TIME	tcpip_recv_wait_time - TCP/IP 受信待ち時間
IPC_SEND_WAIT_TIME	TOTAL_WAIT_TIME	ipc_send_wait_time - プロセス間通信送信待ち時間
IPC_RECV_WAIT_TIME	TOTAL_WAIT_TIME	ipc_recv_wait_time - プロセス間通信受信待ち時間
LOG_BUFFER_WAIT_TIME	TOTAL_WAIT_TIME	log_buffer_wait_time - ログ・バッファー待ち時間
LOG_DISK_WAIT_TIME	TOTAL_WAIT_TIME	log_disk_wait_time - ログ・ディスク待機時間
FCM_MESSAGE_SEND_WAIT_TIME	FCM_SEND_WAIT_TIME	fcm_message_send_wait_time - FCM メッセージ送信待ち時間
FCM_MESSAGE_RECV_WAIT_TIME	FCM_RECV_WAIT_TIME	fcm_message_recv_wait_time - FCM メッセージの受信待機時間
FCM_TQ_SEND_WAIT_TIME	FCM_SEND_WAIT_TIME	fcm_tq_send_wait_time - FCM 表キューの送信待ち時間

表 113. system_metrics エレメント・タイプを含む XML 文書に対する MON_FORMAT_XML_WAIT_TIMES_BY_ROW によって戻されるメトリック名 (続き)

メトリック名	親メトリック名	メトリックについての説明またはモニター・エレメント
FCM_TQ_RECV_WAIT_TIME	FCM_RECV_ WAIT_TIME	fcm_tq_recv_wait_time - FCM 表キューの受信待ち時間
AUDIT_FILE_WRITE _WAIT_TIME	TOTAL_WAIT_TIME	audit_file_write_wait_time - 監査ファイル書き込み待機時間
AUDIT_SUBSYSTEM _WAIT_TIME	TOTAL_WAIT_TIME	audit_subsystem_wait_time - 監査サブシステム待機時間
DIAGLOG_WRITE_ WAIT_TIME	TOTAL_WAIT_TIME	diaglog_write_wait_time - 診断ログ・ファイルの書き込み待機時間

タイプ *activity_metrics* のエレメントを含む XML 文書は、以下のインターフェースから生成されます。

- MON_GET_ACTIVITY_DETAILS
- MON_GET_PKG_CACHE_STMT_DETAILS
- ACTIVITY イベント・モニターからの DETAILS_XML 列
- PACKAGE CACHE イベント・モニターに対する EVMON_FORMAT_UE_TO_TABLES によって生成される METRICS 列
- PACKAGE CACHE イベント・モニターに対する EVMON_FORMAT_UE_TO_XML の XMLREPORT 列

この場合に XML 文書から戻されるメトリックの種類とその親メトリックについては、表 114 を参照してください。

表 114. activity_metrics エレメント・タイプを含む XML 文書に対する MON_FORMAT_XML_WAIT_TIMES_BY_ROW によって戻されるメトリック名

メトリック名	親メトリック名	説明またはモニター・エレメント
TOTAL_ACT_WAIT_TIME	STMT_EXEC_TIME	total_act_wait_time - 合計アクティビティー待機時間
WLM_QUEUE_TIME_TOTAL	NULL	wlm_queue_time_total - ワークロード・マネージャー合計キュー時間
POOL_READ_TIME	TOTAL_ACT_ WAIT_TIME	pool_read_time - バッファー・プール物理読み取り時間の合計
POOL_WRITE_TIME	TOTAL_ACT_ WAIT_TIME	pool_write_time - バッファー・プール物理書き込み時間の合計
DIRECT_READ_TIME	TOTAL_ACT_ WAIT_TIME	direct_read_time - 直接読み取り時間
DIRECT_WRITE_TIME	TOTAL_ACT_ WAIT_TIME	direct_write_time - 直接書き込み時間
LOCK_WAIT_TIME	TOTAL_ACT_ WAIT_TIME	lock_wait_time - ロック待機中の時間
LOG_BUFFER_WAIT_TIME	TOTAL_ACT_ WAIT_TIME	log_buffer_wait_time - ログ・バッファー待ち時間
LOG_DISK_WAIT_TIME	TOTAL_ACT_ WAIT_TIME	log_disk_wait_time - ログ・ディスク待機時間

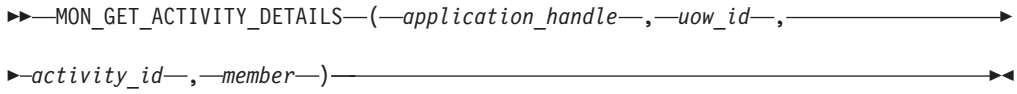
表 114. *activity_metrics* エレメント・タイプを含む *XML* 文書に対する *MON_FORMAT_XML_WAIT_TIMES_BY_ROW* によって戻されるメトリック名 *(続き)*

メトリック名	親メトリック名	説明またはモニター・エレメント
AUDIT_FILE_WRITE _WAIT_TIME	TOTAL_ACT_ WAIT_TIME	audit_file_write_wait_time - 監査ファイル書き込み待機時間
AUDIT_SUBSYSTEM _WAIT_TIME	TOTAL_ACT_ WAIT_TIME	audit_subsystem_wait_time - 監査サブシステム待機時間
DIAGLOG_WRITE_WAIT_TIME	TOTAL_ACT_ WAIT_TIME	diaglog_write_wait_time - 診断ログ・ファイルの書き込み待機時間
FCM_SEND_WAIT_TIME	TOTAL_ACT_ WAIT_TIME	fcm_send_wait_time - FCM 送信待ち時間
FCM_RECV_WAIT_TIME	TOTAL_ACT_ WAIT_TIME	fcm_recv_wait_time - FCM 受信待機時間
FCM_MESSAGE_ SEND_WAIT_TIME	FCM_SEND_ WAIT_TIME	fcm_message_send_wait_time - FCM メッセージ送信待ち時間
FCM_MESSAGE_ RECV_WAIT_TIME	FCM_RECV_ WAIT_TIME	fcm_message_recv_wait_time - FCM メッセージの受信待機時間
FCM_TQ_SEND_ WAIT_TIME	FCM_SEND_ WAIT_TIME	fcm_tq_send_wait_time - FCM 表キューの送信待ち時間
FCM_TQ_RECV_ WAIT_TIME	FCM_RECV_ WAIT_TIME	fcm_tq_recv_wait_time - FCM 表キューの受信待ち時間

MON_GET_ACTIVITY_DETAILS 表関数 - 完全なアクティビティー詳細の取得

MON_GET_ACTIVITY_DETAILS 表関数は、一般的なアクティビティー情報 (ステートメント・テキストなど) やアクティビティーの一連のメトリックを含む、アクティビティーについての詳細を戻します。

構文



スキーマは SYSPROC です。

表関数パラメーター

application_handle
有効なアプリケーション・ハンドルを指定する、タイプ BIGINT の入力引数。引数が NULL の場合、行はこの関数から戻されず、SQL0171N エラーが戻されます。

uow_id
アプリケーション内で固有の有効な作業単位 ID を指定する、タイプ INTEGER の入力引数。引数が NULL の場合、行はこの関数から戻されず、SQL0171N エラーが戻されます。

activity_id
作業単位内で固有の有効なアクティビティー ID を指定する、タイプ INTEGER の入力引数。引数が NULL の場合、行はこの関数から戻されず、SQL0171N エラーが戻されます。

member
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なメンバー番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・メンバーには -1、すべてのデータベース・メンバーには -2 を指定します。NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

MON_GET_ACTIVITY_DETAILS 関数に対する EXECUTE 特権。

例

長時間実行される照会を確認し、時間を要しているのが実行または待機のどちらであるかを判別します (例えば、ロックまたは入出力のどちらでブロックされているかなど)。

注: 以下の照会を結合して 1 つのステートメントにすることができますが、分かりやすくするために 2 ステップで示されています。さらに、テキスト全体を検索したい場合には、実行可能 ID を使って MON_GET_PKG_CACHE_STMT 表関数からステートメント・テキストを取得できます。

1. まず、WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES_V9.7 表関数を使用して、アクティビティーとそれらの開始時刻をリストします。

```
SELECT application_handle,
       activity_id,
       uow_id,
       local_start_time
FROM TABLE(
   WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES_V97(
      cast(NULL as bigint), -1)
) AS T
```

以下はこの照会の出力例です。

```
APPLICATION_HANDLE   ACTIVITY_ID UOW_ID      LOCAL_START_TIME
-------------------- ----------- ----------- --------------------------
                   7           1           2 2008-06-10-10.06.55.675668
                  16           1           7 2008-06-10-10.08.38.613610

  2 record(s) selected.
```

2. 次に、MON_GET_ACTIVITY_DETAILS 表関数を使用してアクティビティーが待機に要した時間のパーセントを表示します。

```
SELECT actmetrics.application_handle,
  actmetrics.activity_id,
  actmetrics.uow_id,
  varchar(actmetrics.stmt_text, 50) as stmt_text,
  actmetrics.total_act_time,
  actmetrics.total_act_wait_time,
  CASE WHEN actmetrics.total_act_time > 0
    THEN DEC((
      FLOAT(actmetrics.total_act_wait_time) /
      FLOAT(actmetrics.total_act_time)) * 100, 5, 2)
    ELSE NULL
```

```
  END AS PERCENTAGE_WAIT_TIME
FROM TABLE(MON_GET_ACTIVITY_DETAILS(7, 2, 1, -2)) AS ACTDETAILS,
XMLTABLE (XMLNAMESPACES( DEFAULT 'http://www.ibm.com/xmlns/prod/db2/mon'),
  '$actmetrics/db2_activity_details'
  PASSING XMLPARSE(DOCUMENT ACTDETAILS.DETAILS) as "actmetrics"
  COLUMNS "APPLICATION_HANDLE" INTEGER PATH 'application_handle',
    "ACTIVITY_ID" INTEGER PATH 'activity_id',
    "UOW_ID" INTEGER PATH 'uow_id',
    "STMT_TEXT" VARCHAR(1024) PATH 'stmt_text',
    "TOTAL_ACT_TIME" INTEGER PATH 'activity_metrics/total_act_time',
    "TOTAL_ACT_WAIT_TIME" INTEGER PATH 'activity_metrics/total_act_wait_time'
  ) AS ACTMETRICS;
```

以下はこの照会の出力例です。

```
APPLICATION_HANDLE ACTIVITY_ID UOW_ID      ...
------------------ ----------- ----------- ...
                 7           1           2 ...

  1 record(s) selected.
```

照会の出力 (続き)。

```
... STMT_TEXT                                          ...
... ------------------------------------------------- ...
... select * from syscat.tables optimize for 1 row     ...
```

照会の出力 (続き)。

```
... TOTAL_ACT_TIME TOTAL_ACT_WAIT_TIME PERCENTAGE_WAIT_TIME
... -------------- -------------------- --------------------
...            459                    0                 0.00
```

MON_GET_ACTIVITY_DETAILS 表関数を使用することにより、システムで現在実行中のすべてのアクティビティーについての情報をキャプチャーする照会を作成します。

- 例 1: DB2 コマンド行プロセッサー (CLP) を使って以下のコマンドを実行します。

```
WITH A1 AS
   (SELECT * FROM TABLE(wlm_get_workload_occurrence_activities_v97(null, -1))
    WHERE activity_id > 0 )
SELECT A1.application_handle,
  A1.activity_id,
  A1.uow_id,
  total_act_time,
  total_act_wait_time,
  varchar(actmetrics.stmt_text, 50) AS stmt_text FROM A1,
TABLE(MON_GET_ACTIVITY_DETAILS(A1.application_handle, A1.uow_id,A1.activity_id, -1))
    AS ACTDETAILS,
XMLTABLE (XMLNAMESPACES( DEFAULT 'http://www.ibm.com/xmlns/prod/db2/mon'),
 '$actmetrics/db2_activity_details'
 PASSING XMLPARSE(DOCUMENT ACTDETAILS.DETAILS) AS "actmetrics"
 COLUMNS "STMT_TEXT" VARCHAR(1024) PATH 'stmt_text',
 "TOTAL_ACT_TIME" INTEGER PATH 'activity_metrics/total_act_time',
 "TOTAL_ACT_WAIT_TIME" INTEGER PATH 'activity_metrics/total_act_wait_time' )
 AS ACTMETRICS
```

以下はこの照会の出力例です。

```
APP...HANDLE  A..._ID  UOW_ID  T...ACT_TIME  T...WAIT_TIME
------------  -------  ------  ------------  -------------
15            1        5       16            5
15            1        3       17            5
7             1        49      0             0
SQL0445W Value "with A1 as (select * from table(wlm_get_workload
3 record(s) selected with 1 warning messages printed.
```

以下は、この照会の出力例の続きを示しています。

```
... STMT_TEXT
... ------------------------------------------------
... select name from sysibm.systables
... select * from sysibm.systables
... with A1 as (select * from table(wlm_get_workload_o
_occurrence_" has been truncated. SQLSTATE=01004

3 record(s) selected with 1 warning messages printed.
```

- 例 2 では FOR EACH ROW OF 節を使用して、wlm_get_workload_occurrence_activities_v97 関数呼び出しを MON_GET_ACTIVITY_DETAILS 関数のパラメーター・リストに組み込みます。DB2 コマンド行プロセッサー (CLP) を使って以下のコマンドを実行します。

```
SELECT actmetrics.application_handle,
  actmetrics.activity_id,
  actmetrics.uow_id,
  varchar(actmetrics.stmt_text, 50) AS stmt_text,
  actmetrics.total_act_time,
  actmetrics.total_act_wait_time,
  CASE WHEN actmetrics.total_act_time > 0
    THEN DEC
      ((FLOAT(actmetrics.total_act_wait_time)/FLOAT(actmetrics.total_act_time)) * 100, 5, 2)
    ELSE NULL
  END AS PERCENTAGE_WAIT_TIME
FROM TABLE(MON_GET_ACTIVITY_DETAILS(for each row of (select application_handle,
  uow_id,
  activity_id from table(wlm_get_workload_occurrence_activities_v97(NULL, -1))
  WHERE activity_id > 0), -1)) AS ACTDETAILS,
XMLTABLE (XMLNAMESPACES( DEFAULT 'http://www.ibm.com/xmlns/prod/db2/mon'),
  '$actmetrics/db2_activity_details'
  PASSING XMLPARSE(DOCUMENT ACTDETAILS.DETAILS) as "actmetrics"
  COLUMNS "APPLICATION_HANDLE" INTEGER PATH 'application_handle',
  "ACTIVITY_ID" INTEGER PATH 'activity_id',
  "UOW_ID" INTEGER PATH 'uow_id',
  "STMT_TEXT" VARCHAR(1024) PATH 'stmt_text',
  "TOTAL_ACT_TIME" INTEGER PATH 'activity_metrics/total_act_time',
  "TOTAL_ACT_WAIT_TIME" INTEGER
  PATH 'activity_metrics/total_act_wait_time' ) AS ACTMETRICS
```

以下はこの照会の出力例です。

```
APP...HANDLE  A..._ID  UOW_ID  STMT_TEXT                                          T...ACT_TIME  T...WT_TIME  P...WT_TIME
------------  -------  ------  -------------------------------------------------  ------------  ----------   -----------
93            1        8       SELECT actmetrics.application_handle, actmetrics.a  0             0            -
SQL0445W Value "SELECT actmetrics.application_handle, actmetrics.activity_id" has been truncated. SQLSTATE=01004
101           1        4       SELECT name FROM sysibm.systables                   12            0            0.00
101           1        3       SELECT 8 FROM sysibm.systables                      8             0            0.00

3 record(s) selected with 1 warning messages printed.
```

使用上の注意

MON_GET_ACTIVITY_DETAILS 関数は、単一のアクティビティーの詳細情報を 1 つの XML 文書として戻すため、出力フォーマットを最大限柔軟性のあるものにします。 XML 出力には、記述情報 (例えば、ステートメント・テキスト) とメトリックの両方が含まれます。出力は XML パーサーで直接解析でき、例に示すように XMLTABLE 関数でリレーショナル形式に変換することもできます。

この関数で報告されるメトリック (例えば、CPU 使用率) は、アクティビティーの存続期間中、定期的にアクティビティーにロールアップされます。したがって、この表関数で報告される値は、直前の ROLLUP 時のシステムの現行状態を反映しています。

アクティビティー・メトリックは、ワークロードの COLLECT ACTIVITY METRICS 節、またはデータベース・レベルの mon_act_metrics データベース構成パラメーターで制御されます。メトリックは、アクティビティーをサブミットする接続が、アクティビティー・メトリックが使用可能なワークロードまたはデータベースに関連付けられている場合に収集されます。アクティビティー・メトリックが、アクティビティーについて収集されない場合、すべてのメトリックが 0 として報告されます。

MON_GET_ACTIVITY_DETAILS 表関数は、アクティビティーが存在するメンバーごとに 1 行のデータを戻します。メトリックについて、メンバー全体の集約は実行されません。しかし、集約は SQL 照会を使用して実行できます。

DETAILS 列に戻される XML 文書のスキーマは、ファイル sqllib/misc/DB2MonRoutines.xsd で入手できます。詳細は、ファイル sqllib/misc/DB2MonCommon.xsd 内にあります。

戻される情報

表 115. MON_GET_ACTIVITY_DETAILS について戻される情報

列名	データ・タイプ	説明
APPLICATION_HANDLE	BIGINT	application_handle - アプリケーション・ハンドル
UOW_ID	INTEGER	uow_id - 作業単位 ID
ACTIVITY_ID	INTEGER	activity_id - アクティビティー ID
MEMBER	SMALLINT	member - データベース・メンバー
DETAILS	BLOB(8M)	アクティビティーの詳細を含む XML 文書。エレメントの説明については、本書の 454 ページの表 116を参照してください。

以下の例は、DETAILS 列で戻される XML 文書の構造を示しています。

```
<db2_activity_details xmlns="http://www.ibm.com/xmlns/prod/db2/mon" release="90700000">
  <member>0</member>
  <application_handle>70</application_handle>
  <activity_id>1</activity_id>
  <activity_state>IDLE</activity_state>
  <activity_type>READ_DML</activity_type>
  <uow_id>1</uow_id>
  ...
  <activity_metrics release="90700000">
    <lock_wait_time>2000</lock_wait_time>
    ...
  </activity_metrics>
</db2_activity_details>
```

完全スキーマについては、sqllib/misc/DB2MonRoutines.xsd を参照してください。本書では、以下のような非プリミティブ XML 型定義を使用しています。

```
<xs:simpleType name = "executable_id_type" >
  <xs:annotation>
     <xs:documentation>
       The binary Executable ID
     </xs:documentation>
  </xs:annotation>
```

```
   <xs:restriction base = "xs:hexBinary" >
      <xs:maxLength value = "32" />
   </xs:restriction>
</xs:simpleType>
```

戻される詳細メトリック

表 116. MON_GET_ACTIVITY_DETAILS について戻される詳細メトリック

エレメント名	データ・タイプ	説明または対応するモニター・エレメント
member	xs:nonNegativeInteger	member - データベース・メンバー
client_userid	xs:string(255)	CURRENT CLIENT_USERID 特殊レジスター
client_wrkstnname	xs:string(255)	CURRENT CLIENT_WRKSTNNAME 特殊レジスター
client_applname	xs:string(255)	CURRENT CLIENT_APPLNAME 特殊レジスター
client_acctng	xs:string(255)	CURRENT CLIENT_ACCTNG 特殊レジスター
application_handle	xs:nonNegativeInteger	application_handle - アプリケーション・ハンドル
coord_member	xs:nonNegativeInteger	coord_member - コーディネーター・メンバー
uow_id	xs:nonNegativeInteger	uow_id - 作業単位 ID
activity_id	xs:nonNegativeInteger	activity_id - アクティビティー ID
parent_uow_id	xs:nonNegativeInteger	parent_uow_id - 親作業単位 ID
parent_activity_id	xs:nonNegativeInteger	parent_activity_id - 親アクティビティー ID
activity_state	xs:string	activity_state - アクティビティー状態
activity_type	xs:string	activity_type - アクティビティー・タイプ
nesting_level	xs:nonNegativeInteger	stmt_nest_level - ステートメント・ネスティング・レベル
invocation_id	xs:nonNegativeInteger	stmt_invocation_id - ステートメント呼び出し ID
routine_id	xs:nonNegativeInteger	routine_id - ルーチン ID
utility_id	xs:nonNegativeInteger	utility_id - ユーティリティー ID
service_class_id	xs:integer	service_class_id - サービス・クラス
database_work_action_set_id	xs:nonNegativeInteger	db_work_action_set_id - データベース作業アクション・セット ID
database_work_class_id	xs:nonNegativeInteger	db_work_class_id - データベース作業クラス ID
service_class_work _action_set_id	xs:nonNegativeInteger	sc_work_action_set_id - サービス・クラス作業アクション・セット ID
service_class_work_class_id	xs:nonNegativeInteger	sc_work_class_id - サービス・クラス作業クラス ID
entry_time	xs:dateTime	entry_time - 入力時刻このアクティビティーがシステムに到達した時刻。
local_start_time	xs:dateTime	local_start_time - ローカル開始時刻。
last_reference_time	xs:dateTime	last_reference_time - 最終参照時刻。要求がこのアクティビティーで発生するたびに、このフィールドは更新されます。
package_name	xs:string (128)	package_name - パッケージ名
package_schema	xs:string (128)	package_schema - パッケージ・スキーマ
package_version_id	xs:string (128)	package_version_id - パッケージ・バージョン
section_number	xs:integer	section_number - セクション番号
stmt_pkg_cache_id	xs:nonNegativeInteger	stmt_pkgcache_id - ステートメント・パッケージ・キャッシュ ID

表 116. MON_GET_ACTIVITY_DETAILS について戻される詳細メトリック (続き)

エレメント名	データ・タイプ	説明または対応するモニター・エレメント
stmt_text	xs:string	stmt_text - SQL 動的ステートメント・テキスト。アクティビティーが動的 SQL であるか、またはステートメント・テキストが使用可能になっている静的 SQL である場合、このフィールドにはそのステートメント・テキストの最初の 1024 文字が入っています。そうでない場合、空ストリングが入っています。
effective_isolation	xs:string	effective_isolation - 有効な分離
effective_lock_timeout	xs:integer	effective_lock_timeout - 有効なロック・タイムアウト
effective_query_degree	xs:integer	effective_query_degree - 有効な照会の度合い
query_cost_estimate	xs:integer	query_cost_estimate - 照会コストの見積もり
qp_query_id	xs:nonNegativeInteger	qp_query_id - Query Patroller 照会 ID
num_remaps	xs:nonNegativeInteger	num_remaps - 再マップ数
concurrentdbcoordactivities_db_threshold_id	xs:int	concurrentdbcoordactivities_db_threshold_id - 並行データベース・コーディネーター・アクティビティーしきい値 ID
concurrentdbcoordactivities_db_threshold_value	xs:long	concurrentdbcoordactivities_db_threshold_value - 並行データベース・コーディネーター・アクティビティーしきい値
concurrentdbcoordactivities_db_threshold_queued	xs:short (1 = はい、0 = いいえ)	concurrentdbcoordactivities_db_threshold_queued - 並行データベース・コーディネーター・アクティビティーしきい値によるキュー待機
concurrentdbcoordactivities_db_threshold_violated	xs:short (1 = はい、0 = いいえ)	concurrentdbcoordactivities_db_threshold_violated - 並行データベース・コーディネーター・アクティビティーしきい値の違反
concurrentdbcoordactivities_superclass_threshold_id	xs:int	concurrentdbcoordactivities_superclass_threshold_id - 並行データベース・コーディネーター・アクティビティーのスーパークラスしきい値 ID
concurrentdbcoordactivities_superclass_threshold_value	xs:long	concurrentdbcoordactivities_superclass_threshold_value - 並行データベース・コーディネーター・アクティビティーのスーパークラスしきい値
concurrentdbcoordactivities_superclass_threshold_queued	xs:short (1 = はい、0 = いいえ)	concurrentdbcoordactivities_superclass_threshold_queued - 並行データベース・コーディネーター・アクティビティーのスーパークラスしきい値によるキュー待機
concurrentdbcoordactivities_superclass_threshold_violated	xs:short (1 = はい、0 = いいえ)	concurrentdbcoordactivities_superclass_threshold_violated - 並行データベース・コーディネーター・アクティビティーのスーパークラスしきい値の違反
concurrentdbcoordactivities_subclass_threshold_id	xs:int	concurrentdbcoordactivities_subclass_threshold_id - 並行データベース・コーディネーター・アクティビティーのサブクラスしきい値 ID
concurrentdbcoordactivities_subclass_threshold_value	xs:long	concurrentdbcoordactivities_subclass_threshold_value - 並行データベース・コーディネーター・アクティビティーのサブクラスしきい値
concurrentdbcoordactivities_subclass_threshold_queued	xs:short (1 = はい、0 = いいえ)	concurrentdbcoordactivities_subclass_threshold_queued - 並行データベース・コーディネーター・アクティビティーのサブクラスしきい値によるキュー待機

表 116. MON_GET_ACTIVITY_DETAILS について戻される詳細メトリック (続き)

エレメント名	データ・タイプ	説明または対応するモニター・エレメント
concurrentdbcoordactivities _subclass_threshold_violated	xs:short (1 = はい、0 = いいえ)	concurrentdbcoordactivities_subclass_threshold_violated - 並行データベース・コーディネーター・アクティビティーのサブクラスしきい値の違反
concurrentdbcoordactivities _work_action_set_threshold _id	xs:int	concurrentdbcoordactivities_work_action_set_threshold_id - 並行データベース・コーディネーター・アクティビティーの作業アクション・セットしきい値 ID
concurrentdbcoordactivities _work_action_set_threshold _value	xs:long	concurrentdbcoordactivities_work_action_set_threshold_value - 並行データベース・コーディネーター・アクティビティーの作業アクション・セットしきい値
concurrentdbcoordactivities _work_action_set_threshold _queued	xs:short (1 = はい、0 = いいえ)	concurrentdbcoordactivities_work_action_set_threshold_queued - 並行データベース・コーディネーター・アクティビティーの作業アクション・セットしきい値によるキュー待機
concurrentdbcoordactivities _work_action_set_threshold _violated	xs:short (1 = はい、0 = いいえ)	concurrentdbcoordactivities_work_action_set_threshold_violated - 並行データベース・コーディネーター・アクティビティーの作業アクション・セットしきい値の違反
estimatedsqlcost_threshold_id	xs:int	estimatedsqlcost_threshold_id - 見積もり SQL コストしきい値 ID
estimatedsqlcost_threshold _value	xs:long	estimatedsqlcost_threshold_value - 見積もり SQL コストしきい値
estimatedsqlcost_threshold _violated	xs:short (1 = はい、0 = いいえ)	estimatedsqlcost_threshold_violated - 見積もり SQL コストしきい値の違反
sqltempspace_threshold_id	xs:int	sqltempspace_threshold_id - SQL 一時スペースしきい値 ID
sqltempspace_threshold _value	xs:long	sqltempspace_threshold_value - SQL 一時スペースしきい値
sqltempspace_threshold _violated	xs:short (1 = はい、0 = いいえ)	sqltempspace_threshold_violated - SQL 一時スペースしきい値の違反
sqlrowsreturned_threshold_id	xs:int	sqlrowsreturned_threshold_id - 戻される SQL 読み取り行数しきい値 ID
sqlrowsreturned_threshold _value	xs:long	sqlrowsreturned_threshold_value - 戻される SQL 読み取り行数しきい値
sqlrowsreturned_threshold _violated	xs:short (1 = はい、0 = いいえ)	sqlrowsreturned_threshold_violated - 戻される SQL 読み取り行数しきい値の違反
activitytotaltime_threshold_id	xs:int	activitytotaltime_threshold_id - アクティビティー合計時間しきい値 ID
activitytotaltime_threshold _value	xs:dateTime	activitytotaltime_threshold_value - アクティビティー合計時間しきい値
activitytotaltime_threshold _violated	xs:short (1 = はい、0 = いいえ)	activitytotaltime_threshold_violated - アクティビティー合計時間しきい値の違反
cputime_threshold_id	xs:int	cputime_threshold_id - CPU 時間しきい値 ID
cputime_threshold_value	xs:long	cputime_threshold_value - CPU 時間しきい値
cputime_threshold_violated	xs:short (1 = はい、0 = いいえ)	cputime_threshold_violated - CPU 時間しきい値の違反
cputimeinsc_threshold_id	xs:int	cputimeinsc_threshold_id - サービス・クラス内の CPU 時間しきい値 ID

表 116. MON_GET_ACTIVITY_DETAILS について戻される詳細メトリック (続き)

エレメント名	データ・タイプ	説明または対応するモニター・エレメント
cputimeinsc_threshold_value	xs:long	cputimeinsc_threshold_value - サービス・クラス内の CPU 時間しきい値
cputimeinsc_threshold _violated	xs:short (1 = はい、0 = いいえ)	cputimeinsc_threshold_violated - サービス・クラス内の CPU 時間しきい値の違反
sqlrowsread_threshold_id	xs:int	sqlrowsread_threshold_ID - SQL 読み取り行数しきい値 ID
sqlrowsread_threshold_value	xs:long	sqlrowsread_threshold_value - SQL 読み取り行数しきい値
sqlrowsread_threshold_violated	xs:short (1 = はい、0 = いいえ)	sqlrowsread_threshold_violated - SQL SQL 読み取り行数しきい値の違反
sqlrowsreadinsc_threshold_id	xs:int	sqlrowsreadinsc_threshold_id - サービス・クラス内の SQL 読み取り行数しきい値 ID
sqlrowsreadinsc_threshold _value	xs:long	sqlrowsreadinsc_threshold_value - サービス・クラス内の SQL 読み取り行数しきい値
sqlrowsreadinsc_threshold _violated	xs:short (1 = はい、0 = いいえ)	sqlrowsreadinsc_threshold_violated - サービス・クラス内の SQL 読み取り行数しきい値の違反
aggsqltempspace_threshold_id	xs:int	aggsqltempspace_threshold_id - AggSQL 一時スペースしきい値 ID。
aggsqltempspace_threshold _value	xs:long	aggsqltempspace_threshold_value - AggSQL 一時スペースしきい値
aggsqltempspace_threshold _violated	xs:short (1 = はい、0 = いいえ)	aggsqltempspace_threshold_violated - AggSQL 一時スペースしきい値の違反
audit_events_total	xs:nonNegativeInteger	audit_events_total - 監査イベント合計数
audit_subsystem_wait_time	xs:nonNegativeInteger	audit_subsystem_wait_time - 監査サブシステム待機時間
audit_subsystem_waits_total	xs:nonNegativeInteger	audit_subsystem_waits_total - 監査サブシステム待機の合計
audit_file_write_wait_time	xs:nonNegativeInteger	audit_file_write_wait_time - 監査ファイル書き込み待機時間
audit_file_writes_total	xs:nonNegativeInteger	audit_file_writes_total - 書き込まれた監査ファイルの合計数
coord_stmt_exec_time		coord_stmt_exec_time - コーディネーター・エージェントによるステートメントの実行時間
deadlocks	xs:nonNegativeInteger	deadlocks - デッドロック検出数
diaglog_writes_total	xs:nonNegativeInteger	diaglog_writes_total - 診断ログ・ファイル書き込みの合計
diaglog_write_wait_time	xs:nonNegativeInteger	diaglog_write_wait_time - 診断ログ・ファイルの書き込み待機時間
direct_read_time	xs:nonNegativeInteger	direct_read_time - 直接読み取り時間
direct_write_time	xs:nonNegativeInteger	direct_write_time - 直接書き込み時間
direct_read_reqs	xs:nonNegativeInteger	direct_read_reqs - 直接読み取り要求
direct_reads	xs:nonNegativeInteger	direct_reads - データベースからの直接読み取り
direct_write_reqs	xs:nonNegativeInteger	direct_write_reqs - 直接書き込み要求
direct_writes	xs:nonNegativeInteger	direct_writes - データベースへの直接書き込み
fcm_recv_volume	xs:nonNegativeInteger	fcm_recv_volume - FCM 受信ボリューム
fcm_recv_wait_time	xs:nonNegativeInteger	fcm_recv_wait_time - FCM 受信待機時間
fcm_recvs_total	xs:nonNegativeInteger	fcm_recvs_total - FCM 合計受信数
fcm_message_recv_volume	xs:nonNegativeInteger	fcm_message_recv_volume - FCM メッセージ受信ボリューム

表 116. MON_GET_ACTIVITY_DETAILS について戻される詳細メトリック (続き)

エレメント名	データ・タイプ	説明または対応するモニター・エレメント
fcm_message_recvs_total	xs:nonNegativeInteger	fcm_message_recvs_total - FCM メッセージ合計受信数
fcm_message_recv_wait_time	xs:nonNegativeInteger	fcm_message_recv_wait_time - FCM メッセージの受信待機時間
fcm_message_send_volume	xs:nonNegativeInteger	fcm_message_send_volume - FCM メッセージ送信ボリューム
fcm_message_send_wait_time	xs:nonNegativeInteger	fcm_message_send_wait_time - FCM メッセージ送信待ち時間
fcm_message_sends_total	xs:nonNegativeInteger	fcm_message_sends_total - FCM メッセージ合計送信数
fcm_send_volume	xs:nonNegativeInteger	fcm_send_volume - FCM 送信ボリューム
fcm_send_wait_time	xs:nonNegativeInteger	fcm_send_wait_time - FCM 送信待ち時間
fcm_sends_total	xs:nonNegativeInteger	fcm_sends_total - FCM 合計送信数
fcm_tq_recv_wait_time	xs:nonNegativeInteger	fcm_tq_recv_wait_time - FCM 表キューの受信待ち時間
fcm_tq_send_wait_time	xs:nonNegativeInteger	fcm_tq_send_wait_time - FCM 表キューの送信待ち時間
fcm_tq_recv_volume	xs:nonNegativeInteger	fcm_tq_recv_volume - FCM 表キューの受信ボリューム
fcm_tq_recvs_total	xs:nonNegativeInteger	fcm_tq_recvs_total - FCM 表キューの合計受信数
fcm_tq_send_volume	xs:nonNegativeInteger	fcm_tq_send_volume - FCM 表キューの送信ボリューム
fcm_tq_sends_total	xs:nonNegativeInteger	fcm_tq_sends_total - FCM 表キューの合計送信数
tq_tot_send_spills	xs:nonNegativeInteger	tq_tot_send_spills - オーバーフローした表キュー・バッファーの合計数
lock_escals	xs:nonNegativeInteger	lock_escals - ロック・エスカレーション数
lock_timeouts	xs:nonNegativeInteger	lock_timeouts - ロック・タイムアウト数
lock_wait_time	xs:nonNegativeInteger	lock_wait_time - ロック待機中の時間
lock_waits	xs:nonNegativeInteger	lock_waits - ロック待機数
log_buffer_wait_time	xs:nonNegativeInteger	log_buffer_wait_time - ログ・バッファー待ち時間
log_disk_wait_time	xs:nonNegativeInteger	log_disk_wait_time - ログ・ディスク待機時間
log_disk_waits_total	xs:nonNegativeInteger	log_disk_waits_total - ログ・ディスク待機の合計
num_lw_thresh_exceeded	xs:nonNegativeInteger	num_lw_thresh_exceeded - しきい値を超えた回数
pool_data_l_reads	xs:nonNegativeInteger	pool_data_l_reads - バッファー・プール・データの論理読み取り
pool_data_p_reads	xs:nonNegativeInteger	pool_data_p_reads - バッファー・プール・データの物理読み取り
pool_data_writes	xs:nonNegativeInteger	pool_data_writes - バッファー・プールへのデータの書き込み
pool_index_l_reads	xs:nonNegativeInteger	pool_index_l_reads - バッファー・プール索引の論理読み取り
pool_index_p_reads	xs:nonNegativeInteger	pool_index_p_reads - バッファー・プール索引の物理読み取り
pool_index_writes	xs:nonNegativeInteger	pool_index_writes - バッファー・プール索引の書き込み
pool_read_time	xs:nonNegativeInteger	pool_read_time - バッファー・プール物理読み取り時間の合計
pool_temp_data_l_reads	xs:nonNegativeInteger	pool_temp_data_l_reads - バッファー・プール一時データの論理読み取り

表 116. MON_GET_ACTIVITY_DETAILS について戻される詳細メトリック (続き)

エレメント名	データ・タイプ	説明または対応するモニター・エレメント
pool_temp_data_p_reads	xs:nonNegativeInteger	pool_temp_data_p_reads - バッファー・プール一時データの物理読み取り
pool_temp_index_l_reads	xs:nonNegativeInteger	pool_temp_index_l_reads - バッファー・プール一時索引の論理読み取り
pool_temp_index_p_reads	xs:nonNegativeInteger	pool_temp_index_p_reads - バッファー・プール一時索引の物理読み取り
pool_temp_xda_l_reads	xs:nonNegativeInteger	pool_temp_xda_l_reads - バッファー・プール一時 XDA データの論理読み取り
pool_temp_xda_p_reads	xs:nonNegativeInteger	pool_temp_xda_p_reads - バッファー・プール一時 XDA データの物理読み取り
pool_write_time	xs:nonNegativeInteger	pool_write_time - バッファー・プール物理書き込み時間の合計
pool_xda_l_reads	xs:nonNegativeInteger	pool_xda_l_reads - バッファー・プール XDA データの論理読み取り
pool_xda_p_reads	xs:nonNegativeInteger	pool_xda_p_reads - バッファー・プール XDA データの物理読み取り
pool_xda_writes	xs:nonNegativeInteger	pool_xda_writes - バッファー・プール XDA データの書き込み
num_log_buffer_full	xs:nonNegativeInteger	num_log_buffer_full - フル・ログ・バッファーの回数
rows_modified	xs:nonNegativeInteger	rows_modified - 変更された行数
rows_read	xs:nonNegativeInteger	rows_read - 読み取り行数
rows_returned	xs:nonNegativeInteger	rows_returned - 戻り行数
stmt_exec_time	xs:nonNegativeInteger	stmt_exec_time - ステートメントの実行時間
thresh_violations	xs:nonNegativeInteger	thresh_violations - しきい値違反の回数
total_cpu_time	xs:nonNegativeInteger	total_cpu_time - 合計 CPU 時間
total_act_time	xs:nonNegativeInteger	total_act_time - 合計アクティビティー時間
total_act_wait_time	xs:nonNegativeInteger	total_act_wait_time - 合計アクティビティー待機時間
total_app_section_executions	xs:nonNegativeInteger	total_app_section_executions - セクション実行の合計回数
total_routine_invocations	xs:nonNegativeInteger	total_routine_invocations - ルーチン呼び出しの合計回数
total_routine_non_sect_proc_time	xs:nonNegativeInteger	total_routine_non_sect_proc_time - 非セクション処理時間
total_routine_non_sect_time	xs:nonNegativeInteger	total_routine_non_sect_time - 非セクション・ルーチン実行時間
total_routine_time	xs:nonNegativeInteger	total_routine_time - ルーチン時間の合計
total_routine_user_code_proc_time	xs:nonNegativeInteger	total_routine_user_code_proc_time - ルーチン・ユーザー・コード処理時間の合計
total_routine_user_code_time	xs:nonNegativeInteger	total_routine_user_code_time - ルーチン・ユーザー・コード時間の合計
total_section_proc_time	xs:nonNegativeInteger	total_section_proc_time - セクション処理時間の合計
total_section_sort_time	xs:nonNegativeInteger	total_section_sort_time - セクション・ソート時間合計
total_section_sort_proc_time	xs:nonNegativeInteger	total_section_sort_proc_time - セクション・ソート処理時間合計

表 116. MON_GET_ACTIVITY_DETAILS について戻される詳細メトリック (続き)

エレメント名	データ・タイプ	説明または対応するモニター・エレメント
total_section_sorts	xs:nonNegativeInteger	total_section_sorts - セクション・ソート合計
total_section_time	xs:nonNegativeInteger	total_section_time - セクション時間の合計
total_sorts	xs:nonNegativeInteger	total_sorts - ソート合計
post_threshold_sorts	xs:nonNegativeInteger	post_threshold_sorts - ポストしきい値ソート
post_shrthreshold_sorts	xs:nonNegativeInteger	post_shrthreshold_sorts - ポスト共有しきい値ソート
sort_overflows	xs:nonNegativeInteger	sort_overflows - ソート・オーバーフロー
executable_id	executable_id_type	executable_id - 実行可能 ID
wlm_queue_time_total	xs:nonNegativeInteger	wlm_queue_time_total - ワークロード・マネージャー合計キュー時間
wlm_queue_assignments_total	xs:nonNegativeInteger	wlm_queue_assignments_total - ワークロード・マネージャー合計キュー割り当て数
eff_stmt_text	xs:string	eff_stmt_text - 有効なステートメント・テキスト。ステートメント・コンセントレーターによって実行されたリテラル置換の後の、対象ステートメント・テキストの最初の 1024 文字です。ステートメント・コンセントレーターが使用可能で、このステートメントがステートメント・コンセントレーターによって変更された場合のみ存在します。
wl_work_action_set_id	xs:nonNegativeInteger	wl_work_action_set_id - ワークロード作業アクション・セット ID モニター・エレメント
wl_work_class_id	xs:nonNegativeInteger	wl_work_class_id - ワークロード作業クラス ID
concurrentdbcoordactivities _wl_was_threshold_id	xs:int	concurrentdbcoordactivities_wl_was_threshold_id - 並行データベース・コーディネーター・アクティビティーのワークロード作業アクション・セットしきい値 ID
concurrentdbcoordactivities _wl_was_threshold_value	xs:long	concurrentdbcoordactivities_wl_was_threshold_value - 並行データベース・コーディネーター・アクティビティーのワークロード作業アクション・セットしきい値
concurrentdbcoordactivities _wl_was_threshold_queued	xs:short (1 = はい、0 = いいえ)	concurrentdbcoordactivities_wl_was_threshold_queued - 並行データベース・コーディネーター・アクティビティーのワークロード作業アクション・セットしきい値によるキュー待機
concurrentdbcoordactivities _wl_was_threshold_violated	xs:short (1 = はい、0 = いいえ)	concurrentdbcoordactivities_wl_was_threshold_violated - 並行データベース・コーディネーター・アクティビティーのワークロード作業アクション・セットしきい値の違反

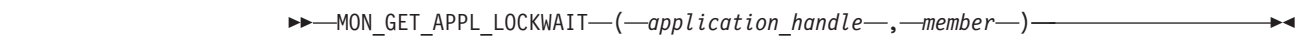
MON_GET_APPL_LOCKWAIT - アプリケーションが待機しているロックについての情報の収集

MON_GET_APPL_LOCKWAIT 表関数は、(現在のデータベースに接続されている) 各アプリケーションのエージェントが取得を待機しているすべてのロックについての情報を戻します。

注: バージョン 9.7 のフィックスパック 1 以前で作成されたデータベースを使用している場合、このルーチンを実行するには、既に db2updv97 コマンドを実行済みでなければなりません。バージョン 9.7 以前に作成されたデータベースを使用してい

る場合は、db2updv97 コマンドを実行する必要はありません (データベースのマイグレーションによって自動的にカタログ更新が実行されるため)。バージョン 9.7 にダウングレードすると、このルーチンは機能しなくなります。

ロックについての情報を取得するには MON_GET_APPL_LOCKWAIT、MON_FORMAT_LOCK_NAME、および MON_GET_LOCKS表関数を使用してください。SNAPLOCKWAIT 管理ビューと SNAP_GET_LOCKWAIT 表関数、および SNAPLOCK 管理ビューと SNAP_GET_LOCK 表関数は バージョン 9.7 フィックスパック 1 で非推奨になりました。



スキーマは SYSPROC です。

表関数パラメーター

application_handle
現在接続されているのと同じデータベース内の有効なアプリケーション・ハンドルを指定する、オプションのタイプ BIGINT の入力パラメーター。この引数が NULL である場合、ロックの取得を現在待機しているすべてのアプリケーションに関するロックが検索されます。

member
現在接続されているデータベースと同じインスタンス内の有効なメンバーを指定する、タイプ INTEGER の入力パラメーター。現在のデータベース・メンバーの場合は -1、すべてのアクティブ・メンバーの場合は -2 を指定します。NULL 値を指定すると、-1 が設定されます。

許可

以下のいずれかの権限または特権が必要です。

- SYSADM 権限
- SYSMON 権限
- MON_GET_APPL_LOCKWAIT 表関数に対する EXECUTE 特権。

例

この例のシナリオでは MON_GET_APPL_LOCKWAIT 表関数を使用して、セッション許可 ID USER1 に関するハングしているアプリケーションを調べます。

1. 次のように WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES_V97 表関数を使用して、SESSION_USER 値 USER1 のすべての接続に関するアプリケーション・ハンドルを検索します。

   ```
   SELECT COORD_PARTITION_NUM,
        APPLICATION_HANDLE
   FROM TABLE(WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES_V97('','',-2))
   WHERE SESSION_USER = 'USER1'
   ```

   この照会は、以下の出力を戻します。

```
COORD_PARTITION_NUM              APPLICATION_HANDLE
-------------------------------  --------------------------
        2                        131130

1 record(s) selected.
```

2. 次のように WLM_GET_SERVICE_CLASS_AGENTS_V97 表関数を使用して、すべてのデータベース・パーティションで、この接続を扱っているすべてのエージェントに関する現行情報を取得します。

```
SELECT SUBSTR(CHAR(DBPARTITIONNUM),1,3) AS DBPART,
     SUBSTR(CHAR(APPLICATION_HANDLE),1,7) AS APP_ID,
     SUBSTR(CHAR(WORKLOAD_OCCURRENCE_ID),1,7) AS WLO_ID,
     SUBSTR(CHAR(AGENT_TID),1,7) AS AGENT_ID,
     SUBSTR(CHAR(AGENT_TYPE),1,12) AS AGENT_TYPE,
     SUBSTR(AGENT_STATE,1, 8) AS STATE,
     SUBSTR(EVENT_TYPE,1, 8) AS EV_TYPE,
     SUBSTR(EVENT_OBJECT,1,12) AS EV_OBJECT
FROM TABLE(WLM_GET_SERVICE_CLASS_AGENTS_V97('','',131130,-2))
ORDER BY AGENT_TYPE, DBPART
```

この照会は、以下の出力を戻します。

```
DBPART  APP_ID  WLO_ID  AGENT_ID  AGENT_TYPE  STATE   EV_TYPE  EV_OBJECT
------  ------- ------- --------- ----------- ------  -------- ----------
2       131130  1       3110      COORDINATOR ACTIVE  WAIT      REQUEST
0       131130  1       7054      PDBSUBAGENT ACTIVE  ACQUIRE   LOCK
1       131130  1       5709      PDBSUBAGENT ACTIVE  ACQUIRE   LOCK
2       131130  1       5960      PDBSUBAGENT ACTIVE  ACQUIRE   LOCK

  4 record(s) selected.
```

タイプ LOCK のイベント・オブジェクトに対するタイプ ACQUIRE のイベントはロック待機シナリオを示しているため、待機の対象となっているオブジェクトは何か、および何がそのロックを保持しているかを調べる必要があります。

3. アプリケーションが待機しているすべてのロックを判別するには、アプリケーション・ハンドル 131130 およびメンバー -2 を入力パラメーターとして指定して MON_GET_APPL_LOCKWAIT 表関数を呼び出します。

```
SELECT lock_name,
       hld_member AS member,
       hld_agent_tid as TID,
       hld_application_handle AS HLD_APP FROM
       TABLE (MON_GET_APPL_LOCKWAIT(131130, -2))
```

この照会は、以下の出力を戻します。

```
LOCK_NAME                  MEMBER TID    HLD_APP
-------------------------- ------ ------ -------
00030005000000000280000452    0   1234   65564
00030005000000000280000452    1   5478   65564
00030005000000000280000452    2   4678   65564

  3 record(s) selected.
```

4. WLM_SERVICE_CLASS_WORKLOAD_OCCURRENCES_V97 表関数を呼び出して、ロックを保持している (アプリケーション・ハンドル 65564 の) アプリケーションについて詳細情報を検出します。

```
SELECT  SYSTEM_AUTH_ID,
   APPLICATION_NAME AS APP_NAME,
   WORKLOAD_NAME AS WORKLOAD,
   WORKLOAD_OCCURRENCE_STATE AS WL_STATE
FROM TABLE(WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES_V97('','',-2))
WHERE APPLICATION_HANDLE = 65564
```

この照会は、以下の出力を戻します。

```
SYSTEM_AUTH_ID APP_NAME WORKLOAD                WL_STATE
-------------- -------- ---------------------- ----------
     ZURBIE    db2bp    SYSDEFAULTUSERWORKLOAD  UOWWAIT

  1 record(s) selected
```

戻される情報

戻される列は、次のような領域の情報を提供します。

- 以下の列は、アプリケーションが取得を現在待機しているロックについての詳細を表します。

  LOCK_WAIT_START_TIME、 LOCK_NAME、 LOCK_OBJECT_TYPE、 LOCK_MODE、 LOCK_CURRENT_MODE、 LOCK_MODE_REQUESTED、 LOCK_STATUS、 LOCK_ESCALATION、 LOCK_ATTRIBUTES、 LOCK_RRIID、 LOCK_COUNT、 TBSP_ID、 TAB_FILE_ID、 SUBSECTION_NUMBER。
- 以下の列は、このロックの取得を待機しているアプリケーションについての詳細を表します。

  REQ_APPLICATION_HANDLE、 REQ_AGENT_TID、 REQ_MEMBER、 REQ_EXECUTABLE_ID
- 以下の列は、ロックを現在保持しているアプリケーションについての詳細を表します。

  HLD_APPLICATION_HANDLE、HLD_MEMBER、ADDITIONAL_DETAILS

表 117. MON_GET_APPL_LOCKWAIT 表関数によって戻される情報

列名	データ・タイプ	説明またはモニター・エレメント
LOCK_WAIT_START_TIME	TIMESTAMP	lock_wait_start_time - ロック待機開始タイム・スタンプ
LOCK_NAME	VARCHAR(32)	lock_name - ロック名 ロックについての詳細情報を得るために、 MON_FORMAT_LOCK_NAME 表関数を使用して内部名をフォーマット設定できます。例えば、表ロックの場合には、ロックが参照している表および表スペースを検出できます。
LOCK_OBJECT_TYPE	VARCHAR(32)	lock_object_type - 待機中のロック対象タイプ 可能な値については、 『lock_object_type - 待機中のロック対象タイプ』モニター・エレメントを参照してください
LOCK_OBJECT_TYPE_ID	CHAR(1) FOR BIT DATA	内部使用のために予約済み

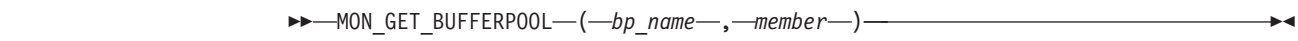
表 117. MON_GET_APPL_LOCKWAIT 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明またはモニター・エレメント
LOCK_MODE	VARCHAR(3)	lock_mode - ロック・モード このロックを保持しているアプリケーションが見つからない場合、NULL 値が戻されます。
LOCK_CURRENT_MODE	VARCHAR(3)	lock_current_mode - 移行前の元のロック・モード 移行が行われなかった場合は、NULL 値が戻されます。
LOCK_MODE_REQUESTED	VARCHAR(3)	lock_mode_requested - 要求されているロック・モード
LOCK_STATUS	CHAR(1)	lock_status - ロック状況
LOCK_ESCALATION	CHAR(1)	lock_escalation - ロック・エスカレーション
LOCK_ATTRIBUTES	CHAR(16)	lock_attributes - ロック属性
LOCK_RRIID	BIGINT	内部使用のために予約済み
LOCK_COUNT	BIGINT	内部使用のために予約済み
TBSP_ID	BIGINT	tablespace_id - 表スペース ID
TAB_FILE_ID	BIGINT	table_file_id - 表ファイル ID
SUBSECTION_NUMBER	BIGINT	ss_number - サブセクション番号 サブセクション番号を入手できない場合には NULL 値が戻されます。
REQ_APPLICATION_HANDLE	BIGINT	req_application_handle - 要求元のアプリケーション・ハンドル
REQ_AGENT_TID	BIGINT	req_agent_tid - 要求元のエージェント TID
REQ_MEMBER	SMALLINT	req_member - 要求元のメンバー
REQ_EXECUTABLE_ID	VARCHAR (32)	req_executable_id - 要求元の実行可能 ID
HLD_APPLICATION_HANDLE	BIGINT	hld_application_handle - 保持しているアプリケーション・ハンドル このロックを保持しているアプリケーションが不明または見つからない場合には、NULL 値が戻されます。
HLD_MEMBER	SMALLINT	hld_member - 保持しているメンバー
ADDITIONAL_DETAILS	BLOB(100K)	内部使用のために予約済み

MON_GET_BUFFERPOOL 表関数 - バッファー・プール・メトリックの取得

MON_GET_BUFFERPOOL 表関数は、1 つ以上のバッファー・プールのモニター・メトリックを戻します。

構文



スキーマは SYSPROC です。

表関数パラメーター

bp_name
この関数を呼び出すときに現在接続されているデータベース内の有効なバッファー・プール名を指定する、タイプ VARCHAR(128) の入力引数。引数が NULL または空ストリングである場合、データベース内のすべてのバッファー・プールについてメトリックが取得されます。

member
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なメンバーを指定する、タイプ INTEGER の入力引数。現行のデータベース・メンバーには -1、すべてのデータベース・メンバーには -2 を指定します。NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

MON_GET_BUFFERPOOL 関数に対する EXECUTE 特権。

例

バッファー・プール・ヒット率を計算します。

```
WITH BPMETRICS AS (
    SELECT bp_name,
           pool_data_l_reads + pool_temp_data_l_reads +
           pool_index_l_reads + pool_temp_index_l_reads +
           pool_xda_l_reads + pool_temp_xda_l_reads as logical_reads,
           pool_data_p_reads + pool_temp_data_p_reads +
           pool_index_p_reads + pool_temp_index_p_reads +
           pool_xda_p_reads + pool_temp_xda_p_reads as physical_reads,
           member
    FROM TABLE(MON_GET_BUFFERPOOL('',-2)) AS METRICS)
   SELECT
    VARCHAR(bp_name,20) AS bp_name,
    logical_reads,
    physical_reads,
    CASE WHEN logical_reads > 0
     THEN DEC((1 - (FLOAT(physical_reads) / FLOAT(logical_reads))) * 100,5,2)
     ELSE NULL
    END AS HIT_RATIO,
    member
   FROM BPMETRICS;
```

以下はこの照会の出力例です。

```
BP_NAME          LOGICAL_READS    PHYSICAL_READS HIT_RATIO MEMBER
---------------- ---------------- -------------- --------- ---------
IBMDEFAULTBP                  619            385     37.80         0
IBMSYSTEMBP4K                   0              0         -         0
IBMSYSTEMBP8K                   0              0         -         0
IBMSYSTEMBP16K                  0              0         -         0
IBMSYSTEMBP32K                  0              0         -         0

  5 record(s) selected.
```

照会の出力 (続き)。

```
... HIT_RATIO MEMBER
... --------- ------
...     37.80      0
...         -      0
...         -      0
...         -      0
...         -      0
```

使用上の注意

MON_GET_BUFFERPOOL 表関数は、データベース・バッファー・プールごとおよびデータベース・メンバーごとに 1 行のデータを戻します。データベース・メンバー全体からの集約は実行されません。ただし、集約は例に示されるように SQL 照会を使用して実行できます。

この関数によって収集されるメトリックは、mon_obj_metrics 構成パラメーターを使用してデータベース・レベルで制御されます。デフォルトでは、メトリック収集は有効になります。

戻される情報

表 118. MON_GET_BUFFERPOOL について戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
BP_NAME	VARCHAR(128)	
MEMBER	SMALLINT	member - データベース・メンバー
AUTOMATIC	SMALLINT	automatic - バッファー・プール自動
DIRECT_READS	BIGINT	direct_reads - データベースからの直接読み取り
DIRECT_READ_REQS	BIGINT	direct_read_reqs - 直接読み取り要求
DIRECT_WRITES	BIGINT	direct_writes - データベースへの直接書き込み
DIRECT_WRITE_REQS	BIGINT	direct_write_reqs - 直接書き込み要求
POOL_DATA_L_READS	BIGINT	pool_data_l_reads - バッファー・プール・データの論理読み取り
POOL_TEMP_DATA_L_READS	BIGINT	pool_temp_data_l_reads - バッファー・プール一時データの論理読み取り
POOL_XDA_L_READS	BIGINT	pool_xda_l_reads - バッファー・プール XDA データの論理読み取り
POOL_TEMP_XDA_L_READS	BIGINT	pool_temp_xda_l_reads - バッファー・プール一時 XDA データの論理読み取り
POOL_INDEX_L_READS	BIGINT	pool_index_l_reads - バッファー・プール索引の論理読み取り

表 118. MON_GET_BUFFERPOOL について戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
POOL_TEMP_INDEX_L_READS	BIGINT	pool_temp_index_l_reads - バッファー・プール一時索引の論理読み取り
POOL_DATA_P_READS	BIGINT	pool_data_p_reads - バッファー・プール・データの物理読み取り
POOL_TEMP_DATA_P_READS	BIGINT	pool_temp_data_p_reads - バッファー・プール一時データの物理読み取り
POOL_XDA_P_READS	BIGINT	pool_xda_p_reads - バッファー・プール XDA データの物理読み取り
POOL_TEMP_XDA_P_READS	BIGINT	pool_temp_xda_p_reads - バッファー・プール一時 XDA データの物理読み取り
POOL_INDEX_P_READS	BIGINT	pool_index_p_reads - バッファー・プール索引の物理読み取り
POOL_TEMP_INDEX_P_READS	BIGINT	pool_temp_index_p_reads - バッファー・プール一時索引の物理読み取り
POOL_DATA_WRITES	BIGINT	pool_data_writes - バッファー・プールへのデータの書き込み
POOL_XDA_WRITES	BIGINT	pool_xda_writes - バッファー・プール XDA データの書き込み
POOL_INDEX_WRITES	BIGINT	pool_index_writes - バッファー・プール索引の書き込み
DIRECT_READ_TIME	BIGINT	direct_read_time - 直接読み取り時間
DIRECT_WRITE_TIME	BIGINT	direct_write_time - 直接書き込み時間
POOL_READ_TIME	BIGINT	pool_read_time - バッファー・プール物理読み取り時間の合計
POOL_WRITE_TIME	BIGINT	pool_write_time - バッファー・プール物理書き込み時間の合計
POOL_ASYNC_DATA_READS	BIGINT	pool_async_data_reads - バッファー・プール非同期データ読み取り
POOL_ASYNC_DATA_READ_REQS	BIGINT	pool_async_data_read_reqs - バッファー・プール非同期読み取り要求
POOL_ASYNC_DATA_WRITES	BIGINT	pool_async_data_writes - バッファー・プール非同期データ書き込み
POOL_ASYNC_INDEX_READS	BIGINT	pool_async_index_reads - バッファー・プール非同期索引読み取り
POOL_ASYNC_INDEX_READ_REQS	BIGINT	pool_async_index_read_reqs - バッファー・プール非同期索引読み取り要求
POOL_ASYNC_INDEX_WRITES	BIGINT	pool_async_index_writes - バッファー・プール非同期索引書き込み
POOL_ASYNC_XDA_READS	BIGINT	pool_async_xda_reads - バッファー・プール非同期 XDA データ読み取り
POOL_ASYNC_XDA_READ_REQS	BIGINT	pool_async_xda_read_reqs - バッファー・プール非同期 XDA 読み取り要求
POOL_ASYNC_XDA_WRITES	BIGINT	pool_async_xda_writes - バッファー・プール非同期 XDA データ書き込み

表 118. MON_GET_BUFFERPOOL について戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
POOL_NO_VICTIM_BUFFER	BIGINT	pool_no_victim_buffer - バッファー・プールの非ビクティム・バッファー数
POOL_LSN_GAP_CLNS	BIGINT	pool_lsn_gap_clns - 起動されたバッファー・プール・ログ・スペース・クリーナー
POOL_DRTY_PG_STEAL_CLNS	BIGINT	pool_drty_pg_steal_clns - 起動されたバッファー・プール・ビクティム・ページ・クリーナー
POOL_DRTY_PG_THRSH_CLNS	BIGINT	pool_drty_pg_thrsh_clns - 起動されたバッファー・プールしきい値クリーナー
VECTORED_IOS	BIGINT	vectored_ios - ベクトル化入出力要求数
PAGES_FROM_VECTORED_IOS	BIGINT	pages_from_vectored_ios - ベクトル化入出力によって読み取られたページ数の合計
BLOCK_IOS	BIGINT	block_ios - ブロック入出力要求数
PAGES_FROM_BLOCK_IOS	BIGINT	pages_from_block_ios - ブロック入出力によって読み取られたページ数の合計
UNREAD_PREFETCH_PAGES	BIGINT	unread_prefetch_pages - 読み取り不能プリフェッチ・ページ
FILES_CLOSED	BIGINT	files_closed - 閉じられたデータベース・ファイル
ADDITIONAL_DETAILS	BLOB(100K)	将来の利用のために予約済み

MON_GET_CONNECTION 表関数 - 接続メトリックの取得

MON_GET_CONNECTION 表関数は、1 つ以上の接続のメトリックを戻します。

構文

スキーマは SYSPROC です。

表関数パラメーター

application_handle
メトリックが戻される特定のアプリケーション・ハンドル (接続を識別する) を指定する、タイプ BIGINT の入力引数。引数が NULL の場合、すべての接続のメトリックが戻されます。

member
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なメンバーを指定する、タイプ INTEGER の入力引数。現行のデータベース・メンバーには -1、すべてのデータベース・メンバーには -2 を指定します。NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

MON_GET_CONNECTION 関数に対する EXECUTE 特権。

例

戻される行で順序付けられた、最も大きなデータ・ボリュームをクライアントに戻す接続を表示します。

```
SELECT application_handle,
       rows_returned,
       tcpip_send_volume
FROM TABLE(MON_GET_CONNECTION(cast(NULL as bigint), -2)) AS t
ORDER BY rows_returned DESC
```

以下はこの照会の出力例です。

```
APPLICATION_HANDLE   ROWS_RETURNED        TCPIP_SEND_VOLUME
-------------------- -------------------- --------------------
                  55                    6                    0

  1 record(s) selected.
```

使用上の注意

MON_GET_CONNECTION 表関数で戻されるメトリックは、接続によってサブミットされた要求のすべてのメトリックの累計を表します。メトリックは、作業単位境界でロールアップされ、要求の実行中には定期的にロールアップされます。したがって、この表関数で報告される値は、直前の ROLLUP 時のシステムの現行状態を反映しています。メトリックの値は確実に増加します。ある時間間隔に対する指定されたメトリックの値を判別するには、MON_GET_CONNECTION 表関数を使用してその間隔の始めと終わりのメトリックを照会し、差異を計算します。

要求メトリックは、サービス・スーパークラスに対する COLLECT REQUEST METRICS 節、およびデータベース・レベルの *mon_req_metrics* データベース構成パラメーターを介して制御されます。親サービス・スーパークラスで要求メトリックを使用可能にしているサービス・サブクラスのエージェントが要求を処理する場合にのみ、またはデータベース全体で要求メトリック・コレクションが有効な場合にのみ、その要求に対しメトリックが収集されます。デフォルトでは、要求メトリックはデータベース・レベルで使用可能です。要求メトリックがデータベース・レベルでもサービス・スーパークラスに対しても使用不可になっている場合、そのサービス・スーパークラスにマッピングされた接続ごとに報告されるメトリックは増加を停止します (または、要求メトリックがデータベースのアクティブ化時に無効であった場合には 0 のままになります)。

ヒント: 接続は存続時間中に複数のサービス・スーパークラスにマッピングできるので、 MON_GET_CONNECTION 表関数によって報告されるメトリックは、接続によってサブミットされたすべての要求のメトリックのサブセットを表す場合があります。これはメトリックの集合が、接続によってマッピングされるスーパークラスの一部に対して使用不可である場合に生じる可能性があります。

MON_GET_CONNECTION 表関数は、接続ごとおよびメンバーごとに 1 行のデータを戻します。 (1 つ以上のサービス・クラスの) メンバー全体の集約は実行されません。しかし、集約は SQL 照会を使用して実行できます。

戻される情報

表 119. MON_GET_CONNECTION について戻される情報

列名	データ・タイプ	説明
APPLICATION_HANDLE	BIGINT	application_handle - アプリケーション・ハンドル
APPLICATION_NAME	VARCHAR(128)	appl_name - アプリケーション名
APPLICATION_ID	VARCHAR(128)	appl_id - アプリケーション ID
MEMBER	SMALLINT	member - データベース・メンバー
CLIENT_WRKSTNNAME	VARCHAR(255)	CURRENT CLIENT_WRKSTNNAME 特殊レジスター
CLIENT_ACCTNG	VARCHAR(255)	CURRENT CLIENT_ACCTNG 特殊レジスター
CLIENT_USERID	VARCHAR(255)	CURRENT CLIENT_USERID 特殊レジスター
CLIENT_APPLNAME	VARCHAR(255)	CURRENT CLIENT_APPLNAME 特殊レジスター
CLIENT_PID	BIGINT	client_pid - クライアント・プロセス ID
CLIENT_PRDID	VARCHAR(128)	client_prdid - クライアント製品およびバージョン ID
CLIENT_PLATFORM	VARCHAR(12)	client_platform - クライアント・プラットフォーム
CLIENT_PROTOCOL	VARCHAR(10)	client_protocol - クライアント通信プロトコル
SYSTEM_AUTH_ID	VARCHAR(128)	system_auth_id - システム許可 ID
SESSION_AUTH_ID	VARCHAR(128)	session_auth_id - セッション許可 ID
COORD_MEMBER	SMALLINT	coord_member - コーディネーター・メンバー
CONNECTION_START_TIME	TIMESTAMP	connection_start_time - 接続開始時刻
ACT_ABORTED_TOTAL	BIGINT	act_aborted_total - 打ち切られたアクティビティーの合計数
ACT_COMPLETED_TOTAL	BIGINT	act_completed_total - 完了したアクティビティーの合計数
ACT_REJECTED_TOTAL	BIGINT	act_rejected_total - リジェクトされたアクティビティーの合計数
AGENT_WAIT_TIME	BIGINT	agent_wait_time - エージェント待機時間
AGENT_WAITS_TOTAL	BIGINT	agent_waits_total - エージェント待機の合計
POOL_DATA_L_READS	BIGINT	pool_data_l_reads - バッファー・プール・データの論理読み取り
POOL_INDEX_L_READS	BIGINT	pool_index_l_reads - バッファー・プール索引の論理読み取り
POOL_TEMP_DATA_L_READS	BIGINT	pool_temp_data_l_reads - バッファー・プール一時データの論理読み取り
POOL_TEMP_INDEX_L_READS	BIGINT	pool_temp_index_l_reads - バッファー・プール一時索引の論理読み取り
POOL_TEMP_XDA_L_READS	BIGINT	pool_temp_xda_l_reads - バッファー・プール一時 XDA データの論理読み取り
POOL_XDA_L_READS	BIGINT	pool_xda_l_reads - バッファー・プール XDA データの論理読み取り
POOL_DATA_P_READS	BIGINT	pool_data_p_reads - バッファー・プール・データの物理読み取り

表 119. MON_GET_CONNECTION について戻される情報 (続き)

列名	データ・タイプ	説明
POOL_INDEX_P_READS	BIGINT	pool_index_p_reads - バッファー・プール索引の物理読み取り
POOL_TEMP_DATA_P_READS	BIGINT	pool_temp_data_p_reads - バッファー・プール一時データの物理読み取り
POOL_TEMP_INDEX_P_READS	BIGINT	pool_temp_index_p_reads - バッファー・プール一時索引の物理読み取り
POOL_TEMP_XDA_P_READS	BIGINT	pool_temp_xda_p_reads - バッファー・プール一時 XDA データの物理読み取り
POOL_XDA_P_READS	BIGINT	pool_xda_p_reads - バッファー・プール XDA データの物理読み取り
POOL_DATA_WRITES	BIGINT	pool_data_writes - バッファー・プールへのデータの書き込み
POOL_INDEX_WRITES	BIGINT	pool_index_writes - バッファー・プール索引の書き込み
POOL_XDA_WRITES	BIGINT	pool_xda_writes - バッファー・プール XDA データの書き込み
POOL_READ_TIME	BIGINT	pool_read_time - バッファー・プール物理読み取り時間の合計
POOL_WRITE_TIME	BIGINT	pool_write_time - バッファー・プール物理書き込み時間の合計
CLIENT_IDLE_WAIT_TIME	BIGINT	client_idle_wait_time - クライアントのアイドル待機時間
DEADLOCKS	BIGINT	deadlocks - デッドロック検出数
DIRECT_READS	BIGINT	direct_reads - データベースからの直接読み取り
DIRECT_READ_TIME	BIGINT	direct_read_time - 直接読み取り時間
DIRECT_WRITES	BIGINT	direct_writes - データベースへの直接書き込み
DIRECT_WRITE_TIME	BIGINT	direct_write_time - 直接書き込み時間
DIRECT_READ_REQS	BIGINT	direct_read_reqs - 直接読み取り要求
DIRECT_WRITE_REQS	BIGINT	direct_write_reqs - 直接書き込み要求
FCM_RECV_VOLUME	BIGINT	fcm_recv_volume - FCM 受信ボリューム
FCM_RECVS_TOTAL	BIGINT	fcm_recvs_total - FCM 合計受信数
FCM_SEND_VOLUME	BIGINT	fcm_send_volume - FCM 送信ボリューム
FCM_SENDS_TOTAL	BIGINT	fcm_sends_total - FCM 合計送信数
FCM_RECV_WAIT_TIME	BIGINT	fcm_recv_wait_time - FCM 受信待機時間
FCM_SEND_WAIT_TIME	BIGINT	fcm_send_wait_time - FCM 送信待ち時間
IPC_RECV_VOLUME	BIGINT	ipc_recv_volume - プロセス間通信受信ボリューム
IPC_RECV_WAIT_TIME	BIGINT	ipc_recv_wait_time - プロセス間通信受信待ち時間
IPC_RECVS_TOTAL	BIGINT	ipc_recvs_total - プロセス間通信合計受信数
IPC_SEND_VOLUME	BIGINT	ipc_send_volume - プロセス間通信送信ボリューム
IPC_SEND_WAIT_TIME	BIGINT	ipc_send_wait_time - プロセス間通信送信待ち時間
IPC_SENDS_TOTAL	BIGINT	ipc_sends_total - プロセス間通信合計送信数
LOCK_ESCALS	BIGINT	lock_escals - ロック・エスカレーション数

表 119. MON_GET_CONNECTION について戻される情報 (続き)

列名	データ・タイプ	説明
LOCK_TIMEOUTS	BIGINT	lock_timeouts - ロック・タイムアウト数
LOCK_WAIT_TIME	BIGINT	lock_wait_time - ロック待機中の時間
LOCK_WAITS	BIGINT	lock_waits - ロック待機数
LOG_BUFFER_WAIT_TIME	BIGINT	log_buffer_wait_time - ログ・バッファー待ち時間
NUM_LOG_BUFFER_FULL	BIGINT	num_log_buffer_full - フル・ログ・バッファーの回数
LOG_DISK_WAIT_TIME	BIGINT	log_disk_wait_time - ログ・ディスク待機時間
LOG_DISK_WAITS_TOTAL	BIGINT	log_disk_waits_total - ログ・ディスク待機の合計
NUM_LOCKS_HELD	BIGINT	locks_held - ロック保持数
RQSTS_COMPLETED_TOTAL	BIGINT	rqsts_completed_total - 完了した要求の合計数
ROWS_MODIFIED	BIGINT	rows_modified - 変更された行数
ROWS_READ	BIGINT	rows_read - 読み取り行数
ROWS_RETURNED	BIGINT	rows_returned - 戻り行数
TCPIP_RECV_VOLUME	BIGINT	tcpip_recv_volume - TCP/IP 受信ボリューム
TCPIP_SEND_VOLUME	BIGINT	tcpip_send_volume - TCP/IP 送信ボリューム
TCPIP_RECV_WAIT_TIME	BIGINT	tcpip_recv_wait_time - TCP/IP 受信待ち時間
TCPIP_RECVS_TOTAL	BIGINT	tcpip_recvs_total - TCP/IP 合計受信数
TCPIP_SEND_WAIT_TIME	BIGINT	tcpip_send_wait_time - TCP/IP 送信待ち時間
TCPIP_SENDS_TOTAL	BIGINT	tcpip_sends_total - TCP/IP 合計送信数
TOTAL_APP_RQST_TIME	BIGINT	total_app_rqst_time - アプリケーション要求合計時間
TOTAL_RQST_TIME	BIGINT	total_rqst_time - 合計要求時間
WLM_QUEUE_TIME_TOTAL	BIGINT	wlm_queue_time_total - ワークロード・マネージャー合計キュー時間
WLM_QUEUE_ASSIGNMENTS_TOTAL	BIGINT	wlm_queue_assignments_total - ワークロード・マネージャー合計キュー割り当て数
TOTAL_CPU_TIME	BIGINT	total_cpu_time - 合計 CPU 時間
TOTAL_WAIT_TIME	BIGINT	total_wait_time - 合計待ち時間
APP_RQSTS_COMPLETED_TOTAL	BIGINT	app_rqsts_completed_total - 完了したアプリケーション要求の合計数
TOTAL_SECTION_SORT_TIME	BIGINT	total_section_sort_time - セクション・ソート時間合計
TOTAL_SECTION_SORT_PROC_TIME	BIGINT	total_section_sort_proc_time - セクション・ソート処理時間合計
TOTAL_SECTION_SORTS	BIGINT	total_section_sorts - セクション・ソート合計
TOTAL_SORTS	BIGINT	total_sorts - ソート合計
POST_THRESHOLD_SORTS	BIGINT	post_threshold_sorts - ポストしきい値ソート
POST_SHRTHRESHOLD_SORTS	BIGINT	post_shrthreshold_sorts - ポスト共有しきい値ソート
SORT_OVERFLOWS	BIGINT	sort_overflows - ソート・オーバーフロー
TOTAL_COMPILE_TIME	BIGINT	total_compile_time - コンパイル時間の合計
TOTAL_COMPILE_PROC_TIME	BIGINT	total_compile_proc_time - コンパイル処理時間の合計
TOTAL_COMPILATIONS	BIGINT	total_compilations - コンパイルの合計回数

表 119. MON_GET_CONNECTION について戻される情報 (続き)

列名	データ・タイプ	説明
TOTAL_IMPLICIT_COMPILE_TIME	BIGINT	total_implicit_compile_time - 暗黙的コンパイル時間の合計
TOTAL_IMPLICIT_COMPILE _PROC_TIME	BIGINT	total_implicit_compile_proc_time - 暗黙的コンパイルの処理時間の合計
TOTAL_IMPLICIT_COMPILATIONS	BIGINT	total_implicit_compilations - 暗黙的コンパイルの合計回数
TOTAL_SECTION_TIME	BIGINT	total_section_time - セクション時間の合計
TOTAL_SECTION_PROC_TIME	BIGINT	total_section_proc_time - セクション処理時間の合計
TOTAL_APP_SECTION_EXECUTIONS	BIGINT	total_app_section_executions - セクション実行の合計回数
TOTAL_ACT_TIME	BIGINT	total_act_time - 合計アクティビティー時間
TOTAL_ACT_WAIT_TIME	BIGINT	total_act_wait_time - 合計アクティビティー待機時間
ACT_RQSTS_TOTAL	BIGINT	act_rqsts_total - アクティビティー要求の合計
TOTAL_ROUTINE_TIME	BIGINT	total_routine_time - ルーチン時間の合計
TOTAL_ROUTINE_INVOCATIONS	BIGINT	total_routine_invocations - ルーチン呼び出しの合計回数
TOTAL_COMMIT_TIME	BIGINT	total_commit_time - コミット時間の合計
TOTAL_COMMIT_PROC_TIME	BIGINT	total_commit_proc_time - コミット処理時間の合計
TOTAL_APP_COMMITS	BIGINT	total_app_commits - アプリケーション・コミットの合計回数
INT_COMMITS	BIGINT	int_commits - 内部コミット数
TOTAL_ROLLBACK_TIME	BIGINT	total_rollback_time - ロールバック時間の合計
TOTAL_ROLLBACK_PROC_TIME	BIGINT	total_rollback_proc_time - ロールバック処理時間の合計
TOTAL_APP_ROLLBACKS	BIGINT	total_app_rollbacks - アプリケーション・ロールバックの合計回数
INT_ROLLBACKS	BIGINT	int_rollbacks - 内部ロールバック数
TOTAL_RUNSTATS_TIME	BIGINT	total_runstats_time - ランタイム統計時間の合計
TOTAL_RUNSTATS_PROC_TIME	BIGINT	total_runstats_proc_time - ランタイム統計の処理時間の合計
TOTAL_RUNSTATS	BIGINT	total_runstats - ランタイム統計の合計回数
TOTAL_REORG_TIME	BIGINT	total_reorg_time - 再編成時間の合計
TOTAL_REORG_PROC_TIME	BIGINT	total_reorg_proc_time - 再編成の処理時間の合計
TOTAL_REORGS	BIGINT	total_reorgs - 再編成の合計回数
TOTAL_LOAD_TIME	BIGINT	total_load_time - ロード時間の合計
TOTAL_LOAD_PROC_TIME	BIGINT	total_load_proc_time - ロード処理時間の合計
TOTAL_LOADS	BIGINT	total_loads - ロードの合計回数
CAT_CACHE_INSERTS	BIGINT	cat_cache_inserts - カタログ・キャッシュ挿入数
CAT_CACHE_LOOKUPS	BIGINT	cat_cache_lookups - カタログ・キャッシュ参照数
PKG_CACHE_INSERTS	BIGINT	pkg_cache_inserts - パッケージ・キャッシュ挿入
PKG_CACHE_LOOKUPS	BIGINT	pkg_cache_lookups - パッケージ・キャッシュ参照

表 119. MON_GET_CONNECTION について戻される情報 (続き)

列名	データ・タイプ	説明
THRESH_VIOLATIONS	BIGINT	thresh_violations - しきい値違反の回数
NUM_LW_THRESH_EXCEEDED	BIGINT	num_lw_thresh_exceeded - しきい値を超えた回数
ADDITIONAL_DETAILS	BLOB(100K)	将来の利用のために予約済み

MON_GET_CONNECTION_DETAILS 表関数 - 詳細な接続メトリックの取得

MON_GET_CONNECTION_DETAILS 表関数は、1 つ以上の接続の詳細メトリックを戻します。

構文

スキーマは SYSPROC です。

表関数パラメーター

application_handle
メトリックが戻される特定のアプリケーション・ハンドル (接続を識別する) を指定する、タイプ BIGINT の入力引数。引数が NULL の場合、すべての接続についてメトリックが戻されます。

member
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なメンバーを指定する、タイプ INTEGER の入力引数。現行のデータベース・メンバーには -1、すべてのデータベース・メンバーには -2 を指定します。NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

MON_GET_CONNECTION_DETAILS 関数に対する EXECUTE 特権。

例

戻される行で順序付けられた、最も大きなボリュームのデータをクライアントに戻す接続を表示します。

```
SELECT detmetrics.application_handle,
       detmetrics.rows_returned,
       detmetrics.tcpip_send_volume
FROM TABLE(MON_GET_CONNECTION_DETAILS(CAST(NULL as bigint), -2))
AS CONNMETRICS,
XMLTABLE (XMLNAMESPACES( DEFAULT 'http://www.ibm.com/xmlns/prod/db2/mon'),
       '$detmetric/db2_connection' PASSING XMLPARSE(DOCUMENT CONNMETRICS.DETAILS)
       as "detmetric"
COLUMNS "APPLICATION_HANDLE" INTEGER PATH 'application_handle',
       "ROWS_RETURNED" BIGINT PATH 'system_metrics/rows_returned',
       "TCPIP_SEND_VOLUME" BIGINT PATH 'system_metrics/tcpip_send_volume'
       ) AS DETMETRICS
ORDER BY rows_returned DESC
```

以下はこの照会の出力例です。

```
APPLICATION_HANDLE ROWS_RETURNED        TCPIP_SEND_VOLUME
------------------ -------------------- --------------------
                21                    4                    0

  1 record(s) selected.
```

使用上の注意

MON_GET_CONNECTION_DETAILS 表関数で戻されるメトリックは、接続によってサブミットされた要求のすべてのメトリックの累計を表します。この関数は、以下の点で MON_GET_CONNECTION 表関数と似ています。

- MON_GET_CONNECTION 表関数は、最も一般的に使用されるメトリックを列ベースの形式で戻すため、メトリックの取得において最も効率的な方法です。
- MON_GET_CONNECTION_DETAILS 表関数は、使用可能なすべてのメトリック一式を XML 文書形式で戻すため、出力フォーマットを最大限柔軟性のあるものにします。XML ベースの出力は、XML パーサーで直接解析でき、XMLTABLE 関数でリレーショナル形式に変換することもできます (例を参照)。

メトリックは、作業単位境界でロールアップされ、要求の実行中には定期的にロールアップされます。したがって、この表関数で報告される値は、直前の ROLLUP 時のシステムの現行状態を反映しています。メトリックの値は確実に増加します。ある時間間隔に対する指定されたメトリックの値を判別するには、MON_GET_CONNECTION_DETAILS 表関数を使用してその間隔の始めと終わりのメトリックを照会し、差異を計算します。

要求メトリックは、サービス・スーパークラスに対する COLLECT REQUEST METRICS 節、およびデータベース・レベルの **mon_req_metrics** データベース構成パラメーターを介して制御されます。親サービス・スーパークラスで要求メトリックを使用可能にしているサービス・サブクラスのエージェントが要求を処理する場合にのみ、またはデータベース全体で要求メトリック・コレクションが有効な場合にのみ、その要求に対しメトリックが収集されます。デフォルトでは、要求メトリックはデータベース・レベルで使用可能です。要求メトリックがデータベース・レベルでもサービス・スーパークラスに対しても使用不可になっている場合、そのサービス・スーパークラスにマッピングされた接続ごとに報告されるメトリックは増加を停止します (または、要求メトリックがデータベースのアクティブ化時に無効であった場合には 0 のままになります)。

ヒント: 接続はその存続時間中に複数のサービス・スーパークラスにマッピングされる場合があるので、収集がデータベース・レベルで無効になっていると、MON_GET_CONNECTION_DETAILS 表関数によって報告されるメトリックは、接続を介してサブミットされたすべての要求のメトリックのサブセットを表していることがあります。これはメトリックの集合が、接続がマッピングするスーパークラスの一部に対して使用不可である場合に生じる可能性があります。

MON_GET_CONNECTION_DETAILS 表関数は、接続ごとおよびメンバーごとに 1 行のデータを戻します。 (1 つ以上のサービス・クラスの) メンバー全体の集約は実行されません。しかし、集約は SQL 照会を使用して実行できます。

DETAILS 列に戻される XML 文書のスキーマは、ファイル sqllib/misc/DB2MonRoutines.xsd で入手できます。詳細は、ファイル sqllib/misc/DB2MonCommon.xsd 内にあります。

戻される情報

表 120. MON_GET_CONNECTION_DETAILS について戻される情報

列名	データ・タイプ	説明
APPLICATION_HANDLE	BIGINT	application_handle - アプリケーション・ハンドル
MEMBER	SMALLINT	member - データベース・メンバー
DETAILS	BLOB(1M)	作業単位についての詳細メトリックを含む XML 文書。エレメントの説明については、本書の表 121を参照してください。

以下の例は、DETAILS 列で戻される XML 文書の構造を示しています。

```
<db2_connection xmlns="http://www.ibm.com/xmlns/prod/db2/mon" release="90700000">
   <application_handle>21</application_handle>
   <member>0</member>
   <system_metrics release="90700000">
   <act_aborted_total>5</act_aborted_total>
   ...
   <wlm_queue_assignments_total>3</wlm_queue_assignments_total>
   </system_metrics>
</db2_connection>
```

完全スキーマについては、sqllib/misc/DB2MonRoutines.xsd を参照してください。

本書では、以下のような非プリミティブ XML 型定義を使用しています。

```
<xs:simpleType name="db2DbObjectString">
   <xs:restriction base="xs:string">
      <xs:maxLength value="128"/>
   </xs:restriction>
</xs:simpleType>

<xs:simpleType name="db2PartitionNum">
   <xs:restriction base="xs:nonNegativeInteger">
      <xs:maxInclusive value="999"/>
   </xs:restriction>
</xs:simpleType>
```

表 121. MON_GET_CONNECTION_DETAILS について戻される詳細メトリック

エレメント名	データ・タイプ	説明
act_aborted_total	xs:nonNegativeInteger	act_aborted_total - 打ち切られたアクティビティーの合計数
act_completed_total	xs:nonNegativeInteger	act_completed_total - 完了したアクティビティーの合計数
act_rejected_total	xs:nonNegativeInteger	act_rejected_total - リジェクトされたアクティビティーの合計数
act_rqsts_total	xs:nonNegativeInteger	act_rqsts_total - アクティビティー要求の合計
agent_wait_time	xs:nonNegativeInteger	agent_wait_time - エージェント待機時間
agent_waits_total	xs:nonNegativeInteger	agent_waits_total - エージェント待機の合計
application_handle	xs:nonNegativeInteger	application_handle - アプリケーション・ハンドル

表 121. MON_GET_CONNECTION_DETAILS について戻される詳細メトリック (続き)

エレメント名	データ・タイプ	説明
application_id	xs:string	appl_id - アプリケーション ID
application_name	xs:string	appl_name - アプリケーション名
app_rqsts_completed_total	xs:nonNegativeInteger	app_rqsts_completed_total - 完了したアプリケーション要求の合計数
audit_events_total	xs:nonNegativeInteger	audit_events_total - 監査イベント合計数
audit_subsystem_wait_time	xs:nonNegativeInteger	audit_subsystem_wait_time - 監査サブシステム待機時間
audit_subsystem_waits_total	xs:nonNegativeInteger	audit_subsystem_waits_total - 監査サブシステム待機の合計
audit_file_write_wait_time	xs:nonNegativeInteger	audit_file_write_wait_time - 監査ファイル書き込み待機時間
audit_file_writes_total	xs:nonNegativeInteger	audit_file_writes_total - 書き込まれた監査ファイルの合計数
cat_cache_inserts	xs:nonNegativeInteger	cat_cache_inserts - カタログ・キャッシュ挿入数
cat_cache_lookups	xs:nonNegativeInteger	cat_cache_lookups - カタログ・キャッシュ参照数
client_acctng	xs:string (255)	CURRENT CLIENT_ACCTNG 特殊レジスター
client_applname	xs:string (255)	CURRENT CLIENT_APPLNAME 特殊レジスター
client_hostname	xs:string	client_hostname - クライアント・ホスト名
client_idle_wait_time	xs:nonNegativeInteger	client_idle_wait_time - クライアントのアイドル待機時間
client_pid	xs:nonNegativeInteger	client_pid - クライアント・プロセス ID
client_platform	xs:string	client_platform - クライアント・プラットフォーム
client_port_number	xs:nonNegativeInteger	client_port_number - クライアント・ポート番号
client_prdid	xs:string	client_prdid - クライアント製品およびバージョン ID
client_protocol	xs:string	client_protocol - クライアント通信プロトコル
client_userid	xs:string (255)	CURRENT CLIENT_USERID 特殊レジスター
client_wrkstnname	xs:string (255)	CURRENT CLIENT_WRKSTNNAME 特殊レジスター
connection_start_time	xs:dateTime	connection_start_time - 接続開始時刻
coord_member	xs:short	coord_member - コーディネーター・メンバー
deadlocks	xs:nonNegativeInteger	deadlocks - デッドロック検出数
diaglog_writes_total	xs:nonNegativeInteger	diaglog_writes_total - 診断ログ・ファイル書き込みの合計
diaglog_write_wait_time	xs:nonNegativeInteger	diaglog_write_wait_time - 診断ログ・ファイルの書き込み待機時間
direct_read_time	xs:nonNegativeInteger	direct_read_time - 直接読み取り時間
direct_write_time	xs:nonNegativeInteger	direct_write_time - 直接書き込み時間
direct_read_reqs	xs:nonNegativeInteger	direct_read_reqs - 直接読み取り要求
direct_reads	xs:nonNegativeInteger	direct_reads - データベースからの直接読み取り
direct_write_reqs	xs:nonNegativeInteger	direct_write_reqs - 直接書き込み要求
direct_writes	xs:nonNegativeInteger	direct_writes - データベースへの直接書き込み
fcm_recv_volume	xs:nonNegativeInteger	fcm_recv_volume - FCM 受信ボリューム
fcm_recv_wait_time	xs:nonNegativeInteger	fcm_recv_wait_time - FCM 受信待機時間
fcm_recvs_total	xs:nonNegativeInteger	fcm_recvs_total - FCM 合計受信数

表 121. MON_GET_CONNECTION_DETAILS について戻される詳細メトリック (続き)

エレメント名	データ・タイプ	説明
fcm_message_recv_volume	xs:nonNegativeInteger	fcm_message_recv_volume - FCM メッセージ受信ボリューム
fcm_message_recvs_total	xs:nonNegativeInteger	fcm_message_recvs_total - FCM メッセージ合計受信数
fcm_message_recv_wait_time	xs:nonNegativeInteger	fcm_message_recv_wait_time - FCM メッセージの受信待機時間
fcm_message_send_volume	xs:nonNegativeInteger	fcm_message_send_volume - FCM メッセージ送信ボリューム
fcm_message_send_wait_time	xs:nonNegativeInteger	fcm_message_send_wait_time - FCM メッセージ送信待ち時間
fcm_message_sends_total	xs:nonNegativeInteger	fcm_message_sends_total - FCM メッセージ合計送信数
fcm_send_volume	xs:nonNegativeInteger	fcm_send_volume - FCM 送信ボリューム
fcm_send_wait_time	xs:nonNegativeInteger	fcm_send_wait_time - FCM 送信待ち時間
fcm_sends_total	xs:nonNegativeInteger	fcm_sends_total - FCM 合計送信数
fcm_tq_recv_wait_time	xs:nonNegativeInteger	fcm_tq_recv_wait_time - FCM 表キューの受信待ち時間
fcm_tq_send_wait_time	xs:nonNegativeInteger	fcm_tq_send_wait_time - FCM 表キューの送信待ち時間
fcm_tq_recv_volume	xs:nonNegativeInteger	fcm_tq_recv_volume - FCM 表キューの受信ボリューム
fcm_tq_recvs_total	xs:nonNegativeInteger	fcm_tq_recvs_total - FCM 表キューの合計受信数
fcm_tq_send_volume	xs:nonNegativeInteger	fcm_tq_send_volume - FCM 表キューの送信ボリューム
fcm_tq_sends_total	xs:nonNegativeInteger	fcm_tq_sends_total - FCM 表キューの合計送信数
int_commits	xs:nonNegativeInteger	int_commits - 内部コミット数
int_rollbacks	xs:nonNegativeInteger	int_rollbacks - 内部ロールバック数
ipc_recv_volume	xs:nonNegativeInteger	ipc_recv_volume - プロセス間通信受信ボリューム
ipc_recv_wait_time	xs:nonNegativeInteger	ipc_recv_wait_time - プロセス間通信受信待ち時間
ipc_recvs_total	xs:nonNegativeInteger	ipc_recvs_total - プロセス間通信合計受信数
ipc_send_volume	xs:nonNegativeInteger	ipc_send_volume - プロセス間通信送信ボリューム
ipc_send_wait_time	xs:nonNegativeInteger	ipc_send_wait_time - プロセス間通信送信待ち時間
ipc_sends_total	xs:nonNegativeInteger	ipc_sends_total - プロセス間通信合計送信数
last_executable_id	xs:hexBinary(32)	last_executable_id - 最後の実行可能 ID
last_request_type	xs:string(32)	last_request_type - 最後の要求タイプ
lock_escals	xs:nonNegativeInteger	lock_escals - ロック・エスカレーション数
lock_timeouts	xs:nonNegativeInteger	lock_timeouts - ロック・タイムアウト数
lock_wait_time	xs:nonNegativeInteger	lock_wait_time - ロック待機中の時間
lock_waits	xs:nonNegativeInteger	lock_waits - ロック待機数
log_buffer_wait_time	xs:nonNegativeInteger	log_buffer_wait_time - ログ・バッファー待ち時間
log_disk_wait_time	xs:nonNegativeInteger	log_disk_wait_time - ログ・ディスク待機時間
log_disk_waits_total	xs:nonNegativeInteger	log_disk_waits_total - ログ・ディスク待機の合計
member	xs:nonNegativeInteger	member - データベース・メンバー
num_locks_held	xs:nonNegativeInteger	locks_held - ロック保持数
num_log_buffer_full	xs:nonNegativeInteger	num_log_buffer_full - フル・ログ・バッファーの回数
num_lw_thresh_exceeded	xs:nonNegativeInteger	num_lw_thresh_exceeded - しきい値を超えた回数

表 121. MON_GET_CONNECTION_DETAILS について戻される詳細メトリック (続き)

エレメント名	データ・タイプ	説明
pkg_cache_inserts	xs:nonNegativeInteger	pkg_cache_inserts - パッケージ・キャッシュ挿入
pkg_cache_lookups	xs:nonNegativeInteger	pkg_cache_lookups - パッケージ・キャッシュ参照
pool_data_l_reads	xs:nonNegativeInteger	pool_data_l_reads - バッファー・プール・データの論理読み取り
pool_data_p_reads	xs:nonNegativeInteger	pool_data_p_reads - バッファー・プール・データの物理読み取り
pool_data_writes	xs:nonNegativeInteger	pool_data_writes - バッファー・プールへのデータの書き込み
pool_index_l_reads	xs:nonNegativeInteger	pool_index_l_reads - バッファー・プール索引の論理読み取り
pool_index_p_reads	xs:nonNegativeInteger	pool_index_p_reads - バッファー・プール索引の物理読み取り
pool_index_writes	xs:nonNegativeInteger	pool_index_writes - バッファー・プール索引の書き込み
pool_read_time	xs:nonNegativeInteger	pool_read_time - バッファー・プール物理読み取り時間の合計
pool_temp_data_l_reads	xs:nonNegativeInteger	pool_temp_data_l_reads - バッファー・プール一時データの論理読み取り
pool_temp_data_p_reads	xs:nonNegativeInteger	pool_temp_data_p_reads - バッファー・プール一時データの物理読み取り
pool_temp_index_l_reads	xs:nonNegativeInteger	pool_temp_index_l_reads - バッファー・プール一時索引の論理読み取り
pool_temp_index_p_reads	xs:nonNegativeInteger	pool_temp_index_p_reads - バッファー・プール一時索引の物理読み取り
pool_temp_xda_l_reads	xs:nonNegativeInteger	pool_temp_xda_l_reads - バッファー・プール一時 XDA データの論理読み取り
pool_temp_xda_p_reads	xs:nonNegativeInteger	pool_temp_xda_p_reads - バッファー・プール一時 XDA データの物理読み取り
pool_write_time	xs:nonNegativeInteger	pool_write_time - バッファー・プール物理書き込み時間の合計
pool_xda_l_reads	xs:nonNegativeInteger	pool_xda_l_reads - バッファー・プール XDA データの論理読み取り
pool_xda_p_reads	xs:nonNegativeInteger	pool_xda_p_reads - バッファー・プール XDA データの物理読み取り
pool_xda_writes	xs:nonNegativeInteger	pool_xda_writes - バッファー・プール XDA データの書き込み
post_shrthreshold_sorts	xs:nonNegativeInteger	post_shrthreshold_sorts - ポスト共有しきい値ソート
post_threshold_sorts	xs:nonNegativeInteger	post_threshold_sorts - ポストしきい値ソート
rows_modified	xs:nonNegativeInteger	rows_modified - 変更された行数
rows_read	xs:nonNegativeInteger	rows_read - 読み取り行数
rows_returned	xs:nonNegativeInteger	rows_returned - 戻り行数
rqsts_completed_total	xs:nonNegativeInteger	rqsts_completed_total - 完了した要求の合計数
session_auth_id	xs:string	session_auth_id - セッション許可 ID
sort_overflows	xs:nonNegativeInteger	sort_overflows - ソート・オーバーフロー

表 121. MON_GET_CONNECTION_DETAILS について戻される詳細メトリック (続き)

エレメント名	データ・タイプ	説明
system_auth_id	xs:string	system_auth_id - システム許可 ID
tcpip_recv_volume	xs:nonNegativeInteger	tcpip_recv_volume - TCP/IP 受信ボリューム
tcpip_recv_wait_time	xs:nonNegativeInteger	tcpip_recv_wait_time - TCP/IP 受信待ち時間
tcpip_recvs_total	xs:nonNegativeInteger	tcpip_recvs_total - TCP/IP 合計受信数
tcpip_send_volume	xs:nonNegativeInteger	tcpip_send_volume - TCP/IP 送信ボリューム
tcpip_send_wait_time	xs:nonNegativeInteger	tcpip_send_wait_time - TCP/IP 送信待ち時間
tcpip_sends_total	xs:nonNegativeInteger	tcpip_sends_total - TCP/IP 合計送信数
thresh_violations	xs:nonNegativeInteger	num_thresh_violations - しきい値違反の回数
total_act_time	xs:nonNegativeInteger	total_act_time - 合計アクティビティー時間
total_act_wait_time	xs:nonNegativeInteger	total_act_wait_time - 合計アクティビティー待機時間
total_app_commits	xs:nonNegativeInteger	total_app_commits - アプリケーション・コミットの合計回数
total_app_rollbacks	xs:nonNegativeInteger	total_app_rollbacks - アプリケーション・ロールバックの合計回数
total_app_rqst_time	xs:nonNegativeInteger	total_app_rqst_time - アプリケーション要求合計時間
total_app_section_executions	xs:nonNegativeInteger	total_app_section_executions - セクション実行の合計回数
total_commit_proc_time	xs:nonNegativeInteger	total_commit_proc_time - コミット処理時間の合計
total_commit_time	xs:nonNegativeInteger	total_commit_time - コミット時間の合計
total_compilations	xs:nonNegativeInteger	total_compilations - コンパイルの合計回数
total_compile_proc_time	xs:nonNegativeInteger	total_compile_proc_time - コンパイル処理時間の合計
total_compile_time	xs:nonNegativeInteger	total_compile_time - コンパイル時間の合計
total_cpu_time	xs:nonNegativeInteger	total_cpu_time - 合計 CPU 時間
total_implicit_compilations	xs:nonNegativeInteger	total_implicit_compilations - 暗黙的コンパイルの合計回数
total_implicit_compile_ proc_time	xs:nonNegativeInteger	total_implicit_compile_proc_time - 暗黙的コンパイルの処理時間の合計
total_implicit_compile_time	xs:nonNegativeInteger	total_implicit_compile_time - 暗黙的コンパイル時間の合計
total_loads	xs:nonNegativeInteger	total_loads - ロードの合計回数
total_load_proc_time	xs:nonNegativeInteger	total_load_proc_time - ロード処理時間の合計
total_load_time	xs:nonNegativeInteger	total_load_time - ロード時間の合計
total_reorgs	xs:nonNegativeInteger	total_reorgs - 再編成の合計回数
total_reorg_proc_time	xs:nonNegativeInteger	total_reorg_proc_time - 再編成の処理時間の合計
total_reorg_time	xs:nonNegativeInteger	total_reorg_time - 再編成時間の合計
total_rollback_proc_time	xs:nonNegativeInteger	total_rollback_proc_time - ロールバック処理時間の合計
total_rollback_time	xs:nonNegativeInteger	total_rollback_time - ロールバック時間の合計
total_routine_invocations	xs:nonNegativeInteger	total_routine_invocations - ルーチン呼び出しの合計回数
total_routine_user_code _proc_time	xs:nonNegativeInteger	total_routine_user_code_proc_time - ルーチン・ユーザー・コード処理時間の合計
total_routine_user_code_time	xs:nonNegativeInteger	total_routine_user_code_time - ルーチン・ユーザー・コード時間の合計

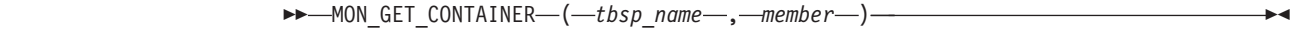
表 121. MON_GET_CONNECTION_DETAILS について戻される詳細メトリック (続き)

エレメント名	データ・タイプ	説明
total_routine_time	xs:nonNegativeInteger	total_routine_time - ルーチン時間の合計
total_rqst_time	xs:nonNegativeInteger	total_rqst_time - 合計要求時間
total_runstats	xs:nonNegativeInteger	total_runstats - ランタイム統計の合計回数
total_runstats_proc_time	xs:nonNegativeInteger	total_runstats_proc_time - ランタイム統計の処理時間の合計
total_runstats_time	xs:nonNegativeInteger	total_runstats_time - ランタイム統計時間の合計
total_section_proc_time	xs:nonNegativeInteger	total_section_proc_time - セクション処理時間の合計
total_section_time	xs:nonNegativeInteger	total_section_time - セクション時間の合計
total_wait_time	xs:nonNegativeInteger	total_wait_time - 合計待ち時間
total_section_sort_time	xs:nonNegativeInteger	total_section_sort_time - セクション・ソート時間合計
total_section_sort_proc_time	xs:nonNegativeInteger	total_section_sort_proc_time - セクション・ソート処理時間合計
total_section_sorts	xs:nonNegativeInteger	total_section_sorts - セクション・ソート合計
total_sorts	xs:nonNegativeInteger	total_sorts - ソート合計
tq_tot_send_spills	xs:nonNegativeInteger	tq_tot_send_spills - オーバーフローした表キュー・バッファーの合計数
wlm_queue_time_total	xs:nonNegativeInteger	wlm_queue_time_total - ワークロード・マネージャー合計キュー時間
wlm_queue_assignments_total	xs:nonNegativeInteger	wlm_queue_assignments_total - ワークロード・マネージャー合計キュー割り当て数

MON_GET_CONTAINER 表関数 - 表スペース・コンテナー・メトリックの取得

MON_GET_CONTAINER 表関数は、1 つ以上の表スペース・コンテナーのモニター・メトリックを戻します。

構文



スキーマは SYSPROC です。

表関数パラメーター

tbsp_name
この関数を呼び出すときに現在接続されているデータベースと同じデータベース内の有効な表スペース名を指定する、タイプ VARCHAR(128) の入力引数。引数が NULL または空ストリングである場合、データベース内のすべての表スペース内のすべてのコンテナーについてメトリックが戻されます。

member
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なメンバーを指定する、タイプ INTEGER の入力引数。現行のデータ

ベース・メンバーには -1、すべてのデータベース・メンバーには -2 を指定します。NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

MON_GET_CONTAINER 関数に対する EXECUTE 特権。

例

例 1: 読み取り時間が最長のすべてのデータベース・メンバーのコンテナーをリストします。

```
SELECT varchar(container_name,70) as container_name,
       varchar(tbsp_name,20) as tbsp_name,
       pool_read_time
FROM TABLE(MON_GET_CONTAINER('',-2)) AS t
ORDER BY pool_read_time DESC
```

以下はこの照会の出力例です。

```
CONTAINER_NAME                                                               ...
---------------------------------------------------------------------------- ...
/home/hotel55/swalkty/swalkty/NODE0000/TEST/T0000000/C0000000.CAT      ...
/home/hotel55/swalkty/swalkty/NODE0000/TEST/T0000002/C0000000.LRG      ...
/home/hotel55/swalkty/swalkty/NODE0000/TEST/T0000001/C0000000.TMP      ...

  3 record(s) selected.
```

照会の出力 (続き)。

```
... TBSP_NAME            POOL_READ_TIME
... -------------------- --------------------
... SYSCATSPACE                         597
... USERSPACE1                           42
... TEMPSPACE1                            0
```

例 2: アクセスできないすべてのコンテナーをリストします。

```
SELECT varchar(container_name, 70) as container_name
FROM TABLE(MON_GET_CONTAINER('',-1)) AS t
WHERE accessible = 0
```

以下はこの照会の出力例です。

```
CONTAINER_NAME
------------------------------------------------------------------------

  0 record(s) selected.
```

例 3: 使用率の高い順にコンテナー・ファイル・システムの使用率をリストします。

```
SELECT varchar(container_name, 65) as container_name,
       fs_id,
       fs_used_size,
       fs_total_size,
       CASE WHEN fs_total_size > 0
            THEN DEC(100*(FLOAT(fs_used_size)/FLOAT(fs_total_size)),5,2)
            ELSE DEC(-1,5,2)
       END as utilization
FROM TABLE(MON_GET_CONTAINER('',-1)) AS t
ORDER BY utilization DESC
```

以下はこの照会の出力例です。

```
CONTAINER_NAME                                                          ...
----------------------------------------------------------------------- ...
/home/hotel55/swalkty/swalkty/NODE0000/TEST/T0000000/C0000000.CAT ...
/home/hotel55/swalkty/swalkty/NODE0000/TEST/T0000001/C0000000.TMP ...
/home/hotel55/swalkty/swalkty/NODE0000/TEST/T0000002/C0000000.LRG ...

  3 record(s) selected.
```

照会の出力 (続き)。

```
FS_ID                FS_USED_SIZE         FS_TOTAL_SIZE        UTILIZATION
-------------------- -------------------- -------------------- -----------
               64768         106879311872         317068410880       33.70
               64768         106879311872         317068410880       33.70
               64768         106879311872         317068410880       33.70
```

使用上の注意

MON_GET_CONTAINER 表関数は、コンテナーごとおよびデータベース・メンバーごとに 1 行のデータを戻します。データは、指定の表スペース内のすべてのコンテナー、またはデータベース内のすべてのコンテナーについて戻すことができます。データベース・パーティション全体からの集約は実行されません。しかし、集約は SQL 照会を使用して実行できます。

この関数によって収集されるメトリックは、mon_obj_metrics 構成パラメーターを使用してデータベース・レベルで制御されます。デフォルトでは、メトリック収集は有効になります。

戻される情報

表 122. MON_GET_CONTAINER について戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
TBSP_NAME	VARCHAR(128)	tablespace_name - 表スペース名
TBSP_ID	BIGINT	tablespace_id - 表スペース ID
CONTAINER_NAME	VARCHAR(256)	container_name - コンテナー名
CONTAINER_ID	BIGINT	container_id - コンテナー ID
MEMBER	SMALLINT	member - データベース・メンバー
CONTAINER_TYPE	VARCHAR(16)	container_type - コンテナー・タイプ。これは、sqlutil.h での定義を基にしたテキスト ID です。以下のいずれかとなります。 • DISK_EXTENT_TAG • DISK_PAGE_TAG • FILE_EXTENT_TAG • FILE_PAGE_TAG • PATH
STRIPE_SET	BIGINT	container_stripe_set - ストライプ・セット
DIRECT_READS	BIGINT	direct_reads - データベースからの直接読み取り
DIRECT_WRITES	BIGINT	direct_writes - データベースへの直接書き込み
DIRECT_READ_TIME	BIGINT	direct_read_time - 直接読み取り時間
DIRECT_WRITE_TIME	BIGINT	direct_write_time - 直接書き込み時間
PAGES_READ	BIGINT	pages_read - 読み取られたページの数

表 122. MON_GET_CONTAINER について戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
PAGES_WRITTEN	BIGINT	pages_written - 書き込まれたページの数
VECTORED_IOS	BIGINT	vectored_ios - ベクトル化入出力要求数
PAGES_FROM_VECTORED_IOS	BIGINT	pages_from_vectored_ios - ベクトル化入出力によって読み取られたページ数の合計
BLOCK_IOS	BIGINT	block_ios - ブロック入出力要求数
PAGES_FROM_BLOCK_IOS	BIGINT	pages_from_block_ios - ブロック入出力によって読み取られたページ数の合計
POOL_READ_TIME	BIGINT	pool_read_time - バッファー・プール物理読み取り時間の合計
POOL_WRITE_TIME	BIGINT	pool_write_time - バッファー・プール物理書き込み時間の合計
TOTAL_PAGES	BIGINT	container_total_pages - コンテナー内の合計ページ数
USABLE_PAGES	BIGINT	container_usable_pages - コンテナー内の使用可能なページ数
ACCESSIBLE	SMALLINT	container_accessible - コンテナーのアクセス可能性
FS_ID	VARCHAR(22)	fs_id - 固有のファイル・システム識別番号
FS_TOTAL_SIZE	BIGINT	fs_total_size - ファイル・システムの合計サイズ
FS_USED_SIZE	BIGINT	fs_used_size - ファイル・システム上で使用されるスペースの量
ADDITIONAL_DETAILS	BLOB(100K)	将来の利用のために予約済み。

MON_GET_EXTENT_MOVEMENT_STATUS - エクステント移動進行状況の取得

MON_GET_EXTENT_MOVEMENT_STATUS 表関数は、エクステント移動操作の状況を戻します。

構文

スキーマは SYSPROC です。

表関数パラメーター

tbsp_name
照会する表スペースを指定する、タイプ VARCHAR(128) の入力引数。引数値が NULL の場合、この関数はすべての表スペースについて情報を戻します。

member
現在接続されているデータベースと同じインスタンス内の有効なメンバーを指定する、タイプ INTEGER の入力引数。現行のデータベース・メンバーには -1、すべてのデータベース・メンバーには -2 を指定します。引数値が NULL の場合は、-1 が暗黙的に設定されます。

許可

MON_GET_EXTENT_MOVEMENT_STATUS 関数に対する EXECUTE 特権。

例

すべての表スペースについて、現在のエクステント進行状況に関するすべての情報を取得します。

```
SELECT * FROM TABLE(SYSPROC.MON_GET_EXTENT_MOVEMENT_STATUS('', -1))
```

以下は、この照会の出力例です。

```
TBSP_NAME   TBSP_ID MEMBER CURRENT_EXTENT LAST_EXTENT NUM_EXTENTS_MOVED
---------   ------- ------ -------------- ----------- -----------------
SYSCATSPACE       0      0             -1          -1                -1
TEMPSPACE1        1      0             -1          -1                -1
USERSPACE1        2      0             -1          -1                -1
TS1               3      0              1           2                 3
SYSTOOLSPACE      4      0             -1          -1                -1

5 record(s) selected.
```

この照会からの出力 (続き):

```
... NUM_EXTENTS_LEFT TOTAL_MOVE_TIME ADDITIONAL_DETAILS
... ---------------- --------------- ------------------
...                -1              -1                  -
...                -1              -1                  -
...                -1              -1                  -
...                 4               0                  -
...                -1              -1                  -
```

戻される情報

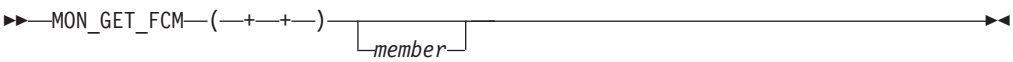
表 123. MON_GET_EXTENT_MOVEMENT_STATUS について戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
TBSP_NAME	VARCHAR(128)	tablespace_name - 表スペース名
TBSP_ID	BIGINT	tablespace_id - 表スペース ID
MEMBER	SMALLINT	member - この情報が収集されたメンバー
CURRENT_EXTENT	INTEGER	current_extent - 移動中の現在のエクステント
LAST_EXTENT	INTEGER	last_extent - 最後に移動されたエクステント
NUM_EXTENTS_MOVED	INTEGER	num_extents_moved - このエクステント移動操作中にこれまで移動したエクステントの数
NUM_EXTENTS_LEFT	INTEGER	num_extents_left - このエクステント移動操作中にまだ移動されていないエクステントの数
TOTAL_MOVE_TIME	BIGINT	total_move_time - 移動されたすべてのエクステントの合計移動時間 (ミリ秒単位)
ADDITIONAL_DETAILS	BLOB(100K)	将来の利用のために予約済み

MON_GET_FCM - FCM メトリックの取得

MON_GET_FCM 表関数は、高速コミュニケーション・マネージャー (FCM) に関するメトリックを戻します。

構文



スキーマは SYSPROC です。

表関数パラメーター

member
有効なデータベース・メンバー番号を指定する、タイプ INTEGER の入力引数。現行データベース・メンバーの場合は -1、すべてのアクティブ・データベース・メンバーからの情報の場合は -2 を指定します。アクティブ・データベース・メンバーとは、アプリケーションがデータベースに接続して使用できるところのことです。

許可

MON_GET_FCM 表関数に対する EXECUTE 特権。

戻される情報

表 124. MON_GET_FCM について戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
HOSTNAME	VARCHAR(128)	hostname - ホスト名
MEMBER	SMALLINT	member - データベース・メンバー
BUFF_MAX	BIGINT	buff_max - FCM バッファーの最大可能数
BUFF_TOTAL	BIGINT	buff_total - 現在割り振られている FCM バッファー数
BUFF_FREE	BIGINT	buff_free - 現在空いている FCM バッファー
BUFF_FREE_BOTTOM	BIGINT	buff_free_bottom - 空き FCM バッファーの最小数
BUFF_AUTO_TUNING	SMALLINT	buff_auto_tuning - FCM バッファー自動チューニング標識
CH_MAX	BIGINT	ch_max - FCM チャネルの最大可能数
CH_TOTAL	BIGINT	ch_total - 現在割り振られている FCM チャネル数
CH_FREE	BIGINT	ch_free - 現在空いているチャネル
CH_FREE_BOTTOM	BIGINT	ch_free_bottom - 空いているチャネルの最小
CH_AUTO_TUNING	SMALLINT	ch_auto_tuning - FCM チャネル自動チューニング標識
ADDITIONAL_DETAILS	BLOB(100K)	将来の利用のために予約済み。

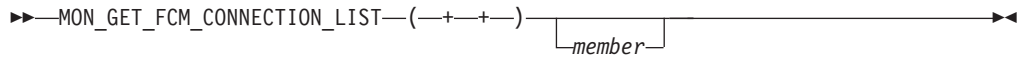
注: この表関数が提供するメトリックは、ある特定のホスト・マシン上のすべてのメンバーに適用されます。ある特定のホスト・マシン上のすべてのメンバーが、同

一セットのバッファーとチャネルを共有します。これは、ある特定のホスト・マシン上の各メンバーで個々のメトリックが同じになることを意味します。

MON_GET_FCM_CONNECTION_LIST - すべての FCM 接続の詳細の取得

MON_GET_FCM_CONNECTION_LIST 表関数は、指定されたメンバーのすべての高速コミュニケーション・マネージャー (FCM) 接続に関するモニター・メトリックを戻します。

構文



スキーマは SYSPROC です。

表関数パラメーター

member
有効なデータベース・メンバー番号を指定する、タイプ INTEGER の入力引数。現行データベース・メンバーの場合は -1、すべてのアクティブ・データベース・メンバーからの情報の場合は -2 を指定します。アクティブ・データベース・メンバーとは、アプリケーションがデータベースに接続して使用できるところのことです。

許可

MON_GET_FCM_CONNECTION_LIST 表関数に対する EXECUTE 特権。

戻される情報

表 125. MON_GET_FCM_CONNECTION_LIST について戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
MEMBER	SMALLINT	member - データベース・メンバー
REMOTE_MEMBER	SMALLINT	remote_member - リモート・メンバー
CONNECTION_STATUS	VARCHAR(16)	connection_status - 接続状況。
TOTAL_BUFFERS_SENT	BIGINT	total_buffers_sent - 送信された FCM バッファーの合計
TOTAL_BUFFERS_RCVD	BIGINT	total_buffers_rcvd - 受信された FCM バッファーの合計
FCM_CONGESTION_TIME	BIGINT	将来の利用のために予約済み。
FCM_CONGESTED_SENDS	BIGINT	将来の利用のために予約済み。
FCM_NUM_CONGESTION _TIMEOUTS	BIGINT	将来の利用のために予約済み。
FCM_SEND_VOLUME	BIGINT	将来の利用のために予約済み。
FCM_RECV_VOLUME	BIGINT	将来の利用のために予約済み。
FCM_MESSAGE_SEND_VOLUME	BIGINT	将来の利用のために予約済み。
FCM_MESSAGE_RECV_VOLUME	BIGINT	将来の利用のために予約済み。
FCM_TQ_SEND_VOLUME	BIGINT	将来の利用のために予約済み。
FCM_TQ_RECV_VOLUME	BIGINT	将来の利用のために予約済み。

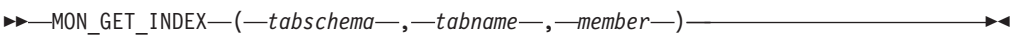
表 125. *MON_GET_FCM_CONNECTION_LIST* について戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
FCM_NUM_CONN_LOST	BIGINT	将来の利用のために予約済み。
FCM_NUM_CONN_TIMEOUTS	BIGINT	将来の利用のために予約済み。
ADDITIONAL_DETAILS	BLOB(100K)	将来の利用のために予約済み。

MON_GET_INDEX 表関数 - 索引メトリックの取得

MON_GET_INDEX 表関数は、1 つ以上の索引のメトリックを戻します。

構文



スキーマは SYSPROC です。

表関数パラメーター

tabschema
この関数を呼び出すときに現在接続されているデータベースと同じデータベース内の有効な表スキーマ名を指定する、タイプ VARCHAR(128) の入力引数。引数が NULL または空ストリングである場合、データベース内の全スキーマの表の索引についてメトリックが取得されます。引数が指定されている場合、指定したスキーマ内の表の索引についてのメトリックのみが戻されます。

tabname
この関数を呼び出すときに現在接続されているデータベースと同じデータベース内の有効な表名を指定する、タイプ VARCHAR(128) の入力引数。指定される表のすべての索引についてのメトリックが戻されます。引数が NULL または空ストリングである場合、データベース内のすべての表のすべての索引についてメトリックが取得されます。

member
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なメンバーを指定する、タイプ INTEGER の入力引数。現行のデータベース・メンバーには -1、すべてのデータベース・メンバーには -2 を指定します。NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

MON_GET_INDEX 関数に対する EXECUTE 特権。

例

最後にデータベースがアクティブ化されてから、DMEXT002.TABLE1 表の索引で最も頻繁に使用された索引を識別します。

```
SELECT VARCHAR(S.INDSCHEMA, 10) AS INDSCHEMA,
  VARCHAR(S.INDNAME, 10) AS INDNAME,
  T.DATA_PARTITION_ID,
  T.MEMBER,
  T.INDEX_SCANS,
```

```
  T.INDEX_ONLY_SCANS
FROM TABLE(MON_GET_INDEX('DMEXT002','TABLE1', -2)) as T, SYSCAT.INDEXES AS S
WHERE T.TABSCHEMA = S.TABSCHEMA AND
  T.TABNAME = S.TABNAME AND
  T.IID = S.IID
ORDER BY INDEX_SCANS DESC
```

以下はこの照会の出力例です。

```
INDSCHEMA  INDNAME    DATA_PARTITION_ID MEMBER         INDEX_SCANS          INDEX_ONLY_SCANS
---------- ---------- ----------------- -------------- -------------------- --------------------
DMEXT002   INDEX3                    -              0                    1                    1
DMEXT002   INDEX4                    -              0                    1                    0
DMEXT002   INDEX1                    -              0                    0                    0
DMEXT002   INDEX2                    -              0                    0                    0
DMEXT002   INDEX5                    -              0                    0                    0
DMEXT002   INDEX6                    -              0                    0                    0

  6 record(s) selected.
```

使用上の注意

MON_GET_INDEX 表関数は、索引およびデータベース・メンバーごとに 1 行のデータを戻します。パーティション化索引が使用されている場合、データベース・メンバーの索引パーティションごとに 1 つの行が戻されます。データベース・メンバー全体からの集約は実行されません。しかし、集約は上の例で示された SQL 照会を使用して実行できます。

データベースがアクティブにされた後にアクセスのあった表の索引についてのみ、メトリックが戻されます。すべてのカウンターは、現在のデータベースがアクティブにされた後のデータを表します。例えば、*pseudo_empty_pages* カウンターは、データベースがアクティブにされた後に疑似空白として識別されたページ数です。これは、索引内の現在の疑似空ページ数ではありません。

メトリックは常に使用可能です。この関数を介して表メトリックにアクセスする場合に、システム・モニター・スイッチを必ずしもオンにする必要はありません。

戻される情報

表 126. MON_GET_INDEX について戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
TABSCHEMA	VARCHAR(128)	table_schema - 表スキーマ名
TABNAME	VARCHAR(128)	table_name - 表名
IID	SMALLINT	iid - 索引 ID
MEMBER	SMALLINT	member - データベース・メンバー
DATA_PARTITION_ID	INTEGER	data_partition_id - データ・パーティション ID。索引がパーティション化されていない場合は、NULL が戻されます。
NLEAF	BIGINT	nleaf - リーフ・ページ数
NLEVELS	SMALLINT	nlevels - 索引レベルの数
INDEX_SCANS	BIGINT	index_scans - 索引スキャン
INDEX_ONLY_SCANS	BIGINT	index_only_scans - 索引のみのスキャン
KEY_UPDATES	BIGINT	key_updates - キー更新
INCLUDE_COL_UPDATES	BIGINT	include_col_updates - 列更新の組み込み

表 126. MON_GET_INDEX について戻される情報 (続き)

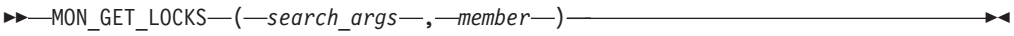
列名	データ・タイプ	説明または対応するモニター・エレメント
PSEUDO_DELETES	BIGINT	pseudo_deletes - 疑似削除
DEL_KEYS_CLEANED	BIGINT	del_keys_cleaned - 疑似削除されたキーのクリーン
ROOT_NODE_SPLITS	BIGINT	root_node_splits - ルート・ノード分割
INT_NODE_SPLITS	BIGINT	int_node_splits - 中間ノード分割
BOUNDARY_LEAF _NODE_SPLITS	BIGINT	boundary_leaf_node_splits - 境界リーフ・ノード分割
NONBOUNDARY_LEAF _NODE_SPLITS	BIGINT	nonboundary_leaf_node_splits - 非境界リーフ・ノード分割
PAGE_ALLOCATIONS	BIGINT	page_allocations - ページ割り振り
PSEUDO_EMPTY_PAGES	BIGINT	pseudo_empty_pages - 疑似空ページ
EMPTY_PAGES_REUSED	BIGINT	empty_pages_reused - 再利用された空ページ
EMPTY_PAGES_DELETED	BIGINT	empty_pages_deleted - 削除された空ページ
PAGES_MERGED	BIGINT	pages_merged - マージされたページ
ADDITIONAL_DETAILS	BLOB(100K)	将来の利用のために予約済み。

MON_GET_LOCKS - 現在接続されているデータベース内のすべてのロックのリスト

MON_GET_LOCKS 表関数は、現在接続されているデータベース内のすべてのロックのリストを戻します。

注: バージョン 9.7 のフィックスパック 1 以前で作成されたデータベースを使用している場合、このルーチンを実行するには、既に db2updv97 コマンドを実行済みでなければなりません。バージョン 9.7 以前に作成されたデータベースを使用している場合は、db2updv97 コマンドを実行する必要はありません (データベースのマイグレーションによって自動的にカタログ更新が実行されるため)。バージョン 9.7 にダウングレードすると、このルーチンは機能しなくなります。

ロックについての情報を取得するには MON_GET_LOCKS、MON_FORMAT_LOCK_NAME、MON_GET_APPL_LOCKWAIT 表関数と MON_LOCKWAIT 管理ビューを使用してください。SNAPLOCKWAIT 管理ビューと SNAP_GET_LOCKWAIT 表関数、SNAPLOCK 管理ビューと SNAP_GET_LOCK 表関数、および LOCKS_HELD 管理ビューはバージョン 9.7 フィックスパック 1 で非推奨になりました。



スキーマは SYSPROC です。

表関数パラメーター

search_args

key-value の組のリストを表すタイプ CLOB(1K) の入力パラメーター。リストが空または NULL の場合には、現在接続されているデータベース内のすべてのロ

ックが戻されます。それ以外の場合は、*key-value* の組のリストによって表されるすべての条件に一致するすべてのロックが戻されます。 *key-value* の組は以下の形式に従う必要があります。

- *key* は、開始タグ、値、終了タグの順序で構成されるストリングです。
- 開始タグは、タグの始まりを示す不等号括弧、キー名、タグの終わりを示す不等号括弧の順序で構成されます。スペースは使用できません。
- 終了タグは、タグの始まりを示す不等号括弧、スラッシュ、キー名、タグの終わりを示す不等号括弧の順序で構成されます。スペースは使用できません。
- すべてのキーは大文字と小文字を区別し、*search_args* パラメーター内で 1 度だけ指定できます。
- 複数のキーの順序は重要ではありません。

無効な *key-value* の組に対しては SQLCODE -171 が戻されます。

表が存在しない場合には SQLCODE -204 が戻されます。

複数の異なるキーの間で AND 演算が実行されます。同じキーの複数の値の間では OR 演算が実行されます。例えば以下のように *search_args* パラメーターを使用した場合に戻されるリストには、共有モードまたは排他モードでハンドル 123 のアプリケーションによって保持されている (または取得待機中の) 表タイプまたは行タイプのすべてのロックが含まれます。

```
CLOB('<application_handle>123</application_handle>
      <lock_object_type>Table:Row</lock_object_type>
      <lock_mode>S:X</lock_mode>')
```

MON_GET_LOCKS 表関数で使用できるキーは次のとおりです。

- application_handle

  指定されたアプリケーション・ハンドルによって現在保持されている、または取得されようとしているすべてのロックのリストを戻します。キー値の単一オカレンスのみ指定できます。値は INTEGER として指定します。例えば、

  ```
  CLOB('<application_handle>145</application_handle>'))
  ```

- lock_name

  指定されたロック名に一致するすべてのロックのリストを戻します。キー値の単一オカレンスのみ指定できます。値は、最大長 32 のストリングとして指定します。例えば、

  ```
  CLOB('<lock_name>000300050000000000280000452</lock_name>')
  ```

- lock_object_type

  指定されたロック対象タイプに一致するすべてのロックのリストを戻します。キー値の複数のオカレンスを指定できます (最大で 5 回まで)。それぞれの値 (大/小文字を区別しない) をコロン (:) で区切る必要があります。値は最大長 32 文字のストリングとして指定します。例えば、

  ```
  CLOB('<lock_object_type>Table:Chunk:Plan</lock_object_type>')
  ```

  可能な入力値のリストについては、『lock_object_type - 待機中のロック対象タイプ』モニター・エレメントを参照してください。

- lock_mode

  指定されたロック・モードに一致するすべてのロックのリストを戻します。キー値の複数のオカレンスを指定できます (最大で 5 回まで)。それぞれの値 (大/小文字を区別しない) をコロン (:) で区切ります。値は最大長 3 のストリングとして指定します。例えば、

  ```
  CLOB('<lock_mode>IS:IN:U</lock_mode>')
  ```

  可能な入力値のリストについては、『lock_mode - ロック・モード・モニター・エレメント』を参照してください。

- lock_status

  指定された状況にあるすべてのロックのリストを戻します。キー値の単一オカレンスのみ指定できます。値は文字として指定します。

  ```
  CLOB('<lock_status>W</lock_status>')
  ```

  可能な入力値のリストについては、『lock_status - ロック状況モニター・エレメント』を参照してください。

- table_schema

  指定されたスキーマ名によって限定されるすべてのロックのリストを戻します。さらに table_name キーも指定する必要があります。キー値の単一オカレンスのみ指定できます。値は、最大長 128 のストリングとして指定します。

- table_name

  指定された表を参照するすべてのロックのリストを戻します。さらに table_schema キーも指定する必要があります。キー値の単一オカレンスのみ指定できます。値は、最大長 128 のストリングとして指定します。例えば、

  ```
  CLOB('<table_schema>USER1</table_schema>
        <table_name>INVENTORY</table_name>')
  ```

以下の例は、*key-value* の組を *search_args* パラメーターで使用する方法を示しています。

1. すべての ROW および TABLE ロックを検索するには、次のようにします。

   ```
   CLOB('<lock_object_type>Table:Row</lock_object_type>')
   ```

2. 表 T1 を参照し、ユーザー USER1 によって作成された、アプリケーション・ハンドル 123 が保持している (または取得待機中の) すべてのロックを検索するには、次のようにします。

   ```
   CLOB('<application_handle>123</application_handle>
         <table_schema>USER1</table_schema>
         <table_name>T1</table_name>')
   ```

3. 共有モードで現在保持されているすべての TABLE、ROW、および BUFFERPOOL ロックを検索するには、次のようにします。

   ```
   CLOB('<lock_mode>S</lock_mode>
         <lock_status>G</lock_status>
         <lock_object_type>Table:Row:Reorg</lock_object_type>')
   ```

member
どのメンバーのデータが戻されるかを指定する、タイプ INTEGER の入力引数。現在のメンバーの場合は -1、すべてのアクティブ・メンバーの場合は -2 を指定します。

許可

以下のいずれかの権限または特権が必要です。

- SYSADM 権限
- SYSMON 権限

例

この例のシナリオでは表関数 MON_GET_LOCKS および MON_GET_APPL_LOCKWAIT を使用して、現在接続されているデータベース内の、すべてのメンバーに関するロック状態を調べます。

1. 次のように MON_GET_APPL_LOCKWAIT 表関数を呼び出して、現在接続されているデータベース内の、すべてのメンバーに関する取得待機中のすべてのロックを判別します。

```
SELECT lock_name,
       hld_member,
       lock_status,
       hld_application_handle FROM
       TABLE (MON_GET_APPL_LOCKWAIT(NULL, -2))
```

この照会は、以下の出力を戻します。

```
LOCK_NAME                  HLD_MEMBER LOCK_STATUS HLD_APPLICATION_HANDLE
-------------------------- ---------- ----------- ----------------------
00030005000000000280000452 -2           W
00030005000000000280000452 -2           W
00030005000000000280000452 -2           W

  3 record(s) selected.
```

HLD_MEMBER が -2 であることを示すレコードは、ロック 0x00030005000000000280000452 がリモート・メンバーで保持されていることを示しています。

2. 次のように MON_GET_LOCKS 表関数を呼び出してロックのホルダーを判別します。その際、検索引数としてロック名 0x00030005000000000280000452 を指定します。

```
SELECT lock_name,
       member,
       lock_status,
       application_handle FROM
    TABLE (MON_GET_LOCKS(
      CLOB('<lock_name>00030005000000000280000452</lock_name>),
        -2))
```

この照会は、以下の出力を戻します。

```
LOCK_NAME                  MEMBER LOCK_STATUS APPLICATION_HANDLE
-------------------------- ------ ----------- ------------------
00030005000000000280000452 0      W             12562
00030005000000000280000452 1      W             12562
```

```
00030005000000000280000452 2      G              65545
00030005000000000280000452 3      W              12562

  4 record(s) selected.
```

ロックを保持しているアプリケーションについてさらに詳しく調べるために、WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES_V97 または WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES_V97 表関数を呼び出すことができます。

戻される情報

表 127. MON_GET_LOCKS 表関数によって戻される情報

列名	データ・タイプ	説明またはモニター・エレメント
APPLICATION_HANDLE	BIGINT	application_handle - アプリケーション・ハンドル LOCK_STATUS 列が G である場合、これはロックを現在保持しているアプリケーションを表します。 LOCK_STATUS 列が W または C である場合、これはロックの取得を現在待機しているアプリケーションを表します。
MEMBER	SMALLINT	この行のデータが取得されたデータベース・メンバー。
LOCK_NAME	VARCHAR(32)	lock_name - ロック名
LOCK_OBJECT_TYPE	VARCHAR(32)	lock_object_type - ロック対象タイプ LOCK_STATUS 列が G である場合、これはアプリケーションが現在保持しているオブジェクトのタイプを表します。 LOCK_STATUS 列が W または C である場合、これはアプリケーションが取得を現在待機しているオブジェクトのタイプを表します。 可能な入力値については、『lock_object_type - 待機中のロック対象タイプ』モニター・エレメントを参照してください。
LOCK_OBJECT_TYPE_ID	CHAR(1) FOR BIT DATA	内部使用のために予約済み

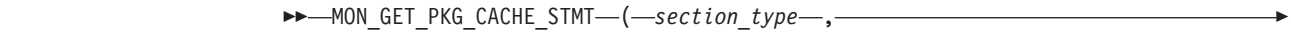
表 127. MON_GET_LOCKS 表関数によって戻される情報 (続き)

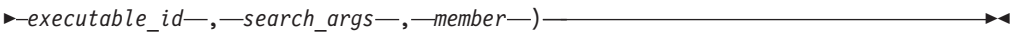
列名	データ・タイプ	説明またはモニター・エレメント
LOCK_MODE	VARCHAR(3)	lock_mode - ロック・モード LOCK_STATUS 列が G である場合、これはアプリケーションがロックを現在保持しているモードを表します。 LOCK_STATUS 列が W または C である場合、これはアプリケーションがロックの取得を現在待機しているモードを表します。 モードが不明な場合、NULL 値がこの列に戻されます。
LOCK_CURRENT_MODE	VARCHAR(3)	lock_current_mode - 移行前の元のロック・モード モードが不明な場合、NULL 値がこの列に戻されます。
LOCK_STATUS	CHAR(1)	lock_status - ロック状況
LOCK_ATTRIBUTES	CHAR(16)	lock_attributes - ロック属性
LOCK_RELEASE_FLAGS	CHAR(16)	内部使用のために予約済み
LOCK_RRIID	BIGINT	内部使用のために予約済み
LOCK_COUNT	BIGINT	内部使用のために予約済み
LOCK_HOLD_COUNT	BIGINT	内部使用のために予約済み
TBSP_ID	BIGINT	tablespace_id - 表スペース ID 表スペースを参照しないロックの場合、NULL 値が戻されます。
TAB_FILE_ID	BIGINT	table_file_id - 表ファイル ID
ADDITIONAL_DETAILS	BLOB(100K)	内部使用のために予約済み

MON_GET_PKG_CACHE_STMT 表関数 - パッケージ・キャッシュ内の SQL ステートメント・アクティビティー・メトリックの取得

MON_GET_PKG_CACHE_STMT 表関数は、データベース・パッケージ・キャッシュ内の静的 SQL ステートメントと動的 SQL ステートメントの両方のポイント・イン・タイム・ビューを戻します。

構文





スキーマは SYSPROC です。

表関数パラメーター

section_type
オプションの入力引数 (D または S)、タイプは CHAR(1)。戻されるステートメントの情報の種類を指定します。引数が NULL または空ストリングである場合、すべての SQL ステートメントについての情報が戻されます。大/小文字は区別されません。D は動的を表し、S は静的を表します。

executable_id
オプションの入力引数、タイプは VARCHAR(32)。データベース・パッケージ・キャッシュの固有のセクションを指定するビット・データを表します。NULL 値を指定すると、すべての SQL ステートメントについての情報が戻されます。*executable_id* が指定されている場合、*section_type* 引数は無視されることに注意してください。例えば、動的ステートメントで *executable_id* が指定されている場合、*section_type* が静的 ("S") として指定されていても、この表関数によって動的ステートメントの詳細が戻されます。

search_args
オプションのタイプ CLOB(1K) の入力パラメーター。これを使用して 1 つ以上の検索引数ストリングをオプションで指定できます。例えば、

```
'<modified_within>5</modified_within><update_boundary_time>myPkgEvmon
    </update_boundary_time>'
```

使用可能な検索引数タグは次のとおりです。

- '<modified_within>X</modified_within>'

  現在までの *X* 分間にキャッシュに挿入された、または実行されたステートメント項目だけを戻します (*X* は正の整数値)。引数が指定されない場合、キャッシュ内のすべての項目が戻されます。

- '<update_boundary_time>evmon_name</update_boundary_time>'

  evmon_name で指定されたパッケージ・キャッシュ・イベント・モニターに関して、イベント・モニター境界タイム・スタンプを現在時刻に更新します。このイベント・モニターの WHERE 節で出力基準として `where updated_since_boundary_time` が指定される場合、今後にメトリックが更新されるパッケージ・キャッシュ項目だけが、パッケージ・キャッシュからの退去時にキャプチャーされます。指定されたパッケージ・キャッシュ・イベント・モニターがコマンド発行時にアクティブである場合にのみ、この操作が効果を及ぼします。

それぞれの入力引数は一度だけ指定できます。検索引数タグは小文字で指定する必要があります。

member
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なメンバーを指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・メンバーには -1、すべてのデータベース・メンバーには -2 を指定します。NULL 値を指定すると、-1 が設定されます。

許可

MON_GET_PKG_CACHE_STMT 関数に対する EXECUTE 特権。

例

平均 CPU 時間ごとに配列されているデータベース・パッケージ・キャッシュから、すべての動的 SQL ステートメントをリストします。

```
db2 SELECT MEMBER,
      SECTION_TYPE ,
      TOTAL_CPU_TIME/NUM_EXEC_WITH_METRICS as
      AVG_CPU_TIME,EXECUTABLE_ID
      FROM TABLE(MON_GET_PKG_CACHE_STMT ( 'D', NULL, NULL, -2)) as T
        WHERE T.NUM_EXEC_WITH_METRICS <> 0 ORDER BY AVG_CPU_TIME
```

以下はこの照会の出力例です。

```
MEMBER SECTION_TYPE AVG_CPU_TIME         EXECUTABLE_ID
------ ------------ -------------------- --------------------------------------------------------------------------------
     0 D                             754 x'0100000000000000A0000000000000000000000020020081126171554951791'
     0 D                            2964 x'0100000000000000790000000000000000000000020020081126171533551120'
     0 D                            5664 x'0100000000000000C0000000000000000000000020020081126171720728997'
     0 D                            5723 x'0100000000000000B0000000000000000000000020020081126171657272914'
     0 D                            9762 x'0100000000000000D0000000000000000000000020020081126172409987719'

5 record(s) selected.
```

注: コンパイル環境を作成して、ステートメント・テキスト (2 MB の大きさになることもある) をメンバー間で転送するには、長い時間がかかります。パッケージ・キャッシュからすべてのステートメントのリストを取り出すときのパフォーマンスを改善するため、 STMT_TEXT および COMP_ENV_DESC 列を選択しないでください。

上の出力を使用した場合、*executable_id* を使用して、最もコストの高い (平均 CPU 時間の観点から) ステートメントに関する詳細情報を検出できます。

```
db2 SELECT STMT_TEXT FROM TABLE(MON_GET_PKG_CACHE_STMT)
      (null, x'0100000000000000D0000000000000000000000020020081126172409987719', null, -2))

STMT_TEXT
--------------------------------------------------------------------------------
SELECT * FROM EMPLOYEE
```

別の例として、ワークロード A に関連付けられた接続を持つ Alex という名前のユーザーを想定します。ワークロード A には COLLECT ACTIVITY METRICS セットがあります。別のユーザー Brent はワークロード B に関連付けられています。ワークロード B には、NONE に設定された COLLECT ACTIVITY METRICS があります。さらに、データベースの **mon_act_metrics** 構成パラメーターが NONE に設定されています。 Brent が次の照会を実行すると、

```
SELECT count(*) FROM syscat.tables
```

すべてのメトリックが 0 として返され、**num_exec_with_metrics** の値も 0 になります。その後、Alex が同じステートメントを実行しますが、ステートメントの実行に対して今回はメトリックが収集され、**num_exec_with_metrics** が増分します。そのため、Brent と Alex がそのステートメントを実行した後、以下の照会の結果は、

```
SELECT num_executions, num_exec_with_metrics, SUBSTR(stmt_text,1,50) AS stmt_text
    FROM TABLE (MON_GET_PKG_CACHE_STMT('d', null, null, -1)) AS tf
    WHERE stmt_text LIKE 'SELECT count%'
```

SELECT ステートメントが 2 回実行され、そのうちの 1 回でアクティビティー・メトリックが収集されたことを示します。

```
NUM_EXECUTIONS NUM_EXEC_WITH_METRICS STMT_TEXT
-------------- -------------------- --------------------
             2                    1 SELECT count(*) FROM syscat.tables

  1 record(s) selected.
```

使用上の注意

MON_GET_PKG_CACHE_STMT 表関数は、データベース・パッケージ・キャッシュ内の静的 SQL ステートメントと動的 SQL ステートメントの両方のポイント・イン・タイム・ビューを戻します。これにより、特定の SQL ステートメントに対して集約されたメトリックを検討し、照会パフォーマンスが低下している理由を迅速に判別することができます。戻されるメトリックは、ステートメントのそれぞれの実行中に収集されたメトリックの集約になります。

さらに、他のステートメントとの相対関係で、個々のキャッシュ付きセクションの振る舞いを比較し、 (実行コストの観点から) 最もコストの高いセクションまたはステートメントの識別に役立てることができます。

この関数でレポートされたアクティビティー・メトリックは、アクティビティーの実行の終わりにデータベース・キャッシュにロールアップされます。

あらゆるステートメントの実行のメトリック収集は、ワークロードの COLLECT ACTIVITY METRICS 節を介して、またはデータベース・レベルでの **mon_act_metrics** データベース構成パラメーターを介して制御されます。アクティビティー・メトリックが使用可能になっているワークロードまたはデータベースに関連付けられた接続でステートメントがサブミットされた場合、そのステートメントの実行のメトリックのみが収集されます。 MON_GET_PKG_CACHE_STMT 関数によって戻される **num_exec_with_metrics** エレメントは、収集されたメトリックを持っていて、報告された集約メトリックに寄与したステートメントの実行数を示します。ステートメントのどの実行に対してもメトリックが収集されていない場合には、**num_exec_with_metrics** エレメントは 0 であり、すべてのメトリック値は 0 として戻されます。

戻される情報

表 128. MON_GET_PKG_CACHE_STMT について戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
MEMBER	SMALLINT	member - データベース・メンバー
SECTION_TYPE	CHAR(1)	section_type - セクション・タイプ標識
INSERT_TIMESTAMP	TIMESTAMP	insert_timestamp - ステートメント挿入タイム・スタンプ
EXECUTABLE_ID	VARCHAR(32) FOR BIT DATA	executable_id - 実行可能 ID
PACKAGE_NAME	VARCHAR(128)	package_name - パッケージ名。この出力は、静的 SQL ステートメントでのみ有効です。ステートメントが動的である場合、NULL 値が戻されます。

表 128. MON_GET_PKG_CACHE_STMT について戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
PACKAGE_SCHEMA	VARCHAR(128)	package_schema - パッケージ・スキーマ。この出力は、静的 SQL ステートメントでのみ有効です。ステートメントが動的である場合、NULL 値が戻されます。
PACKAGE_VERSION_ID	VARCHAR(64)	package_version_id - パッケージ・バージョン。この出力は、静的 SQL ステートメントでのみ有効です。ステートメントが動的であるか、または静的ステートメントのパッケージ・バージョンを指定しなかった場合、NULL 値が戻されます。パッケージの作成時にパッケージのバージョン ID が指定されなかった場合、静的ステートメントに対して空ストリングが戻されます。
SECTION_NUMBER	BIGINT	section_number - セクション番号。ステートメントが動的である場合、NULL 値が戻されます。
EFFECTIVE_ISOLATION	CHAR(2)	effective_isolation - 有効な分離。これは、セクションで有効な分離値です。コンパイル時に最初に要求されたものとは異なる可能性があります。
NUM_EXECUTIONS	BIGINT	num_executions - ステートメント実行回数
NUM_EXEC_WITH_METRICS	BIGINT	num_exec_with_metrics - メトリック収集を伴う実行数
PREP_TIME	BIGINT	prep_time - 準備時間。 PREP_TIME は、動的 SQL ステートメントでのみ有効であることに注意してください。静的 SQL ステートメントの場合、PREP_TIME は 0 としてレポートされます。
TOTAL_ACT_TIME	BIGINT	total_act_time - 合計アクティビティー時間
TOTAL_ACT_WAIT_TIME	BIGINT	total_act_wait_time - 合計アクティビティー待機時間
TOTAL_CPU_TIME	BIGINT	total_cpu_time - 合計 CPU 時間
POOL_READ_TIME	BIGINT	pool_read_time - バッファー・プール物理読み取り時間の合計
POOL_WRITE_TIME	BIGINT	pool_write_time - バッファー・プール物理書き込み時間の合計
DIRECT_READ_TIME	BIGINT	direct_read_time - 直接読み取り時間
DIRECT_WRITE_TIME	BIGINT	direct_write_time - 直接書き込み時間
LOCK_WAIT_TIME	BIGINT	lock_wait_time - ロック待機中の時間
TOTAL_SECTION_SORT_TIME	BIGINT	total_section_sort_time - セクション・ソート時間合計
TOTAL_SECTION_SORT_PROC_TIME	BIGINT	total_section_sort_proc_time - セクション・ソート処理時間合計
TOTAL_SECTION_SORTS	BIGINT	total_section_sorts - セクション・ソート合計
LOCK_ESCALS	BIGINT	lock_escals - ロック・エスカレーション数
LOCK_WAITS	BIGINT	lock_waits - ロック待機数
ROWS_MODIFIED	BIGINT	rows_modified - 変更された行数
ROWS_READ	BIGINT	rows_read - 読み取り行数
ROWS_RETURNED	BIGINT	rows_returned - 戻り行数

表 128. MON_GET_PKG_CACHE_STMT について戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
DIRECT_READS	BIGINT	direct_reads - データベースからの直接読み取り
DIRECT_READ_REQS	BIGINT	direct_read_reqs - 直接読み取り要求
DIRECT_WRITES	BIGINT	direct_writes - データベースへの直接書き込み
DIRECT_WRITE_REQS	BIGINT	direct_write_reqs - 直接書き込み要求
POOL_DATA_L_READS	BIGINT	pool_data_l_reads - バッファー・プール・データの論理読み取り
POOL_TEMP_DATA_L_READS	BIGINT	pool_temp_data_l_reads - バッファー・プール一時データの論理読み取り
POOL_XDA_L_READS	BIGINT	pool_xda_l_reads - バッファー・プール XDA データの論理読み取り
POOL_TEMP_XDA_L_READS	BIGINT	pool_temp_xda_l_reads - バッファー・プール一時 XDA データの論理読み取り
POOL_INDEX_L_READS	BIGINT	pool_index_l_reads - バッファー・プール索引の論理読み取り
POOL_TEMP_INDEX_L_READS	BIGINT	pool_temp_index_l_reads - バッファー・プール一時索引の論理読み取り
POOL_DATA_P_READS	BIGINT	pool_data_p_reads - バッファー・プール・データの物理読み取り
POOL_TEMP_DATA_P_READS	BIGINT	pool_temp_data_p_reads - バッファー・プール一時データの物理読み取り
POOL_XDA_P_READS	BIGINT	pool_xda_p_reads - バッファー・プール XDA データの物理読み取り
POOL_TEMP_XDA_P_READS	BIGINT	pool_temp_xda_p_reads - バッファー・プール一時 XDA データの物理読み取り
POOL_INDEX_P_READS	BIGINT	pool_index_p_reads - バッファー・プール索引の物理読み取り
POOL_TEMP_INDEX_P_READS	BIGINT	pool_temp_index_p_reads - バッファー・プール一時索引の物理読み取り
POOL_DATA_WRITES	BIGINT	pool_data_writes - バッファー・プールへのデータの書き込み
POOL_XDA_WRITES	BIGINT	pool_xda_writes - バッファー・プール XDA データの書き込み
POOL_INDEX_WRITES	BIGINT	pool_index_writes - バッファー・プール索引の書き込み
TOTAL_SORTS	BIGINT	total_sorts - ソート合計
POST_THRESHOLD_SORTS	BIGINT	post_threshold_sorts - ポストしきい値ソート
POST_SHRTHRESHOLD_SORTS	BIGINT	post_shrthreshold_sorts - ポスト共有しきい値ソート
SORT_OVERFLOWS	BIGINT	sort_overflows - ソート・オーバーフロー
WLM_QUEUE_TIME_TOTAL	BIGINT	wlm_queue_time_total - ワークロード・マネージャー合計キュー時間
WLM_QUEUE_ASSIGNMENTS_TOTAL	BIGINT	wlm_queue_assignments_total - ワークロード・マネージャー合計キュー割り当て数
DEADLOCKS	BIGINT	deadlocks - デッドロック検出数

表 128. MON_GET_PKG_CACHE_STMT について戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
FCM_RECV_VOLUME	BIGINT	fcm_recv_volume - FCM 受信ボリューム
FCM_RECVS_TOTAL	BIGINT	fcm_recvs_total - FCM 合計受信数
FCM_SEND_VOLUME	BIGINT	fcm_send_volume - FCM 送信ボリューム
FCM_SENDS_TOTAL	BIGINT	fcm_sends_total - FCM 合計送信数
FCM_RECV_WAIT_TIME	BIGINT	fcm_recv_wait_time - FCM 受信待機時間
FCM_SEND_WAIT_TIME	BIGINT	fcm_send_wait_time - FCM 送信待ち時間
LOCK_TIMEOUTS	BIGINT	lock_timeouts - ロック・タイムアウト数
LOG_BUFFER_WAIT_TIME	BIGINT	log_buffer_wait_time - ログ・バッファー待ち時間
NUM_LOG_BUFFER_FULL	BIGINT	num_log_buffer_full - フル・ログ・バッファーの回数
LOG_DISK_WAIT_TIME	BIGINT	log_disk_wait_time - ログ・ディスク待機時間
LOG_DISK_WAITS_TOTAL	BIGINT	log_disk_waits_total - ログ・ディスク待機の合計
LAST_METRICS_UPDATE	TIMESTAMP	last_metrics_update - メトリックの最終更新タイム・スタンプ
NUM_COORD_EXEC	BIGINT	num_coord_exec - コーディネーター・エージェントによる実行数
NUM_COORD_EXEC_WITH_METRICS	BIGINT	num_coord_exec_with_metrics - コーディネーター・エージェントによる実行数
VALID	CHAR(1)	valid - セクション妥当性検査標識
TOTAL_ROUTINE_TIME	BIGINT	total_routine_time - ルーチン時間の合計
TOTAL_ROUTINE_INVOCATIONS	BIGINT	total_routine_invocations - ルーチン呼び出しの合計回数
ROUTINE_ID	BIGINT	将来の利用のために予約済み。
STMT_TYPE_ID	VARCHAR(32)	stmt_type_id - ステートメント・タイプ ID
QUERY_COST_ESTIMATE	BIGINT	query_cost_estimate - 照会コストの見積もり
STMT_PKG_CACHE_ID	BIGINT	stmt_pkgcache_id - ステートメント・パッケージ・キャッシュ ID
COORD_STMT_EXEC_TIME	BIGINT	coord_stmt_exec_time - コーディネーター・エージェントによるステートメントの実行時間
STMT_EXEC_TIME	BIGINT	stmt_exec_time - ステートメントの実行時間
TOTAL_SECTION_TIME	BIGINT	total_section_time - セクション時間の合計
TOTAL_SECTION_PROC_TIME	BIGINT	total_section_proc_time - セクション処理時間の合計
TOTAL_ROUTINE_NON_SECT_TIME	BIGINT	total_routine_non_sect_time - 非セクション・ルーチン実行時間
TOTAL_ROUTINE_NON_SECT_PROC_TIME	BIGINT	total_routine_non_sect_proc_time - 非セクション処理時間
STMT_TEXT	CLOB(2MB)	stmt_text - SQL 動的ステートメント・テキスト
COMP_ENV_DESC	BLOB(10K)	comp_env_desc - コンパイル環境ハンドル。既存の COMPILATION_ENV 表関数を使用して、必要に応じて特定のステートメントの詳細なコンパイル環境を取得できます。
ADDITIONAL_DETAILS	BLOB(100K)	将来追加されるメトリックのために予約されています。

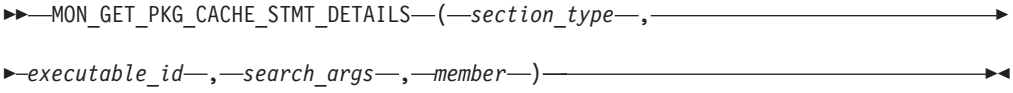
MON_GET_PKG_CACHE_STMT_DETAILS - パッケージ・キャッシュ項目に関する詳細メトリックの取得

MON_GET_PKG_CACHE_STMT_DETAILS 表関数は、1 つ以上のパッケージ・キャッシュ項目に関する詳細メトリックを戻します。

注: バージョン 9.7 のフィックスパック 1 以前で作成されたデータベースを使用している場合、このルーチンを実行するには、既に db2updv97 コマンドを実行済みでなければなりません。バージョン 9.7 以前に作成されたデータベースを使用している場合は、db2updv97 コマンドを実行する必要はありません (データベースのマイグレーションによって自動的にカタログ更新が実行されるため)。バージョン 9.7 にダウングレードすると、このルーチンは機能しなくなります。

MON_GET_PKG_CACHE_STMT_DETAILS 表関数で戻されるメトリックは、パッケージ・キャッシュ内のステートメントに関するすべてのメトリックの累計を表します。ステートメント・メトリックは、アクティビティー完了時にパッケージ・キャッシュにロールアップされます。

構文



スキーマは SYSPROC です。

表関数パラメーター

section_type
オプションの入力引数 (D または S)、タイプは CHAR(1)。戻されるステートメントの情報の種類を指定します。引数が NULL または空ストリングである場合、すべての SQL ステートメントについての情報が戻されます。大/小文字は区別されません。D は動的を表し、S 静的を表します。

executable_id
オプションの入力引数、タイプは VARCHAR(32)。データベース・パッケージ・キャッシュの固有のセクションを指定するビット・データを表します。NULL 値を指定すると、すべての SQL ステートメントについての情報が戻されます。*executable_id* が指定されている場合、*section_type* 引数は無視されます。例えば、動的ステートメントで *executable_id* が指定されている場合、*section_type* が静的 ("S") として指定されていても、この表関数によって動的ステートメントの詳細が戻されます。

search_args
オプションのタイプ CLOB(1K) の入力パラメーター。これを使用して 1 つ以上の検索引数ストリングをオプションで指定できます。例えば、

```
'<modified_within>5</modified_within><update_boundary_time>myPkgEvmon
    </update_boundary_time>'
```

使用可能な検索引数タグは次のとおりです。

- '<modified_within>X</modified_within>'

現在までの *X* 分間にキャッシュに挿入された、または実行されたステートメント項目だけを戻します (*X* は正の整数値)。引数が指定されない場合、キャッシュ内のすべての項目が戻されます。

- '<update_boundary_time>evmon_name</update_boundary_time>'

  evmon_name で指定されたパッケージ・キャッシュ・イベント・モニターに関して、イベント・モニター境界タイム・スタンプを現在時刻に更新します。このイベント・モニターの WHERE 節で出力基準として `where updated_since_boundary_time` が指定される場合、今後にメトリックが更新されるパッケージ・キャッシュ項目だけが、パッケージ・キャッシュからの退去時にキャプチャーされます。指定されたパッケージ・キャッシュ・イベント・モニターがコマンド発行時にアクティブである場合にのみ、この操作が効果を及ぼします。

- '<stmt_details>true</stmt_details>' または '<stmt_details>false</stmt_details>'

  結果の XML 文書に *stmt_text* および *comp_env_desc* データを含めるか、または除外します。これを使用すると、文書のこのような比較的大きい部分が不要な場合に、それらを除外できます (例えば、フォーマット設定された行ベースの出力を戻す MON_FORMAT_XML_* 表関数に入力を提供するために XML 文書を使用する場合)。この引数タグを指定しない場合、デフォルトでは *stmt_text* および *comp_env_desc* データが含まれます。

それぞれの入力引数は一度だけ指定できます。検索引数タグは小文字で指定する必要があります。

member
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なメンバーを指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・メンバーの場合は -1、すべてのデータベース・メンバーの場合は -2 を指定します。 NULL 値を指定すると、-1 が設定されます。

許可

MON_GET_PKG_CACHE_STMT_DETAILS 関数に対する EXECUTE 特権。

例

最初の例は、パッケージ・キャッシュを調査して、既に読み取った (および戻した) 行数の多い上位 10 個のステートメントを選ぶ方法を示しています。さらに、これらの各ステートメントの実行に費やされた時間の累計が結果に示されます (STMT_EXEC_TIME 出力列)。

```
SELECT SUBSTR(DETMETRICS.STMT_TEXT, 1, 40) STMT_TEXT,
       DETMETRICS.ROWS_RETURNED,
       DETMETRICS.STMT_EXEC_TIME
FROM TABLE(MON_GET_PKG_CACHE_STMT_DETAILS(CAST(NULL AS CHAR(1)),
    CAST(NULL AS VARCHAR(32) FOR BIT DATA),
    CAST(NULL AS CLOB(1K)), -1)) AS STMT_METRICS,
    XMLTABLE (XMLNAMESPACES( DEFAULT 'http://www.ibm.com/xmlns/prod/db2/mon'),
       '$DETMETRICS/db2_pkg_cache_stmt_details' PASSING
    XMLPARSE(DOCUMENT STMT_METRICS.DETAILS) as "DETMETRICS"
    COLUMNS "STMT_TEXT" CLOB PATH 'stmt_text',
        "ROWS_RETURNED" BIGINT PATH 'activity_metrics/rows_returned',
```

```
        "STMT_EXEC_TIME" BIGINT PATH 'activity_metrics/stmt_exec_time'
    ) AS DETMETRICS
ORDER BY rows_returned DESC
FETCH FIRST 10 ROWS ONLY
```

以下はこの照会の出力例です。

```
STMT_TEXT                              ROWS_RETURNED STMT_EXEC_TIME
-------------------------------------- ------------- --------------
SELECT CREATOR, NAME, CTIME FROM SYSIBM.           134             38
SELECT SUBSTR(DETMETRICS.STMT_TEXT, 1, 4            44            336
SELECT SUBSTR(DETMETRICS.STMT_TEXT, 1, 4            10            333
SELECT COLNAME, TYPENAME FROM  SYSCAT.CO            10              6
SELECT SUBSTR(DETMETRICS.STMT_TEXT, 1, 4            10            334
SELECT TRIGNAME FROM  SYSCAT.TRIGGERS WH             8              1
SELECT COUNT(*) FROM SYSCAT.TABLESPACES              2              0
SELECT POLICY FROM SYSTOOLS.POLICY WHERE             1              0
CALL SYSPROC.POLICY_INSTALL ('I','DB2Tab             1             62
CALL SYSPROC.POLICY_INSTALL ('I','DB2Tab             1             64

  10 record(s) selected.
```

2 番目の例は、実行中にロックを待機した動的 SQL ステートメントに関して、実行の数、ロック待機の数、およびロック待機ごとに費やされた平均時間を示します。出力にはパッケージ・キャッシュ項目の存続期間にわたって累算された値が示されますが、(modified_within 引数タグを 1 に設定することで) 最近 1 分以内に実行されたステートメントの情報に限定します。ステートメントの詳細情報 (*stmt_text* および *comp_env_desc* データ) は不要であり、レポートの生成処理でのコストが高いため、(stmt_details 引数タグを false に設定することにより) 照会ではこれらが除外されます。

```
SELECT NUM_EXEC_WITH_METRICS, LOCK_WAITS,
  (LOCK_WAIT_TIME / LOCK_WAITS) AVG_LOCK_WAIT_TIME
FROM TABLE(MON_GET_PKG_CACHE_STMT_DETAILS('D', CAST(NULL
   AS VARCHAR(32) FOR BIT DATA),
   CLOB(
      '<modified_within>1</modified_within><stmt_details>false</stmt_details>')
        , -1))
   AS STMT_METRICS,
   XMLTABLE (XMLNAMESPACES( DEFAULT 'http://www.ibm.com/xmlns/prod/db2/mon'),
      '$DETMETRICS/db2_pkg_cache_stmt_details' PASSING
   XMLPARSE(DOCUMENT STMT_METRICS.DETAILS) as "DETMETRICS"
   COLUMNS "NUM_EXEC_WITH_METRICS" BIGINT PATH 'num_exec_with_metrics',
      "LOCK_WAITS" BIGINT PATH 'lock_waits',
      "LOCK_WAIT_TIME" BIGINT PATH 'activity_metrics/lock_wait_time'
   ) AS DETMETRICS
WHERE LOCK_WAITS <> 0
ORDER BY AVG_LOCK_WAIT_TIME DESC
```

以下はこの照会の出力例です。

```
NUM_EXEC_WITH_METRICS LOCK_WAITS           AVG_LOCK_WAIT_TIME
--------------------- -------------------- --------------------
                    4                    2                  139
                    9                    3                   90
```

使用上の注意

この関数によって戻されるメトリックは、パッケージ・キャッシュ内のステートメントに関するすべてのメトリックの累計を表します。ステートメント・メトリックは、アクティビティー完了時にパッケージ・キャッシュにロールアップされます。

戻される情報

表 129. MON_GET_PKG_CACHE_STMT_DETAILS で戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
MEMBER	SMALLINT	member - データベース・メンバー
SECTION_TYPE	CHAR(1)	section_type - セクション・タイプ標識
EXECUTABLE_ID	VARCHAR(32) FOR BIT DATA	executable_id - 実行可能 ID
DETAILS	BLOB(8M)	作業単位についての詳細メトリックを含む XML 文書。エレメントの説明については、本書の表 130を参照してください。

表 130. MON_GET_PKG_CACHE_STMT_DETAILS で戻される詳細メトリック

エレメント名	データ・タイプ	説明
member	xs:short	member - データベース・メンバー
valid	xs:string(1)	valid - セクション妥当性検査標識
executable_id	xs:hexBinary(32)	executable_id - 実行可能 ID
section_type	xs:string(1)	section_type - セクション・タイプ標識
num_executions	xs:nonNegativeInteger	num_executions - ステートメント実行回数
num_exec_with_metrics	xs:nonNegativeInteger	num_exec_with_metrics - メトリック収集を伴う実行数
prep_time	xs:nonNegativeInteger	prep_time - 準備時間。 PREP_TIME は、動的 SQL ステートメントでのみ有効であることに注意してください。静的 SQL ステートメントの場合、PREP_TIME は 0 としてレポートされます。
effective_isolation	xs:string(2)	effective_isolation - 有効な分離。これは、セクションで有効な分離値です。コンパイル時に最初に要求されたものとは異なる可能性があります。
stmt_pkgcache_id	xs:long	stmt_pkgcache_id - ステートメント・パッケージ・キャッシュ ID
query_cost_estimate	xs:long	query_cost_estimate - 照会コストの見積もり
stmt_type_id	xs:string	stmt_type_id - ステートメント・タイプ ID
insert_timestamp	xs:dateTime	insert_timestamp - ステートメント挿入タイム・スタンプ
last_metrics_update	xs:dateTime	last_metrics_update - メトリックの最終更新タイム・スタンプ
package_name	xs:string(128)	package_name - パッケージ名。この出力は、静的 SQL ステートメントでのみ有効です。ステートメントが動的である場合、NULL 値が戻されます。
package_schema	xs:string(128)	package_schema - パッケージ・スキーマ。この出力は、静的 SQL ステートメントでのみ有効です。ステートメントが動的である場合、NULL 値が戻されます。

表 130. MON_GET_PKG_CACHE_STMT_DETAILS で戻される詳細メトリック (続き)

エレメント名	データ・タイプ	説明
package_version_id	xs:string(64)	package_version_id - パッケージ・バージョン。この出力は、静的 SQL ステートメントでのみ有効です。ステートメントが動的であるか、または静的ステートメントのパッケージ・バージョンを指定しなかった場合には、このエレメントは生成されません。パッケージの作成時にパッケージのバージョン ID を指定しなかった場合には、静的ステートメントに対して空ストリングが戻されます。
section_number	xs:short	section_number - セクション番号。ステートメントが動的である場合には、このエレメントは生成されません。
stmt_text	xs:string(2097152)	stmt_text - SQL 動的ステートメント・テキスト
comp_env_desc	xs:hexBinary(10240)	comp_env_desc - コンパイル環境ハンドル。既存の COMPILATION_ENV 表関数を使用して、必要に応じて特定のステートメントの詳細なコンパイル環境を取得できます。
wlm_queue_time_total	xs:long	wlm_queue_time_total - ワークロード・マネージャー合計キュー時間
wlm_queue_assignments_total	xs:long	wlm_queue_assignments_total - ワークロード・マネージャー合計キュー割り当て数
fcm_tq_recv_wait_time	xs:long	fcm_tq_recv_wait_time - FCM 表キューの受信待ち時間
fcm_message_recv_wait_time	xs:long	fcm_message_recv_wait_time - FCM メッセージの受信待機時間
fcm_tq_send_wait_time	xs:long	fcm_tq_send_wait_time - FCM 表キューの送信待ち時間
fcm_message_send_wait_time	xs:long	fcm_message_send_wait_time - FCM メッセージ送信待ち時間
lock_wait_time	xs:long	lock_wait_time - ロック待機中の時間
lock_waits	xs:long	lock_waits - ロック待機数
direct_read_time	xs:long	direct_read_time - 直接読み取り時間
direct_read_reqs	xs:long	direct_read_reqs - 直接読み取り要求
direct_write_time	xs:long	direct_write_time - 直接書き込み時間
direct_write_reqs	xs:long	direct_write_reqs - 直接書き込み要求
log_buffer_wait_time	xs:long	log_buffer_wait_time - ログ・バッファー待ち時間
num_log_buffer_full	xs:long	num_log_buffer_full - フル・ログ・バッファーの回数
log_disk_wait_time	xs:long	log_disk_wait_time - ログ・ディスク待機時間
log_disk_waits_total	xs:long	log_disk_waits_total - ログ・ディスク待機の合計
pool_write_time	xs:long	pool_write_time - バッファー・プール物理書き込み時間の合計
pool_read_time	xs:long	pool_read_time - バッファー・プール物理読み取り時間の合計
audit_file_write_wait_time	xs:long	audit_file_write_wait_time - 監査ファイル書き込み待機時間
audit_file_writes_total	xs:long	audit_file_writes_total - 書き込まれた監査ファイルの合計数

表 130. MON_GET_PKG_CACHE_STMT_DETAILS で戻される詳細メトリック (続き)

エレメント名	データ・タイプ	説明
audit_subsystem_wait_time	xs:long	audit_subsystem_wait_time - 監査サブシステム待機時間
audit_subsystem_waits_total	xs:long	audit_subsystem_waits_total - 監査サブシステム待機の合計
diaglog_write_wait_time	xs:long	diaglog_write_wait_time - 診断ログ・ファイルの書き込み待機時間
diaglog_writes_total	xs:long	diaglog_writes_total - 診断ログ・ファイル書き込みの合計
fcm_send_wait_time	xs:long	fcm_send_wait_time - FCM 送信待ち時間
fcm_recv_wait_time	xs:long	fcm_recv_wait_time - FCM 受信待機時間
total_act_wait_time	xs:long	total_act_wait_time - 合計アクティビティー待機時間
total_section_sort_proc_time	xs:long	total_section_sort_proc_time - セクション・ソート処理時間合計
total_section_sort_time	xs:long	total_section_sort_time - セクション・ソート時間合計
total_section_sorts	xs:long	total_section_sorts - セクション・ソート合計
total_act_time	xs:long	total_act_time - 合計アクティビティー時間
rows_read	xs:long	rows_read - 読み取り行数
rows_modified	xs:long	rows_modified - 変更された行数
pool_data_l_reads	xs:long	pool_data_l_reads - バッファー・プール・データの論理読み取り
pool_index_l_reads	xs:long	pool_index_l_reads - バッファー・プール索引の論理読み取り
pool_temp_data_l_reads	xs:long	pool_temp_data_l_reads - バッファー・プール一時データの論理読み取り
pool_temp_index_l_reads	xs:long	pool_temp_index_l_reads - バッファー・プール一時索引の論理読み取り
total_cpu_time	xs:long	total_cpu_time - 合計 CPU 時間
pool_data_p_reads	xs:long	pool_data_p_reads - バッファー・プール・データの物理読み取り
pool_temp_data_p_reads	xs:long	pool_temp_data_p_reads - バッファー・プール一時データの物理読み取り
pool_xda_p_reads	xs:long	pool_xda_p_reads - バッファー・プール XDA データの物理読み取り
pool_temp_xda_p_reads	xs:long	pool_temp_xda_p_reads - バッファー・プール一時 XDA データの物理読み取り
pool_index_p_reads	xs:long	pool_index_p_reads - バッファー・プール索引の物理読み取り
pool_temp_index_p_reads	xs:long	pool_temp_index_p_reads - バッファー・プール一時索引の物理読み取り
pool_data_writes	xs:long	pool_data_writes - バッファー・プールへのデータの書き込み
pool_xda_writes	xs:long	pool_xda_writes - バッファー・プール XDA データの書き込み
pool_index_writes	xs:long	pool_index_writes - バッファー・プール索引の書き込み

表 130. MON_GET_PKG_CACHE_STMT_DETAILS で戻される詳細メトリック (続き)

エレメント名	データ・タイプ	説明
direct_reads	xs:long	direct_reads - データベースからの直接読み取り
direct_writes	xs:long	direct_writes - データベースへの直接書き込み
rows_returned	xs:long	rows_returned - 戻り行数
deadlocks	xs:long	deadlocks - デッドロック検出数
lock_timeouts	xs:long	lock_timeouts - ロック・タイムアウト数
lock_escals	xs:long	lock_escals - ロック・エスカレーション数
fcm_sends_total	xs:long	fcm_sends_total - FCM 合計送信数
fcm_recvs_total	xs:long	fcm_recvs_total - FCM 合計受信数
fcm_send_volume	xs:long	fcm_send_volume - FCM 送信ボリューム
fcm_recv_volume	xs:long	fcm_recv_volume - FCM 受信ボリューム
fcm_message_sends_total	xs:long	fcm_message_sends_total - FCM メッセージ合計送信数
fcm_message_recvs_total	xs:long	fcm_message_recvs_total - FCM メッセージ合計受信数
fcm_message_send_volume	xs:long	fcm_message_send_volume - FCM メッセージ送信ボリューム
fcm_message_recv_volume	xs:long	fcm_message_recv_volume - FCM メッセージ受信ボリューム
fcm_tq_sends_total	xs:long	fcm_tq_sends_total - FCM 表キューの合計送信数
fcm_tq_recvs_total	xs:long	fcm_tq_recvs_total - FCM 表キューの合計受信数
fcm_tq_send_volume	xs:long	fcm_tq_send_volume - FCM 表キューの送信ボリューム
fcm_tq_recv_volume	xs:long	fcm_tq_recv_volume - FCM 表キューの受信ボリューム
tq_tot_send_spills	xs:long	tq_tot_send_spills - オーバーフローした表キュー・バッファーの合計数
post_threshold_sorts	xs:long	post_threshold_sorts - ポストしきい値ソート
post_shrthreshold_sorts	xs:long	post_shrthreshold_sorts - ポスト共有しきい値ソート
sort_overflows	xs:long	sort_overflows - ソート・オーバーフロー
audit_events_total	xs:long	audit_events_total - 監査イベント合計数
total_sorts	xs:long	total_sorts - ソート合計
stmt_exec_time	xs:long	stmt_exec_time - ステートメントの実行時間
coord_stmt_exec_time	xs:long	coord_stmt_exec_time - コーディネーター・エージェントによるステートメントの実行時間
total_routine_non_sect_proc_time	xs:long	total_routine_non_sect_proc_time - 非セクション処理時間
total_routine_non_sect_time	xs:long	total_routine_non_sect_time - 非セクション・ルーチン実行時間
total_section_proc_time	xs:long	total_section_proc_time - セクション処理時間の合計
total_section_time	xs:long	total_section_time - セクション時間の合計
total_app_section_executions	xs:long	total_app_section_executions - セクション実行の合計回数
total_routine_user_code_proc_time	xs:long	total_routine_user_code_proc_time - ルーチン・ユーザー・コード処理時間の合計
total_routine_user_code_time	xs:long	total_routine_user_code_time - ルーチン・ユーザー・コード時間の合計

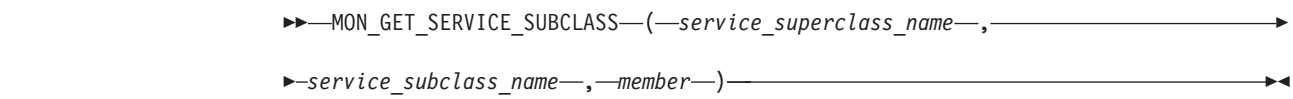
表 130. MON_GET_PKG_CACHE_STMT_DETAILS で戻される詳細メトリック (続き)

エレメント名	データ・タイプ	説明
total_routine_time	xs:long	total_routine_time - ルーチン時間の合計
num_coord_exec	xs:long	num_coord_exec - コーディネーター・エージェントによる実行数
num_coord_exec_with_metrics	xs:long	num_coord_exec_with_metrics - メトリックを伴うコーディネーター・エージェントによる実行数
num_thresh_violations	xs:long	num_threshold_violations - しきい値違反の回数
num_lw_thresh_exceeded	xs:long	num_lw_thresh_exceeded - しきい値を超えた回数
total_routine_invocations	xs:long	total_routine_invocations - ルーチン呼び出しの合計回数

MON_GET_SERVICE_SUBCLASS 表関数 - サービス・サブクラスのメトリックの取得

MON_GET_SERVICE_SUBCLASS 表関数は、1 つ以上のサービス・サブクラスのメトリックを戻します。

構文



スキーマは SYSPROC です。

表関数パラメーター

service_superclass_name
この関数を呼び出すときに現在接続されているデータベースで有効なサービス・スーパークラス名を指定する、タイプ VARCHAR(128) の入力引数。引数が NULL または空ストリングである場合、データベース内のすべてのスーパークラスについてメトリックが取得されます。

service_subclass_name
この関数を呼び出すときに現在接続されているデータベースで有効なサービス・サブクラス名を指定する、タイプ VARCHAR(128) の入力引数。引数が NULL または空ストリングである場合、データベース内のすべてのサブクラスについてメトリックが取得されます。

member
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なメンバーを指定する、タイプ INTEGER の入力引数。現行のデータベース・メンバーには -1、すべてのデータベース・メンバーには -2 を指定します。NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

MON_GET_SERVICE_SUBCLASS 関数に対する EXECUTE 特権。

例

CPU 使用率で順序付けられる、使用された合計 CPU 時間と、各サービス・クラスの処理された要求の合計数を表示します。

```
SELECT varchar(service_superclass_name,30) as service_superclass,
                  varchar(service_subclass_name,30) as service_subclass,
                  sum(total_cpu_time) as total_cpu,
                  sum(app_rqsts_completed_total) as total_rqsts
FROM TABLE(MON_GET_SERVICE_SUBCLASS('','',-2)) AS t
GROUP BY service_superclass_name, service_subclass_name
ORDER BY total_cpu desc
```

以下はこの照会の出力例です。

```
SERVICE_SUPERCLASS             SERVICE_SUBCLASS               ...
------------------------------ ------------------------------ ...
SYSDEFAULTUSERCLASS            SYSDEFAULTSUBCLASS             ...
SYSDEFAULTMAINTENANCECLASS     SYSDEFAULTSUBCLASS             ...
SYSDEFAULTSYSTEMCLASS          SYSDEFAULTSUBCLASS             ...

  3 record(s) selected.
```

照会の出力 (続き)。

```
... TOTAL_CPU            TOTAL_RQSTS
... -------------------- --------------------
...               967673                  100
... .                  0                    0
...                    0                    0
```

使用上の注意

MON_GET_SERVICE_SUBCLASS 表関数で戻されるメトリックは、指示されたサービス・サブクラスの下で実行された要求についてのすべてのメトリックの累計を表します。メトリックは、作業単位境界でサービス・クラスにロールアップされ、要求の実行中には定期的にロールアップされます。したがって、この表関数で報告される値は、直前の ROLLUP 時のシステムの現行状態を反映しています。メトリックの値は確実に増加します。ある時間間隔に対する指定されたメトリックの値を判別するには、 MON_GET_SERVICE_SUBCLASS 表関数を使用してその間隔の始めと終わりのメトリックを照会し、差異を計算します。

要求メトリックは、サービス・スーパークラスに対する COLLECT REQUEST METRICS 節、およびデータベース・レベルの *mon_req_metrics* データベース構成パラメーターを介して制御されます。親サービス・スーパークラスで要求メトリックを使用可能にしているサービス・サブクラスのエージェントが要求を処理する場合にのみ、またはデータベース全体で要求メトリック・コレクションが有効な場合にのみ、その要求に対しメトリックが収集されます。デフォルトでは、要求メトリックはデータベース・レベルで使用可能です。要求メトリックがデータベース・レベルでもサービス・スーパークラスに対しても使用不可になっている場合、そのサービス・スーパークラスにマッピングされた接続ごとに報告されるメトリックは増加を停止します (または、要求メトリックがデータベースのアクティブ化時に無効であった場合には 0 のままになります)。

MON_GET_SERVICE_SUBCLASS 表関数は、サービス・サブクラスごとおよびメンバーごとに 1 行のデータを戻します。 (メンバー上の) サービス・クラス全体または (1 つ以上のサービス・クラスの) メンバー全体の集約は実行されません。ただ

し、集約は例に示されるように SQL 照会を使用して実行できます。この入力パラメーターの影響として、ANDing されます。したがって、競合する入力パラメーター (例えば、スーパークラス名 SUPA と SUPA のサブクラスではないサブクラス名 SUBB など) を指定する場合、行は戻されません。

ヒント: 要求は、複数のサービス・サブクラスで実行される場合があります。例えば、REMAP ACTIVITY アクションによってワークロード・マネージャー (WLM) しきい値を使用して要求を 1 つのサービス・サブクラスから別のサービス・サブクラスへマッピングする場合、この状況が生じる可能性があります。メトリックに使用された時間は、要求が実行されるサービス・サブクラスごとに更新されますが、要求カウンターは、要求が完了したサービス・サブクラスに対して増分します。したがって、単一のサブクラスの要求時間の平均を分析する必要はありません。アクティビティーをマッピングできるすべてのサブクラスは相互に関連付けて分析する必要があります。例えば、サービス・サブクラス A からサービス・クラス B にアクティビティーをマッピングできるしきい値があり、要求の平均を計算する場合、サービス・サブサービス A および B についてのカウンターおよびメトリックを集約し、その集約を使用して平均を計算する必要があります。

戻される情報

表 131. MON_GET_SERVICE_SUBCLASS について戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SERVICE_SUPERCLASS_NAME	VARCHAR(128)	service_superclass_name - サービス・スーパークラス名
SERVICE_SUBCLASS_NAME	VARCHAR(128)	service_subclass_name - サービス・サブクラス名
SERVICE_CLASS_ID	INTEGER	service_class_id - サービス・クラス ID
MEMBER	SMALLINT	member - データベース・メンバー
ACT_ABORTED_TOTAL	BIGINT	act_aborted_total - 打ち切られたアクティビティーの合計数
ACT_COMPLETED_TOTAL	BIGINT	act_completed_total - 完了したアクティビティーの合計数
ACT_REJECTED_TOTAL	BIGINT	act_rejected_total - リジェクトされたアクティビティーの合計数
AGENT_WAIT_TIME	BIGINT	agent_wait_time - エージェント待機時間
AGENT_WAITS_TOTAL	BIGINT	agent_waits_total - エージェント待機の合計
POOL_DATA_L_READS	BIGINT	pool_data_l_reads - バッファー・プール・データの論理読み取り
POOL_INDEX_L_READS	BIGINT	pool_index_l_reads - バッファー・プール索引の論理読み取り
POOL_TEMP_DATA_L_READS	BIGINT	pool_temp_data_l_reads - バッファー・プール一時データの論理読み取り
POOL_TEMP_INDEX_L_READS	BIGINT	pool_temp_index_l_reads - バッファー・プール一時索引の論理読み取り
POOL_TEMP_XDA_L_READS	BIGINT	pool_temp_xda_l_reads - バッファー・プール一時 XDA データの論理読み取り

表 131. MON_GET_SERVICE_SUBCLASS について戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
POOL_XDA_L_READS	BIGINT	pool_temp_xda_l_reads - バッファー・プール一時 XDA データの論理読み取り
POOL_DATA_P_READS	BIGINT	pool_data_p_reads - バッファー・プール・データの物理読み取り
POOL_INDEX_P_READS	BIGINT	pool_index_p_reads - バッファー・プール索引の物理読み取り
POOL_TEMP_DATA_P_READS	BIGINT	pool_temp_data_p_reads - バッファー・プール一時データの物理読み取り
POOL_TEMP_INDEX_P_READS	BIGINT	pool_temp_index_p_reads - バッファー・プール一時索引の物理読み取り
POOL_TEMP_XDA_P_READS	BIGINT	pool_temp_xda_p_reads - バッファー・プール一時 XDA データの物理読み取り
POOL_XDA_P_READS	BIGINT	pool_xda_p_reads - バッファー・プール XDA データの物理読み取り
POOL_DATA_WRITES	BIGINT	pool_data_writes - バッファー・プールへのデータの書き込み
POOL_INDEX_WRITES	BIGINT	pool_index_writes - バッファー・プール索引の書き込み
POOL_XDA_WRITES	BIGINT	pool_xda_writes - バッファー・プール XDA データの書き込み
POOL_READ_TIME	BIGINT	pool_read_time - バッファー・プール物理読み取り時間の合計
POOL_WRITE_TIME	BIGINT	pool_write_time - バッファー・プール物理書き込み時間の合計
CLIENT_IDLE_WAIT_TIME	BIGINT	client_idle_wait_time - クライアントのアイドル待機時間
DEADLOCKS	BIGINT	deadlocks - デッドロック検出数
DIRECT_READS	BIGINT	direct_reads - データベースからの直接読み取り
DIRECT_READ_TIME	BIGINT	direct_read_time - 直接読み取り時間
DIRECT_WRITES	BIGINT	direct_writes - データベースへの直接書き込み
DIRECT_WRITE_TIME	BIGINT	direct_write_time - 直接書き込み時間
DIRECT_READ_REQS	BIGINT	direct_read_reqs - 直接読み取り要求
DIRECT_WRITE_REQS	BIGINT	direct_write_reqs - 直接書き込み要求
FCM_RECV_VOLUME	BIGINT	fcm_recv_volume - FCM 受信ボリューム
FCM_RECVS_TOTAL	BIGINT	fcm_recvs_total - FCM 合計受信数
FCM_SEND_VOLUME	BIGINT	fcm_send_volume - FCM 送信ボリューム
FCM_SENDS_TOTAL	BIGINT	fcm_sends_total - FCM 合計送信数
FCM_RECV_WAIT_TIME	BIGINT	fcm_recv_wait_time - FCM 受信待機時間
FCM_SEND_WAIT_TIME	BIGINT	fcm_send_wait_time - FCM 送信待ち時間
IPC_RECV_VOLUME	BIGINT	ipc_recv_volume - プロセス間通信受信ボリューム

表 131. MON_GET_SERVICE_SUBCLASS について戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
IPC_RECV_WAIT_TIME	BIGINT	ipc_recv_wait_time - プロセス間通信受信待ち時間
IPC_RECVS_TOTAL	BIGINT	ipc_recvs_total - プロセス間通信合計受信数
IPC_SEND_VOLUME	BIGINT	ipc_send_volume - プロセス間通信送信ボリューム
IPC_SEND_WAIT_TIME	BIGINT	ipc_send_wait_time - プロセス間通信送信待ち時間
IPC_SENDS_TOTAL	BIGINT	ipc_sends_total - プロセス間通信合計送信数
LOCK_ESCALS	BIGINT	lock_escals - ロック・エスカレーション数
LOCK_TIMEOUTS	BIGINT	lock_timeouts - ロック・タイムアウト数
LOCK_WAIT_TIME	BIGINT	lock_wait_time - ロック待機中の時間
LOCK_WAITS	BIGINT	lock_waits - ロック待機数
LOG_BUFFER_WAIT_TIME	BIGINT	log_buffer_wait_time - ログ・バッファー待ち時間
NUM_LOG_BUFFER_FULL	BIGINT	num_log_buffer_full - フル・ログ・バッファーの回数
LOG_DISK_WAIT_TIME	BIGINT	log_disk_wait_time - ログ・ディスク待機時間
LOG_DISK_WAITS_TOTAL	BIGINT	log_disk_waits_total - ログ・ディスク待機の合計
RQSTS_COMPLETED_TOTAL	BIGINT	rqsts_completed_total - 完了した要求の合計数
ROWS_MODIFIED	BIGINT	rows_modified - 変更された行数
ROWS_READ	BIGINT	rows_read - 読み取り行数
ROWS_RETURNED	BIGINT	rows_returned - 戻り行数
TCPIP_RECV_VOLUME	BIGINT	tcpip_recv_volume - TCP/IP 受信ボリューム
TCPIP_SEND_VOLUME	BIGINT	tcpip_send_volume - TCP/IP 送信ボリューム
TCPIP_RECV_WAIT_TIME	BIGINT	tcpip_recv_wait_time - TCP/IP 受信待ち時間
TCPIP_RECVS_TOTAL	BIGINT	tcpip_recvs_total - TCP/IP 合計受信数
TCPIP_SEND_WAIT_TIME	BIGINT	tcpip_send_wait_time - TCP/IP 送信待ち時間
TCPIP_SENDS_TOTAL	BIGINT	tcpip_sends_total - TCP/IP 合計送信数
TOTAL_APP_RQST_TIME	BIGINT	total_app_rqst_time - アプリケーション要求合計時間
TOTAL_RQST_TIME	BIGINT	total_rqst_time - 合計要求時間
WLM_QUEUE_TIME_TOTAL	BIGINT	wlm_queue_time_total - ワークロード・マネージャー合計キュー時間
WLM_QUEUE_ASSIGNMENTS _TOTAL	BIGINT	wlm_queue_assignments_total - ワークロード・マネージャー合計キュー割り当て数
TOTAL_RQST_MAPPED_IN	BIGINT	total_rqst_mapped_in - マップで含められた要求の合計
TOTAL_RQST_MAPPED_OUT	BIGINT	total_rqst_mapped_out - マップで除外された要求の合計
TOTAL_CPU_TIME	BIGINT	total_cpu_time - 合計 CPU 時間
TOTAL_WAIT_TIME	BIGINT	total_wait_time - 合計待ち時間

表 131. MON_GET_SERVICE_SUBCLASS について戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
APP_RQSTS_COMPLETED_TOTAL	BIGINT	app_rqsts_completed_total - 完了したアプリケーション要求の合計数
TOTAL_SECTION_SORT_TIME	BIGINT	total_section_sort_time - セクション・ソート時間合計
TOTAL_SECTION_SORT_PROC_TIME	BIGINT	total_section_sort_proc_time - セクション・ソート処理時間合計
TOTAL_SECTION_SORTS	BIGINT	total_section_sorts - セクション・ソート合計
TOTAL_SORTS	BIGINT	total_sorts - ソート合計
POST_THRESHOLD_SORTS	BIGINT	post_threshold_sorts - ポストしきい値ソート
POST_SHRTHRESHOLD_SORTS	BIGINT	post_shrthreshold_sorts - ポスト共有しきい値ソート
SORT_OVERFLOWS	BIGINT	sort_overflows - ソート・オーバーフロー
TOTAL_COMPILE_TIME	BIGINT	total_compile_time - コンパイル時間の合計
TOTAL_COMPILE_PROC_TIME	BIGINT	total_compile_proc_time - コンパイル処理時間の合計
TOTAL_COMPILATIONS	BIGINT	total_compilations - コンパイルの合計回数
TOTAL_IMPLICIT_COMPILE_TIME	BIGINT	total_implicit_compile_time - 暗黙的コンパイル時間の合計
TOTAL_IMPLICIT_COMPILE_PROC_TIME	BIGINT	total_implicit_compile_proc_time - 暗黙的コンパイルの処理時間の合計
TOTAL_IMPLICIT_COMPILATIONS	BIGINT	total_implicit_compilations - 暗黙的コンパイルの合計回数
TOTAL_SECTION_TIME	BIGINT	total_section_time - セクション時間の合計
TOTAL_SECTION_PROC_TIME	BIGINT	total_section_proc_time - セクション処理時間の合計
TOTAL_APP_SECTION_EXECUTIONS	BIGINT	total_app_section_executions - セクション実行の合計回数
TOTAL_ACT_TIME	BIGINT	total_act_time - 合計アクティビティー時間
TOTAL_ACT_WAIT_TIME	BIGINT	total_act_wait_time - 合計アクティビティー待機時間
ACT_RQSTS_TOTAL	BIGINT	act_rqsts_total - アクティビティー要求の合計
TOTAL_ROUTINE_TIME	BIGINT	total_routine_time - ルーチン時間の合計
TOTAL_ROUTINE_INVOCATIONS	BIGINT	total_routine_invocations - ルーチン呼び出しの合計回数
TOTAL_COMMIT_TIME	BIGINT	total_commit_time - コミット時間の合計
TOTAL_COMMIT_PROC_TIME	BIGINT	total_commit_proc_time - コミット処理時間の合計
TOTAL_APP_COMMITS	BIGINT	total_app_commits - アプリケーション・コミットの合計回数
INT_COMMITS	BIGINT	int_commits - 内部コミット数
TOTAL_ROLLBACK_TIME	BIGINT	total_rollback_time - ロールバック時間の合計
TOTAL_ROLLBACK_PROC_TIME	BIGINT	total_rollback_proc_time - ロールバック処理時間の合計

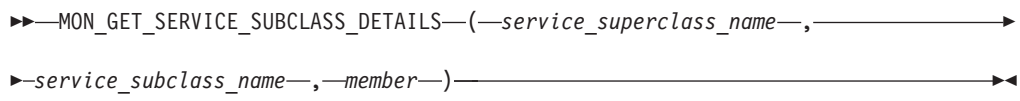
表 131. MON_GET_SERVICE_SUBCLASS について戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
TOTAL_APP_ROLLBACKS	BIGINT	total_app_rollbacks - アプリケーション・ロールバックの合計回数
INT_ROLLBACKS	BIGINT	int_rollbacks - 内部ロールバック数
TOTAL_RUNSTATS_TIME	BIGINT	total_runstats_time - ランタイム統計時間の合計
TOTAL_RUNSTATS_PROC_TIME	BIGINT	total_runstats_proc_time - ランタイム統計の処理時間の合計
TOTAL_RUNSTATS	BIGINT	total_runstats - ランタイム統計の合計回数
TOTAL_REORG_TIME	BIGINT	total_reorg_time - 再編成時間の合計
TOTAL_REORG_PROC_TIME	BIGINT	total_reorg_proc_time - 再編成の処理時間の合計
TOTAL_REORGS	BIGINT	total_reorgs - 再編成の合計回数
TOTAL_LOAD_TIME	BIGINT	total_load_time - ロード時間の合計
TOTAL_LOAD_PROC_TIME	BIGINT	total_load_proc_time - ロード処理時間の合計
TOTAL_LOADS	BIGINT	total_loads - ロードの合計回数
CAT_CACHE_INSERTS	BIGINT	cat_cache_inserts - カタログ・キャッシュ挿入数
CAT_CACHE_LOOKUPS	BIGINT	cat_cache_lookups - カタログ・キャッシュ参照数
PKG_CACHE_INSERTS	BIGINT	pkg_cache_inserts - パッケージ・キャッシュ挿入
PKG_CACHE_LOOKUPS	BIGINT	pkg_cache_lookups - パッケージ・キャッシュ参照
THRESH_VIOLATIONS	BIGINT	thresh_violations - しきい値違反の回数
NUM_LW_THRESH_EXCEEDED	BIGINT	num_lw_thresh_exceeded - しきい値を超えた回数
ADDITIONAL_DETAILS	BLOB(100K)	将来の利用のために予約済み

MON_GET_SERVICE_SUBCLASS_DETAILS 表関数 - 詳細サービス・サブクラスのメトリックの取得

MON_GET_SERVICE_SUBCLASS_DETAILS 表関数は、1 つ以上のサービス・サブクラスの詳細メトリックを戻します。

構文



スキーマは SYSPROC です。

表関数パラメーター

service_superclass_name
この関数を呼び出すときに現在接続されているデータベースで有効なサービス・スーパークラス名を指定する、タイプ VARCHAR(128) の入力引数。引数が NULL または空ストリングである場合、データベース内のすべてのスーパークラスについてメトリックが取得されます。

service_subclass_name
この関数を呼び出すときに現在接続されているデータベースで有効なサービス・サブクラス名を指定する、タイプ VARCHAR(128) の入力引数。引数が NULL または空ストリングである場合、データベース内のすべてのサブクラスについてメトリックが取得されます。

member
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なメンバーを指定する、タイプ INTEGER の入力引数。現行のデータベース・メンバーには -1、すべてのデータベース・メンバーには -2 を指定します。NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

MON_GET_SERVICE_SUBCLASS_DETAILS 関数に対する EXECUTE 特権。

例

サービス・スーパークラスごとに使用された合計 CPU 時間、および処理された要求の合計数を CPU 使用時間順に配列し、リレーショナル形式で表示します (XMLTABLE を使用)。

```
SELECT varchar(scmetrics.service_superclass_name,30) as service_superclass,
       sum(detmetrics.total_cpu_time) as total_cpu,
       sum(detmetrics.app_rqsts_completed_total) as total_rqsts
FROM TABLE(MON_GET_SERVICE_SUBCLASS_DETAILS('','',-2)) AS SCMETRICS,
XMLTABLE (XMLNAMESPACES( DEFAULT 'http://www.ibm.com/xmlns/prod/db2/mon'),
        '$detmetric/db2_service_subclass'
        PASSING XMLPARSE(DOCUMENT SCMETRICS.DETAILS)
        as "detmetric"
COLUMNS "TOTAL_CPU_TIME" INTEGER PATH 'system_metrics/total_cpu_time',
        "APP_RQSTS_COMPLETED_TOTAL" INTEGER PATH
        'system_metrics/app_rqsts_completed_total')
        AS DETMETRICS
GROUP BY service_superclass_name
ORDER BY total_cpu desc
```

以下はこの照会の出力例です。

```
SERVICE_SUPERCLASS             TOTAL_CPU            TOTAL_RQSTS
------------------------------ -------------------- --------------------
SYSDEFAULTUSERCLASS                         2428188                   26
SYSDEFAULTMAINTENANCECLASS                        0                    0
SYSDEFAULTSYSTEMCLASS                             0                    0

  3 record(s) selected.
```

使用上の注意

MON_GET_SERVICE_SUBCLASS_DETAILS 表関数で戻されるメトリックは、指示されたサービス・サブクラスの下で実行された要求についてのすべてのメトリックの累計を表します。この関数は、以下の点で MON_GET_SERVICE_SUBCLASS 表関数と似ています。

- MON_GET_SERVICE_SUBCLASS 表関数は、列ベース形式の最も一般的に使用されるメトリックを戻し、メトリックを取得する際に最もパフォーマンス効率が高い方法です。
- MON_GET_SERVICE_SUBCLASS_DETAILS 表関数は、使用可能なすべてのメトリック一式を XML 文書形式で戻すため、出力フォーマットを最大限柔軟性のあるものにします。XML ベースの出力は、XML パーサーで直接解析でき、XMLTABLE 関数でリレーショナル形式に変換することもできます (例を参照)。

メトリックは、作業単位境界でサービス・クラスにロールアップされ、要求の実行中には定期的にロールアップされます。したがって、この表関数で報告される値は、直前の ROLLUP 時のシステムの現行状態を反映しています。メトリックの値は確実に増加します。ある時間間隔に対する指定されたメトリックの値を判別するには、 MON_GET_SERVICE_SUBCLASS_DETAILS 表関数を使用してその間隔の始めと終わりのメトリックを照会し、差異を計算します。

要求メトリックは、サービス・スーパークラスに対する COLLECT REQUEST METRICS 節、およびデータベース・レベルの **mon_req_metrics** データベース構成パラメーターを介して制御されます。親サービス・スーパークラスで要求メトリックを使用可能にしているサービス・サブクラスのエージェントが要求を処理する場合にのみ、またはデータベース全体で要求メトリック・コレクションが有効な場合にのみ、その要求に対しメトリックが収集されます。デフォルトでは、要求メトリックはデータベース・レベルで使用可能です。要求メトリックがデータベース・レベルでもサービス・スーパークラスに対しても使用不可になっている場合、そのサービス・スーパークラスにマッピングされた接続ごとに報告されるメトリックは増加を停止します (または、要求メトリックがデータベースのアクティブ化時に無効であった場合には 0 のままになります)。

MON_GET_SERVICE_SUBCLASS_DETAILS 表関数は、サービス・サブクラスごとおよびメンバーごとに 1 行のデータを戻します。 (メンバー上の) サービス・クラス全体または (1 つ以上のサービス・クラスの) メンバー全体の集約は実行されません。しかし、集約は例に示されるように SQL 照会を使用して実行できます (例を参照)。この入力パラメーターの影響として、ANDing されます。したがって、競合する入力パラメーター (例えば、スーパークラス名 SUPA と SUPA のサブクラスではないサブクラス名 SUBB など) を指定する場合、行は戻されません。

ヒント: 要求は、複数のサービス・サブクラスで実行される場合があります。例えば、REMAP ACTIVITY アクションによってワークロード・マネージャー (WLM) しきい値を使用して要求を 1 つのサービス・サブクラスから別のサービス・サブクラスへマッピングする場合、この状況が生じる可能性があります。メトリックに使用された時間は、要求が実行されるサービス・サブクラスごとに更新されますが、要求カウンターは、要求が完了したサービス・サブクラスに対して増分します。したがって、単一のサブクラスの要求時間の平均を分析する必要はありません。アクティビティーをマッピングできるすべてのサブクラスは相互に関連付けて分析する

必要があります。例えば、サービス・サブクラス A からサービス・クラス B にアクティビティーをマッピングできるしきい値があり、要求の平均を計算する場合、サービス・サブサービス A および B についてのカウンターおよびメトリックを集約し、その集約を使用して平均を計算する必要があります。

DETAILS 列に戻される XML 文書のスキーマは、ファイル sqllib/misc/DB2MonRoutines.xsd で入手できます。詳細は、ファイル sqllib/misc/DB2MonCommon.xsd 内にあります。

戻される情報

表 132. MON_GET_SERVICE_SUBCLASS_DETAILS について戻される情報

列名	データ・タイプ	説明
SERVICE_SUPERCLASS_NAME	VARCHAR(128)	service_superclass_name - サービス・スーパークラス名
SERVICE_SUBCLASS_NAME	VARCHAR(128)	service_subclass_name - サービス・サブクラス名
SERVICE_CLASS_ID	INTEGER	service_class_id - サービス・クラス ID
MEMBER	SMALLINT	member - データベース・メンバー
DETAILS	BLOB(1M)	サービス・クラスについての詳細メトリックを含む XML 文書。エレメントの説明については、本書の表 133を参照してください。

以下の例は、DETAILS 列で戻される XML 文書の構造を示しています。

```
<db2_service_subclass xmlns="http://www.ibm.com/xmlns/prod/db2/mon" release="90700000">
   <service_superclass_name>SYSDEFAULTSYSTEMCLASS</service_superclass_name>
   <service_subclass_name>SYSDEFAULTSUBCLASS</service_subclass_name>
   <service_subclass_id>11</service_subclass_id>
   <member>0</member>
   <system_metrics release="90700000">
   <act_aborted_total>5</act_aborted_total>
    ...
   <wlm_queue_assignments_total>3</wlm_queue_assignments_total>
   </system_metrics>
</db2_service_subclass>
```

完全スキーマについては、sqllib/misc/DB2MonRoutines.xsd を参照してください。

表 133. MON_GET_SERVICE_SUBCLASS_DETAILS について戻される詳細メトリック

エレメント名	データ・タイプ	説明または対応するモニター・エレメント
service_superclass_name	xs:string(128)	service_superclass_name - サービス・スーパークラス名
service_subclass_name	xs:string(128)	service_subclass_name - サービス・サブクラス名
service_class_id	xs:nonNegativeInteger	service_class_id - サービス・クラス ID
member	xs:nonNegativeInteger	member - データベース・メンバー
act_aborted_total	xs:nonNegativeInteger	act_aborted_total - 打ち切られたアクティビティーの合計数
act_completed_total	xs:nonNegativeInteger	act_completed_total - 完了したアクティビティーの合計数
act_rejected_total	xs:nonNegativeInteger	act_rejected_total - リジェクトされたアクティビティーの合計数
act_rqsts_total	xs:nonNegativeInteger	act_rqsts_total - アクティビティー要求の合計
agent_wait_time	xs:nonNegativeInteger	agent_wait_time - エージェント待機時間
agent_waits_total	xs:nonNegativeInteger	agent_waits_total - エージェント待機の合計

表 133. MON_GET_SERVICE_SUBCLASS_DETAILS について戻される詳細メトリック (続き)

エレメント名	データ・タイプ	説明または対応するモニター・エレメント
app_rqsts_completed_total	xs:nonNegativeInteger	app_rqsts_completed_total - 完了したアプリケーション要求の合計数
audit_events_total	xs:nonNegativeInteger	audit_events_total - 監査イベント合計数
audit_subsystem_wait_time	xs:nonNegativeInteger	audit_subsystem_wait_time - 監査サブシステム待機時間
audit_subsystem_waits_total	xs:nonNegativeInteger	audit_subsystem_waits_total - 監査サブシステム待機の合計
audit_file_write_wait_time	xs:nonNegativeInteger	audit_file_write_wait_time - 監査ファイル書き込み待機時間
audit_file_writes_total	xs:nonNegativeInteger	audit_file_writes_total - 書き込まれた監査ファイルの合計数
cat_cache_inserts	xs:nonNegativeInteger	cat_cache_inserts - カタログ・キャッシュ挿入数
cat_cache_lookups	xs:nonNegativeInteger	cat_cache_lookups - カタログ・キャッシュ参照数
client_idle_wait_time	xs:nonNegativeInteger	client_idle_wait_time - クライアントのアイドル待機時間
deadlocks	xs:nonNegativeInteger	deadlocks - デッドロック検出数
diaglog_writes_total	xs:nonNegativeInteger	diaglog_writes_total - 診断ログ・ファイル書き込みの合計
diaglog_write_wait_time	xs:nonNegativeInteger	diaglog_write_wait_time - 診断ログ・ファイルの書き込み待機時間
direct_read_time	xs:nonNegativeInteger	direct_read_time - 直接読み取り時間
direct_write_time	xs:nonNegativeInteger	direct_write_time - 直接書き込み時間
direct_read_reqs	xs:nonNegativeInteger	direct_read_reqs - 直接読み取り要求
direct_reads	xs:nonNegativeInteger	direct_reads - データベースからの直接読み取り
direct_write_reqs	xs:nonNegativeInteger	direct_write_reqs - 直接書き込み要求
direct_writes	xs:nonNegativeInteger	direct_writes - データベースへの直接書き込み
fcm_recv_volume	xs:nonNegativeInteger	fcm_recv_volume - FCM 受信ボリューム
fcm_recv_wait_time	xs:nonNegativeInteger	fcm_recv_wait_time - FCM 受信待機時間
fcm_recvs_total	xs:nonNegativeInteger	fcm_recvs_total - FCM 合計受信数
fcm_message_recv_volume	xs:nonNegativeInteger	fcm_message_recv_volume - FCM メッセージ受信ボリューム
fcm_message_recvs_total	xs:nonNegativeInteger	fcm_message_recvs_total - FCM メッセージ合計受信数
fcm_message_recv_wait_time	xs:nonNegativeInteger	fcm_message_recv_wait_time - FCM メッセージの受信待機時間
fcm_message_send_volume	xs:nonNegativeInteger	fcm_message_send_volume - FCM メッセージ送信ボリューム
fcm_message_send_wait_time	xs:nonNegativeInteger	fcm_message_send_wait_time - FCM メッセージ送信待ち時間
fcm_message_sends_total	xs:nonNegativeInteger	fcm_message_sends_total - FCM メッセージ合計送信数
fcm_send_volume	xs:nonNegativeInteger	fcm_send_volume - FCM 送信ボリューム
fcm_send_wait_time	xs:nonNegativeInteger	fcm_send_wait_time - FCM 送信待ち時間
fcm_sends_total	xs:nonNegativeInteger	fcm_sends_total - FCM 合計送信数
fcm_tq_recv_wait_time	xs:nonNegativeInteger	fcm_tq_recv_wait_time - FCM 表キューの受信待ち時間
fcm_tq_send_wait_time	xs:nonNegativeInteger	fcm_tq_send_wait_time - FCM 表キューの送信待ち時間
fcm_tq_recv_volume	xs:nonNegativeInteger	fcm_tq_recv_volume - FCM 表キューの受信ボリューム
fcm_tq_recvs_total	xs:nonNegativeInteger	fcm_tq_recvs_total - FCM 表キューの合計受信数
fcm_tq_send_volume	xs:nonNegativeInteger	fcm_tq_send_volume - FCM 表キューの送信ボリューム

表 133. MON_GET_SERVICE_SUBCLASS_DETAILS について戻される詳細メトリック (続き)

エレメント名	データ・タイプ	説明または対応するモニター・エレメント
fcm_tq_sends_total	xs:nonNegativeInteger	fcm_tq_sends_total - FCM 表キューの合計送信数
int_commits	xs:nonNegativeInteger	int_commits - 内部コミット数
int_rollbacks	xs:nonNegativeInteger	int_rollbacks - 内部ロールバック数
tq_tot_send_spills	xs:nonNegativeInteger	tq_tot_send_spills - オーバーフローした表キュー・バッファーの合計数
ipc_recv_volume	xs:nonNegativeInteger	ipc_recv_volume - プロセス間通信受信ボリューム
ipc_recv_wait_time	xs:nonNegativeInteger	ipc_recv_wait_time - プロセス間通信受信待ち時間
ipc_recvs_total	xs:nonNegativeInteger	ipc_recvs_total - プロセス間通信合計受信数
ipc_send_volume	xs:nonNegativeInteger	ipc_send_volume - プロセス間通信送信ボリューム
ipc_send_wait_time	xs:nonNegativeInteger	ipc_send_wait_time - プロセス間通信送信待ち時間
ipc_sends_total	xs:nonNegativeInteger	ipc_sends_total - プロセス間通信合計送信数
lock_escals	xs:nonNegativeInteger	lock_escals - ロック・エスカレーション数
lock_timeouts	xs:nonNegativeInteger	lock_timeouts - ロック・タイムアウト数
lock_wait_time	xs:nonNegativeInteger	lock_wait_time - ロック待機中の時間
lock_waits	xs:nonNegativeInteger	lock_waits - ロック待機数
log_buffer_wait_time	xs:nonNegativeInteger	log_buffer_wait_time - ログ・バッファー待ち時間
log_disk_wait_time	xs:nonNegativeInteger	log_disk_wait_time - ログ・ディスク待機時間
log_disk_waits_total	xs:nonNegativeInteger	log_disk_waits_total - ログ・ディスク待機の合計
num_lw_thresh_exceeded	xs:nonNegativeInteger	num_lw_thresh_exceeded - しきい値を超えた回数
pkg_cache_inserts	xs:nonNegativeInteger	pkg_cache_inserts - パッケージ・キャッシュ挿入
pkg_cache_lookups	xs:nonNegativeInteger	pkg_cache_lookups - パッケージ・キャッシュ参照
pool_data_l_reads	xs:nonNegativeInteger	pool_data_l_reads - バッファー・プール・データの論理読み取り
pool_data_p_reads	xs:nonNegativeInteger	pool_data_p_reads - バッファー・プール・データの物理読み取り
pool_data_writes	xs:nonNegativeInteger	pool_data_writes - バッファー・プールへのデータの書き込み
pool_index_l_reads	xs:nonNegativeInteger	pool_index_l_reads - バッファー・プール索引の論理読み取り
pool_index_p_reads	xs:nonNegativeInteger	pool_index_p_reads - バッファー・プール索引の物理読み取り
pool_index_writes	xs:nonNegativeInteger	pool_index_writes - バッファー・プール索引の書き込み
pool_read_time	xs:nonNegativeInteger	pool_read_time - バッファー・プール物理読み取り時間の合計
pool_temp_data_l_reads	xs:nonNegativeInteger	pool_temp_data_l_reads - バッファー・プール一時データの論理読み取り
pool_temp_data_p_reads	xs:nonNegativeInteger	pool_temp_data_p_reads - バッファー・プール一時データの物理読み取り
pool_temp_index_l_reads	xs:nonNegativeInteger	pool_temp_index_l_reads - バッファー・プール一時索引の論理読み取り
pool_temp_index_p_reads	xs:nonNegativeInteger	pool_temp_index_p_reads - バッファー・プール一時索引の物理読み取り

表 133. MON_GET_SERVICE_SUBCLASS_DETAILS について戻される詳細メトリック (続き)

エレメント名	データ・タイプ	説明または対応するモニター・エレメント
pool_temp_xda_l_reads	xs:nonNegativeInteger	pool_temp_xda_l_reads - バッファー・プール一時 XDA データの論理読み取り
pool_temp_xda_p_reads	xs:nonNegativeInteger	pool_temp_xda_p_reads - バッファー・プール一時 XDA データの物理読み取り
pool_write_time	xs:nonNegativeInteger	pool_write_time - バッファー・プール物理書き込み時間の合計
pool_xda_l_reads	xs:nonNegativeInteger	pool_xda_l_reads - バッファー・プール XDA データの論理読み取り
pool_xda_p_reads	xs:nonNegativeInteger	pool_xda_p_reads - バッファー・プール XDA データの物理読み取り
pool_xda_writes	xs:nonNegativeInteger	pool_xda_writes - バッファー・プール XDA データの書き込み
num_log_buffer_full	xs:nonNegativeInteger	num_log_buffer_full - フル・ログ・バッファーの回数
rqsts_completed_total	xs:nonNegativeInteger	rqsts_completed_total - 完了した要求の合計数
total_rqst_mapped_in	xs:nonNegativeInteger	total_rqst_mapped_in - マップで含められた要求の合計
total_rqst_mapped_out	xs:nonNegativeInteger	total_rqst_mapped_out - マップで除外された要求の合計
rows_modified	xs:nonNegativeInteger	rows_modified - 変更された行数
rows_read	xs:nonNegativeInteger	rows_read - 読み取り行数
rows_returned	xs:nonNegativeInteger	rows_returned - 戻り行数
tcpip_recv_volume	xs:nonNegativeInteger	tcpip_recv_volume - TCP/IP 受信ボリューム
tcpip_recv_wait_time	xs:nonNegativeInteger	tcpip_recv_wait_time - TCP/IP 受信待ち時間
tcpip_recvs_total	xs:nonNegativeInteger	tcpip_recvs_total - TCP/IP 合計受信数
tcpip_send_volume	xs:nonNegativeInteger	tcpip_send_volume - TCP/IP 送信ボリューム
tcpip_send_wait_time	xs:nonNegativeInteger	tcpip_send_wait_time - TCP/IP 送信待ち時間
tcpip_sends_total	xs:nonNegativeInteger	tcpip_sends_total - TCP/IP 合計送信数
thresh_violations	xs:nonNegativeInteger	thresh_violations - しきい値違反の回数
total_act_time	xs:nonNegativeInteger	total_act_time - 合計アクティビティー時間
total_act_wait_time	xs:nonNegativeInteger	total_act_wait_time - 合計アクティビティー待機時間
total_app_commits	xs:nonNegativeInteger	total_app_commits - アプリケーション・コミットの合計回数
total_app_rollbacks	xs:nonNegativeInteger	total_app_rollbacks - アプリケーション・ロールバックの合計回数
total_app_rqst_time	xs:nonNegativeInteger	total_app_rqst_time - アプリケーション要求合計時間
total_app_section_executions	xs:nonNegativeInteger	total_app_section_executions - セクション実行の合計回数
total_commit_proc_time	xs:nonNegativeInteger	total_commit_proc_time - コミット処理時間の合計
total_commit_time	xs:nonNegativeInteger	total_commit_time - コミット時間の合計
total_compilations	xs:nonNegativeInteger	total_compilations - コンパイルの合計回数
total_compile_proc_time	xs:nonNegativeInteger	total_compile_proc_time - コンパイル処理時間の合計
total_compile_time	xs:nonNegativeInteger	total_compile_time - コンパイル時間の合計
total_cpu_time	xs:nonNegativeInteger	total_cpu_time - 合計 CPU 時間
total_implicit_compilations	xs:nonNegativeInteger	total_implicit_compilations - 暗黙的コンパイルの合計回数

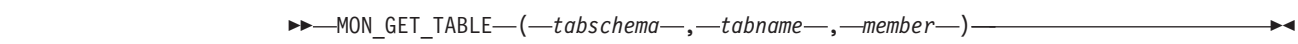
表 133. MON_GET_SERVICE_SUBCLASS_DETAILS について戻される詳細メトリック (続き)

エレメント名	データ・タイプ	説明または対応するモニター・エレメント
total_implicit_compile _proc_time	xs:nonNegativeInteger	total_implicit_compile_proc_time - 暗黙的コンパイルの処理時間の合計
total_implicit_compile_time	xs:nonNegativeInteger	total_implicit_compile_time - 暗黙的コンパイル時間の合計
total_loads	xs:nonNegativeInteger	total_loads - ロードの合計回数
total_load_proc_time	xs:nonNegativeInteger	total_load_proc_time - ロード処理時間の合計
total_load_time	xs:nonNegativeInteger	total_load_time - ロード時間の合計
total_reorgs	xs:nonNegativeInteger	total_reorgs - 再編成の合計回数
total_reorg_proc_time	xs:nonNegativeInteger	total_reorg_proc_time - 再編成の処理時間の合計
total_reorg_time	xs:nonNegativeInteger	total_reorg_time - 再編成時間の合計
total_rollback_proc_time	xs:nonNegativeInteger	total_rollback_proc_time - ロールバック処理時間の合計
total_rollback_time	xs:nonNegativeInteger	total_rollback_time - ロールバック時間の合計
total_routine_invocations	xs:nonNegativeInteger	total_routine_invocations - ルーチン呼び出しの合計回数
total_routine_time	xs:nonNegativeInteger	total_routine_time - ルーチン時間の合計
total_routine_user_ code_proc_time	xs:nonNegativeInteger	total_routine_user_code_proc_time - ルーチン・ユーザー・コード処理時間の合計
total_routine_user_ code_time	xs:nonNegativeInteger	total_routine_user_code_time - ルーチン・ユーザー・コード時間の合計
total_rqst_time	xs:nonNegativeInteger	total_rqst_time - 合計要求時間
total_runstats	xs:nonNegativeInteger	total_runstats - ランタイム統計の合計回数
total_runstats_proc_time	xs:nonNegativeInteger	total_runstats_proc_time - ランタイム統計の処理時間の合計
total_runstats_time	xs:nonNegativeInteger	total_runstats_time - ランタイム統計時間の合計
total_section_proc_time	xs:nonNegativeInteger	total_section_proc_time - セクション処理時間の合計
total_section_sort_time	xs:nonNegativeInteger	total_section_sort_time - セクション・ソート時間合計
total_section_sort_proc_time	xs:nonNegativeInteger	total_section_sort_proc_time - セクション・ソート処理時間合計
total_section_sorts	xs:nonNegativeInteger	total_section_sorts - セクション・ソート合計
total_section_time	xs:nonNegativeInteger	total_section_time - セクション時間の合計
total_sorts	xs:nonNegativeInteger	total_sorts - ソート合計
post_threshold_sorts	xs:nonNegativeInteger	post_shrthreshold_sorts - ポスト共有しきい値ソート
post_shrthreshold_sorts	xs:nonNegativeInteger	post_shrthreshold_sorts - ポスト共有しきい値ソート
sort_overflows	xs:nonNegativeInteger	sort_overflows - ソート・オーバーフロー
tq_tot_send_spills	xs:nonNegativeInteger	tq_tot_send_spills - オーバーフローした表キュー・バッファーの合計数
total_wait_time	xs:nonNegativeInteger	total_wait_time - 合計待ち時間
wlm_queue_time_total	xs:nonNegativeInteger	wlm_queue_time_total - ワークロード・マネージャー合計キュー時間
wlm_queue_assignments_total	xs:nonNegativeInteger	wlm_queue_assignments_total - ワークロード・マネージャー合計キュー割り当て数

MON_GET_TABLE 表関数 - 表メトリックの取得

MON_GET_TABLE 表関数は、1 つ以上の表のモニター・メトリックを戻します。

構文



スキーマは SYSPROC です。

表関数パラメーター

tabschema
この関数を呼び出すときに現在接続されているデータベースで有効な表スキーマ名を指定する、タイプ VARCHAR(128) の入力引数。引数が NULL または空ストリングである場合、データベース内のすべてのスキーマのすべての表についてメトリックが取得されます。引数が指定される場合、メトリックは指定したスキーマ内の表についてのみ戻されます。

tabname
この関数を呼び出すときに現在接続されているデータベースで有効な表名を指定する、タイプ VARCHAR(128) の入力引数。引数が NULL または空ストリングである場合、データベース内のすべての表についてメトリックが取得されます。

member
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なメンバーを指定する、タイプ INTEGER の入力引数。現行のデータベース・メンバーには -1、すべてのデータベース・メンバーには -2 を指定します。NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

MON_GET_TABLE 関数に対する EXECUTE 特権。

例

すべてのデータベース・メンバー全体で集約し、読み取り数の高い順に、データベースのアクティブ化後にアクセスされたすべての表のアクティビティーをリストします。

```
SELECT varchar(tabschema,20) as tabschema,
       varchar(tabname,20) as tabname,
       sum(rows_read) as total_rows_read,
       sum(rows_inserted) as total_rows_inserted,
       sum(rows_updated) as total_rows_updated,
       sum(rows_deleted) as total_rows_deleted
FROM TABLE(MON_GET_TABLE('','',-2)) AS t
GROUP BY tabschema, tabname
ORDER BY total_rows_read DESC
```

以下はこの照会の出力例です。

```
TABSCHEMA            TABNAME              TOTAL_ROWS_READ      ...
-------------------- -------------------- -------------------- ...
SYSIBM               SYSHISTO                             113 ...
SYSIBM               SYSWORKL                              22 ...
SYSIBM               SYSROUTI                              13 ...
```

```
SYSIBM              SYSSERVI                              13 ...
SYSIBM              SYSTHRES                               6 ...
SYSIBM              SYSTABLE                               3 ...
SYSIBM              SYSCONTE                               2 ...
SYSIBM              SYSDBAUT                               2 ...
SYSIBM              SYSEVENT                               2 ...
SYSIBM              SYSPLAN                                1 ...
SYSIBM              SYSSURRO                               1 ...
SYSIBM              SYSVERSI                               1 ...
SYSIBM              SYSXMLST                               1 ...
SYSIBM              SYSAUDIT                               0 ...
SYSIBM              SYSROLEA                               0 ...
SYSIBM              SYSROLES                               0 ...
SYSIBM              SYSTASKS                               0 ...
SYSIBM              SYSWORKA                               0 ...
SYSIBM              SYSXMLPA                               0 ...

  19 record(s) selected.
```

照会の出力 (続き)。

```
... TOTAL_ROWS_INSERTED  TOTAL_ROWS_UPDATED   TOTAL_ROWS_DELETED
... -------------------- -------------------- --------------------
...                    0                    0                    0
...                    0                    0                    0
...                    0                    0                    0
...                    0                    0                    0
...                    0                    0                    0
...                    0                    0                    0
...                    0                    0                    0
...                    0                    0                    0
...                    0                    0                    0
...                    0                    0                    0
...                    0                    0                    0
...                    0                    0                    0
...                    0                    0                    0
...                    0                    0                    0
...                    0                    0                    0
...                    0                    0                    0
...                    0                    0                    0
...                    0                    0                    0
...                    0                    0                    0
...                    0                    0                    0
```

使用上の注意

MON_GET_TABLE 表関数は、データベース表ごとおよびデータベース・メンバーごとに 1 行のデータを戻します。範囲パーティション表が使用される場合、データベース・メンバーごとに、各表パーティションについて 1 行が戻されます。データベース・メンバー全体からの集約は実行されません。ただし、集約は例に示されるように SQL 照会を使用して実行できます。

メトリックは、データベースのアクティブ化後にアクセスされた表についてのみ戻されます。

メトリックは常に使用可能です。この関数を使用して表メトリックにアクセスするために、システム・モニター・スイッチをオンにする必要はありません。

戻される情報

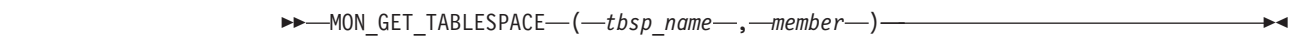
表 134. MON_GET_TABLE について戻される情報

列名	データ・タイプ	説明
TABSCHEMA	VARCHAR(128)	table_schema - 表スキーマ名
TABNAME	VARCHAR(128)	table_name - 表名
MEMBER	SMALLINT	member - データベース・メンバー
TAB_TYPE	VARCHAR(14)	table_type - 表タイプ。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • USER_TABLE • DROPPED_TABLE • TEMP_TABLE • CATALOG_TABLE • REORG_TABLE
TAB_FILE_ID	BIGINT	table_file_id - 表ファイル ID
DATA_PARTITION_ID	INTEGER	data_partition_id - データ・パーティション ID
TBSP_ID	BIGINT	tablespace_id - 表スペース ID
INDEX_TBSP_ID	BIGINT	index_tbsp_id - 索引表スペース ID
LONG_TBSP_ID	BIGINT	long_tbsp_id - 長い表スペース ID
TABLE_SCANS	BIGINT	table_scans - 表スキャン
ROWS_READ	BIGINT	rows_read - 読み取り行数
ROWS_INSERTED	BIGINT	rows_inserted - 挿入行数
ROWS_UPDATED	BIGINT	rows_updated - 更新行数
ROWS_DELETED	BIGINT	rows_deleted - 削除行数
OVERFLOW_ACCESSES	BIGINT	overflow_accesses - オーバーフロー・レコードへのアクセス
OVERFLOW_CREATES	BIGINT	overflow_creates - オーバーフローの作成
PAGE_REORGS	BIGINT	page_reorgs - ページ再編成
ADDITIONAL_DETAILS	BLOB(100K)	将来の利用のために予約済み。

MON_GET_TABLESPACE 表関数 - 表スペース・メトリックの取得

MON_GET_TABLESPACE 表関数は、1 つ以上の表スペースのモニター・メトリックを戻します。

構文



スキーマは SYSPROC です。

表関数パラメーター

tbsp_name
この関数を呼び出すときに現在接続されているデータベースで有効な表スペース名を指定する、タイプ VARCHAR(128) の入力引数。引数が NULL または空ストリングである場合、データベース内のすべての表スペースについてメトリックが取得されます。

member
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なメンバーを指定する、タイプ INTEGER の入力引数。現行のデータベース・メンバーには -1、すべてのデータベース・メンバーには -2 を指定します。NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

MON_GET_TABLESPACE 関数に対する EXECUTE 特権。

例

表スペース・コンテナーからの物理的な読み取りの数で順序付けられた表スペースをリストします。

```
SELECT varchar(tbsp_name, 30) as tbsp_name,
                  member,
                  tbsp_type,
                  pool_data_p_reads
FROM TABLE(MON_GET_TABLESPACE('',-2)) AS t
ORDER BY pool_data_p_reads DESC
```

以下はこの照会の出力例です。

```
TBSP_NAME                      MEMBER TBSP_TYPE  POOL_DATA_P_READS
------------------------------ ------ ---------- --------------------
SYSCATSPACE                         0 DMS                          79
USERSPACE1                          0 DMS                          34
TEMPSPACE1                          0 SMS                           0

  3 record(s) selected.
```

使用上の注意

MON_GET_TABLESPACE 表関数は、データベース表スペースごとおよびデータベース・メンバーごとに 1 行のデータを戻します。データベース・メンバー全体からの集約は実行されません。しかし、集約は SQL 照会を使用して実行できます。

この関数によって収集されるメトリックは、mon_obj_metrics 構成パラメーターを使用してデータベース・レベルで制御されます。デフォルトでは、メトリック収集は有効になります。

戻される情報

表 135. MON_GET_TABLESPACE について戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
TBSP_NAME	VARCHAR(128)	tablespace_name - 表スペース名
TBSP_ID	BIGINT	tablespace_id - 表スペース ID

表 135. MON_GET_TABLESPACE について戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
MEMBER	SMALLINT	member - データベース・メンバー
TBSP_TYPE	VARCHAR(10)	tablespace_type - 表スペース・タイプ。このインターフェースは、sqlutil.h の定義に基づくテキスト ID を戻します。これは次のいずれかです。 • DMS • SMS
TBSP_CONTENT_TYPE	VARCHAR(10)	tablespace_content_type - 表スペースのコンテンツ・タイプ。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • ANY • LARGE • SYSTEMP • USRTEMP
TBSP_PAGE_SIZE	BIGINT	tablespace_page_size - 表スペースのページ・サイズ
TBSP_EXTENT_SIZE	BIGINT	tablespace_extent_size - 表スペースのエクステント・サイズ
TBSP_PREFETCH_SIZE	BIGINT	tablespace_prefetch_size - 表スペースのプリフェッチ・サイズ
TBSP_CUR_POOL_ID	BIGINT	tablespace_cur_pool_id - 現在使用中のバッファー・プール
TBSP_NEXT_POOL_ID	BIGINT	tablespace_next_pool_id - 次の始動時に使用されるバッファー・プール
FS_CACHING	SMALLINT	fs_caching - ファイル・システム・キャッシング
TBSP_REBALANCER_MODE	VARCHAR(30)	tablespace_rebalancer_mode - リバランサー・モード。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • NO_REBAL • FWD_REBAL • REV_REBAL • FWD_REBAL_OF_2PASS • REV_REBAL_OF_2PASS
TBSP_USING_AUTO_STORAGE	SMALLINT	tablespace_using_auto_storage - 自動ストレージが使用可能な表スペース
TBSP_AUTO_RESIZE_ENABLED	SMALLINT	tablespace_auto_resize_enabled - 表スペースの自動サイズ変更可能
DIRECT_READS	BIGINT	direct_reads - データベースからの直接読み取り
DIRECT_READ_REQS	BIGINT	direct_read_reqs - 直接読み取り要求
DIRECT_WRITES	BIGINT	direct_writes - データベースへの直接書き込み
DIRECT_WRITE_REQS	BIGINT	direct_write_reqs - 直接書き込み要求
POOL_DATA_L_READS	BIGINT	pool_data_l_reads - バッファー・プール・データの論理読み取り

表 135. MON_GET_TABLESPACE について戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
POOL_TEMP_DATA_L_READS	BIGINT	pool_temp_data_l_reads - バッファー・プール一時データの論理読み取り
POOL_XDA_L_READS	BIGINT	pool_xda_l_reads - バッファー・プール XDA データの論理読み取り
POOL_TEMP_XDA_L_READS	BIGINT	pool_temp_xda_l_reads - バッファー・プール一時 XDA データの論理読み取り
POOL_INDEX_L_READS	BIGINT	pool_index_l_reads - バッファー・プール索引の論理読み取り
POOL_TEMP_INDEX_L_READS	BIGINT	pool_temp_index_l_reads - バッファー・プール一時索引の論理読み取り
POOL_DATA_P_READS	BIGINT	pool_data_p_reads - バッファー・プール・データの物理読み取り
POOL_TEMP_DATA_P_READS	BIGINT	pool_temp_data_p_reads - バッファー・プール一時データの物理読み取り
POOL_XDA_P_READS	BIGINT	pool_xda_p_reads - バッファー・プール XDA データの物理読み取り
POOL_TEMP_XDA_P_READS	BIGINT	pool_temp_xda_p_reads - バッファー・プール一時 XDA データの物理読み取り
POOL_INDEX_P_READS	BIGINT	pool_index_p_reads - バッファー・プール索引の物理読み取り
POOL_TEMP_INDEX_P_READS	BIGINT	pool_temp_index_p_reads - バッファー・プール一時索引の物理読み取り
POOL_DATA_WRITES	BIGINT	pool_data_writes - バッファー・プールへのデータの書き込み
POOL_XDA_WRITES	BIGINT	pool_xda_writes - バッファー・プール XDA データの書き込み
POOL_INDEX_WRITES	BIGINT	pool_index_writes - バッファー・プール索引の書き込み
DIRECT_READ_TIME	BIGINT	direct_read_time - 直接読み取り時間
DIRECT_WRITE_TIME	BIGINT	direct_write_time - 直接書き込み時間
POOL_READ_TIME	BIGINT	pool_read_time - バッファー・プール物理読み取り時間の合計
POOL_WRITE_TIME	BIGINT	pool_write_time - バッファー・プール物理書き込み時間の合計
POOL_ASYNC_DATA_READS	BIGINT	pool_async_data_reads - バッファー・プール非同期データ読み取り
POOL_ASYNC_DATA_READ_REQS	BIGINT	pool_async_data_read_reqs - バッファー・プール非同期読み取り要求
POOL_ASYNC_DATA_WRITES	BIGINT	pool_async_data_writes - バッファー・プール非同期データ書き込み
POOL_ASYNC_INDEX_READS	BIGINT	pool_async_index_reads - バッファー・プール非同期索引読み取り
POOL_ASYNC_INDEX_READ_REQS	BIGINT	pool_async_index_read_reqs - バッファー・プール非同期索引読み取り要求

表 135. MON_GET_TABLESPACE について戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
POOL_ASYNC_INDEX_WRITES	BIGINT	pool_async_index_writes - バッファー・プール非同期索引書き込み
POOL_ASYNC_XDA_READS	BIGINT	pool_async_xda_reads - バッファー・プール非同期 XDA データ読み取り
POOL_ASYNC_XDA_READ_REQS	BIGINT	pool_async_xda_read_reqs - バッファー・プール非同期 XDA 読み取り要求
POOL_ASYNC_XDA_WRITES	BIGINT	pool_async_xda_writes - バッファー・プール非同期 XDA データ書き込み
VECTORED_IOS	BIGINT	vectored_ios - ベクトル化入出力要求数
PAGES_FROM_VECTORED_IOS	BIGINT	pages_from_vectored_ios - ベクトル化入出力によって読み取られたページ数の合計
BLOCK_IOS	BIGINT	block_ios - ブロック入出力要求数
PAGES_FROM_BLOCK_IOS	BIGINT	pages_from_block_ios - ブロック入出力によって読み取られたページ数の合計
UNREAD_PREFETCH_PAGES	BIGINT	unread_prefetch_pages - 読み取り不能プリフェッチ・ページ
FILES_CLOSED	BIGINT	files_closed - 閉じられたデータベース・ファイル
TBSP_STATE	VARCHAR(256)	tablespace_state - 表スペースの状態
TBSP_USED_PAGES	BIGINT	tablespace_used_pages - 表スペース内の使用されているページ数
TBSP_FREE_PAGES	BIGINT	tablespace_free_pages - 表スペース内のフリー・ページ数
TBSP_USABLE_PAGES	BIGINT	tablespace_usable_pages - 表スペース内の使用可能ページ数
TBSP_TOTAL_PAGES	BIGINT	tablespace_total_pages - 表スペース内の合計ページ数
TBSP_PENDING_FREE_PAGES	BIGINT	tablespace_pending_free_pages - 表スペース内のペンディング・フリー・ページ数
TBSP_PAGE_TOP	BIGINT	tablespace_page_top - 表スペース最高水準点
TBSP_MAX_PAGE_TOP	BIGINT	tbsp_max_page_top - 最大表スペース・ページの最高水準点
RECLAIMABLE_SPACE_ENABLED	SMALLINT	reclaimable_space_enabled - 有効な利用可能スペース標識
AUTO_STORAGE_HYBRID	SMALLINT	auto_storage_hybrid - ハイブリッド自動ストレージ表スペース標識
TBSP_PATHS_DROPPED	SMALLINT	tablespace_paths_dropped - ドロップされるパスを使用する表スペース
ADDITIONAL_DETAILS	BLOB(100K)	将来の利用のために予約済み。

MON_GET_UNIT_OF_WORK 表関数 - 作業単位メトリックの取得

MON_GET_UNIT_OF_WORK 表関数は、1 つ以上の作業単位のメトリックを戻します。

構文

スキーマは SYSPROC です。

表関数パラメーター

application_handle
この関数を呼び出すときに現在接続されているデータベースと同じデータベース内の有効なアプリケーション・ハンドルを指定する、タイプ BIGINT のオプション入力引数。引数が NULL である場合、データベース内のすべてのスーパークラスで実行される作業単位についてメトリックが取得されます。

member
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なメンバーを指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・メンバーには -1、すべてのデータベース・メンバーには -2 を指定します。NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

MON_GET_UNIT_OF_WORK 関数に対する EXECUTE 特権。

例

システムで最長の CPU 時間を消費する作業単位を識別します。

```
SELECT application_handle,
       uow_id,
       total_cpu_time,
       app_rqsts_completed_total,
       rqsts_completed_total
FROM TABLE(MON_GET_UNIT_OF_WORK(NULL,-1)) AS t
ORDER BY total_cpu_time DESC
```

以下はこの照会の出力例です。

```
APPLICATION_HANDLE   UOW_ID      TOTAL_CPU_TIME       ...
-------------------- ----------- -------------------- ...
                  46           5                27959 ...

  1 record(s) selected.
```

照会の出力 (続き)。

```
... APP_RQSTS_COMPLETED_TOTAL RQSTS_COMPLETED_TOTAL
... ------------------------- --------------------
...                        72                   48
```

使用上の注意

MON_GET_UNIT_OF_WORK 表関数で戻されるメトリックは、作業単位の間にサブミットされた要求についてのすべてのメトリックの累計を表します。メトリックは作業単位の間に定期的にロールアップされます。したがって、この表関数で報告される値は、直前の ROLLUP 時のシステムの現行状態を反映しています。メトリッ

クの値は確実に増加します。ある時間間隔に対する指定されたメトリックの値を判別するには、この関数を使用してその間隔の始めと終わりのメトリックを照会し、差異を計算します。

要求メトリックは、サービス・スーパークラスに対する COLLECT REQUEST METRICS 節、およびデータベース・レベルの *mon_req_metrics* データベース構成パラメーターを介して制御されます。親サービス・スーパークラスで要求メトリックを使用可能にしているサービス・サブクラスのエージェントが要求を処理する場合にのみ、またはデータベース全体で要求メトリック・コレクションが有効な場合にのみ、その要求に対しメトリックが収集されます。デフォルトでは、要求メトリックはデータベース・レベルで使用可能です。要求メトリックがデータベース・レベルでもサービス・スーパークラスに対しても使用不可になっている場合、そのサービス・スーパークラスにマッピングされた作業単位ごとに報告されるメトリックは増加を停止します (または、要求メトリックがデータベースのアクティブ化時に無効であった場合には 0 のままになります)。

MON_GET_UNIT_OF_WORK 表関数は、作業単位ごとおよびメンバーごとに 1 行のデータを戻します。 (メンバー上の) 作業単位全体または (1 つ以上のサービス・クラスの) メンバー全体の集約は実行されません。しかし、集約は SQL 照会を使用して実行できます。この入力パラメーターの影響として、ANDing されます。

戻される情報

表 136. MON_GET_UNIT_OF_WORK について戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SERVICE_SUPERCLASS_NAME	VARCHAR(128)	service_superclass_name - サービス・スーパークラス名
SERVICE_SUBCLASS_NAME	VARCHAR(128)	service_subclass_name - サービス・サブクラス名
SERVICE_CLASS_ID	INTEGER	service_class_id - サービス・クラス ID
MEMBER	SMALLINT	member - データベース・メンバー
COORD_MEMBER	SMALLINT	coord_member - コーディネーター・メンバー
APPLICATION_HANDLE	BIGINT	application_handle - アプリケーション・ハンドル
APPLICATION_ID	VARCHAR(128)	appl_id - アプリケーション ID
WORKLOAD_NAME	VARCHAR(128)	workload_name - ワークロード名
WORKLOAD_OCCURRENCE_ID	INTEGER	workload_occurrence_id - ワークロード・オカレンス ID。ワークロード・オカレンスがコーディネーター・メンバーおよびワークロード名と結合されていなければ、この ID はワークロード・オカレンスを一意的に識別しません。
UOW_ID	INTEGER	uow_id - 作業単位 ID
WORKLOAD_OCCURRENCE_STATE	VARCHAR(32)	workload_occurrence_state - ワークロード・オカレンスの状態
CLIENT_WRKSTNNAME	VARCHAR(255)	CURRENT CLIENT_WRKSTNNAME 特殊レジスター
CLIENT_ACCTNG	VARCHAR(255)	CURRENT CLIENT_ACCTNG 特殊レジスター
CLIENT_USERID	VARCHAR(255)	CURRENT CLIENT_USERID 特殊レジスター
CLIENT_APPLNAME	VARCHAR(255)	CURRENT CLIENT_APPLNAME 特殊レジスター

表 136. MON_GET_UNIT_OF_WORK について戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
UOW_START_TIME	TIMESTAMP	uow_start_time - 作業単位開始タイム・スタンプ
SESSION_AUTH_ID	VARCHAR(128)	session_auth_id - セッション許可 ID
ACT_ABORTED_TOTAL	BIGINT	act_aborted_total - 打ち切られたアクティビティーの合計数
ACT_COMPLETED_TOTAL	BIGINT	act_completed_total - 完了したアクティビティーの合計数
ACT_REJECTED_TOTAL	BIGINT	act_rejected_total - リジェクトされたアクティビティーの合計数
AGENT_WAIT_TIME	BIGINT	agent_wait_time - エージェント待ち時間
AGENT_WAITS_TOTAL	BIGINT	agent_waits_total - エージェント待機の合計
POOL_DATA_L_READS	BIGINT	pool_data_l_reads - バッファー・プール・データの論理読み取り
POOL_INDEX_L_READS	BIGINT	pool_index_l_reads - バッファー・プール索引の論理読み取り
POOL_TEMP_DATA_L_READS	BIGINT	pool_temp_data_l_reads - バッファー・プール一時データの論理読み取り
POOL_TEMP_INDEX_L_READS	BIGINT	pool_temp_index_l_reads - バッファー・プール一時索引の論理読み取り
POOL_TEMP_XDA_L_READS	BIGINT	pool_temp_xda_l_reads - バッファー・プール一時 XDA データの論理読み取り
POOL_XDA_L_READS	BIGINT	pool_xda_l_reads - バッファー・プール XDA データの論理読み取り
POOL_DATA_P_READS	BIGINT	pool_data_p_reads - バッファー・プール・データの物理読み取り
POOL_INDEX_P_READS	BIGINT	pool_index_p_reads - バッファー・プール索引の物理読み取り
POOL_TEMP_DATA_P_READS	BIGINT	pool_temp_data_p_reads - バッファー・プール一時データの物理読み取り
POOL_TEMP_INDEX_P_READS	BIGINT	pool_temp_index_p_reads - バッファー・プール一時索引の物理読み取り
POOL_TEMP_XDA_P_READS	BIGINT	pool_temp_xda_p_reads - バッファー・プール一時 XDA データの物理読み取り
POOL_XDA_P_READS	BIGINT	pool_xda_p_reads - バッファー・プール XDA データの物理読み取り
POOL_DATA_WRITES	BIGINT	pool_data_writes - バッファー・プールへのデータの書き込み
POOL_INDEX_WRITES	BIGINT	pool_index_writes - バッファー・プール索引の書き込み
POOL_XDA_WRITES	BIGINT	pool_xda_writes - バッファー・プール XDA データ書き込み
POOL_READ_TIME	BIGINT	pool_read_time - バッファー・プール物理読み取り時間の合計

表 136. MON_GET_UNIT_OF_WORK について戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
POOL_WRITE_TIME	BIGINT	pool_write_time - バッファー・プール物理書き込み時間の合計
CLIENT_IDLE_WAIT_TIME	BIGINT	client_idle_wait_time - クライアント・アイドル待ち時間
DEADLOCKS	BIGINT	deadlocks - デッドロック検出数
DIRECT_READS	BIGINT	direct_reads - データベースからの直接読み取り
DIRECT_READ_TIME	BIGINT	direct_read_time - 直接読み取り時間
DIRECT_WRITES	BIGINT	direct_writes - データベースへの直接書き込み
DIRECT_WRITE_TIME	BIGINT	direct_write_time - 直接書き込み時間
DIRECT_READ_REQS	BIGINT	direct_read_reqs - 直接読み取り要求
DIRECT_WRITE_REQS	BIGINT	direct_write_reqs - 直接書き込み要求
FCM_RECV_VOLUME	BIGINT	fcm_recv_volume - FCM 受信ボリューム
FCM_RECVS_TOTAL	BIGINT	fcm_recvs_total - FCM 合計受信数
FCM_SEND_VOLUME	BIGINT	fcm_send_volume - FCM 送信ボリューム
FCM_SENDS_TOTAL	BIGINT	fcm_sends_total - FCM 合計送信数
FCM_RECV_WAIT_TIME	BIGINT	fcm_recv_wait_time - FCM 受信待ち時間
FCM_SEND_WAIT_TIME	BIGINT	fcm_send_wait_time - FCM 送信待ち時間
IPC_RECV_VOLUME	BIGINT	ipc_recv_volume - プロセス間通信受信ボリューム
IPC_RECV_WAIT_TIME	BIGINT	ipc_recv_wait_time - プロセス間通信受信待ち時間
IPC_RECVS_TOTAL	BIGINT	ipc_recvs_total - プロセス間通信合計受信数
IPC_SEND_VOLUME	BIGINT	ipc_send_volume - プロセス間通信送信ボリューム
IPC_SEND_WAIT_TIME	BIGINT	ipc_send_wait_time - プロセス間通信送信待ち時間
IPC_SENDS_TOTAL	BIGINT	ipc_sends_total - プロセス間通信合計送信数
LOCK_ESCALS	BIGINT	lock_escals - ロック・エスカレーション数
LOCK_TIMEOUTS	BIGINT	lock_timeouts - ロック・タイムアウト数
LOCK_WAIT_TIME	BIGINT	lock_wait_time - ロック待機中の時間
LOCK_WAITS	BIGINT	lock_waits - ロック待機数
LOG_BUFFER_WAIT_TIME	BIGINT	log_buffer_wait_time - ログ・バッファー待ち時間
NUM_LOG_BUFFER_FULL	BIGINT	num_log_buffer_full - フル・ログ・バッファーの回数
LOG_DISK_WAIT_TIME	BIGINT	log_disk_wait_time - ログ・ディスク待ち時間
LOG_DISK_WAITS_TOTAL	BIGINT	log_disk_waits_total - ログ・ディスク待機の合計
NUM_LOCKS_HELD	BIGINT	locks_held - ロック保持数
RQSTS_COMPLETED_TOTAL	BIGINT	rqsts_completed_total - 完了した要求の合計数
ROWS_MODIFIED	BIGINT	rows_modified - 変更された行数
ROWS_READ	BIGINT	rows_read - 読み取り行数
ROWS_RETURNED	BIGINT	rows_returned - 戻り行数
TCPIP_RECV_VOLUME	BIGINT	tcpip_recv_volume - TCP/IP 受信ボリューム
TCPIP_SEND_VOLUME	BIGINT	tcpip_send_volume - TCP/IP 送信ボリューム
TCPIP_RECV_WAIT_TIME	BIGINT	tcpip_recv_wait_time - TCP/IP 受信待ち時間

表 136. MON_GET_UNIT_OF_WORK について戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
TCPIP_RECVS_TOTAL	BIGINT	tcpip_recvs_total - TCP/IP 合計受信数
TCPIP_SEND_WAIT_TIME	BIGINT	tcpip_send_wait_time - TCP/IP 送信待ち時間
TCPIP_SENDS_TOTAL	BIGINT	tcpip_sends_total - TCP/IP 合計送信数
TOTAL_APP_RQST_TIME	BIGINT	total_app_rqst_time - アプリケーション要求合計時間
TOTAL_RQST_TIME	BIGINT	total_rqst_time - 合計要求時間
WLM_QUEUE_TIME_TOTAL	BIGINT	wlm_queue_time_total - ワークロード・マネージャー合計キュー時間
WLM_QUEUE_ASSIGNMENTS _TOTAL	BIGINT	wlm_queue_assignments_total - ワークロード・マネージャー合計キュー割り当て数
TOTAL_CPU_TIME	BIGINT	total_cpu_time - 合計 CPU 時間
TOTAL_WAIT_TIME	BIGINT	total_wait_time - 合計待ち時間
APP_RQSTS_COMPLETED_TOTAL	BIGINT	app_rqsts_completed_total - 完了したアプリケーション要求の合計数
TOTAL_SECTION_SORT_TIME	BIGINT	total_section_sort_time - セクション・ソート時間合計
TOTAL_SECTION_SORT_PROC_TIME	BIGINT	total_section_sort_proc_time - セクション・ソート処理時間合計
TOTAL_SECTION_SORTS	BIGINT	total_section_sorts - セクション・ソート合計
TOTAL_SORTS	BIGINT	total_sorts - ソート合計
POST_THRESHOLD_SORTS	BIGINT	post_threshold_sorts - ポストしきい値ソート
POST_SHRTHRESHOLD_SORTS	BIGINT	post_shrthreshold_sorts - ポスト共有しきい値ソート
SORT_OVERFLOWS	BIGINT	sort_overflows - ソート・オーバーフロー
TOTAL_COMPILE_TIME	BIGINT	total_compile_time - コンパイル時間の合計
TOTAL_COMPILE_PROC_TIME	BIGINT	total_compile_proc_time - コンパイル処理時間の合計
TOTAL_COMPILATIONS	BIGINT	total_compilations - コンパイルの合計回数
TOTAL_IMPLICIT_COMPILE_TIME	BIGINT	total_implicit_compile_time - 暗黙的コンパイル時間の合計
TOTAL_IMPLICIT_COMPILE _PROC_TIME	BIGINT	total_implicit_compile_proc_time - 暗黙的コンパイルの処理時間の合計
TOTAL_IMPLICIT_COMPILATIONS	BIGINT	total_implicit_compilations - 暗黙的コンパイルの合計回数
TOTAL_SECTION_TIME	BIGINT	total_section_time - セクション時間の合計
TOTAL_SECTION_PROC_TIME	BIGINT	total_section_proc_time - セクション処理時間の合計
TOTAL_APP_SECTION_EXECUTIONS	BIGINT	total_app_section_executions - セクション実行の合計回数
TOTAL_ACT_TIME	BIGINT	total_act_time - 合計アクティビティー時間
TOTAL_ACT_WAIT_TIME	BIGINT	total_act_wait_time - 合計アクティビティー待機時間
ACT_RQSTS_TOTAL	BIGINT	act_rqsts_total - アクティビティー要求の合計
TOTAL_ROUTINE_TIME	BIGINT	total_routine_time - ルーチン時間の合計

表 136. MON_GET_UNIT_OF_WORK について戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
TOTAL_ROUTINE_INVOCATIONS	BIGINT	total_routine_invocations - ルーチン呼び出しの合計回数
TOTAL_COMMIT_TIME	BIGINT	total_commit_time - コミット時間の合計
TOTAL_COMMIT_PROC_TIME	BIGINT	total_commit_proc_time - コミット処理時間の合計
TOTAL_APP_COMMITS	BIGINT	total_app_commits - アプリケーション・コミットの合計回数
INT_COMMITS	BIGINT	int_commits - 内部コミット数
TOTAL_ROLLBACK_TIME	BIGINT	total_rollback_time - ロールバック時間の合計
TOTAL_ROLLBACK_PROC_TIME	BIGINT	total_rollback_proc_time - ロールバック処理時間の合計
TOTAL_APP_ROLLBACKS	BIGINT	total_app_rollbacks - アプリケーション・ロールバックの合計回数
INT_ROLLBACKS	BIGINT	int_rollbacks - 内部ロールバック数
TOTAL_RUNSTATS_TIME	BIGINT	total_runstats_time - ランタイム統計時間の合計
TOTAL_RUNSTATS_PROC_TIME	BIGINT	total_runstats_proc_time - ランタイム統計の処理時間の合計
TOTAL_RUNSTATS	BIGINT	total_runstats - ランタイム統計の合計回数
TOTAL_REORG_TIME	BIGINT	total_reorg_time - 再編成時間の合計
TOTAL_REORG_PROC_TIME	BIGINT	total_reorg_proc_time - 再編成の処理時間の合計
TOTAL_REORGS	BIGINT	total_reorgs - 再編成の合計回数
TOTAL_LOAD_TIME	BIGINT	total_load_time - ロード時間の合計
TOTAL_LOAD_PROC_TIME	BIGINT	total_load_proc_time - ロード処理時間の合計
TOTAL_LOADS	BIGINT	total_loads - ロードの合計回数
CAT_CACHE_INSERTS	BIGINT	cat_cache_inserts - カタログ・キャッシュ挿入数
CAT_CACHE_LOOKUPS	BIGINT	cat_cache_lookups - カタログ・キャッシュ参照数
PKG_CACHE_INSERTS	BIGINT	pkg_cache_inserts - パッケージ・キャッシュ挿入
PKG_CACHE_LOOKUPS	BIGINT	pkg_cache_lookups - パッケージ・キャッシュ参照
THRESH_VIOLATIONS	BIGINT	thresh_violations - しきい値違反の回数
NUM_LW_THRESH_EXCEEDED	BIGINT	num_lw_thresh_exceeded - しきい値を超えた回数
UOW_LOG_SPACE_USED	BIGINT	uow_log_space_used - 作業単位ログ・スペース
ADDITIONAL_DETAILS	BLOB(100K)	将来の利用のために予約済み。

MON_GET_UNIT_OF_WORK_DETAILS 表関数 - 作業単位の詳細メトリックの取得

MON_GET_UNIT_OF_WORK_DETAILS 表関数は、1 つ以上の作業単位の詳細メトリックを戻します。

構文

スキーマは SYSPROC です。

表関数パラメーター

application_handle
この関数を呼び出すときに現在接続されているデータベースと同じデータベース内の有効なアプリケーション・ハンドルを指定する、タイプ BIGINT の入力引数。引数が NULL である場合、データベース内のすべてのスーパークラスで実行される作業単位についてメトリックが取得されます。

member
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なメンバーを指定する、タイプ INTEGER の入力引数。現行のデータベース・メンバーには -1、すべてのデータベース・メンバーには -2 を指定します。NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

MON_GET_UNIT_OF_WORK_DETAILS 関数に対する EXECUTE 特権。

例

システムで最長の CPU 時間を消費する作業単位を識別します。

```
SELECT detmetrics.application_handle,
    detmetrics.uow_id,
    detmetrics.total_cpu_time,
    detmetrics.app_rqsts_completed_total,
    detmetrics.rqsts_completed_total
FROM TABLE(MON_GET_UNIT_OF_WORK_DETAILS(NULL,-2)) AS UOWMETRICS,
XMLTABLE (
    XMLNAMESPACES( DEFAULT 'http://www.ibm.com/xmlns/prod/db2/mon'),
        '$detmetric/db2_unit_of_work' PASSING
        XMLPARSE(DOCUMENT UOWMETRICS.DETAILS)
        as "detmetric"
COLUMNS
    "APPLICATION_HANDLE" INTEGER PATH 'application_handle',
    "UOW_ID" INTEGER PATH 'uow_id',
    "TOTAL_CPU_TIME" INTEGER PATH 'system_metrics/total_cpu_time',
    "APP_RQSTS_COMPLETED_TOTAL" INTEGER
        PATH 'system_metrics/app_rqsts_completed_total',
    "RQSTS_COMPLETED_TOTAL" INTEGER
        PATH 'system_metrics/rqsts_completed_total'
) AS DETMETRICS
ORDER BY total_cpu_time DESC
```

以下はこの照会の出力例です。

```
APPLICATION_HANDLE   UOW_ID      TOTAL_CPU_TIME       ...
-------------------- ----------- -------------------- ...
                  46           5                27959 ...

  1 record(s) selected.
```

照会の出力 (続き)。

```
... APP_RQSTS_COMPLETED_TOTAL RQSTS_COMPLETED_TOTAL
... ------------------------- --------------------
...                        72                   48
```

使用上の注意

MON_GET_UNIT_OF_WORK 関数で戻されるメトリックは、作業単位の間にサブミットされた要求についてのすべてのメトリックの累計を表します。この関数は、以下の点で MON_GET_UNIT_OF_WORK 表関数と似ています。

- MON_GET_UNIT_OF_WORK 表関数は、最も一般的に使用されるメトリックを列ベースの形式で戻すため、メトリックの取得において最も効率的な方法です。
- MON_GET_UNIT_OF_WORK_DETAILS 表関数は、使用可能なすべてのメトリック一式を XML 文書形式で戻すため、出力フォーマットを最大限柔軟性のあるものにします。XML ベースの出力は、XML パーサーで直接解析でき、XMLTABLE 関数でリレーショナル形式に変換することもできます (例を参照)。

メトリックは作業単位の間に定期的にロールアップされます。したがって、この表関数で報告される値は、直前の ROLLUP 時のシステムの現行状態を反映しています。メトリックの値は確実に増加します。ある時間間隔に対する指定されたメトリックの値を判別するには、 MON_GET_UNIT_OF_WORK_DETAILS 表関数を使用してその間隔の始めと終わりのメトリックを照会し、差異を計算します。

要求メトリックは、サービス・スーパークラスに対する COLLECT REQUEST METRICS 節、およびデータベース・レベルの **mon_req_metrics** データベース構成パラメーターを介して制御されます。親サービス・スーパークラスで要求メトリックを使用可能にしているサービス・サブクラスのエージェントが要求を処理する場合にのみ、またはデータベース全体で要求メトリック・コレクションが有効な場合にのみ、その要求に対しメトリックが収集されます。デフォルトでは、要求メトリックはデータベース・レベルで使用可能です。要求メトリックがデータベース・レベルでもサービス・スーパークラスに対しても使用不可になっている場合、そのサービス・スーパークラスにマッピングされた作業単位ごとに報告されるメトリックは増加を停止します (または、要求メトリックがデータベースのアクティブ化時に無効であった場合には 0 のままになります)。

MON_GET_UNIT_OF_WORK_DETAILS 表関数は、作業単位ごとおよびメンバーごとに 1 行のデータを戻します。 (メンバー上の) 作業単位全体または (1 つ以上のサービス・クラスの) メンバー全体の集約は実行されません。しかし、集約は SQL 照会を使用して実行できます。この入力パラメーターの影響として、ANDing されます。

DETAILS 列に戻される XML 文書のスキーマは、ファイル `sqllib/misc/DB2MonRoutines.xsd` で入手できます。詳細は、ファイル `sqllib/misc/DB2MonCommon.xsd` 内にあります。

戻される情報

表 137. MON_GET_UNIT_OF_WORK_DETAILS について戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SERVICE_SUPERCLASS_NAME	VARCHAR(128)	service_superclass_name - サービス・スーパークラス名
SERVICE_SUBCLASS_NAME	VARCHAR(128)	service_subclass_name - サービス・サブクラス名
SERVICE_CLASS_ID	INTEGER	service_class_id - サービス・クラス ID
MEMBER	SMALLINT	member - データベース・メンバー

表 137. MON_GET_UNIT_OF_WORK_DETAILS について戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
COORD_MEMBER	SMALLINT	coord_member - コーディネーター・メンバー。指定された作業単位のコーディネーター・パーティションのデータベース・メンバー。
APPLICATION_HANDLE	BIGINT	application_handle - アプリケーション・ハンドル
WORKLOAD_NAME	VARCHAR(128)	workload_name - ワークロード名
WORKLOAD_OCCURRENCE_ID	INTEGER	workload_occurrence_id - ワークロード・オカレンス ID。ワークロード・オカレンスがコーディネーター・データベース・パーティション番号およびワークロード名と結合されていなければ、この ID はワークロード・オカレンスを一意的に識別しません。
UOW_ID	INTEGER	uow_id - 作業単位 ID
DETAILS	BLOB(1M)	作業単位についての詳細メトリックを含む XML 文書。エレメントの説明については、本書の表 138を参照してください。

以下の例は、DETAILS 列で戻される XML 文書の構造を示しています。

```
<db2_unit_of_work xmlns="http://www.ibm.com/xmlns/prod/db2/mon" release="90700000">
   <service_superclass_name>SYSDEFAULTUSERCLASS</service_superclass_name>
   <service_subclass_name>SYSDEFAULTSUBCLASS</service_subclass_name>
   <service_class_id>13</service_class_id>
   <workload_name>SYSDEFAULTUSERWORKLOAD</workload_name>
   <member>0</member>
   <coord_member>0</coord_member>
   <application_handle>21</application_handle>
   <workload_occurrence_id>1</workload_occurrence_id>
   <uow_id>2</uow_id>
   <workload_occurrence_state>UOWEXEC</workload_occurrence_state>
   <system_metrics>
   <act_aborted_total>5</act_aborted_total>
   ...
   <wlm_queue_assignments_total>3</wlm_queue_assignments_total>
   </system_metrics>
</db2_unit_of_work_metrics>
```

完全スキーマについては、sqllib/misc/DB2MonRoutines.xsd を参照してください。

表 138. MON_GET_UNIT_OF_WORK_DETAILS について戻される詳細メトリック

エレメント名	データ・タイプ	説明または対応するモニター・エレメント
service_superclass_name	xs:string (128)	service_superclass_name - サービス・スーパークラス名
service_subclass_name	xs:string (128)	service_subclass_name - サービス・サブクラス名
service_class_id	xs:nonNegativeInteger	service_class_id - サービス・クラス ID
workload_name	xs:string (128)	workload_name - ワークロード名
member	xs:nonNegativeInteger	member - データベース・メンバー
coord_member	xs:nonNegativeInteger	coord_member - コーディネーター・メンバー
application_handle	xs:nonNegativeInteger	application_handle - アプリケーション・ハンドル
application_id	xs:string	appl_id - アプリケーション ID

表 138. MON_GET_UNIT_OF_WORK_DETAILS について戻される詳細メトリック (続き)

エレメント名	データ・タイプ	説明または対応するモニター・エレメント
workload_occurrence_id	xs:nonNegativeInteger	workload_occurrence_id - ワークロード・オカレンス ID ワークロード・オカレンスがコーディネーター・メンバーおよびワークロード名と結合されていなければ、この ID はワークロード・オカレンスを一意的に識別しません。
uow_id	xs:nonNegativeInteger	uow_id - 作業単位 ID
workload_occurrence_state	xs:string	workload_occurrence_state - ワークロード・オカレンスの状態
client_userid	xs:string	CURRENT CLIENT_USERID 特殊レジスター
client_wrkstnname	xs:string	CURRENT CLIENT_WRKSTNNAME 特殊レジスター
client_applname	xs:string	CURRENT CLIENT_APPLNAME 特殊レジスター
client_acctng	xs:string	CURRENT CLIENT_ACCTNG 特殊レジスター
act_aborted_total	xs:nonNegativeInteger	act_aborted_total - 打ち切られたアクティビティーの合計数
act_completed_total	xs:nonNegativeInteger	act_completed_total - 完了したアクティビティーの合計数
act_rejected_total	xs:nonNegativeInteger	act_rejected_total - リジェクトされたアクティビティーの合計数
act_rqsts_total	xs:nonNegativeInteger	act_rqsts_total - アクティビティー要求の合計
agent_wait_time	xs:nonNegativeInteger	agent_wait_time - エージェント待機時間
agent_waits_total	xs:nonNegativeInteger	agent_waits_total - エージェント待機の合計
app_rqsts_completed_total	xs:nonNegativeInteger	app_rqsts_completed_total - 完了したアプリケーション要求の合計数
audit_events_total	xs:nonNetagiveIngteger	audit_events_total - 監査イベント合計数
audit_subsystem_wait_time	xs:nonNegativeInteger	audit_subsystem_wait_time - 監査サブシステム待機時間
audit_subsystem_waits_total	xs:nonNegativeInteger	audit_subsystem_waits_total - 監査サブシステム待機の合計
audit_file_write_wait_time	xs:nonNegativeInteger	audit_file_write_wait_time - 監査ファイル書き込み待機時間
audit_file_writes_total	xs:nonNegativeInteger	audit_file_writes_total - 書き込まれた監査ファイルの合計数
cat_cache_inserts	xs:nonNegativeInteger	cat_cache_inserts - カタログ・キャッシュ挿入数
cat_cache_lookups	xs:nonNegativeInteger	cat_cache_lookups - カタログ・キャッシュ参照数
client_hostname	xs:string	client_hostname - クライアント・ホスト名
client_idle_wait_time	xs:nonNegativeInteger	client_idle_wait_time - クライアントのアイドル待機時間
client_port_number	xs:nonNegativeInteger	client_port_number - クライアント・ポート番号
deadlocks	xs:nonNegativeInteger	deadlocks - デッドロック検出数
diaglog_writes_total	xs:nonNegativeInteger	diaglog_writes_total - 診断ログ・ファイル書き込みの合計
diaglog_write_wait_time	xs:nonNegativeInteger	diaglog_write_wait_time - 診断ログ・ファイルの書き込み待機時間

表 138. MON_GET_UNIT_OF_WORK_DETAILS について戻される詳細メトリック (続き)

エレメント名	データ・タイプ	説明または対応するモニター・エレメント
direct_read_time	xs:nonNegativeInteger	direct_read_time - 直接読み取り時間
direct_write_time	xs:nonNegativeInteger	direct_write_time - 直接書き込み時間
direct_read_reqs	xs:nonNegativeInteger	direct_read_reqs - 直接読み取り要求
direct_reads	xs:nonNegativeInteger	direct_reads - データベースからの直接読み取り
direct_write_reqs	xs:nonNegativeInteger	direct_write_reqs - 直接書き込み要求
direct_writes	xs:nonNegativeInteger	direct_writes - データベースへの直接書き込み
fcm_recv_volume	xs:nonNegativeInteger	fcm_recv_volume - FCM 受信ボリューム
fcm_recv_wait_time	xs:nonNegativeInteger	fcm_recv_wait_time - FCM 受信待機時間
fcm_recvs_total	xs:nonNegativeInteger	fcm_recvs_total - FCM 合計受信数
fcm_message_recv_volume	xs:nonNegativeInteger	fcm_message_recv_volume - FCM メッセージ受信ボリューム
fcm_message_recvs_total	xs:nonNegativeInteger	fcm_message_recvs_total - FCM メッセージ合計受信数
fcm_message_recv_wait_time	xs:nonNegativeInteger	fcm_message_recv_wait_time - FCM メッセージの受信待機時間
fcm_message_send_volume	xs:nonNegativeInteger	fcm_message_send_volume - FCM メッセージ送信ボリューム
fcm_message_send_wait_time	xs:nonNegativeInteger	fcm_message_send_wait_time - FCM メッセージ送信待ち時間
fcm_message_sends_total	xs:nonNegativeInteger	fcm_message_sends_total - FCM メッセージ合計送信数
fcm_send_volume	xs:nonNegativeInteger	fcm_send_volume - FCM 送信ボリューム
fcm_send_wait_time	xs:nonNegativeInteger	fcm_send_wait_time - FCM 送信待ち時間
fcm_sends_total	xs:nonNegativeInteger	fcm_sends_total - FCM 合計送信数
fcm_tq_recv_wait_time	xs:nonNegativeInteger	fcm_tq_recv_wait_time - FCM 表キューの受信待ち時間
fcm_tq_send_wait_time	xs:nonNegativeInteger	fcm_tq_send_wait_time - FCM 表キューの送信待ち時間
fcm_tq_recv_volume	xs:nonNegativeInteger	fcm_tq_recv_volume - FCM 表キューの受信ボリューム
fcm_tq_recvs_total	xs:nonNegativeInteger	fcm_tq_recvs_total - FCM 表キューの合計受信数
fcm_tq_send_volume	xs:nonNegativeInteger	fcm_tq_send_volume - FCM 表キューの送信ボリューム
fcm_tq_sends_total	xs:nonNegativeInteger	fcm_tq_sends_total - FCM 表キューの合計送信数
int_commits	xs:nonNegativeInteger	int_commits - 内部コミット数
int_rollbacks	xs:nonNegativeInteger	int_rollbacks - 内部ロールバック数
tq_tot_send_spills	xs:nonNegativeInteger	tq_tot_send_spills - オーバーフローした表キュー・バッファーの合計数
ipc_recv_volume	xs:nonNegativeInteger	ipc_recv_volume - プロセス間通信受信ボリューム
ipc_recv_wait_time	xs:nonNegativeInteger	ipc_recv_wait_time - プロセス間通信受信待ち時間
ipc_recvs_total	xs:nonNegativeInteger	ipc_recvs_total - プロセス間通信合計受信数
ipc_send_volume	xs:nonNegativeInteger	ipc_send_volume - プロセス間通信送信ボリューム
ipc_send_wait_time	xs:nonNegativeInteger	ipc_send_wait_time - プロセス間通信送信待ち時間

表 138. MON_GET_UNIT_OF_WORK_DETAILS について戻される詳細メトリック (続き)

エレメント名	データ・タイプ	説明または対応するモニター・エレメント
ipc_sends_total	xs:nonNegativeInteger	ipc_sends_total - プロセス間通信合計送信数
last_executable_id	xs:hexBinary(32)	last_executable_id - 最後の実行可能 ID
last_request_type	xs:string(32)	last_request_type - 最後の要求タイプ
lock_escals	xs:nonNegativeInteger	lock_escals - ロック・エスカレーション数
lock_timeouts	xs:nonNegativeInteger	lock_timeouts - ロック・タイムアウト数
lock_wait_time	xs:nonNegativeInteger	lock_wait_time - ロック待機中の時間
lock_waits	xs:nonNegativeInteger	lock_waits - ロック待機数
log_buffer_wait_time	xs:nonNegativeInteger	log_buffer_wait_time - ログ・バッファー待ち時間
log_disk_wait_time	xs:nonNegativeInteger	log_disk_wait_time - ログ・ディスク待機時間
log_disk_waits_total	xs:nonNegativeInteger	log_disk_waits_total - ログ・ディスク待機の合計
num_locks_held	xs:nonNegativeInteger	locks_held - ロック保持数
num_lw_thresh_exceeded	xs:nonNegativeInteger	num_lw_thresh_exceeded - しきい値を超えた回数
thresh_violations	xs:nonNegativeInteger	thresh_violations - しきい値違反の回数
pkg_cache_inserts	xs:nonNegativeInteger	pkg_cache_inserts - パッケージ・キャッシュ挿入
pkg_cache_lookups	xs:nonNegativeInteger	pkg_cache_lookups - パッケージ・キャッシュ参照
pool_data_l_reads	xs:nonNegativeInteger	pool_data_l_reads - バッファー・プール・データの論理読み取り
pool_data_p_reads	xs:nonNegativeInteger	pool_data_p_reads - バッファー・プール・データの物理読み取り
pool_data_writes	xs:nonNegativeInteger	pool_data_writes - バッファー・プールへのデータの書き込み
pool_index_l_reads	xs:nonNegativeInteger	pool_index_l_reads - バッファー・プール索引の論理読み取り
pool_index_p_reads	xs:nonNegativeInteger	pool_index_p_reads - バッファー・プール索引の物理読み取り
pool_index_writes	xs:nonNegativeInteger	pool_index_writes - バッファー・プール索引の書き込み
pool_read_time	xs:nonNegativeInteger	pool_read_time - バッファー・プール物理読み取り時間の合計
pool_temp_data_l_reads	xs:nonNegativeInteger	pool_temp_data_l_reads - バッファー・プール一時データの論理読み取り
pool_temp_data_p_reads	xs:nonNegativeInteger	pool_temp_data_p_reads - バッファー・プール一時データの物理読み取り
pool_temp_index_l_reads	xs:nonNegativeInteger	pool_temp_index_l_reads - バッファー・プール一時索引の論理読み取り
pool_temp_index_p_reads	xs:nonNegativeInteger	pool_temp_index_p_reads - バッファー・プール一時索引の物理読み取り
pool_temp_xda_l_reads	xs:nonNegativeInteger	pool_temp_xda_l_reads - バッファー・プール一時 XDA データの論理読み取り
pool_temp_xda_p_reads	xs:nonNegativeInteger	pool_temp_xda_p_reads - バッファー・プール一時 XDA データの物理読み取り

表 138. MON_GET_UNIT_OF_WORK_DETAILS について戻される詳細メトリック (続き)

エレメント名	データ・タイプ	説明または対応するモニター・エレメント
pool_write_time	xs:nonNegativeInteger	pool_write_time - バッファー・プール物理書き込み時間の合計
pool_xda_l_reads	xs:nonNegativeInteger	pool_xda_l_reads - バッファー・プール XDA データの論理読み取り
pool_xda_p_reads	xs:nonNegativeInteger	pool_xda_p_reads - バッファー・プール XDA データの物理読み取り
pool_xda_writes	xs:nonNegativeInteger	pool_xda_writes - バッファー・プール XDA データの書き込み
num_log_buffer_full	xs:nonNegativeInteger	num_log_buffer_full - フル・ログ・バッファーの回数
rqsts_completed_total	xs:nonNegativeInteger	rqsts_completed_total - 完了した要求の合計数
rows_modified	xs:nonNegativeInteger	rows_modified - 変更された行数
rows_read	xs:nonNegativeInteger	rows_read - 読み取り行数
rows_returned	xs:nonNegativeInteger	rows_returned - 戻り行数
session_auth_id	xs:string	session_auth_id - セッション許可 ID
tcpip_recv_volume	xs:nonNegativeInteger	tcpip_recv_volume - TCP/IP 受信ボリューム
tcpip_recv_wait_time	xs:nonNegativeInteger	tcpip_recv_wait_time - TCP/IP 受信待ち時間
tcpip_recvs_total	xs:nonNegativeInteger	tcpip_recvs_total - TCP/IP 合計受信数
tcpip_send_volume	xs:nonNegativeInteger	tcpip_send_volume - TCP/IP 送信ボリューム
tcpip_send_wait_time	xs:nonNegativeInteger	tcpip_send_wait_time - TCP/IP 送信待ち時間
tcpip_sends_total	xs:nonNegativeInteger	tcpip_sends_total - TCP/IP 合計送信数
total_act_time	xs:nonNegativeInteger	total_act_time - 合計アクティビティー時間
total_act_wait_time	xs:nonNegativeInteger	total_act_wait_time - 合計アクティビティー待機時間
total_app_commits	xs:nonNegativeInteger	total_app_commits - アプリケーション・コミットの合計回数
total_app_rollbacks	xs:nonNegativeInteger	total_app_rollbacks - アプリケーション・ロールバックの合計回数
total_app_rqst_time	xs:nonNegativeInteger	total_app_rqst_time - アプリケーション要求合計時間
total_app_section_executions	xs:nonNegativeInteger	total_app_section_executions - セクション実行の合計回数
total_commit_proc_time	xs:nonNegativeInteger	total_commit_proc_time - コミット処理時間の合計
total_commit_time	xs:nonNegativeInteger	total_commit_time - コミット時間の合計
total_compilations	xs:nonNegativeInteger	total_compilations - コンパイルの合計回数
total_compile_proc_time	xs:nonNegativeInteger	total_compile_proc_time - コンパイル処理時間の合計
total_compile_time	xs:nonNegativeInteger	total_compile_time - コンパイル時間の合計
total_cpu_time	xs:nonNegativeInteger	total_cpu_time - 合計 CPU 時間
total_implicit_compilations	xs:nonNegativeInteger	total_implicit_compilations - 暗黙的コンパイルの合計回数
total_implicit_compile_ proc_time	xs:nonNegativeInteger	total_implicit_compile_proc_time - 暗黙的コンパイルの処理時間の合計
total_implicit_compile_time	xs:nonNegativeInteger	total_implicit_compile_time - 暗黙的コンパイル時間の合計
total_loads	xs:nonNegativeInteger	total_loads - ロードの合計回数

表 138. MON_GET_UNIT_OF_WORK_DETAILS について戻される詳細メトリック (続き)

エレメント名	データ・タイプ	説明または対応するモニター・エレメント
total_load_proc_time	xs:nonNegativeInteger	total_load_proc_time - ロード処理時間の合計
total_load_time	xs:nonNegativeInteger	total_load_time - ロード時間の合計
total_reorgs	xs:nonNegativeInteger	total_reorgs - 再編成の合計回数
total_reorg_proc_time	xs:nonNegativeInteger	total_reorg_proc_time - 再編成の処理時間の合計
total_reorg_time	xs:nonNegativeInteger	total_reorg_time - 再編成時間の合計
total_rollback_proc_time	xs:nonNegativeInteger	total_rollback_proc_time - ロールバック処理時間の合計
total_rollback_time	xs:nonNegativeInteger	total_rollback_time - ロールバック時間の合計
total_routine_invocations	xs:nonNegativeInteger	total_routine_invocations - ルーチン呼び出しの合計回数
total_routine_time	xs:nonNegativeInteger	total_routine_time - ルーチン時間の合計
total_routine_user_code _proc_time	xs:nonNegativeInteger	total_routine_user_code_proc_time - ルーチン・ユーザー・コード処理時間の合計
total_routine_user_code_time	xs:nonNegativeInteger	total_routine_user_code_time - ルーチン・ユーザー・コード時間の合計
total_rqst_time	xs:nonNegativeInteger	total_rqst_time - 合計要求時間
total_runstats	xs:nonNegativeInteger	total_runstats - ランタイム統計の合計回数
total_runstats_proc_time	xs:nonNegativeInteger	total_runstats_proc_time - ランタイム統計の処理時間の合計
total_runstats_time	xs:nonNegativeInteger	total_runstats_time - ランタイム統計時間の合計
total_section_proc_time	xs:nonNegativeInteger	total_section_proc_time - セクション処理時間の合計
total_section_sort_time	xs:nonNegativeInteger	total_section_sort_time - セクション・ソート時間合計
total_section_sort_proc_time	xs:nonNegativeInteger	total_section_sort_proc_time - セクション・ソート処理時間合計
total_section_sorts	xs:nonNegativeInteger	total_section_sorts - セクション・ソート合計
total_section_time	xs:nonNegativeInteger	total_section_time - セクション時間の合計
total_sorts	xs:nonNegativeInteger	total_sorts - ソート合計
post_threshold_sorts	xs:nonNegativeInteger	post_threshold_sorts - ポストしきい値ソート
post_shrthreshold_sorts	xs:nonNegativeInteger	post_shrthreshold_sorts - ポスト共有しきい値ソート
sort_overflows	xs:nonNegativeInteger	sort_overflows - ソート・オーバーフロー
tq_tot_send_spills	xs:nonNegativeInteger	tq_tot_send_spills - オーバーフローした表キュー・バッファーの合計数
total_wait_time	xs:nonNegativeInteger	total_wait_time - 合計待ち時間
uow_log_space_used	xs:nonNegativeInteger	uow_log_space_used - 作業単位ログ・スペース
uow_start_time	xs:dateTime	uow_start_time - 作業単位開始タイム・スタンプ
wlm_queue_time_total	xs:nonNegativeInteger	wlm_queue_time_total - ワークロード・マネージャー合計キュー時間
wlm_queue_assignments_total	xs:nonNegativeInteger	wlm_queue_assignments_total - ワークロード・マネージャー合計キュー割り当て数

MON_GET_WORKLOAD 表関数 - ワークロード・メトリックの取得

MON_GET_WORKLOAD 表関数は、1 つ以上のワークロードのメトリックを戻します。

構文

スキーマは SYSPROC です。

表関数パラメーター

workload_name
メトリックが戻される特定のワークロードを指定する、タイプ VARCHAR(128) の入力引数。引数が NULL または空ストリングである場合、すべてのワークロードについてメトリックが戻されます。

member
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なメンバーを指定する、タイプ INTEGER の入力引数。現行のデータベース・メンバーには -1、すべてのデータベース・メンバーには -2 を指定します。NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

MON_GET_WORKLOAD 関数に対する EXECUTE 特権。

例

メンバー全体で集約し、ロック待機時間の長い順に、ワークロードごとのロック情報を表示します。

```
SELECT varchar(workload_name,30) as workload_name,
       sum(lock_wait_time) as total_lock_wait_time,
       sum(lock_waits) as total_lock_waits,
       sum(lock_timeouts) as total_lock_timeouts,
       sum(lock_escals) as total_lock_escals
FROM TABLE(MON_GET_WORKLOAD('',-2)) AS t
GROUP BY workload_name
ORDER BY total_lock_wait_time DESC
```

以下はこの照会の出力例です。

```
WORKLOAD_NAME                  TOTAL_LOCK_WAIT_TIME TOTAL_LOCK_WAITS     ...
------------------------------ -------------------- -------------------- ...
SYSDEFAULTADMWORKLOAD                             0                    0 ...
SYSDEFAULTUSERWORKLOAD                            0                    0 ...

  2 record(s) selected.
```

照会の出力 (続き)。

```
... TOTAL_LOCK_TIMEOUTS  TOTAL_LOCK_ESCALS
... -------------------- --------------------
...                    0                    0
...                    0                    0
```

使用上の注意

MON_GET_WORKLOAD 表関数で戻されるメトリックは、識別されたワークロード・オブジェクトにマッピングされた接続によってサブミットされた要求のすべてのメトリックの累計を表します。メトリックは、作業単位境界でワークロードにロールアップされ、要求の実行中には定期的にロールアップされます。したがって、この表関数で報告される値は、直前の ROLLUP 時のシステムの現行状態を反映しています。メトリックの値は確実に増加します。ある時間間隔に対する指定されたメトリックの値を判別するには、MON_GET_WORKLOAD 表関数を使用してその間隔の始めと終わりのメトリックを照会し、差異を計算します。

要求メトリックは、サービス・スーパークラスに対する COLLECT REQUEST METRICS 節、およびデータベース・レベルの *mon_req_metrics* データベース構成パラメーターを介して制御されます。親サービス・スーパークラスで要求メトリックを使用可能にしているサービス・サブクラスのエージェントが要求を処理する場合にのみ、またはデータベース全体で要求メトリック・コレクションが有効な場合にのみ、その要求に対しメトリックが収集されます。デフォルトでは、要求メトリックはデータベース・レベルで使用可能です。要求メトリックがデータベース・レベルでもサービス・スーパークラスに対しても使用不可になっている場合、そのサービス・スーパークラスにマッピングされたワークロードごとに報告されるメトリックは増加を停止します (または、要求メトリックがデータベースのアクティブ化時に無効であった場合には 0 のままになります)。

MON_GET_WORKLOAD 表関数は、ワークロードおよびメンバーごとに 1 行のデータを戻します。 (メンバー上の) ワークロード全体または (1 つ以上のサービス・クラスの) メンバー全体の集約は実行されません。しかし、集約は例に示されるように SQL 照会を使用して実行できます (例を参照)。

戻される情報

表 139. MON_GET_WORKLOAD について戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
WORKLOAD_NAME	VARCHAR(128)	workload_name - ワークロード名
WORKLOAD_ID	INTEGER	workload_id - ワークロード ID
MEMBER	SMALLINT	member - データベース・メンバー
ACT_ABORTED_TOTAL	BIGINT	act_aborted_total - 打ち切られたアクティビティーの合計数
ACT_COMPLETED_TOTAL	BIGINT	act_completed_total - 完了したアクティビティーの合計数
ACT_REJECTED_TOTAL	BIGINT	act_rejected_total - リジェクトされたアクティビティーの合計数
AGENT_WAIT_TIME	BIGINT	agent_wait_time - エージェント待機時間
AGENT_WAITS_TOTAL	BIGINT	agent_waits_total - エージェント待機の合計
POOL_DATA_L_READS	BIGINT	pool_data_l_reads - バッファー・プール・データの論理読み取り
POOL_INDEX_L_READS	BIGINT	pool_index_l_reads - バッファー・プール索引の論理読み取り

表 139. MON_GET_WORKLOAD について戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
POOL_TEMP_DATA_L_READS	BIGINT	pool_temp_data_l_reads - バッファー・プール一時データの論理読み取り
POOL_TEMP_INDEX_L_READS	BIGINT	pool_temp_index_l_reads - バッファー・プール一時索引の論理読み取り
POOL_TEMP_XDA_L_READS	BIGINT	pool_temp_xda_l_reads - バッファー・プール一時 XDA データの論理読み取り
POOL_XDA_L_READS	BIGINT	pool_xda_l_reads - バッファー・プール XDA データの論理読み取り
POOL_DATA_P_READS	BIGINT	pool_data_p_reads - バッファー・プール・データの物理読み取り
POOL_INDEX_P_READS	BIGINT	pool_index_p_reads - バッファー・プール索引の物理読み取り
POOL_TEMP_DATA_P_READS	BIGINT	pool_temp_data_p_reads - バッファー・プール一時データの物理読み取り
POOL_TEMP_INDEX_P_READS	BIGINT	pool_temp_index_p_reads - バッファー・プール一時索引の物理読み取り
POOL_TEMP_XDA_P_READS	BIGINT	pool_temp_xda_p_reads - バッファー・プール一時 XDA データの物理読み取り
POOL_XDA_P_READS	BIGINT	pool_xda_p_reads - バッファー・プール XDA データの物理読み取り
POOL_DATA_WRITES	BIGINT	pool_data_writes - バッファー・プールへのデータの書き込み
POOL_INDEX_WRITES	BIGINT	pool_index_writes - バッファー・プール索引の書き込み
POOL_XDA_WRITES	BIGINT	pool_xda_writes - バッファー・プール XDA データの書き込み
POOL_READ_TIME	BIGINT	pool_read_time - バッファー・プール物理読み取り時間の合計
POOL_WRITE_TIME	BIGINT	pool_write_time - バッファー・プール物理書き込み時間の合計
CLIENT_IDLE_WAIT_TIME	BIGINT	client_idle_wait_time - クライアントのアイドル待機時間
DEADLOCKS	BIGINT	deadlocks - デッドロック検出数
DIRECT_READS	BIGINT	direct_reads - データベースからの直接読み取り
DIRECT_READ_TIME	BIGINT	direct_read_time - 直接読み取り時間
DIRECT_WRITES	BIGINT	direct_writes - データベースへの直接書き込み
DIRECT_WRITE_TIME	BIGINT	direct_write_time - 直接書き込み時間
DIRECT_READ_REQS	BIGINT	direct_read_reqs - 直接読み取り要求
DIRECT_WRITE_REQS	BIGINT	direct_write_reqs - 直接書き込み要求
FCM_RECV_VOLUME	BIGINT	fcm_recv_volume - FCM 受信ボリューム
FCM_RECVS_TOTAL	BIGINT	fcm_recvs_total - FCM 合計受信数
FCM_SEND_VOLUME	BIGINT	fcm_send_volume - FCM 送信ボリューム
FCM_SENDS_TOTAL	BIGINT	fcm_sends_total - FCM 合計送信数

表 139. MON_GET_WORKLOAD について戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
FCM_RECV_WAIT_TIME	BIGINT	fcm_recv_wait_time - FCM 受信待機時間
FCM_SEND_WAIT_TIME	BIGINT	fcm_send_wait_time - FCM 送信待ち時間
IPC_RECV_VOLUME	BIGINT	ipc_recv_volume - プロセス間通信受信ボリューム
IPC_RECV_WAIT_TIME	BIGINT	ipc_recv_wait_time - プロセス間通信受信待ち時間
IPC_RECVS_TOTAL	BIGINT	ipc_recvs_total - プロセス間通信合計受信数
IPC_SEND_VOLUME	BIGINT	ipc_send_volume - プロセス間通信送信ボリューム
IPC_SEND_WAIT_TIME	BIGINT	ipc_send_wait_time - プロセス間通信送信待ち時間
IPC_SENDS_TOTAL	BIGINT	ipc_sends_total - プロセス間通信合計送信数
LOCK_ESCALS	BIGINT	lock_escals - ロック・エスカレーション数
LOCK_TIMEOUTS	BIGINT	lock_timeouts - ロック・タイムアウト数
LOCK_WAIT_TIME	BIGINT	lock_wait_time - ロック待機中の時間
LOCK_WAITS	BIGINT	lock_waits - ロック待機数
LOG_BUFFER_WAIT_TIME	BIGINT	log_buffer_wait_time - ログ・バッファー待ち時間
NUM_LOG_BUFFER_FULL	BIGINT	num_log_buffer_full - フル・ログ・バッファーの回数
LOG_DISK_WAIT_TIME	BIGINT	log_disk_wait_time - ログ・ディスク待機時間
LOG_DISK_WAITS_TOTAL	BIGINT	log_disk_waits_total - ログ・ディスク待機の合計
RQSTS_COMPLETED_TOTAL	BIGINT	rqsts_completed_total - 完了した要求の合計数
ROWS_MODIFIED	BIGINT	rows_modified - 変更された行数
ROWS_READ	BIGINT	rows_read - 読み取り行数
ROWS_RETURNED	BIGINT	rows_returned - 戻り行数
TCPIP_RECV_VOLUME	BIGINT	tcpip_recv_volume - TCP/IP 受信ボリューム
TCPIP_SEND_VOLUME	BIGINT	tcpip_send_volume - TCP/IP 送信ボリューム
TCPIP_RECV_WAIT_TIME	BIGINT	tcpip_recv_wait_time - TCP/IP 受信待ち時間
TCPIP_RECVS_TOTAL	BIGINT	tcpip_recvs_total - TCP/IP 合計受信数
TCPIP_SEND_WAIT_TIME	BIGINT	tcpip_send_wait_time - TCP/IP 送信待ち時間
TCPIP_SENDS_TOTAL	BIGINT	tcpip_sends_total - TCP/IP 合計送信数
TOTAL_APP_RQST_TIME	BIGINT	total_app_rqst_time - アプリケーション要求合計時間
TOTAL_RQST_TIME	BIGINT	total_rqst_time - 合計要求時間
WLM_QUEUE_TIME_TOTAL	BIGINT	wlm_queue_time_total - ワークロード・マネージャー合計キュー時間
WLM_QUEUE_ASSIGNMENTS _TOTAL	BIGINT	wlm_queue_assignments_total - ワークロード・マネージャー合計キュー割り当て数
TOTAL_CPU_TIME	BIGINT	total_cpu_time - 合計 CPU 時間
TOTAL_WAIT_TIME	BIGINT	total_wait_time - 合計待ち時間
APP_RQSTS_COMPLETED_TOTAL	BIGINT	app_rqsts_completed_total - 完了したアプリケーション要求の合計数
TOTAL_SECTION_SORT_TIME	BIGINT	total_section_sort_time - セクション・ソート時間合計
TOTAL_SECTION_SORT_PROC_TIME	BIGINT	total_section_sort_proc_time - セクション・ソート処理時間合計
TOTAL_SECTION_SORTS	BIGINT	total_section_sorts - セクション・ソート合計

表 139. MON_GET_WORKLOAD について戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
TOTAL_SORTS	BIGINT	total_sorts - ソート合計
POST_THRESHOLD_SORTS	BIGINT	post_threshold_sorts - ポストしきい値ソート
POST_SHRTHRESHOLD_SORTS	BIGINT	post_shrthreshold_sorts - ポスト共有しきい値ソート
SORT_OVERFLOWS	BIGINT	sort_overflows - ソート・オーバーフロー
TOTAL_COMPILE_TIME	BIGINT	total_compile_time - コンパイル時間の合計
TOTAL_COMPILE_PROC_TIME	BIGINT	total_compile_proc_time - コンパイル処理時間の合計
TOTAL_COMPILATIONS	BIGINT	total_compilations - コンパイルの合計回数
TOTAL_IMPLICIT_COMPILE_TIME	BIGINT	total_implicit_compile_time - 暗黙的コンパイル時間の合計
TOTAL_IMPLICIT_COMPILE _PROC_TIME	BIGINT	total_implicit_compile_proc_time - 暗黙的コンパイルの処理時間の合計
TOTAL_IMPLICIT_COMPILATIONS	BIGINT	total_implicit_compilations - 暗黙的コンパイルの合計回数
TOTAL_SECTION_TIME	BIGINT	total_section_time - セクション時間の合計
TOTAL_SECTION_PROC_TIME	BIGINT	total_section_proc_time - セクション処理時間の合計
TOTAL_APP_SECTION_EXECUTIONS	BIGINT	total_app_section_executions - セクション実行の合計回数
TOTAL_ACT_TIME	BIGINT	total_activity_time - 合計アクティビティー時間
TOTAL_ACT_WAIT_TIME	BIGINT	total_activity_wait_time - 合計アクティビティー待機時間
ACT_RQSTS_TOTAL	BIGINT	act_rqsts_total - アクティビティー要求の合計
TOTAL_ROUTINE_TIME	BIGINT	total_routine_time - ルーチン時間の合計
TOTAL_ROUTINE_INVOCATIONS	BIGINT	total_routine_invocations - ルーチン呼び出しの合計回数
TOTAL_COMMIT_TIME	BIGINT	total_commit_time - コミット時間の合計
TOTAL_COMMIT_PROC_TIME	BIGINT	total_commit_proc_time - コミット処理時間の合計
TOTAL_APP_COMMITS	BIGINT	total_app_commits - アプリケーション・コミットの合計回数
INT_COMMITS	BIGINT	int_commits - 内部コミット数
TOTAL_ROLLBACK_TIME	BIGINT	total_rollback_time - ロールバック時間の合計
TOTAL_ROLLBACK_PROC_TIME	BIGINT	total_rollback_proc_time - ロールバック処理時間の合計
TOTAL_APP_ROLLBACKS	BIGINT	total_app_rollbacks - アプリケーション・ロールバックの合計回数
INT_ROLLBACKS	BIGINT	int_rollbacks - 内部ロールバック数
TOTAL_RUNSTATS_TIME	BIGINT	total_runstats_time - ランタイム統計時間の合計
TOTAL_RUNSTATS_PROC_TIME	BIGINT	total_runstats_proc_time - ランタイム統計の処理時間の合計
TOTAL_RUNSTATS	BIGINT	total_runstats - ランタイム統計の合計回数
TOTAL_REORG_TIME	BIGINT	total_reorg_time - 再編成時間の合計
TOTAL_REORG_PROC_TIME	BIGINT	total_reorg_proc_time - 再編成の処理時間の合計
TOTAL_REORGS	BIGINT	total_reorgs - 再編成の合計回数

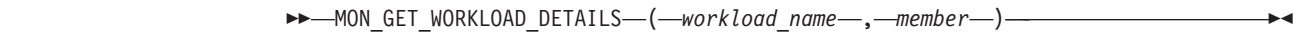
表 139. MON_GET_WORKLOAD について戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
TOTAL_LOAD_TIME	BIGINT	total_load_time - ロード時間の合計
TOTAL_LOAD_PROC_TIME	BIGINT	total_load_proc_time - ロード処理時間の合計
TOTAL_LOADS	BIGINT	total_loads - ロードの合計回数
CAT_CACHE_INSERTS	BIGINT	cat_cache_inserts - カタログ・キャッシュ挿入数
CAT_CACHE_LOOKUPS	BIGINT	cat_cache_lookups - カタログ・キャッシュ参照数
PKG_CACHE_INSERTS	BIGINT	pkg_cache_inserts - パッケージ・キャッシュ挿入
PKG_CACHE_LOOKUPS	BIGINT	pkg_cache_lookups - パッケージ・キャッシュ参照
THRESH_VIOLATIONS	BIGINT	hresh_violations - しきい値違反の回数
NUM_LW_THRESH_EXCEEDED	BIGINT	num_lw_thresh_exceeded - しきい値を超えた回数
ADDITIONAL_DETAILS	BLOB(100K)	将来の利用のために予約済み。

MON_GET_WORKLOAD_DETAILS 表関数 - 詳細ワークロード・メトリックの取得

MON_GET_WORKLOAD_DETAILS 表関数は、1 つ以上のワークロードの詳細メトリックを戻します。

構文



スキーマは SYSPROC です。

表関数パラメーター

workload_name
メトリックが戻される特定のワークロードを指定する、タイプ VARCHAR(128) の入力引数。引数が NULL または空ストリングである場合、すべてのワークロードについてメトリックが戻されます。

member
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なメンバーを指定する、タイプ INTEGER の入力引数。現行のデータベース・メンバーには -1、すべてのデータベース・メンバーには -2 を指定します。NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

MON_GET_WORKLOAD_DETAILS 関数に対する EXECUTE 特権。

例

メンバー全体で集約し、ロック待機時間の長い順に、ワークロードごとのロック情報を表示します。

```
SELECT varchar(wlmetrics.workload_name,30) as workload_name,
       sum(detmetrics.lock_wait_time) as total_lock_wait_time,
       sum(detmetrics.lock_waits) as total_lock_waits,
       sum(detmetrics.lock_timeouts) as total_lock_timeouts,
       sum(detmetrics.lock_escals) as total_lock_escals
FROM TABLE(MON_GET_WORKLOAD_DETAILS('',-2)) AS WLMETRICS,
XMLTABLE (XMLNAMESPACES( DEFAULT 'http://www.ibm.com/xmlns/prod/db2/mon'),
           '$detmetric/db2_workload' PASSING
           XMLPARSE(DOCUMENT WLMETRICS.DETAILS)
           as "detmetric"
COLUMNS "LOCK_WAIT_TIME" INTEGER PATH 'system_metrics/lock_wait_time',
        "LOCK_WAITS" INTEGER PATH 'system_metrics/lock_waits',
        "LOCK_TIMEOUTS" INTEGER PATH 'system_metrics/lock_timeouts',
        "LOCK_ESCALS" INTEGER PATH 'system_metrics/lock_escals'
) AS DETMETRICS
GROUP BY workload_name
ORDER BY total_lock_wait_time desc;
```

以下はこの照会の出力例です。

```
WORKLOAD_NAME                  TOTAL_LOCK_WAIT_TIME TOTAL_LOCK_WAITS ...
------------------------------ -------------------- ---------------- ...
SYSDEFAULTADMWORKLOAD                             0                0 ...
SYSDEFAULTUSERWORKLOAD                            0                0 ...

  2 record(s) selected.
```

照会の出力 (続き)。

```
... TOTAL_LOCK_TIMEOUTS TOTAL_LOCK_ESCALS
... ------------------- -----------------
...                   0                 0
...                   0                 0
```

使用上の注意

MON_GET_WORKLOAD 関数で戻されるメトリックは、識別されたワークロード・オブジェクトにマッピングされた接続によってサブミットされた要求のすべてのメトリックの累計を表します。この関数は、以下の点で MON_GET_WORKLOAD 表関数と似ています。

- MON_GET_WORKLOAD 表関数は、最も一般的に使用されるメトリックを列ベースの形式で戻すため、メトリックの取得において最も効率的な方法です。
- MON_GET_WORKLOAD_DETAILS 表関数は、使用可能なすべてのメトリック一式を XML 文書形式で戻すため、出力フォーマットを最大限柔軟性のあるものにします。XML ベースの出力は、XML パーサーで直接解析でき、XMLTABLE 関数でリレーショナル形式に変換することもできます (例を参照)。

メトリックは、作業単位境界でワークロードにロールアップされ、要求の実行中には定期的にロールアップされます。したがって、この表関数で報告される値は、直前の ROLLUP 時のシステムの現行状態を反映しています。メトリックの値は確実に増加します。ある時間間隔に対する指定されたメトリックの値を判別するには、MON_GET_WORKLOAD_DETAILS 表関数を使用してその間隔の始めと終わりのメトリックを照会し、差異を計算します。

要求メトリックは、サービス・スーパークラスに対する COLLECT REQUEST METRICS 節、およびデータベース・レベルの **mon_req_metrics** データベース構成パラメーターを介して制御されます。親サービス・スーパークラスで要求メトリックを使用可能にしているサービス・サブクラスのエージェントが要求を処理する場

合にのみ、またはデータベース全体で要求メトリック・コレクションが有効な場合にのみ、その要求に対しメトリックが収集されます。デフォルトでは、要求メトリックはデータベース・レベルで使用可能です。要求メトリックがデータベース・レベルでもサービス・スーパークラスに対しても使用不可になっている場合、そのサービス・スーパークラスにマッピングされたワークロードごとに報告されるメトリックは増加を停止します (または、要求メトリックがデータベースのアクティブ化時に無効であった場合には 0 のままになります)。

MON_GET_WORKLOAD_DETAILS 表関数は、ワークロードおよびメンバーごとに 1 行のデータを戻します。 (メンバー上の) ワークロード全体または (1 つ以上のサービス・クラスの) メンバー全体の集約は実行されません。ただし、集約は例に示されるように SQL 照会を使用して実行できます。

DETAILS 列に戻される XML 文書のスキーマは、ファイル sqllib/misc/DB2MonRoutines.xsd で入手できます。詳細は、ファイル sqllib/misc/DB2MonCommon.xsd 内にあります。

戻される情報

表 140. MON_GET_WORKLOAD_DETAILS について戻される情報

列名	データ・タイプ	説明
WORKLOAD_NAME	VARCHAR(128)	workload_name - ワークロード名
WORKLOAD_ID	INTEGER	workload_id - ワークロード ID
MEMBER	SMALLINT	member - データベース・メンバー
DETAILS	BLOB(1M)	ワークロードについての詳細メトリックを含む XML 文書。エレメントの説明については、本書の表 141を参照してください。

以下の例は、DETAILS 列で戻される XML 文書の構造を示しています。

```
<db2_workload xmlns="http://www.ibm.com/xmlns/prod/db2/mon" release="90700000">
   <workload_name>SYSDEFAULTADMWORKLOAD</workload_name>
   <workload_id>11</workload_id>
   <member>0</member>
   <system_metrics release="90700000">
   <act_aborted_total>5</act_aborted_total>
    ...
   <wlm_queue_assignments_total>3</wlm_queue_assignments_total>
   </system_metrics>
</db2_workload>
```

完全スキーマについては、sqllib/misc/DB2MonRoutines.xsd を参照してください。

表 141. MON_GET_WORKLOAD_DETAILS について戻される詳細メトリック

エレメント名	データ・タイプ	説明または対応するモニター・エレメント
workload_name	xs:string (128)	workload_name - ワークロード名
workload_id	xs:nonNegativeInteger	workload_id - ワークロード ID
member	xs:nonNegativeInteger	member - データベース・メンバー
act_aborted_total	xs:nonNegativeInteger	act_aborted_total - 打ち切られたアクティビティーの合計数
act_completed_total	xs:nonNegativeInteger	act_completed_total - 完了したアクティビティーの合計数

表 141. MON_GET_WORKLOAD_DETAILS について戻される詳細メトリック (続き)

エレメント名	データ・タイプ	説明または対応するモニター・エレメント
act_rejected_total	xs:nonNegativeInteger	act_rejected_total - リジェクトされたアクティビティーの合計数
act_rqsts_total	xs:nonNegativeInteger	act_rqsts_total - アクティビティー要求の合計
agent_wait_time	xs:nonNegativeInteger	agent_wait_time - エージェント待機時間
agent_waits_total	xs:nonNegativeInteger	agent_waits_total - エージェント待機の合計
app_rqsts_completed_total	xs:nonNegativeInteger	app_rqsts_completed_total - 完了したアプリケーション要求の合計数
audit_events_total	xs:nonNetagiveIngteger	audit_events_total - 監査イベント合計数
audit_subsystem_wait_time	xs:nonNegativeInteger	audit_subsystem_wait_time - 監査サブシステム待機時間
audit_subsystem_waits_total	xs:nonNegativeInteger	audit_subsystem_waits_total - 監査サブシステム待機の合計
audit_file_write_wait_time	xs:nonNegativeInteger	audit_file_write_wait_time - 監査ファイル書き込み待機時間
audit_file_writes_total	xs:nonNegativeInteger	audit_file_writes_total - 書き込まれた監査ファイルの合計数
cat_cache_inserts	xs:nonNegativeInteger	cat_cache_inserts - カタログ・キャッシュ挿入数
cat_cache_lookups	xs:nonNegativeInteger	cat_cache_lookups - カタログ・キャッシュ参照数
client_idle_wait_time	xs:nonNegativeInteger	client_idle_wait_time - クライアントのアイドル待機時間
deadlocks	xs:nonNegativeInteger	deadlocks - デッドロック検出数
diaglog_writes_total	xs:nonNegativeInteger	diaglog_writes_total - 診断ログ・ファイル書き込みの合計
diaglog_write_wait_time	xs:nonNegativeInteger	diaglog_write_wait_time - 診断ログ・ファイルの書き込み待機時間
direct_read_time	xs:nonNegativeInteger	direct_read_time - 直接読み取り時間
direct_write_time	xs:nonNegativeInteger	direct_write_time - 直接書き込み時間
direct_read_reqs	xs:nonNegativeInteger	direct_read_reqs - 直接読み取り要求
direct_reads	xs:nonNegativeInteger	direct_reads - データベースからの直接読み取り
direct_write_reqs	xs:nonNegativeInteger	direct_write_reqs - 直接書き込み要求
direct_writes	xs:nonNegativeInteger	direct_writes - データベースへの直接書き込み
fcm_recv_volume	xs:nonNegativeInteger	fcm_recv_volume - FCM 受信ボリューム
fcm_recv_wait_time	xs:nonNegativeInteger	fcm_recv_wait_time - FCM 受信待機時間
fcm_recvs_total	xs:nonNegativeInteger	fcm_recvs_total - FCM 合計受信数
fcm_message_recv_volume	xs:nonNegativeInteger	fcm_message_recv_volume - FCM メッセージ受信ボリューム
fcm_message_recvs_total	xs:nonNegativeInteger	fcm_message_recvs_total - FCM メッセージ合計受信数
fcm_message_recv_wait_time	xs:nonNegativeInteger	fcm_message_recv_wait_time - FCM メッセージの受信待機時間
fcm_message_send_volume	xs:nonNegativeInteger	fcm_message_send_volume - FCM メッセージ送信ボリューム
fcm_message_send_wait_time	xs:nonNegativeInteger	fcm_message_send_wait_time - FCM メッセージ送信待ち時間
fcm_message_sends_total	xs:nonNegativeInteger	fcm_message_sends_total - FCM メッセージ合計送信数
fcm_send_volume	xs:nonNegativeInteger	fcm_send_volume - FCM 送信ボリューム

表 141. MON_GET_WORKLOAD_DETAILS について戻される詳細メトリック (続き)

エレメント名	データ・タイプ	説明または対応するモニター・エレメント
fcm_send_wait_time	xs:nonNegativeInteger	fcm_send_wait_time - FCM 送信待ち時間
fcm_sends_total	xs:nonNegativeInteger	fcm_sends_total - FCM 合計送信数
fcm_tq_recv_wait_time	xs:nonNegativeInteger	fcm_tq_recv_wait_time - FCM 表キューの受信待ち時間
fcm_tq_send_wait_time	xs:nonNegativeInteger	fcm_tq_send_wait_time - FCM 表キューの送信待ち時間
fcm_tq_recv_volume	xs:nonNegativeInteger	fcm_tq_recv_volume - FCM 表キューの受信ボリューム
fcm_tq_recvs_total	xs:nonNegativeInteger	fcm_tq_recvs_total - FCM 表キューの合計受信数
fcm_tq_send_volume	xs:nonNegativeInteger	fcm_tq_send_volume - FCM 表キューの送信ボリューム
fcm_tq_sends_total	xs:nonNegativeInteger	fcm_tq_sends_total - FCM 表キューの合計送信数
int_commits	xs:nonNegativeInteger	int_commits - 内部コミット数
int_rollbacks	xs:nonNegativeInteger	int_rollbacks - 内部ロールバック数
tq_tot_send_spills	xs:nonNegativeInteger	tq_tot_send_spills - オーバーフローした表キュー・バッファーの合計数
ipc_recv_volume	xs:nonNegativeInteger	ipc_recv_volume - プロセス間通信受信ボリューム
ipc_recv_wait_time	xs:nonNegativeInteger	ipc_recv_wait_time - プロセス間通信受信待ち時間
ipc_recvs_total	xs:nonNegativeInteger	ipc_recvs_total - プロセス間通信合計受信数
ipc_send_volume	xs:nonNegativeInteger	ipc_send_volume - プロセス間通信送信ボリューム
ipc_send_wait_time	xs:nonNegativeInteger	ipc_send_wait_time - プロセス間通信送信待ち時間
ipc_sends_total	xs:nonNegativeInteger	ipc_sends_total - プロセス間通信合計送信数
lock_escals	xs:nonNegativeInteger	lock_escals - ロック・エスカレーション数
lock_timeouts	xs:nonNegativeInteger	lock_timeouts - ロック・タイムアウト数
lock_wait_time	xs:nonNegativeInteger	lock_wait_time - ロック待機中の時間
lock_waits	xs:nonNegativeInteger	lock_waits - ロック待機数
log_buffer_wait_time	xs:nonNegativeInteger	log_buffer_wait_time - ログ・バッファー待ち時間
log_disk_wait_time	xs:nonNegativeInteger	log_disk_wait_time - ログ・ディスク待機時間
log_disk_waits_total	xs:nonNegativeInteger	log_disk_waits_total - ログ・ディスク待機の合計
num_lw_thresh_exceeded	xs:nonNegativeInteger	num_lw_thresh_exceeded - しきい値を超えた回数
pkg_cache_inserts	xs:nonNegativeInteger	pkg_cache_inserts - パッケージ・キャッシュ挿入
pkg_cache_lookups	xs:nonNegativeInteger	pkg_cache_lookups - パッケージ・キャッシュ参照
pool_data_l_reads	xs:nonNegativeInteger	pool_data_l_reads - バッファー・プール・データの論理読み取り
pool_data_p_reads	xs:nonNegativeInteger	pool_data_p_reads - バッファー・プール・データの物理読み取り
pool_data_writes	xs:nonNegativeInteger	pool_data_writes - バッファー・プールへのデータの書き込み
pool_index_l_reads	xs:nonNegativeInteger	pool_index_l_reads - バッファー・プール索引の論理読み取り
pool_index_p_reads	xs:nonNegativeInteger	pool_index_p_reads - バッファー・プール索引の物理読み取り
pool_index_writes	xs:nonNegativeInteger	pool_index_writes - バッファー・プール索引の書き込み
pool_read_time	xs:nonNegativeInteger	pool_read_time - バッファー・プール物理読み取り時間の合計

表 141. MON_GET_WORKLOAD_DETAILS について戻される詳細メトリック (続き)

エレメント名	データ・タイプ	説明または対応するモニター・エレメント
pool_temp_data_l_reads	xs:nonNegativeInteger	pool_temp_data_l_reads - バッファー・プール一時データの論理読み取り
pool_temp_data_p_reads	xs:nonNegativeInteger	pool_temp_data_p_reads - バッファー・プール一時データの物理読み取り
pool_temp_index_l_reads	xs:nonNegativeInteger	pool_temp_index_l_reads - バッファー・プール一時索引の論理読み取り
pool_temp_index_p_reads	xs:nonNegativeInteger	pool_temp_index_p_reads - バッファー・プール一時索引の物理読み取り
pool_temp_xda_l_reads	xs:nonNegativeInteger	pool_temp_xda_l_reads - バッファー・プール一時 XDA データの論理読み取り
pool_temp_xda_p_reads	xs:nonNegativeInteger	pool_temp_xda_p_reads - バッファー・プール一時 XDA データの物理読み取り
pool_write_time	xs:nonNegativeInteger	pool_write_time - バッファー・プール物理書き込み時間の合計
pool_xda_l_reads	xs:nonNegativeInteger	pool_xda_l_reads - バッファー・プール XDA データの論理読み取り
pool_xda_p_reads	xs:nonNegativeInteger	pool_xda_p_reads - バッファー・プール XDA データの物理読み取り
pool_xda_writes	xs:nonNegativeInteger	pool_xda_writes - バッファー・プール XDA データの書き込み
num_log_buffer_full	xs:nonNegativeInteger	num_log_buffer_full - フル・ログ・バッファーの回数
rqsts_completed_total	xs:nonNegativeInteger	rqsts_completed_total - 完了した要求の合計数
rows_modified	xs:nonNegativeInteger	rows_modified - 変更された行数
rows_read	xs:nonNegativeInteger	rows_read - 読み取り行数
rows_returned	xs:nonNegativeInteger	rows_returned - 戻り行数
tcpip_recv_volume	xs:nonNegativeInteger	tcpip_recv_volume - TCP/IP 受信ボリューム
tcpip_recv_wait_time	xs:nonNegativeInteger	tcpip_recv_wait_time - TCP/IP 受信待ち時間
tcpip_recvs_total	xs:nonNegativeInteger	tcpip_recvs_total - TCP/IP 合計受信数
tcpip_send_volume	xs:nonNegativeInteger	tcpip_send_volume - TCP/IP 送信ボリューム
tcpip_send_wait_time	xs:nonNegativeInteger	tcpip_send_wait_time - TCP/IP 送信待ち時間
tcpip_sends_total	xs:nonNegativeInteger	tcpip_sends_total - TCP/IP 合計送信数
thresh_violations	xs:nonNegativeInteger	thresh_violations - しきい値違反の回数
total_act_time	xs:nonNegativeInteger	total_act_time - 合計アクティビティー時間
total_act_wait_time	xs:nonNegativeInteger	total_act_wait_time - 合計アクティビティー待機時間
total_app_commits	xs:nonNegativeInteger	total_app_commits - アプリケーション・コミットの合計回数
total_app_rollbacks	xs:nonNegativeInteger	total_app_rollbacks - アプリケーション・ロールバックの合計回数
total_app_rqst_time	xs:nonNegativeInteger	total_app_rqst_time - アプリケーション要求合計時間
total_app_section_executions	xs:nonNegativeInteger	total_app_section_executions - セクション実行の合計回数
total_commit_proc_time	xs:nonNegativeInteger	total_commit_proc_time - コミット処理時間の合計
total_commit_time	xs:nonNegativeInteger	total_commit_time - コミット時間の合計

表 141. MON_GET_WORKLOAD_DETAILS について戻される詳細メトリック (続き)

エレメント名	データ・タイプ	説明または対応するモニター・エレメント
total_compilations	xs:nonNegativeInteger	total_compilations - コンパイルの合計回数
total_compile_proc_time	xs:nonNegativeInteger	total_compile_proc_time - コンパイル処理時間の合計
total_compile_time	xs:nonNegativeInteger	total_compile_time - コンパイル時間の合計
total_cpu_time	xs:nonNegativeInteger	total_cpu_time - 合計 CPU 時間
total_implicit_compilations	xs:nonNegativeInteger	total_implicit_compilations - 暗黙的コンパイルの合計回数
total_implicit_compile_ proc_time	xs:nonNegativeInteger	total_implicit_compile_proc_time - 暗黙的コンパイルの処理時間の合計
total_implicit_compile_time	xs:nonNegativeInteger	total_implicit_compile_time - 暗黙的コンパイル時間の合計
total_loads	xs:nonNegativeInteger	total_loads - ロードの合計回数
total_load_proc_time	xs:nonNegativeInteger	total_load_proc_time - ロード処理時間の合計
total_load_time	xs:nonNegativeInteger	total_load_time - ロード時間の合計
total_reorgs	xs:nonNegativeInteger	total_reorgs - 再編成の合計回数
total_reorg_proc_time	xs:nonNegativeInteger	total_reorg_proc_time - 再編成の処理時間の合計
total_reorg_time	xs:nonNegativeInteger	total_reorg_time - 再編成時間の合計
total_rollback_proc_time	xs:nonNegativeInteger	total_rollback_proc_time - ロールバック処理時間の合計
total_rollback_time	xs:nonNegativeInteger	total_rollback_time - ロールバック時間の合計
total_routine_invocations	xs:nonNegativeInteger	total_routine_invocations - ルーチン呼び出しの合計回数
total_routine_time	xs:nonNegativeInteger	total_routine_time - ルーチン時間の合計
total_routine_user_ code_proc_time	xs:nonNegativeInteger	total_routine_user_code_proc_time - ルーチン・ユーザー・コード処理時間の合計
total_routine_user_ code_time	xs:nonNegativeInteger	total_routine_user_code_time - ルーチン・ユーザー・コード時間の合計
total_rqst_time	xs:nonNegativeInteger	total_rqst_time - 合計要求時間
total_runstats	xs:nonNegativeInteger	total_runstats - ランタイム統計の合計回数
total_runstats_proc_time	xs:nonNegativeInteger	total_runstats_proc_time - ランタイム統計の処理時間の合計
total_runstats_time	xs:nonNegativeInteger	total_runstats_time - ランタイム統計時間の合計
total_section_proc_time	xs:nonNegativeInteger	total_section_proc_time - セクション処理時間の合計
total_section_sort_time	xs:nonNegativeInteger	total_section_sort_time - セクション・ソート時間合計
total_section_sort_proc_time	xs:nonNegativeInteger	total_section_sort_proc_time - セクション・ソート処理時間合計
total_section_sorts	xs:nonNegativeInteger	total_section_sorts - セクション・ソート合計
total_section_time	xs:nonNegativeInteger	total_section_time - セクション時間の合計
total_sorts	xs:nonNegativeInteger	total_sorts - ソート合計
post_threshold_sorts	xs:nonNegativeInteger	post_threshold_sorts - ポストしきい値ソート
post_shrthreshold_sorts	xs:nonNegativeInteger	post_shrthreshold_sorts - ポスト共有しきい値ソート
sort_overflows	xs:nonNegativeInteger	sort_overflows - ソート・オーバーフロー
tq_tot_send_spills	xs:nonNegativeInteger	tq_tot_send_spills - オーバーフローした表キュー・バッファーの合計数
total_wait_time	xs:nonNegativeInteger	total_wait_time - 合計待ち時間

表 141. MON_GET_WORKLOAD_DETAILS について戻される詳細メトリック (続き)

エレメント名	データ・タイプ	説明または対応するモニター・エレメント
wlm_queue_time_total	xs:nonNegativeInteger	wlm_queue_time_total - ワークロード・マネージャー合計キュー時間
wlm_queue_assignments_total	xs:nonNegativeInteger	wlm_queue_assignments_total - ワークロード・マネージャー合計キュー割り当て数

MON_LOCKWAITS 管理ビュー - ロックの取得を待機しているアプリケーションに関するメトリックの取得

MON_LOCKWAITS 管理ビューは、現在接続しているデータベースでのロック取得を待機しているアプリケーションのために機能するエージェントについての情報を戻します。この照会はロックの問題を識別するうえで役立ちます。この管理ビューは SNAPLOCKWAIT 管理ビューに置き換わるものです。SNAPLOCKWAIT は DB2 バージョン 9.7 フィックスパック 1 で非推奨になり、将来のリリースでは廃止される可能性があります。

注: バージョン 9.7 のフィックスパック 1 以前で作成されたデータベースを使用している場合、このルーチンを実行するには、既に db2updv97 コマンドを実行済みでなければなりません。バージョン 9.7 以前に作成されたデータベースを使用している場合は、db2updv97 コマンドを実行する必要はありません (データベースのマイグレーションによって自動的にカタログ更新が実行されるため)。バージョン 9.7 にダウングレードすると、このルーチンは機能しなくなります。

許可

以下のいずれかの権限が必要です。

- MON_LOCKWAITS 管理ビューに対する SELECT 特権
- MON_LOCKWAITS 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

戻される情報

表 142. MON_LOCKWAITS 管理ビューによって戻される情報

列名	データ・タイプ	説明またはモニター・エレメント
LOCK_WAIT_ELAPSED_TIME	INTEGER	エージェントがロックの取得を待機し始めてから経過した時間。この値は秒単位です。

表 142. MON_LOCKWAITS 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明またはモニター・エレメント
LOCK_NAME	VARCHAR(32)	lock_name - ロック名 MON_FORMAT_LOCK_NAME ルーチンを使用することにより、この内部バイナリー・ロック名のフォーマットを設定したり、ロックに関する詳細情報 (例えば表ロックが参照する表と表スペースの情報) を取得したりできます。
LOCK_OBJECT_TYPE	VARCHAR(32)	lock_object_type - 待機中のロック対象タイプ
TABSCHEMA	VARCHAR(128)	table_schema - 表スキーマ名 表を参照しないロックの場合、NULL が戻されます。
TABNAME	VARCHAR(128)	table_name - 表名 表を参照しないロックの場合、NULL が戻されます。
DATA_PARTITION_ID	INTEGER	data_partition_id - データ・パーティション ID このエレメントは、パーティション表およびパーティション化索引にのみ該当します。ロック・レベル情報が戻されるとき、値 -1 は表全体へのアクセスを制御するロックを表します。
LOCK_MODE	VARCHAR(10)	lock_mode - ロック・モード
LOCK_CURRENT_MODE	VARCHAR(10)	lock_current_mode - 移行前の元のロック・モード LOCK_STATUS が "C" (移行) でない場合、NULL 値が戻されます。
LOCK_MODE_REQUESTED	VARCHAR(10)	lock_mode_requested - 要求されているロック・モード
REQ_APPLICATION_HANDLE	BIGINT	req_application_handle - 要求元のアプリケーション・ハンドル
REQ_AGENT_TID	BIGINT	req_agent_tid - 要求元のエージェント TID
REQ_MEMBER	SMALLINT	req_member - 要求元のメンバー

表 142. MON_LOCKWAITS 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明またはモニター・エレメント
REQ_APPLICATION_NAME	VARCHAR(128)	このロックの取得を待機しているクライアントで実行中のアプリケーションの名前。
REQ_USERID	VARCHAR(128)	このロックの取得を待機しているアプリケーションによって使用されているセッションの現在の許可 ID。
REQ_STMT_TEXT	CLOB(2MB)	ロックの取得を待機しているアプリケーションで実行されている SQL ステートメント・セクション。 非 SQL アクティビティーの場合、長さ 0 のストリング値が戻されます。
HLD_APPLICATION_HANDLE	BIGINT	hld_application_handle - 保持しているアプリケーション・ハンドル このロックを保持しているアプリケーションが不明または見つからない場合には、NULL 値が戻されます。
HLD_MEMBER	SMALLINT	hld_member - 保持しているメンバー
HLD_APPLICATION_NAME	VARCHAR(128)	このロックを保持しているクライアントで実行中のアプリケーションの名前。 このロックを保持しているアプリケーションが不明または見つからない場合には、長さ 0 のストリング値が戻されます。
HLD_USERID	VARCHAR(128)	このロックを保持しているアプリケーションによって使われているセッションの現在の許可 ID。
HLD_CURRENT_STMT_TEXT	CLOB(2MB)	ロックを保持しているアプリケーションに現在関連付けられている SQL ステートメント・テキスト。これは、ロックの原因となっているステートメントとは限らないことに注意してください。

MON_PKG_CACHE_SUMMARY - データベース・パッケージ・キャッシュの概要情報の取得

MON_PKG_CACHE_SUMMARY 管理ビューは、キャッシュ内の静的 SQL ステートメントと動的 SQL ステートメントの両方に関する主要なメトリックを戻し、データベース・パッケージ・キャッシュの概要情報を提供します。戻されるメトリックは、データベースの全メンバーのすべてのステートメント実行にわたって集計されます。

注: バージョン 9.7 のフィックスパック 1 以前で作成されたデータベースを使用している場合、このルーチンを実行するには、既に db2updv97 コマンドを実行済みでなければなりません。バージョン 9.7 以前に作成されたデータベースを使用している場合は、db2updv97 コマンドを実行する必要はありません (データベースのマイグレーションによって自動的にカタログ更新が実行されるため)。バージョン 9.7 にダウングレードすると、このルーチンは機能しなくなります。

スキーマは SYSIBMADM です。

許可

以下のいずれかの権限が必要です。

- MON_PKG_CACHE_SUMMARY 管理ビューに対する SELECT 特権
- MON_PKG_CACHE_SUMMARY 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

戻される情報

表 143. MON_PKG_CACHE_SUMMARY 管理ビューによって戻される情報

列名	データ・タイプ	説明またはモニター・エレメント
SECTION_TYPE	CHAR(1)	section_type - セクション・タイプ標識
EXECUTABLE_ID	VARCHAR(32) FOR BIT DATA	executable_id - 実行可能 ID
NUM_COORD_EXEC	BIGINT	num_coord_exec - コーディネーター・エージェントによる実行数
NUM_COORD_EXEC_WITH_METRICS	BIGINT	num_coord_exec_with_metrics - メトリックを伴うコーディネーター・エージェントによる実行数
TOTAL_STMT_EXEC_TIME	BIGINT	メトリックの収集対象となるすべてのステートメント実行にわたって、(ネストされたアクティビティーを含む) そのステートメントの実行に費やされた時間の合計 (ミリ秒)。

表 143. MON_PKG_CACHE_SUMMARY 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明またはモニター・エレメント
AVG_STMT_EXEC_TIME	BIGINT	メトリックの収集対象となるすべてのステートメント実行にわたって、(ネストされたアクティビティーを含む) そのステートメントの実行に費やされた時間の平均 (ミリ秒)。
TOTAL_CPU_TIME	BIGINT	DB2 データベース・マネージャー内で費やされた CPU 時間の合計 (マイクロ秒)。この値は、ユーザーとシステムの両方の CPU 時間の合計を表します。これは、そのステートメントのすべての total_cpu_time - 合計 CPU 時間値の累積として計算されます。
AVG_CPU_TIME	BIGINT	メトリックの収集対象となるすべてのステートメント実行にわたって、DB2 データベース・マネージャー内で費やされた CPU 時間の平均 (マイクロ秒)。
TOTAL_LOCK_WAIT_TIME	BIGINT	ロックの待機に費やされた経過時間の合計 (ミリ秒)。この値は、そのステートメントに関するすべての lock_wait_time - ロック待機中の時間値の合計として計算されます。
AVG_LOCK_WAIT_TIME	BIGINT	メトリックの収集対象となるすべてのステートメント実行にわたって、ロックの待機に費やされた経過時間の平均 (ミリ秒)。
TOTAL_IO_WAIT_TIME	BIGINT	入出力操作に費やされた経過時間の合計 (ミリ秒)。この値は、直接読み取りまたは直接書き込みの実行に必要とされた経過時間と、表スペース・コンテナーとの間のデータおよび索引ページの物理的読み取り/書き込みに費やされた経過時間の合計として計算されます。

表 143. MON_PKG_CACHE_SUMMARY 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明またはモニター・エレメント
AVG_IO_WAIT_TIME	BIGINT	メトリックの収集対象となるすべてのステートメント実行にわたって、入出力操作に費やされた経過時間の平均 (ミリ秒)。
PREP_TIME	BIGINT	prep_time - 準備時間
ROWS_READ_PER_ ROWS_RETURNED	BIGINT	メトリックの収集対象となるすべてのステートメント実行にわたって、戻された各行に対する読み取られた行数の平均。
STMT_TEXT	CLOB(2MB)	stmt_text - SQL 動的ステートメント・テキスト

MON_SERVICE_SUBCLASS_SUMMARY - すべてのサービス・サブクラスに関するメトリックの取得

MON_SERVICE_SUBCLASS_SUMMARY 管理ビューは、現在接続されているデータベース内のすべてのサービス・サブクラスに関する主要なメトリックを戻します。これはサービス・クラスごとに実行された処理を示すため、システムの概要をモニターするのに役立ちます。

注: バージョン 9.7 のフィックスパック 1 以前で作成されたデータベースを使用している場合、このルーチンを実行するには、既に db2updv97 コマンドを実行済みでなければなりません。バージョン 9.7 以前に作成されたデータベースを使用している場合は、db2updv97 コマンドを実行する必要はありません (データベースのマイグレーションによって自動的にカタログ更新が実行されるため)。バージョン 9.7 にダウングレードすると、このルーチンは機能しなくなります。

戻されるメトリックは、データベースの全メンバーにおいて、指定されたサービス・サブクラスのもとで実行された要求に関するすべてのメトリックの累計を表します。

スキーマは SYSIBMADM です。

許可

以下のいずれかの権限が必要です。

- MON_SERVICE_SUBCLASS_SUMMARY 管理ビューに対する SELECT 特権
- MON_SERVICE_SUBCLASS_SUMMARY 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

戻される情報

表 144. MON_SERVICE_SUBCLASS_SUMMARY 管理ビューによって戻される情報

列名	データ・タイプ	説明またはモニター・エレメント
SERVICE_SUPERCLASS _NAME	VARCHAR(128)	service_superclass_name - サービス・スーパークラス名
SERVICE_SUBCLASS_NAME	VARCHAR(128)	service_subclass_name - サービス・サブクラス名
SERVICE_CLASS_ID	INTEGER	service_class_id - サービス・クラス ID
TOTAL_APP_COMMITS	BIGINT	指定されたサービス・サブクラスに関する、データベースの全メンバーでのアプリケーション・コミットの総数。
TOTAL_APP_ROLLBACKS	BIGINT	指定されたサービス・サブクラスに関する、データベースの全メンバーでのアプリケーション・ロールバックの総数。
ACT_COMPLETED_TOTAL	BIGINT	指定されたサービス・サブクラスに関する、データベースの全メンバーにおける、正常に完了した任意のネスト・レベルのコーディネーター・アクティビティーの総数。
APP_RQSTS_COMPLETED _TOTAL	BIGINT	指定されたサービス・サブクラスに関する、データベースの全メンバーで正常に完了した外部 (アプリケーション) 要求の総数。
AVG_RQST_CPU_TIME	BIGINT	正常に完了したすべての外部要求によって費やされた平均 CPU 時間 (マイクロ秒)。これは、ユーザーとシステムの両方の CPU 時間の合計を表します。
ROUTINE_TIME_ RQST_PERCENT	DECIMAL(5,2)	データベース・サーバーが要求を処理するのに費やした時間のうち、ユーザー・ルーチンの実行に費やした時間のパーセンテージ。
RQST_WAIT_ TIME_PERCENT	DECIMAL(5,2)	要求を処理するのに費やされた時間のうち、DB2 データベース・サーバー内部での待機に費やされた時間のパーセンテージ。
ACT_WAIT_TIME_ PERCENT	DECIMAL(5,2)	アクティビティーの実行に費やされた時間のうち、DB2 データベース・サーバー内部での待機に費やされた時間のパーセンテージ。

表 144. MON_SERVICE_SUBCLASS_SUMMARY 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明またはモニター・エレメント
IO_WAIT_TIME_PERCENT	DECIMAL(5,2)	DB2 データベース・サーバー内部での待機に費やされた時間のうち、入出力操作に費やされた時間のパーセンテージ。これには、直接読み取り/直接書き込みの実行に費やされた時間に加えて、表スペースからバッファー・プールへのデータおよび索引ページの読み取りや元のディスクへの書き込みに費やされた時間が含まれます。
LOCK_WAIT_TIME_PERCENT	DECIMAL(5,2)	DB2 データベース・サーバー内部での待機に費やされた時間のうち、ロック待機に費やされた時間のパーセンテージ。
AGENT_WAIT_TIME_ PERCENT	DECIMAL(5,2)	DB2 データベース・サーバー内部での待機に費やされた時間のうち、コンセントレーター構成のもとでエージェントを待機するためにキューに入れられたアプリケーションによって費やされた時間のパーセンテージ。
NETWORK_WAIT_ TIME_PERCENT	DECIMAL(5,2)	DB2 データベース・サーバー内部での待機に費やされた時間のうち、クライアント/サーバー通信のために費やされた時間のパーセンテージ。これには、TCP/IP または IPC プロトコルを使ってデータを送受信するのに費やされた時間が含まれます。
SECTION_PROC_ TIME_PERCENT	DECIMAL(5,2)	要求をアクティブに処理するデータベース・サーバーが、セクションの実行に費やした時間のパーセンテージ。これには、ソートの実行に費やされた時間が含まれます。
SECTION_SORT_ PROC_TIME_PERCENT	DECIMAL(5,2)	要求をアクティブに処理するデータベース・サーバーが、セクションの実行中にソートの実行に費やした時間のパーセンテージ。
COMPILE_PROC_ TIME_PERCENT	DECIMAL(5,2)	要求をアクティブに処理するデータベース・サーバーが、SQL ステートメントのコンパイルに費やした時間のパーセンテージ。これには、明示的および暗黙的コンパイル時間が含まれます。

表 144. MON_SERVICE_SUBCLASS_SUMMARY 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明またはモニター・エレメント
TRANSACT_END_PROC _TIME_PERCENT	DECIMAL(5,2)	要求をアクティブに処理するデータベース・サーバーが、コミット処理の実行またはトランザクションのロールバックに費やした時間のパーセンテージ。
UTILS_PROC_ TIME_PERCENT	DECIMAL(5,2)	要求をアクティブに処理するデータベース・サーバーが、ユーティリティーの実行に費やした時間のパーセンテージ。これには、runstats、再編成、およびロード操作の実行が含まれます。
AVG_LOCK_WAITS _PER_ACT	BIGINT	各コーディネーター・アクティビティー (成功およびアボート) に対して、アプリケーションまたは接続がロックを待機した平均回数。
AVG_LOCK_TIMEOUTS _PER_ACT	BIGINT	各コーディネーター・アクティビティー (成功およびアボート) に対して、オブジェクトのロック要求がタイムアウトになった平均回数。
AVG_DEADLOCKS_ PER_ACT	BIGINT	コーディネーター・アクティビティー (成功およびアボート) ごとのデッドロックの平均数。
AVG_LOCK_ESCALS _PER_ACT	BIGINT	各コーディネーター・アクティビティー (成功およびアボート) に対して、いくつかの行ロックから表ロックにロックがエスカレートした平均回数。
ROWS_READ_PER_ ROWS_RETURNED	BIGINT	アプリケーションに戻された各行に対して、表から読み取られた平均行数。
TOTAL_BP_HIT_ RATIO_PERCENT	DECIMAL(5,2)	XML ストレージ・オブジェクト (XDA) の要求を含めて、データまたは索引ページの要求を処理するためにデータベース・マネージャーがディスクからページをロードする必要のなかった時間のパーセンテージ。

MON_TBSP_UTILIZATION - すべての表スペースとすべてのデータベース・パーティションに関するモニタリング・メトリックの取得

MON_TBSP_UTILIZATION 管理ビューは、現在接続されているデータベース内のすべての表スペースとすべてのデータベース・パーティションに関する、ヒット率や使用率パーセンテージなどの主要なモニタリング・メトリックを戻します。これにより、パフォーマンスとスペース使用状況をモニターするうえで重要な情報が提供されます。この管理ビューは、TBSP_UTILIZATION 管理ビューに置き換わるものです。

注: バージョン 9.7 のフィックスパック 1 以前で作成されたデータベースを使用している場合、このルーチンを実行するには、既に db2updv97 コマンドを実行済みでなければなりません。バージョン 9.7 以前に作成されたデータベースを使用している場合は、db2updv97 コマンドを実行する必要はありません (データベースのマイグレーションによって自動的にカタログ更新が実行されるため)。バージョン 9.7 にダウングレードすると、このルーチンは機能しなくなります。

許可

以下のいずれかの権限が必要です。

- MON_TBSP_UTILIZATION 管理ビューに対する SELECT 特権
- MON_TBSP_UTILIZATION 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

戻される情報

表 145. MON_TBSP_UTILIZATION 管理ビューによって戻される情報

列名	データ・タイプ	説明またはモニター・エレメント
TBSP_NAME	VARCHAR(128)	tablespace_name - 表スペース名
MEMBER	SMALLINT	member - データベース・メンバー
TBSP_TYPE	VARCHAR(10)	tablespace_type - 表スペース・タイプ。このインターフェースは、sqlutil.h の定義に基づくテキスト ID を戻します。これは次のいずれかです。 • DMS • SMS

表 145. MON_TBSP_UTILIZATION 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明またはモニター・エレメント
TBSP_CONTENT_TYPE	VARCHAR(10)	tablespace_content_type - 表スペースのコンテンツ・タイプ。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • ANY • LARGE • SYSTEMP • USRTEMP
TBSP_STATE	VARCHAR(256)	tablespace_state - 表スペースの状態
TBSP_PAGE_SIZE	BIGINT	tablespace_page_size - 表スペースのページ・サイズ
TBSP_EXTENT_SIZE	BIGINT	tablespace_extent_size - 表スペースのエクステント・サイズ
TBSP_PREFETCH_SIZE	BIGINT	tablespace_prefetch_size - 表スペースのプリフェッチ・サイズ
TBSP_USING_ AUTO_STORAGE	SMALLINT	tablespace_using_auto_storage - 自動ストレージが使用可能な表スペース
TBSP_AUTO_ RESIZE_ENABLED	SMALLINT	tablespace_auto_resize_enabled - 表スペースの自動サイズ変更可能
TBSP_TOTAL_SIZE_KB	BIGINT	表スペースの合計サイズ (キロバイト)。これは (*tablespace_total_pages* * *tablespace_page_size*) / 1024 として計算されます。ここで *tablespace_total_pages* と *tablespace_page_size* は以下のモニター・エレメントを表しています。 • tablespace_total_pages - 表スペース内の合計ページ数 • tablespace_page_size - 表スペースのページ・サイズ

表 145. MON_TBSP_UTILIZATION 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明またはモニター・エレメント
TBSP_USABLE_SIZE_KB	BIGINT	使用可能な表スペースの合計サイズ (キロバイト)。これは、表スペースの合計サイズからオーバーヘッド・ページに使われるスペースを差し引いた数値に等しくなります。これは (*tablespace_usable_pages* * *tablespace_page_size*) / 1024 として計算されます。ここで *tablespace_usable_pages* と *tablespace_page_size* は以下のモニター・エレメントを表しています。 • tablespace_usable_pages - 表スペース内の使用可能ページ数 • tablespace_page_size - 表スペースのページ・サイズ
TBSP_UTILIZATION _PERCENT	DECIMAL(5,2)	表スペースの使用率 (パーセンテージ)。*tablespace_usable_pages* がゼロより大きい場合、これは (*tablespace_used_pages* / *tablespace_usable_pages*) * 100 として計算されます。ここで *tablespace_used_pages* と *tablespace_usable_pages* は以下のモニター・エレメントを表しています。 • tablespace_used_pages - 表スペース内の使用されているページ数 • tablespace_usable_pages - 表スペース内の使用可能ページ数 *tablespace_usable_pages* がゼロより大きくない場合は NULL が戻されます。
TBSP_PAGE_TOP	BIGINT	tablespace_page_top - 表スペース最高水準点

表 145. MON_TBSP_UTILIZATION 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明またはモニター・エレメント
DATA_PHYSICAL_READS	BIGINT	TEMPORARY、REGULAR、および LARGE 表スペースに対して、表スペース・コンテナー (物理) から読み取られたデータ・ページの数を示します。これは (*pool_data_p_reads* + *pool_temp_data_p_reads*) として計算されます。ここで *pool_data_p_reads* と *pool_temp_data_p_reads* は以下のモニター・エレメントを表しています。 • pool_data_p_reads - バッファー・プール・データの物理読み取り • pool_temp_data_p_reads - バッファー・プール一時データの物理読み取り
DATA_HIT_ RATIO_PERCENT	DECIMAL(5,2)	データのヒット率。つまり、データ・ページ要求を処理するためにデータベース・マネージャーがディスクからページをロードする必要のなかった時間のパーセンテージ。
INDEX_PHYSICAL_READS	BIGINT	TEMPORARY、REGULAR、および LARGE 表スペースに対して、表スペース・コンテナー (物理) から読み取られた索引ページの数を示します。これは (*pool_index_p_reads* + *pool_temp_index_p_reads*) として計算されます。ここで *pool_index_p_reads* と *pool_temp_index_p_reads* は以下のモニター・エレメントを表しています。 • pool_index_p_reads - バッファー・プール索引の物理読み取り • pool_temp_index_p_reads - バッファー・プール一時索引の物理読み取り

表 145. MON_TBSP_UTILIZATION 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明またはモニター・エレメント
INDEX_HIT_RATIO _PERCENT	DECIMAL(5,2)	索引のヒット率。つまり、索引データ・ページ要求を処理するためにデータベース・マネージャーがディスクからページをロードする必要のなかった時間のパーセンテージ。
XDA_PHYSICAL_READS	BIGINT	TEMPORARY、REGULAR、および LARGE 表スペースに対して、表スペース・コンテナー (物理) から読み取られた XML ストレージ・オブジェクト (XDA) 用データ・ページの数を示します。これは (*pool_xda_p_reads* + *pool_temp_xda_p_reads*) として計算されます。ここで *pool_xda_p_reads* と *pool_temp_xda_p_reads* は以下のモニター・エレメントを表しています。 • pool_xda_p_reads - バッファー・プール XDA データの物理読み取り • pool_temp_xda_p_reads - バッファー・プール一時 XDA データの物理読み取り
XDA_HIT_RATIO_PERCENT	DECIMAL(5,2)	補助ストレージ・オブジェクトのヒット率。つまり、XML ストレージ・オブジェクト (XDA) に関するデータ・ページ要求を処理するためにデータベース・マネージャーがディスクからページをロードする必要のなかった時間のパーセンテージ。

MON_WORKLOAD_SUMMARY - すべてのワークロードに関するメトリックの取得

MON_WORKLOAD_SUMMARY 管理ビューは、現在接続されているデータベース内のすべてのワークロードに関する主要なメトリックを戻します。これはワークロードごとに受け取る処理を示すため、システムの概要をモニターするのに役立ちます。

注: バージョン 9.7 のフィックスパック 1 以前で作成されたデータベースを使用している場合、このルーチンを実行するには、既に db2updv97 コマンドを実行済みでなければなりません。バージョン 9.7 以前に作成されたデータベースを使用している場合は、db2updv97 コマンドを実行する必要はありません (データベースのマイグレーションによって自動的にカタログ更新が実行されるため)。バージョン 9.7 にダウングレードすると、このルーチンは機能しなくなります。

戻されるメトリックは、データベースの全メンバーにおいて、識別されたワークロード・オブジェクトにマップされる接続によってサブミットされた要求についてのすべてのメトリックの累計を表します。

スキーマは SYSIBMADM です。

許可

以下のいずれかの権限が必要です。

- MON_WORKLOAD_SUMMARY 管理ビューに対する SELECT 特権
- MON_WORKLOAD_SUMMARY 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

戻される情報

表 146. MON_WORKLOAD_SUMMARY 管理ビューによって戻される情報

列名	データ・タイプ	説明またはモニター・エレメント
WORKLOAD_NAME	VARCHAR(128)	workload_name - ワークロード名
WORKLOAD_ID	INTEGER	workload_id - ワークロード ID
TOTAL_APP_COMMITS	BIGINT	指定されたサービス・サブクラスに関する、データベースの全メンバーでのアプリケーション・コミットの総数。
TOTAL_APP_ROLLBACKS	BIGINT	指定されたサービス・サブクラスに関する、データベースの全メンバーでのアプリケーション・ロールバックの総数。
ACT_COMPLETED_TOTAL	BIGINT	指定されたサービス・サブクラスに関する、データベースの全メンバーにおける、正常に完了した任意のネスト・レベルのコーディネーター・アクティビティーの総数。
APP_RQSTS_COMPLETED_TOTAL	BIGINT	指定されたサービス・サブクラスに関する、データベースの全メンバーで正常に完了した外部 (アプリケーション) 要求の総数。

表 146. MON_WORKLOAD_SUMMARY 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明またはモニター・エレメント
AVG_RQST_CPU_TIME	BIGINT	正常に完了したすべての外部要求によって費やされた平均 CPU 時間 (マイクロ秒)。これは、ユーザーとシステムの両方の CPU 時間の合計を表します。
ROUTINE_TIME_RQST_PERCENT	DECIMAL(5,2)	データベース・サーバーが要求を処理するのに費やした時間のうち、ユーザー・ルーチンの実行に費やした時間のパーセンテージ。
RQST_WAIT_TIME_PERCENT	DECIMAL(5,2)	要求を処理するのに費やされた時間のうち、DB2 データベース・サーバー内部での待機に費やされた時間のパーセンテージ。
ACT_WAIT_TIME_PERCENT	DECIMAL(5,2)	アクティビティーの実行に費やされた時間のうち、DB2 データベース・サーバー内部での待機に費やされた時間のパーセンテージ。
IO_WAIT_TIME_PERCENT	DECIMAL(5,2)	DB2 データベース・サーバー内部での待機に費やされた時間のうち、入出力操作に費やされた時間のパーセンテージ。これには、直接読み取り/直接書き込みの実行に費やされた時間に加えて、表スペースからバッファー・プールへのデータおよび索引ページの読み取りや元のディスクへの書き込みに費やされた時間が含まれます。
LOCK_WAIT_TIME_PERCENT	DECIMAL(5,2)	DB2 データベース・サーバー内部での待機に費やされた時間のうち、ロック待機に費やされた時間のパーセンテージ。
AGENT_WAIT_TIME_PERCENT	DECIMAL(5,2)	DB2 データベース・サーバー内部での待機に費やされた時間のうち、コンセントレーター構成のもとでエージェントを待機するためにキューに入れられたアプリケーションによって費やされた時間のパーセンテージ。

表 146. MON_WORKLOAD_SUMMARY 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明またはモニター・エレメント
NETWORK_WAIT_ TIME_PERCENT	DECIMAL(5,2)	DB2 データベース・サーバー内部での待機に費やされた時間のうち、クライアント/サーバー通信のために費やされた時間のパーセンテージ。これには、TCP/IP または IPC プロトコルを使ってデータを送受信するのに費やされた時間が含まれます。
SECTION_PROC_ TIME_PERCENT	DECIMAL(5,2)	要求をアクティブに処理するデータベース・サーバーが、セクションの実行に費やした時間のパーセンテージ。これには、ソートの実行に費やされた時間が含まれます。
SECTION_SORT_ PROC_TIME_PERCENT	DECIMAL(5,2)	要求をアクティブに処理するデータベース・サーバーが、セクションの実行中にソートの実行に費やした時間のパーセンテージ。
COMPILE_PROC_ TIME_PERCENT	DECIMAL(5,2)	要求をアクティブに処理するデータベース・サーバーが、SQL ステートメントのコンパイルに費やした時間のパーセンテージ。これには、明示的および暗黙的コンパイル時間が含まれます。
TRANSACT_END_PROC _TIME_PERCENT	DECIMAL(5,2)	要求をアクティブに処理するデータベース・サーバーが、コミット処理の実行またはトランザクションのロールバックに費やした時間のパーセンテージ。
UTILS_PROC_ TIME_PERCENT	DECIMAL(5,2)	要求をアクティブに処理するデータベース・サーバーが、ユーティリティーの実行に費やした時間のパーセンテージ。これには、RUNSTATS、再編成、およびロード操作の実行が含まれます。
AVG_LOCK_WAITS _PER_ACT	BIGINT	各コーディネーター・アクティビティー (成功およびアボート) に対して、アプリケーションまたは接続がロックを待機した平均回数。

表 146. MON_WORKLOAD_SUMMARY 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明またはモニター・エレメント
AVG_LOCK_TIMEOUTS_PER_ACT	BIGINT	各コーディネーター・アクティビティー (成功およびアボート) に対して、オブジェクトのロック要求がタイムアウトになった平均回数。
AVG_DEADLOCKS_PER_ACT	BIGINT	コーディネーター・アクティビティー (成功およびアボート) ごとのデッドロックの平均数。
AVG_LOCK_ESCALS_PER_ACT	BIGINT	各コーディネーター・アクティビティー (成功およびアボート) に対して、いくつかの行ロックから表ロックにロックがエスカレートした平均回数。
ROWS_READ_PER_ROWS_RETURNED	BIGINT	アプリケーションに戻された各行に対して、表から読み取られた平均行数。
TOTAL_BP_HIT_RATIO_PERCENT	DECIMAL(5,2)	XML ストレージ・オブジェクト (XDA) の要求を含めて、データまたは索引ページの要求を処理するためにデータベース・マネージャーがディスクからページをロードする必要のなかった時間のパーセンテージ。

第 13 章 MQSeries ルーチン

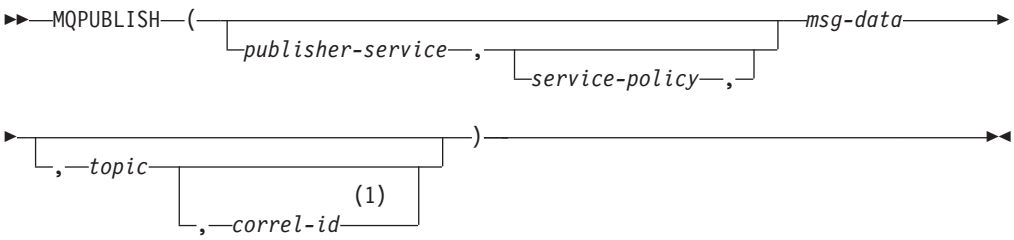
MQPUBLISH

MQPUBLISH 関数はデータを MQSeries にパブリッシュします。詳細については、http://www.ibm.com/software/MQSeries を参照してください。

MQPUBLISH 関数は、*msg-data* に含まれるデータを、 *publisher-service* に指定された MQSeries パブリッシャーにパブリッシュします。その際、*service-policy* に定義されたサービス・ポリシーの品質を使用します。 オプションで、メッセージ・トピックおよびユーザー定義のメッセージ相関 ID を指定することができます。

この結果のデータ・タイプは VARCHAR(1) です。関数の結果は、成功した場合は '1' で失敗した場合は '0' です。

構文



注:

1 *service* および *policy* が指定されていない限り、 *correl-id* を指定することはできません。

スキーマは、非トランザクション・メッセージ・キューイング関数の場合は DB2MQ、 1 フェーズ・コミット・トランザクション MQ 関数の場合は DB2MQ1C です。

関数のパラメーター

publisher-service
メッセージの送信先である MQSeries 論理宛先を含むストリング。 publisher-service を指定する場合、パブリッシャー・サービスのタイプ値が「P」の DB2MQ.MQPUBSUB 表に定義されているパブリッシャー・サービス・ポイントを参照する必要があります。 publisher-service を指定しない場合、 DB2.DEFAULT.PUBLISHER が使用されます。 *publisher-service* の最大サイズは 48 バイトです。

service-policy
このメッセージの処理に使われる MQSeriesサービス・ポリシーを含むストリング。 *service-policy* を指定する場合、DB2MQ.MQPOLICY 表に定義されたポリシーを参照する必要があります。 サービス・ポリシーは、このメッセージング操作に適用されるサービス・オプションの一連の品質を定義します。これらのオ

© Copyright IBM Corp. 2006, 2010

プションには、メッセージ優先順位やメッセージ・パーシスタンスが含まれます。 *service-policy* を指定しない場合、デフォルトの DB2.DEFAULT.POLICY が使用されます。 *service-policy* の最大サイズは 48 バイトです。

msg-data
MQSeries を介して送られるデータを含むストリング式。 VARCHAR ストリング式の最大サイズは 32 000 バイトで、CLOB ストリング式の最大サイズは 1M バイトです。

topic
メッセージ公表のトピックを含むストリング式。トピックを指定しない場合、メッセージにはトピックが関連付けられません。 *topic* の最大サイズは 40 バイトです。 1 つのストリングに複数のトピックを指定することができます (最大で 40 文字の長さ)。それぞれのトピックは、コロンで区切ります。例えば "t1:t2:the third topic" は、3 つのトピック t1、 t2、および "the third topic" がメッセージに関連付けられていることを示します。

correl-id
このメッセージに関連した相関 ID を含むオプションのストリング式。 *correl-id* は、要求/応答シナリオで要求と応答を関連付けるためにしばしば使用されます。これを指定しない場合、メッセージには相関 ID が追加されません。 *correl-id* の最大サイズは 24 バイトです。

例

例 1: この例では、ストリング "Testing 123" をデフォルト・ パブリッシャー・サービス (DB2.DEFAULT.PUBLISHER) にパブリッシュします。その際、デフォルト・ポリシー (DB2.DEFAULT.POLICY) を使用します。メッセージには相関 ID とトピックが指定されません。

```
VALUES MQPUBLISH('Testing 123')
```

例 2: この例では、トピック "TESTS" を使って、ストリング "Testing 345" をパブリッシャー・サービス "MYPUBLISHER" にパブリッシュします。デフォルト・ポリシーを使用し、相関 ID は指定しません。

```
VALUES MQPUBLISH('MYPUBLISHER','Testing 345', 'TESTS')
```

例 3: この例では、ストリング "Testing 678" をパブリッシャー・サービス "MYPUBLISHER" にパブリッシュします。その際、ポリシー"MYPOLICY" と相関 ID "TEST1" を使用します。メッセージは、トピック "TESTS" とともにパブリッシュされます。

```
VALUES MQPUBLISH('MYPUBLISHER','MYPOLICY','Testing 678','TESTS','TEST1')
```

例 4: この例では、ストリング "Testing 901" をパブリッシャー・サービス "MYPUBLISHER" にパブリッシュします。トピック "TESTS" とデフォルト・ポリシー (DB2.DEFAULT.POLICY) を使用し、相関 ID は使用しません。

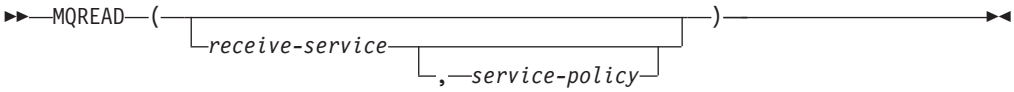
```
VALUES MQPUBLISH('Testing 901','TESTS')
```

MQREAD

MQREAD 関数は、*receive-service* で指定された MQSeries のロケーションからメッセージを戻します。その際、サービス・ポリシー *service-policy* で定義された品質を使用します。 この操作を実行しても、 *receive-service* に関連付けられたキューからメッセージは除去されません。ただし、キューの先頭にあるメッセージのみを戻します。

この結果のデータ・タイプは VARCHAR(32000) です。戻すメッセージが存在しない場合、結果は NULL 値になります。

構文



スキーマは、非トランザクション・メッセージ・キューイング関数の場合は DB2MQ、 1 フェーズ・コミット・トランザクション MQ 関数の場合は DB2MQ1C です。

関数のパラメーター

receive-service
受信するメッセージの送信元である MQSeries 論理宛先を含むストリング。 *receive-service* を指定する場合、DB2MQ.MQSERVICE 表に定義されたサービス・ポイントを参照する必要があります。 サービス・ポイントとは、メッセージが送受信される論理上のエンドポイントです。サービス・ポイントの定義には、MQSeries キュー・マネージャーおよびキューの名前が含まれます。 *receive-service* を指定しない場合、DB2.DEFAULT.SERVICE が使用されます。 *receive-service* の最大サイズは 48 バイトです。

service-policy
このメッセージの処理に使われる MQSeriesサービス・ポリシーを含むストリング。 *service-policy* を指定する場合、DB2MQ.MQPOLICY 表に定義されたポリシーを参照する必要があります。 サービス・ポリシーは、このメッセージング操作に適用されるサービス・オプションの一連の品質を定義します。これらのオプションには、メッセージ優先順位やメッセージ・パーシスタンスが含まれます。 *service-policy* を指定しない場合、デフォルトの DB2.DEFAULT.POLICY が使用されます。 *service-policy* の最大サイズは 48 バイトです。

例

例 1: この例では、デフォルト・サービス (DB2.DEFAULT.SERVICE) に指定されたキューの先頭にあるメッセージを読み取ります。その際、デフォルト・ポリシー (DB2.DEFAULT.POLICY) を使用します。

```
VALUES MQREAD()
```

例 2: この例では、サービス "MYSERVICE" に指定されたキューの先頭にあるメッセージを、デフォルト・ポリシー (DB2.DEFAULT.POLICY) を使用して読み取ります。

```
VALUES MQREAD('MYSERVICE')
```

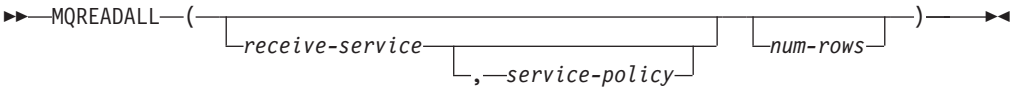
例 3: この例では、サービス "MYSERVICE" に指定されたキューの先頭にあるメッセージを、ポリシー "MYPOLICY" を使用して読み取ります。

```
VALUES MQREAD('MYSERVICE','MYPOLICY')
```

MQREADALL

MQREADALL 表関数は *receive-service* で指定された MQSeries のロケーションから、メッセージおよびメッセージ・メタデータを含む表を戻します。その際、サービス・ポリシー *service-policy* の品質を使用します。この操作を実行しても、*receive-service* に関連付けられたキューからメッセージは除去されません。

構文



スキーマは、非トランザクション・メッセージ・キューイング関数の場合は DB2MQ、 1 フェーズ・コミット・トランザクション MQ 関数の場合は DB2MQ1C です。

表関数パラメーター

receive-service
読み取る対象のメッセージが格納されている MQSeries 論理宛先を含むストリング。 *receive-service* を指定する場合、DB2MQ.MQSERVICE 表に定義されたサービス・ポイントを参照する必要があります。 サービス・ポイントとは、メッセージが送受信される論理上のエンドポイントです。サービス・ポイントの定義には、MQSeries キュー・マネージャーおよびキューの名前が含まれます。 *receive-service* を指定しない場合、DB2.DEFAULT.SERVICE が使用されます。 *receive-service* の最大サイズは 48 バイトです。

service-policy
このメッセージの処理に使われる MQSeriesサービス・ポリシーを含むストリング。 *service-policy* を指定する場合、DB2MQ.MQPOLICY 表に定義されたポリシーを参照させます。 サービス・ポリシーは、このメッセージング操作に適用されるサービス・オプションの一連の品質を定義します。これらのオプションには、メッセージ優先順位やメッセージ・パーシスタンスが含まれます。 *service-policy* を指定しない場合、デフォルトの DB2.DEFAULT.POLICY が使用されます。 *service-policy* の最大サイズは 48 バイトです。

num-rows
関数によって戻されるメッセージの最大数を示す正の整数。

num-rows を指定すると、最大で *num-rows* の数だけメッセージが戻されます。 *num-rows* を指定しない場合、入手可能なすべてのメッセージが戻されます。

許可

MQREADALL 表関数に対する EXECUTE 特権

例

例 1: この例では、デフォルト・サービス (DB2.DEFAULT.SERVICE) に指定されたキューからすべてのメッセージを受信します。その際、デフォルト・ポリシー (DB2.DEFAULT.POLICY) を使用します。 メッセージおよびすべてのメタデータが、1 つの表として戻されます。

```
SELECT * FROM table (MQREADALL()) AS T
```

例 2: この例では、サービス MYSERVICE に指定されたキューの先頭からすべてのメッセージを受信します。その際、デフォルト・ポリシー (DB2.DEFAULT.POLICY) を使用します。 MSG 列と CORRELID 列のみが戻されます。

```
SELECT T.MSG, T.CORRELID FROM table (MQREADALL('MYSERVICE')) AS T
```

例 3: この例では、デフォルト・サービス (DB2.DEFAULT.SERVICE) に指定されたキューの先頭を読み取ります。その際、デフォルト・ポリシー (DB2.DEFAULT.POLICY) を使用します。 CORRELID が '1234' のメッセージのみが戻されます。すべての列が戻されます。

```
SELECT * FROM table (MQREADALL()) AS T WHERE T.CORRELID = '1234'
```

例 4: この例では、デフォルト・サービス (DB2.DEFAULT.SERVICE) に指定されたキューの先頭から数えて最初の 10 個のメッセージを受信します。その際、デフォルト・ポリシー (DB2.DEFAULT.POLICY) を使用します。すべての列が戻されます。

```
SELECT * FROM table (MQREADALL(10)) AS T
```

戻される情報

表 147. MQREADALL 表関数によって戻される情報

列名	データ・タイプ	説明
MSG	VARCHAR(32000)	MQSeries メッセージの内容を含みます。
CORRELID	VARCHAR(24)	メッセージの識別に使用できる相関 ID を含みます。この ID を使用して、キューから特定メッセージを選択できます。要求と応答のシナリオでは、相関 ID を使用すると、応答を特定の要求と関連付けられるようになります。
TOPIC	VARCHAR(40)	メッセージがパブリッシュされたときのトピックを含みます (使用可能な場合)。
QNAME	VARCHAR(48)	メッセージが受信されたキューの名前を含みます。
MSGID	CHAR(24)	このメッセージの割り当てられた固有の MQSeries ID を含みます。

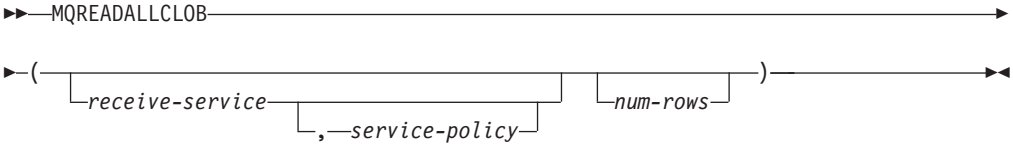
表 147. MQREADALL 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明
MSGFORMAT	VARCHAR(8)	MQSeries によって定義された、メッセージの形式を含みます。通常、ストリングの形式は MQSTR です。

MQREADALLCLOB

MQREADALLCLOB 表関数は *receive-service* で指定された MQSeries のロケーションから、メッセージおよびメッセージ・メタデータを含む表を戻します。その際、サービス・ポリシー *service-policy* の品質を使用します。この操作を実行しても、*receive-service* に関連付けられたキューからメッセージは除去されません。

構文



スキーマは DB2MQ です。

表関数パラメーター

receive-service
読み取る対象のメッセージが格納されている MQSeries 論理宛先を含むストリング。 *receive-service* を指定する場合、DB2MQ.MQSERVICE 表に定義されたサービス・ポイントを参照する必要があります。 サービス・ポイントとは、メッセージが送受信される論理上のエンドポイントです。サービス・ポイントの定義には、MQSeries キュー・マネージャーおよびキューの名前が含まれます。 *receive-service* を指定しない場合、DB2.DEFAULT.SERVICE が使用されます。 *receive-service* の最大サイズは 48 バイトです。

service-policy
このメッセージの処理に使われる MQSeriesサービス・ポリシーを含むストリング。 *service-policy* を指定する場合、DB2MQ.MQPOLICY 表に定義されたポリシーを参照させます。 サービス・ポリシーは、このメッセージング操作に適用されるサービス・オプションの一連の品質を定義します。これらのオプションには、メッセージ優先順位やメッセージ・パーシスタンスが含まれます。 *service-policy* を指定しない場合、デフォルトの DB2.DEFAULT.POLICY が使用されます。 *service-policy* の最大サイズは 48 バイトです。

num-rows
関数によって戻されるメッセージの最大数を示す正の整数。

num-rows を指定すると、最大で *num-rows* の数だけメッセージが戻されます。 *num-rows* を指定しない場合、入手可能なすべてのメッセージが戻されます。

許可

MQREADALLCLOB 表関数に対する EXECUTE 特権

例

例 1: この例では、デフォルト・サービス (DB2.DEFAULT.SERVICE) に指定されたキューからすべてのメッセージを受信します。その際、デフォルト・ポリシー (DB2.DEFAULT.POLICY) を使用します。 メッセージおよびすべてのメタデータが、1 つの表として戻されます。

```
SELECT * FROM table (MQREADALLCLOB()) AS T
```

例 2: この例では、サービス MYSERVICE に指定されたキューの先頭からすべてのメッセージを受信します。その際、デフォルト・ポリシー (DB2.DEFAULT.POLICY) を使用します。 MSG 列と CORRELID 列のみが戻されます。

```
SELECT T.MSG, T.CORRELID FROM table (MQREADALLCLOB('MYSERVICE')) AS T
```

例 3: この例では、デフォルト・サービス (DB2.DEFAULT.SERVICE) に指定されたキューの先頭を読み取ります。その際、デフォルト・ポリシー (DB2.DEFAULT.POLICY) を使用します。 CORRELID が '1234' のメッセージのみが戻されます。すべての列が戻されます。

```
SELECT * FROM table (MQREADALLCLOB()) AS T WHERE T.CORRELID = '1234'
```

例 4: この例では、デフォルト・サービス (DB2.DEFAULT.SERVICE) に指定されたキューの先頭から数えて最初の 10 個のメッセージを受信します。その際、デフォルト・ポリシー (DB2.DEFAULT.POLICY) を使用します。すべての列が戻されます。

```
SELECT * FROM table (MQREADALLCLOB(10)) AS T
```

戻される情報

表 148. MQREADALLCLOB 表関数によって戻される情報

列名	データ・タイプ	説明
MSG	CLOB(1M)	MQSeries メッセージの内容を含みます。
CORRELID	VARCHAR(24)	メッセージの識別に使用できる相関 ID を含みます。この ID を使用して、キューから特定メッセージを選択できます。要求と応答のシナリオでは、相関 ID を使用すると、応答を特定の要求と関連付けられるようになります。
TOPIC	VARCHAR(40)	メッセージがパブリッシュされたときのトピックを含みます (使用可能な場合)。
QNAME	VARCHAR(48)	メッセージが受信されたキューの名前を含みます。

表 148. MQREADALLCLOB 表関数によって戻される情報 (続き)

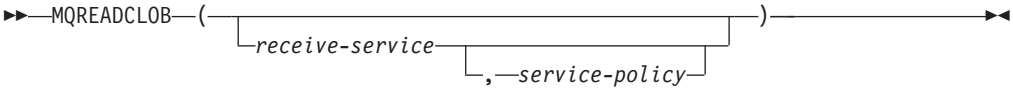
列名	データ・タイプ	説明
MSGID	CHAR(24)	このメッセージの割り当てられた固有の MQSeries ID を含みます。
MSGFORMAT	VARCHAR(8)	MQSeries によって定義された、メッセージの形式を含みます。通常、ストリングの形式は MQSTR です。

MQREADCLOB

MQREADCLOB 関数は、*receive-service* で指定された MQSeries のロケーションからメッセージを戻します。その際、サービス・ポリシー *service-policy* で定義された品質を使用します。 この操作を実行しても、 *receive-service* に関連付けられたキューからメッセージは除去されません。ただし、キューの先頭にあるメッセージのみを戻します。

結果のデータ・タイプは CLOB(1M) です。戻すメッセージが存在しない場合、結果は NULL 値になります。

構文



スキーマは DB2MQ です。

関数のパラメーター

receive-service
受信するメッセージの送信元である MQSeries 論理宛先を含むストリング。 *receive-service* を指定する場合、DB2MQ.MQSERVICE 表に定義されたサービス・ポイントを参照する必要があります。 サービス・ポイントとは、メッセージが送受信される論理上のエンドポイントです。サービス・ポイントの定義には、MQSeries キュー・マネージャーおよびキューの名前が含まれます。 *receive-service* を指定しない場合、DB2.DEFAULT.SERVICE が使用されます。 *receive-service* の最大サイズは 48 バイトです。

service-policy
このメッセージの処理に使われる MQSeriesサービス・ポリシーを含むストリング。 *service-policy* を指定する場合、DB2MQ.MQPOLICY 表に定義されたポリシーを参照する必要があります。 サービス・ポリシーは、このメッセージング操作に適用されるサービス・オプションの一連の品質を定義します。これらのオプションには、メッセージ優先順位やメッセージ・パーシスタンスが含まれます。 *service-policy* を指定しない場合、デフォルトの DB2.DEFAULT.POLICY が使用されます。 *service-policy* の最大サイズは 48 バイトです。

例

例 1: この例では、デフォルト・サービス (DB2.DEFAULT.SERVICE) に指定されたキューの先頭にあるメッセージを読み取ります。その際、デフォルト・ポリシー (DB2.DEFAULT.POLICY) を使用します。

```
VALUES MQREADCLOB()
```

例 2: この例では、サービス "MYSERVICE" に指定されたキューの先頭にあるメッセージを、デフォルト・ポリシー (DB2.DEFAULT.POLICY) を使用して読み取ります。

```
VALUES MQREADCLOB('MYSERVICE')
```

例 3: この例では、サービス "MYSERVICE" に指定されたキューの先頭にあるメッセージを、ポリシー "MYPOLICY" を使用して読み取ります。

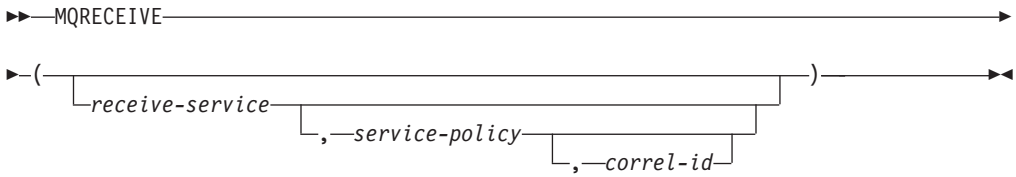
```
VALUES MQREADCLOB('MYSERVICE','MYPOLICY')
```

MQRECEIVE

MQRECEIVE 関数は、*receive-service* で指定された MQSeries のロケーションからメッセージを戻します。その際、サービス・ポリシー *service-policy* の品質を使用します。この操作を実行すると、*receive-service* に関連付けられたキューからメッセージが除去されます。 *correl-id* を指定すると、相関 ID が一致する最初のメッセージが戻されます。 *correl-id* を指定しない場合、キューの先頭にあるメッセージが戻されます。

この結果のデータ・タイプは VARCHAR(32000) です。戻すメッセージが存在しない場合、結果は NULL 値になります。

構文



スキーマは、非トランザクション・メッセージ・キューイング関数の場合は DB2MQ、 1 フェーズ・コミット・トランザクション MQ 関数の場合は DB2MQ1C です。

関数のパラメーター

receive-service

受信するメッセージの送信元である MQSeries 論理宛先を含むストリング。 *receive-service* を指定する場合、DB2MQ.MQSERVICE 表に定義されたサービス・ポイントを参照する必要があります。 サービス・ポイントとは、メッセージが送受信される論理上のエンドポイントです。サービス・ポイントの定義には、MQSeries キュー・マネージャーおよびキューの名前が含まれます。

receive-service を指定しない場合、DB2.DEFAULT.SERVICE が使用されます。 *receive-service* の最大サイズは 48 バイトです。

service-policy
このメッセージの処理に使われる MQSeriesサービス・ポリシーを含むストリング。 *service-policy* を指定する場合、DB2MQ.MQPOLICY 表に定義されたポリシーを参照する必要があります。サービス・ポリシーは、このメッセージング操作に適用されるサービス・オプションの一連の品質を定義します。これらのオプションには、メッセージ優先順位やメッセージ・パーシスタンスが含まれます。 *service-policy* を指定しない場合、デフォルトの DB2.DEFAULT.POLICY が使用されます。 *service-policy* の最大サイズは 48 バイトです。

correl-id
このメッセージに関連付けられた、オプションの相関 ID を含むストリング。 *correl-id* は、要求/応答シナリオで要求と応答を関連付けるためにしばしば使用されます。これを指定しない場合、相関 ID は指定されません。 *correl-id* の最大サイズは 24 バイトです。

例

例 1: この例では、デフォルト・サービス (DB2.DEFAULT.SERVICE) に指定されたキューの先頭にあるメッセージを受信します。その際、デフォルト・ポリシー (DB2.DEFAULT.POLICY) を使用します。

```
VALUES MQRECEIVE()
```

例 2: この例では、サービス "MYSERVICE" に指定されたキューの先頭にあるメッセージを、デフォルト・ポリシー (DB2.DEFAULT.POLICY) を使って受信します。

```
VALUES MQRECEIVE('MYSERVICE')
```

例 3: この例では、サービス "MYSERVICE" に指定されたキューの先頭にあるメッセージを、ポリシー "MYPOLICY" を使って受信します。

```
VALUES MQRECEIVE('MYSERVICE','MYPOLICY')
```

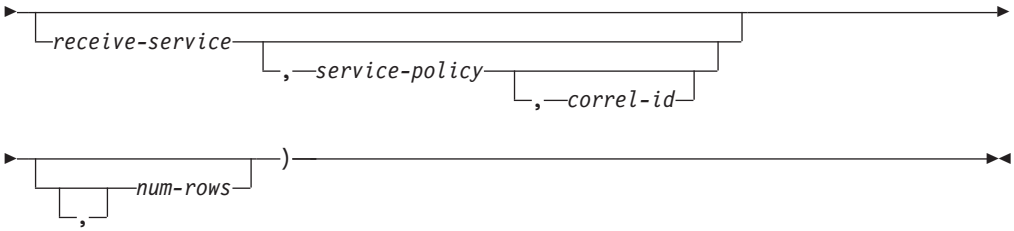
例 4: この例では、相関 ID '1234' に一致する最初のメッセージを、サービス "MYSERVICE" に指定されたキューの先頭から検索して受信します。その際、ポリシー "MYPOLICY" を使用します。

```
VALUES MQRECEIVE('MYSERVICE','MYPOLICY','1234')
```

MQRECEIVEALL

MQRECEIVEALL 表関数は *receive-service* で指定された MQSeries のロケーションから、メッセージおよびメッセージ・メタデータを含む表を戻します。その際、サービス・ポリシー *service-policy* の品質を使用します。この操作を実行すると、 *receive-service* に関連付けられたキューからメッセージが除去されます。

構文



スキーマは、非トランザクション・メッセージ・キューイング関数の場合は DB2MQ、 1 フェーズ・コミット・トランザクション MQ 関数の場合は DB2MQ1C です。

表関数パラメーター

receive-service

受信するメッセージの送信元である MQSeries 論理宛先を含むストリング。 *receive-service* を指定する場合、DB2MQ.MQSERVICE 表に定義されたサービス・ポイントを参照する必要があります。 サービス・ポイントとは、メッセージが送受信される論理上のエンドポイントです。サービス・ポイントの定義には、MQSeries キュー・マネージャーおよびキューの名前が含まれます。 *receive-service* を指定しない場合、DB2.DEFAULT.SERVICE が使用されます。 *receive-service* の最大サイズは 48 バイトです。

service-policy

このメッセージの処理に使われる MQSeriesサービス・ポリシーを含むストリング。 *service-policy* を指定する場合、DB2MQ.MQPOLICY 表に定義されたポリシーを参照させます。 サービス・ポリシーは、このメッセージング操作に適用されるサービス・オプションの一連の品質を定義します。これらのオプションには、メッセージ優先順位やメッセージ・パーシスタンスが含まれます。 *service-policy* を指定しない場合、デフォルトの DB2.DEFAULT.POLICY が使用されます。 *service-policy* の最大サイズは 48 バイトです。

correl-id

このメッセージに関連した相関 ID を含むオプション・ストリング。 *correl-id* は、要求/応答シナリオで要求と応答を関連付けるためにしばしば使用されます。これを指定しない場合、相関 ID は指定されません。 correl-id の最大サイズは 24 バイトです。

correl-id を指定すると、相関 ID が一致するすべてのメッセージが戻され、キューから除去されます。 *correl-id* を指定しない場合、キューの先頭にあるメッセージが戻されます。

num-rows

関数によって戻されるメッセージの最大数を示す正の整数。

num-rows を指定すると、最大で *num-rows* の数だけメッセージが戻されます。 *num-rows* を指定しない場合、入手可能なすべてのメッセージが戻されます。

許可

MQRECEIVEALL 表関数に対する EXECUTE 特権

例

例 1: この例では、デフォルト・サービス (DB2.DEFAULT.SERVICE) に指定されたキューからすべてのメッセージを受信します。その際、デフォルト・ポリシー (DB2.DEFAULT.POLICY) を使用します。 メッセージおよびすべてのメタデータが、1 つの表として戻されます。

```
SELECT * FROM table (MQRECEIVEALL()) AS T
```

例 2: この例では、サービス MYSERVICE に指定されたキューの先頭からすべてのメッセージを受信します。その際、デフォルト・ポリシー (DB2.DEFAULT.POLICY) を使用します。 MSG 列と CORRELID 列のみが戻されます。

```
SELECT T.MSG, T.CORRELID FROM table (MQRECEIVEALL('MYSERVICE')) AS T
```

例 3: この例では、ポリシー "MYPOLICY" を使用して、サービス "MYSERVICE" に指定されたキューの先頭からすべてのメッセージを受信します。 CORRELID が '1234' のメッセージのみが戻されます。 MSG 列と CORRELID 列のみが戻されます。

```
SELECT T.MSG, T.CORRELID FROM table
   (MQRECEIVEALL('MYSERVICE','MYPOLICY','1234')) AS T
```

例 4: この例では、デフォルト・サービス (DB2.DEFAULT.SERVICE) に指定されたキューの先頭から数えて最初の 10 個のメッセージを受信します。その際、デフォルト・ポリシー (DB2.DEFAULT.POLICY) を使用します。すべての列が戻されます。

```
SELECT * FROM table (MQRECEIVEALL(10)) AS T
```

戻される情報

表 149. MQRECEIVEALL 表関数によって戻される情報

列名	データ・タイプ	説明
MSG	VARCHAR(32000)	MQSeries メッセージの内容を含みます。
CORRELID	VARCHAR(24)	メッセージの識別に使用できる相関 ID を含みます。この ID を使用して、キューから特定メッセージを選択できます。要求と応答のシナリオでは、相関 ID を使用すると、応答を特定の要求と関連付けられるようになります。
TOPIC	VARCHAR(40)	メッセージがパブリッシュされたときのトピックを含みます (使用可能な場合)。
QNAME	VARCHAR(48)	メッセージが受信されたキューの名前を含みます。
MSGID	CHAR(24)	このメッセージの割り当てられた固有の MQSeries ID を含みます。

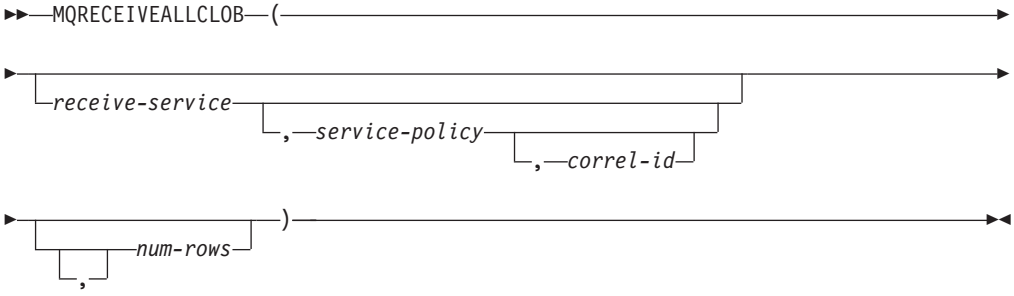
表 149. MQRECEIVEALL 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明
MSGFORMAT	VARCHAR(8)	MQSeries によって定義された、メッセージの形式を含みます。通常、ストリングの形式は MQSTR です。

MQRECEIVEALLCLOB

MQRECEIVEALLCLOB 表関数は *receive-service* で指定された MQSeries のロケーションから、メッセージおよびメッセージ・メタデータを含む表を戻します。その際、サービス・ポリシー *service-policy* の品質を使用します。この操作を実行すると、 *receive-service* に関連付けられたキューからメッセージが除去されます。

構文



スキーマは DB2MQ です。

表関数パラメーター

receive-service
受信するメッセージの送信元である MQSeries 論理宛先を含むストリング。 *receive-service* を指定する場合、DB2MQ.MQSERVICE 表に定義されたサービス・ポイントを参照する必要があります。 サービス・ポイントとは、メッセージが送受信される論理上のエンドポイントです。サービス・ポイントの定義には、MQSeries キュー・マネージャーおよびキューの名前が含まれます。 *receive-service* を指定しない場合、DB2.DEFAULT.SERVICE が使用されます。 *receive-service* の最大サイズは 48 バイトです。

service-policy
このメッセージの処理に使われる MQSeriesサービス・ポリシーを含むストリング。 *service-policy* を指定する場合、DB2MQ.MQPOLICY 表に定義されたポリシーを参照させます。サービス・ポリシーは、このメッセージング操作に適用されるサービス・オプションの一連の品質を定義します。これらのオプションには、メッセージ優先順位やメッセージ・パーシスタンスが含まれます。 *service-policy* を指定しない場合、デフォルトの DB2.DEFAULT.POLICY が使用されます。 *service-policy* の最大サイズは 48 バイトです。

correl-id
このメッセージに関連した相関 ID を含むオプション・ストリング。 *correl-id*

は、要求/応答シナリオで要求と応答を関連付けるためにしばしば使用されます。これを指定しない場合、相関 ID は指定されません。 correl-id の最大サイズは 24 バイトです。

correl-id を指定すると、相関 ID が一致するメッセージのみが戻されます。*correl-id* を指定しない場合、キューの先頭にあるメッセージが戻されます。

num-rows
関数によって戻されるメッセージの最大数を示す正の整数。

num-rows を指定すると、最大で *num-rows* の数だけメッセージが戻されます。*num-rows* を指定しない場合、入手可能なすべてのメッセージが戻されます。

許可

MQRECEIVEALLCLOB 表関数に対する EXECUTE 特権

例

例 1: この例では、デフォルト・サービス (DB2.DEFAULT.SERVICE) に指定されたキューからすべてのメッセージを受信します。その際、デフォルト・ポリシー (DB2.DEFAULT.POLICY) を使用します。 メッセージおよびすべてのメタデータが、1 つの表として戻されます。

```
SELECT * FROM table (MQRECEIVEALLCLOB()) AS T
```

例 2: この例では、サービス MYSERVICE に指定されたキューの先頭からすべてのメッセージを受信します。その際、デフォルト・ポリシー (DB2.DEFAULT.POLICY) を使用します。 MSG 列と CORRELID 列のみが戻されます。

```
SELECT T.MSG, T.CORRELID
   FROM table (MQRECEIVEALLCLOB('MYSERVICE')) AS T
```

例 3: この例では、ポリシー "MYPOLICY" を使用して、サービス "MYSERVICE" に指定されたキューの先頭からすべてのメッセージを受信します。 CORRELID が '1234' のメッセージのみが戻されます。 MSG 列と CORRELID 列のみが戻されます。

```
SELECT T.MSG, T.CORRELID
   FROM table (MQRECEIVEALLCLOB('MYSERVICE','MYPOLICY','1234')) AS T
```

例 4: この例では、デフォルト・サービス (DB2.DEFAULT.SERVICE) に指定されたキューの先頭から数えて最初の 10 個のメッセージを受信します。その際、デフォルト・ポリシー (DB2.DEFAULT.POLICY) を使用します。すべての列が戻されます。

```
SELECT * FROM table (MQRECEIVEALLCLOB(10)) AS T
```

戻される情報

表 150. MQRECEIVEALLCLOB 表関数によって戻される情報

列名	データ・タイプ	説明
MSG	CLOB(1M)	MQSeries メッセージの内容を含みます。

表 150. MQRECEIVEALLCLOB 表関数によって戻される情報 (続き)

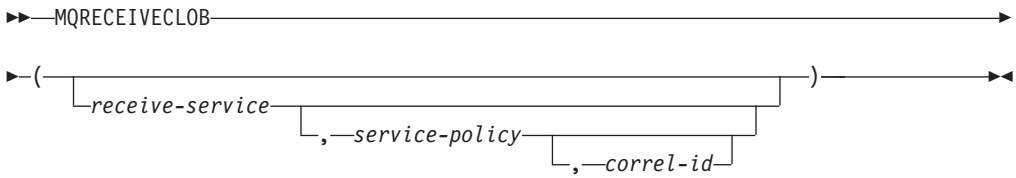
列名	データ・タイプ	説明
CORRELID	VARCHAR(24)	メッセージの識別に使用できる相関 ID を含みます。この ID を使用して、キューから特定メッセージを選択できます。要求と応答のシナリオでは、相関 ID を使用すると、応答を特定の要求と関連付けられるようになります。
TOPIC	VARCHAR(40)	メッセージがパブリッシュされたときのトピックを含みます (使用可能な場合)。
QNAME	VARCHAR(48)	メッセージが受信されたキューの名前を含みます。
MSGID	CHAR(24)	このメッセージの割り当てられた固有の MQSeries ID を含みます。
MSGFORMAT	VARCHAR(8)	MQSeries によって定義された、メッセージの形式を含みます。通常、ストリングの形式は MQSTR です。

MQRECEIVECLOB

MQRECEIVECLOB 関数は、*receive-service* で指定された MQSeries のロケーションからメッセージを戻します。その際、サービス・ポリシー *service-policy* の品質を使用します。この操作を実行すると、*receive-service* に関連付けられたキューからメッセージが除去されます。 *correl-id* を指定すると、相関 ID が一致する最初のメッセージが戻されます。 *correl-id* を指定しない場合、キューの先頭にあるメッセージが戻されます。

結果のデータ・タイプは CLOB(1M) です。戻すメッセージが存在しない場合、結果は NULL 値になります。

構文



スキーマは DB2MQ です。

関数のパラメーター

receive-service
受信するメッセージの送信元である MQSeries 論理宛先を含むストリング。 *receive-service* を指定する場合、DB2MQ.MQSERVICE 表に定義されたサービス・ポイントを参照する必要があります。 サービス・ポイントとは、メッセージが送受信される論理上のエンドポイントです。サービス・ポイントの定義には、MQSeries キュー・マネージャーおよびキューの名前が含まれます。 *receive-service* を指定しない場合、DB2.DEFAULT.SERVICE が使用されます。 *receive-service* の最大サイズは 48 バイトです。

service-policy
このメッセージの処理に使われる MQSeriesサービス・ポリシーを含むストリング。 *service-policy* を指定する場合、DB2MQ.MQPOLICY 表に定義されたポリシーを参照する必要があります。 サービス・ポリシーは、このメッセージング操作に適用されるサービス・オプションの一連の品質を定義します。これらのオプションには、メッセージ優先順位やメッセージ・パーシスタンスが含まれます。 *service-policy* を指定しない場合、デフォルトの DB2.DEFAULT.POLICY が使用されます。 *service-policy* の最大サイズは 48 バイトです。

correl-id
このメッセージに関連付けられた、オプションの相関 ID を含むストリング。 *correl-id* は、要求/応答シナリオで要求と応答を関連付けるためにしばしば使用されます。これを指定しない場合、相関 ID は使用されません。 *correl-id* の最大サイズは 24 バイトです。

例

例 1: この例では、デフォルト・サービス (DB2.DEFAULT.SERVICE) に指定されたキューの先頭にあるメッセージを受信します。その際、デフォルト・ポリシー (DB2.DEFAULT.POLICY) を使用します。

```
VALUES MQRECEIVECLOB()
```

例 2: この例では、サービス "MYSERVICE" に指定されたキューの先頭にあるメッセージを、デフォルト・ポリシー (DB2.DEFAULT.POLICY) を使って受信します。

```
VALUES MQRECEIVECLOB('MYSERVICE')
```

例 3: この例では、サービス "MYSERVICE" に指定されたキューの先頭にあるメッセージを、ポリシー "MYPOLICY" を使って受信します。

```
VALUES MQRECEIVECLOB('MYSERVICE','MYPOLICY')
```

例 4: この例では、相関 ID '1234' に一致する最初のメッセージを、サービス "MYSERVICE" に指定されたキューの先頭から検索して受信します。その際、ポリシー "MYPOLICY" を使用します。

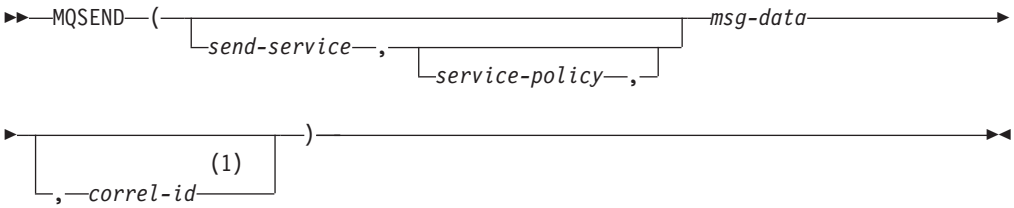
```
VALUES MQRECEIVECLOB('MYSERVICE',MYPOLICY','1234')
```

MQSEND

MQSEND 関数は *msg-data* に含まれているデータを、 *send-service* に指定された MQSeries のロケーションに送ります。その際、*service-policy* に定義されたサービス・ポリシーの品質を使用します。 オプションのユーザー定義メッセージ相関 ID は、 *correl-id* を使用して指定できます。

この結果のデータ・タイプは VARCHAR(1) です。関数の結果は、成功した場合は '1' で失敗した場合は '0' です。

構文



注:

1 *service* および *policy* が指定されていない限り、 *correl-id* を指定することはできません。

スキーマは、非トランザクション・メッセージ・キューイング関数の場合は DB2MQ、 1 フェーズ・コミット・トランザクション MQ 関数の場合は DB2MQ1C です。

関数のパラメーター

msg-data
MQSeries を介して送られるデータを含むストリング式。 VARCHAR ストリング式の最大サイズは 32 000 バイトで、CLOB ストリング式の最大サイズは 1M バイトです。

send-service
メッセージの送信先である MQSeries 論理宛先を含むストリング。 *send-service* を指定する場合、DB2MQ.MQSERVICE 表に定義されたサービス・ポイントを参照させます。 サービス・ポイントとは、メッセージが送受信される論理上のエンドポイントです。サービス・ポイントの定義には、MQSeries キュー・マネージャーおよびキューの名前が含まれます。 *send-service* を指定しない場合、DB2.DEFAULT.SERVICE の値が使用されます。 *send-service* の最大サイズは 48 バイトです。

service-policy
このメッセージの処理に使われる MQSeriesサービス・ポリシーを含むストリング。 *service-policy* を指定する場合、DB2MQ.MQPOLICY 表に定義されたサービス・ポリシーを参照する必要があります。 サービス・ポリシーは、このメッセージング操作に適用されるサービス・オプションの一連の品質を定義します。これらのオプションには、メッセージ優先順位やメッセージ・パーシスタンスが

含まれます。 *service-policy* を指定しない場合、デフォルト値 DB2.DEFAULT.POLICY が使用されます。 *service-policy* の最大サイズは 48 バイトです。

correl-id
このメッセージに関連した相関 ID を含むオプション・ストリング。 *correl-id* は、要求/応答シナリオで要求と応答を関連付けるためにしばしば使用されます。これを指定しない場合、相関 ID は送信されません。 *correl-id* の最大サイズは 24 バイトです。

例

例 1: この例では、ストリング "Testing 123" をデフォルト・サービス (DB2.DEFAULT.SERVICE) に送ります。その際、デフォルト・ポリシー (DB2.DEFAULT.POLICY) を使用し、相関 ID は使用しません。

```
VALUES MQSEND('Testing 123')
```

例 2: この例では、ストリング "Testing 345" をサービス "MYSERVICE" に送ります。その際、ポリシー "MYPOLICY" を使用し、相関 ID は使用しません。

```
VALUES MQSEND('MYSERVICE','MYPOLICY','Testing 345')
```

例 3: この例では、ストリング "Testing 678" をサービス "MYSERVICE" に送ります。その際、ポリシー "MYPOLICY" と相関 ID "TEST3" を使用します。

```
VALUES MQSEND('MYSERVICE','MYPOLICY','Testing 678','TEST3')
```

例 4: この例では、ストリング "Testing 901" をサービス "MYSERVICE" に送ります。その際、デフォルト・ポリシー (DB2.DEFAULT.POLICY) を使用し、相関 ID は使用しません。

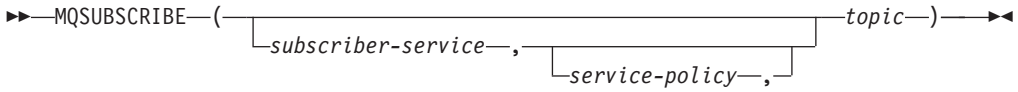
```
VALUES MQSEND('MYSERVICE','Testing 901')
```

MQSUBSCRIBE

MQSUBSCRIBE 関数は、指定されたトピックで公表された MQSeries メッセージに対する関心事を登録するために使用されます。この関数が正常に実行されると、パブリッシュ/サブスクライブ・サーバーは、トピックに一致するメッセージを、*subscriber-service* に定義されたサービス・ポイントに転送するようになります。*subscriber-service* は、指定されたトピックに一致するメッセージの論理宛先を指定します。 *topic* に一致するメッセージは *subscriber-service* に定義されたキューの中に入れられて、その後 MQREAD、 MQRECEIVE、 MQREADALL、または MQRECEIVEALL の呼び出し時に読み取りまたは受信することができます。詳細については、`http://www.ibm.com/software/MQSeries` を参照してください。

この結果のデータ・タイプは VARCHAR(1) です。関数の結果は、成功した場合は '1' で失敗した場合は '0' です。

構文



スキーマは、非トランザクション・メッセージ・キューイング関数の場合は DB2MQ、 1 フェーズ・コミット・トランザクション MQ 関数の場合は DB2MQ1C です。

関数のパラメーター

subscriber-service

topic に一致するメッセージの送信先である MQSeries 論理サブスクリプション・ポイントを含むストリング。 *subscriber-service* を指定する場合、パブリッシャー・サービスのタイプ値が「S」の DB2MQ.MQPUBSUB 表に定義されているサブスクライバー・サービス・ポイントを参照する必要があります。 *subscriber-service* を指定しない場合、代わりに DB2.DEFAULT.SUBSCRIBER が使用されます。 subscriber-service の最大サイズは 48 バイトです。

service-policy

メッセージの処理に使われる MQSeriesサービス・ポリシーを含むストリング。 *service-policy* を指定する場合、DB2MQ.MQPOLICY 表に定義されたポリシーを参照する必要があります。 サービス・ポリシーは、このメッセージング操作に適用されるサービス・オプションの一連の品質を定義します。これらのオプションには、メッセージ優先順位やメッセージ・パーシスタンスが含まれます。 *service-policy* を指定しない場合、代わりにデフォルト DB2.DEFAULT.POLICY が使用されます。 service-policy の最大サイズは 48 バイトです。

topic

受信するメッセージのタイプを定義するストリング。指定されたトピックでパブリッシュされたメッセージのみが、このサブスクリプションによって受信されます。複数のサブスクリプションを同時に存在させることができます。 topic の最大サイズは 40 バイトです。 1 つのストリングに複数のトピックを指定することができます (最大で 40 バイトの長さ)。それぞれのトピックは、コロンで区切ります。例えば "t1:t2:the third topic" は、3 つのトピック t1、 t2、および "the third topic" がメッセージに関連付けられていることを示します。

例

例 1: この例では、トピック "Weather" を含んでいるメッセージに対する関心事を登録します。デフォルト・サブスクライバー・サービス (DB2.DEFAULT.SUBSCRIBER) がサブスクライバーとして登録され、デフォルト・サービス・ポリシー (DB2.DEFAULT.POLICY) でサービスの品質を指定します。

```
VALUES MQSUBSCRIBE('Weather')
```

例 2: この例では、"Stocks" を含むメッセージに対する関心事をサブスクライバーが登録します。サブスクライバーはポリシー "BASIC-POLICY" を使用し、"PORTFOLIO-UPDATES" として登録します。

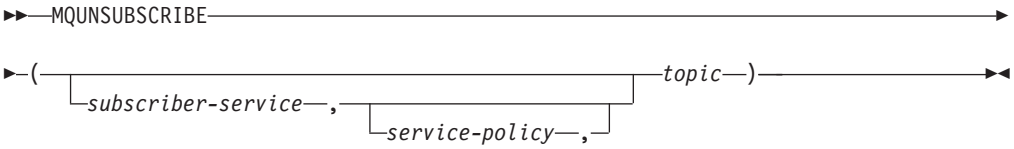
```
VALUES MQSUBSCRIBE('PORTFOLIO-UPDATES','BASIC-POLICY','Stocks')
```

MQUNSUBSCRIBE

MQUNSUBSCRIBE 関数は、既存のメッセージ・サブスクリプションを登録抹消するために使用されます。 *subscriber-service*、*service-policy*、および *topic* を使って、取り消す予定のサブスクリプションを識別します。この関数が正常に実行されると、パブリッシュ/サブスクライブ・サーバーは、指定したサブスクリプションを除去するようになります。指定された *topic* を持つメッセージは、 *subscriber-service* に定義された論理宛先にもはや送信されなくなります。詳細については、 `http://www.ibm.com/software/MQSeries` を参照してください。

この結果のデータ・タイプは VARCHAR(1) です。関数の結果は、成功した場合は '1' で失敗した場合は '0' です。

構文



スキーマは、非トランザクション・メッセージ・キューイング関数の場合は DB2MQ、 1 フェーズ・コミット・トランザクション MQ 関数の場合は DB2MQ1C です。

関数のパラメーター

subscriber-service

subscriber-service を指定する場合、パブリッシャー・サービスのタイプ値が「S」の DB2MQ.MQPUBSUB 表に定義されているサブスクライバー・サービス・ポイントを参照する必要があります。 *subscriber-service* を指定しない場合、代わりに DB2.DEFAULT.SUBSCRIBER が使用されます。 subscriber-service の最大サイズは 48 バイトです。

service-policy

service-policy を指定する場合、DB2MQ.MQPOLICY 表に定義されたポリシーを参照する必要があります。 サービス・ポリシーは、このメッセージング操作に適用されるサービス・オプションの一連の品質を定義します。 *service-policy* を指定しない場合、デフォルトの DB2.DEFAULT.POLICY が使用されます。 *service-policy* の最大サイズは 48 バイトです。

topic

もはや受信しないメッセージの主題を指定するストリング。 *topic* の最大サイズは 40 バイトです。 1 つのストリングに複数のトピックを指定することができます (最大で 40 バイトの長さ)。それぞれのトピックは、コロンで区切ります。例えば "t1:t2:the third topic" は、3 つのトピック t1、 t2、および "the third topic" がメッセージに関連付けられていることを示します。

例

例 1: この例では、トピック "Weather" を含んでいるメッセージに対する関心を取り消します。デフォルト・サブスクライバー・サービス

(DB2.DEFAULT.SUBSCRIBER) がアンサブスクライバーとして登録され、デフォルト・サービス・ポリシー (DB2.DEFAULT.POLICY) でサービスの品質を指定します。

```
VALUES MQUNSUBSCRIBE('Weather')
```

例 2: この例では、"Stocks" を含むメッセージに対する関心をサブスクライバーが取り消します。 サブスクライバーはポリシー "BASIC-POLICY" を使用し、"PORTFOLIO-UPDATES" として登録されています。

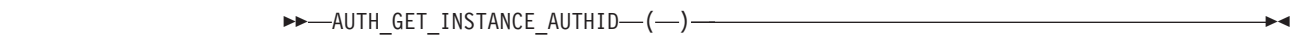
```
VALUES MQUNSUBSCRIBE('PORTFOLIO-UPDATES','BASIC-POLICY','Stocks')
```

第 14 章 セキュリティー・ルーチンおよびビュー

AUTH_GET_INSTANCE_AUTHID - インスタンス所有者の許可 ID の取得

AUTH_GET_INSTANCE_AUTHID スカラー関数はインスタンス所有者の許可 ID を戻します

構文



スキーマは SYSPROC です。

許可

AUTH_GET_INSTANCE_AUTHID スカラー関数に対する EXECUTE 特権。

例

次の例は、DB2 コマンド行プロセッサー (CLP) を使用してインスタンス所有者の許可 ID を取得する方法を示しています。

```
db2 "values SYSPROC.AUTH_GET_INSTANCE_AUTHID()"
```

このコマンドの出力例は、次のとおりです。

```
1
-------------------------...-------------
ZURBIE

  1 record(s) selected.
```

使用上の注意

一般的な構成では、インスタンス所有者アカウントが SYSADM グループのメンバーとして含まれます。このため DB2 バージョン 9.7 より前では、インスタンス所有者アカウントのもとで実行されるアプリケーションはデータベースに対する無制限の権限を持っていました。 DB2 バージョン 9.7 では、SYSADM 権限を持つユーザーは暗黙的な DBADM 権限を持たなくなりました。その結果、インスタンス所有者アカウントのもとで実行されるアプリケーションでは、SYSADM 権限のスコープに含まれなくなった操作を実行すると SQL1092N、SQL0551N、SQL0552N などの許可エラーが発生する可能性があります。

UPGRADE DATABASE コマンドと RESTORE DATABASE コマンド (下位レベルのデータベース用) は、DBADM 権限を SYSADM グループに付与しますが、新しいバージョン 9.7 データベースにはこれが該当しません。

インスタンス所有者の許可 ID が保持している権限のリストを取得するには、次の手順に従います。

© Copyright IBM Corp. 2006, 2010

1. SYSPROC.AUTH_GET_INSTANCE_AUTHID() スカラー関数を使用して、インスタンス所有者の許可 ID を判別します。以下に例を示します。

```
db2 "VALUES SYSPROC.AUTH_GET_INSTANCE_AUTHID()"
```

このコマンドにより、以下が戻されます。

```
1
------------------------------------
BOB

  1 record(s) selected.
```

2. この許可 ID が持つ権限のリストを取得します。以下に例を示します。

```
SELECT * FROM
   TABLE (SYSPROC.AUTH_LIST_AUTHORITIES_FOR_AUTHID ('BOB', 'U') ) AS T
   ORDER BY AUTHORITY
```

3. 必要に応じて、欠落している権限を付与します。以下に例を示します。

```
GRANT DBADM ON DATABASE TO USER BOB
```

戻される情報

表 151. AUTH_GET_INSTANCE_AUTHID スカラー関数によって戻される情報

列名	データ・タイプ	説明
InstanceAuthId	VARCHAR(128)	インスタンス所有者の許可 ID。

AUTH_LIST_AUTHORITIES_FOR_AUTHID

AUTH_LIST_AUTHORITIES_FOR_AUTHID 表関数は、データベース構成ファイルにあるか、許可 ID に直接付与されたか、あるいはグループまたはロールを介して間接的に付与された許可 ID によって保持されているすべての権限を戻します。

構文

スキーマは SYSPROC です。

表関数パラメーター

authid
照会対象の許可 ID を指定する、タイプ VARCHAR(128) の入力引数。許可 ID はユーザー、グループ、またはロールです。*authid* が NULL または空ストリングである場合、空の結果表が戻されます。

authidtype
照会対象の許可 ID を指定する、タイプ VARCHAR(1) の入力引数。 *authidtype* が存在しないか、NULL または空ストリングである場合、空の結果表が戻されます。 *authidtype* に考えられる値は以下のとおりです。

- G: グループ
- R: ロール

- U: ユーザー

許可

AUTH_LIST_AUTHORITIES_FOR_AUTHID 関数に対する EXECUTE 特権。

戻される情報

表 152. AUTH_LIST_AUTHORITIES_FOR_AUTHID について戻される情報

列名	データ・タイプ	説明
AUTHORITY	VARCHAR(128)	許可 ID によって保持される権限。
D_USER	CHAR(1)	*authidtype* がユーザー (U) の場合に、*authid* に直接付与される権限。 *authidtype* がグループ (G) またはロール (R) の場合、値は適用外 ('*') です。 • N = 保有しない • Y= 保有する • * = 適用外
D_GROUP	CHAR(1)	*authidtype* がグループ (G) の場合に、*authid* に直接付与される権限、または *authidtype* がユーザー (U) の場合に *authid* が属するグループに付与される権限。 *authidtype* がロール (R) の場合、値は適用外 ('*') です。 • N = 保有しない • Y= 保有する • * = 適用外
D_PUBLIC	CHAR(1)	*authidtype* がユーザー (U) またはグループ (G) の場合に、PUBLIC という *authid* に直接付与される権限。 *authidtype* がロール (R) の場合、値は適用外 ('*') です。 • N = 保有しない • Y= 保有する • * = 適用外
ROLE_USER	CHAR(1)	*authidtype* がユーザー (U) の場合に、*authid* に付与されたロールに直接付与される権限。 *authidtype* がグループ (G) またはロール (R) の場合、値は適用外 ('*') です。ロールはロール階層の一部である可能性があります。 • N = 保有しない • Y= 保有する • * = 適用外
ROLE_GROUP	CHAR(1)	*authidtype* がグループ (G) の場合に、*authid* に付与されたロールに直接付与される権限。 *authidtype* がユーザー (U) またはロール (R) の場合、値は適用外 ('*') です。ロールはロール階層の一部である可能性があります。 • N = 保有しない • Y= 保有する • * = 適用外

表 152. AUTH_LIST_AUTHORITIES_FOR_AUTHID について戻される情報 (続き)

列名	データ・タイプ	説明
ROLE_PUBLIC	CHAR(1)	*authidtype* がユーザー (U) またはグループ (G) の場合に、PUBLIC という *authid* に付与されたロールに直接付与される権限。 *authidtype* がロール (R) の場合、値は適用外 ('*') です。ロールはロール階層の一部である可能性があります。 • N = 保有しない • Y= 保有する • * = 適用外
D_ROLE	CHAR(1)	ロールに付与された権限、またはそのロールに付与されているロールに付与された権限。 *authidtype* がユーザー (U) またはグループ (G) の場合、値は適用外 ('*') です。ロールはロール階層の一部である可能性があります。 • N = 保有しない • Y= 保有する • * = 適用外

例

特殊グループ PUBLIC を介してデフォルトで BIND、CONNECT、CREATETAB、および IMPLICIT_SCHEMA 特権を保持するユーザー ALICE について考慮します。 ALICE は、SYSADM、SYSCTRL、および SYSMAINT というシステム権限を持つグループ ADMIN1 のメンバーです。また、DBADM 権限を持つグループ ADMIN2 のメンバーでもあります。さらに、DBADM および SECADM データベース権限も付与されています。ロール R1 が ALICE に付与されました。 LOAD 権限がロール R1 に付与されました。ロール R2 がグループ ADMIN1 に付与されました。 CREATE_NOT_FENCED_ROUTINE 権限がロール R2 に付与されました。

例 1: ユーザー ALICE が自分自身に直接、またはグループ PUBLIC またはロールを介して間接に付与したすべての権限を取得します。

```
SELECT AUTHORITY, D_USER, D_GROUP, D_PUBLIC, ROLE_USER, ROLE_GROUP, ROLE_PUBLIC, D_ROLE
   FROM TABLE (SYSPROC.AUTH_LIST_AUTHORITIES_FOR_AUTHID ('ALICE', 'U') ) AS T
   ORDER BY AUTHORITY


AUTHORITY                D_USER D_GROUP D_PUBLIC ROLE_USER ROLE_GROUP ROLE_PUBLIC D_ROLE
------------------------ ------ ------- -------- --------- ---------- ----------- ------
ACCESSCTRL               N      N       N        N         N          N           *
BINDADD                  N      N       Y        N         N          N           *
CONNECT                  N      N       Y        N         N          N           *
CREATE_EXTERNAL_ROUTINE  N      N       N        N         N          N           *
CREATE_NOT_FENCED_ROUTINE N     N       N        N         Y          N           *
CREATETAB                N      N       Y        N         N          N           *
DATAACCESS               N      N       N        N         N          N           *
DBADM                    Y      Y       N        N         N          N           *
EXPLAIN                  N      N       N        N         N          N           *
IMPLICIT_SCHEMA          N      N       Y        N         N          N           *
LOAD                     N      N       N        Y         N          N           *
QUIESCE_CONNECT          N      N       N        N         N          N           *
SECADM                   Y      N       N        N         N          N           *
SQLADM                   N      N       N        N         N          N           *
SYSADM                   *      Y       *        *         *          *           *
SYSCTRL                  *      Y       *        *         *          *           *
SYSMAINT                 *      Y       *        *         *          *           *
SYSMON                   *      N       *        *         *          *           *
WLMADM                   N      N       N        N         N          N           *
```

例 2: グループ ADMIN1 がそれ自身に直接、または PUBLIC またはロールを介して間接に付与したすべての権限を取得します。

```
SELECT AUTHORITY, D_USER, D_GROUP, D_PUBLIC, ROLE_USER, ROLE_GROUP, ROLE_PUBLIC, D_ROLE
   FROM TABLE (SYSPROC.AUTH_LIST_AUTHORITIES_FOR_AUTHID ('ADMIN1', 'G') ) AS T
   ORDER BY AUTHORITY

AUTHORITY                D_USER D_GROUP D_PUBLIC ROLE_USER ROLE_GROUP ROLE_PUBLIC D_ROLE
------------------------ ------ ------- -------- --------- ---------- ----------- ------
ACCESSCTRL               *      N       *        *         N          *           *
BINDADD                  *      N       *        *         N          *           *
CONNECT                  *      N       *        *         N          *           *
CREATE_EXTERNAL_ROUTINE  *      N       *        *         N          *           *
CREATE_NOT_FENCED_ROUTINE *     N       *        *         Y          *           *
CREATETAB                *      N       *        *         N          *           *
DATAACCESS               *      N       *        *         N          *           *
DBADM                    *      N       *        *         N          *           *
EXPLAIN                  *      N       *        *         N          *           *
IMPLICIT_SCHEMA          *      N       *        *         N          *           *
LOAD                     *      N       *        *         N          *           *
QUIESCE_CONNECT          *      N       *        *         N          *           *
SECADM                   *      N       *        *         N          *           *
SQLADM                   *      N       *        *         N          *           *
SYSADM                   *      Y       *        *         *          *           *
SYSCTRL                  *      Y       *        *         *          *           *
SYSMAINT                 *      Y       *        *         *          *           *
SYSMON                   *      N       *        *         *          *           *
WLMADM                   *      N       *        *         N          *           *
```

例 3: 特殊グループ PUBLIC がそれ自身に直接、またはロールを介して間接に付与したすべての権限を取得します。

```
SELECT AUTHORITY, D_USER, D_GROUP, D_PUBLIC, ROLE_USER, ROLE_GROUP, ROLE_PUBLIC, D_ROLE
   FROM TABLE (SYSPROC.AUTH_LIST_AUTHORITIES_FOR_AUTHID ('PUBLIC', 'G') ) AS T
   ORDER BY AUTHORITY

1                        D_USER D_GROUP D_PUBLIC ROLE_USER ROLE_GROUP ROLE_PUBLIC D_ROLE
------------------------ ------ ------- -------- --------- ---------- ----------- ------
ACCESSCTRL               *      *       N        *         *          N           *
BINDADD                  *      *       Y        *         *          N           *
CONNECT                  *      *       Y        *         *          N           *
CREATE_EXTERNAL_ROUTINE  *      *       N        *         *          N           *
CREATE_NOT_FENCED_ROUTINE *     *       N        *         *          N           *
CREATETAB                *      *       Y        *         *          N           *
DATAACCESS               *      *       N        *         *          N           *
DBADM                    *      *       N        *         *          N           *
EXPLAIN                  *      *       N        *         *          N           *
IMPLICIT_SCHEMA          *      *       Y        *         *          N           *
LOAD                     *      *       N        *         *          N           *
QUIESCE_CONNECT          *      *       N        *         *          N           *
SECADM                   *      *       N        *         *          N           *
SQLADM                   *      *       N        *         *          N           *
SYSADM                   *      *       *        *         *          *           *
SYSCTRL                  *      *       *        *         *          *           *
SYSMAINT                 *      *       *        *         *          *           *
SYSMON                   *      *       *        *         *          *           *
WLMADM                   *      *       N        *         *          N           *
```

例 4: ロール R1 がそれ自身に直接、またはロールを介して間接に付与したすべての権限を取得します。この場合、ロール R2 もロール R1 に付与されたことを考慮してください。

```
SELECT AUTHORITY, D_USER, D_GROUP, D_PUBLIC, ROLE_USER, ROLE_GROUP, ROLE_PUBLIC, D_ROLE
   FROM TABLE (SYSPROC.AUTH_LIST_AUTHORITIES_FOR_AUTHID ('R1', 'R') ) AS T
   ORDER BY AUTHORITY

AUTHORITY                D_USER D_GROUP D_PUBLIC ROLE_USER ROLE_GROUP ROLE_PUBLIC D_ROLE
------------------------ ------ ------- -------- --------- ---------- ----------- ------
ACCESSCTRL               *      *       *        *         *          *           N
BINDADD                  *      *       *        *         *          *           N
CONNECT                  *      *       *        *         *          *           N
CREATE_EXTERNAL_ROUTINE  *      *       *        *         *          *           N
CREATE_NOT_FENCED_ROUTINE *     *       *        *         *          *           Y
CREATETAB                *      *       *        *         *          *           N
DATAACCESS               *      *       *        *         *          *           N
DBADM                    *      *       *        *         *          *           N
```

```
EXPLAIN                 *       *       *       *       *       *       N
IMPLICIT_SCHEMA         *       *       *       *       *       *       N
LOAD                    *       *       *       *       *       *       Y
QUIESCE_CONNECT         *       *       *       *       *       *       N
SECADM                  *       *       *       *       *       *       N
SYSADM                  *       *       *       *       *       *       *
SQLADM                  *       *       *       *       *       *       N
SYSCTRL                 *       *       *       *       *       *       *
SYSMAINT                *       *       *       *       *       *       *
SYSMON                  *       *       *       *       *       *       *
WLMADM                  *       *       *       *       *       *       N
```

使用上の注意

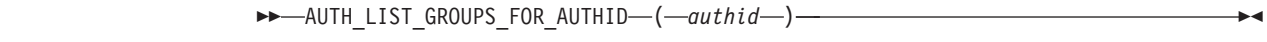
AUTH_LIST_AUTHORITIES_FOR_AUTHID 表関数の出力は *authidtype* に応じて異なります。例えば、USER の *authidtype* の場合、*authid* が以下の手段で保持しているすべての権限を戻します。

- *authid* に直接付与された
- *authid* が属するグループ (またはそのグループに付与されたロール) に付与された
- *authid* に付与されたロール (またはそのロールに付与されたロール) に付与された
- PUBLIC (または PUBLIC に付与されたロール) に付与された

AUTH_LIST_GROUPS_FOR_AUTHID 表関数 - 指定の許可 ID のグループ・メンバーシップ・リストの検索

AUTH_LIST_GROUPS_FOR_AUTHID 表関数は、指定の許可 ID がメンバーとして所属するグループのリストを戻します。

構文



スキーマは SYSPROC です。

表関数パラメーター

authid
照会対象の許可 ID を指定する、タイプ VARCHAR(128) の入力引数。許可 ID は、ユーザーのみを表すことができます。 *authid* が存在しない、NULL である、または空ストリングの場合、空の結果表が戻されます。

許可

AUTH_LIST_GROUPS_FOR_AUTHID 表関数に対する EXECUTE 特権。

例

AMY が属するすべてのグループを検索します。

```
SELECT * FROM TABLE (SYSPROC.AUTH_LIST_GROUPS_FOR_AUTHID('AMY')) AS T
```

以下はこの照会の出力例です。

```
GROUP
------------------------...-------------
BUILD
PDXDB2

  2 record(s) selected.
```

使用上の注意

以下の理由により、戻されるグループ情報は予想と異なる場合があります。

- Windows Active Directory 環境の場合、データベース・マネージャーについて次のことが言えます。
  - ローカル・グループ内での 1 レベルのグループ・ネストはサポートしているものの、ローカル・グループ内でのドメイン・ローカル・グループのネストは例外となります。たとえば、*authid* がグローバル・グループ G1 に属し、G1 がローカル・グループ L1 に属する場合、*authid* のグループとしてローカル・グループ L1 が戻されます。しかし、*authid* がドメイン・ローカル・グループ DL1 に属し、DL1 がローカル・グループ L1 に属する場合、*authid* のグループ情報は戻されません。
  - グローバル・グループのネストはサポートしていません。たとえば、*authid* がグローバル G2 に属し、G2 がグローバル G3 に属する場合、G2 のみが *authid* のグループとして戻されます。
- ユーザーの属するグループを列挙するために使用される Windows セキュリティー機能は、レジストリー変数 DB2_GRP_LOOKUP によって指定されます。
- 特定のドメインに属する許可 ID に関し、*authid* の一部としてドメインを指定せず、かつローカルとドメインの両方に同じ名前の *authid* が存在している場合には、ローカル許可 ID のグループが戻されます。

戻される情報

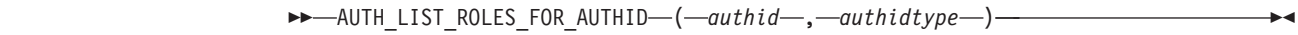
表 153. AUTH_LIST_GROUPS_FOR_AUTHID 表関数によって戻される情報

列名	データ・タイプ	説明
GROUP	VARCHAR(128)	許可 ID を保有するグループ

AUTH_LIST_ROLES_FOR_AUTHID 関数 - ロールのリストを戻す

AUTH_LIST_ROLES_FOR_AUTHID 関数は、指定の許可 ID がメンバーになっているロールのリストを戻します。

構文



スキーマは SYSPROC です。

表関数パラメーター

authid
照会対象の許可 ID を指定する、タイプ VARCHAR(128) の入力引数。許可 ID

はユーザー、グループ、またはロールです。*authid* が NULL または空ストリングである場合、空の結果表が戻されます。

authidtype
照会対象の許可 ID を指定する、タイプ VARCHAR(1) の入力引数。 *authidtype* が存在しないか、NULL または空ストリングである場合、空の結果表が戻されます。 *authidtype* に考えられる値は以下のとおりです。

- G: グループ
- R: ロール
- U: ユーザー

許可

AUTH_LIST_ROLES_FOR_AUTHID 関数に対する EXECUTE 特権。

戻される情報

表 154. AUTH_LIST_ROLES_FOR_AUTHID の結果セット

列名	データ・タイプ	説明
GRANTOR	VARCHAR(128)	ロールの認可者。
GRANTORTYPE	CHAR(1)	認可者のタイプ: • U = 認可者は個々のユーザー
GRANTEE	VARCHAR(128)	ユーザーがロールを付与しました。
GRANTEETYPE	CHAR(1)	被認可者のタイプ: • G = GRANTEE はグループ。 • R = GRANTEE はロール。 • U= 被認可者はユーザー。
ROLENAME	VARCHAR(128)	グループまたは別のロールを介して許可 ID に直接または間接に付与されたロールの名前。
CREATE_TIME	TIMESTAMP	ロールが作成された時刻。
ADMIN	CHAR(1)	ロールを付与したり、ロールを取り消したり、またはロールに関するコメントを付けたりする特権。 • N = 保有しない • Y= 保有する

例

ロール INTERN をロール DOCTOR に、さらにロール DOCTOR をロール SPECIALIST に付与してから、ロール SPECIALIST をユーザー ALICE に付与することについて考慮します。 ALICE はグループ HOSPITAL に属しており、ロール EMPLOYEE はグループ HOSPITAL に付与されます。また、ALICE は特殊グループ PUBLIC にも属しており、ロール PATIENTS が PUBLIC に付与されています。

例 1: ユーザー ALICE に付与されたすべてのロールを取得します。

```
SELECT GRANTOR, GRANTORTYPE, GRANTEE, GRANTEETYPE, ROLENAME,
  CREATE_TIME, ADMIN
FROM TABLE (SYSPROC.AUTH_LIST_ROLES_FOR_AUTHID ('ALICE', 'U') ) AS T
```

以下はこの照会の出力例です。

```
GRANTOR GRANTORTYPE GRANTEE    GRANTEETYPE ROLENAME   CREATE_TIME                ADMIN
------- ----------- -------    ----------- --------   -------------------------- -----
ZURBIE  U           DOCTOR     R           INTERN     2006-08-01-15.09.58.537399 N
ZURBIE  U           SPECIALIST R           DOCTOR     2006-08-01-15.10.04.540660 N
ZURBIE  U           ALICE      U           SPECIALIST 2006-08-01-15.10.08.776218 N
ZURBIE  U           HOSPITAL   G           EMPLOYEE   2006-08-01-15.10.14.277576 N
ZURBIE  U           PUBLIC     G           PATIENTS   2006-08-01-15.10.18.878609 N

  5 record(s) selected.
```

例 2: グループ HOSPITAL に付与されたすべてのロールを取得します。

```
SELECT GRANTOR, GRANTORTYPE, GRANTEE, GRANTEETYPE, ROLENAME,
  CREATE_TIME, ADMIN
FROM TABLE (SYSPROC.AUTH_LIST_ROLES_FOR_AUTHID ('HOSPITAL', 'G') ) AS T
```

以下はこの照会の出力例です。

```
GRANTOR GRANTORTYPE GRANTEE    GRANTEETYPE ROLENAME   CREATE_TIME                ADMIN
------- ----------- -------    ----------- --------   -------------------------- -----
ZURBIE  U           HOSPITAL   G           EMPLOYEE   2006-08-01-15.10.14.277576 N

  1  record(s) selected.
```

例 3: ロール SPECIALIST に付与されたすべてのロールを取得します。

```
SELECT GRANTOR, GRANTORTYPE, GRANTEE, GRANTEETYPE, ROLENAME,
  CREATE_TIME, ADMIN
FROM TABLE (SYSPROC.AUTH_LIST_ROLES_FOR_AUTHID ('SPECIALIST', 'R') ) AS T
```

以下はこの照会の出力例です。

```
GRANTOR GRANTORTYPE GRANTEE    GRANTEETYPE ROLENAME   CREATE_TIME                ADMIN
------- ----------- -------    ----------- --------   -------------------------- -----
ZURBIE  U           DOCTOR     R           INTERN     2006-08-01-15.09.58.537399 N
ZURBIE  U           SPECIALIST R           DOCTOR     2006-08-01-15.10.04.540660 N

  2  record(s) selected.
```

例 4: グループ PUBLIC に付与されたすべてのロールを取得します。

```
SELECT GRANTOR, GRANTORTYPE, GRANTEE, GRANTEETYPE, ROLENAME,
  CREATE_TIME, ADMIN
FROM TABLE (SYSPROC.AUTH_LIST_ROLES_FOR_AUTHID ('PUBLIC', 'G') ) AS T
```

以下はこの照会の出力例です。

```
GRANTOR GRANTORTYPE GRANTEE    GRANTEETYPE ROLENAME   CREATE_TIME                ADMIN
------- ----------- -------    ----------- --------   -------------------------- -----
ZURBIE  U           PUBLIC     G           PATIENTS   2006-08-01-15.10.18.878609 N

  1 record(s) selected.
```

使用上の注意

AUTH_LIST_ROLES_FOR_AUTHID 表関数の出力は AUTHIDTYPE に応じて異なります。

- ユーザーの場合、別のロールまたはそのユーザーが属する別のグループ (または PUBLIC) を介して直接または間接にユーザーに付与されたロールを戻します。
- グループの場合、別のロールを介して直接または間接にグループに付与されたロールを戻します。
- ロールの場合、別のロールを介して直接または間接にロールに付与されたロールを戻します。

AUTHORIZATIONIDS 管理ビュー - 許可 ID およびタイプの検索

AUTHORIZATIONIDS 管理ビューは、現在接続されているデータベースのシステム・カタログに定義されているすべての許可 ID に関して、特権または権限を認可された許可 ID のリストを、それらのタイプとともに戻します。特権または権限がグループまたはロールに付与されている場合は、グループ名またはロール名のみが戻されます。

スキーマは SYSIBMADM です。

許可

以下のいずれかの権限が必要です。

- AUTHORIZATIONIDS 管理ビューに対する SELECT 特権
- AUTHORIZATIONIDS 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

例

特権または権限を付与された許可 ID すべてを、それらのタイプとともに検索します。

```
SELECT * FROM SYSIBMADM.AUTHORIZATIONIDS
```

以下はこの照会の出力例です。

```
AUTHID                          AUTHIDTYPE
------------------------...------- ----------
PUBLIC                          G
JESSICAE                        U
DOCTOR                          R

  3 record(s) selected.
```

戻される情報

表 155. AUTHORIZATIONIDS 管理ビューによって戻される情報

列名	データ・タイプ	説明
AUTHID	VARCHAR(128)	特権または権限を明示的に付与された許可 ID。
AUTHIDTYPE	CHAR(1)	許可 ID タイプ。 • U: ユーザー • R: ロール • G: グループ

OBJECTOWNERS 管理ビュー - オブジェクト所有権情報の検索

OBJECTOWNERS 管理ビューは、現在接続中のデータベースのシステム・カタログに定義されているオブジェクトを所有するタイプ USER のすべての許可 ID のすべてのオブジェクト所有権情報を戻します。

スキーマは SYSIBMADM です。

許可

以下のいずれかの権限が必要です。

- OBJECTOWNERS 管理ビューに対する SELECT 特権
- OBJECTOWNERS 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

例

オブジェクト・スキーマ 'THERESAX' のすべてのオブジェクト所有権情報を検索します。

```
SELECT SUBSTR(OWNER,1,10) AS OWNER, OWNERTYPE,
   SUBSTR(OBJECTNAME,1,30) AS OBJECTNAME,
   SUBSTR(OBJECTSCHEMA,1,10) AS OBJECTSCHEMA, OBJECTTYPE
   FROM SYSIBMADM.OBJECTOWNERS WHERE OBJECTSCHEMA='THERESAX'
```

以下はこの照会の出力例です。

```
OWNER      OWNERTYPE OBJECTNAME           OBJECTSCHEMA OBJECTTYPE
---------- --------- -----------------... ------------ ----------------
THERESAX   U         MIN_SALARY           THERESAX     TRIGGER
THERESAX   U         POLICY_IR            SYSTOOLS     TRIGGER
THERESAX   U         CUSTOMER             THERESAX     XML SCHEMA
THERESAX   U         DB2DETAILDEADLOCK                 EVENTMONITORS
THERESAX   U         SAMPSEQUENCE         THERESAX     SEQUENCE
THERESAX   U         SQLE0F00             NULLID       PACKAGE
...
THERESAX   U         HI_OBJ_UNIQ          SYSTOOLS     TABLE CONSTRAINT

  257 record(s) selected.
```

戻される情報

表 156. OBJECTOWNERS 管理ビューによって戻される情報

列名	データ・タイプ	説明
OWNER	VARCHAR(128)	このオブジェクトを所有する許可 ID。
OWNERTYPE	VARCHAR(1)	許可 ID タイプ。 • U: ユーザー
OBJECTNAME	VARCHAR(128)	データベース・オブジェクト名。
OBJECTSCHEMA	VARCHAR(128)	データベース・オブジェクト・スキーマ。
OBJECTTYPE	VARCHAR(24)	データベース・オブジェクト・タイプ。

PRIVILEGES 管理ビュー - 特権情報の検索

PRIVILEGES 管理ビューは、現在接続中のデータベースのシステム・カタログに定義されているすべての許可 ID に関するすべての明示的な特権を戻します。

スキーマは SYSIBMADM です。

許可

以下のいずれかの権限が必要です。

- PRIVILEGES 管理ビューに対する SELECT 特権
- PRIVILEGES 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

例

すべての許可 ID に対して付与された特権を、オブジェクト名、スキーマ、およびタイプと共に検索します。

```
SELECT AUTHID, PRIVILEGE, OBJECTNAME, OBJECTSCHEMA, OBJECTTYPE
   FROM SYSIBMADM.PRIVILEGES
```

以下はこの照会の出力例です。

```
AUTHID       PRIVILEGE   OBJECTNAME              OBJECTSCHEMA     OBJECTTYPE
--------...- ----------- ------------------...- ------------...- ----------...-
JESSICAE     EXECUTE     SQLE0F00                NULLID           PACKAGE
PUBLIC       EXECUTE     SYSSH201                NULLID           PACKAGE
JESSICAE     EXECUTE     SYSSH202                NULLID           PACKAGE
PUBLIC       EXECUTE     SYSSH202                NULLID           PACKAGE
DOCTOR       EXECUTE     PKG0123                 NULLID           PACKAGE
...
PUBLIC       EXECUTE     SQL051109185227800      SYSPROC          FUNCTION
JESSICAE     EXECUTE     SQL051109185227801      SYSPROC          FUNCTION
PUBLIC       EXECUTE     SQL051109185227801      SYSPROC          FUNCTION
JESSICAE     EXECUTE     SQL051109185227838      SYSPROC          FUNCTION
PUBLIC       EXECUTE     SQL051109185227838      SYSPROC          FUNCTION
...
PUBLIC       EXECUTE     LIST_SRVR_TYPES         SYSPROC          PROCEDURE
PUBLIC       EXECUTE     LIST_SRVR_VERSIONS      SYSPROC          PROCEDURE
PUBLIC       EXECUTE     LIST_WRAP_OPTIONS       SYSPROC          PROCEDURE
PUBLIC       EXECUTE     LIST_SRVR_OPTIONS       SYSPROC          PROCEDURE
...
SYSTEM                   POLICY_UNQ              SYSTOOLS         INDEX
PUBLIC       CREATEIN                            NULLID           SCHEMA
PUBLIC       UPDATE      COLUMNS                 SYSSTAT          VIEW
PUBLIC       UPDATE      COLGROUPS               SYSSTAT          VIEW
...
```

戻される情報

表 157. PRIVILEGES 管理ビューによって戻される情報

列名	データ・タイプ	説明
AUTHID	VARCHAR(128)	この特権を明示的に付与された許可 ID。

表 157. PRIVILEGES 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明
AUTHIDTYPE	CHAR(1)	許可 ID タイプ。 • U: ユーザー • R: ロール • G: グループ
PRIVILEGE	VARCHAR(11)	この許可 ID に明示的に認可された特権。
GRANTABLE	VARCHAR(1)	特権が付与可能か否かを示します。 • Y: 付与可能 • N: 付与不能
OBJECTNAME	VARCHAR(128)	データベース・オブジェクト名。
OBJECTSCHEMA	VARCHAR(128)	データベース・オブジェクト・スキーマ。
OBJECTTYPE	VARCHAR(24)	データベース・オブジェクト・タイプ。

第 15 章 スナップショット・ルーチンおよびビュー

APPL_PERFORMANCE 管理ビュー - アプリケーションで選択される行のパーセンテージの検索

APPL_PERFORMANCE 管理ビューは、アプリケーションによって選択される行のパーセンテージに関する情報を表示します。戻されるのは現在接続されているデータベースのすべてのデータベース・パーティションに関する情報です。このビューは、大量の表スキャンを実行しているアプリケーションや、問題が発生する可能性のある照会を探すのに使用できます。

スキーマは SYSIBMADM です。

許可

以下のいずれかの権限が必要です。

- APPL_PERFORMANCE 管理ビューに対する SELECT 特権
- APPL_PERFORMANCE 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

アプリケーション・パフォーマンスのレポートを検索します。

```
SELECT SNAPSHOT_TIMESTAMP, SUBSTR(AUTHID,1,10) AS AUTHID,
   SUBSTR(APPL_NAME,1,10) AS APPL_NAME,AGENT_ID,
   PERCENT_ROWS_SELECTED, DBPARTITIONNUM
   FROM SYSIBMADM.APPL_PERFORMANCE
```

以下はこの照会の出力例です。

```
SNAPSHOT_TIMESTAMP         AUTHID      APPL_NAME ...
-------------------------- ----------- ---------- ...
2006-01-07-17.01.15.966668 JESSICAE    db2bp.exe  ...
2006-01-07-17.01.15.980278 JESSICAE    db2taskd   ...
2006-01-07-17.01.15.980278 JESSICAE    db2bp.exe  ...
                                                  ...
   3 record(s) selected.                          ...
```

この照会の出力 (続き)。

© Copyright IBM Corp. 2006, 2010

```
... AGENT_ID      PERCENT_ROWS_SELECTED DBPARTITIONNUM
... --------...-- --------------------- --------------
...           67                     -              1
...           68                     -              0
...           67                 57.14              0
...
```

戻される情報

表 158. APPL_PERFORMANCE 管理ビューによって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
AUTHID	VARCHAR(128)	auth_id - 許可 ID
APPL_NAME	VARCHAR(256)	appl_name - アプリケーション名
AGENT_ID	BIGINT	agent_id - アプリケーション・ハンドル (エージェント ID)
PERCENT_ROWS_SELECTED	DECIMAL(5,2)	ディスクから読み取られ、実際にアプリケーションに戻された行のパーセント。 **注:** 表示されるパーセンテージは、100.00 % より大きくなることはありません。
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

APPLICATIONS 管理ビュー - 接続されているデータベース・アプリケーション情報の検索

APPLICATIONS 管理ビューは、接続されているデータベース・アプリケーションに関する情報を戻します。ビューは LIST APPLICATIONS SHOW DETAIL CLP コマンドの SQL インターフェースです。ただし、現在接続されているデータベースに限ります。その情報は、SNAPAPPL_INFO 管理ビューに基づきます。

スキーマは SYSIBMADM です。

許可

以下のいずれかの権限が必要です。

- APPLICATIONS 管理ビューに対する SELECT 特権
- APPLICATIONS 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT

- SYSADM

例

例 1: 単一パーティション・データベース SAMPLE 内のすべてのアクティブ・アプリケーションに関する情報をリストします。

```
SELECT AGENT_ID, SUBSTR(APPL_NAME,1,10) AS APPL_NAME, AUTHID,
   APPL_STATUS FROM SYSIBMADM.APPLICATIONS WHERE DB_NAME = 'SAMPLE'
```

以下はこの照会の出力例です。

```
AGENT_ID             APPL_NAME  AUTHID    APPL_STATUS
-------------------- ---------- ---------- --------------------
                  23 db2bp.exe  JESSICAE   UOWEXEC

  1 record(s) selected.
```

例 2: 複数パーティション・データベース SAMPLE の、データベース・パーティション 0 上のアプリケーションごとのエージェントの数をリストします。

```
SELECT SUBSTR(APPL_NAME, 1, 10) AS APPL_NAME, COUNT(*) AS NUM
   FROM SYSIBMADM.APPLICATIONS WHERE DBPARTITIONNUM = 0
   AND DB_NAME = 'SAMPLE' GROUP BY APPL_NAME
```

以下はこの照会の出力例です。

```
APPL_NAME  NUM
---------- -----------
db2bp.exe            3
javaw.exe            1

  2 record(s) selected.
```

使用上の注意

ビューは GLOBAL 構文 (CLP で使用可能な構文) をサポートしていません。しかし、データベース・パーティションのデータはすべてビューから戻されるので、集約は SQL 集約関数を使うことによって行えます。

戻される情報

表 159. APPLICATIONS 管理ビューによって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
CLIENT_DB_ALIAS	VARCHAR(128)	client_db_alias - アプリケーションで使用するデータベース別名
DB_NAME	VARCHAR(128)	db_name - データベース名
AGENT_ID	BIGINT	agent_id - アプリケーション・ハンドル (エージェント ID)
APPL_NAME	VARCHAR(256)	appl_name - アプリケーション名
AUTHID	VARCHAR(128)	auth_id - 許可 ID
APPL_ID	VARCHAR(128)	appl_id - アプリケーション ID

表 159. APPLICATIONS 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
APPL_STATUS	VARCHAR(22)	appl_status - アプリケーション状況。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • BACKUP • COMMIT_ACT • COMP • CONNECTED • CONNECTPEND • CREATE_DB • DECOUPLED • DISCONNECTPEND • INTR • IOERROR_WAIT • LOAD • LOCKWAIT • QUIESCE_TABLESPACE • RECOMP • REMOTE_RQST • RESTART • RESTORE • ROLLBACK_ACT • ROLLBACK_TO_SAVEPOINT • TEND • THABRT • THCOMT • TPREP • UNLOAD • UOWEXEC • UOWWAIT • WAITFOR_REMOTE
STATUS_CHANGE_TIME	TIMESTAMP	status_change_time - アプリケーション状況変更時刻
SEQUENCE_NO	VARCHAR(4)	sequence_no - シーケンス番号
CLIENT_PRDID	VARCHAR(128)	client_prdid - クライアント製品/バージョン ID
CLIENT_PID	BIGINT	client_pid - クライアント・プロセス ID

表 159. APPLICATIONS 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
CLIENT_PLATFORM	VARCHAR(12)	client_platform - クライアント・オペレーティング・プラットフォーム。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • AIX • AIX64 • AS400_DRDA • DOS • DYNIX • HP • HP64 • HPIA • HPIA64 • LINUX • LINUX390 • LINUXIA64 • LINUXPPC • LINUXPPC64 • LINUXX8664 • LINUXZ64 • MAC • MVS_DRDA • NT • NT64 • OS2 • OS390 • SCO • SGI • SNI • SUN • SUN64 • 不明 (UNKNOWN) • UNKNOWN_DRDA • VM_DRDA • VSE_DRDA • WINDOWS • WINDOWS95

表 159. APPLICATIONS 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
CLIENT_PROTOCOL	VARCHAR(10)	client_protocol - クライアント通信プロトコル。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。 • CPIC • LOCAL • NETBIOS • NPIPE • TCPIP (DB2 UDB の場合) • TCPIP4 • TCPIP6
CLIENT_NNAME	VARCHAR(128)	client_nname モニター・エレメントは使用すべきではありません。返される値は無効な値です。
COORD_NODE_NUM	SMALLINT	coord_node - コーディネーター・ノード
COORD_AGENT_PID	BIGINT	coord_agent_pid - コーディネーター・エージェント
NUM_ASSOC_AGENTS	BIGINT	num_assoc_agents - 関連したエージェント数
TPMON_CLIENT_USERID	VARCHAR(256)	tpmon_client_userid - TP モニター・クライアント・ユーザー ID
TPMON_CLIENT_WKSTN	VARCHAR(256)	tpmon_client_wkstn - TP モニター・クライアント・ワークステーション名
TPMON_CLIENT_APP	VARCHAR(256)	tpmon_client_app - TP モニター・クライアント・アプリケーション名
TPMON_ACC_STR	VARCHAR(200)	tpmon_acc_str - TP モニター・クライアント・アカウンティング・ストリング
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

BP_HITRATIO 管理ビュー - バッファー・プールのヒット率に関する情報の検索

BP_HITRATIO 管理ビューは、バッファー・プールのヒット率を戻します。これには、現在接続されているデータベースのすべてのバッファー・プールおよびすべてのデータベース・パーティションに関する、全体のヒット率、データのヒット率、XDA のヒット率、および索引のヒット率が含まれます。

スキーマは SYSIBMADM です。

許可

以下のいずれかの権限が必要です。

- BP_HITRATIO 管理ビューに対する SELECT 特権
- BP_HITRATIO 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

接続されているデータベースのすべてのバッファー・プールに関するレポートを検索します。

```
SELECT SUBSTR(DB_NAME,1,8) AS DB_NAME, SUBSTR(BP_NAME,1,14) AS BP_NAME,
   TOTAL_HIT_RATIO_PERCENT, DATA_HIT_RATIO_PERCENT,
   INDEX_HIT_RATIO_PERCENT, XDA_HIT_RATIO_PERCENT, DBPARTITIONNUM
   FROM SYSIBMADM.BP_HITRATIO ORDER BY DBPARTITIONNUM
```

以下はこの照会の出力例です。

```
DB_NAME  BP_NAME        TOTAL_HIT_RATIO_PERCENT DATA_HIT_RATIO_PERCENT ...
-------- -------------- ----------------------- ---------------------- ...
TEST     IBMDEFAULTBP                    63.09                  68.94 ...
TEST     IBMSYSTEMBP4K                       -                      - ...
TEST     IBMSYSTEMBP8K                       -                      - ...
TEST     IBMSYSTEMBP16K                      -                      - ...
TEST     IBMSYSTEMBP32K                      -                      - ...
```

この照会の出力 (続き)。

```
... INDEX_HIT_RATIO_PERCENT XDA_HIT_RATIO_PERCENT DBPARTITIONNUM
... ----------------------- ---------------------- --------------
...                   43.20                      -              0
...                       -                      -              0
...                       -                      -              0
...                       -                      -              0
...                       -                      -              0
```

使用上の注意

バッファー・プールのヒット率は、読み取り合計に対する物理読み取りの率から算出されます。ヒット率が低いと、それだけキャッシュ・バッファー・プールよりもディスクから読み取られるデータの量が多くなり、その分だけ操作にコストがかかることになります。

戻される情報

表 160. BP_HITRATIO 管理ビューによって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	レポートが要求された時刻のタイム・スタンプ。
DB_NAME	VARCHAR(128)	db_name - データベース名
BP_NAME	VARCHAR(128)	bp_name - バッファー・プール名
TOTAL_LOGICAL_READS	BIGINT	バッファー・プール内の論理読み取り (索引、XDA、およびデータ) の合計。
TOTAL_PHYSICAL_READS	BIGINT	バッファー・プール内の物理読み取り (索引、XDA、およびデータ) の合計。
TOTAL_HIT_RATIO_PERCENT	DECIMAL(5,2)	ヒット率の合計 (索引、XDA、およびデータの読み取り)。
DATA_LOGICAL_READS	BIGINT	pool_data_l_reads - バッファー・プール・データの論理読み取り
DATA_PHYSICAL_READS	BIGINT	pool_data_p_reads - バッファー・プール・データの物理読み取り
DATA_HIT_RATIO_PERCENT	DECIMAL(5,2)	データのヒット率。
INDEX_LOGICAL_READS	BIGINT	pool_index_l_reads - バッファー・プール索引の論理読み取り
INDEX_PHYSICAL_READS	BIGINT	pool_index_p_reads - バッファー・プール索引の物理読み取り
INDEX_HIT_RATIO_PERCENT	DECIMAL(5,2)	索引のヒット率。
XDA_LOGICAL_READS	BIGINT	pool_xda_l_reads - バッファー・プール XDA データの論理読み取り
XDA_PHYSICAL_READS	BIGINT	pool_xda_p_reads - バッファー・プール XDA データの物理読み取り
XDA_HIT_RATIO_PERCENT	DECIMAL(5,2)	補助ストレージ・オブジェクトのヒット率。
DBPARTITIONNUM	SMALLINT	行のデータが検索されたデータベース・パーティション。

BP_READ_IO 管理ビュー - バッファー・プール読み取りパフォーマンス情報の検索

BP_READ_IO 管理ビューは、バッファー・プール読み取りパフォーマンス情報を戻します。このビューは、プリフェッチャーの効果を調べるために各バッファー・プールを表示するのに使用できます。

スキーマは SYSIBMADM です。

許可

以下のいずれかの権限が必要です。

- BP_READ_IO 管理ビューに対する SELECT 特権
- BP_READ_IO 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベースの全パーティション上のすべてのバッファー・プールに関する、物理読み取りの合計と平均読み取り時間を検索します。

```
SELECT SUBSTR(BP_NAME, 1, 15) AS BP_NAME, TOTAL_PHYSICAL_READS,
   AVERAGE_READ_TIME_MS, DBPARTITIONNUM
   FROM SYSIBMADM.BP_READ_IO ORDER BY DBPARTITIONNUM
```

以下はこの照会の出力例です。

```
BP_NAME         TOTAL_PHYSICAL_READS AVERAGE_READ_TIME_MS DBPARTITIONNUM
--------------- -------------------- -------------------- --------------
IBMDEFAULTBP                     811                    4              0
IBMSYSTEMBP4K                      0                    -              0
IBMSYSTEMBP8K                      0                    -              0
IBMSYSTEMBP16K                     0                    -              0
IBMDEFAULTBP                      34                    0              1
IBMSYSTEMBP4K                      0                    -              1
IBMSYSTEMBP8K                      0                    -              1
IBMDEFAULTBP                      34                    0              2
IBMSYSTEMBP4K                      0                    -              2
IBMSYSTEMBP8K                      0                    -              2

   10 record(s) selected.
```

戻される情報

表 161. BP_READ_IO 管理ビューによって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	レポートの生成日時。
BP_NAME	VARCHAR(128)	bp_name - バッファー・プール名
TOTAL_PHYSICAL_READS	BIGINT	物理読み取りの合計。
AVERAGE_READ_TIME_MS	BIGINT	平均読み取り時間 (ミリ秒)。
TOTAL_ASYNC_READS	BIGINT	非同期読み取りの合計。

表 161. BP_READ_IO 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
AVERAGE_ASYNC_READ_TIME_MS	BIGINT	平均非同期読み取り時間 (ミリ秒)。
TOTAL_SYNC_READS	BIGINT	同期読み取りの合計。
AVERAGE_SYNC_READ_TIME_MS	BIGINT	平均同期読み取り時間 (ミリ秒)。
PERCENT_SYNC_READS	DECIMAL(5,2)	プリフェッチなしで同期的に読み取られるページのパーセンテージ。多数のアプリケーションがプリフェッチなしで同期的にデータの読み取りを行うと、システムが最適に調整されない可能性があります。
ASYNC_NOT_READ_PERCENT	DECIMAL(5,2)	ディスクから非同期で読み取られ、照会でアクセスされなかったページのパーセンテージ。ディスクからバッファー・プールに非同期で読み取られ、照会でアクセスされないページが多すぎると、プリフェッチによってパフォーマンスが低下する可能性があります。
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

BP_WRITE_IO 管理ビュー - バッファー・プール書き込みパフォーマンス情報の検索

BP_WRITE_IO 管理ビューは、各バッファー・プールごとのバッファー・プール書き込みパフォーマンス情報を戻します。

スキーマは SYSIBMADM です。

許可

以下のいずれかの権限が必要です。

- BP_WRITE_IO 管理ビューに対する SELECT 特権
- BP_WRITE_IO 管理ビューに対する CONTROL 特権

- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベースの全データベース・パーティション上のすべてのバッファー・プールに関する、書き込みの合計と平均書き込み時間を検索します。

```
SELECT SUBSTR(BP_NAME, 1, 15) AS BP_NAME, TOTAL_WRITES,
   AVERAGE_WRITE_TIME_MS, DBPARTITIONNUM
   FROM SYSIBMADM.BP_WRITE_IO ORDER BY DBPARTITIONNUM
```

以下はこの照会の出力例です。

```
BP_NAME         TOTAL_WRITES     AVERAGE_WRITE_TIME_MS DBPARTITIONNUM
--------------- ------------...- --------------------- --------------
IBMDEFAULTBP                 11                     5              0
IBMSYSTEMBP4K                 0                     -              0
IBMSYSTEMBP8K                 0                     -              0
IBMSYSTEMBP16K                0                     -              0
IBMSYSTEMBP32K                0                     -              0
IBMDEFAULTBP                  0                     -              1
IBMSYSTEMBP4K                 0                     -              1
IBMSYSTEMBP8K                 0                     -              1
IBMDEFAULTBP                  0                     -              2
IBMSYSTEMBP4K                 0                     -              2
IBMSYSTEMBP8K                 0                     -              2

  11 record(s) selected.
```

戻される情報

表 162. BP_WRITE_IO 管理ビューによって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	レポートの生成日時。
BP_NAME	VARCHAR(128)	bp_name - バッファー・プール名
TOTAL_WRITES	BIGINT	書き込みの合計。
AVERAGE_WRITE_TIME_MS	BIGINT	平均書き込み時間 (ミリ秒)。
TOTAL_ASYNC_WRITES	BIGINT	非同期書き込みの合計。
PERCENT_WRITES_ASYNC	BIGINT	非同期書き込みのパーセント。
AVERAGE_ASYNC_WRITE_TIME_MS	BIGINT	平均非同期書き込み時間 (ミリ秒)。
TOTAL_SYNC_WRITES	BIGINT	同期書き込みの合計。

表 162. BP_WRITE_IO 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
AVERAGE_SYNC_WRITE_TIME_MS	BIGINT	平均同期書き込み時間 (ミリ秒)。
DBPARTITIONNUM	SMALLINT	行のデータが検索されたデータベース・パーティション。

CONTAINER_UTILIZATION 管理ビュー - 表スペース・コンテナーおよび使用率に関する情報の検索

CONTAINER_UTILIZATION 管理ビューは、表スペース・コンテナーと使用率に関する情報を戻します。これは、単一パーティション・データベースに対する LIST TABLESPACES コマンドと同様のレポートを取得します。その情報は、SNAPCONTAINER 管理ビューに基づきます。

スキーマは SYSIBMADM です。

許可

以下のいずれかの権限が必要です。

- CONTAINER_UTILIZATION 管理ビューに対する SELECT 特権
- CONTAINER_UTILIZATION 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

接続されている単一パーティション・データベース内のすべての表スペース・コンテナーのリスト (ページの合計、使用可能ページ、およびそのアクセス可能性の状況に関する情報を含む) を検索します。

```
SELECT SUBSTR(TBSP_NAME,1,20) AS TBSP_NAME, INT(TBSP_ID) AS TBSP_ID,
   SUBSTR(CONTAINER_NAME,1,45) AS CONTAINER_NAME, INT(CONTAINER_ID)
   AS CONTAINER_ID, CONTAINER_TYPE, INT(TOTAL_PAGES) AS TOTAL_PAGES,
   INT(USABLE_PAGES) AS USABLE_PAGES, ACCESSIBLE
   FROM SYSIBMADM.CONTAINER_UTILIZATION
```

以下はこの照会の出力例です。

```
TBSP_NAME            TBSP_ID     CONTAINER_NAME                              ...
---------------...- ----------- ------------------------------------...-- ...
SYSCATSPACE                    0 D:¥DB2¥NODE0000¥SQL00001¥SQLT0000.0        ...
TEMPSPACE1                     1 D:¥DB2¥NODE0000¥SQL00001¥SQLT0001.0        ...
```

```
USERSPACE1                     2 D:¥DB2¥NODE0000¥SQL00001¥SQLT0002.0        ...
SYSTOOLSPACE                   3 D:¥DB2¥NODE0000¥SQL00001¥SYSTOOLSPACE      ...
SYSTOOLSTMPSPACE               4 D:¥DB2¥NODE0000¥SQL00001¥SYSTOOLSTMPSPACE  ...

  5 record(s) selected.
```

この照会の出力 (続き)。

```
... CONTAINER_ID CONTAINER_TYPE TOTAL_PAGES USABLE_PAGES ACCESSIBLE
... ------------ -------------- ----------- ------------ ----------
...            0 PATH                    0            0          1
...            0 PATH                    0            0          1
...            0 PATH                    0            0          1
...            0 PATH                    0            0          1
...            0 PATH                    0            0          1
```

戻される情報

ファイル・システムの情報を戻すためには、データベース・マネージャー構成で BUFFERPOOL スナップショット・モニター・スイッチを有効にする必要があります。

表 163. CONTAINER_UTILIZATION 管理ビューによって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
TBSP_NAME	VARCHAR(128)	tablespace_name - 表スペース名
TBSP_ID	BIGINT	tablespace_id - 表スペース ID
CONTAINER_NAME	VARCHAR(256)	container_name - コンテナー名
CONTAINER_ID	BIGINT	container_id - コンテナー ID
CONTAINER_TYPE	VARCHAR(16)	container_type - コンテナー・タイプ これは、sqlutil.h での定義を基にしたテキスト ID です。以下のいずれかとなります。 • DISK_EXTENT_TAG • DISK_PAGE_TAG • FILE_EXTENT_TAG • FILE_PAGE_TAG • PATH
TOTAL_PAGES	BIGINT	container_total_pages - コンテナー内の合計ページ数
USABLE_PAGES	BIGINT	container_usable_pages - コンテナー内の使用可能なページ数
ACCESSIBLE	SMALLINT	container_accessible - コンテナーのアクセス可能性
STRIPE_SET	BIGINT	container_stripe_set - ストライプ・セット
FS_ID	VARCHAR(22)	fs_id - 固有のファイル・システム識別番号

表 163. CONTAINER_UTILIZATION 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
FS_TOTAL_SIZE_KB	BIGINT	fs_total_size - ファイル・システムの合計サイズ。このインターフェースは値を KB 単位で戻します。
FS_USED_SIZE_KB	BIGINT	fs_used_size - ファイル・システム上で使用されるスペースの量。このインターフェースは値を KB 単位で戻します。
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

LOCKS_HELD 管理ビュー - 保持されているロックに関する情報の検索

注: この管理ビューは非推奨になり、以下のものに置き換えられました。すなわち、 460 ページの『MON_GET_APPL_LOCKWAIT - アプリケーションが待機しているロックについての情報の収集』、 490 ページの『MON_GET_LOCKS - 現在接続されているデータベース内のすべてのロックのリスト』、および 425 ページの『MON_FORMAT_LOCK_NAME - 内部ロック名のフォーマット設定と詳細の出力』。

LOCKS_HELD 管理ビューは、現在保持されているロックに関する情報を戻します。

スキーマは SYSIBMADM です。

許可

以下のいずれかの権限が必要です。

- LOCKS_HELD 管理ビューに対する SELECT 特権
- LOCKS_HELD 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

例 1: データベース SAMPLE の各表で保持されているロックの総数をリストします。

```
SELECT TABSCHEMA, TABNAME, COUNT(*) AS NUMBER_OF_LOCKS_HELD
   FROM SYSIBMADM.LOCKS_HELD WHERE DB_NAME = 'SAMPLE'
   GROUP BY DBPARTITIONNUM, TABSCHEMA, TABNAME
```

以下はこの照会の出力例です。

```
TABSCHEMA       TABNAME         NUMBER_OF_LOCKS_HELD
----------...- ---------...- --------------------
JESSICAE        EMPLOYEE                           5
JESSICAE        EMP_RESUME                         1
JESSICAE        ORG                                3
```

例 2: 現在接続されているデータベース SAMPLE 内の、エスカレートされていないすべてのロックをリストします。

```
SELECT AGENT_ID, TABSCHEMA, TABNAME, LOCK_OBJECT_TYPE, LOCK_MODE,
   LOCK_STATUS FROM SYSIBMADM.LOCKS_HELD WHERE LOCK_ESCALATION = 0
   AND DBPARTITIONNUM = 0
```

以下はこの照会の出力例です。

```
AGENT_ID     TABSCHEMA     TABNAME      LOCK_OBJECT_TYPE   LOCK_MODE  LOCK_STATUS
--------...- ---------...- --------...- ------------------ ---------- -----------
         680 JESSICAE      EMPLOYEE     INTERNALV_LOCK     S          GRNT
         680 JESSICAE      EMPLOYEE     INTERNALP_LOCK     S          GRNT
```

例 3: エージェント ID 310 を持つアプリケーションで現在保持されているロックに関するロック情報をリストします。

```
SELECT TABSCHEMA, TABNAME, LOCK_OBJECT_TYPE, LOCK_MODE, LOCK_STATUS,
   LOCK_ESCALATION FROM SYSIBMADM.LOCKS_HELD WHERE AGENT_ID = 310
```

以下はこの照会の出力例です。

```
TABSCHEMA     TABNAME      LOCK_OBJECT_TYPE   LOCK_MODE  LOCK_STATUS
---------...- ---------...- ------------------ ---------- -----------
JESSICAE      EMP_RESUME   TABLE_LOCK         S          GRNT
JESSICAE      EMPLOYEE     ROW_LOCK           S          GRNT
```

戻される情報

表 164. LOCKS_HELD 管理ビューによって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	レポートの生成日時。
DB_NAME	VARCHAR(128)	db_name - データベース名
AGENT_ID	BIGINT	agent_id - アプリケーション・ハンドル (エージェント ID)
APPL_NAME	VARCHAR(256)	appl_name - アプリケーション名
AUTHID	VARCHAR(128)	auth_id - 許可 ID
TBSP_NAME	VARCHAR(128)	tablespace_name - 表スペース名
TABSCHEMA	VARCHAR(128)	table_schema - 表スキーマ名
TABNAME	VARCHAR(128)	table_name - 表名
TAB_FILE_ID	BIGINT	table_file_id - 表ファイル ID

表 164. LOCKS_HELD 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
LOCK_OBJECT_TYPE	VARCHAR(18)	lock_object_type - 待機中のロック対象タイプ。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • AUTORESIZE_LOCK • AUTOSTORAGE_LOCK • BLOCK_LOCK • EOT_LOCK • INPLACE_REORG_LOCK • INTERNAL_LOCK • INTERNALB_LOCK • INTERNALC_LOCK • INTERNALJ_LOCK • INTERNALL_LOCK • INTERNALO_LOCK • INTERNALQ_LOCK • INTERNALP_LOCK • INTERNALS_LOCK • INTERNALT_LOCK • INTERNALV_LOCK • KEYVALUE_LOCK • ROW_LOCK • SYSBOOT_LOCK • TABLE_LOCK • TABLE_PART_LOCK • TABLESPACE_LOCK • XML_PATH_LOCK
LOCK_NAME	VARCHAR(32)	lock_name - ロック名

表 164. LOCKS_HELD 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
LOCK_MODE	VARCHAR(10)	lock_mode - ロック・モード。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • IN • IS • IX • NON (ロックなしの場合) • NS • NW • S • SIX • U • X • Z
LOCK_STATUS	VARCHAR(10)	lock_status - ロック状況。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • CONV • GRNT
LOCK_ESCALATION	SMALLINT	lock_escalation - ロック・エスカレーション
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

LOCKWAITS 管理ビュー - 現行のロック待機情報の検索

注: この管理ビューは非推奨になり、以下のものに置き換えられました。すなわち、556 ページの『MON_LOCKWAITS 管理ビュー - ロックの取得を待機しているアプリケーションに関するメトリックの取得』。

LOCKWAITS 管理ビューは、ロック取得待機中のアプリケーションのために作業を代行している DB2 エージェントに関する情報を戻します。

スキーマは SYSIBMADM です。

許可

以下のいずれかの権限が必要です。

- LOCKWAITS 管理ビューに対する SELECT 特権
- LOCKWAITS 管理ビューに対する CONTROL 特権

- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

例 1: エージェント ID 89 を持つアプリケーションのすべてのロック待機に関する情報をリストします。

```
SELECT SUBSTR(TABSCHEMA,1,8) AS TABSCHEMA, SUBSTR(TABNAME,1,15) AS TABNAME,
   LOCK_OBJECT_TYPE, LOCK_MODE, LOCK_MODE_REQUESTED, AGENT_ID_HOLDING_LK
   FROM SYSIBMADM.LOCKWAITS WHERE AGENT_ID = 89
```

以下はこの照会の出力例です。

```
TABSCHEMA TABNAME     LOCK_OBJECT_TYPE LOCK_MODE  ...
--------- -------...- ---------------- ---------- ...
JESSICAE  T1          ROW_LOCK         X          ...

  1 record(s) selected.
```

この照会の出力 (続き)。

```
... LOCK_MODE_REQUESTED AGENT_ID_HOLDING_LK
... ------------------- --------------------
... NS                                    7
```

例 2: データベース SAMPLE における、表ごとの未解決ロック要求の総数をリストします。要求の数で出力をソートすると、競合が最も多い表を識別できます。

```
SELECT SUBSTR(TABSCHEMA,1,8) AS TABSCHEMA, SUBSTR(TABNAME, 1, 15)
   AS TABNAME, COUNT(*) AS NUM_OF_LOCK_REQUESTS_WAITING,
   DBPARTITIONNUM
   FROM SYSIBMADM.LOCKWAITS WHERE DB_NAME = 'SAMPLE'
   GROUP BY TABSCHEMA, TABNAME, DBPARTITIONNUM
   ORDER BY NUM_OF_LOCK_REQUESTS_WAITING DESC
```

以下はこの照会の出力例です。

```
TABSCHEMA TABNAME     NUM_OF_LOCK_REQUESTS_WAITING DBPARTITIONNUM
--------- -------...- ---------------------------- --------------
JESSICAE  T3                                     2              0
JESSICAE  T1                                     1              0
JESSICAE  T2                                     1              0

  3 record(s) selected.
```

戻される情報

表 165. LOCKWAITS 管理ビューによって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	レポートの生成日時。
DB_NAME	VARCHAR(128)	db_name - データベース名

表 165. LOCKWAITS 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
AGENT_ID	BIGINT	agent_id - アプリケーション・ハンドル (エージェント ID)
APPL_NAME	VARCHAR(256)	appl_name - アプリケーション名
AUTHID	VARCHAR(128)	auth_id - 許可 ID
TBSP_NAME	VARCHAR(128)	tablespace_name - 表スペース名
TABSCHEMA	VARCHAR(128)	table_schema - 表スキーマ名
TABNAME	VARCHAR(128)	table_name - 表名
SUBSECTION_NUMBER	BIGINT	ss_number - サブセクション番号
LOCK_OBJECT_TYPE	VARCHAR(18)	lock_object_type - 待機中のロック対象タイプ。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • AUTORESIZE_LOCK • AUTOSTORAGE_LOCK • BLOCK_LOCK • EOT_LOCK • INPLACE_REORG_LOCK • INTERNAL_LOCK • INTERNALB_LOCK • INTERNALC_LOCK • INTERNALJ_LOCK • INTERNALL_LOCK • INTERNALO_LOCK • INTERNALQ_LOCK • INTERNALP_LOCK • INTERNALS_LOCK • INTERNALT_LOCK • INTERNALV_LOCK • KEYVALUE_LOCK • ROW_LOCK • SYSBOOT_LOCK • TABLE_LOCK • TABLE_PART_LOCK • TABLESPACE_LOCK • XML_PATH_LOCK
LOCK_WAIT_START_TIME	TIMESTAMP	lock_wait_start_time - ロック待機開始タイム・スタンプ
LOCK_NAME	VARCHAR(32)	lock_name - ロック名

表 165. LOCKWAITS 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
LOCK_MODE	VARCHAR(10)	lock_mode - ロック・モード。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • IN • IS • IX • NON (ロックなしの場合) • NS • NW • S • SIX • U • X • Z
LOCK_MODE_REQUESTED	VARCHAR(10)	lock_mode_requested - 要求されているロック・モード。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • IN • IS • IX • NON (ロックなしの場合) • NS • NW • S • SIX • U • X • Z
AGENT_ID_HOLDING_LK	BIGINT	agent_id_holding_lock - ロックを保持しているエージェント ID
APPL_ID_HOLDING_LK	VARCHAR(128)	appl_id_holding_lk - ロックを保持しているアプリケーション ID
LOCK_ESCALATION	SMALLINT	lock_escalation - ロック・エスカレーション
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

LOG_UTILIZATION 管理ビュー - ログ使用率に関する情報の検索

LOG_UTILIZATION 管理ビューは、現在接続されているデータベースのログ使用率に関する情報を戻します。各データベース・パーティションごとに 1 つの行が戻されます。

スキーマは SYSIBMADM です。

許可

以下のいずれかの権限が必要です。

- LOG_UTILIZATION 管理ビューに対する SELECT 特権
- LOG_UTILIZATION 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベース SAMPLE のログ使用率をリストします。

```
SELECT * FROM SYSIBMADM.LOG_UTILIZATION
```

以下はこの照会の出力例です。

```
DB_NAME  ... LOG_UTILIZATION_PERCENT TOTAL_LOG_USED_KB    ...
-------- ... ----------------------- -------------------- ...
SAMPLE   ...                    9.75                 1989 ...
                                                          ...
  1 record(s) selected.                                   ...
```

この照会の出力 (続き)。

```
... TOTAL_LOG_AVAILABLE_KB TOTAL_LOG_USED_TOP_KB DBPARTITIONNUM
... ---------------------- --------------------- --------------
...                  18411                  1990              0
...
...
```

使用上の注意

無限ロギングが構成されているデータベースの場合、LOG_UTILIZATION_PERCENT および TOTAL_LOG_AVAILABLE_KB は NULL になります。

戻される情報

表 166. LOG_UTILIZATION 管理ビューによって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
DB_NAME	VARCHAR(128)	db_name - データベース名
LOG_UTILIZATION_PERCENT	DECIMAL(5,2)	合計ログ・スペースの使用率パーセント。
TOTAL_LOG_USED_KB	BIGINT	total_log_used - 使用されているログ・スペースの合計。このインターフェースは値を KB 単位で戻します。
TOTAL_LOG_AVAILABLE_KB	BIGINT	total_log_available - 使用可能なログ合計。このインターフェースは値を KB 単位で戻します。
TOTAL_LOG_USED_TOP_KB	BIGINT	tot_log_used_top - 使用された最大合計ログ・スペース。このインターフェースは値を KB 単位で戻します。
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

LONG_RUNNING_SQL 管理ビュー

LONG_RUNNING_SQL 管理ビューは、現在接続されているデータベースで実行時間が最も長い SQL ステートメントを戻します。

スキーマは SYSIBMADM です。

許可

以下のいずれかの権限が必要です。

- LONG_RUNNING_SQL 管理ビューに対する SELECT 特権
- LONG_RUNNING_SQL 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベースで実行時間が最も長い SQL ステートメントに関するレポートを検索します。

```
SELECT SUBSTR(STMT_TEXT, 1, 50) AS STMT_TEXT, AGENT_ID,
   ELAPSED_TIME_MIN, APPL_STATUS, DBPARTITIONNUM
   FROM SYSIBMADM.LONG_RUNNING_SQL ORDER BY DBPARTITIONNUM
```

以下はこの照会の出力例です。

```
STMT_TEXT                          AGENT_ID     ...
-----------------------------...- --------...- ...
select * from dbuser.employee               228 ...
select * from dbuser.employee               228 ...
select * from dbuser.employee               228 ...
                                            ...
3 record(s) selected.                       ...
```

この照会の出力 (続き)。

```
... ELAPSED_TIME_MIN APPL_STATUS     DBPARTITIONNUM
... ---------------- -----------...- --------------
...                2 UOWWAIT                      0
...                0 CONNECTED                    1
...                0 CONNECTED                    2
```

使用上の注意

このビューを使用することにより、データベース内の、実行時間の長い SQL ステートメントを識別することができます。現在実行中の照会を見て、実行時間が最も長いステートメント、および照会の現行状況を調べることができます。エージェント ID をユニーク ID として使用することにより、その SQL ステートメントを含むアプリケーションをさらに詳しく調査することができます。長時間実行していてロック待機している場合は、LOCKWAITS または LOCKS_HELD 管理ビューを使用してさらに深く掘り下げることもできます。「waiting on User (ユーザー待機中)」という表示は、DB2 サーバーが何の動作も行っておらず、アプリケーションによる何らかの動作 (例えば次のフェッチの発行または次の SQL ステートメントのサブミット) を待機していることを意味します。

戻される情報

表 167. LONG_RUNNING_SQL 管理ビューによって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	レポートの生成時刻。
ELAPSED_TIME_MIN	INTEGER	ステートメントの経過時間 (分)。
AGENT_ID	BIGINT	agent_id - アプリケーション・ハンドル (エージェント ID)
APPL_NAME	VARCHAR(256)	appl_name - アプリケーション名

表 167. LONG_RUNNING_SQL 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
APPL_STATUS	VARCHAR(22)	appl_status - アプリケーション状況。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • BACKUP • COMMIT_ACT • COMP • CONNECTED • CONNECTPEND • CREATE_DB • DECOUPLED • DISCONNECTPEND • INTR • IOERROR_WAIT • LOAD • LOCKWAIT • QUIESCE_TABLESPACE • RECOMP • REMOTE_RQST • RESTART • RESTORE • ROLLBACK_ACT • ROLLBACK_TO_SAVEPOINT • TEND • THABRT • THCOMT • TPREP • UNLOAD • UOWEXEC • UOWWAIT • WAITFOR_REMOTE
AUTHID	VARCHAR(128)	auth_id - 許可 ID
INBOUND_COMM_ADDRESS	VARCHAR(32)	inbound_comm_address - インバウンド通信アドレス
STMT_TEXT	CLOB(16 M)	stmt_text - SQL 動的ステートメント・テキスト
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

QUERY_PREP_COST 管理ビュー - ステートメント準備時間に関する情報の検索

QUERY_PREP_COST 管理ビューは、ステートメントのリストおよびステートメントの準備に必要な時間に関する情報を戻します。

スキーマは SYSIBMADM です。

許可

以下のいずれかの権限が必要です。

- QUERY_PREP_COST 管理ビューに対する SELECT 特権
- SNAPAGENT 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAPDYN_SQL 表関数に対する EXECUTE 特権
- DATAACCESS 権限

例

準備に最も高い割合の時間が費やされた照会に関するレポートを検索します。

```
SELECT NUM_EXECUTIONS, AVERAGE_EXECUTION_TIME_S, PREP_TIME_PERCENT,
   SUBSTR(STMT_TEXT, 1, 30) AS STMT_TEXT, DBPARTITIONNUM
   FROM SYSIBMADM.QUERY_PREP_COST ORDER BY PREP_TIME_PERCENT
```

以下はこの照会の出力例です。

```
NUM_EXECUTIONS     AVERAGE_EXECUTION_TIME_S ...
--------------...- ------------------------ ...
                1                       25 ...

  1 record(s) selected.
```

この照会の出力 (続き)。

```
... PREP_TIME_PERCENT STMT_TEXT                      DBPARTITIONNUM
... ----------------- ------------------------------ --------------
...               0.0 select * from dbuser.employee               0
```

使用上の注意

ビューから選択するときに ORDER BY 節を使用することにより、最も準備コストの高い照会を識別することができます。このビューで、照会の実行頻度および各照会の平均実行時間を調べることができます。照会をコンパイルして最適化する時間と照会の実行にかかる時間にほとんど差がない場合には、使用している最適化クラスを調べたほうが良いかもしれません。最適化クラスを下げることで照会での完全な最適化がより短時間で行われるようになるため、結果がより早く戻される可能性があります。しかし、照会での準備に多大な時間がかかり、しかもそれが (再準備されることなく) 何度も実行されるとなると、最適化クラスの問題ではない可能性があります。

戻される情報

表 168. QUERY_PREP_COST 管理ビューによって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	レポートの生成日時。
NUM_EXECUTIONS	BIGINT	num_executions - ステートメント実行回数
AVERAGE_EXECUTION_TIME_S	BIGINT	平均実行時間 (秒単位)。
PREP_TIME_MS	BIGINT	prep_time_worst - ステートメント最長準備時間 (ミリ秒単位).
PREP_TIME_PERCENT	DECIMAL(5,2)	準備に費やされる実行時間のパーセント。
STMT_TEXT	CLOB(2 M)	stmt_text - SQL 動的ステートメント・テキスト
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

SNAPAGENT 管理ビューおよび SNAP_GET_AGENT 表関数 - agent 論理データ・グループのアプリケーション・スナップショット情報の検索

SNAPAGENT 管理ビューおよび SNAP_GET_AGENT 表関数は、アプリケーション・スナップショットから、特に agent 論理データ・グループのエージェント情報を戻します。

SNAPAGENT 管理ビュー

この管理ビューを使用して、現在接続中のデータベースに関する agent 論理データ・グループのアプリケーション・スナップショット情報を取得することができます。

SNAPAGENT 管理ビューを SNAPAGENT_MEMORY_POOL、SNAPAPPL、SNAPAPPL_INFO、 SNAPSTMT、および SNAPSUBSECTION 管理ビューとともに使用すると、GET SNAPSHOT FOR APPLICATIONS ON database-alias CLP コマンドに相当する情報が提供されます。ただし、すべてのデータベース・パーティションからデータを取得します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 639 ページの表 169を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPAGENT 管理ビューに対する SELECT 特権
- SNAPAGENT 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_AGENT 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続中のデータベースに関するすべてのアプリケーション・スナップショット情報を、agent 論理データ・グループから取得します。

```
SELECT * FROM SYSIBMADM.SNAPAGENT
```

以下はこの照会の出力例です。

```
SNAPSHOT_TIMESTAMP         DB_NAME     AGENT_ID        ...
-------------------------- -------...- ---------...--- ...
2005-07-19-11.03.26.740423 SAMPLE                 101 ...
2005-07-19-11.03.26.740423 SAMPLE                  49 ...
                                                      ...
  2 record(s) selected.                               ...
```

この照会からの出力 (続き)。

```
...  AGENT_PID            LOCK_TIMEOUT_VAL     DBPARTITIONNUM
...  -------------------- -------------------- --------------
...                 11980                   -1              0
...                 15940                   -1              0
...
...
```

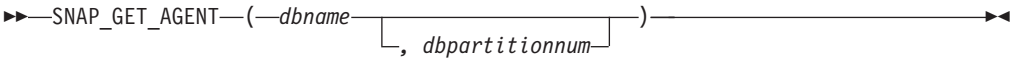
SNAP_GET_AGENT 表関数

SNAP_GET_AGENT 表関数は、SNAPAGENT 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションに関する情報を取得することができます。

SNAP_GET_AGENT 表関数を SNAP_GET_AGENT_MEMORY_POOL、SNAP_GET_APPL_V95、 SNAP_GET_APPL_INFO_V95、 SNAP_GET_STMT、および SNAP_GET_SUBSECTION 表関数とともに使用すると、GET SNAPSHOT FOR ALL APPLICATIONS CLP コマンドに相当する情報が提供されます。ただし、すべてのデータベース・パーティションからデータを取得します。

戻される可能性のある情報の完全なリストは、 639 ページの表 169を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、空ストリングを指定します。現在接続されているデータベースと同じインスタンス内のすべてのデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプを持つファイルが存在しない場合、SNAP_GET_AGENT 表関数は、現在接続中のデータベースおよびデータベース・パーティション番号のスナップショットを取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_AGENT 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT

- SYSADM

例

すべてのアクティブ・データベース内のすべてのアプリケーションに関する、すべてのアプリケーション・スナップショット情報を取得します。

```
SELECT * FROM TABLE(SNAP_GET_AGENT(CAST(NULL AS VARCHAR(128)), -1)) AS T
```

以下はこの照会の出力例です。

```
SNAPSHOT_TIMESTAMP         DB_NAME     AGENT_ID      ...
-------------------------- -------...- --------...-- ...
2006-01-03-17.21.38.530785 SAMPLE                48 ...
2006-01-03-17.21.38.530785 SAMPLE                47 ...
2006-01-03-17.21.38.530785 SAMPLE                46 ...
2006-01-03-17.21.38.530785 TESTDB                30 ...
2006-01-03-17.21.38.530785 TESTDB                29 ...
2006-01-03-17.21.38.530785 TESTDB                28 ...

6 record(s) selected.
```

この照会からの出力 (続き)。

```
... AGENT_PID        LOCK_TIMEOUT_VAL     DBPARTITIONNUM
... ---------...---- -------------------- --------------
...             7696                   -1              0
...             8536                   -1              0
...             6672                   -1              0
...             2332                   -1              0
...             8360                   -1              0
...             6736                   -1              0
...
```

戻される情報

表 169. SNAPAGENT 管理ビューおよび SNAP_GET_AGENT 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
DB_NAME	VARCHAR(128)	db_name - データベース名
AGENT_ID	BIGINT	agent_id - アプリケーション・ハンドル (エージェント ID)
AGENT_PID	BIGINT	agent_pid - エンジン・ディスパッチ可能単位 (EDU)
LOCK_TIMEOUT_VAL	BIGINT	lock_timeout_val - ロック・タイムアウト (秒)
DBPARTITIONNUM	SMALLINT	行のデータが検索されたデータベース・パーティション。

SNAPAGENT_MEMORY_POOL 管理ビューおよび SNAP_GET_AGENT_MEMORY_POOL 表関数 - memory_pool 論理データ・グループのスナップショット情報の検索

SNAPAGENT_MEMORY_POOL 管理ビューおよび SNAP_GET_AGENT_MEMORY_POOL 表関数は、エージェント・レベルでのメモリー使用量についての情報を戻します。

SNAPAGENT_MEMORY_POOL 管理ビュー

この管理ビューを使用して、現在接続中のデータベースのエージェント・レベルでのメモリー使用量に関する memory_pool 論理データ・グループのスナップショット情報を取得することができます。

SNAPAGENT_MEMORY_POOL 管理ビューを SNAPAGENT、SNAPAPPL、SNAPAPPL_INFO、SNAPSTMT、および SNAPSUBSECTION 管理ビューとともに使用すると、GET SNAPSHOT FOR APPLICATIONS ON database-alias CLP コマンドに相当する情報が提供されます。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 643 ページの表 170を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPAGENT_MEMORY_POOL 管理ビューに対する SELECT 特権
- SNAPAGENT_MEMORY_POOL 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_AGENT_MEMORY_POOL 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

メモリー・プールおよびそれらの現在のサイズのリストを取得します。

```
SELECT AGENT_ID, POOL_ID, POOL_CUR_SIZE FROM SYSIBMADM.SNAPAGENT_MEMORY_POOL
```

以下はこの照会の出力例です。

```
AGENT_ID       POOL_ID POOL_  CUR_SIZE
--------...--- -------------- ---------...------
            48 APPLICATION                65536
            48 OTHER                      65536
            48 APPL_CONTROL               65536
            47 APPLICATION                65536
            47 OTHER                     131072
            47 APPL_CONTROL               65536
            46 OTHER                     327680
            46 APPLICATION               262144
            46 APPL_CONTROL               65536

9 record(s) selected.
```

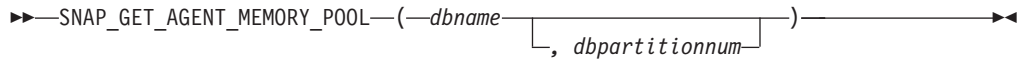
SNAP_GET_AGENT_MEMORY_POOL 表関数

SNAP_GET_AGENT_MEMORY_POOL 表関数は、SNAPAGENT_MEMORY_POOL 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションに関する情報を取得することができます。

SNAP_GET_AGENT_MEMORY_POOL 表関数を SNAP_GET_AGENT、SNAP_GET_APPL_V95、 SNAP_GET_APPL_INFO_V95、 SNAP_GET_STMT、および SNAP_GET_SUBSECTION 表関数とともに使用すると、GET SNAPSHOT FOR ALL APPLICATIONS CLP コマンドに相当する情報が提供されます。

戻される可能性のある情報の完全なリストは、 643 ページの表 170を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、空ストリングを指定します。現在接続されているデータベースと同じインスタンス内のすべてのデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプション

が使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプを持つファイルが存在しない場合、SNAP_GET_AGENT_MEMORY_POOL 表関数は、現在接続中のデータベースおよびデータベース・パーティション番号のスナップショットを取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_AGENT_MEMORY_POOL 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

すべてのデータベースのメモリー・プールおよびそれらの現在のサイズのリストを取得します。

```
SELECT SUBSTR(DB_NAME,1,8) AS DB_NAME, AGENT_ID, POOL_ID, POOL_CUR_SIZE
   FROM TABLE(SNAP_GET_AGENT_MEMORY_POOL(CAST (NULL AS VARCHAR(128)), -1))
   AS T
```

以下はこの照会の出力例です。

```
DB_NAME  AGENT_ID       POOL_ID        POOL_CUR_SIZE
-------- --------...--- -------------- --------------------
SAMPLE              48 APPLICATION                   65536
SAMPLE              48 OTHER                         65536
SAMPLE              48 APPL_CONTROL                  65536
SAMPLE              47 APPLICATION                   65536
SAMPLE              47 OTHER                        131072
SAMPLE              47 APPL_CONTROL                  65536
SAMPLE              46 OTHER                        327680
SAMPLE              46 APPLICATION                  262144
SAMPLE              46 APPL_CONTROL                  65536
TESTDB              30 APPLICATION                   65536
TESTDB              30 OTHER                         65536
TESTDB              30 APPL_CONTROL                  65536
TESTDB              29 APPLICATION                   65536
TESTDB              29 OTHER                        131072
TESTDB              29 APPL_CONTROL                  65536
TESTDB              28 OTHER                        327680
```

```
TESTDB               28 APPLICATION                 65536
TESTDB               28 APPL_CONTROL                65536

18 record(s) selected.
```

戻される情報

表 170. SNAPAGENT_MEMORY_POOL 管理ビューおよび SNAP_GET_AGENT_MEMORY_POOL 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
DB_NAME	VARCHAR(128)	db_name - データベース名
AGENT_ID	BIGINT	agent_id - アプリケーション・ハンドル (エージェント ID)
AGENT_PID	BIGINT	agent_pid - エンジン・ディスパッチ可能単位 (EDU)
POOL_ID	VARCHAR(14)	pool_id - メモリー・プール ID。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • APP_GROUP • APPL_CONTROL • APPLICATION • BP • CAT_CACHE • DATABASE • DFM • FCMBP • IMPORT_POOL • LOCK_MGR • MONITOR • OTHER • PACKAGE_CACHE • QUERY • SHARED_SORT • SORT • STATEMENT • STATISTICS • UTILITY
POOL_CUR_SIZE	BIGINT	pool_cur_size - メモリー・プールの現行サイズ
POOL_WATERMARK	BIGINT	pool_watermark - メモリー・プール水準点

表 170. SNAPAGENT_MEMORY_POOL 管理ビューおよび SNAP_GET_AGENT_MEMORY_POOL 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
POOL_CONFIG_SIZE	BIGINT	pool_config_size - メモリー・プールの構成済みサイズ
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

SNAPAPPL_INFO 管理ビューおよび SNAP_GET_APPL_INFO_V95 表関数 - appl_info 論理データ・グループのスナップショット情報の検索

SNAPAPPL_INFO 管理ビューおよび SNAP_GET_APPL_INFO_V95 表関数は、アプリケーション・スナップショットから、特に appl_info 論理データ・グループのアプリケーション情報を戻します。

SNAPAPPL_INFO 管理ビュー

この管理ビューを使用して、現在接続中のデータベースに関する appl_info 論理データ・グループのスナップショット情報を取得することができます。

SNAPAPPL_INFO 管理ビューを SNAPAGENT、 SNAPAGENT_MEMORY_POOL、 SNAPAPPL、 SNAPSTMT、および SNAPSUBSECTION 管理ビューとともに使用すると、GET SNAPSHOT FOR APPLICATIONS ON database-alias CLP コマンドに相当する情報が提供されます。ただし、すべてのデータベース・パーティションからデータを取得します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 647 ページの表 171を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPAPPL_INFO 管理ビューに対する SELECT 特権
- SNAPAPPL_INFO 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの権限が必要です。

- SNAP_GET_APPL_INFO_V95 表関数に対する EXECUTE 特権
- DATAACCESS 権限

また、以下のいずれかの権限が必要です。

- SYSMON
- SYSMAINT
- SYSCTRL

- SYSADM

例

現在のデータベースに接続中のアプリケーションの状況を取得します。

```
SELECT AGENT_ID, SUBSTR(APPL_NAME,1,10) AS APPL_NAME, APPL_STATUS
   FROM SYSIBMADM.SNAPAPPL_INFO
```

以下はこの照会の出力例です。

```
AGENT_ID             APPL_NAME  APPL_STATUS
-------------------- ---------- ----------------------
                 101 db2bp.exe  UOWEXEC
                  49 db2bp.exe  CONNECTED

  2 record(s) selected.
```

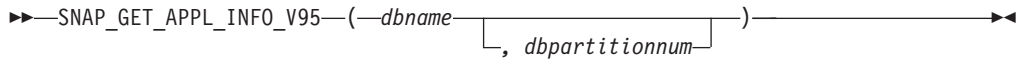
SNAP_GET_APPL_INFO_V95 表関数

SNAP_GET_APPL_INFO_V95 表関数は、SNAPAPPL_INFO 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションに関する情報を取得することができます。

SNAP_GET_APPL_INFO_V95 表関数を SNAP_GET_AGENT、SNAP_GET_AGENT_MEMORY_POOL、 SNAP_GET_APPL_V95、SNAP_GET_STMT、および SNAP_GET_SUBSECTION 表関数とともに使用すると、GET SNAPSHOT FOR ALL APPLICATIONS CLP コマンドに相当する情報が提供されます。ただし、すべてのデータベース・パーティションからデータを取得します。

戻される可能性のある情報の完全なリストは、 647 ページの表 171を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、空ストリングを指定します。現在接続されているデータベースと同じインスタンス内のすべてのデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプを持つファイルが存在しない場合、SNAP_GET_APPL_INFO_V95 表関数は、現在接続中のデータベースおよびデータベース・パーティション番号のスナップショットを取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_APPL_INFO_V95 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

接続中のデータベース・パーティション上のすべてのアプリケーションの状況を取得します。

```
SELECT SUBSTR(DB_NAME,1,8) AS DB_NAME, AGENT_ID,
   SUBSTR(APPL_NAME,1,10) AS APPL_NAME, APPL_STATUS
   FROM TABLE(SNAP_GET_APPL_INFO_V95(CAST(NULL AS VARCHAR(128)),-1)) AS T
```

以下はこの照会の出力例です。

```
DB_NAME  AGENT_ID             APPL_NAME  APPL_STATUS
-------- -------------------- ---------- ----------------------
TOOLSDB                   14 db2bp.exe  CONNECTED
SAMPLE                    15 db2bp.exe  UOWEXEC
SAMPLE                     8 javaw.exe  CONNECTED
SAMPLE                     7 db2bp.exe  UOWWAIT

  4 record(s) selected.
```

以下は、表関数の結果からの SELECT の実行時に入手できる内容について示しています。

```
SELECT SUBSTR(DB_NAME,1,8) AS DB_NAME, AUTHORITY_LVL
   FROM TABLE(SNAP_GET_APPL_INFO_V95(CAST(NULL AS VARCHAR(128)),-1)) AS T
```

以下はこの照会の出力例です。

```
DB_NAME  AUTHORITY_LVL
-------- -----------------------------------------------------....
TESTDB   SYSADM(GROUP) + DBADM(USER) + CREATETAB(USER, GROUP) +
            BINDADD(USER, GROUP) + CONNECT(USER, GROUP) +
            CREATE_NOT_FENC(USER) + IMPLICIT_SCHEMA(USER, GROUP) +
            LOAD(USER) + CREATE_EXT_RT(USER) + QUIESCE_CONN(USER)
TESTDB   SYSADM(GROUP) + DBADM(USER) + CREATETAB(USER, GROUP) +
            BINDADD(USER, GROUP) + CONNECT(USER, GROUP) +
            CREATE_NOT_FENC(USER) + IMPLICIT_SCHEMA(USER, GROUP) +
            LOAD(USER) + CREATE_EXT_RT(USER) + QUIESCE_CONN(USER)
TESTDB   SYSADM(GROUP) + DBADM(USER) + CREATETAB(USER, GROUP) +
            BINDADD(USER, GROUP) + CONNECT(USER, GROUP) +
            CREATE_NOT_FENC(USER) + IMPLICIT_SCHEMA(USER, GROUP) +
            LOAD(USER) + CREATE_EXT_RT(USER) + QUIESCE_CONN(USER)

3 record(s) selected.
```

戻される情報

表 171. SNAPAPPL_INFO 管理ビューおよび SNAP_GET_APPL_INFO_V95 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
AGENT_ID	BIGINT	agent_id - アプリケーション・ハンドル (エージェント ID)

表 171. SNAPAPPL_INFO 管理ビューおよび SNAP_GET_APPL_INFO_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
APPL_STATUS	VARCHAR(22)	appl_status - アプリケーション状況。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • BACKUP • COMMIT_ACT • COMP • CONNECTED • CONNECTPEND • CREATE_DB • DECOUPLED • DISCONNECTPEND • INTR • IOERROR_WAIT • LOAD • LOCKWAIT • QUIESCE_TABLESPACE • RECOMP • REMOTE_RQST • RESTART • RESTORE • ROLLBACK_ACT • ROLLBACK_TO_SAVEPOINT • TEND • THABRT • THCOMT • TPREP • UNLOAD • UOWEXEC • UOWWAIT • WAITFOR_REMOTE
CODEPAGE_ID	BIGINT	codepage_id - アプリケーションで使用するコード・ページ ID
NUM_ASSOC_AGENTS	BIGINT	num_assoc_agents - 関連したエージェント数
COORD_NODE_NUM	SMALLINT	coord_node - コーディネーター・ノード

表 171. SNAPAPPL_INFO 管理ビューおよび SNAP_GET_APPL_INFO_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
AUTHORITY_LVL	VARCHAR(512)	authority_bitmap - ユーザー許可レベル。 このインターフェースは、sql.h で定義されたデータベース権限およびそれらのソースに基づくテキスト ID を戻し、その形式は次のとおりです。 authority(source, ...) + authority(source, ...) + ... 権限のソースは複数でも構いません。USER、GROUP、または USER と GROUP のいずれかです。 "authority" に使用できる値 • ACCESSCTRL • BINDADD • CONNECT • CREATE_EXT_RT • CREATE_NOT_FENC • CREATETAB • DATAACCESS • DBADM • EXPLAIN • IMPLICIT_SCHEMA • LOAD • LIBADM • QUIESCE_CONN • SECADM • SQLADM • SYSADM • SYSCTRL • SYSMAINT • SYSMON • SYSQUIESCE • WLMADM "source" に使用できる値 • USER - ユーザーに付与された権限、またはそのユーザーに付与されているロールに付与された権限。 • GROUP - ユーザーが属するグループに付与された権限、またはユーザーが属するグループに付与されるロールに付与された権限。
CLIENT_PID	BIGINT	client_pid - クライアント・プロセス ID
COORD_AGENT_PID	BIGINT	coord_agent_pid - コーディネーター・エージェント

表 171. SNAPAPPL_INFO 管理ビューおよび SNAP_GET_APPL_INFO_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
STATUS_CHANGE_TIME	TIMESTAMP	status_change_time - アプリケーション状況変更時刻
CLIENT_PLATFORM	VARCHAR(12)	client_platform - クライアント・オペレーティング・プラットフォーム。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。 • AIX • AIX64 • AS400_DRDA • DOS • DYNIX • HP • HP64 • HPIA • HPIA64 • LINUX • LINUX390 • LINUXIA64 • LINUXPPC • LINUXPPC64 • LINUXX8664 • LINUXZ64 • MAC • MVS_DRDA • NT • NT64 • OS2 • OS390 • SCO • SGI • SNI • SUN • SUN64 • UNKNOWN • UNKNOWN_DRDA • VM_DRDA • VSE_DRDA • WINDOWS

表 171. SNAPAPPL_INFO 管理ビューおよび SNAP_GET_APPL_INFO_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
CLIENT_PROTOCOL	VARCHAR(10)	client_protocol - クライアント通信プロトコル。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。 • CPIC • LOCAL • NETBIOS • NPIPE • TCPIP (DB2 UDB の場合) • TCPIP4 • TCPIP6
TERRITORY_CODE	SMALLINT	territory_code - データベース・テリトリー・コード
APPL_NAME	VARCHAR(256)	appl_name - アプリケーション名
APPL_ID	VARCHAR(128)	appl_id - アプリケーション ID
SEQUENCE_NO	VARCHAR(4)	sequence_no - シーケンス番号
PRIMARY_AUTH_ID	VARCHAR(128)	auth_id - 許可 ID
SESSION_AUTH_ID	VARCHAR(128)	session_auth_id - セッション許可 ID
CLIENT_NNAME	VARCHAR(128)	client_nname モニター・エレメントは使用すべきではありません。返される値は無効な値です。
CLIENT_PRDID	VARCHAR(128)	client_prdid - クライアント製品/バージョン ID
INPUT_DB_ALIAS	VARCHAR(128)	input_db_alias - 入力データベース別名
CLIENT_DB_ALIAS	VARCHAR(128)	client_db_alias - アプリケーションで使用するデータベース別名
DB_NAME	VARCHAR(128)	db_name - データベース名
DB_PATH	VARCHAR(1024)	db_path - データベース・パス
EXECUTION_ID	VARCHAR(128)	execution_id - ユーザー・ログイン ID
CORR_TOKEN	VARCHAR(128)	corr_token - DRDA 相関トークン
TPMON_CLIENT_USERID	VARCHAR(256)	tpmon_client_userid - TP モニター・クライアント・ユーザー ID
TPMON_CLIENT_WKSTN	VARCHAR(256)	tpmon_client_wkstn - TP モニター・クライアント・ワークステーション名
TPMON_CLIENT_APP	VARCHAR(256)	tpmon_client_app - TP モニター・クライアント・アプリケーション名
TPMON_ACC_STR	VARCHAR(200)	tpmon_acc_str - TP モニター・クライアント・アカウンティング・ストリング
DBPARTITIONNUM	SMALLINT	行のデータが検索されたデータベース・パーティション。
WORKLOAD_ID	INTEGER	現行のワークロード ID.

表 171. SNAPAPPL_INFO 管理ビューおよび SNAP_GET_APPL_INFO_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
IS_SYSTEM_APPL	SMALLINT	IS_SYSTEM_APPL の値は、アプリケーションが DB2 内部システム・アプリケーションかどうかを示します。 0 はユーザー・アプリケーションであることを示します。 1 はシステム・アプリケーションであることを示します。 DB2 システム・アプリケーションの例は DB2 イベント・モニターです。 一般に、DB2 システム・アプリケーションの名前は "db2" で始まります。例えば、db2stmm, db2taskd などです。

SNAPAPPL 管理ビューおよび SNAP_GET_APPL_V95 表関数 - appl 論理データ・グループのスナップショット情報の検索

『SNAPAPPL 管理ビュー』および 653 ページの『SNAP_GET_APPL_V95 表関数』は、アプリケーション・スナップショットからアプリケーションに関する情報 (特に appl 論理データ・グループ) を戻します。

SNAPAPPL 管理ビュー

この管理ビューを使用して、現在接続中のデータベースに関する appl 論理データ・グループのスナップショット情報を取得することができます。

SNAPAPPL 管理ビューを SNAPAGENT、 SNAPAGENT_MEMORY_POOL、 SNAPAPPL_INFO、 SNAPSTMT、および SNAPSUBSECTION 管理ビューとともに使用すると、GET SNAPSHOT FOR APPLICATIONS ON database-alias CLP コマンドに相当する情報が提供されます。ただし、すべてのデータベース・パーティションからデータを取得します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 655 ページの表 172を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPAPPL 管理ビューに対する SELECT 特権
- SNAPAPPL 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_APPL_V95 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

接続中のデータベース内の各アプリケーションについて読み取りおよび書き込みが行われた行の詳細を取得します。

```
SELECT SUBSTR(DB_NAME,1,8) AS DB_NAME, AGENT_ID, ROWS_READ, ROWS_WRITTEN
   FROM SYSIBMADM.SNAPAPPL
```

以下はこの照会の出力例です。

```
DB_NAME  AGENT_ID             ROWS_READ            ROWS_WRITTEN
-------- -------------------- -------------------- --------------------
SAMPLE                     7                   25                    0

  1 record(s) selected.
```

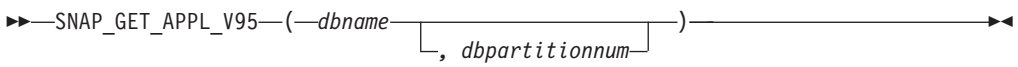
SNAP_GET_APPL_V95 表関数

SNAP_GET_APPL_V95 表関数は、SNAPAPPL 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションに関する情報を取得することができます。

SNAP_GET_APPL_V95 表関数を SNAP_GET_AGENT、SNAP_GET_AGENT_MEMORY_POOL、 SNAP_GET_APPL_INFO_V95、SNAP_GET_STMT、および SNAP_GET_SUBSECTION 表関数とともに使用すると、GET SNAPSHOT FOR ALL APPLICATIONS CLP コマンドに相当する情報が提供されます。ただし、すべてのデータベース・パーティションからデータを取得します。

戻される可能性のある情報の完全なリストは、 655 ページの表 172を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、空ストリングを指定します。現在接続されているデータベースと同じインスタンス内のすべてのデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプを持つファイルが存在しない場合、SNAP_GET_APPL_V95 表関数は、現在接続中のデータベースおよびデータベース・パーティション番号のスナップショットを取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_APPL_V95 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

すべてのアクティブ・データベースの各アプリケーションについて読み取りおよび書き込みが行われた行の詳細を取得します。

```
SELECT SUBSTR(DB_NAME,1,8) AS DB_NAME, AGENT_ID, ROWS_READ, ROWS_WRITTEN
   FROM TABLE (SNAP_GET_APPL_V95(CAST(NULL AS VARCHAR(128)),-1)) AS T
```

以下はこの照会の出力例です。

```
DB_NAME  AGENT_ID       ROWS_READ      ROWS_WRITTEN
-------- --------...--- ---------...-- ------------...-
WSDB                679              0                 0
WSDB                461              3                 0
WSDB                460              4                 0
TEST                680              4                 0
TEST                455              6                 0
TEST                454              0                 0
TEST                453             50                 0
```

戻される情報

表 172. SNAPAPPL 管理ビューおよび SNAP_GET_APPL_V95 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
DB_NAME	VARCHAR(128)	db_name - データベース名
AGENT_ID	BIGINT	agent_id - アプリケーション・ハンドル (エージェント ID)
UOW_LOG_SPACE_USED	BIGINT	uow_log_space_used - 作業単位ログ・スペース
ROWS_READ	BIGINT	rows_read - 読み取り行数
ROWS_WRITTEN	BIGINT	rows_written - 書き込み行数
INACT_STMTHIST_SZ	BIGINT	stmt_history_list_size - ステートメント履歴リストのサイズ
POOL_DATA_L_READS	BIGINT	pool_data_l_reads - バッファー・プール・データの論理読み取り
POOL_DATA_P_READS	BIGINT	pool_data_p_reads - バッファー・プール・データの物理読み取り
POOL_DATA_WRITES	BIGINT	pool_data_writes - バッファー・プールへのデータの書き込み
POOL_INDEX_L_READS	BIGINT	pool_index_l_reads - バッファー・プール索引の論理読み取り
POOL_INDEX_P_READS	BIGINT	pool_index_p_reads - バッファー・プール索引の物理読み取り
POOL_INDEX_WRITES	BIGINT	pool_index_writes - バッファー・プール索引の書き込み
POOL_TEMP_DATA_L_READS	BIGINT	pool_temp_data_l_reads - バッファー・プール一時データの論理読み取り
POOL_TEMP_DATA_P_READS	BIGINT	pool_temp_data_p_reads - バッファー・プール一時データの物理読み取り
POOL_TEMP_INDEX_L_READS	BIGINT	pool_temp_index_l_reads - バッファー・プール一時索引の論理読み取り
POOL_TEMP_INDEX_P_READS	BIGINT	pool_temp_index_p_reads - バッファー・プール一時索引の物理読み取り

表 172. SNAPAPPL 管理ビューおよび SNAP_GET_APPL_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
POOL_TEMP_XDA_L_READS	BIGINT	pool_temp_xda_l_reads - バッファー・プール一時 XDA データの論理読み取り
POOL_TEMP_XDA_P_READS	BIGINT	pool_temp_xda_p_reads - バッファー・プール一時 XDA データの物理読み取り : モニター・エレメント
POOL_XDA_L_READS	BIGINT	pool_xda_l_reads - バッファー・プール XDA データの論理読み取り
POOL_XDA_P_READS	BIGINT	pool_xda_p_reads - バッファー・プール XDA データの物理読み取り
POOL_XDA_WRITES	BIGINT	pool_xda_writes - バッファー・プール XDA データの書き込み
POOL_READ_TIME	BIGINT	pool_read_time - バッファー・プール物理読み取り時間の合計
POOL_WRITE_TIME	BIGINT	pool_write_time - バッファー・プール物理書き込み時間の合計
DIRECT_READS	BIGINT	direct_reads - データベースからの直接読み取り
DIRECT_WRITES	BIGINT	direct_writes - データベースへの直接書き込み
DIRECT_READ_REQS	BIGINT	direct_read_reqs - 直接読み取り要求
DIRECT_WRITE_REQS	BIGINT	direct_write_reqs - 直接書き込み要求
DIRECT_READ_TIME	BIGINT	direct_read_time - 直接読み取り時間
DIRECT_WRITE_TIME	BIGINT	direct_write_time - 直接書き込み時間
UNREAD_PREFETCH_PAGES	BIGINT	unread_prefetch_pages - 読み取り不能プリフェッチ・ページ
LOCKS_HELD	BIGINT	locks_held - ロック保持数
LOCK_WAITS	BIGINT	lock_waits - ロック待機数
LOCK_WAIT_TIME	BIGINT	lock_wait_time - ロック待機中の時間
LOCK_ESCALS	BIGINT	lock_escals - ロック・エスカレーション数
X_LOCK_ESCALS	BIGINT	x_lock_escals - 排他ロック・エスカレーション数
DEADLOCKS	BIGINT	deadlocks - デッドロック検出数
TOTAL_SORTS	BIGINT	total_sorts - ソート合計
TOTAL_SORT_TIME	BIGINT	total_sort_time - ソート時間合計

表 172. SNAPAPPL 管理ビューおよび SNAP_GET_APPL_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
SORT_OVERFLOWS	BIGINT	sort_overflows - ソート・オーバーフロー
COMMIT_SQL_STMTS	BIGINT	commit_sql_stmts - 試行されたコミット・ステートメント
ROLLBACK_SQL_STMTS	BIGINT	rollback_sql_stmts - 試行されたロールバック・ステートメント
DYNAMIC_SQL_STMTS	BIGINT	dynamic_sql_stmts - 試行された動的 SQL ステートメント
STATIC_SQL_STMTS	BIGINT	static_sql_stmts - 試行された静的 SQL ステートメント
FAILED_SQL_STMTS	BIGINT	failed_sql_stmts - 失敗したステートメント操作
SELECT_SQL_STMTS	BIGINT	select_sql_stmts - 実行された選択 SQL ステートメント
DDL_SQL_STMTS	BIGINT	ddl_sql_stmts - データ定義言語 (DDL) SQL ステートメント
UID_SQL_STMTS	BIGINT	uid_sql_stmts - 実行された更新/挿入/削除 SQL ステートメント
INT_AUTO_REBINDS	BIGINT	int_auto_rebinds - 内部自動再バインド
INT_ROWS_DELETED	BIGINT	int_rows_deleted - 削除された内部行数
INT_ROWS_UPDATED	BIGINT	int_rows_updated - 更新された内部行数
INT_COMMITS	BIGINT	int_commits - 内部コミット数
INT_ROLLBACKS	BIGINT	int_rollbacks - 内部ロールバック数
INT_DEADLOCK_ROLLBACKS	BIGINT	int_deadlock_rollbacks - デッドロックによる内部ロールバック数
ROWS_DELETED	BIGINT	rows_deleted - 削除行数
ROWS_INSERTED	BIGINT	rows_inserted - 挿入行数
ROWS_UPDATED	BIGINT	rows_updated - 更新行数
ROWS_SELECTED	BIGINT	rows_selected - 選択行数
BINDS_PRECOMPILES	BIGINT	binds_precompiles - 試行されたバインド/プリコンパイル
OPEN_REM_CURS	BIGINT	open_rem_curs - 開かれているリモート・カーソル
OPEN_REM_CURS_BLK	BIGINT	open_rem_curs_blk - 開かれているリモート・ブロック・カーソル
REJ_CURS_BLK	BIGINT	rej_curs_blk - リジェクトされたブロック・カーソル要求
ACC_CURS_BLK	BIGINT	acc_curs_blk - 受け入れられたブロック・カーソル要求

表 172. SNAPAPPL 管理ビューおよび SNAP_GET_APPL_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
SQL_REQS_SINCE_COMMIT	BIGINT	sql_reqs_since_commit - 最終コミット後の SQL 要求数
LOCK_TIMEOUTS	BIGINT	lock_timeouts - ロック・タイムアウト数
INT_ROWS_INSERTED	BIGINT	int_rows_inserted - 挿入された内部行数
OPEN_LOC_CURS	BIGINT	open_loc_curs - 開かれているローカル・カーソル
OPEN_LOC_CURS_BLK	BIGINT	open_loc_curs_blk - 開かれているローカル・ブロック・カーソル
PKG_CACHE_LOOKUPS	BIGINT	pkg_cache_lookups - パッケージ・キャッシュ参照
PKG_CACHE_INSERTS	BIGINT	pkg_cache_inserts - パッケージ・キャッシュ挿入
CAT_CACHE_LOOKUPS	BIGINT	cat_cache_lookups - カタログ・キャッシュ参照数
CAT_CACHE_INSERTS	BIGINT	cat_cache_inserts - カタログ・キャッシュ挿入数
CAT_CACHE_OVERFLOWS	BIGINT	cat_cache_overflows - カタログ・キャッシュ・オーバーフロー数
NUM_AGENTS	BIGINT	num_agents - ステートメントで作動しているエージェントの数
AGENTS_STOLEN	BIGINT	agents_stolen - スチールされたエージェント
ASSOCIATED_AGENTS_TOP	BIGINT	associated_agents_top - 関連エージェント最大数
APPL_PRIORITY	BIGINT	appl_priority - アプリケーション・エージェント優先順位
APPL_PRIORITY_TYPE	VARCHAR(16)	appl_priority_type - アプリケーション優先順位タイプ。このインターフェースは、`sqlmon.h` 内の定義に基づいてテキスト ID を戻します。それは、次のうちの 1 つです。 • DYNAMIC_PRIORITY • FIXED_PRIORITY
PREFETCH_WAIT_TIME	BIGINT	prefetch_wait_time - プリフェッチ待ち時間
APPL_SECTION_LOOKUPS	BIGINT	appl_section_lookups - セクションの参照回数
APPL_SECTION_INSERTS	BIGINT	appl_section_inserts - セクション挿入数

表 172. SNAPAPPL 管理ビューおよび SNAP_GET_APPL_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
LOCKS_WAITING	BIGINT	locks_waiting - ロックで待機中の現行エージェント
TOTAL_HASH_JOINS	BIGINT	total_hash_joins - ハッシュ結合の合計
TOTAL_HASH_LOOPS	BIGINT	total_hash_loops - ハッシュ・ループの合計
HASH_JOIN_OVERFLOWS	BIGINT	hash_join_overflows - ハッシュ結合のオーバーフロー
HASH_JOIN_SMALL_ OVERFLOWS	BIGINT	hash_join_small_overflows - ハッシュ結合の短精度オーバーフロー
APPL_IDLE_TIME	BIGINT	appl_idle_time - アプリケーション・アイドル時間
UOW_LOCK_WAIT_TIME	BIGINT	uow_lock_wait_time - ロック待機中の作業単位の合計時間
UOW_COMP_STATUS	VARCHAR(14)	uow_comp_status - 作業単位完了状況。このインターフェースは、sqlmon.h 内の定義に基づいてテキスト ID を戻します。それは、次のうちの 1 つです。 • APPL_END • UOWABEND • UOWCOMMIT • UOWDEADLOCK • UOWLOCKTIMEOUT • UOWROLLBACK • UOWUNKNOWN
AGENT_USR_CPU_TIME_S	BIGINT	agent_usr_cpu_time - エージェントが使用したユーザー CPU 時間 (秒単位)*
AGENT_USR_CPU_TIME_MS	BIGINT	agent_usr_cpu_time - エージェントが使用したユーザー CPU 時間 (小数部、マイクロ秒単位)*
AGENT_SYS_CPU_TIME_S	BIGINT	agent_sys_cpu_time - エージェントが使用したシステム CPU 時間 (秒単位)*
AGENT_SYS_CPU_TIME_MS	BIGINT	agent_sys_cpu_time - エージェントが使用したシステム CPU 時間 (小数部、マイクロ秒単位)*
APPL_CON_TIME	TIMESTAMP	appl_con_time - 接続要求開始タイム・スタンプ

表 172. SNAPAPPL 管理ビューおよび SNAP_GET_APPL_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
CONN_COMPLETE_TIME	TIMESTAMP	conn_complete_time - 接続要求完了タイム・スタンプ
LAST_RESET	TIMESTAMP	last_reset - 最後のリセット・タイム・スタンプ
UOW_START_TIME	TIMESTAMP	uow_start_time - 作業単位開始タイム・スタンプ
UOW_STOP_TIME	TIMESTAMP	uow_stop_time - 作業単位停止タイム・スタンプ
PREV_UOW_STOP_TIME	TIMESTAMP	prev_uow_stop_time - 直前の作業単位完了タイム・スタンプ
UOW_ELAPSED_TIME_S	BIGINT	uow_elapsed_time - 最新の作業単位の経過時間 (秒単位)*
UOW_ELAPSED_TIME_MS	BIGINT	uow_elapsed_time - 最新の作業単位の経過時間 (小数部、マイクロ秒単位)*
ELAPSED_EXEC_TIME_S	BIGINT	elapsed_exec_time - ステートメント実行経過時間 (秒単位)*
ELAPSED_EXEC_TIME_MS	BIGINT	elapsed_exec_time - ステートメント実行経過時間 (小数部、マイクロ秒単位)*
INBOUND_COMM_ADDRESS	VARCHAR(32)	inbound_comm_address - インバウンド通信アドレス
LOCK_TIMEOUT_VAL	BIGINT	lock_timeout_val - ロック・タイムアウト (秒)
PRIV_WORKSPACE_NUM_OVERFLOWS	BIGINT	priv_workspace_num_overflows - 専用ワークスペースのオーバーフロー回数
PRIV_WORKSPACE_SECTION_INSERTS	BIGINT	priv_workspace_section_inserts - 専用ワークスペース・セクション挿入
PRIV_WORKSPACE_SECTION_LOOKUPS	BIGINT	priv_workspace_section_lookups - 専用ワークスペース・セクションの参照
PRIV_WORKSPACE_SIZE_TOP	BIGINT	priv_workspace_size_top - 専用ワークスペースの最大サイズ
SHR_WORKSPACE_NUM_OVERFLOWS	BIGINT	shr_workspace_num_overflows - 共有ワークスペースのオーバーフロー回数
SHR_WORKSPACE_SECTION_INSERTS	BIGINT	shr_workspace_section_inserts - 共有ワークスペース・セクション挿入数

表 172. SNAPAPPL 管理ビューおよび SNAP_GET_APPL_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
SHR_WORKSPACE_SECTION_LOOKUPS	BIGINT	shr_workspace_section_lookups - 共有ワークスペース・セクションの参照回数
SHR_WORKSPACE_SIZE_TOP	BIGINT	shr_workspace_size_top - 最大共有ワークスペース・サイズ
DBPARTITIONNUM	SMALLINT	行のデータが検索されたデータベース・パーティション。
CAT_CACHE_SIZE_TOP	BIGINT	cat_cache_size_top - カタログ・キャッシュ最高水準点
TOTAL_OLAP_FUNCS	BIGINT	実行される OLAP 関数の合計数。
OLAP_FUNC_OVERFLOWS	BIGINT	OLAP 関数データが使用可能なソート・ヒープ・スペースを超えた回数。
* この列の元になるモニター・エレメントの合計消費時間を計算するには、このモニター・エレメントに関する _S で終わっている列で報告されている整数秒と、このモニター・エレメントに関する _MS で終わっている列に報告されている小数秒を、次の式を使用して加算しなければなりません: (*monitor-element-name*_S × 1,000,000 + *monitor-element-name*_MS) ÷ 1,000,000. 例えば、(ELAPSED_EXEC_TIME_S × 1,000,000 + ELAPSED_EXEC_TIME_MS) ÷ 1,000,000。		

SNAPBP 管理ビューおよび SNAP_GET_BP_V95 表関数 - bufferpool 論理グループのスナップショット情報の検索

SNAPBP 管理ビューおよび SNAP_GET_BP_V95 表関数は、bufferpool スナップショットから、特に bufferpool 論理データ・グループのバッファー・プール情報を戻します。

SNAPBP 管理ビュー

この管理ビューを使用して、現在接続中のデータベースに関する bufferpool 論理グループのスナップショット情報を取得することができます。

SNAPBP 管理ビューを SNAPBP_PART 管理ビューとともに使用すると、GET SNAPSHOT FOR BUFFERPOOLS ON database-alias CLP コマンドに相当するデータが提供されます。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、664 ページの表 173を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPBP 管理ビューに対する SELECT 特権
- SNAPBP 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_BP_V95 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続中のデータベースのすべてのバッファー・プールについて、データおよび索引の書き込みを取得します。

```
SELECT SUBSTR(DB_NAME,1,8) AS DB_NAME,SUBSTR(BP_NAME,1,15)
   AS BP_NAME,POOL_DATA_WRITES,POOL_INDEX_WRITES
   FROM SYSIBMADM.SNAPBP
```

以下はこの照会の出力例です。

```
DB_NAME  BP_NAME         POOL_DATA_WRITES     POOL_INDEX_WRITES
-------- --------------- -------------------- --------------------
TEST     IBMDEFAULTBP                       0                    0
TEST     IBMSYSTEMBP4K                      0                    0
TEST     IBMSYSTEMBP8K                      0                    0
TEST     IBMSYSTEMBP16K                     0                    0
TEST     IBMSYSTEMBP32K                     0                    0

5 record(s) selected
```

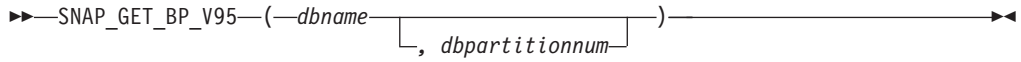
SNAP_GET_BP_V95 表関数

SNAP_GET_BP_V95 表関数は SNAPBP 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションに関する情報を取得することができます。

SNAP_GET_BP_V95 表関数を SNAP_GET_BP_PART 表関数とともに使用すると、GET SNAPSHOT FOR ALL BUFFERPOOLS CLP コマンドに相当するデータが提供されます。

戻される可能性のある情報の完全なリストは、664 ページの表 173を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、空ストリングを指定します。現在接続されているデータベースと同じインスタンス内のすべてのデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプを持つファイルが存在しない場合、SNAP_GET_BP_V95 表関数は、現在接続中のデータベースおよびデータベース・パーティション番号のスナップショットを取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_BP_V95 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT

- SYSADM

例

現在接続されているデータベース・パーティションのすべてのアクティブ・データベースのすべてのバッファー・プールについて、物理および論理読み取りの合計を取得します。

```
SELECT SUBSTR(T.DB_NAME,1,10) AS DB_NAME,
   SUBSTR(T.BP_NAME,1,20) AS BP_NAME,
   (T.POOL_DATA_L_READS+T.POOL_INDEX_L_READS) AS TOTAL_LOGICAL_READS,
   (T.POOL_DATA_P_READS+T.POOL_INDEX_P_READS) AS TOTAL_PHYSICAL_READS,
   T.DBPARTITIONNUM
   FROM TABLE(SNAP_GET_BP_V95(CAST(NULL AS VARCHAR(128)), -1)) AS T
```

以下はこの照会の出力例です。

```
DB_NAME    BP_NAME          TOTAL_LOGICAL_READS  ...
---------- ------------...- -------------------- ...
SAMPLE     IBMDEFAULTBP                       0 ...
TOOLSDB    IBMDEFAULTBP                       0 ...
TOOLSDB    BP32K0000                          0 ...

  3 record(s) selected.
```

この照会からの出力 (続き)。

```
... TOTAL_PHYSICAL_READS DBPARTITIONNUM
... -------------------- --------------
...                    0              0
...                    0              0
...                    0              0
```

戻される情報

表 173. SNAPBP 管理ビューおよび SNAP_GET_BP_V95 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
BP_NAME	VARCHAR(128)	bp_name - バッファー・プール名
DB_NAME	VARCHAR(128)	db_name - データベース名
DB_PATH	VARCHAR(1024)	db_path - データベース・パス
INPUT_DB_ALIAS	VARCHAR(128)	input_db_alias - 入力データベース別名
POOL_DATA_L_READS	BIGINT	pool_data_l_reads - バッファー・プール・データの論理読み取り
POOL_DATA_P_READS	BIGINT	pool_data_p_reads - バッファー・プール・データの物理読み取り
POOL_DATA_WRITES	BIGINT	pool_data_writes - バッファー・プールへのデータの書き込み
POOL_INDEX_L_READS	BIGINT	pool_index_l_reads - バッファー・プール索引の論理読み取り
POOL_INDEX_P_READS	BIGINT	pool_index_p_reads - バッファー・プール索引の物理読み取り

表 173. SNAPBP 管理ビューおよび SNAP_GET_BP_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
POOL_INDEX_WRITES	BIGINT	pool_index_writes - バッファー・プール索引の書き込み
POOL_XDA_L_READS	BIGINT	pool_xda_l_reads - バッファー・プール XDA データの論理読み取り
POOL_XDA_P_READS	BIGINT	pool_xda_p_reads - バッファー・プール XDA データの物理読み取り
POOL_XDA_WRITES	BIGINT	pool_xda_writes - バッファー・プール XDA データの書き込み
POOL_READ_TIME	BIGINT	pool_read_time - バッファー・プール物理読み取り時間の合計
POOL_WRITE_TIME	BIGINT	pool_write_time - バッファー・プール物理書き込み時間の合計
POOL_ASYNC_DATA_READS	BIGINT	pool_async_data_reads - バッファー・プール非同期データ読み取り
POOL_ASYNC_DATA_WRITES	BIGINT	pool_async_data_writes - バッファー・プール非同期データ書き込み
POOL_ASYNC_INDEX_READS	BIGINT	pool_async_index_reads - バッファー・プール非同期索引読み取り
POOL_ASYNC_INDEX_WRITES	BIGINT	pool_async_index_writes - バッファー・プール非同期索引書き込み
POOL_ASYNC_XDA_READS	BIGINT	pool_async_xda_reads - バッファー・プール非同期 XDA データ読み取り
POOL_ASYNC_XDA_WRITES	BIGINT	pool_async_xda_writes - バッファー・プール非同期 XDA データ書き込み
POOL_ASYNC_READ_TIME	BIGINT	pool_async_read_time - バッファー・プール非同期読み取り時間
POOL_ASYNC_WRITE_TIME	BIGINT	pool_async_write_time - バッファー・プール非同期書き込み時間
POOL_ASYNC_DATA_READ_REQS	BIGINT	pool_async_data_read_reqs - バッファー・プール非同期読み取り要求
POOL_ASYNC_INDEX_READ_REQS	BIGINT	pool_async_index_read_reqs - バッファー・プール非同期索引読み取り要求
POOL_ASYNC_XDA_READ_REQS	BIGINT	pool_async_xda_read_reqs - バッファー・プール非同期 XDA 読み取り要求
DIRECT_READS	BIGINT	direct_reads - データベースからの直接読み取り
DIRECT_WRITES	BIGINT	direct_writes - データベースへの直接書き込み
DIRECT_READ_REQS	BIGINT	direct_read_reqs - 直接読み取り要求

表 173. SNAPBP 管理ビューおよび SNAP_GET_BP_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
DIRECT_WRITE_REQS	BIGINT	direct_write_reqs - 直接書き込み要求
DIRECT_READ_TIME	BIGINT	direct_read_time - 直接読み取り時間
DIRECT_WRITE_TIME	BIGINT	direct_write_time - 直接書き込み時間
UNREAD_PREFETCH_PAGES	BIGINT	unread_prefetch_pages - 読み取り不能プリフェッチ・ページ
FILES_CLOSED	BIGINT	files_closed - 閉じられたデータベース・ファイル
POOL_TEMP_DATA_L_READS	BIGINT	pool_temp_data_l_reads - バッファー・プール一時データの論理読み取り
POOL_TEMP_DATA_P_READS	BIGINT	pool_temp_data_p_reads - バッファー・プール一時データの物理読み取り
POOL_TEMP_INDEX_L_READS	BIGINT	pool_temp_index_l_reads - バッファー・プール一時索引の論理読み取り
POOL_TEMP_INDEX_P_READS	BIGINT	pool_temp_index_p_reads - バッファー・プール一時索引の物理読み取り
POOL_TEMP_XDA_L_READS	BIGINT	pool_temp_xda_l_reads - バッファー・プール一時 XDA データの論理読み取り
POOL_TEMP_XDA_P_READS	BIGINT	pool_temp_xda_p_reads - バッファー・プール一時 XDA データの物理読み取り : モニター・エレメント
POOL_NO_VICTIM_BUFFER	BIGINT	pool_no_victim_buffer - バッファー・プールの非ビクティム・バッファー数
PAGES_FROM_BLOCK_IOS	BIGINT	pages_from_block_ios - ブロック入出力によって読み取られたページ数の合計
PAGES_FROM_VECTORED_IOS	BIGINT	pages_from_vectored_ios - ベクトル化入出力によって読み取られたページ数の合計
VECTORED_IOS	BIGINT	vectored_ios - ベクトル化入出力要求数
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

SNAPBP_PART 管理ビューおよび SNAP_GET_BP_PART 表関数 - bufferpool_nodeinfo 論理データ・グループのスナップショット情報の検索

SNAPBP_PART 管理ビューおよび SNAP_GET_BP_PART 表関数は、バッファー・プール・スナップショットからバッファー・プール情報について、特に bufferpool_nodeinfo 論理データ・グループのバッファー・プール情報を戻します。

SNAPBP_PART 管理ビュー

この管理ビューを使用して、現在接続中のデータベースに関する bufferpool_nodeinfo 論理データ・グループのスナップショット情報を取得することができます。

SNAPBP_PART 管理ビューを SNAPBP 管理ビューとともに使用すると、GET SNAPSHOT FOR BUFFERPOOLS ON database-alias CLP コマンドに相当するデータが提供されます。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、670 ページの表 174を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPBP_PART 管理ビューに対する SELECT 特権
- SNAPBP_PART 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_BP_PART 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

SAMPLE データベースへの接続中に、すべてのバッファー・プールのデータを取得します。

```
SELECT SUBSTR(DB_NAME,1,8) AS DB_NAME, SUBSTR(BP_NAME,1,15) AS BP_NAME,
   BP_CUR_BUFFSZ, BP_NEW_BUFFSZ, BP_PAGES_LEFT_TO_REMOVE, BP_TBSP_USE_COUNT
   FROM SYSIBMADM.SNAPBP_PART
```

以下はこの照会の出力例です。

```
DB_NAME  BP_NAME         BP_CUR_BUFFSZ        BP_NEW_BUFFSZ        ...
-------- --------------- -------------------- -------------------- ...
SAMPLE   IBMDEFAULTBP                   1000                 1000 ...
SAMPLE   IBMSYSTEMBP4K                    16                   16 ...
SAMPLE   IBMSYSTEMBP8K                    16                   16 ...
SAMPLE   IBMSYSTEMBP16K                   16                   16 ...
                                                                  ...
  4 record(s) selected.
```

この照会からの出力 (続き)。

```
... BP_PAGES_LEFT_TO_REMOVE BP_TBSP_USE_COUNT
... ----------------------- --------------------
...                       0                    3
...                       0                    0
...                       0                    0
...                       0                    0
...
```

SNAP_GET_BP_PART 表関数

SNAP_GET_BP_PART 表関数は、SNAPBP_PART 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションに関する情報を取得することができます。

SNAP_GET_BP_PART 表関数を SNAP_GET_BP_V95 表関数とともに使用すると、GET SNAPSHOT FOR ALL BUFFERPOOLS CLP コマンドに相当するデータが提供されます。

戻される可能性のある情報の完全なリストは、 670 ページの表 174を参照してください。

構文

スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、空ストリングを指定します。現在接続されているデータベースと同じインスタンス内のすべてのデータベースにある、すべてのバッファー・プールのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。

dbname が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプを持つファイルが存在しない場合、SNAP_GET_BP_PART 表関数は、現在接続中のデータベースおよびデータベース・パーティション番号のスナップショットを取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_BP_PART 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

SAMPLE データベースへの接続中に、すべてのアクティブ・データベースのすべてのバッファー・プールのデータを取得します。

```
SELECT SUBSTR(DB_NAME,1,8) AS DB_NAME, SUBSTR(BP_NAME,1,15) AS BP_NAME,
   BP_CUR_BUFFSZ, BP_NEW_BUFFSZ, BP_PAGES_LEFT_TO_REMOVE, BP_TBSP_USE_COUNT
   FROM TABLE(SNAP_GET_BP_PART(CAST(NULL AS VARCHAR(128)),-1)) AS T
```

以下はこの照会の出力例です。

```
DB_NAME  BP_NAME         BP_CUR_BUFFSZ        BP_NEW_BUFFSZ        ...
-------- --------------- -------------------- -------------------- ...
SAMPLE   IBMDEFAULTBP                    250                  250 ...
SAMPLE   IBMSYSTEMBP4K                    16                   16 ...
SAMPLE   IBMSYSTEMBP8K                    16                   16 ...
SAMPLE   IBMSYSTEMBP16K                   16                   16 ...
SAMPLE   IBMSYSTEMBP32K                   16                   16 ...
TESTDB   IBMDEFAULTBP                    250                  250 ...
TESTDB   IBMSYSTEMBP4K                    16                   16 ...
TESTDB   IBMSYSTEMBP8K                    16                   16 ...
TESTDB   IBMSYSTEMBP16K                   16                   16 ...
TESTDB   IBMSYSTEMBP32K                   16                   16 ...

...
```

この照会からの出力 (続き)。

```
... BP_PAGES_LEFT_TO_REMOVE BP_TBSP_USE_COUNT
... ---------------------- --------------------
...                      0                    3
...                      0                    0
...                      0                    0
...                      0                    0
...                      0                    0
...                      0                    3
...                      0                    0
...                      0                    0
...                      0                    0
...                      0                    0

...
```

戻される情報

表 174. SNAPBP_PART 管理ビューおよび SNAP_GET_BP_PART 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
BP_NAME	VARCHAR(128)	bp_name - バッファー・プール名
DB_NAME	VARCHAR(128)	db_name - データベース名
BP_CUR_BUFFSZ	BIGINT	bp_cur_buffsz - バッファー・プールの現行サイズ
BP_NEW_BUFFSZ	BIGINT	bp_new_buffsz - 新規バッファー・プール・サイズ
BP_PAGES_LEFT_TO_REMOVE	BIGINT	bp_pages_left_to_remove - 除去残ページ数
BP_TBSP_USE_COUNT	BIGINT	bp_tbsp_use_count - バッファー・プールにマップされている表スペースの数
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

SNAPCONTAINER 管理ビューおよび SNAP_GET_CONTAINER_V91 表関数 - tablespace_container 論理データ・グループ・スナップショット情報の検索

SNAPCONTAINER 管理ビューおよび SNAP_GET_CONTAINER_V91 表関数は、tablespace_container 論理データ・グループからの表スペース・スナップショット情報を戻します。

SNAPCONTAINER 管理ビュー

この管理ビューでは、現在接続されているデータベースの tablespace_container 論理データ・グループ・スナップショット情報を検索できます。

SNAPTBSP、SNAPTBSP_PART、SNAPTBSP_QUIESCER、および SNAPTBSP_RANGE 管理ビューと共に使用すると、SNAPCONTAINER 管理ビューは、GET SNAPSHOT FOR TABLESPACES ON database-alias CLP コマンドと同等のデータを戻します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 674 ページの表 175を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPCONTAINER 管理ビューに対する SELECT 特権
- SNAPCONTAINER 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_CONTAINER_V91 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベースのすべてのデータベース・パーティションの表スペース・コンテナーの詳細を検索します。

```
SELECT SNAPSHOT_TIMESTAMP, SUBSTR(TBSP_NAME, 1, 15) AS TBSP_NAME,
   TBSP_ID, SUBSTR(CONTAINER_NAME, 1, 20) AS CONTAINER_NAME,
   CONTAINER_ID, CONTAINER_TYPE, ACCESSIBLE, DBPARTITIONNUM
   FROM SYSIBMADM.SNAPCONTAINER ORDER BY DBPARTITIONNUM
```

以下はこの照会の出力例です。

```
SNAPSHOT_TIMESTAMP         TBSP_NAME       TBSP_ID     ...
-------------------------- --------------- --------...- ...
2006-01-08-16.49.24.639945 SYSCATSPACE               0 ...
2006-01-08-16.49.24.639945 TEMPSPACE1                1 ...
2006-01-08-16.49.24.639945 USERSPACE1                2 ...
2006-01-08-16.49.24.639945 SYSTOOLSPACE              3 ...
2006-01-08-16.49.24.640747 TEMPSPACE1                1 ...
2006-01-08-16.49.24.640747 USERSPACE1                2 ...
2006-01-08-16.49.24.639981 TEMPSPACE1                1 ...
2006-01-08-16.49.24.639981 USERSPACE1                2 ...
                                                       ...
8 record(s) selected.
```

この照会からの出力 (続き)。

```
... CONTAINER_NAME       CONTAINER_ID     CONTAINER_TYPE   ...
... -------------------- ------------...- ---------------- ...
... /home/swalkty/swalkt                0 FILE_EXTENT_TAG  ...
... /home/swalkty/swalkt                0 PATH             ...
... /home/swalkty/swalkt                0 FILE_EXTENT_TAG  ...
... /home/swalkty/swalkt                0 FILE_EXTENT_TAG  ...
... /home/swalkty/swalkt                0 PATH             ...
... /home/swalkty/swalkt                0 FILE_EXTENT_TAG  ...
... /home/swalkty/swalkt                0 PATH             ...
... /home/swalkty/swalkt                0 FILE_EXTENT_TAG  ...
```

この照会からの出力 (続き)。

```
... ACCESSIBLE DBPARTITIONNUM
... ---------- --------------
...          1              0
...          1              0
...          1              0
...          1              0
...          1              1
...          1              1
...          1              2
...          1              2
```

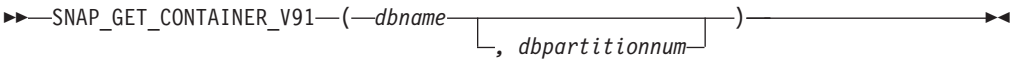
SNAP_GET_CONTAINER_V91 表関数

SNAP_GET_CONTAINER_V91 表関数は SNAPCONTAINER 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションのいずれかの特定のデータベースを対象とした情報を検索することができます。

SNAP_GET_TBSP_V91、 SNAP_GET_TBSP_PART_V91、 SNAP_GET_TBSP_QUIESCER、および SNAP_GET_TBSP_RANGE 表関数と共に使用すると、SNAP_GET_CONTAINER_V91 表関数は、GET SNAPSHOT FOR TABLESPACES ON database-alias CLP コマンドと同等のデータを戻します。

戻される可能性のある情報の完全なリストは、 674 ページの表 175を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL または空ストリングを指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプが含まれるファイルが存在しない場合、SNAP_GET_CONTAINER_V91 表関数は、現在接続されているデータベースのスナップショットとデータベース・パーティション番号を取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_CONTAINER_V91 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベース・パーティション上で現在接続されているデータベースの表スペース・コンテナーの詳細を検索します。

```
SELECT SNAPSHOT_TIMESTAMP, TBSP_NAME, TBSP_ID, CONTAINER_NAME,
   CONTAINER_ID, CONTAINER_TYPE, ACCESSIBLE
   FROM TABLE(SNAP_GET_CONTAINER_V91('',-1)) AS T
```

以下はこの照会の出力例です。

```
SNAPSHOT_TIMESTAMP         TBSP_NAME            TBSP_ID ...
-------------------------- -------------------- ------- ...
2005-04-25-14.42.10.899253 SYSCATSPACE                0 ...
2005-04-25-14.42.10.899253 TEMPSPACE1                 1 ...
2005-04-25-14.42.10.899253 USERSPACE1                 2 ...
2005-04-25-14.42.10.899253 SYSTOOLSPACE               3 ...
2005-04-25-14.42.10.899253 MYTEMP                     4 ...
2005-04-25-14.42.10.899253 WHATSNEWTEMPSPACE          5 ...
```

この照会からの出力 (続き)。

```
... CONTAINER_NAME                                CONTAINER_ID ...
... ----------------------------------------- ------------ ...
... D:¥DB2¥NODE0000¥SQL00002¥SQLT0000.0                  0 ...
... D:¥DB2¥NODE0000¥SQL00002¥SQLT0001.0                  0 ...
... D:¥DB2¥NODE0000¥SQL00002¥SQLT0002.0                  0 ...
... D:¥DB2¥NODE0000¥SQL00002¥SYSTOOLSPACE                0 ...
... D:¥DB2¥NODE0000¥SQL003                               0 ...
... d:¥DGTTsWhatsNewContainer                            0 ...
```

この照会からの出力 (続き)。

```
... CONTAINER_TYPE ACCESSIBLE
... -------------- ----------
... CONT_PATH               1
... CONT_PATH               1
... CONT_PATH               1
... CONT_PATH               1
... CONT_PATH               1
... CONT_PATH               1
```

戻される情報

注: ファイル・システム情報を戻すためには、BUFFERPOOL データベース・マネージャーのモニター・スイッチをオンにする必要があります。

表 175. SNAPCONTAINER 管理ビューおよび SNAP_GET_CONTAINER_V91 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
TBSP_NAME	VARCHAR(128)	tablespace_name - 表スペース名
TBSP_ID	BIGINT	tablespace_id - 表スペース ID
CONTAINER_NAME	VARCHAR(256)	container_name - コンテナー名
CONTAINER_ID	BIGINT	container_id - コンテナー ID
CONTAINER_TYPE	VARCHAR(16)	container_type - コンテナー・タイプ。これは、sqlutil.h での定義を基にしたテキスト ID です。以下のいずれかとなります。 • DISK_EXTENT_TAG • DISK_PAGE_TAG • FILE_EXTENT_TAG • FILE_PAGE_TAG • PATH
TOTAL_PAGES	BIGINT	container_total_pages - コンテナー内の合計ページ数
USABLE_PAGES	BIGINT	container_usable_pages - コンテナー内の使用可能なページ数
ACCESSIBLE	SMALLINT	container_accessible - コンテナーのアクセス可能性
STRIPE_SET	BIGINT	container_stripe_set - ストライプ・セット

表 175. SNAPCONTAINER 管理ビューおよび SNAP_GET_CONTAINER_V91 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。
FS_ID	VARCHAR(22)	fs_id - 固有のファイル・システム識別番号
FS_TOTAL_SIZE	BIGINT	fs_total_size - ファイル・システムの合計サイズ
FS_USED_SIZE	BIGINT	fs_used_size - ファイル・システム上で使用されるスペースの量

SNAPDB 管理ビューおよび SNAP_GET_DB_V97 表関数 - dbase 論理グループからのスナップショット情報の検索

SNAPDB 管理ビューおよび SNAP_GET_DB_V97 表関数は、データベース (dbase) 論理グループからのスナップショット情報を戻します。

SNAPDB 管理ビュー

この管理ビューを使用すると、現在接続されているデータベースに関するスナップショット情報を dbase 論理グループから検索できます。

SNAPDB 管理ビューを SNAPDB_MEMORY_POOL、 SNAPDETAILLOG、SNAPHADR、および SNAPSTORAGE_PATHS 管理ビューと併せて使用することにより、GET SNAPSHOT FOR DATABASE on database-alias CLP コマンドと同等の情報を戻します。

スキーマは SYSIBMADM です。

戻される情報の完全なリストは、 679 ページの表 176 を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPDB 管理ビューに対する SELECT 特権
- SNAPDB 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_DB_V97 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL

- SYSMAINT
- SYSADM

例

現在接続されているデータベースのすべてのデータベース・パーティションに関する状況、プラットフォーム、ロケーション、および接続時間を取り出します。

```
SELECT SUBSTR(DB_NAME, 1, 20) AS DB_NAME, DB_STATUS, SERVER_PLATFORM,
   DB_LOCATION, DB_CONN_TIME, DBPARTITIONNUM
   FROM SYSIBMADM.SNAPDB ORDER BY DBPARTITIONNUM
```

以下はこの照会の出力例です。

```
DB_NAME     DB_STATUS    SERVER_PLATFORM DB_LOCATION  ...
-------...- ------------ --------------- ------------ ...
TEST        ACTIVE       AIX64           LOCAL        ...
TEST        ACTIVE       AIX64           LOCAL        ...
TEST        ACTIVE       AIX64           LOCAL        ...

3 record(s) selected.
```

この照会からの出力 (続き)。

```
... DB_CONN_TIME               DBPARTITIONNUM
... -------------------------- --------------
... 2006-01-08-16.48.30.665477              0
... 2006-01-08-16.48.34.005328              1
... 2006-01-08-16.48.34.007937              2
```

このルーチンは、コマンド行で以下を呼び出すことにより使用できます。

```
SELECT TOTAL_OLAP_FUNCS, OLAP_FUNC_OVERFLOWS, ACTIVE_OLAP_FUNCS
   FROM SYSIBMADM.SNAPDB

TOTAL_OLAP_FUNCS     OLAP_FUNC_OVERFLOWS  ACTIVE_OLAP_FUNCS
-------------------- -------------------- -----------------
                   7                    2                 1

  1 record(s) selected.
```

ワークロードの実行後に、ユーザーは次の照会を使用できます。

```
SELECT STATS_CACHE_SIZE, STATS_FABRICATIONS, SYNC_RUNSTATS,
   ASYNC_RUNSTATS, STATS_FABRICATE_TIME, SYNC_RUNSTATS_TIME
   FROM SYSIBMADM.SNAPDB

STATS_CACHE_SIZE STATS_FABRICATIONS SYNC_RUNSTATS ASYNC_RUNSTATS ...
---------------- ----------------- ------------- -------------- ...
             128                 2             1              0 ...


... STATS_FABRICATE_TIME SYNC_RUNSTATS_TIME
... -------------------- ------------------
...                   10                100

1 record(s) selected.
```

SNAP_GET_DB_V97 表関数

SNAP_GET_DB_V97 表関数は、SNAPDB 管理ビューと同じ情報を戻します。

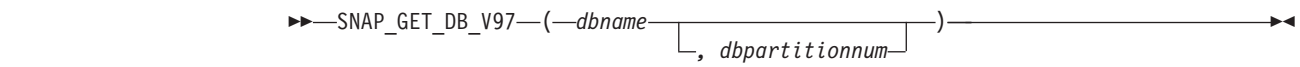
注: バージョン 9.7 のフィックスパック 1 以前で作成されたデータベースを使用している場合、このルーチンを実行するには、既に db2updv97 コマンドを実行済みで

なければなりません。バージョン 9.7 以前に作成されたデータベースを使用している場合は、db2updv97 コマンドを実行する必要はありません (データベースのマイグレーションによって自動的にカタログ更新が実行されるため)。バージョン 9.7 にダウングレードすると、このルーチンは機能しなくなります。

SNAP_GET_DB_V97 表関数を SNAP_GET_DB_MEMORY_POOL、SNAP_GET_DETAILLOG_V91、 SNAP_GET_HADR、および SNAP_GET_STORAGE_PATHS 表関数と併せて使用することにより、GET SNAPSHOT FOR ALL DATABASES CLP コマンドと同等の情報が戻されます。

戻される情報の完全なリストは、 679 ページの表 176 を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、空ストリングを指定します。現在接続されているデータベースと同じインスタンス内のすべてのデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプを持つファイルが存在しない場合には、現在接続されているデータベースのスナップショットとデータベース・パーティション番号を SNAP_GET_DB_V97 表関数が取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_DB_V97 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

例 1: 現在接続されているデータベースのすべてのデータベース・パーティションに渡る集約ビューとして、状況、プラットフォーム、ロケーション、および接続時間を取り出します。

```
SELECT SUBSTR(DB_NAME, 1, 20) AS DB_NAME, DB_STATUS, SERVER_PLATFORM,
   DB_LOCATION, DB_CONN_TIME FROM TABLE(SNAP_GET_DB_V97('', -2)) AS T
```

以下はこの照会の出力例です。

```
DB_NAME     DB_STATUS    SERVER_PLATFORM ...
--------...- ---------... --------------- ...
SAMPLE      ACTIVE       AIX64           ...

  1 record(s) selected.
```

この照会からの出力 (続き)。

```
... DB_LOCATION  DB_CONN_TIME
... ------------ --------------------------
... LOCAL        2005-07-24-22.09.22.013196
```

例 2: 現在接続されているデータベースを含む同じインスタンス内にあるすべてのアクティブ・データベースのすべてのデータベース・パーティションに渡る集約ビューとして、状況、プラットフォーム、ロケーション、および接続時間を取り出します。

```
SELECT SUBSTR(DB_NAME, 1, 20) AS DB_NAME, DB_STATUS, SERVER_PLATFORM,
   DB_LOCATION, DB_CONN_TIME
   FROM TABLE(SNAP_GET_DB_V97(CAST (NULL AS VARCHAR(128)), -2)) AS T
```

以下はこの照会の出力例です。

```
DB_NAME      DB_STATUS    SERVER_PLATFORM ...
--------...- ---------... --------------- ...
TOOLSDB      ACTIVE       AIX64           ...
SAMPLE       ACTIVE       AIX64           ...
```

この照会からの出力 (続き)。

```
... DB_LOCATION  DB_CONN_TIME
... ------------ --------------------------
... LOCAL        2005-07-24-22.26.54.396335
... LOCAL        2005-07-24-22.09.22.013196
```

例 3: このルーチンは、データベースへの接続時にコマンド行で以下を呼び出すことにより使用できます。

```
SELECT TOTAL_OLAP_FUNCS, OLAP_FUNC_OVERFLOWS, ACTIVE_OLAP_FUNCS
   FROM TABLE (SNAP_GET_DB_V97('', 0)) AS T
```

出力は次のようになります。

```
TOTAL_OLAP_FUNCS  OLAP_FUNC_OVERFLOWS  ACTIVE_OLAP_FUNCS
----------------  -------------------- --------------------
               7                    2                    1

1 record(s) selected.
```

例 4: ワークロードの実行後に、ユーザーは次の照会を表関数とともに使用できます。

```
 SELECT STATS_CACHE_SIZE, STATS_FABRICATIONS, SYNC_RUNSTATS,
   ASYNC_RUNSTATS, STATS_FABRICATE_TIME, SYNC_RUNSTATS_TIME
   FROM TABLE (SNAP_GET_DB_V97('mytestdb', -1)) AS SNAPDB

STATS_CACHE_SIZE STATS_FABRICATIONS SYNC_RUNSTATS ASYNC_RUNSTATS ...
---------------- ----------------- ------------- -------------- ...
             200                 1             2              0 ...

Continued

...STATS_FABRICATE_TIME  SYNC_RUNSTATS_TIME
...--------------------  ------------------
...                   2                  32

  1 record(s) selected.
```

例 5: 以下の例は、SNAP_GET_DB_V97 表関数を使ってデータベースの状況を判別する方法を示しています。

```
SELECT SUBSTR
   (DB_NAME, 1, 20) AS DB_NAME, DB_STATUS
    FROM  table(SNAP_GET_DB_V97('hadrdb', 0))

DB_NAME              DB_STATUS
-------------------- ----------------
HADRDB               ACTIVE_STANDBY
```

SNAPDB 管理ビューおよび SNAP_GET_DB_V97 表関数のメタデータ

表 176. SNAPDB 管理ビューおよび SNAP_GET_DB_V97 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
DB_NAME	VARCHAR(128)	db_name - データベース名
DB_PATH	VARCHAR(1024)	db_path - データベース・パス
INPUT_DB_ALIAS	VARCHAR(128)	input_db_alias - 入力データベース別名

表 176. SNAPDB 管理ビューおよび SNAP_GET_DB_V97 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
DB_STATUS	VARCHAR(16)	db_status - データベース状況。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • ACTIVE • QUIESCE_PEND • QUIESCED • ROLLFWD • ACTIVE_STANDBY - HADR データベースはスタンバイ・モードで、スタンバイ中の読み取りが有効になっています。 • STANDBY - HADR データベースはスタンバイ・モードです (スタンバイ中の読み取りは有効になっていません)。
CATALOG_PARTITION	SMALLINT	catalog_node - カタログ・ノード番号
CATALOG_PARTITION_NAME	VARCHAR(128)	catalog_node_name - カタログ・ノード・ネットワーク名

表 176. SNAPDB 管理ビューおよび SNAP_GET_DB_V97 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
SERVER_PLATFORM	VARCHAR(12)	server_platform - サーバーのオペレーティング・システム。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • AIX • AIX64 • AS400_DRDA • DOS • DYNIX • HP • HP64 • HPIA • HPIA64 • LINUX • LINUX390 • LINUXIA64 • LINUXPPC • LINUXPPC64 • LINUXX8664 • LINUXZ64 • MAC • MVS_DRDA • NT • NT64 • OS2 • OS390 • SCO • SGI • SNI • SUN • SUN64 • UNKNOWN • UNKNOWN_DRDA • VM_DRDA • VSE_DRDA • WINDOWS

表 176. SNAPDB 管理ビューおよび SNAP_GET_DB_V97 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
DB_LOCATION	VARCHAR(12)	db_location - データベース・ロケーション。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • LOCAL • REMOTE
DB_CONN_TIME	TIMESTAMP	db_conn_time - データベース活動化タイム・スタンプ
LAST_RESET	TIMESTAMP	last_reset - 最後のリセット・タイム・スタンプ
LAST_BACKUP	TIMESTAMP	last_backup - 最終バックアップ・タイム・スタンプ
CONNECTIONS_TOP	BIGINT	connections_top - 同時接続の最大数
TOTAL_CONS	BIGINT	total_cons - データベース活動化以降の接続
TOTAL_SEC_CONS	BIGINT	total_sec_cons - 2 次接続
APPLS_CUR_CONS	BIGINT	appls_cur_cons - 現在接続されているアプリケーション
APPLS_IN_DB2	BIGINT	appls_in_db2 - データベースで現在実行中のアプリケーション
NUM_ASSOC_AGENTS	BIGINT	num_assoc_agents - 関連したエージェント数
AGENTS_TOP	BIGINT	agents_top - 作成されたエージェントの数
COORD_AGENTS_TOP	BIGINT	coord_agents_top - コーディネーター・エージェント最大数
LOCKS_HELD	BIGINT	locks_held - ロック保持数
LOCK_WAITS	BIGINT	lock_waits - ロック待機数
LOCK_WAIT_TIME	BIGINT	lock_wait_time - ロック待機中の時間
LOCK_LIST_IN_USE	BIGINT	lock_list_in_use - 使用中のロック・リスト・メモリーの合計
DEADLOCKS	BIGINT	deadlocks - デッドロック検出数
LOCK_ESCALS	BIGINT	lock_escals - ロック・エスカレーション数
X_LOCK_ESCALS	BIGINT	x_lock_escals - 排他ロック・エスカレーション数
LOCKS_WAITING	BIGINT	locks_waiting - ロックで待機中の現行エージェント
LOCK_TIMEOUTS	BIGINT	lock_timeouts - ロック・タイムアウト数
NUM_INDOUBT_TRANS	BIGINT	num_indoubt_trans - 未確定トランザクション数
SORT_HEAP_ALLOCATED	BIGINT	sort_heap_allocated - 割り振られたソート・ヒープの合計
SORT_SHRHEAP_ALLOCATED	BIGINT	sort_shrheap_allocated - 現在割り振られているソート共有ヒープ
SORT_SHRHEAP_TOP	BIGINT	sort_shrheap_top - ソート共有ヒープの最高水準点

表 176. SNAPDB 管理ビューおよび SNAP_GET_DB_V97 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
POST_SHRTHRESHOLD_SORTS	BIGINT	post_shrthreshold_sorts - ポスト共有しきい値ソート
TOTAL_SORTS	BIGINT	total_sorts - ソート合計
TOTAL_SORT_TIME	BIGINT	total_sort_time - ソート時間合計
SORT_OVERFLOWS	BIGINT	sort_overflows - ソート・オーバーフロー
ACTIVE_SORTS	BIGINT	active_sorts - アクティブ・ソート
POOL_DATA_L_READS	BIGINT	pool_data_l_reads - バッファー・プール・データの論理読み取り
POOL_DATA_P_READS	BIGINT	pool_data_p_reads - バッファー・プール・データの物理読み取り
POOL_TEMP_DATA_L_READS	BIGINT	pool_temp_data_l_reads - バッファー・プール一時データの論理読み取り
POOL_TEMP_DATA_P_READS	BIGINT	pool_temp_data_p_reads - バッファー・プール一時データの物理読み取り
POOL_ASYNC_DATA_READS	BIGINT	pool_async_data_reads - バッファー・プール非同期データ読み取り
POOL_DATA_WRITES	BIGINT	pool_data_writes - バッファー・プールへのデータの書き込み
POOL_ASYNC_DATA_WRITES	BIGINT	pool_async_data_writes - バッファー・プール非同期データ書き込み
POOL_INDEX_L_READS	BIGINT	pool_index_l_reads - バッファー・プール索引の論理読み取り
POOL_INDEX_P_READS	BIGINT	pool_index_p_reads - バッファー・プール索引の物理読み取り
POOL_TEMP_INDEX_L_READS	BIGINT	pool_temp_index_l_reads - バッファー・プール一時索引の論理読み取り
POOL_TEMP_INDEX_P_READS	BIGINT	pool_temp_index_p_reads - バッファー・プール一時索引の物理読み取り
POOL_ASYNC_INDEX_READS	BIGINT	pool_async_index_reads - バッファー・プール非同期索引読み取り
POOL_INDEX_WRITES	BIGINT	pool_index_writes - バッファー・プール索引の書き込み
POOL_ASYNC_INDEX_WRITES	BIGINT	pool_async_index_writes - バッファー・プール非同期索引書き込み
POOL_XDA_P_READS	BIGINT	pool_xda_p_reads - バッファー・プール XDA データの物理読み取り
POOL_XDA_L_READS	BIGINT	pool_xda_l_reads - バッファー・プール XDA データの論理読み取り
POOL_XDA_WRITES	BIGINT	pool_xda_writes - バッファー・プール XDA データの書き込み
POOL_ASYNC_XDA_READS	BIGINT	pool_async_xda_reads - バッファー・プール非同期 XDA データ読み取り

表 176. SNAPDB 管理ビューおよび SNAP_GET_DB_V97 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
POOL_ASYNC_XDA_WRITES	BIGINT	pool_async_xda_writes - バッファー・プール非同期 XDA データ書き込み
POOL_TEMP_XDA_P_READS	BIGINT	pool_temp_xda_p_reads - バッファー・プール一時 XDA データの物理読み取り : モニター・エレメント
POOL_TEMP_XDA_L_READS	BIGINT	pool_temp_xda_l_reads - バッファー・プール一時 XDA データの論理読み取り
POOL_READ_TIME	BIGINT	pool_read_time - バッファー・プール物理読み取り時間の合計
POOL_WRITE_TIME	BIGINT	pool_write_time - バッファー・プール物理書き込み時間の合計
POOL_ASYNC_READ_TIME	BIGINT	pool_async_read_time - バッファー・プール非同期読み取り時間
POOL_ASYNC_WRITE_TIME	BIGINT	pool_async_write_time - バッファー・プール非同期書き込み時間
POOL_ASYNC_DATA_ READ_REQS	BIGINT	pool_async_data_read_reqs - バッファー・プール非同期読み取り要求
POOL_ASYNC_INDEX_ READ_REQS	BIGINT	pool_async_index_read_reqs - バッファー・プール非同期索引読み取り要求
POOL_ASYNC_XDA_ READ_REQS	BIGINT	pool_async_xda_read_reqs - バッファー・プール非同期 XDA 読み取り要求
POOL_NO_VICTIM_BUFFER	BIGINT	pool_no_victim_buffer - バッファー・プールの非ビクティム・バッファー数
POOL_LSN_GAP_CLNS	BIGINT	pool_lsn_gap_clns - 起動されたバッファー・プール・ログ・スペース・クリーナー
POOL_DRTY_PG_STEAL_CLNS	BIGINT	pool_drty_pg_steal_clns - 起動されたバッファー・プール・ビクティム・ページ・クリーナー
POOL_DRTY_PG_THRSH_CLNS	BIGINT	pool_drty_pg_thrsh_clns - 起動されたバッファー・プールしきい値クリーナー
PREFETCH_WAIT_TIME	BIGINT	prefetch_wait_time - プリフェッチ待ち時間
UNREAD_PREFETCH_PAGES	BIGINT	unread_prefetch_pages - 読み取り不能プリフェッチ・ページ
DIRECT_READS	BIGINT	direct_reads - データベースからの直接読み取り
DIRECT_WRITES	BIGINT	direct_writes - データベースへの直接書き込み
DIRECT_READ_REQS	BIGINT	direct_read_reqs - 直接読み取り要求
DIRECT_WRITE_REQS	BIGINT	direct_write_reqs - 直接書き込み要求
DIRECT_READ_TIME	BIGINT	direct_read_time - 直接読み取り時間
DIRECT_WRITE_TIME	BIGINT	direct_write_time - 直接書き込み時間

表 176. SNAPDB 管理ビューおよび SNAP_GET_DB_V97 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
FILES_CLOSED	BIGINT	files_closed - 閉じられたデータベース・ファイル
ELAPSED_EXEC_TIME_S	BIGINT	elapsed_exec_time - ステートメント実行経過時間
ELAPSED_EXEC_TIME_MS	BIGINT	elapsed_exec_time - ステートメント実行経過時間
COMMIT_SQL_STMTS	BIGINT	commit_sql_stmts - 試行されたコミット・ステートメント
ROLLBACK_SQL_STMTS	BIGINT	rollback_sql_stmts - 試行されたロールバック・ステートメント
DYNAMIC_SQL_STMTS	BIGINT	dynamic_sql_stmts - 試行された動的 SQL ステートメント
STATIC_SQL_STMTS	BIGINT	static_sql_stmts - 試行された静的 SQL ステートメント
FAILED_SQL_STMTS	BIGINT	failed_sql_stmts - 失敗したステートメント操作
SELECT_SQL_STMTS	BIGINT	select_sql_stmts - 実行された選択 SQL ステートメント
UID_SQL_STMTS	BIGINT	uid_sql_stmts - 実行された更新/挿入/削除 SQL ステートメント
DDL_SQL_STMTS	BIGINT	ddl_sql_stmts - データ定義言語 (DDL) SQL ステートメント
INT_AUTO_REBINDS	BIGINT	int_auto_rebinds - 内部自動再バインド
INT_ROWS_DELETED	BIGINT	int_rows_deleted - 削除された内部行数
INT_ROWS_INSERTED	BIGINT	int_rows_inserted - 挿入された内部行数
INT_ROWS_UPDATED	BIGINT	int_rows_updated - 更新された内部行数
INT_COMMITS	BIGINT	int_commits - 内部コミット数
INT_ROLLBACKS	BIGINT	int_rollbacks - 内部ロールバック数
INT_DEADLOCK_ROLLBACKS	BIGINT	int_deadlock_rollbacks - デッドロックによる内部ロールバック数
ROWS_DELETED	BIGINT	rows_deleted - 削除行数
ROWS_INSERTED	BIGINT	rows_inserted - 挿入行数
ROWS_UPDATED	BIGINT	rows_updated - 更新行数
ROWS_SELECTED	BIGINT	rows_selected - 選択行数
ROWS_READ	BIGINT	rows_read - 読み取り行数
BINDS_PRECOMPILES	BIGINT	binds_precompiles - 試行されたバインド/プリコンパイル
TOTAL_LOG_AVAILABLE	BIGINT	total_log_available - 使用可能なログ合計
TOTAL_LOG_USED	BIGINT	total_log_used - 使用されているログ・スペースの合計
SEC_LOG_USED_TOP	BIGINT	sec_log_used_top - 使用された最大 2 次ログ・スペース

表 176. SNAPDB 管理ビューおよび SNAP_GET_DB_V97 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
TOT_LOG_USED_TOP	BIGINT	tot_log_used_top - 使用された最大合計ログ・スペース
SEC_LOGS_ALLOCATED	BIGINT	sec_logs_allocated - 現在割り振られている 2 次ログ
LOG_READS	BIGINT	log_reads - 読み取られたログ・ページの数
LOG_READ_TIME_S	BIGINT	log_read_time - ログ読み取り時間
LOG_READ_TIME_NS	BIGINT	log_read_time - ログ読み取り時間
LOG_WRITES	BIGINT	log_writes - 書き込まれたログ・ページの数
LOG_WRITE_TIME_S	BIGINT	log_write_time - ログ書き込み時間
LOG_WRITE_TIME_NS	BIGINT	log_write_time - ログ書き込み時間
NUM_LOG_WRITE_IO	BIGINT	num_log_write_io - ログ書き込み数
NUM_LOG_READ_IO	BIGINT	num_log_read_io - ログ読み取り数
NUM_LOG_PART_PAGE_IO	BIGINT	num_log_part_page_io - 部分ログ・ページ書き込み数
NUM_LOG_BUFFER_FULL	BIGINT	num_log_buffer_full - フル・ログ・バッファーの回数
NUM_LOG_DATA_FOUND_ IN_BUFFER	BIGINT	num_log_data_found_in_buffer - ログ・データがバッファーにある回数
APPL_ID_OLDEST_XACT	BIGINT	appl_id_oldest_xact - 最も古いトランザクションを使用するアプリケーション
LOG_TO_REDO_FOR_ RECOVERY	BIGINT	log_to_redo_for_recovery - リカバリーの場合に再実行されるログの量
LOG_HELD_BY_DIRTY_PAGES	BIGINT	log_held_by_dirty_pages - ダーティー・ページ別に計算されるログ・スペースの量
PKG_CACHE_LOOKUPS	BIGINT	pkg_cache_lookups - パッケージ・キャッシュ参照
PKG_CACHE_INSERTS	BIGINT	pkg_cache_inserts - パッケージ・キャッシュ挿入
PKG_CACHE_NUM_ OVERFLOWS	BIGINT	pkg_cache_num_overflows - パッケージ・キャッシュ・オーバーフロー数
PKG_CACHE_SIZE_TOP	BIGINT	pkg_cache_size_top - パッケージ・キャッシュ最高水準点
APPL_SECTION_LOOKUPS	BIGINT	appl_section_lookups - セクションの参照回数
APPL_SECTION_INSERTS	BIGINT	appl_section_inserts - セクション挿入数
CAT_CACHE_LOOKUPS	BIGINT	cat_cache_lookups - カタログ・キャッシュ参照数
CAT_CACHE_INSERTS	BIGINT	cat_cache_inserts - カタログ・キャッシュ挿入数
CAT_CACHE_OVERFLOWS	BIGINT	cat_cache_overflows - カタログ・キャッシュ・オーバーフロー数

表 176. SNAPDB 管理ビューおよび SNAP_GET_DB_V97 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
CAT_CACHE_SIZE_TOP	BIGINT	cat_cache_size_top - カタログ・キャッシュ最高水準点
PRIV_WORKSPACE_SIZE_TOP	BIGINT	priv_workspace_size_top - 専用ワークスペースの最大サイズ
PRIV_WORKSPACE_NUM_OVERFLOWS	BIGINT	priv_workspace_num_overflows - 専用ワークスペースのオーバーフロー回数
PRIV_WORKSPACE_SECTION_INSERTS	BIGINT	priv_workspace_section_inserts - 専用ワークスペース・セクション挿入
PRIV_WORKSPACE_SECTION_LOOKUPS	BIGINT	priv_workspace_section_lookups - 専用ワークスペース・セクションの参照
SHR_WORKSPACE_SIZE_TOP	BIGINT	shr_workspace_size_top - 最大共有ワークスペース・サイズ
SHR_WORKSPACE_NUM_OVERFLOWS	BIGINT	shr_workspace_num_overflows - 共有ワークスペースのオーバーフロー回数
SHR_WORKSPACE_SECTION_INSERTS	BIGINT	shr_workspace_section_inserts - 共有ワークスペース・セクション挿入数
SHR_WORKSPACE_SECTION_LOOKUPS	BIGINT	shr_workspace_section_lookups - 共有ワークスペース・セクションの参照回数
TOTAL_HASH_JOINS	BIGINT	total_hash_joins - ハッシュ結合の合計
TOTAL_HASH_LOOPS	BIGINT	total_hash_loops - ハッシュ・ループの合計
HASH_JOIN_OVERFLOWS	BIGINT	hash_join_overflows - ハッシュ結合のオーバーフロー
HASH_JOIN_SMALL_OVERFLOWS	BIGINT	hash_join_small_overflows - ハッシュ結合の短精度オーバーフロー
POST_SHRTHRESHOLD_HASH_JOINS	BIGINT	post_shrthreshold_hash_joins - ポストしきい値ハッシュ結合
ACTIVE_HASH_JOINS	BIGINT	active_hash_joins - アクティブ・ハッシュ結合
NUM_DB_STORAGE_PATHS	BIGINT	num_db_storage_paths - 自動ストレージ・パスの数
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。
SMALLEST_LOG_AVAIL_NODE	INTEGER	smallest_log_avail_node - 使用可能なログ・スペースが最小のノード
TOTAL_OLAP_FUNCS	BIGINT	実行される OLAP 関数の合計数。
OLAP_FUNC_OVERFLOWS	BIGINT	OLAP 関数データが使用可能なソート・ヒープ・スペースを超えた回数。

表 176. SNAPDB 管理ビューおよび SNAP_GET_DB_V97 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
ACTIVE_OLAP_FUNCS	BIGINT	現在実行中で、ソート・ヒープ・メモリーを消費している OLAP 関数の合計数。
STATS_CACHE_SIZE	BIGINT	統計キャッシュのサイズ (バイト)。
STATS_FABRICATIONS	BIGINT	表または索引のスキャンを実行しないでシステムが統計を作成するための statistics-collect アクティビティーの合計数。
SYNC_RUNSTATS	BIGINT	照会コンパイル中の同期 statistics-collect アクティビティーの合計数。
ASYNC_RUNSTATS	BIGINT	この列の出力は、成功した非同期 statistics-collect アクティビティーの合計数に変更されます。
STATS_FABRICATE_TIME	BIGINT	照会コンパイル中に表または索引のスキャンを実行しないでシステムが統計を作成するのに費やされる合計時間 (ミリ秒)。
SYNC_RUNSTATS_TIME	BIGINT	同期 statistics-collect アクティビティーに費やされる合計時間 (ミリ秒)。
NUM_THRESHOLD_VIOLATIONS	BIGINT	データベースで発生したしきい値違反の数。

SNAPDB_MEMORY_POOL 管理ビューおよび SNAP_GET_DB_MEMORY_POOL 表関数 - データベース・レベルのメモリー使用量情報の検索

SNAPDB_MEMORY_POOL 管理ビューおよび SNAP_GET_DB_MEMORY_POOL 表関数は、データベース・レベルでのメモリー使用量についての情報を戻します (UNIX プラットフォームの場合のみ)。

SNAPDB_MEMORY_POOL 管理ビュー

この管理ビューを使用して、現在接続中のデータベースに関するデータベース・レベルのメモリー使用量情報を取得することができます。

SNAPDB_MEMORY_POOL 管理ビューを SNAPDB、 SNAPDETAILLOG、 SNAPHADR、および SNAPSTORAGE_PATHS 管理ビューとともに使用すると、 GET SNAPSHOT FOR DATABASE ON database-alias CLP コマンドに相当する情報が提供されます。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 691 ページの表 177を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPDB_MEMORY_POOL 管理ビューに対する SELECT 特権

- SNAPDB_MEMORY_POOL 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_DB_MEMORY_POOL 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続中のデータベースである SAMPLE のメモリー・プールおよびそれらの現在のサイズのリストを取得します。

```
SELECT POOL_ID, POOL_CUR_SIZE FROM SYSIBMADM.SNAPDB_MEMORY_POOL
```

以下はこの照会の出力例です。

```
POOL_ID       POOL_CUR_SIZE
------------- --------------------
UTILITY                  32768
PACKAGE_CACHE           475136
CAT_CACHE                65536
BP                     2097152
BP                     1081344
BP                      540672
BP                      278528
BP                      147456
BP                       81920
LOCK_MGR                294912
DATABASE               3833856
OTHER                        0

  12 record(s) selected.
```

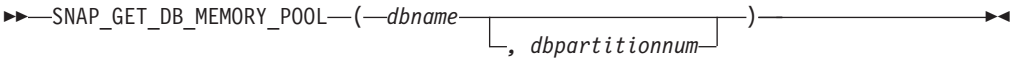
SNAP_GET_DB_MEMORY_POOL 表関数

SNAP_GET_DB_MEMORY_POOL 表関数は、SNAPDB_MEMORY_POOL 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションに関する情報を取得することができます。

SNAP_GET_DB_MEMORY_POOL 表関数を SNAP_GET_DB_V95、SNAP_GET_DETAILLOG_V91、 SNAP_GET_HADR、および SNAP_GET_STORAGE_PATHS 表関数とともに使用すると、GET SNAPSHOT FOR ALL DATABASES CLP コマンドに相当する情報が提供されます。

戻される可能性のある情報の完全なリストは、 691 ページの表 177を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、空ストリングを指定します。現在接続されているデータベースと同じインスタンス内のすべてのデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプを持つファイルが存在しない場合、SNAP_GET_DB_MEMORY_POOL 表関数は、現在接続中のデータベースおよびデータベース・パーティション番号のスナップショットを取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_DB_MEMORY_POOL 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT

- SYSADM

例

すべてのデータベースのメモリー・プールおよびそれらの現在のサイズのリストを取得します。

```
SELECT SUBSTR(DB_NAME,1,8) AS DB_NAME, POOL_ID, POOL_CUR_SIZE
   FROM TABLE(SNAPSHOT_GET_DB_MEMORY_POOL
   (CAST(NULL AS VARCHAR(128)), -1)) AS T
```

以下はこの照会の出力例です。

```
DB_NAME  POOL_ID        POOL_CUR_SIZE
-------- -------------- --------------------
TESTDB   UTILITY                  65536
TESTDB   PACKAGE_CACHE           851968
TESTDB   CAT_CACHE                65536
TESTDB   BP                    35913728
TESTDB   BP                      589824
TESTDB   BP                      327680
TESTDB   BP                      196608
TESTDB   BP                      131072
TESTDB   SHARED_SORT              65536
TESTDB   LOCK_MGR              10092544
TESTDB   DATABASE               4980736
TESTDB   OTHER                   196608
SAMPLE   UTILITY                  65536
SAMPLE   PACKAGE_CACHE           655360
SAMPLE   CAT_CACHE               131072
SAMPLE   BP                     4325376
SAMPLE   BP                      589824
SAMPLE   BP                      327680
SAMPLE   BP                      196608
SAMPLE   BP                      131072
SAMPLE   SHARED_SORT                  0
SAMPLE   LOCK_MGR                655360
SAMPLE   DATABASE               4653056
SAMPLE   OTHER                   196608

24 record(s) selected.
```

戻される情報

表 177. SNAPDB_MEMORY_POOL 管理ビューおよび SNAP_GET_DB_MEMORY_POOL 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
DB_NAME	VARCHAR(128)	db_name - データベース名

表 177. SNAPDB_MEMORY_POOL 管理ビューおよび SNAP_GET_DB_MEMORY_POOL 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
POOL_ID	VARCHAR(14)	pool_id - メモリー・プール ID。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • APP_GROUP • APPL_CONTROL • APPLICATION • BP • CAT_CACHE • DATABASE • DFM • FCMBP • IMPORT_POOL • LOCK_MGR • MONITOR • OTHER • PACKAGE_CACHE • QUERY • SHARED_SORT • SORT • STATEMENT • STATISTICS • UTILITY
POOL_SECONDARY_ID	VARCHAR(32)	pool_secondary_id - メモリー・プール 2 次 ID
POOL_CUR_SIZE	BIGINT	pool_cur_size - メモリー・プールの現行サイズ
POOL_WATERMARK	BIGINT	pool_watermark - メモリー・プール水準点
POOL_CONFIG_SIZE	BIGINT	pool_config_size - メモリー・プールの構成済みサイズ
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

SNAPDBM 管理ビューおよび SNAP_GET_DBM_V95 表関数 - dbm 論理グループ・スナップショット情報の検索

SNAPDBM 管理ビューおよび SNAP_GET_DBM_V95 表関数 は、スナップショット・モニターの DB2 データベース・マネージャー (dbm) 論理グループ情報を戻します。

SNAPDBM 管理ビュー

SNAPDBM_MEMORY_POOL、SNAPFCM、SNAPFCM_PART、および SNAPSWITCHES 管理ビューと共に使用すると、SNAPDBM 管理ビューは、GET SNAPSHOT FOR DBM コマンドと同等のデータを提供します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、695 ページの表 178を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPDBM 管理ビューに対する SELECT 特権
- SNAPDBM 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_DBM_V95 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

すべてのデータベース・パーティションにおけるデータベース・マネージャーの状況と接続情報を検索します。

```
SELECT DB2_STATUS, DB2START_TIME, LAST_RESET, LOCAL_CONS, REM_CONS_IN,
   (AGENTS_CREATED_EMPTY_POOL/AGENTS_FROM_POOL) AS AGENT_USAGE,
   DBPARTITIONNUM FROM SYSIBMADM.SNAPDBM ORDER BY DBPARTITIONNUM
```

以下はこの照会の出力例です。

```
DB2_STATUS   DB2START_TIME              LAST_RESET     ...
------------ -------------------------- ----------...- ...
ACTIVE       2006-01-06-14.59.59.059879              - ...
ACTIVE       2006-01-06-14.59.59.097605              - ...
```

```
ACTIVE       2006-01-06-14.59.59.062798             - ...

  3 record(s) selected.                                ...
```

この照会からの出力 (続き)。

```
... LOCAL_CONS     REM_CONS_IN     AGENT_USAGE     DBPARTITIONNUM
... -----------...- -----------...- -----------...- --------------
...              1               1               0              0
...              0               0               0              1
...              0               0               0              2
```

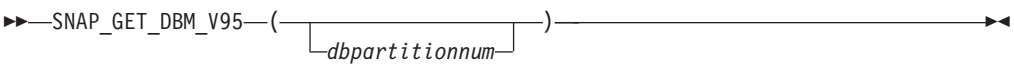
SNAP_GET_DBM_V95 表関数

SNAP_GET_DBM_V95 表関数は SNAPDBM 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションを対象とした情報を検索することができます。

SNAP_GET_DBM_MEMORY_POOL、SNAP_GET_FCM、SNAP_GET_FCM_PART、および SNAP_GET_SWITCHES 表関数と共に使用すると、SNAP_GET_DBM_V95 表関数は、GET SNAPSHOT FOR DBM コマンドと同等のデータを提供します。

戻される可能性のある情報の完全なリストは、695 ページの表 178を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。この入力オプションが使用されない場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbpartitionnum が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプが含まれるファイルが存在しない場合、SNAP_GET_DBM_V95 表関数はメモリーからスナップショットを呼び出します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_DBM_V95 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

データベース・パーティション番号 2 の開始時刻と現行状況を検索します。

```
SELECT DB2START_TIME, DB2_STATUS FROM TABLE(SNAP_GET_DBM_V95(2)) AS T
```

以下はこの照会の出力例です。

```
DB2START_TIME              DB2_STATUS
-------------------------- ------------
2006-01-06-14.59.59.062798 ACTIVE
```

戻される情報

表 178. SNAPDBM 管理ビューおよび SNAP_GET_DBM_V95 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
SORT_HEAP_ALLOCATED	BIGINT	sort_heap_allocated - 割り振られたソート・ヒープの合計
POST_THRESHOLD_SORTS	BIGINT	post_threshold_sorts - ポストしきい値ソート
PIPED_SORTS_REQUESTED	BIGINT	piped_sorts_requested - 要求されたパイプ・ソート数
PIPED_SORTS_ACCEPTED	BIGINT	piped_sorts_accepted - 受け入れられたパイプ・ソート
REM_CONS_IN	BIGINT	rem_cons_in - データベース・マネージャーへのリモート接続
REM_CONS_IN_EXEC	BIGINT	rem_cons_in_exec - データベース・マネージャーで実行中のリモート接続 : モニター・エレメント
LOCAL_CONS	BIGINT	local_cons - ローカル接続
LOCAL_CONS_IN_EXEC	BIGINT	local_cons_in_exec - データベース・マネージャーで実行中のローカル接続 : モニター・エレメント
CON_LOCAL_DBASES	BIGINT	con_local_dbases - 現行接続を使用したローカル・データベース
AGENTS_REGISTERED	BIGINT	agents_registered - 登録済みエージェント

表 178. SNAPDBM 管理ビューおよび SNAP_GET_DBM_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
AGENTS_WAITING_ON_TOKEN	BIGINT	agents_waiting_on_token - トークン待ちエージェント
DB2_STATUS	VARCHAR(12)	db2_status - DB2 インスタンス状況。 このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • ACTIVE • QUIESCE_PEND • QUIESCED
AGENTS_REGISTERED_TOP	BIGINT	agents_registered_top - エージェント最大登録数
AGENTS_WAITING_TOP	BIGINT	agents_waiting_top - エージェント最大待機数
COMM_PRIVATE_MEM	BIGINT	comm_private_mem - コミット済み専用メモリー
IDLE_AGENTS	BIGINT	idle_agents - アイドル・エージェント数
AGENTS_FROM_POOL	BIGINT	agents_from_pool - プールから割り当てられたエージェント
AGENTS_CREATED_EMPTY_POOL	BIGINT	agents_created_empty_pool - エージェント・プールが空のために作成されたエージェント
COORD_AGENTS_TOP	BIGINT	coord_agents_top - コーディネーター・エージェント最大数
MAX_AGENT_OVERFLOWS	BIGINT	max_agent_overflows - 最大エージェント・オーバーフロー回数
AGENTS_STOLEN	BIGINT	agents_stolen - スチールされたエージェント
GW_TOTAL_CONS	BIGINT	gw_total_cons - DB2 Connect の接続試行合計回数
GW_CUR_CONS	BIGINT	gw_cur_cons - DB2 Connect の現在の接続数
GW_CONS_WAIT_HOST	BIGINT	gw_cons_wait_host - ホストの応答を待機している接続の数
GW_CONS_WAIT_CLIENT	BIGINT	gw_cons_wait_client - クライアントの要求送信を待機している接続の数
POST_THRESHOLD_ HASH_JOINS	BIGINT	post_threshold_hash_joins - ハッシュ結合のしきい値
NUM_GW_CONN_SWITCHES	BIGINT	num_gw_conn_switches - 接続切り替え回数
DB2START_TIME	TIMESTAMP	db2start_time - データベース・マネージャー開始タイム・スタンプ
LAST_RESET	TIMESTAMP	last_reset - 最後のリセット・タイム・スタンプ
NUM_NODES_IN_ DB2_INSTANCE	INTEGER	num_nodes_in_db2_instance - データベース・パーティション内のノード数
PRODUCT_NAME	VARCHAR(32)	product_name - 製品名
SERVICE_LEVEL	VARCHAR(18)	service_level - サービス・レベル

表 178. SNAPDBM 管理ビューおよび SNAP_GET_DBM_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
SORT_HEAP_TOP	BIGINT	sort_heap_top - ソート専用ヒープの最高水準点
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。
POST_THRESHOLD_OLAP_FUNCS	BIGINT	ソート・ヒープしきい値を超えた後でソート・ヒープを要求した OLAP 関数の数。 ソート、ハッシュ結合、および OLAP 関数は、ソート・ヒープを使用する操作の例です。通常の条件では、データベース・マネージャーは sortheap 構成パラメーターによって指定された値を使用して、ソート・ヒープを割り振ります。ソート・ヒープに割り振られたメモリーの量がソート・ヒープしきい値 (sheapthres 構成パラメーター) を超える場合、データベース・マネージャーは sortheap 構成パラメーターで指定されたものより小さい値を使用して、その後のソート・ヒープを割り振ります。 ソート・ヒープしきい値に達した後で開始する OLAP 関数は、実行に最適な量のメモリーを受け取ることができない場合があります。

SNAPDBM_MEMORY_POOL 管理ビューおよび SNAP_GET_DBM_MEMORY_POOL 表関数 - データベース・マネージャー・レベルのメモリー使用量情報の検索

SNAPDBM_MEMORY_POOL 管理ビューおよび SNAP_GET_DBM_MEMORY_POOL 表関数は、データベース・マネージャーでのメモリー使用量についての情報を戻します。

SNAPDBM_MEMORY_POOL 管理ビュー

SNAPDBM_MEMORY_POOL 管理ビューを SNAPDBM、SNAPFCM、SNAPFCM_PART、および SNAPSWITCHES 管理ビューとともに使用すると、GET SNAPSHOT FOR DBM コマンドに相当するデータが提供されます。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 700 ページの表 179を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPDBM_MEMORY_POOL 管理ビューに対する SELECT 特権
- SNAPDBM_MEMORY_POOL 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_DBM_MEMORY_POOL 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

接続中のデータベースのデータベース・マネージャーのメモリー・プールおよびそれらの現在のサイズのリストを取得します。

```
SELECT POOL_ID, POOL_CUR_SIZE FROM SNAPDBM_MEMORY_POOL
```

以下はこの照会の出力例です。

```
POOL_ID        POOL_CUR_SIZE
-------------- --------------------
MONITOR                   65536
OTHER                  29622272
FCMBP                  57606144
...
```

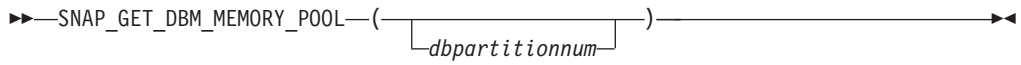
SNAP_GET_DBM_MEMORY_POOL 表関数

SNAP_GET_DBM_MEMORY_POOL 表関数は、SNAPDBM_MEMORY_POOL 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションに関する情報を取得することができます。

SNAP_GET_DBM_MEMORY_POOL 表関数を SNAP_GET_DBM_V95、SNAP_GET_FCM、 SNAP_GET_FCM_PART、および SNAP_GET_SWITCHES 表関数とともに使用すると、GET SNAPSHOT FOR DBM コマンドに相当するデータが提供されます。

戻される可能性のある情報の完全なリストは、 700 ページの表 179を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。この入力オプションが使用されない場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbpartitionnum が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプを持つファイルが存在しない場合、SNAP_GET_DBM_MEMORY_POOL 表関数は、現在接続中のデータベースおよびデータベース・パーティション番号のスナップショットを取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_DBM_MEMORY_POOL 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

接続中のデータベースのデータベース・マネージャーのすべてのデータベース・パーティションのメモリー・プールおよびそれらの現在のサイズのリストを取得します。

```
SELECT POOL_ID, POOL_CUR_SIZE, DBPARTITIONNUM
   FROM TABLE(SYSPROC.SNAP_GET_DBM_MEMORY_POOL())
   AS T ORDER BY DBPARTITIONNUM
```

以下はこの照会の出力例です。

```
POOL_ID        POOL_CUR_SIZE        DBPARTITIONNUM
-------------- -------------------- --------------
MONITOR                     65536              0
OTHER                    29622272              0
```

```
FCMBP                    57606144             0
MONITOR                     65536             1
OTHER                    29425664             1
FCMBP                    57606144             1
MONITOR                     65536             2
OTHER                    29425664             2
FCMBP                    57606144             2
```

戻される情報

表 179. SNAPDBM_MEMORY_POOL 管理ビューおよび SNAP_GET_DBM_MEMORY_POOL 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
POOL_ID	VARCHAR(14)	pool_id - メモリー・プール ID。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • APP_GROUP • APPL_CONTROL • APPLICATION • BP • CAT_CACHE • DATABASE • DFM • FCMBP • IMPORT_POOL • LOCK_MGR • MONITOR • OTHER • PACKAGE_CACHE • QUERY • SHARED_SORT • SORT • STATEMENT • STATISTICS • UTILITY
POOL_CUR_SIZE	BIGINT	pool_cur_size - メモリー・プールの現行サイズ
POOL_WATERMARK	BIGINT	pool_watermark - メモリー・プール水準点
POOL_CONFIG_SIZE	BIGINT	pool_config_size - メモリー・プールの構成済みサイズ
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

SNAPDETAILLOG 管理ビューおよび SNAP_GET_DETAILLOG_V91 表関数 - detail_log 論理データ・グループからのスナップショット情報の検索

SNAPDETAILLOG 管理ビューおよび SNAP_GET_DETAILLOG_V91 表関数は、detail_log 論理データ・グループからのスナップショット情報を戻します。

SNAPDETAILLOG 管理ビュー

この管理ビューを使用すると、現在接続されているデータベースに関するスナップショット情報を detail_log 論理データ・グループから検索できます。

SNAPDETAILLOG 管理ビューを SNAPDB、 SNAPDB_MEMORY_POOL、SNAPHADR、および SNAPSTORAGE_PATHS 管理ビューと併せて使用することにより、GET SNAPSHOT FOR DATABASE on database-alias CLP コマンドと同等の情報を戻します。

スキーマは SYSIBMADM です。

戻される情報の完全なリストは、 704 ページの表 180 を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPDETAILLOG 管理ビューに対する SELECT 特権
- SNAPDETAILLOG 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_DETAILLOG_V91 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

すべてのデータベース・パーティションに関して、現在接続されているデータベースのログ情報を検索します。

```
SELECT SUBSTR(DB_NAME, 1, 8) AS DB_NAME, FIRST_ACTIVE_LOG,
   LAST_ACTIVE_LOG, CURRENT_ACTIVE_LOG, CURRENT_ARCHIVE_LOG,
   DBPARTITIONNUM
   FROM SYSIBMADM.SNAPDETAILLOG ORDER BY DBPARTITIONNUM
```

以下はこの照会の出力例です。

```
DB_NAME  FIRST_ACTIVE_LOG     LAST_ACTIVE_LOG      ...
-------- -------------------- -------------------- ...
TEST                        0                    8 ...
TEST                        0                    8 ...
TEST                        0                    8 ...

  3 record(s) selected.
```

この照会からの出力 (続き)。

```
... CURRENT_ACTIVE_LOG   CURRENT_ARCHIVE_LOG  DBPARTITIONNUM
... -------------------- -------------------- --------------
...                    0                    -              0
...                    0                    -              1
...                    0                    -              2
```

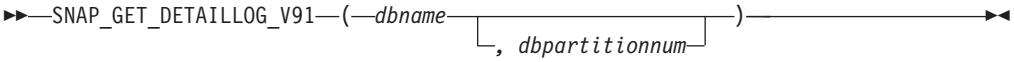
SNAP_GET_DETAILLOG_V91 表関数

SNAP_GET_DETAILLOG_V91 表関数は、SNAPDETAILLOG 管理ビューと同じ情報を戻します。

SNAP_GET_DETAILLOG 表関数を SNAP_GET_DB_V95、SNAP_GET_DB_MEMORY_POOL、 SNAP_GET_HADR、および SNAP_GET_STORAGE_PATHS 表関数と併せて使用することにより、GET SNAPSHOT FOR ALL DATABASES CLP コマンドと同等の情報を戻します。

戻される情報の完全なリストは、 704 ページの表 180 を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、空ストリングを指定します。現在接続されているデータベースと同じインスタンス内のすべてのデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブ

なデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプを持つファイルが存在しない場合には、現在接続されているデータベースのスナップショットとデータベース・パーティション番号を SNAP_GET_DETAILLOG_V91 表関数が取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_DETAILLOG_V91 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

データベース・パーティション 1 に関して、現在接続されているデータベースのログ情報を検索します。

```
SELECT SUBSTR(DB_NAME, 1, 8) AS DB_NAME, FIRST_ACTIVE_LOG,
   LAST_ACTIVE_LOG, CURRENT_ACTIVE_LOG, CURRENT_ARCHIVE_LOG
   FROM TABLE(SNAP_GET_DETAILLOG_V91('', 1)) AS T
```

以下はこの照会の出力例です。

```
DB_NAME  FIRST_ACTIVE_LOG     LAST_ACTIVE_LOG      ...
-------- -------------------- -------------------- ...
TEST                        0                    8 ...

  1 record(s) selected.
```

この照会からの出力 (続き)。

```
... CURRENT_ACTIVE_LOG   CURRENT_ARCHIVE_LOG
... -------------------- --------------------
...                    0                    -
```

SNAPDETAILLOG 管理ビューおよび SNAP_GET_DETAILLOG_V91 表関数のメタデータ

表 180. SNAPDETAILLOG 管理ビューおよび SNAP_GET_DETAILLOG_V91 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
DB_NAME	VARCHAR(128)	db_name - データベース名
FIRST_ACTIVE_LOG	BIGINT	first_active_log - 先頭アクティブ・ログ・ファイル番号
LAST_ACTIVE_LOG	BIGINT	last_active_log - 最終アクティブ・ログ・ファイル番号
CURRENT_ACTIVE_LOG	BIGINT	current_active_log - 現行アクティブ・ログ・ファイル番号
CURRENT_ARCHIVE_LOG	BIGINT	current_archive_log - 現行アーカイブ・ログ・ファイル番号
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

SNAPDYN_SQL 管理ビューおよび SNAP_GET_DYN_SQL_V95 表関数 - dynsql 論理グループのスナップショット情報の検索

『SNAPDYN_SQL 管理ビュー』と 706 ページの『SNAP_GET_DYN_SQL_V95 表関数』は、dynsql 論理データ・グループからのスナップショット情報を戻します。

SNAPDYN_SQL 管理ビュー

この管理ビューを使用すると、現在接続されているデータベースに関する dynsql 論理グループのスナップショット情報を検索できます。

このビューは、GET SNAPSHOT FOR DYNAMIC SQL ON database-alias CLP コマンドと同等の情報を戻します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、708 ページの表 181を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPDYN_SQL 管理ビューに対する SELECT 特権
- SNAPDYN_SQL 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_DYN_SQL_V95 表関数に対する EXECUTE 特権

- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベースのすべてのデータベース・パーティションで実行される動的 SQL のリスト (読み取られる行の番号順に並んでいるもの) を検索します。

```
SELECT PREP_TIME_WORST, NUM_COMPILATIONS, SUBSTR(STMT_TEXT, 1, 60)
   AS STMT_TEXT, DBPARTITIONNUM
   FROM SYSIBMADM.SNAPDYN_SQL ORDER BY ROWS_READ
```

以下はこの照会の出力例です。

```
PREP_TIME_WORST      NUM_COMPILATIONS     ...
-------------------- -------------------- ...
                  98                    1 ...
                   9                    1 ...
                   0                    0 ...
                   0                    1 ...
                   0                    1 ...
                   0                    1 ...
                   0                    1 ...
                   0                    1 ...
                  40                    1 ...

  9 record(s) selected.
```

この照会からの出力 (続き)。

```
... STMT_TEXT                                                  ...
... ------------------------------------------------------------ ...
... select prep_time_worst, num_compilations, substr(stmt_text,  ...
... select * from dbuser.employee                              ...
... SET CURRENT LOCALE LC_CTYPE = 'en_US'                      ...
... select prep_time_worst, num_compilations, substr(stmt_text,  ...
... select prep_time_worst, num_compilations, substr(stmt_text,  ...
... select * from dbuser.employee                              ...
... insert into dbuser.employee values(1)                      ...
... select * from dbuser.employee                              ...
... insert into dbuser.employee values(1)                      ...
```

この照会からの出力 (続き)。

```
... DBPARTITIONNUM
... -------------
...               0
...               0
...               0
...               2
...               1
...               2
...               2
...               1
...               0
```

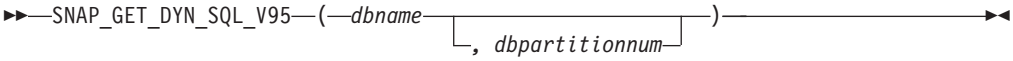
SNAP_GET_DYN_SQL_V95 表関数

SNAP_GET_DYN_SQL_V91 表関数は SNAPDYN_SQL 管理ビューと同じ情報を戻します。ただし、SNAP_GET_DYN_SQL_V95 表関数の場合は、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションについて、特定のデータベースの情報を取り出すことができます。

この表関数は、GET SNAPSHOT FOR DYNAMIC SQL ON database-alias CLP コマンドと同等の情報を戻します。

戻される可能性のある情報の完全なリストは、 708 ページの表 181を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL または空ストリングを指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプを持つファイルが存在しない場合には、現在接続されているデータベースのスナップショットとデータベース・パーティション番号を SNAP_GET_DYN_SQL_V95 表関数が取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_DYN_SQL_V95 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベースの現在接続されているデータベース・パーティションで実行される動的 SQL のリスト (読み取られる行の番号順に並んでいるもの) を検索します。

```
SELECT PREP_TIME_WORST, NUM_COMPILATIONS, SUBSTR(STMT_TEXT, 1, 60)
   AS STMT_TEXT FROM TABLE(SNAP_GET_DYN_SQL_V95('',-1)) as T
    ORDER BY ROWS_READ
```

以下はこの照会の出力例です。

```
PREP_TIME_WORST      ...
-------------------- ...
                   0 ...
                   3 ...
                     ...
                   4 ...
                     ...
                   4 ...
                     ...
                   4 ...
                     ...
                   3 ...
                     ...
                   4 ...
                     ...
```

この照会からの出力 (続き)。

```
... NUM_COMPILATIONS     STMT_TEXT
... -------------------- ----------------------------------------...-
...                    0 SET CURRENT LOCALE LC_CTYPE = 'en_US'
...                    1 select rows_read, rows_written,
...                          substr(stmt_text, 1, 40) as
...                    1 select * from table
...                          (snap_get_dyn_sqlv9('',-1)) as t
...                    1 select * from table
...                          (snap_getdetaillog9('',-1)) as t
...                    1 select * from table
...                          (snap_get_hadr('',-1)) as t
...                    1 select prep_time_worst, num_compilations,
...                          substr(stmt_text,
...                    1 select prep_time_worst, num_compilations,
...                          substr(stmt_text,
```

ワークロードの実行後に、ユーザーは次の照会を表関数とともに使用できます。

```
SELECT STATS_FABRICATE_TIME,SYNC_RUNSTATS_TIME
   FROM TABLE (SNAP_GET_DYN_SQL_V95('mytestdb', -1))
   AS SNAPDB

STATS_FABRICATE_TIME   SYNC_RUNSTATS_TIME
---------------------- ------------------
                     2                 12
                     1                 30
```

この表関数に基づくビューの場合:

```
SELECT STATS_FABRICATE_TIME,SYNC_RUNSTATS_TIME
   FROM SYSIBMADM.SNAPDYN_SQL

STATS_FABRICATE_TIME   SYNC_RUNSTATS_TIME
---------------------- ------------------
                     5                 10
                     3                 20
  2 record(s) selected.
```

戻される情報

表 181. SNAPDYN_SQL 管理ビューおよび SNAP_GET_DYN_SQL_V95 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
NUM_EXECUTIONS	BIGINT	num_executions - ステートメント実行回数
NUM_COMPILATIONS	BIGINT	num_compilations - ステートメント・コンパイル数
PREP_TIME_WORST	BIGINT	prep_time_worst - ステートメント最長準備時間
PREP_TIME_BEST	BIGINT	prep_time_best - ステートメント最短準備時間
INT_ROWS_DELETED	BIGINT	int_rows_deleted - 削除された内部行数
INT_ROWS_INSERTED	BIGINT	int_rows_inserted - 挿入された内部行数
INT_ROWS_UPDATED	BIGINT	int_rows_updated - 更新された内部行数
ROWS_READ	BIGINT	rows_read - 読み取り行数
ROWS_WRITTEN	BIGINT	rows_written - 書き込み行数
STMT_SORTS	BIGINT	stmt_sorts - ステートメント・ソート回数
SORT_OVERFLOWS	BIGINT	sort_overflows - ソート・オーバーフロー
TOTAL_SORT_TIME	BIGINT	total_sort_time - ソート時間合計
POOL_DATA_L_READS	BIGINT	pool_data_l_reads - バッファー・プール・データの論理読み取り
POOL_DATA_P_READS	BIGINT	pool_data_p_reads - バッファー・プール・データの物理読み取り
POOL_TEMP_DATA_L_READS	BIGINT	pool_temp_data_l_reads - バッファー・プール一時データの論理読み取り
POOL_TEMP_DATA_P_READS	BIGINT	pool_temp_data_p_reads - バッファー・プール一時データの物理読み取り
POOL_INDEX_L_READS	BIGINT	pool_index_l_reads - バッファー・プール索引の論理読み取り
POOL_INDEX_P_READS	BIGINT	pool_index_p_reads - バッファー・プール索引の物理読み取り

表 181. SNAPDYN_SQL 管理ビューおよび SNAP_GET_DYN_SQL_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
POOL_TEMP_INDEX_L_READS	BIGINT	pool_temp_index_l_reads - バッファー・プール一時索引の論理読み取り
POOL_TEMP_INDEX_P_READS	BIGINT	pool_temp_index_p_reads - バッファー・プール一時索引の物理読み取り
POOL_XDA_L_READS	BIGINT	pool_xda_l_reads - バッファー・プール XDA データの論理読み取り
POOL_XDA_P_READS	BIGINT	pool_xda_p_reads - バッファー・プール XDA データの物理読み取り
POOL_TEMP_XDA_L_READS	BIGINT	pool_temp_xda_l_reads - バッファー・プール一時 XDA データの論理読み取り
POOL_TEMP_XDA_P_READS	BIGINT	pool_temp_xda_p_reads - バッファー・プール一時 XDA データの物理読み取り : モニター・エレメント
TOTAL_EXEC_TIME	BIGINT	total_exec_time - ステートメント実行の経過時間 (秒単位)*
TOTAL_EXEC_TIME_MS	BIGINT	total_exec_time - ステートメント実行の経過時間 (小数部、マイクロ秒単位)*
TOTAL_USR_CPU_TIME	BIGINT	total_usr_cpu_time - ステートメントのユーザー CPU の合計 (秒単位)*
TOTAL_USR_CPU_TIME_MS	BIGINT	total_usr_cpu_time - ステートメントのユーザー CPU の合計 (小数部、マイクロ秒単位)*
TOTAL_SYS_CPU_TIME	BIGINT	total_sys_cpu_time - ステートメントのシステム CPU の合計 (秒単位)*
TOTAL_SYS_CPU_TIME_MS	BIGINT	total_sys_cpu_time - ステートメントのシステム CPU の合計 (小数部、マイクロ秒単位)*
STMT_TEXT	CLOB(2 M)	stmt_text - SQL 動的ステートメント・テキスト
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。
STATS_FABRICATE_TIME	BIGINT	動的ステートメントの照会コンパイル中に表または索引のスキャンを実行しないで、システムが必要とされる統計を作成するのに費やす合計時間 (ミリ秒)。
SYNC_RUNSTATS_TIME	BIGINT	動的ステートメントの照会コンパイル中に同期 statistics-collect アクティビティーに費やされる合計時間 (ミリ秒)。
* この列の元になるモニター・エレメントの合計消費時間を計算するには、このモニター・エレメントに関する _S で終わっている列で報告されている整数秒と、このモニター・エレメントに関する _MS で終わっている列に報告されている小数秒を、次の式を使用して加算しなければなりません: (*monitor-element-name*_S × 1,000,000 + *monitor-element-name*_MS) ÷ 1,000,000. 例えば、(ELAPSED_EXEC_TIME_S × 1,000,000 + ELAPSED_EXEC_TIME_MS) ÷ 1,000,000。		

SNAPFCM 管理ビューおよび SNAP_GET_FCM 表関数 - fcm 論理データ・グループ・スナップショット情報の検索

SNAPFCM 管理ビューおよび SNAP_GET_FCM 表関数は、データベース・マネージャー・スナップショットから高速コミュニケーション・マネージャーについて、特に fcm 論理データ・グループについての情報を戻します。

SNAPFCM 管理ビュー

SNAPDBM、SNAPDBM_MEMORY_POOL、SNAPFCM_PART、および SNAPSWITCHES 管理ビューと共に使用すると、SNAPFCM 管理ビューは、GET SNAPSHOT FOR DBM コマンドと同等のデータを提供します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 712 ページの表 182を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPFCM 管理ビューに対する SELECT 特権
- SNAPFCM 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_FCM 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

すべてのデータベース・パーティションにおける高速コミュニケーション・マネージャーのメッセージ・バッファーについての情報を検索します。

```
SELECT BUFF_FREE, BUFF_FREE_BOTTOM, DBPARTITIONNUM
    FROM SYSIBMADM.SNAPFCM ORDER BY DBPARTITIONNUM
```

以下はこの照会の出力例です。

```
BUFF_FREE        BUFF_FREE_BOTTOM     DBPARTITIONNUM
---------...---- -------------------- --------------
            5120                 5100              0
            5120                 5100              1
            5120                 5100              2
```

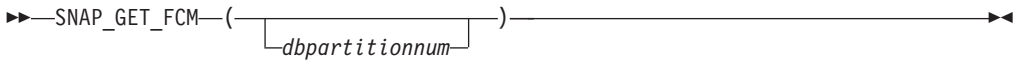
SNAP_GET_FCM 表関数

SNAP_GET_FCM 表関数は SNAPFCM 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションを対象とした情報を検索することができます。

SNAP_GET_DBM_V95、 SNAP_GET_DBM_MEMORY_POOL、SNAP_GET_FCM_PART、および SNAP_GET_SWITCHES 表関数と共に使用すると、SNAP_GET_FCM 表関数は、GET SNAPSHOT FOR DBM コマンドと同等のデータを提供します。

戻される可能性のある情報の完全なリストは、 712 ページの表 182を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。この入力オプションが使用されない場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbpartitionnum が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプが含まれるファイルが存在しない場合、SNAP_GET_FCM 表関数は、現在接続されているデータベースのスナップショットとデータベース・パーティション番号を取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_FCM 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL

- SYSMAINT
- SYSADM

例

データベース・パーティション 1 における高速コミュニケーション・マネージャーのメッセージ・バッファーについての情報を検索します。

```
SELECT BUFF_FREE, BUFF_FREE_BOTTOM, DBPARTITIONNUM
   FROM TABLE(SYSPROC.SNAP_GET_FCM( 1 )) AS T
```

以下はこの照会の出力例です。

```
BUFF_FREE            BUFF_FREE_BOTTOM     DBPARTITIONNUM
-------------------- -------------------- --------------
                5120                 5100              1
```

戻される情報

表 182. SNAPFCM 管理ビューおよび SNAP_GET_FCM 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
BUFF_FREE	BIGINT	buff_free - 現在空いている FCM バッファー
BUFF_FREE_BOTTOM	BIGINT	buff_free_bottom - 空き FCM バッファーの最小数
CH_FREE	BIGINT	ch_free - 現在空いているチャネル
CH_FREE_BOTTOM	BIGINT	ch_free_bottom - 空いているチャネルの最小
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

SNAPFCM_PART 管理ビューおよび SNAP_GET_FCM_PART 表関数 - fcm_node 論理データ・グループ・スナップショット情報の検索

SNAPFCM_PART 管理ビューおよび SNAP_GET_FCM_PART 表関数は、データベース・マネージャー・スナップショットから、特に fcm_node 論理データ・グループの、高速コミュニケーション・マネージャー情報を戻します。

SNAPFCM_PART 管理ビュー

SNAPDBM、SNAPDBM_MEMORY_POOL、SNAPFCM、および SNAPSWITCHES 管理ビューと共に使用すると、SNAPFCM_PART 管理ビューは、GET SNAPSHOT FOR DBM コマンドと同等のデータを提供します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、715 ページの表 183を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPFCM_PART 管理ビューに対する SELECT 特権
- SNAPFCM_PART 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_FCM_PART 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

高速コミュニケーション・マネージャーのバッファー送受信情報を検索します。

```
SELECT CONNECTION_STATUS, TOTAL_BUFFERS_SENT, TOTAL_BUFFERS_RECEIVED
   FROM SYSIBMADM.SNAPFCM_PART WHERE DBPARTITIONNUM = 0
```

以下はこの照会の出力例です。

```
CONNECTION_STATUS    TOTAL_BUFFERS_SENT   TOTAL_BUFFERS_RCVD
-------------------- -------------------- --------------------
INACTIVE                                2                    1

  1 record(s) selected.
```

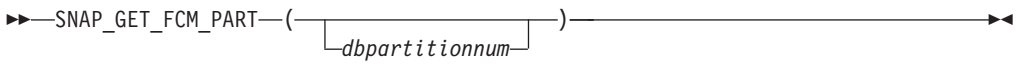
SNAP_GET_FCM_PART 表関数

SNAP_GET_FCM_PART 表関数は SNAPFCM_PART 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションを対象とした情報を検索することができます。

SNAP_GET_DBM_V95、 SNAP_GET_DBM_MEMORY_POOL、 SNAP_GET_FCM、および SNAP_GET_SWITCHES 表関数と共に使用すると、SNAP_GET_FCM_PART 表関数は、GET SNAPSHOT FOR DBM コマンドと同等のデータを提供します。

戻される可能性のある情報の完全なリストは、 715 ページの表 183を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のパーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。この入力オプションが使用されない場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbpartitionnum が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプが含まれるファイルが存在しない場合、SNAP_GET_FCM_PART 表関数は、現在接続されているデータベースのスナップショットとデータベース・パーティション番号を取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_FCM_PART 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

すべてのデータベース・パーティションにおける高速コミュニケーション・マネージャーのバッファー送受信情報を検索します。

```
SELECT FCM_DBPARTITIONNUM, TOTAL_BUFFERS_SENT, TOTAL_BUFFERS_RCVD,
   DBPARTITIONNUM FROM TABLE(SNAP_GET_FCM_PART()) AS T
   ORDER BY DBPARTITIONNUM
```

以下はこの照会の出力例です。

```
FCM_DBPARTITIONNUM TOTAL_BUFFERS_SENT   TOTAL_BUFFERS_RCVD   DBPARTITIONNUM
------------------ -------------------- -------------------- --------------
                 0                  305                  305              0
                 1                 5647                 1664              0
                 2                 5661                 1688              0
                 0                   19                   19              1
                 1                  305                  301              1
                 2                 1688                 5661              1
                 0                 1664                 5647              2
                 1                   10                   10              2
                 2                  301                  305              2
```

戻される情報

表 183. SNAPFCM_PART 管理ビューおよび SNAP_GET_FCM_PART 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
CONNECTION_STATUS	VARCHAR(10)	connection_status - 接続状況。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • INACTIVE • ACTIVE • CONGESTED
TOTAL_BUFFERS_SENT	BIGINT	total_buffers_sent - 送信された FCM バッファーの合計
TOTAL_BUFFERS_RCVD	BIGINT	total_buffers_rcvd - 受信された FCM バッファーの合計
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。
FCM_DBPARTITIONNUM	SMALLINT	データの送信先または受信元のデータベース・パーティション番号 (TOTAL_BUFFERS_SENT および TOTAL_BUFFERS_RCVD 列ごとに)。

SNAPHADR 管理ビューおよび SNAP_GET_HADR 表関数 - hadr 論理データ・グループのスナップショット情報の検索

SNAPHADR 管理ビューおよび SNAP_GET_HADR 表関数は、データベース・スナップショットから、特に hadr 論理データ・グループの高可用性災害時リカバリー情報を戻します。

SNAPHADR 管理ビュー

この管理ビューを使用して、現在接続中のデータベースに関する hadr 論理データ・グループのスナップショット情報を取得することができます。データベースが 1 次またはスタンバイ高可用性災害時リカバリー (HADR) データベースの場合にのみ、このビューによってデータが戻されます。

SNAPHADR 管理ビューを SNAPDB、 SNAPDB_MEMORY_POOL、SNAPDETAILLOG、および SNAPSTORAGE_PATHS 管理ビューとともに使用すると、GET SNAPSHOT FOR DATABASE ON database-alias CLP コマンドに相当する情報が提供されます。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 718 ページの表 184を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPHADR 管理ビューに対する SELECT 特権
- SNAPHADR 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_HADR 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

1 次 HADR データベース上の HADR に関する構成および状況情報を取得します。

```
SELECT SUBSTR(DB_NAME, 1, 8) AS DBNAME, HADR_ROLE, HADR_STATE,
   HADR_SYNCMODE, HADR_CONNECT_STATUS
   FROM SYSIBMADM.SNAPHADR
```

以下はこの照会の出力例です。

```
DBNAME   HADR_ROLE HADR_STATE     HADR_SYNCMODE HADR_CONNECT_STATUS
-------- --------- -------------- ------------- ------------------
SAMPLE   PRIMARY   PEER           SYNC          CONNECTED

  1 record(s) selected.
```

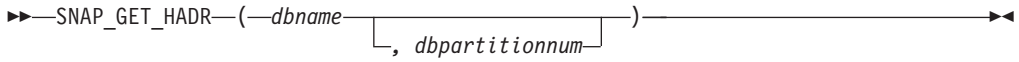
SNAP_GET_HADR 表関数

SNAP_GET_HADR 表関数は、SNAPHADR 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションに関する情報を取得することができます。

SNAP_GET_HADR 表関数を SNAP_GET_DB_V95、SNAP_GET_DB_MEMORY_POOL、 SNAP_GET_DETAILLOG_V91、および SNAP_GET_STORAGE_PATHS 表関数とともに使用すると、GET SNAPSHOT FOR ALL DATABASES CLP コマンドに相当する情報が提供されます。

戻される可能性のある情報の完全なリストは、 718 ページの表 184を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、空ストリングを指定します。現在接続されているデータベースと同じインスタンス内のすべてのデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が *NULL* に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプを持つファイルが存在しない場合、SNAP_GET_HADR 表関数は、現在接続中のデータベースおよびデータベース・パーティション番号のスナップショットを取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_HADR 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT

- SYSADM

例

すべてのデータベースの HADR に関する構成および状況情報を取得します。

```
SELECT SUBSTR(DB_NAME, 1, 8) AS DBNAME, HADR_ROLE, HADR_STATE,
   HADR_SYNCMODE, HADR_CONNECT_STATUS
   FROM TABLE (SNAP_GET_HADR (CAST (NULL as VARCHAR(128)), 0)) as T
```

以下はこの照会の出力例です。

```
DBNAME   HADR_ROLE HADR_STATE     HADR_SYNCMODE HADR_CONNECT_STATUS
-------- --------- -------------- ------------- ------------------
SAMPLE   PRIMARY   PEER           SYNC          CONNECTED
TESTDB   PRIMARY   DISCONNECTED   NEARSYNC      DISCONNECTED

2 record(s) selected.
```

戻される情報

表 184. SNAPHADR 管理ビューおよび SNAP_GET_HADR 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
DB_NAME	VARCHAR(128)	db_name - データベース名
HADR_ROLE	VARCHAR(10)	hadr_role - HADR のロール。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • PRIMARY • STANDARD • STANDBY
HADR_STATE	VARCHAR(14)	hadr_state - HADR の状態。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • DISCONNECTED • LOCAL_CATCHUP • PEER • REM_CATCH_PEN • REM_CATCHUP
HADR_SYNCMODE	VARCHAR(10)	hadr_syncmode - HADR 同期モード。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • ASYNC • NEARSYNC • SYNC

表 184. SNAPHADR 管理ビューおよび SNAP_GET_HADR 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
HADR_CONNECT_STATUS	VARCHAR(12)	hadr_connect_status - HADR 接続状況。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • CONGESTED • CONNECTED • DISCONNECTED
HADR_CONNECT_TIME	TIMESTAMP	hadr_connect_time - HADR 接続時刻
HADR_HEARTBEAT	INTEGER	hadr_heartbeat - HADR ハートビート
HADR_LOCAL_HOST	VARCHAR(255)	hadr_local_host - HADR ローカル・ホスト
HADR_LOCAL_SERVICE	VARCHAR(40)	hadr_local_service - HADR ローカル・サービス
HADR_REMOTE_HOST	VARCHAR(255)	hadr_remote_host - HADR リモート・ホスト
HADR_REMOTE_SERVICE	VARCHAR(40)	hadr_remote_service - HADR リモート・サービス
HADR_REMOTE_INSTANCE	VARCHAR(128)	hadr_remote_instance - HADR リモート・インスタンス
HADR_TIMEOUT	BIGINT	hadr_timeout - HADR タイムアウト
HADR_PRIMARY_LOG_FILE	VARCHAR(255)	hadr_primary_log_file - HADR 1 次ログ・ファイル
HADR_PRIMARY_LOG_PAGE	BIGINT	hadr_primary_log_page - HADR 1 次ログ・ページ
HADR_PRIMARY_LOG_LSN	BIGINT	hadr_primary_log_lsn - HADR 1 次ログ LSN
HADR_STANDBY_LOG_FILE	VARCHAR(255)	hadr_standby_log_file - HADR スタンバイ・ログ・ファイル
HADR_STANDBY_LOG_PAGE	BIGINT	hadr_standby_log_page - HADR スタンバイ・ログ・ページ
HADR_STANDBY_LOG_LSN	BIGINT	hadr_standby_log_lsn - HADR スタンバイ・ログ LSN
HADR_LOG_GAP	BIGINT	hadr_log_gap - HADR ログ・ギャップ
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

SNAPLOCK 管理ビューおよび SNAP_GET_LOCK 表関数 - lock 論理データ・グループのスナップショット情報の検索

注: この管理ビューおよび表関数は非推奨になり、以下のものに置き換えられました。すなわち、 460 ページの『MON_GET_APPL_LOCKWAIT - アプリケーションが待機しているロックについての情報の収集』、 490 ページの『MON_GET_LOCKS - 現在接続されているデータベース内のすべてのロックのリスト』、および 425 ページの『MON_FORMAT_LOCK_NAME - 内部ロック名のフォーマット設定と詳細の出力』。

SNAPLOCK 管理ビューおよび SNAP_GET_LOCK 表関数は、特に lock 論理データ・グループの、ロック・スナップショット情報を戻します。

SNAPLOCK 管理ビュー

この管理ビューでは、現在接続されているデータベースの lock 論理データ・グループ・スナップショット情報を検索できます。

SNAPLOCKWAIT 管理ビューと共に使用すると、SNAPLOCK 管理ビューは、GET SNAPSHOT FOR LOCKS ON database-alias CLP コマンドと同等の情報を提供します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 722 ページの表 185を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPLOCK 管理ビューに対する SELECT 特権
- SNAPLOCK 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_LOCK 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベースのデータベース・パーティション 0 のロック情報を検索します。

```
SELECT AGENT_ID, LOCK_OBJECT_TYPE, LOCK_MODE, LOCK_STATUS
   FROM SYSIBMADM.SNAPLOCK WHERE DBPARTITIONNUM = 0
```

以下はこの照会の出力例です。

```
AGENT_ID             LOCK_OBJECT_TYPE LOCK_MODE LOCK_STATUS
-------------------- ---------------- --------- -----------
                   7 TABLE            IX        GRNT

  1 record(s) selected.
```

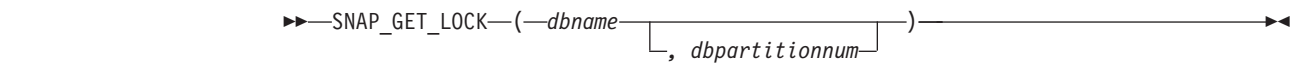
SNAP_GET_LOCK 表関数

SNAP_GET_LOCK 表関数は SNAPLOCK 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションのいずれかの特定のデータベースを対象とした情報を検索することができます。

SNAP_GET_LOCKWAIT 表関数と共に使用すると、SNAP_GET_LOCK 表関数は、GET SNAPSHOT FOR LOCKS ON database-alias CLP コマンドと同等の情報を提供します。

戻される可能性のある情報の完全なリストは、722 ページの表 185を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL 値または空ストリングを指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプが含まれるファイルが存在しない場合、SNAP_GET_LOCK 表関数は、現在接続されているデータベースのスナップショットとデータベース・パーティション番号を取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_LOCK 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベースの現行データベース・パーティションのロック情報を検索します。

```
SELECT AGENT_ID, LOCK_OBJECT_TYPE, LOCK_MODE, LOCK_STATUS
   FROM TABLE(SNAP_GET_LOCK('',-1)) as T
```

以下はこの照会の出力例です。

```
AGENT_ID       LOCK_OBJECT_TYPE   LOCK_MODE  LOCK_STATUS
--------...--- ------------------ ---------- -----------
           680 INTERNALV_LOCK     S          GRNT
           680 INTERNALP_LOCK     S          GRNT

2 record(s) selected.
```

戻される情報

表 185. SNAPLOCK 管理ビューおよび SNAP_GET_LOCK 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
AGENT_ID	BIGINT	agent_id - アプリケーション・ハンドル (エージェント ID)
TAB_FILE_ID	BIGINT	table_file_id - 表ファイル ID

表 185. SNAPLOCK 管理ビューおよび SNAP_GET_LOCK 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
LOCK_OBJECT_TYPE	VARCHAR(18)	lock_object_type - 待機中のロック対象タイプ。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • AUTORESIZE_LOCK • AUTOSTORAGE_LOCK • BLOCK_LOCK • EOT_LOCK • INPLACE_REORG_LOCK • INTERNAL_LOCK • INTERNALB_LOCK • INTERNALC_LOCK • INTERNALJ_LOCK • INTERNALL_LOCK • INTERNALO_LOCK • INTERNALQ_LOCK • INTERNALP_LOCK • INTERNALS_LOCK • INTERNALT_LOCK • INTERNALV_LOCK • KEYVALUE_LOCK • ROW_LOCK • SYSBOOT_LOCK • TABLE_LOCK • TABLE_PART_LOCK • TABLESPACE_LOCK • XML_PATH_LOCK

表 185. SNAPLOCK 管理ビューおよび SNAP_GET_LOCK 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
LOCK_MODE	VARCHAR(10)	lock_mode - ロック・モード。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • IN • IS • IX • NON (ロックなしの場合) • NS • NW • S • SIX • U • X • Z
LOCK_STATUS	VARCHAR(10)	lock_status - ロック状況。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • CONV • GRNT
LOCK_ESCALATION	SMALLINT	lock_escalation - ロック・エスカレーション
TABNAME	VARCHAR(128)	table_name - 表名
TABSCHEMA	VARCHAR(128)	table_schema - 表スキーマ名
TBSP_NAME	VARCHAR(128)	tablespace_name - 表スペース名

表 185. SNAPLOCK 管理ビューおよび SNAP_GET_LOCK 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
LOCK_ATTRIBUTES	VARCHAR(128)	lock_attributes - ロック属性。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。ロックがない場合、テキスト ID は NONE となり、それ以外の場合、以下のいずれかの組み合わせを '+' 記号で区切ったものとなります。 • ALLOW_NEW • DELETE_IN_BLOCK • ESCALATED • INSERT • NEW_REQUEST • RR • RR_IN_BLOCK • UPDATE_DELETE • WAIT_FOR_AVAIL
LOCK_COUNT	BIGINT	lock_count - ロック・カウント
LOCK_CURRENT_MODE	VARCHAR(10)	lock_current_mode - 移行前の元のロック・モード。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • IN • IS • IX • NON (ロックなしの場合) • NS • NW • S • SIX • U • X • Z
LOCK_HOLD_COUNT	BIGINT	lock_hold_count - ロック保留カウント
LOCK_NAME	VARCHAR(32)	lock_name - ロック名
LOCK_RELEASE_FLAGS	BIGINT	lock_release_flags - ロック保留解除フラグ

表 185. SNAPLOCK 管理ビューおよび SNAP_GET_LOCK 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
DATA_PARTITION_ID	INTEGER	data_partition_id - データ・パーティション ID。非パーティション表では、このエレメントは NULL です。
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

SNAPLOCKWAIT 管理ビューおよび SNAP_GET_LOCKWAIT 表関数 - lockwait 論理データ・グループのスナップショット情報の検索

注: この管理ビューおよび表関数は非推奨になり、以下のものに置き換えられました。すなわち、 556 ページの『MON_LOCKWAITS 管理ビュー - ロックの取得を待機しているアプリケーションに関するメトリックの取得』、および 460 ページの『MON_GET_APPL_LOCKWAIT - アプリケーションが待機しているロックについての情報の収集』、 490 ページの『MON_GET_LOCKS - 現在接続されているデータベース内のすべてのロックのリスト』、 425 ページの『MON_FORMAT_LOCK_NAME - 内部ロック名のフォーマット設定と詳細の出力』。

SNAPLOCKWAIT 管理ビューおよび SNAP_GET_LOCKWAIT 表関数は、特に lockwait 論理データ・グループの、ロック待機スナップショット情報を戻します。

SNAPLOCKWAIT 管理ビュー

この管理ビューでは、現在接続されているデータベースの lockwait 論理データ・グループ・スナップショット情報を検索できます。

SNAPLOCK 管理ビューと共に使用すると、SNAPLOCKWAIT 管理ビューは、GET SNAPSHOT FOR LOCKS ON database-alias CLP コマンドと同等の情報を提供します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 729 ページの表 186を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPLOCKWAIT 管理ビューに対する SELECT 特権
- SNAPLOCKWAIT 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_LOCKWAIT 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベースのデータベース・パーティション 0 のロック待機情報を検索します。

```
SELECT AGENT_ID, LOCK_MODE, LOCK_OBJECT_TYPE, AGENT_ID_HOLDING_LK,
   LOCK_MODE_REQUESTED FROM SYSIBMADM.SNAPLOCKWAIT
   WHERE DBPARTITIONNUM = 0
```

以下はこの照会の出力例です。

```
AGENT_ID     LOCK_MODE LOCK_OBJECT_TYPE ...
---------...- --------- ---------------- ...
           7 IX        TABLE            ...

  1 record(s) selected.
```

この照会からの出力 (続き)。

```
... AGENT_ID_HOLDING_LK  LOCK_MODE_REQUESTED
... -------------------- --------------------
...                   12 IS
```

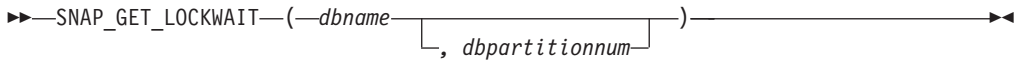
SNAP_GET_LOCKWAIT 表関数

SNAP_GET_LOCKWAIT 表関数は SNAPLOCKWAIT 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションのいずれかの特定のデータベースを対象とした情報を検索することができます。

SNAP_GET_LOCK 表関数と共に使用すると、SNAP_GET_LOCKWAIT 表関数は、GET SNAPSHOT FOR LOCKS ON database-alias CLP コマンドと同等の情報を提供します。

戻される可能性のある情報の完全なリストは、 729 ページの表 186を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL 値または空ストリングを指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプが含まれるファイルが存在しない場合、SNAP_GET_LOCKWAIT 表関数は、現在接続されているデータベースのスナップショットとデータベース・パーティション番号を取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_LOCKWAIT 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベースの現行データベース・パーティションのロック待機情報を検索します。

```
SELECT AGENT_ID, LOCK_MODE, LOCK_OBJECT_TYPE, AGENT_ID_HOLDING_LK,
   LOCK_MODE_REQUESTED FROM TABLE(SNAP_GET_LOCKWAIT('',-1)) AS T
```

以下はこの照会の出力例です。

```
AGENT_ID      LOCK_MODE  LOCK_OBJECT_TYPE   ...
---------...-- ---------- ------------------ ...
           12 X          ROW_LOCK           ...

1 record(s) selected.
```

この照会からの出力 (続き)。

```
... AGENT_ID_HOLDING_LK  LOCK_MODE_REQUESTED
... -------------------- ------------------
...                    7 X
```

使用上の注意

ロック待機情報を表示するには、まずデータベース・マネージャー構成でデフォルトの LOCK モニター・スイッチをオンにする必要があります。変更を即時に有効にするには、CLP を使用してインスタンスに明示的にアタッチし、次いで以下の CLP コマンドを発行します。

```
UPDATE DATABASE MANAGER CONFIGURATION CLP USING DFT_MON_LOCK ON
```

デフォルトの設定値も、ADMIN_CMD ストアード・プロシージャーによりオンにすることができます。以下に例を示します。

```
CALL SYSPROC.ADMIN_CMD('update dbm cfg using DFT_MON_LOCK ON')
```

ADMIN_CMD ストアード・プロシージャーを使用する場合、または事前にインスタンスにアタッチせずに CLP コマンドを使用する場合、インスタンスをリサイクルしなければ変更は有効になりません。

戻される情報

表 186. SNAPLOCKWAIT 管理ビューおよび SNAP_GET_LOCKWAIT 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
AGENT_ID	BIGINT	agent_id - アプリケーション・ハンドル (エージェント ID)
SUBSECTION_NUMBER	BIGINT	ss_number - サブセクション番号

表 186. SNAPLOCKWAIT 管理ビューおよび SNAP_GET_LOCKWAIT 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
LOCK_MODE	VARCHAR(10)	lock_mode - ロック・モード。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • IN • IS • IX • NON (ロックなしの場合) • NS • NW • S • SIX • U • X • Z

表 186. SNAPLOCKWAIT 管理ビューおよび SNAP_GET_LOCKWAIT 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
LOCK_OBJECT_TYPE	VARCHAR(18)	lock_object_type - 待機中のロック対象タイプ。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • AUTORESIZE_LOCK • AUTOSTORAGE_LOCK • BLOCK_LOCK • EOT_LOCK • INPLACE_REORG_LOCK • INTERNAL_LOCK • INTERNALB_LOCK • INTERNALC_LOCK • INTERNALJ_LOCK • INTERNALL_LOCK • INTERNALO_LOCK • INTERNALQ_LOCK • INTERNALP_LOCK • INTERNALS_LOCK • INTERNALT_LOCK • INTERNALV_LOCK • KEYVALUE_LOCK • ROW_LOCK • SYSBOOT_LOCK • TABLE_LOCK • TABLE_PART_LOCK • TABLESPACE_LOCK • XML_PATH_LOCK
AGENT_ID_HOLDING_LK	BIGINT	agent_id_holding_lock - ロックを保持しているエージェント ID
LOCK_WAIT_START_TIME	TIMESTAMP	lock_wait_start_time - ロック待機開始タイム・スタンプ

表 186. SNAPLOCKWAIT 管理ビューおよび SNAP_GET_LOCKWAIT 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
LOCK_MODE_REQUESTED	VARCHAR(10)	lock_mode_requested - 要求されているロック・モード。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • IN • IS • IX • NON (ロックなしの場合) • NS • NW • S • SIX • U • X • Z
LOCK_ESCALATION	SMALLINT	lock_escalation - ロック・エスカレーション
TABNAME	VARCHAR(128)	table_name - 表名
TABSCHEMA	VARCHAR(128)	table_schema - 表スキーマ名
TBSP_NAME	VARCHAR(128)	tablespace_name - 表スペース名
APPL_ID_HOLDING_LK	VARCHAR(128)	appl_id_holding_lk - ロックを保持しているアプリケーション ID
LOCK_ATTRIBUTES	VARCHAR(128)	lock_attributes - ロック属性。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。ロックがない場合、テキスト ID は NONE となり、それ以外の場合、以下のいずれかの組み合わせを '+' 記号で区切ったものとなります。 • ALLOW_NEW • DELETE_IN_BLOCK • ESCALATED • INSERT • NEW_REQUEST • RR • RR_IN_BLOCK • UPDATE_DELETE • WAIT_FOR_AVAIL

表 186. SNAPLOCKWAIT 管理ビューおよび SNAP_GET_LOCKWAIT 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
LOCK_CURRENT_MODE	VARCHAR(10)	lock_current_mode - 移行前の元のロック・モード。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • IN • IS • IX • NON (ロックなしの場合) • NS • NW • S • SIX • U • X • Z
LOCK_NAME	VARCHAR(32)	lock_name - ロック名
LOCK_RELEASE_FLAGS	BIGINT	lock_release_flags - ロック保留解除フラグ。
DATA_PARTITION_ID	INTEGER	data_partition_id - データ・パーティション ID。非パーティション表では、このエレメントは NULL です。
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

SNAPSTMT 管理ビューおよび SNAP_GET_STMT 表関数 - ステートメント・スナップショット情報の検索

SNAPSTMT 管理ビューおよび SNAP_GET_STMT 表関数は、アプリケーション・スナップショットから SQL または XQuery ステートメントについての情報を戻します。

SNAPSTMT 管理ビュー

この管理ビューでは、現在接続されているデータベースのステートメント・スナップショット情報を検索できます。

SNAPAGENT、 SNAPAGENT_MEMORY_POOL、 SNAPAPPL、 SNAPAPPL_INFO、および SNAPSUBSECTION 管理ビューと共に使用すると、SNAPSTMT 管理ビューは、GET SNAPSHOT FOR APPLICATIONS on

database-alias CLP コマンドと同等の情報を提供しますが、すべてのデータベース・パーティションからデータを検索します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 736 ページの表 187を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPSTMT 管理ビューに対する SELECT 特権
- SNAPSTMT 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_STMT 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されている単一パーティション・データベース上で実行されたステートメントのために読み取り、書き込み、および操作の実行が行われた行を検索します。

```
SELECT SUBSTR(STMT_TEXT,1,30) AS STMT_TEXT, ROWS_READ, ROWS_WRITTEN,
   STMT_OPERATION FROM SYSIBMADM.SNAPSTMT
```

以下はこの照会の出力例です。

```
STMT_TEXT     ROWS_READ     ROWS_WRITTEN     STMT_OPERATION
---------...- ---------...- -------------...- --------------------
-                         0                0 FETCH
-                         0                0 STATIC_COMMIT

  2 record(s) selected.
```

SNAP_GET_STMT 表関数

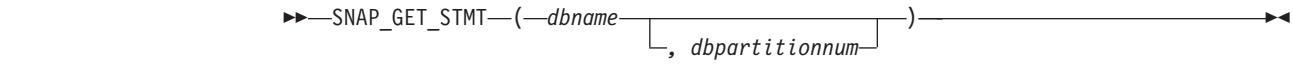
SNAP_GET_STMT 表関数は SNAPSTMT 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションのいずれかの特定のデータベースを対象とした情報を検索することができます。

SNAP_GET_AGENT、 SNAP_GET_AGENT_MEMORY_POOL、
SNAP_GET_APPL_V95、 SNAP_GET_APPL_INFO_V95、および
SNAP_GET_SUBSECTION 表関数と共に使用すると、SNAP_GET_STMT 表関数

は、GET SNAPSHOT FOR ALL APPLICATIONS CLP コマンドと同等の情報を提供しますが、すべてのデータベース・パーティションからデータを検索します。

戻される可能性のある情報の完全なリストは、 736 ページの表 187を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、空ストリングを指定します。現在接続されているデータベースと同じインスタンス内のすべてのデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプが含まれるファイルが存在しない場合、SNAP_GET_STMT 表関数は、現在接続されているデータベースのスナップショットとデータベース・パーティション番号を取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_STMT 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベースの現行データベース・パーティション上で実行されたステートメントのために読み取り、書き込み、および操作の実行が行われた行を検索します。

```
SELECT SUBSTR(STMT_TEXT,1,30) AS STMT_TEXT, ROWS_READ,
   ROWS_WRITTEN, STMT_OPERATION FROM TABLE(SNAP_GET_STMT('',-1)) AS T
```

以下はこの照会の出力例です。

```
STMT_TEXT                      ROWS_READ     ...
------------------------------ ---------...- ...
update t set a=3                           0 ...
SELECT SUBSTR(STMT_TEXT,1,30)              0 ...
-                                          0 ...
-                                          0 ...
update t set a=2                           9 ...
                                             ...
5 record(s) selected.                        ...
```

この照会からの出力 (続き)。

```
... ROWS_WRITTEN     STMT_OPERATION
... ------------...- --------------------
...                0 EXECUTE_IMMEDIATE
...                0 FETCH
...                0 NONE
...                0 NONE
...                1 EXECUTE_IMMEDIATE
...
```

戻される情報

表 187. SNAPSTMT 管理ビューおよび SNAP_GET_STMT 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
DB_NAME	VARCHAR(128)	db_name - データベース名
AGENT_ID	BIGINT	agent_id - アプリケーション・ハンドル (エージェント ID)
ROWS_READ	BIGINT	rows_read - 読み取り行数
ROWS_WRITTEN	BIGINT	rows_written - 書き込み行数
NUM_AGENTS	BIGINT	num_agents - ステートメントで作動しているエージェントの数
AGENTS_TOP	BIGINT	agents_top - 作成されたエージェントの数

表 187. SNAPSTMT 管理ビューおよび SNAP_GET_STMT 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
STMT_TYPE	VARCHAR(20)	stmt_type - ステートメント・タイプ。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • DYNAMIC • NON_STMT • STATIC • STMT_TYPE_UNKNOWN
STMT_OPERATION	VARCHAR(20)	stmt_operation/operation - ステートメント操作。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • CALL • CLOSE • COMPILE • DESCRIBE • EXECUTE • EXECUTE_IMMEDIATE • FETCH • FREE_LOCATOR • GETAA • GETNEXTCHUNK • GETTA • NONE • OPEN • PREP_COMMIT • PREP_EXEC • PREP_OPEN • PREPARE • REBIND • REDIST • REORG • RUNSTATS • SELECT • SET • STATIC_COMMIT • STATIC_ROLLBACK
SECTION_NUMBER	BIGINT	section_number - セクション番号

表 187. SNAPSTMT 管理ビューおよび SNAP_GET_STMT 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
QUERY_COST_ESTIMATE	BIGINT	query_cost_estimate - 照会コストの見積もり
QUERY_CARD_ESTIMATE	BIGINT	query_card_estimate - 照会行数の見積もり
DEGREE_PARALLELISM	BIGINT	degree_parallelism - 並列処理の度合い
STMT_SORTS	BIGINT	stmt_sorts - ステートメント・ソート回数
TOTAL_SORT_TIME	BIGINT	total_sort_time - ソート時間合計
SORT_OVERFLOWS	BIGINT	sort_overflows - ソート・オーバーフロー
INT_ROWS_DELETED	BIGINT	int_rows_deleted - 削除された内部行数
INT_ROWS_UPDATED	BIGINT	int_rows_updated - 更新された内部行数
INT_ROWS_INSERTED	BIGINT	int_rows_inserted - 挿入された内部行数
FETCH_COUNT	BIGINT	fetch_count - 成功した取り出しの数
STMT_START	TIMESTAMP	stmt_start - ステートメント操作開始タイム・スタンプ
STMT_STOP	TIMESTAMP	stmt_stop - ステートメント操作停止タイム・スタンプ
STMT_USR_CPU_TIME_S	BIGINT	stmt_usr_cpu_time - ステートメントに使用されたユーザー CPU 時間 (秒単位)*
STMT_USR_CPU_TIME_MS	BIGINT	stmt_usr_cpu_time - ステートメントに使用されたユーザー CPU 時間 (小数部、マイクロ秒単位)*
STMT_SYS_CPU_TIME_S	BIGINT	stmt_sys_cpu_time - ステートメントが使用したシステム CPU 時間 (秒単位)*
STMT_SYS_CPU_TIME_MS	BIGINT	stmt_sys_cpu_time - ステートメントが使用したシステム CPU 時間 (小数部、マイクロ秒単位)*
STMT_ELAPSED_TIME_S	BIGINT	stmt_elapsed_time - 最新のステートメント経過時間 (秒単位)*
STMT_ELAPSED_TIME_MS	BIGINT	stmt_elapsed_time - 最新のステートメント経過時間 (小数部、マイクロ秒単位)*
BLOCKING_CURSOR	SMALLINT	blocking_cursor - ブロック・カーソル

表 187. SNAPSTMT 管理ビューおよび SNAP_GET_STMT 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
STMT_NODE_NUMBER	SMALLINT	stmt_node_number - ステートメント・ノード
CURSOR_NAME	VARCHAR(128)	cursor_name - カーソル名
CREATOR	VARCHAR(128)	creator - アプリケーション作成者
PACKAGE_NAME	VARCHAR(128)	package_name - パッケージ名
STMT_TEXT	CLOB(16 M)	stmt_text - SQL 動的ステートメント・テキスト
CONSISTENCY_TOKEN	VARCHAR(128)	consistency_token - パッケージ整合性トークン
PACKAGE_VERSION_ID	VARCHAR(128)	package_version_id - パッケージ・バージョン
POOL_DATA_L_READS	BIGINT	pool_data_l_reads - バッファー・プール・データの論理読み取り
POOL_DATA_P_READS	BIGINT	pool_data_p_reads - バッファー・プール・データの物理読み取り
POOL_INDEX_L_READS	BIGINT	pool_index_l_reads - バッファー・プール索引の論理読み取り
POOL_INDEX_P_READS	BIGINT	pool_index_p_reads - バッファー・プール索引の物理読み取り
POOL_XDA_L_READS	BIGINT	pool_xda_l_reads - バッファー・プール XDA データの論理読み取り : モニター・エレメント
POOL_XDA_P_READS	BIGINT	pool_xda_p_reads - バッファー・プール XDA データの物理読み取り : モニター・エレメント
POOL_TEMP_DATA_L_READS	BIGINT	pool_temp_data_l_reads - バッファー・プール一時データの論理読み取り
POOL_TEMP_DATA_P_READS	BIGINT	pool_temp_data_p_reads - バッファー・プール一時データの物理読み取り
POOL_TEMP_INDEX_L_READS	BIGINT	pool_temp_index_l_reads - バッファー・プール一時索引の論理読み取り
POOL_TEMP_INDEX_P_READS	BIGINT	pool_temp_index_p_reads - バッファー・プール一時索引の物理読み取り
POOL_TEMP_XDA_L_READS	BIGINT	pool_temp_xda_l_reads - バッファー・プール一時 XDA データの論理読み取り
POOL_TEMP_XDA_P_READS	BIGINT	pool_temp_xda_p_reads - バッファー・プール一時 XDA データの物理読み取り : モニター・エレメント

表 187. SNAPSTMT 管理ビューおよび SNAP_GET_STMT 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。
* この列の元になるモニター・エレメントの合計消費時間を計算するには、このモニター・エレメントに関する _S で終わっている列で報告されている整数秒と、このモニター・エレメントに関する _MS で終わっている列に報告されている小数秒を、次の式を使用して加算しなければなりません: (*monitor-element-name*_S × 1,000,000 + *monitor-element-name*_MS) ÷ 1,000,000. 例えば、(ELAPSED_EXEC_TIME_S × 1,000,000 + ELAPSED_EXEC_TIME_MS) ÷ 1,000,000。		

SNAPSTORAGE_PATHS 管理ビューおよび SNAP_GET_STORAGE_PATHS_V97 表関数 - 自動ストレージ・パスの情報の検索

SNAPSTORAGE_PATHS 管理ビューと SNAP_GET_STORAGE_PATHS_V97 表関数は、データベースの自動ストレージ・パスのリストを戻します。これには、各ストレージ・パスのファイル・システムの情報、特に db_storage_group 論理データ・グループからの情報が含まれます。

SNAPSTORAGE_PATHS 管理ビュー

この管理ビューを使用すると、現在接続されているデータベースに関する自動ストレージ・パスの情報を検索できます。

SNAPSTORAGE_PATHS 管理ビューを SNAPDB、 SNAPDETAILLOG、SNAPHADR、および SNAPDB_MEMORY_POOL 管理ビューと併せて使用することにより、GET SNAPSHOT FOR DATABASE ON database-alias CLP コマンドと同等の情報を戻します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 742 ページの表 188を参照してください。

許可

- SYSMON 権限
- SNAPSTORAGE_PATHS 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_STORAGE_PATHS_V97 表関数に対する EXECUTE 特権

例

現在接続されている単一パーティション・データベースのストレージ・パスを検索します。

```
SELECT SUBSTR(DB_NAME,1,8) AS DB_NAME, SUBSTR(DB_STORAGE_PATH,1,8)
   AS DB_STORAGE_PATH, SUBSTR(HOSTNAME,1,10) AS HOSTNAME
   FROM SYSIBMADM.SNAPSTORAGE_PATHS
```

以下はこの照会の出力例です。

```
DB_NAME  DB_STORAGE_PATH HOSTNAME
-------- --------------- ----------
STOPATH  d:              JESSICAE

  1 record(s) selected.
```

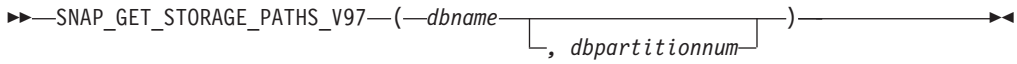
SNAP_GET_STORAGE_PATHS_V97 表関数

SNAP_GET_STORAGE_PATHS_V97 表関数は SNAPSTORAGE_PATHS 管理ビューと同じ情報を戻します。ただし、SNAP_GET_STORAGE_PATHS_V97 表関数の場合は、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションについて、特定のデータベースの情報を検索することができます。

SNAP_GET_STORAGE_PATHS_V97 表関数を SNAP_GET_DB_V95、SNAP_GET_DETAILLOG_V91、 SNAP_GET_HADR、および SNAP_GET_DB_MEMORY_POOL 表関数と併せて使用することにより、GET SNAPSHOT FOR ALL DATABASES CLP コマンドと同等の情報を戻します。

戻される可能性のある情報の完全なリストは、 742 ページの表 188を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、空ストリングを指定します。現在接続されているデータベースと同じインスタンス内のすべてのデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプを持つファイルが存在しない場合には、現在接続されているデータベースのスナップショットとデータベース・パーティション番号を SNAP_GET_STORAGE_PATHS_V97 表関数が取得します。

許可

- SYSMON 権限
- SNAP_GET_STORAGE_PATHS_V97 表関数に対する EXECUTE 特権

例

すべてのアクティブ・データベースに関するストレージ・パスの情報を取り出します。

```
SELECT SUBSTR(DB_NAME,1,8) AS DB_NAME, DB_STORAGE_PATH
   FROM TABLE(SNAP_GET_STORAGE_PATHS_V97(CAST (NULL AS VARCHAR(128)), -1)) AS T
```

以下はこの照会の出力例です。

```
DB_NAME  DB_STORAGE_PATH
-------- -------------------...
STOPATH  /home/jessicae/sdb
MYDB     /home/jessicae/mdb

  2 record(s) selected
```

戻される情報

ファイル・システムの情報を戻すためには、BUFFERPOOL モニター・スイッチをオンにする必要があります。

表 188. SNAPSTORAGE_PATHS 管理ビューおよび SNAP_GET_STORAGE_PATHS_V97 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
DB_NAME	VARCHAR(128)	db_name - データベース名
DB_STORAGE_PATH	VARCHAR(256)	db_storage_path - 自動ストレージ・パス
DB_STORAGE_PATH_WITH_DPE	VARCHAR(256)	未評価のデータベース・パーティション式 (DPE) が含まれる自動ストレージ・パス。ストレージ・パスに DPE が含まれない場合には NULL が戻ります。
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。
DB_STORAGE_PATH_STATE	VARCHAR(16)	自動ストレージ・パスの状態 (現在、可能な値は「IN_USE」、「NOT_IN_USE」、「DROP_PENDING」のいずれか)。

表 188. SNAPSTORAGE_PATHS 管理ビューおよび SNAP_GET_STORAGE_PATHS_V97 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
FS_ID	VARCHAR(22)	fs_id - 固有のファイル・システム識別番号
FS_TOTAL_SIZE	BIGINT	fs_total_size - ファイル・システムの合計サイズ
FS_USED_SIZE	BIGINT	fs_used_size - ファイル・システム上で使用されるスペースの量
STO_PATH_FREE_SIZE	BIGINT	sto_path_free_sz - 自動ストレージ・パスのフリー・スペース

SNAPSUBSECTION 管理ビューおよび SNAP_GET_SUBSECTION 表関数 - subsection 論理モニター・グループ・スナップショット情報の検索

SNAPSUBSECTION 管理ビューおよび SNAP_GET_SUBSECTION 表関数は、アプリケーション・サブセクション情報として、subsection 論理モニター・グループの情報を戻します。

SNAPSUBSECTION 管理ビュー

この管理ビューでは、現在接続されているデータベースの subsection 論理モニター・グループ・スナップショット情報を検索できます。

SNAPAGENT、 SNAPAGENT_MEMORY_POOL、 SNAPAPPL、 SNAPAPPL_INFO、および SNAPSTMT 管理ビューと共に使用すると、 SNAPSUBSECTION 管理ビューは、GET SNAPSHOT FOR APPLICATIONS on database-alias CLP コマンドと同等の情報を提供しますが、すべてのデータベース・パーティションからデータを検索します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 746 ページの表 189を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPSUBSECTION 管理ビューに対する SELECT 特権
- SNAPSUBSECTION 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_SUBSECTION 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

すべてのデータベース・パーティションで実行するサブセクションの状況を取得します。

```
SELECT DB_NAME, STMT_TEXT, SS_STATUS, DBPARTITIONNUM
   FROM SYSIBMADM.SNAPSUBSECTION
   ORDER BY DB_NAME, SS_STATUS, DBPARTITIONNUM
```

以下はこの照会の出力例です。

```
DB_NAME     STMT_TEXT                  SS_STATUS     DBPARTITIONNUM
--------...- ----------------------...- ----------...- --------------
SAMPLE      select * from EMPLOYEE     EXEC                       0
SAMPLE      select * from EMPLOYEE     EXEC                       1
```

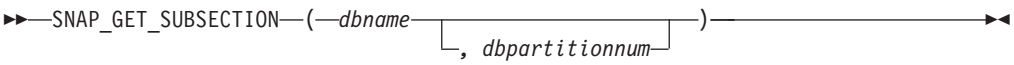
SNAP_GET_SUBSECTION 表関数

SNAP_GET_SUBSECTION 表関数は SNAPSUBSECTION 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションのいずれかの特定のデータベースを対象とした情報を検索することができます。

戻される可能性のある情報の完全なリストは、 746 ページの表 189を参照してください。

SNAP_GET_AGENT、 SNAP_GET_AGENT_MEMORY_POOL、 SNAP_GET_APPL_V95、 SNAP_GET_APPL_INFO_V95、および SNAP_GET_STMT 表関数と共に使用すると、SNAP_GET_SUBSECTION 表関数は、GET SNAPSHOT FOR ALL APPLICATIONS CLP コマンドと同等の情報を提供しますが、すべてのデータベース・パーティションからデータを検索します。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できま

す。現在接続されているデータベースからのスナップショットを取得するには、空ストリングを指定します。現在接続されているデータベースと同じインスタンス内のすべてのデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプが含まれるファイルが存在しない場合、SNAP_GET_SUBSECTION 表関数は、現在接続されているデータベースのスナップショットとデータベース・パーティション番号を取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_SUBSECTION 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

すべてのデータベース・パーティションで実行するサブセクションの状況を取得します。

```
SELECT DB_NAME, STMT_TEXT, SS_STATUS, DBPARTITIONNUM
   FROM TABLE(SYSPROC.SNAP_GET_SUBSECTION( '', 0 )) as T
   ORDER BY DB_NAME, SS_STATUS, DBPARTITIONNUM
```

以下はこの照会の出力例です。

```
DB_NAME     STMT_TEXT                  SS_STATUS     DBPARTITIONNUM
-------...- ----------------------...- ---------...- --------------
SAMPLE      select * from EMPLOYEE     EXEC                       0
SAMPLE      select * from EMPLOYEE     EXEC                       1
```

戻される情報

表 189. SNAPSUBSECTION 管理ビューおよび SNAP_GET_SUBSECTION 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
DB_NAME	VARCHAR(128)	db_name - データベース名
STMT_TEXT	CLOB(16 M)	stmt_text - SQL 動的ステートメント・テキスト
SS_EXEC_TIME	BIGINT	ss_exec_time - サブセクション実行経過時間
TQ_TOT_SEND_SPILLS	BIGINT	tq_tot_send_spills - オーバーフローした表キュー・バッファーの合計数
TQ_CUR_SEND_SPILLS	BIGINT	tq_cur_send_spills - オーバーフローした表キュー・バッファーの現在数
TQ_MAX_SEND_SPILLS	BIGINT	tq_max_send_spills - 表キュー・バッファー・オーバーフローの最大数
TQ_ROWS_READ	BIGINT	tq_rows_read - 表キューから読み取られた行数
TQ_ROWS_WRITTEN	BIGINT	tq_rows_written - 表キューに書き込まれた行数
ROWS_READ	BIGINT	rows_read - 読み取り行数
ROWS_WRITTEN	BIGINT	rows_written - 書き込み行数
SS_USR_CPU_TIME_S	BIGINT	ss_usr_cpu_time - サブセクションに使用されたユーザー CPU 時間 (秒単位)*
SS_USR_CPU_TIME_MS	BIGINT	ss_usr_cpu_time - サブセクションに使用されたユーザー CPU 時間 (小数部、マイクロ秒単位)*
SS_SYS_CPU_TIME_S	BIGINT	ss_sys_cpu_time - サブセクションに使用されたシステム CPU 時間 (秒単位)*
SS_SYS_CPU_TIME_MS	BIGINT	ss_sys_cpu_time - サブセクションに使用されたシステム CPU 時間 (小数部、マイクロ秒単位)*
SS_NUMBER	INTEGER	ss_number - サブセクション番号
SS_STATUS	VARCHAR(20)	ss_status - サブセクションの状況。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • EXEC • TQ_WAIT_TO_RCV • TQ_WAIT_TO_SEND • COMPLETED

表 189. SNAPSUBSECTION 管理ビューおよび SNAP_GET_SUBSECTION 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
SS_NODE_NUMBER	SMALLINT	ss_node_number - サブセクション・ノード番号
TQ_NODE_WAITED_FOR	SMALLINT	tq_node_waited_for - 表キュー上のノード待機
TQ_WAIT_FOR_ANY	INTEGER	tq_wait_for_any - 表キュー上のノード送信待機
TQ_ID_WAITING_ON	INTEGER	tq_id_waiting_on - ノード上の表キュー待機
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。
* この列の元になるモニター・エレメントの合計消費時間を計算するには、このモニター・エレメントに関する _S で終わっている列で報告されている整数秒と、このモニター・エレメントに関する _MS で終わっている列に報告されている小数秒を、次の式を使用して加算しなければなりません: (*monitor-element-name*_S × 1,000,000 + *monitor-element-name*_MS) ÷ 1,000,000. 例えば、(ELAPSED_EXEC_TIME_S × 1,000,000 + ELAPSED_EXEC_TIME_MS) ÷ 1,000,000。		

SNAPSWITCHES 管理ビューおよび SNAP_GET_SWITCHES 表関数 - データベース・スナップショットのスイッチ状態情報の検索

SNAPSWITCHES 管理ビューおよび SNAP_GET_SWITCHES 表関数は、データベース・スナップショットのスイッチ状態に関する情報を戻します。

SNAPSWITCHES 管理ビュー

このビューは、GET DBM MONITOR SWITCHES CLP コマンドと同等のデータを提供します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 750 ページの表 190を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPSWITCHES 管理ビューに対する SELECT 特権
- SNAPSWITCHES 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_SWITCHES 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

すべてのデータベース・パーティションにおける DBM モニターのスイッチ状態情報を検索します。

```
SELECT UOW_SW_STATE, STATEMENT_SW_STATE, TABLE_SW_STATE, BUFFPOOL_SW_STATE,
   LOCK_SW_STATE, SORT_SW_STATE, TIMESTAMP_SW_STATE,
   DBPARTITIONNUM FROM SYSIBMADM.SNAPSWITCHES
```

以下はこの照会の出力例です。

```
UOW_SW_STATE STATEMENT_SW_STATE TABLE_SW_STATE BUFFPOOL_SW_STATE ...
------------ ------------------ -------------- ----------------- ...
           0                  0              0                 0 ...
           0                  0              0                 0 ...
           0                  0              0                 0 ...
                                                                 ...
  3 record selected.
```

この照会からの出力 (続き)。

```
... LOCK_SW_STATE SORT_SW_STATE TIMESTAMP_SW_STATE DBPARTITIONNUM
... ------------- ------------- ------------------ --------------
...             1             0                  1              0
...             1             0                  1              1
...             1             0                  1              2
```

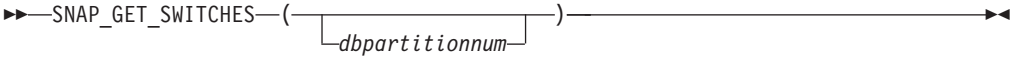
SNAP_GET_SWITCHES 表関数

SNAP_GET_SWITCHES 表関数は SNAPSWITCHES 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションを対象とした情報を検索することができます。

この表関数は、GET DBM MONITOR SWITCHES CLP コマンドと同等のデータを提供します。

戻される可能性のある情報の完全なリストは、 750 ページの表 190を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。この入力オプションが使用されない場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbpartitionnum が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプが含まれるファイルが存在しない場合、SNAP_GET_SWITCHES 表関数は、現在接続されているデータベースのスナップショットとデータベース・パーティション番号を取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_SWITCHES 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現行データベース・パーティションにおける DBM モニターのスイッチ状態情報を検索します。

```
SELECT UOW_SW_STATE, STATEMENT_SW_STATE, TABLE_SW_STATE,
   BUFFPOOL_SW_STATE,LOCK_SW_STATE, SORT_SW_STATE, TIMESTAMP_SW_STATE
   FROM TABLE(SNAP_GET_SWITCHES(-1)) AS T
```

以下はこの照会の出力例です。

```
UOW_SW_STATE STATEMENT_SW_STATE TABLE_SW_STATE...
------------ ----------------- -------------...
           1                 1             1...
                                             ...
1 record(s) selected.                        ...
```

この照会からの出力 (続き)。

```
... BUFFPOOL_SW_STATE LOCK_SW_STATE SORT_SW_STATE TIMESTAMP_SW_STATE
... ----------------- ------------- ------------- ------------------
...                 1             1             0                  1
```

戻される情報

表 190. SNAPSWITCHES 管理ビューおよび SNAP_GET_SWITCHES 表関数によって戻される情報

列名	データ・タイプ	説明
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
UOW_SW_STATE	SMALLINT	作業単位モニター記録スイッチの状態 (0 または 1)。
UOW_SW_TIME	TIMESTAMP	作業単位モニター記録スイッチがオンの場合、このスイッチがオンになった日時。
STATEMENT_SW_STATE	SMALLINT	SQL ステートメント・モニター記録スイッチの状態 (0 または 1)。
STATEMENT_SW_TIME	TIMESTAMP	SQL ステートメント・モニター記録スイッチがオンの場合、このスイッチがオンになった日時。
TABLE_SW_STATE	SMALLINT	表アクティビティー・モニター記録スイッチの状態 (0 または 1)。
TABLE_SW_TIME	TIMESTAMP	表アクティビティー・モニター記録スイッチがオンの場合、このスイッチがオンになった日時。
BUFFPOOL_SW_STATE	SMALLINT	バッファー・プール・アクティビティー・モニター記録スイッチの状態 (0 または 1)。
BUFFPOOL_SW_TIME	TIMESTAMP	バッファー・プール・アクティビティー・モニター記録スイッチがオンの場合、このスイッチがオンになった日時。
LOCK_SW_STATE	SMALLINT	ロック・モニター記録スイッチの状態 (0 または 1)。
LOCK_SW_TIME	TIMESTAMP	ロック・モニター記録スイッチがオンの場合、このスイッチがオンになった日時。
SORT_SW_STATE	SMALLINT	ソート・モニター記録スイッチの状態 (0 または 1)。
SORT_SW_TIME	TIMESTAMP	ソート・モニター記録スイッチがオンの場合、このスイッチがオンになった日時。
TIMESTAMP_SW_STATE	SMALLINT	タイム・スタンプのモニター記録スイッチの状態 (0 または 1)。
TIMESTAMP_SW_TIME	TIMESTAMP	タイム・スタンプのモニター記録スイッチがオンの場合、このスイッチがオンになった日付と時刻。
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

SNAPTAB 管理ビューおよび SNAP_GET_TAB_V91 表関数 - table 論理データ・グループのスナップショット情報の検索

SNAPTAB 管理ビューおよび SNAP_GET_TAB_V91 表関数は、table 論理データ・グループからのスナップショット情報を戻します。

SNAPTAB 管理ビュー

この管理ビューを使用すると、現在接続されているデータベースに関する table 論理データ・グループのスナップショット情報を検索できます。

SNAPTAB_REORG 管理ビューと併せて使用することにより、SNAPTAB 管理ビューは GET SNAPSHOT FOR TABLES ON database-alias CLP コマンドと同等の情報を戻します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 753 ページの表 191を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPTAB 管理ビューに対する SELECT 特権
- SNAPTAB 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_TAB_V91 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

すべてのアクティブな表のスキーマと名前を取り出します。

```
SELECT SUBSTR(TABSCHEMA,1,8), SUBSTR(TABNAME,1,15) AS TABNAME, TAB_TYPE,
   DBPARTITIONNUM FROM SYSIBMADM.SNAPTAB
```

以下はこの照会の出力例です。

```
TABSCHEMA TABNAME         TAB_TYPE     DBPARTITIONNUM
--------- --------------- ------------ --------------
SYSTOOLS  HMON_ATM_INFO   USER_TABLE                0

  1 record selected.
```

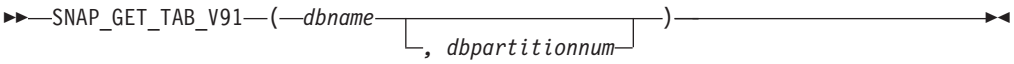
SNAP_GET_TAB_V91 表関数

SNAP_GET_TAB_V91 表関数は SNAPTAB 管理ビューと同じ情報を戻します。ただし、SNAP_GET_TAB_V91 表関数の場合は、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションについて、特定のデータベースの情報を検索することができます。

SNAP_GET_TAB_REORG 表関数と併せて使用することにより、SNAP_GET_TAB_V91 表関数は GET SNAPSHOT FOR TABLES ON database-alias CLP コマンドと同等の情報を戻します。

戻される可能性のある情報の完全なリストは、 753 ページの表 191を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL または空ストリングを指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプを持つファイルが存在しない場合には、現在接続されているデータベースのスナップショットとデータベース・パーティション番号を SNAP_GET_TAB_V91 表関数が取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_TAB_V91 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベースの集約ビューとして、アクティブな表のリストを検索します。

```
SELECT SUBSTR(TABSCHEMA,1,8) AS TABSCHEMA, SUBSTR(TABNAME,1,15) AS TABNAME,
   TAB_TYPE, DBPARTITIONNUM FROM TABLE(SNAP_GET_TAB('',-2)) AS T
```

以下はこの照会の出力例です。

```
TABSCHEMA TABNAME         TAB_TYPE      DBPARTITIONNUM
--------- --------------- ------------- --------------
SYSTOOLS  HMON_ATM_INFO   USER_TABLE                 -
JESSICAE  EMPLOYEE        USER_TABLE                 -
```

戻される情報

表 191. SNAPTAB 管理ビューおよび SNAP_GET_TAB_V91 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
TABSCHEMA	VARCHAR(128)	table_schema - 表スキーマ名
TABNAME	VARCHAR(128)	table_name - 表名
TAB_FILE_ID	BIGINT	table_file_id - 表ファイル ID
TAB_TYPE	VARCHAR(14)	table_type - 表タイプ。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • USER_TABLE • DROPPED_TABLE • TEMP_TABLE • CATALOG_TABLE • REORG_TABLE
DATA_OBJECT_PAGES	BIGINT	data_object_pages - データ・オブジェクト・ページ数
INDEX_OBJECT_PAGES	BIGINT	index_object_pages - 索引オブジェクト・ページ数

表 191. SNAPTAB 管理ビューおよび SNAP_GET_TAB_V91 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
LOB_OBJECT_PAGES	BIGINT	lob_object_pages - LOB オブジェクト・ページ数
LONG_OBJECT_PAGES	BIGINT	long_object_pages - 長オブジェクト・ページ数
XDA_OBJECT_PAGES	BIGINT	xda_object_pages - XDA オブジェクト・ページ数
ROWS_READ	BIGINT	rows_read - 読み取り行数
ROWS_WRITTEN	BIGINT	rows_written - 書き込み行数
OVERFLOW_ACCESSES	BIGINT	overflow_accesses - オーバーフロー・レコードへのアクセス
PAGE_REORGS	BIGINT	page_reorgs - ページ再編成
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。
TBSP_ID	BIGINT	tablespace_id - 表スペース ID
DATA_PARTITION_ID	INTEGER	data_partition_id - データ・パーティション ID。非パーティション表では、このエレメントは NULL になります。

SNAPTAB_REORG 管理ビューおよび SNAP_GET_TAB_REORG 表関数 - 表再編成スナップショット情報の検索

SNAPTAB_REORG 管理ビューおよび SNAP_GET_TAB_REORG 表関数は、表再編成情報を戻します。再編成された表がない場合は、0 行が戻されます。データ・パーティション表が再編成されるとき、各データ・パーティションに対して 1 つのレコードが戻されます。データ・パーティション表の特定の 1 つのデータ・パーティションだけが再編成される場合、そのパーティションのレコードだけが戻されます。

SNAPTAB_REORG 管理ビュー

この管理ビューでは、現在接続されているデータベースの表再編成スナップショット情報を検索できます。

SNAPTAB 管理ビューと共に使用すると、SNAPTAB_REORG 管理ビューは、GET SNAPSHOT FOR TABLES ON database-alias CLP コマンドと同等のデータを提供します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 758 ページの表 192を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPTAB_REORG 管理ビューに対する SELECT 特権
- SNAPTAB_REORG 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_TAB_REORG 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベース上のすべてのデータベース・パーティションでの再編成操作の詳細を選択します。

```
SELECT SUBSTR(TABNAME, 1, 15) AS TAB_NAME, SUBSTR(TABSCHEMA, 1, 15)
   AS TAB_SCHEMA, REORG_PHASE, SUBSTR(REORG_TYPE, 1, 20) AS REORG_TYPE,
   REORG_STATUS, REORG_COMPLETION, DBPARTITIONNUM
   FROM SYSIBMADM.SNAPTAB_REORG ORDER BY DBPARTITIONNUM
```

以下はこの照会の出力例です。

```
TAB_NAME     TAB_SCHEMA     REORG_PHASE      ...
---------...- ----------...- ---------------- ...
EMPLOYEE     DBUSER         REPLACE          ...
EMPLOYEE     DBUSER         REPLACE          ...
EMPLOYEE     DBUSER         REPLACE          ...
                                             ...
3 record(s) selected.
```

この照会からの出力 (続き)。

```
... REORG_TYPE           REORG_STATUS REORG_COMPLETION DBPARTITIONNUM
... -------------------- ------------ ---------------- --------------
... RECLAIM+OFFLINE+ALLO COMPLETED    SUCCESS                       0
... RECLAIM+OFFLINE+ALLO COMPLETED    SUCCESS                       1
... RECLAIM+OFFLINE+ALLO COMPLETED    SUCCESS                       2
```

マルチディメンション・クラスタリング (MDC) 表からエクステントを再利用するための再編成操作に関するすべての情報を、SNAPTAB_REORG 管理ビューから選択します。

```
db2 -v "select * from sysibmadm.snaptab_reorg"

TABNAME  REORG_PHASE       REORG_MAX_PHASE   REORG_TYPE
-------- ----------------- ----------------- --------------------------
T1       RELEASE           3                 RECLAIM_EXTENTS+ALLOW_WRITE
```

```
REORG_STATUS REORG_COMPLETION REORG_START                REORG_END
------------ ---------------- -------------------------- --------------------------
COMPLETED    SUCCESS          2008-09-24-14.35.30.734741 2008-09-24-14.35.31.460674
```

SNAP_GET_TAB_REORG 表関数

SNAP_GET_TAB_REORG 表関数は SNAPTAB_REORG 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションのいずれかの特定のデータベースを対象とした情報を検索することができます。

SNAP_GET_TAB 表関数と共に使用すると、SNAP_GET_TAB_REORG 表関数は、GET SNAPSHOT FOR TABLES ON database-alias CLP コマンドと同等のデータを提供します。

戻される可能性のある情報の完全なリストは、 758 ページの表 192を参照してください。

構文

スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL または空ストリングを指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプが含まれるファイルが存在しない場合、SNAP_GET_TAB_REORG 表関数は、現在接続されているデータベースのスナップ

ショットとデータベース・パーティション番号を取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_TAB_REORG 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベース上のデータベース・パーティション 1 での再編成操作の詳細を選択します。

```
SELECT SUBSTR(TABNAME, 1, 15) AS TAB_NAME, SUBSTR(TABSCHEMA, 1, 15)
   AS TAB_SCHEMA, REORG_PHASE, SUBSTR(REORG_TYPE, 1, 20) AS REORG_TYPE,
   REORG_STATUS, REORG_COMPLETION, DBPARTITIONNUM
   FROM TABLE( SNAP_GET_TAB_REORG('', 1)) AS T
```

以下はこの照会の出力例です。

```
TAB_NAME     TAB_SCHEMA     REORG_PHASE     REORG_TYPE           ...
---------...- ----------...- ----------...- -------------------- ...
EMPLOYEE     DBUSER         REPLACE         RECLAIM+OFFLINE+ALLO ...
                                                                 ...
  1 record(s) selected.                                          ...
```

この照会からの出力 (続き)。

```
... REORG_STATUS REORG_COMPLETION DBPARTITIONNUM
... ------------ ---------------- --------------
... COMPLETED    SUCCESS                     1
...
```

SNAP_GET_TAB_REORG 表関数を使用して、マルチディメンション・クラスタリング (MDC) 表からエクステントを再利用するための再編成操作に関するすべての情報を選択します。

```
db2 -v "select * from table(snap_get_tab_reorg(''))"

TABNAME  REORG_PHASE       REORG_MAX_PHASE   REORG_TYPE
-------- ----------------- ----------------- -------------------------
T1       RELEASE           3                 RECLAIM_EXTENTS+ALLOW_WRITE

REORG_STATUS REORG_COMPLETION REORG_START                REORG_END
------------ ---------------- -------------------------- --------------------------
COMPLETED    SUCCESS          2008-09-24-14.35.30.734741 2008-09-24-14.35.31.460674
```

戻される情報

表 192. SNAPTAB_REORG 管理ビューおよび SNAP_GET_TAB_REORG 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
TABNAME	VARCHAR (128)	table_name - 表名
TABSCHEMA	VARCHAR (128)	table_schema - 表スキーマ名
PAGE_REORGS	BIGINT	page_reorgs - ページ再編成
REORG_PHASE	VARCHAR (16)	reorg_phase - 表再編成フェーズ。このインターフェースは、`sqlmon.h` での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • BUILD • DICT_SAMPLE • INDEX_RECREATE • REPLACE • SORT • SCAN • DRAIN • RELEASE または SORT+DICT_SAMPLE。
REORG_MAX_PHASE	INTEGER	reorg_max_phase - 表再編成の最大フェーズ数
REORG_CURRENT_COUNTER	BIGINT	reorg_current_counter - 表再編成の進行状況
REORG_MAX_COUNTER	BIGINT	reorg_max_counter - 表再編成の合計量

表 192. SNAPTAB_REORG 管理ビューおよび SNAP_GET_TAB_REORG 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント

REORG_TYPE	VARCHAR (128)	reorg_type - 表再編成の属性。このインターフェースは、以下の ID の組み合わせを '+' 記号で区切ったものを使用してテキスト ID を戻します。
		
		以下のいずれかが使用されます。
		• RECLAIM
		• RECLUSTER
		• RECLAIM_EXTS
		さらに以下のいずれかが使用されます。
		• +OFFLINE
		• +ONLINE
		アクセス・モードが指定されている場合、以下のいずれかが使用されます。
		• +ALLOW_NONE
		• +ALLOW_READ
		• +ALLOW_WRITE
		オフラインで RECLUSTER オプションが指定されている場合、以下のいずれかが使用されます。
		• +INDEXSCAN
		• +TABLESCAN
		オフラインの場合、以下のいずれかが使用されます。
		• +LONGLOB
		• +DATAONLY
		オフラインで、オプションが指定されている場合、以下の任意のものが使用されます。
		• +CHOOSE_TEMP
		• +KEEPDICTIONARY
		• +RESETDICTIONARY
		オンラインで、オプションが指定されている場合、以下が使用されます。
		• +NOTRUNCATE
		
		例 1: REORG TABLE TEST.EMPLOYEE が実行された場合、以下のように表示されます。
		`RECLAIM+OFFLINE+ALLOW_READ+DATAONLY +KEEPDICTIONARY`
		
		例 2: REORG TABLE TEST.EMPLOYEE INDEX EMPIDX INDEXSCAN が実行された場合、以下のように表示されます。
		`RECLUSTER+OFFLINE+ALLOW_READ+INDEXSCAN +DATAONLY+KEEPDICTIONARY`

表 192. SNAPTAB_REORG 管理ビューおよび SNAP_GET_TAB_REORG 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
REORG_STATUS	VARCHAR (10)	reorg_status - 表再編成の状況。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • COMPLETED • PAUSED • STARTED • STOPPED • TRUNCATE
REORG_COMPLETION	VARCHAR (10)	reorg_completion - 表再編成完了フラグ。このインターフェースは、sqlmon.h での定義を基にしてテキスト ID を戻します。以下のいずれかとなります。 • FAIL • SUCCESS
REORG_START	TIMESTAMP	reorg_start - 表再編成開始時刻
REORG_END	TIMESTAMP	reorg_end - 表再編成終了時刻
REORG_PHASE_START	TIMESTAMP	reorg_phase_start - 表再編成フェーズ開始時刻
REORG_INDEX_ID	BIGINT	reorg_index_id - 表の再編成に使用される索引
REORG_TBSPC_ID	BIGINT	reorg_tbspc_id - 表が再編成される表スペース
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。
DATA_PARTITION_ID	INTEGER	data_partition_id - データ・パーティション ID。非パーティション表では、このエレメントは NULL になります。
REORG_ ROWSCOMPRESSED	BIGINT	reorg_rows_compressed - 圧縮行数
REORG_ROWSREJECTED	BIGINT	reorg_rows_rejected_for_compression - 圧縮がリジェクトされる行
REORG_LONG_TBSPC_ID	BIGINT	reorg_long_tbspc_id - 長いオブジェクトが再編成される表スペース

SNAPTBSP 管理ビューおよび SNAP_GET_TBSP_V91 表関数 - 表スペース論理データ・グループのスナップショット情報の検索

SNAPTBSP 管理ビューおよび SNAP_GET_TBSP_V91 表関数は、tablespace 論理データ・グループからのスナップショット情報を戻します。

SNAPTBSP 管理ビュー

この管理ビューを使用すると、現在接続されているデータベースに関する table space 論理データ・グループのスナップショット情報を検索できます。

SNAPTBSP_PART、 SNAPTBSP_QUIESCER、 SNAPTBSP_RANGE、 SNAPCONTAINER 管理ビューと併せて使用することにより、SNAPTBSP 管理ビューは GET SNAPSHOT FOR TABLESPACES ON database-alias CLP コマンドと同等の情報を戻します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 763 ページの表 193を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPTBSP 管理ビューに対する SELECT 特権
- SNAPTBSP 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_TBSP_V91 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベースの、カタログ・データベース・パーティションの表スペースのリストを検索します。

```
SELECT SUBSTR(TBSP_NAME,1,30) AS TBSP_NAME, TBSP_ID, TBSP_TYPE,
   TBSP_CONTENT_TYPE FROM SYSIBMADM.SNAPTBSP WHERE DBPARTITIONNUM = 1
```

以下はこの照会の出力例です。

```
TBSP_NAME  TBSP_ID     TBSP_TYPE  TBSP_CONTENT_TYPE
---------- -------...- ---------- -----------------
TEMPSPACE1           1 SMS        SYSTEMP
USERSPACE1           2 DMS        LONG

2 record(s) selected.
```

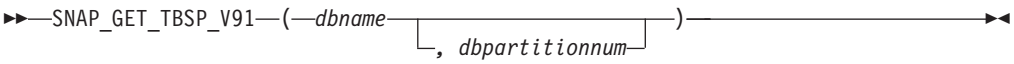
SNAP_GET_TBSP_V91 表関数

SNAP_GET_TBSP_V91 表関数は SNAPTBSP 管理ビューと同じ情報を戻します。ただし、SNAP_GET_TBSP_V91 表関数の場合は、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションについて、特定のデータベースに関する情報を検索することができます。

SNAP_GET_TBSP_PART_V91、 SNAP_GET_TBSP_QUIESCER、 SNAP_GET_TBSP_RANGE、 SNAP_GET_CONTAINER_V91 表関数と併せて使用することにより、SNAP_GET_TBSP_V91 表関数は GET SNAPSHOT FOR TABLESPACES ON database-alias CLP コマンドと同等の情報を戻します。

戻される可能性のある情報の完全なリストは、 763 ページの表 193を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL または空ストリングを指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。 *dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプを持つファイルが存在しない場合には、現在接続されているデータベースのスナップショットとデータベース・パーティション番号を SNAP_GET_TBSP_V91 表関数が取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_TBSP_V91 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベースの、すべてのデータベース・パーティションの表スペースのリストを検索します。

```
SELECT SUBSTR(TBSP_NAME,1,10) AS TBSP_NAME, TBSP_ID, TBSP_TYPE,
   TBSP_CONTENT_TYPE, DBPARTITIONNUM FROM TABLE(SNAP_GET_TBSP_V91('')) AS T
```

以下はこの照会の出力例です。

```
TBSP_NAME  TBSP_ID     TBSP_TYPE  TBSP_CONTENT_TYPE DBPARTITIONNUM
---------- -------...- ---------- ----------------- --------------
TEMPSPACE1           1 SMS        SYSTEMP                         1
USERSPACE1           2 DMS        LONG                            1
SYSCATSPAC           0 DMS        ANY                             0
TEMPSPACE1           1 SMS        SYSTEMP                         0
USERSPACE1           2 DMS        LONG                            0
SYSTOOLSPA           3 DMS        LONG                            0
TEMPSPACE1           1 SMS        SYSTEMP                         2
USERSPACE1           2 DMS        LONG                            2

  8 record(s) selected.
```

戻される情報

表 193. SNAPTBSP 管理ビューおよび SNAP_GET_TBSP_V91 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
TBSP_NAME	VARCHAR(128)	tablespace_name - 表スペース名
TBSP_ID	BIGINT	tablespace_id - 表スペース ID
TBSP_TYPE	VARCHAR(10)	tablespace_type - 表スペース・タイプ。このインターフェースは、sqlutil.h の定義に基づくテキスト ID を戻します。これは次のいずれかです。 • DMS • SMS

表 193. SNAPTBSP 管理ビューおよび SNAP_GET_TBSP_V91 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
TBSP_CONTENT_TYPE	VARCHAR(10)	tablespace_content_type - 表スペースのコンテンツ・タイプ。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • ANY • LARGE • SYSTEMP • USRTEMP
TBSP_PAGE_SIZE	BIGINT	tablespace_page_size - 表スペースのページ・サイズ
TBSP_EXTENT_SIZE	BIGINT	tablespace_extent_size - 表スペースのエクステント・サイズ
TBSP_PREFETCH_SIZE	BIGINT	tablespace_prefetch_size - 表スペースのプリフェッチ・サイズ
TBSP_CUR_POOL_ID	BIGINT	tablespace_cur_pool_id - 現在使用中のバッファー・プール
TBSP_NEXT_POOL_ID	BIGINT	tablespace_next_pool_id - 次の始動時に使用されるバッファー・プール
FS_CACHING	SMALLINT	fs_caching - ファイル・システム・キャッシング
POOL_DATA_L_READS	BIGINT	pool_data_l_reads - バッファー・プール・データの論理読み取り
POOL_DATA_P_READS	BIGINT	pool_data_p_reads - バッファー・プール・データの物理読み取り
POOL_TEMP_DATA_L_READS	BIGINT	pool_temp_data_l_reads - バッファー・プール一時データの論理読み取り
POOL_TEMP_DATA_P_READS	BIGINT	pool_temp_data_p_reads - バッファー・プール一時データの物理読み取り
POOL_ASYNC_DATA_READS	BIGINT	pool_async_data_reads - バッファー・プール非同期データ読み取り
POOL_DATA_WRITES	BIGINT	pool_data_writes - バッファー・プールへのデータの書き込み
POOL_ASYNC_DATA_WRITES	BIGINT	pool_async_data_writes - バッファー・プール非同期データ書き込み
POOL_INDEX_L_READS	BIGINT	pool_index_l_reads - バッファー・プール索引の論理読み取り
POOL_INDEX_P_READS	BIGINT	pool_index_p_reads - バッファー・プール索引の物理読み取り

表 193. SNAPTBSP 管理ビューおよび SNAP_GET_TBSP_V91 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
POOL_TEMP_INDEX_L_READS	BIGINT	pool_temp_index_l_reads - バッファー・プール一時索引の論理読み取り
POOL_TEMP_INDEX_P_READS	BIGINT	pool_temp_index_p_reads - バッファー・プール一時索引の物理読み取り
POOL_ASYNC_INDEX_READS	BIGINT	pool_async_index_reads - バッファー・プール非同期索引読み取り
POOL_INDEX_WRITES	BIGINT	pool_index_writes - バッファー・プール索引の書き込み
POOL_ASYNC_INDEX_WRITES	BIGINT	pool_async_index_writes - バッファー・プール非同期索引書き込み
POOL_XDA_L_READS	BIGINT	pool_xda_l_reads - バッファー・プール XDA データの論理読み取り
POOL_XDA_P_READS	BIGINT	pool_xda_p_reads - バッファー・プール XDA データの物理読み取り
POOL_XDA_WRITES	BIGINT	pool_xda_writes - バッファー・プール XDA データの書き込み
POOL_ASYNC_XDA_READS	BIGINT	pool_async_xda_reads - バッファー・プール非同期 XDA データ読み取り
POOL_ASYNC_XDA_WRITES	BIGINT	pool_async_xda_writes - バッファー・プール非同期 XDA データ書き込み
POOL_TEMP_XDA_L_READS	BIGINT	pool_temp_xda_l_reads - バッファー・プール一時 XDA データの論理読み取り
POOL_TEMP_XDA_P_READS	BIGINT	pool_temp_xda_p_reads - バッファー・プール一時 XDA データの物理読み取り : モニター・エレメント
POOL_READ_TIME	BIGINT	pool_read_time - バッファー・プール物理読み取り時間の合計
POOL_WRITE_TIME	BIGINT	pool_write_time - バッファー・プール物理書き込み時間の合計
POOL_ASYNC_READ_TIME	BIGINT	pool_async_read_time - バッファー・プール非同期読み取り時間
POOL_ASYNC_WRITE_TIME	BIGINT	pool_async_write_time - バッファー・プール非同期書き込み時間
POOL_ASYNC_DATA_ READ_REQS	BIGINT	pool_async_data_read_reqs - バッファー・プール非同期読み取り要求
POOL_ASYNC_INDEX_ READ_REQS	BIGINT	pool_async_index_read_reqs - バッファー・プール非同期索引読み取り要求

表 193. SNAPTBSP 管理ビューおよび SNAP_GET_TBSP_V91 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
POOL_ASYNC_XDA_ READ_REQS	BIGINT	pool_async_xda_read_reqs - バッファー・プール非同期 XDA 読み取り要求
POOL_NO_VICTIM_BUFFER	BIGINT	pool_no_victim_buffer - バッファー・プールの非ビクティム・バッファー数
DIRECT_READS	BIGINT	direct_reads - データベースからの直接読み取り
DIRECT_WRITES	BIGINT	direct_writes - データベースへの直接書き込み
DIRECT_READ_REQS	BIGINT	direct_read_reqs - 直接読み取り要求
DIRECT_WRITE_REQS	BIGINT	direct_write_reqs - 直接書き込み要求
DIRECT_READ_TIME	BIGINT	direct_read_time - 直接読み取り時間
DIRECT_WRITE_TIME	BIGINT	direct_write_time - 直接書き込み時間
FILES_CLOSED	BIGINT	files_closed - 閉じられたデータベース・ファイル
UNREAD_PREFETCH_PAGES	BIGINT	unread_prefetch_pages - 読み取り不能プリフェッチ・ページ
TBSP_REBALANCER_MODE	VARCHAR(10)	tablespace_rebalancer_mode - リバランサー・モード。このインターフェースは、`sqlmon.h` での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • NO_REBAL • FWD_REBAL • REV_REBAL
TBSP_USING_AUTO_STORAGE	SMALLINT	tablespace_using_auto_storage - 自動ストレージが使用可能な表スペース
TBSP_AUTO_RESIZE_ENABLED	SMALLINT	tablespace_auto_resize_enabled - 表スペースの自動サイズ変更可能
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

SNAPTBSP_PART 管理ビューおよび SNAP_GET_TBSP_PART_V97 表関数 - tablespace_nodeinfo 論理データ・グループのスナップショット情報の検索

SNAPTBSP_PART 管理ビューおよび SNAP_GET_TBSP_PART_V97 表関数は、tablespace_nodeinfo 論理データ・グループからのスナップショット情報を戻します。

SNAPTBSP_PART 管理ビュー

この管理ビューを使用すると、現在接続されているデータベースに関する tablespace_nodeinfo 論理データ・グループのスナップショット情報を検索することができます。

SNAPTBSP、SNAPTBSP_QUIESCER、SNAPTBSP_RANGE、SNAPCONTAINER 管理ビューと併せて使用することにより、SNAPTBSP_PART 管理ビューは GET SNAPSHOT FOR TABLESPACES ON database-alias CLP コマンドと同等の情報を戻します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、769 ページの表 194を参照してください。

許可

- SYSMON 権限
- SNAPTBSP_PART 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_TBSP_PART_V97 表関数に対する EXECUTE 特権

例

現在接続されているデータベースのすべてのデータベース・パーティションの表スペースとその状態のリストを検索します。

```
SELECT SUBSTR(TBSP_NAME,1,30) AS TBSP_NAME, TBSP_ID,
   SUBSTR(TBSP_STATE,1,30) AS TBSP_STATE, DBPARTITIONNUM
   FROM SYSIBMADM.SNAPTBSP_PART
```

以下はこの照会の出力例です。

```
TBSP_NAME       TBSP_ID     TBSP_STATE     DBPARTITIONNUM
-----------...- -------...- ----------...- --------------
SYSCATSPACE               0 NORMAL                       0
TEMPSPACE1                1 NORMAL                       0
USERSPACE1                2 NORMAL                       0
TEMPSPACE1                1 NORMAL                       1
USERSPACE1                2 NORMAL                       1

   5 record(s) selected.
```

SNAP_GET_TBSP_PART_V97 表関数

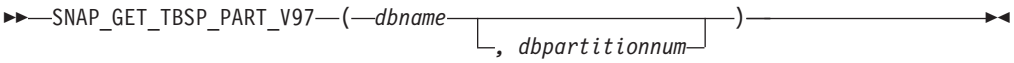
SNAP_GET_TBSP_PART_V97 表関数は SNAPTBSP_PART 管理ビューと同じ情報を戻します。ただし、SNAP_GET_TBSP_PART_V97 表関数の場合は、特定のデータベ

ース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションについて、特定のデータベースに関する情報を検索することができます。

SNAP_GET_TBSP_V97、 SNAP_GET_TBSP_QUIESCER、 SNAP_GET_TBSP_RANGE、 SNAP_GET_CONTAINER_V91 表関数と併せて使用することにより、SNAP_GET_TBSP_PART_V97 表関数は GET SNAPSHOT FOR TABLESPACES ON database-alias CLP コマンドと同等の情報を戻します。

戻される可能性のある情報の完全なリストは、 769 ページの表 194を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL または空ストリングを指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。 *dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプを持つファイルが存在しない場合には、現在接続されているデータベースのスナップショットとデータベース・パーティション番号を SNAP_GET_TBSP_PART_V97 表関数が取得します。

許可

- SYSMON 権限
- SNAP_GET_TBSP_PART_V97 表関数に対する EXECUTE 特権

例

接続されているデータベースの接続されているデータベース・パーティションの表スペースとその状態のリストを検索します。

```
SELECT SUBSTR(TBSP_NAME,1,30) AS TBSP_NAME, TBSP_ID,
   SUBSTR(TBSP_STATE,1,30) AS TBSP_STATE
   FROM TABLE(SNAP_GET_TBSP_PART_V97(CAST(NULL AS VARCHAR(128)),-1)) AS T
```

以下はこの照会の出力例です。

```
TBSP_NAME                      TBSP_ID              TBSP_STATE
------------------------------ -------------------- ------------...-
SYSCATSPACE                                       0 NORMAL
TEMPSPACE1                                        1 NORMAL
USERSPACE1                                        2 NORMAL
SYSTOOLSPACE                                      3 NORMAL
SYSTOOLSTMPSPACE                                  4 NORMAL

  5 record(s) selected.
```

戻される情報

表 194. SNAPTBSP_PART 管理ビューおよび SNAP_GET_TBSP_PART_V97 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
TBSP_NAME	VARCHAR (128)	tablespace_name - 表スペース名
TBSP_ID	BIGINT	tablespace_id - 表スペース ID

表 194. SNAPTBSP_PART 管理ビューおよび SNAP_GET_TBSP_PART_V97 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
TBSP_STATE	VARCHAR (256)	tablespace_state - 表スペースの状態。このインターフェースは、sqlutil.h での定義に基づくテキスト ID を戻します。これは次のものを「+」符号で分離して組み合わせたものになります。 • BACKUP_IN_PROGRESS • BACKUP_PENDING • DELETE_PENDING • DISABLE_PENDING • DROP_PENDING • LOAD_IN_PROGRESS • LOAD_PENDING • NORMAL • OFFLINE • PSTAT_CREATION • PSTAT_DELETION • QUIESCED_EXCLUSIVE • QUIESCED_SHARE • QUIESCED_UPDATE • REBAL_IN_PROGRESS • REORG_IN_PROGRESS • RESTORE_IN_PROGRESS • RESTORE_PENDING • ROLLFORWARD_IN_PROGRESS • ROLLFORWARD_PENDING • STORDEF_ALLOWED • STORDEF_CHANGED • STORDEF_FINAL_VERSION • STORDEF_PENDING • SUSPEND_WRITE
TBSP_PREFETCH_SIZE	BIGINT	tablespace_prefetch_size - 表スペースのプリフェッチ・サイズ
TBSP_NUM_QUIESCERS	BIGINT	tablespace_num_quiescers - 静止プログラム数
TBSP_STATE_CHANGE_OBJECT_ID	BIGINT	tablespace_state_change_object_id - 状態変更オブジェクト ID
TBSP_STATE_CHANGE_TBSP_ID	BIGINT	tablespace_state_change_ts_id - 状態変更表スペース ID

表 194. SNAPTBSP_PART 管理ビューおよび SNAP_GET_TBSP_PART_V97 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
TBSP_MIN_RECOVERY_TIME	TIMESTAMP	tablespace_min_recovery_time - ロールフォワードの最小リカバリー時間
TBSP_TOTAL_PAGES	BIGINT	tablespace_total_pages - 表スペース内の合計ページ数
TBSP_USABLE_PAGES	BIGINT	tablespace_usable_pages - 表スペース内の使用可能ページ数
TBSP_USED_PAGES	BIGINT	tablespace_used_pages - 表スペース内の使用されているページ数
TBSP_FREE_PAGES	BIGINT	tablespace_free_pages - 表スペース内のフリー・ページ数
TBSP_PENDING_FREE_PAGES	BIGINT	tablespace_pending_free_pages - 表スペース内のペンディング・フリー・ページ数
TBSP_PAGE_TOP	BIGINT	tablespace_page_top - 表スペース最高水準点
REBALANCER_MODE	VARCHAR (30)	tablespace_rebalancer_mode - リバランサー・モード。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • FWD_REBAL • NO_REBAL • REV_REBAL • FWD_REBAL_OF_2PASS • REV_REBAL_OF_2PASS
REBALANCER_EXTENTS_REMAINING	BIGINT	tablespace_rebalancer_extents_remaining - リバランサーで処理されるエクステントの合計数
REBALANCER_EXTENTS_PROCESSED	BIGINT	tablespace_rebalancer_extents_processed - リバランサーで処理されたエクステントの数
REBALANCER_PRIORITY	BIGINT	tablespace_rebalancer_priority - 現行のリバランサー優先順位
REBALANCER_START_TIME	TIMESTAMP	tablespace_rebalancer_start_time - リバランサー開始時刻
REBALANCER_RESTART_TIME	TIMESTAMP	tablespace_rebalancer_restart_time - リバランサー再始動時刻
REBALANCER_LAST_EXTENT_MOVED	BIGINT	tablespace_rebalancer_last_extent_moved - リバランサーによって最後に移動されたエクステント
TBSP_NUM_RANGES	BIGINT	tablespace_num_ranges - 表スペース・マップ内の範囲数

表 194. SNAPTBSP_PART 管理ビューおよび SNAP_GET_TBSP_PART_V97 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
TBSP_NUM_CONTAINERS	BIGINT	tablespace_num_containers - 表スペース内のコンテナー数
TBSP_INITIAL_SIZE	BIGINT	tablespace_initial_size - 表スペースの初期サイズ
TBSP_CURRENT_SIZE	BIGINT	tablespace_current_size - 表スペースの現行サイズ
TBSP_MAX_SIZE	BIGINT	tablespace_max_size - 表スペースの最大サイズ
TBSP_INCREASE_SIZE	BIGINT	tablespace_increase_size - バイト単位のサイズの増加
TBSP_INCREASE_SIZE_PERCENT	SMALLINT	tablespace_increase_size_percent - パーセント単位のサイズの増加
TBSP_LAST_RESIZE_TIME	TIMESTAMP	tablespace_last_resize_time - 最後にサイズ変更が正常に行われた時刻
TBSP_LAST_RESIZE_FAILED	SMALLINT	tablespace_last_resize_failed - 失敗した最後のサイズ変更
TBSP_PATHS_DROPPED	SMALLINT	表スペースが、ドロップされた 1 つ以上のストレージ・パスにあることを示します (0 - いいえ、1 - はい)
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

SNAPTBSP_QUIESCER 管理ビューおよび SNAP_GET_TBSP_QUIESCER 表関数 - quiescer 表スペース・スナップショット情報の検索

SNAPTBSP_QUIESCER 管理ビューおよび SNAP_GET_TBSP_QUIESCER 表関数は、表スペース・スナップショットから、静止プログラムに関する情報を戻します。

SNAPTBSP_QUIESCER 管理ビュー

この管理ビューでは、現在接続されているデータベースの静止プログラム表スペース・スナップショット情報を検索できます。

SNAPTBSP、SNAPTBSP_PART、SNAPTBSP_RANGE、SNAPCONTAINER 管理ビューと共に使用すると、SNAPTBSP_QUIESCER 管理ビューは、GET SNAPSHOT FOR TABLESPACES ON database-alias CLP コマンドと同等の情報を提供します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 776 ページの表 195を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPTBSP_QUIESCER 管理ビューに対する SELECT 特権
- SNAPTBSP_QUIESCER 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_TBSP_QUIESCER 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベースのすべてのデータベース・パーティションの静止した表スペースの情報を検索します。

```
SELECT SUBSTR(TBSP_NAME, 1, 10) AS TBSP_NAME, QUIESCER_TS_ID,
   QUIESCER_OBJ_ID, QUIESCER_AUTH_ID, QUIESCER_AGENT_ID,
   QUIESCER_STATE, DBPARTITIONNUM
   FROM SYSIBMADM.SNAPTBSP_QUIESCER ORDER BY DBPARTITIONNUM
```

以下はこの照会の出力例です。

```
TBSP_NAME  QUIESCER_TS_ID     QUIESCER_OBJ_ID     QUIESCER_AUTH_ID     ..
---------- --------------...- ---------------...- ----------------...- ..
USERSPACE1                  2                   5 SWALKTY              ..
USERSPACE1                  2                   5 SWALKTY              ..

  2 record(s) selected.
```

この照会からの出力 (続き)。

```
... QUIESCER_AGENT_ID    QUIESCER_STATE DBPARTITIONNUM
... -------------------- -------------- --------------
...                    0 EXCLUSIVE                   0
...                65983 EXCLUSIVE                   1
```

例: 範囲パーティション表の名前の判別

表が範囲パーティション化されていて、静止状態に保たれている場合、表スペース ID および表 ID の値は、SYSCAT.TABLES 内のものとは異なる表記となります。これらの ID は、符号なしの短い表記となります。静止した表の名前を検索するには、QEUIESCER_TS_ID から 65536 (最大値) を減算した表スペース ID を計算することによって、まず符号付きの短い表記を検索する必要があります。それから、

この表スペース ID を使用して、静止した表を特定します。(実際の表スペース ID は、表内の各範囲パーティションの SYSCAT.DATAPARTITIONS にあります。)

```
SELECT SUBSTR(TBSP_NAME, 1, 10) AS TBSP_NAME,
  CASE WHEN QUIESCER_TS_ID = 65530
    THEN QUIESCER_TS_ID - 65536
    ELSE QUIESCER_TS_ID END as tbspaceid,
  CASE WHEN QUIESCER_TS_ID = 65530
    THEN QUIESCER_OBJ_ID - 65536
    ELSE QUIESCER_OBJ_ID END as tableid
  FROM SYSIBMADM.SNAPTBSP_QUIESCER
  ORDER BY DBPARTITIONNUM
```

以下はこの照会の出力例です。

```
TBSP_NAME     TBSPACEID    TABLEID
------------- ------------ --------
TABDATA       -6           -32768
DATAMART      -6           -32765
SMALL         5            17

  3 record(s) selected.
```

上記の照会で得られた特定の TBSPACEID および TABLEID を使用して、SYSCAT.TABLES から表スキーマおよび表名を検索します。

```
SELECT CHAR(tabschema, 10)tabschema, CHAR(tabname,15)tabname
  FROM SYSCAT.TABLES
  WHERE tbspaceid = -6 AND tableid in (-32768,-32765)
```

以下はこの照会の出力例です。

```
TABSCHEMA    TABNAME
------------ --------
TPCD         ORDERS_RP
TPCD         ORDERS_DMART

  2 record(s) selected.
```

```
SELECT CHAR(tabschema, 10)tabschema, CHAR(tabname,15)tabname
  FROM SYSCAT.TABLES
  WHERE tbspaceid = 5 AND tableid = 17
```

以下はこの照会の出力例です。

```
TABSCHEMA    TABNAME
------------ --------
TPCD         NATION

  1 record(s) selected.
```

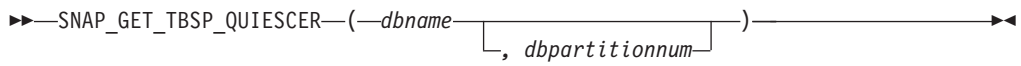
SNAP_GET_TBSP_QUIESCER 表関数

SNAP_GET_TBSP_QUIESCER 表関数は SNAPTBSP_QUIESCER 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションのいずれかの特定のデータベースを対象とした情報を検索することができます。

SNAP_GET_TBSP_V91、 SNAP_GET_TBSP_PART_V91、 SNAP_GET_TBSP_RANGE、 SNAP_GET_CONTAINER_V91 表関数と共に使用すると、SNAP_GET_TBSP_QUIESCER 表関数は、GET SNAPSHOT FOR TABLESPACES ON database-alias CLP コマンドと同等の情報を提供します。

戻される可能性のある情報の完全なリストは、 776 ページの表 195を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL または空ストリングを指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプが含まれるファイルが存在しない場合、SNAP_GET_TBSP_QUIESCER 表関数は、現在接続されているデータベースのスナップショットとデータベース・パーティション番号を取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_TBSP_QUIESCER 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT

- SYSADM

例

現在接続されているデータベースのデータベース・パーティション 1 の静止した表スペースの情報を検索します。

```
SELECT SUBSTR(TBSP_NAME, 1, 10) AS TBSP_NAME, QUIESCER_TS_ID,
   QUIESCER_OBJ_ID, QUIESCER_AUTH_ID, QUIESCER_AGENT_ID,
   QUIESCER_STATE, DBPARTITIONNUM
   FROM TABLE( SYSPROC.SNAP_GET_TBSP_QUIESCER( '', 1)) AS T
```

以下はこの照会の出力例です。

```
TBSP_NAME  QUIESCER_TS_ID     QUIESCER_OBJ_ID     QUIESCER_AUTH_ID     ...
---------- --------------...- ---------------...- ----------------...- ...
USERSPACE1                 2                   5 SWALKTY              ...

  1 record(s) selected.
```

この照会からの出力 (続き)。

```
... QUIESCER_AGENT_ID    QUIESCER_STATE DBPARTITIONNUM
... -------------------- -------------- --------------
...                65983 EXCLUSIVE                   1
```

戻される情報

表 195. SNAPTBSP_QUIESCER 管理ビューおよび SNAP_GET_TBSP_QUIESCER 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
TBSP_NAME	VARCHAR(128)	tablespace_name - 表スペース名
QUIESCER_TS_ID	BIGINT	quiescer_ts_id - 静止プログラム表スペース ID
QUIESCER_OBJ_ID	BIGINT	quiescer_obj_id - 静止プログラム・オブジェクト ID
QUIESCER_AUTH_ID	VARCHAR(128)	quiescer_auth_id - 静止プログラム・ユーザー許可 ID
QUIESCER_AGENT_ID	BIGINT	quiescer_agent_id - 静止プログラム・エージェント ID
QUIESCER_STATE	VARCHAR(14)	quiescer_state - 静止プログラムの状態。このインターフェースは、sqlutil.h での定義を基にしてテキスト ID を戻します。以下のいずれかとなります。 • EXCLUSIVE • UPDATE • SHARE
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

SNAPTBSP_RANGE 管理ビューおよび SNAP_GET_TBSP_RANGE 表関数 - 範囲スナップショット情報の検索

SNAPTBSP_RANGE 管理ビューおよび SNAP_GET_TBSP_RANGE 表関数は、範囲スナップショットから情報を戻します。

SNAPTBSP_RANGE 管理ビュー

この管理ビューでは、現在接続されているデータベースの範囲スナップショット情報を検索できます。

SNAPTBSP、SNAPTBSP_PART、SNAPTBSP_QUIESCER、および SNAPCONTAINER 管理ビューと共に使用すると、SNAPTBSP_RANGE 管理ビューは、GET SNAPSHOT FOR TABLESPACES ON database-alias CLP コマンドと同等の情報を提供します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 780 ページの表 196を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPTBSP_RANGE 管理ビューに対する SELECT 特権
- SNAPTBSP_RANGE 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_TBSP_RANGE 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベースのすべてのデータベース・パーティションの表スペース範囲についての情報を選択します。

```
SELECT TBSP_ID, SUBSTR(TBSP_NAME, 1, 15) AS TBSP_NAME, RANGE_NUMBER,
   RANGE_STRIPE_SET_NUMBER, RANGE_OFFSET, RANGE_MAX_PAGE,
   RANGE_MAX_EXTENT, RANGE_START_STRIPE, RANGE_END_STRIPE,
   RANGE_ADJUSTMENT, RANGE_NUM_CONTAINER, RANGE_CONTAINER_ID,
   DBPARTITIONNUM FROM SYSIBMADM.SNAPTBSP_RANGE
   ORDER BY DBPARTITIONNUM
```

以下はこの照会の出力例です。

```
TBSP_ID     TBSP_NAME       RANGE_NUMBER     RANGE_STRIPE_SET_NUMBER ...
-------...- --------------- ------------...- ----------------------- ...
          0 SYSCATSPACE                    0                       0 ...
          2 USERSPACE1                     0                       0 ...
          3 SYSTOOLSPACE                   0                       0 ...
          2 USERSPACE1                     0                       0 ...
          2 USERSPACE1                     0                       0 ...

  5 record(s) selected.
```

この照会からの出力 (続き)。

```
... RANGE_OFFSET     RANGE_MAX_PAGE       RANGE_MAX_EXTENT     ...
... ------------...- -------------------- -------------------- ...
...                0                11515                 2878 ...
...                0                  479                   14 ...
...                0                  251                   62 ...
...                0                  479                   14 ...
...                0                  479                   14 ...
```

この照会からの出力 (続き)。

```
... RANGE_START_STRIPE   RANGE_END_STRIPE     RANGE_ADJUSTMENT     ...
... -------------------- -------------------- -------------------- ...
...                    0                 2878                    0 ...
...                    0                   14                    0 ...
...                    0                   62                    0 ...
...                    0                   14                    0 ...
...                    0                   14                    0 ...
```

この照会からの出力 (続き)。

```
... RANGE_NUM_CONTAINER  RANGE_CONTAINER_ID   DBPARTITIONNUM
... -------------------- -------------------- --------------
...                    1                    0              0
...                    1                    0              0
...                    1                    0              0
...                    1                    0              1
...                    1                    0              2
```

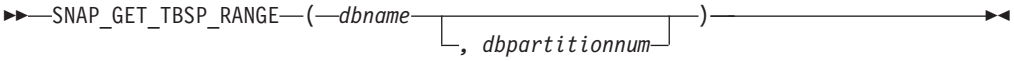
SNAP_GET_TBSP_RANGE 表関数

SNAP_GET_TBSP_RANGE 表関数は SNAPTBSP_RANGE 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションのいずれかの特定のデータベースを対象とした情報を検索することができます。

SNAP_GET_TBSP_V91、SNAP_GET_TBSP_PART_V91、SNAP_GET_TBSP_QUIESCER、および SNAP_GET_CONTAINER_V91 表関数と共に使用すると、SNAP_GET_TBSP_RANGE 表関数は、GET SNAPSHOT FOR TABLESPACES ON database-alias CLP コマンドと同等の情報を提供します。

戻される可能性のある情報の完全なリストは、 780 ページの表 196を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL または空ストリングを指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプが含まれるファイルが存在しない場合、SNAP_GET_TBSP_RANGE 表関数は、現在接続されているデータベースのスナップショットとデータベース・パーティション番号を取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_TBSP_RANGE 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベース・パーティション上で tbsp_id = 2 である表スペースの表スペース範囲の情報を選択します。

```
SELECT TBSP_ID, SUBSTR(TBSP_NAME, 1, 15) AS TBSP_NAME, RANGE_NUMBER,
   RANGE_STRIPE_SET_NUMBER, RANGE_OFFSET, RANGE_MAX_PAGE, RANGE_MAX_EXTENT,
   RANGE_START_STRIPE, RANGE_END_STRIPE, RANGE_ADJUSTMENT,
   RANGE_NUM_CONTAINER, RANGE_CONTAINER_ID
   FROM TABLE(SNAP_GET_TBSP_RANGE('',-1)) AS T WHERE TBSP_ID = 2
```

以下はこの照会の出力例です。

```
TBSP_ID     TBSP_NAME       RANGE_NUMBER     ...
-------...- --------------- ------------...- ...
          2 USERSPACE1                    0 ...

  1 record(s) selected.
```

この照会からの出力 (続き)。

```
... RANGE_STRIPE_SET_NUMBER RANGE_OFFSET     RANGE_MAX_PAGE        ...
... ----------------------- ------------...- --------------...---- ...
...                       0               0                  3967 ...
```

この照会からの出力 (続き)。

```
... RANGE_MAX_EXTENT     RANGE_START_STRIPE   RANGE_END_STRIPE     ...
... -------------------- -------------------- -------------------- ...
...                  123                    0                  123 ...
```

この照会からの出力 (続き)。

```
... RANGE_ADJUSTMENT     RANGE_NUM_CONTAINER RANGE_CONTAINER_ID
... -------------------- ------------------- -------------------
...                    0                   1                   0
```

戻される情報

表 196. SNAPTBSP_RANGE 管理ビューおよび SNAP_GET_TBSP_RANGE 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
TBSP_ID	BIGINT	tablespace_id - 表スペース ID
TBSP_NAME	VARCHAR(128)	tablespace_name - 表スペース名
RANGE_NUMBER	BIGINT	range_number - 範囲番号
RANGE_STRIPE_SET_NUMBER	BIGINT	range_stripe_set_number - ストライプ・セット番号
RANGE_OFFSET	BIGINT	range_offset - 範囲オフセット
RANGE_MAX_PAGE	BIGINT	range_max_page_number - 範囲内の最大ページ
RANGE_MAX_EXTENT	BIGINT	range_max_extent - 範囲内の最大エクステント
RANGE_START_STRIPE	BIGINT	range_start_stripe - 開始ストライプ
RANGE_END_STRIPE	BIGINT	range_end_stripe - 終了ストライプ
RANGE_ADJUSTMENT	BIGINT	range_adjustment - 範囲調整
RANGE_NUM_CONTAINER	BIGINT	range_num_containers - 範囲内コンテナーの数
RANGE_CONTAINER_ID	BIGINT	range_container_id - 範囲コンテナー

表 196. SNAPTBSP_RANGE 管理ビューおよび SNAP_GET_TBSP_RANGE 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

SNAPUTIL 管理ビューおよび SNAP_GET_UTIL 表関数 - utility_info 論理データ・グループ・スナップショット情報の検索

SNAPUTIL 管理ビューおよび SNAP_GET_UTIL 表関数は、utility_info 論理データ・グループからのユーティリティー・スナップショット情報を戻します。

SNAPUTIL 管理ビュー

SNAPUTIL_PROGRESS 管理ビューと組み合わせて使用すると、SNAPUTIL 管理ビューは、LIST UTILITIES SHOW DETAIL CLP コマンドと同じ情報を提供します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、783 ページの表 197を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPUTIL 管理ビューに対する SELECT 特権
- SNAPUTIL 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_UTIL 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

接続されているデータベースが含まれるインスタンス内のすべてのアクティブ・データベースのすべてのデータベース・パーティション上にある、ユーティリティーとその状態のリストを検索します。

```
SELECT UTILITY_TYPE, UTILITY_PRIORITY, SUBSTR(UTILITY_DESCRIPTION, 1, 72)
   AS UTILITY_DESCRIPTION, SUBSTR(UTILITY_DBNAME, 1, 17) AS
   UTILITY_DBNAME, UTILITY_STATE, UTILITY_INVOKER_TYPE, DBPARTITIONNUM
   FROM SYSIBMADM.SNAPUTIL ORDER BY DBPARTITIONNUM
```

以下はこの照会の出力例です。

```
UTILITY_TYPE     UTILITY_PRIORITY ...
------------...- ---------------- ...
LOAD                            - ...
LOAD                            - ...
LOAD                            - ...

3 record(s) selected.
```

この照会からの出力 (続き)。

```
... UTILITY_DESCRIPTION                                                    ...
... -----------------------------------------------------------------------...
... ONLINE LOAD DEL AUTOMATIC INDEXING INSERT COPY NO TEST .LOADTEST ...
... ONLINE LOAD DEL AUTOMATIC INDEXING INSERT COPY NO TEST .LOADTEST ...
... ONLINE LOAD DEL AUTOMATIC INDEXING INSERT COPY NO TEST .LOADTEST ...
```

この照会からの出力 (続き)。

```
... UTILITY_DBNAME    UTILITY_STATE UTILITY_INVOKER_TYPE DBPARTITIONNUM
... ----------------- ------------- -------------------- --------------
... SAMPLE            EXECUTE       USER                            0
... SAMPLE            EXECUTE       USER                            1
... SAMPLE            EXECUTE       USER                            2
```

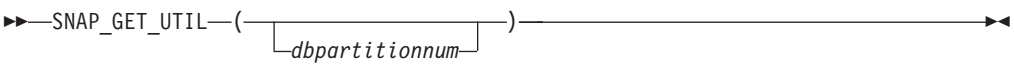
SNAP_GET_UTIL 表関数

SNAP_GET_UTIL 表関数は SNAPUTIL 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションを対象とした情報を検索することができます。

SNAP_GET_UTIL_PROGRESS 表関数と組み合わせて使用すると、SNAP_GET_UTIL 表関数は、LIST UTILITIES SHOW DETAIL CLP コマンドと同じ情報を提供します。

戻される可能性のある情報の完全なリストは、 783 ページの表 197を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。この入力オプションが使用されない場合、データはすべてのアクティブなデータベー

ス・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbpartitionnum が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプが含まれるファイルが存在しない場合、SNAP_GET_UTIL 表関数は、現在接続されているデータベースのスナップショットとデータベース・パーティション番号を取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_UTIL 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

データベース SAMPLE 上の現在接続されているデータベース・パーティションのユーティリティー ID (そのタイプと状態を含む) のリストを検索します。

```
SELECT UTILITY_ID, UTILITY_TYPE, STATE
   FROM TABLE(SNAP_GET_UTIL(-1)) AS T WHERE UTILITY_DBNAME='SAMPLE'
```

以下はこの照会の出力例です。

```
UTILITY_ID           UTILITY_TYPE               STATE
-------------------- -------------------------- --------
                   1 BACKUP                     EXECUTE

  1 record(s) selected.
```

戻される情報

表 197. SNAPUTIL 管理ビューおよび SNAP_GET_UTIL 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
UTILITY_ID	INTEGER	utility_id - ユーティリティー ID。データベース・パーティションに固有。

表 197. SNAPUTIL 管理ビューおよび SNAP_GET_UTIL 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
UTILITY_TYPE	VARCHAR(26)	utility_type - ユーティリティー・タイプ。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • ASYNC_INDEX_CLEANUP • BACKUP • CRASH_RECOVERY • LOAD • REBALANCE • REDISTRIBUTE • REORG • RESTART_RECREATE_INDEX • RESTORE • ROLLFORWARD_RECOVERY • RUNSTATS
UTILITY_PRIORITY	INTEGER	utility_priority - ユーティリティー優先度。ユーティリティーがスロットルをサポートする場合には優先順位、それ以外の場合は NULL。
UTILITY_DESCRIPTION	VARCHAR(2048)	utility_description - ユーティリティー記述。NULL にすることもできます。
UTILITY_DBNAME	VARCHAR(128)	utility_dbname - ユーティリティーで操作されるデータベース
UTILITY_START_TIME	TIMESTAMP	utility_start_time - ユーティリティー開始時刻
UTILITY_STATE	VARCHAR(10)	utility_state - ユーティリティー状態。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • ERROR • EXECUTE • WAIT
UTILITY_INVOKER_TYPE	VARCHAR(10)	utility_invoker_type - ユーティリティー呼び出し側タイプ。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • AUTO • USER

表 197. SNAPUTIL 管理ビューおよび SNAP_GET_UTIL 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。
PROGRESS_LIST_ATTR	VARCHAR(10)	progress_list_attr - 現在の進行リストの属性
PROGRESS_LIST_CUR_SEQ_NUM	INTEGER	progress_list_current_seq_num - 現行の進行リストのシーケンス番号

SNAPUTIL_PROGRESS 管理ビューおよび SNAP_GET_UTIL_PROGRESS 表関数 - progress 論理データ・グループ・スナップショット情報の検索

SNAPUTIL_PROGRESS 管理ビューおよび SNAP_GET_UTIL_PROGRESS 表関数は、特に progress 論理データ・グループの、ユーティリティー進行状況のスナップショット情報を戻します。

SNAPUTIL_PROGRESS 管理ビュー

SNAPUTIL 管理ビューと組み合わせて使用すると、SNAPUTIL_PROGRESS 管理ビューは、LIST UTILITIES SHOW DETAIL CLP コマンドと同じ情報を提供します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、787 ページの表 198を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPUTIL_PROGRESS 管理ビューに対する SELECT 特権
- SNAPUTIL_PROGRESS 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_UTIL_PROGRESS 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

ユーティリティー ID ごとの合計進行単位および完了した進行単位の詳細を検索します。

```
SELECT SELECT UTILITY_ID, PROGRESS_TOTAL_UNITS, PROGRESS_COMPLETED_UNITS,
   DBPARTITIONNUM FROM SYSIBMADM.SNAPUTIL_PROGRESS
```

以下はこの照会の出力例です。

```
UTILITY_ID PROGRESS_TOTAL_UNITS PROGRESS_COMPLETED_UNITS DBPARTITIONNU
---------- ------------------- ----------------------- ------------
         7                  10                       5            0
         9                  10                       5            1

  1 record(s) selected.
```

SNAP_GET_UTIL_PROGRESS 表関数

SNAP_GET_UTIL_PROGRESS 表関数は SNAPUTIL_PROGRESS 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションのいずれかの特定のデータベースを対象とした情報を検索することができます。

SNAP_GET_UTIL 表関数と組み合わせて使用すると、SNAP_GET_UTIL_PROGRESS 表関数は、LIST UTILITIES SHOW DETAIL CLP コマンドと同じ情報を提供します。

戻される可能性のある情報の完全なリストは、 787 ページの表 198を参照してください。

構文

スキーマは SYSPROC です。

表関数パラメーター

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。この入力オプションが使用されない場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbpartitionnum が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプが含まれるファイルが存在しない場合、SNAP_GET_UTIL_PROGRESS 表関数は、現在接続されてい

るデータベースのスナップショットとデータベース・パーティション番号を取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_UTIL_PROGRESS 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているパーティション上のユーティリティーの進行状況の詳細を検索します。

```
SELECT UTILITY_ID, PROGRESS_TOTAL_UNITS, PROGRESS_COMPLETED_UNITS,
   DBPARTITIONNUM FROM TABLE(SNAP_GET_UTIL_PROGRESS(-1)) as T
```

以下はこの照会の出力例です。

```
UTILITY_ID PROGRESS_TOTAL_UNITS PROGRESS_COMPLETED_UNITS DBPARTITIONNUM
---------- -------------------- ------------------------ --------------
         7                   10                        5              0

  1 record(s) selected.
```

戻される情報

表 198. SNAPUTIL_PROGRESS 管理ビューおよび SNAP_GET_UTIL_PROGRESS 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
UTILITY_ID	INTEGER	utility_id - ユーティリティー ID。データベース・パーティションに固有。
PROGRESS_SEQ_NUM	INTEGER	progress_seq_num - 進行シーケンス番号。逐次の場合、フェーズの数。並行の場合、NULL の場合があります。

表 198. SNAPUTIL_PROGRESS 管理ビューおよび SNAP_GET_UTIL_PROGRESS 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
UTILITY_STATE	VARCHAR(16)	utility_state - ユーティリティー状態。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • ERROR • EXECUTE • WAIT
PROGRESS_DESCRIPTION	VARCHAR(2048)	progress_description - 進行の記述
PROGRESS_START_TIME	TIMESTAMP	progress_start_time - 進行開始時刻。フェーズが開始済みの場合には開始時刻、それ以外の場合は NULL。
PROGRESS_WORK_METRIC	VARCHAR(16)	progress_work_metric - 進行作業メトリック。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • NOT_SUPPORT • BYTES • EXTENTS • INDEXES • PAGES • ROWS • TABLES
PROGRESS_TOTAL_UNITS	BIGINT	progress_total_units - 合計進行作業単位
PROGRESS_COMPLETED_UNITS	BIGINT	progress_completed_units - 完了した進行作業単位
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

SNAP_WRITE_FILE プロシージャー

SNAP_WRITE_FILE プロシージャーはシステム・スナップショット・データを、インスタンス・ディレクトリーの tmp サブディレクトリーにあるファイルに書き込みます。

構文

スキーマは SYSPROC です。

プロシージャー・パラメーター

requestType
有効なスナップショット要求タイプを指定する、タイプ VARCHAR(32) の入力引数。可能な要求タイプは、sqlmon.h での定義を基にしたテキスト ID です。以下のいずれかとなります。

- APPL_ALL
- BUFFERPOOLS_ALL
- DB2
- DBASE_ALL
- DBASE_LOCKS
- DBASE_TABLES
- DBASE_TABLESPACES
- DYNAMIC_SQL

dbname
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。"Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL または空ストリングを指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

プロシージャーを実行するには、ユーザーに SYSADM、SYSCTRL、SYSMON、または SYSMON 権限が必要です。保存されたスナップショットは、スナップショット表関数への入力として NULL 値を渡すことにより、SYSADM、 SYSCTRL、SYSMAINT、または SYSMON 権限のないユーザーでも読み取れます。

例

'DB2' の要求タイプ (SQLMA_DB2 に相当) を指定し、現在接続されているデータベースおよび現行データベース・パーティションをデフォルトにすることで、データベース・マネージャー情報のスナップショットをとります。

```
CALL SYSPROC.SNAP_WRITE_FILE ('DB2', '', -1)
```

この場合、スナップショット・データは、インスタンス一時ディレクトリー (UNIX オペレーティング・システムでは sqllib/tmp/SQLMA_DB2.dat、Windows オペレー

ティング・システムでは sqllib¥DB2¥tmp¥SQLMA_DB2.dat) に書き込まれます。

使用上の注意

未認識の入力パラメーターが指定された場合、「SQL2032N "REQUEST_TYPE" パラメーターが無効です」エラーが戻されます。

SNAPAGENT 管理ビューおよび SNAP_GET_AGENT 表関数 - agent 論理データ・グループのアプリケーション・スナップショット情報の検索

SNAPAGENT 管理ビューおよび SNAP_GET_AGENT 表関数は、アプリケーション・スナップショットから、特に agent 論理データ・グループのエージェント情報を戻します。

SNAPAGENT 管理ビュー

この管理ビューを使用して、現在接続中のデータベースに関する agent 論理データ・グループのアプリケーション・スナップショット情報を取得することができます。

SNAPAGENT 管理ビューを SNAPAGENT_MEMORY_POOL、SNAPAPPL、SNAPAPPL_INFO、 SNAPSTMT、および SNAPSUBSECTION 管理ビューとともに使用すると、GET SNAPSHOT FOR APPLICATIONS ON database-alias CLP コマンドに相当する情報が提供されます。ただし、すべてのデータベース・パーティションからデータを取得します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 639 ページの表 169を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPAGENT 管理ビューに対する SELECT 特権
- SNAPAGENT 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_AGENT 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続中のデータベースに関するすべてのアプリケーション・スナップショット情報を、agent 論理データ・グループから取得します。

```
SELECT * FROM SYSIBMADM.SNAPAGENT
```

以下はこの照会の出力例です。

```
SNAPSHOT_TIMESTAMP         DB_NAME     AGENT_ID        ...
-------------------------- -------...- ---------...--- ...
2005-07-19-11.03.26.740423 SAMPLE                  101 ...
2005-07-19-11.03.26.740423 SAMPLE                   49 ...
                                                       ...
  2 record(s) selected.                                ...
```

この照会からの出力 (続き)。

```
...  AGENT_PID            LOCK_TIMEOUT_VAL     DBPARTITIONNUM
...  -------------------- -------------------- --------------
...                 11980                   -1              0
...                 15940                   -1              0
...
...
```

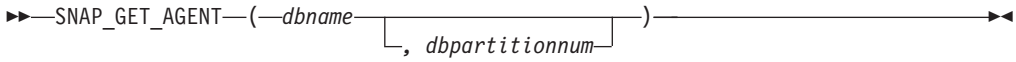
SNAP_GET_AGENT 表関数

SNAP_GET_AGENT 表関数は、SNAPAGENT 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションに関する情報を取得することができます。

SNAP_GET_AGENT 表関数を SNAP_GET_AGENT_MEMORY_POOL、SNAP_GET_APPL_V95、 SNAP_GET_APPL_INFO_V95、 SNAP_GET_STMT、および SNAP_GET_SUBSECTION 表関数とともに使用すると、GET SNAPSHOT FOR ALL APPLICATIONS CLP コマンドに相当する情報が提供されます。ただし、すべてのデータベース・パーティションからデータを取得します。

戻される可能性のある情報の完全なリストは、 639 ページの表 169を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、

空ストリングを指定します。現在接続されているデータベースと同じインスタンス内のすべてのデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプを持つファイルが存在しない場合、SNAP_GET_AGENT 表関数は、現在接続中のデータベースおよびデータベース・パーティション番号のスナップショットを取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_AGENT 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

すべてのアクティブ・データベース内のすべてのアプリケーションに関する、すべてのアプリケーション・スナップショット情報を取得します。

```
SELECT * FROM TABLE(SNAP_GET_AGENT(CAST(NULL AS VARCHAR(128)), -1)) AS T
```

以下はこの照会の出力例です。

```
SNAPSHOT_TIMESTAMP         DB_NAME     AGENT_ID      ...
-------------------------- --------...- --------...-- ...
2006-01-03-17.21.38.530785 SAMPLE                 48 ...
2006-01-03-17.21.38.530785 SAMPLE                 47 ...
2006-01-03-17.21.38.530785 SAMPLE                 46 ...
2006-01-03-17.21.38.530785 TESTDB                 30 ...
```

```
2006-01-03-17.21.38.530785 TESTDB                29 ...
2006-01-03-17.21.38.530785 TESTDB                28 ...

6 record(s) selected.
```

この照会からの出力 (続き)。

```
... AGENT_PID        LOCK_TIMEOUT_VAL     DBPARTITIONNUM
... ---------...---- -------------------- --------------
...             7696                   -1              0
...             8536                   -1              0
...             6672                   -1              0
...             2332                   -1              0
...             8360                   -1              0
...             6736                   -1              0
...
```

戻される情報

表 199. SNAPAGENT 管理ビューおよび SNAP_GET_AGENT 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
DB_NAME	VARCHAR(128)	db_name - データベース名
AGENT_ID	BIGINT	agent_id - アプリケーション・ハンドル (エージェント ID)
AGENT_PID	BIGINT	agent_pid - エンジン・ディスパッチ可能単位 (EDU)
LOCK_TIMEOUT_VAL	BIGINT	lock_timeout_val - ロック・タイムアウト (秒)
DBPARTITIONNUM	SMALLINT	行のデータが検索されたデータベース・パーティション。

SNAPAGENT_MEMORY_POOL 管理ビューおよび SNAP_GET_AGENT_MEMORY_POOL 表関数 - memory_pool 論理データ・グループのスナップショット情報の検索

SNAPAGENT_MEMORY_POOL 管理ビューおよび SNAP_GET_AGENT_MEMORY_POOL 表関数は、エージェント・レベルでのメモリー使用量についての情報を戻します。

SNAPAGENT_MEMORY_POOL 管理ビュー

この管理ビューを使用して、現在接続中のデータベースのエージェント・レベルでのメモリー使用量に関する memory_pool 論理データ・グループのスナップショット情報を取得することができます。

SNAPAGENT_MEMORY_POOL 管理ビューを SNAPAGENT、SNAPAPPL、SNAPAPPL_INFO、SNAPSTMT、および SNAPSUBSECTION 管理ビューとともに使用すると、GET SNAPSHOT FOR APPLICATIONS ON database-alias CLP コマンドに相当する情報が提供されます。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 643 ページの表 170を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPAGENT_MEMORY_POOL 管理ビューに対する SELECT 特権
- SNAPAGENT_MEMORY_POOL 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_AGENT_MEMORY_POOL 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

メモリー・プールおよびそれらの現在のサイズのリストを取得します。

```
SELECT AGENT_ID, POOL_ID, POOL_CUR_SIZE FROM SYSIBMADM.SNAPAGENT_MEMORY_POOL
```

以下はこの照会の出力例です。

```
AGENT_ID       POOL_ID POOL_  CUR_SIZE
--------...--- -------------- ---------...------
            48 APPLICATION                65536
            48 OTHER                      65536
            48 APPL_CONTROL               65536
            47 APPLICATION                65536
            47 OTHER                     131072
            47 APPL_CONTROL               65536
            46 OTHER                     327680
            46 APPLICATION               262144
            46 APPL_CONTROL               65536

9 record(s) selected.
```

SNAP_GET_AGENT_MEMORY_POOL 表関数

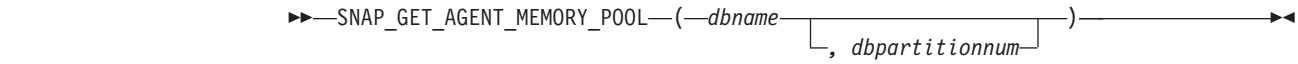
SNAP_GET_AGENT_MEMORY_POOL 表関数は、SNAPAGENT_MEMORY_POOL 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションに関する情報を取得することができます。

SNAP_GET_AGENT_MEMORY_POOL 表関数を SNAP_GET_AGENT、SNAP_GET_APPL_V95、 SNAP_GET_APPL_INFO_V95、 SNAP_GET_STMT、およ

び SNAP_GET_SUBSECTION 表関数とともに使用すると、GET SNAPSHOT FOR ALL APPLICATIONS CLP コマンドに相当する情報が提供されます。

戻される可能性のある情報の完全なリストは、 643 ページの表 170を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、空ストリングを指定します。現在接続されているデータベースと同じインスタンス内のすべてのデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。 *dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプを持つファイルが存在しない場合、SNAP_GET_AGENT_MEMORY_POOL 表関数は、現在接続中のデータベースおよびデータベース・パーティション番号のスナップショットを取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_AGENT_MEMORY_POOL 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

すべてのデータベースのメモリー・プールおよびそれらの現在のサイズのリストを取得します。

```
SELECT SUBSTR(DB_NAME,1,8) AS DB_NAME, AGENT_ID, POOL_ID, POOL_CUR_SIZE
   FROM TABLE(SNAP_GET_AGENT_MEMORY_POOL(CAST (NULL AS VARCHAR(128)), -1))
   AS T
```

以下はこの照会の出力例です。

```
DB_NAME  AGENT_ID       POOL_ID        POOL_CUR_SIZE
-------- --------...--- -------------- --------------------
SAMPLE               48 APPLICATION                   65536
SAMPLE               48 OTHER                         65536
SAMPLE               48 APPL_CONTROL                  65536
SAMPLE               47 APPLICATION                   65536
SAMPLE               47 OTHER                        131072
SAMPLE               47 APPL_CONTROL                  65536
SAMPLE               46 OTHER                        327680
SAMPLE               46 APPLICATION                  262144
SAMPLE               46 APPL_CONTROL                  65536
TESTDB               30 APPLICATION                   65536
TESTDB               30 OTHER                         65536
TESTDB               30 APPL_CONTROL                  65536
TESTDB               29 APPLICATION                   65536
TESTDB               29 OTHER                        131072
TESTDB               29 APPL_CONTROL                  65536
TESTDB               28 OTHER                        327680
TESTDB               28 APPLICATION                   65536
TESTDB               28 APPL_CONTROL                  65536

18 record(s) selected.
```

戻される情報

表 200. SNAPAGENT_MEMORY_POOL 管理ビューおよび SNAP_GET_AGENT_MEMORY_POOL 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
DB_NAME	VARCHAR(128)	db_name - データベース名
AGENT_ID	BIGINT	agent_id - アプリケーション・ハンドル (エージェント ID)
AGENT_PID	BIGINT	agent_pid - エンジン・ディスパッチ可能単位 (EDU)

表 200. SNAPAGENT_MEMORY_POOL 管理ビューおよび SNAP_GET_AGENT_MEMORY_POOL 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
POOL_ID	VARCHAR(14)	pool_id - メモリー・プール ID。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • APP_GROUP • APPL_CONTROL • APPLICATION • BP • CAT_CACHE • DATABASE • DFM • FCMBP • IMPORT_POOL • LOCK_MGR • MONITOR • OTHER • PACKAGE_CACHE • QUERY • SHARED_SORT • SORT • STATEMENT • STATISTICS • UTILITY
POOL_CUR_SIZE	BIGINT	pool_cur_size - メモリー・プールの現行サイズ
POOL_WATERMARK	BIGINT	pool_watermark - メモリー・プール水準点
POOL_CONFIG_SIZE	BIGINT	pool_config_size - メモリー・プールの構成済みサイズ
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

SNAPAPPL_INFO 管理ビューおよび SNAP_GET_APPL_INFO_V95 表関数 - appl_info 論理データ・グループのスナップショット情報の検索

SNAPAPPL_INFO 管理ビューおよび SNAP_GET_APPL_INFO_V95 表関数は、アプリケーション・スナップショットから、特に appl_info 論理データ・グループのアプリケーション情報を戻します。

SNAPAPPL_INFO 管理ビュー

この管理ビューを使用して、現在接続中のデータベースに関する appl_info 論理データ・グループのスナップショット情報を取得することができます。

SNAPAPPL_INFO 管理ビューを SNAPAGENT、 SNAPAGENT_MEMORY_POOL、SNAPAPPL、 SNAPSTMT、および SNAPSUBSECTION 管理ビューとともに使用すると、GET SNAPSHOT FOR APPLICATIONS ON database-alias CLP コマンドに相当する情報が提供されます。ただし、すべてのデータベース・パーティションからデータを取得します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 647 ページの表 171を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPAPPL_INFO 管理ビューに対する SELECT 特権
- SNAPAPPL_INFO 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの権限が必要です。

- SNAP_GET_APPL_INFO_V95 表関数に対する EXECUTE 特権
- DATAACCESS 権限

また、以下のいずれかの権限が必要です。

- SYSMON
- SYSMAINT
- SYSCTRL
- SYSADM

例

現在のデータベースに接続中のアプリケーションの状況を取得します。

```
SELECT AGENT_ID, SUBSTR(APPL_NAME,1,10) AS APPL_NAME, APPL_STATUS
   FROM SYSIBMADM.SNAPAPPL_INFO
```

以下はこの照会の出力例です。

```
AGENT_ID             APPL_NAME  APPL_STATUS
-------------------- ---------- ----------------------
                 101 db2bp.exe  UOWEXEC
                  49 db2bp.exe  CONNECTED

  2 record(s) selected.
```

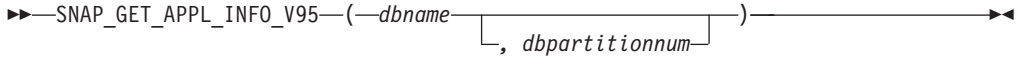
SNAP_GET_APPL_INFO_V95 表関数

SNAP_GET_APPL_INFO_V95 表関数は、SNAPAPPL_INFO 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションに関する情報を取得することができます。

SNAP_GET_APPL_INFO_V95 表関数を SNAP_GET_AGENT、SNAP_GET_AGENT_MEMORY_POOL、 SNAP_GET_APPL_V95、SNAP_GET_STMT、および SNAP_GET_SUBSECTION 表関数とともに使用すると、GET SNAPSHOT FOR ALL APPLICATIONS CLP コマンドに相当する情報が提供されます。ただし、すべてのデータベース・パーティションからデータを取得します。

戻される可能性のある情報の完全なリストは、 647 ページの表 171を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、空ストリングを指定します。現在接続されているデータベースと同じインスタンス内のすべてのデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショ

ット API 要求タイプを持つファイルが存在しない場合、
SNAP_GET_APPL_INFO_V95 表関数は、現在接続中のデータベースおよびデータベース・パーティション番号のスナップショットを取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_APPL_INFO_V95 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

接続中のデータベース・パーティション上のすべてのアプリケーションの状況を取得します。

```
SELECT SUBSTR(DB_NAME,1,8) AS DB_NAME, AGENT_ID,
   SUBSTR(APPL_NAME,1,10) AS APPL_NAME, APPL_STATUS
   FROM TABLE(SNAP_GET_APPL_INFO_V95(CAST(NULL AS VARCHAR(128)),-1)) AS T
```

以下はこの照会の出力例です。

```
DB_NAME  AGENT_ID             APPL_NAME  APPL_STATUS
-------- -------------------- ---------- ----------------------
TOOLSDB                    14 db2bp.exe  CONNECTED
SAMPLE                     15 db2bp.exe  UOWEXEC
SAMPLE                      8 javaw.exe  CONNECTED
SAMPLE                      7 db2bp.exe  UOWWAIT

  4 record(s) selected.
```

以下は、表関数の結果からの SELECT の実行時に入手できる内容について示しています。

```
SELECT SUBSTR(DB_NAME,1,8) AS DB_NAME, AUTHORITY_LVL
   FROM TABLE(SNAP_GET_APPL_INFO_V95(CAST(NULL AS VARCHAR(128)),-1)) AS T
```

以下はこの照会の出力例です。

```
DB_NAME  AUTHORITY_LVL
-------- ------------------------------------------------------....
TESTDB   SYSADM(GROUP) + DBADM(USER) + CREATETAB(USER, GROUP) +
            BINDADD(USER, GROUP) + CONNECT(USER, GROUP) +
            CREATE_NOT_FENC(USER) + IMPLICIT_SCHEMA(USER, GROUP) +
            LOAD(USER) + CREATE_EXT_RT(USER) + QUIESCE_CONN(USER)
TESTDB   SYSADM(GROUP) + DBADM(USER) + CREATETAB(USER, GROUP) +
            BINDADD(USER, GROUP) + CONNECT(USER, GROUP) +
            CREATE_NOT_FENC(USER) + IMPLICIT_SCHEMA(USER, GROUP) +
            LOAD(USER) + CREATE_EXT_RT(USER) + QUIESCE_CONN(USER)
TESTDB   SYSADM(GROUP) + DBADM(USER) + CREATETAB(USER, GROUP) +
            BINDADD(USER, GROUP) + CONNECT(USER, GROUP) +
```

```
CREATE_NOT_FENC(USER) + IMPLICIT_SCHEMA(USER, GROUP) +
LOAD(USER) + CREATE_EXT_RT(USER) + QUIESCE_CONN(USER)

3 record(s) selected.
```

戻される情報

表 201. SNAPAPPL_INFO 管理ビューおよび SNAP_GET_APPL_INFO_V95 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
AGENT_ID	BIGINT	agent_id - アプリケーション・ハンドル (エージェント ID)
APPL_STATUS	VARCHAR(22)	appl_status - アプリケーション状況。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • BACKUP • COMMIT_ACT • COMP • CONNECTED • CONNECTPEND • CREATE_DB • DECOUPLED • DISCONNECTPEND • INTR • IOERROR_WAIT • LOAD • LOCKWAIT • QUIESCE_TABLESPACE • RECOMP • REMOTE_RQST • RESTART • RESTORE • ROLLBACK_ACT • ROLLBACK_TO_SAVEPOINT • TEND • THABRT • THCOMT • TPREP • UNLOAD • UOWEXEC • UOWWAIT • WAITFOR_REMOTE
CODEPAGE_ID	BIGINT	codepage_id - アプリケーションで使用するコード・ページ ID
NUM_ASSOC_AGENTS	BIGINT	num_assoc_agents - 関連したエージェント数

表 201. SNAPAPPL_INFO 管理ビューおよび SNAP_GET_APPL_INFO_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
COORD_NODE_NUM	SMALLINT	coord_node - コーディネーター・ノード
AUTHORITY_LVL	VARCHAR(512)	authority_bitmap - ユーザー許可レベル。 このインターフェースは、sql.h で定義されたデータベース権限およびそれらのソースに基づくテキスト ID を戻し、その形式は次のとおりです。 authority(source, ...) + authority(source, ...) + ... 権限のソースは複数でも構いません。USER、GROUP、または USER と GROUP のいずれかです。 "authority" に使用できる値 • ACCESSCTRL • BINDADD • CONNECT • CREATE_EXT_RT • CREATE_NOT_FENC • CREATETAB • DATAACCESS • DBADM • EXPLAIN • IMPLICIT_SCHEMA • LOAD • LIBADM • QUIESCE_CONN • SECADM • SQLADM • SYSADM • SYSCTRL • SYSMAINT • SYSMON • SYSQUIESCE • WLMADM "source" に使用できる値 • USER - ユーザーに付与された権限、またはそのユーザーに付与されているロールに付与された権限。 • GROUP - ユーザーが属するグループに付与された権限、またはユーザーが属するグループに付与されるロールに付与された権限。
CLIENT_PID	BIGINT	client_pid - クライアント・プロセス ID

表 201. SNAPAPPL_INFO 管理ビューおよび SNAP_GET_APPL_INFO_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
COORD_AGENT_PID	BIGINT	coord_agent_pid - コーディネーター・エージェント
STATUS_CHANGE_TIME	TIMESTAMP	status_change_time - アプリケーション状況変更時刻
CLIENT_PLATFORM	VARCHAR(12)	client_platform - クライアント・オペレーティング・プラットフォーム。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。 • AIX • AIX64 • AS400_DRDA • DOS • DYNIX • HP • HP64 • HPIA • HPIA64 • LINUX • LINUX390 • LINUXIA64 • LINUXPPC • LINUXPPC64 • LINUXX8664 • LINUXZ64 • MAC • MVS_DRDA • NT • NT64 • OS2 • OS390 • SCO • SGI • SNI • SUN • SUN64 • UNKNOWN • UNKNOWN_DRDA • VM_DRDA • VSE_DRDA • WINDOWS

表 201. SNAPAPPL_INFO 管理ビューおよび SNAP_GET_APPL_INFO_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
CLIENT_PROTOCOL	VARCHAR(10)	client_protocol - クライアント通信プロトコル。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。 • CPIC • LOCAL • NETBIOS • NPIPE • TCPIP (DB2 UDB の場合) • TCPIP4 • TCPIP6
TERRITORY_CODE	SMALLINT	territory_code - データベース・テリトリー・コード
APPL_NAME	VARCHAR(256)	appl_name - アプリケーション名
APPL_ID	VARCHAR(128)	appl_id - アプリケーション ID
SEQUENCE_NO	VARCHAR(4)	sequence_no - シーケンス番号
PRIMARY_AUTH_ID	VARCHAR(128)	auth_id - 許可 ID
SESSION_AUTH_ID	VARCHAR(128)	session_auth_id - セッション許可 ID
CLIENT_NNAME	VARCHAR(128)	client_nname モニター・エレメントは使用すべきではありません。返される値は無効な値です。
CLIENT_PRDID	VARCHAR(128)	client_prdid - クライアント製品/バージョン ID
INPUT_DB_ALIAS	VARCHAR(128)	input_db_alias - 入力データベース別名
CLIENT_DB_ALIAS	VARCHAR(128)	client_db_alias - アプリケーションで使用するデータベース別名
DB_NAME	VARCHAR(128)	db_name - データベース名
DB_PATH	VARCHAR(1024)	db_path - データベース・パス
EXECUTION_ID	VARCHAR(128)	execution_id - ユーザー・ログイン ID
CORR_TOKEN	VARCHAR(128)	corr_token - DRDA 相関トークン
TPMON_CLIENT_USERID	VARCHAR(256)	tpmon_client_userid - TP モニター・クライアント・ユーザー ID
TPMON_CLIENT_WKSTN	VARCHAR(256)	tpmon_client_wkstn - TP モニター・クライアント・ワークステーション名
TPMON_CLIENT_APP	VARCHAR(256)	tpmon_client_app - TP モニター・クライアント・アプリケーション名
TPMON_ACC_STR	VARCHAR(200)	tpmon_acc_str - TP モニター・クライアント・アカウンティング・ストリング
DBPARTITIONNUM	SMALLINT	行のデータが検索されたデータベース・パーティション。
WORKLOAD_ID	INTEGER	現行のワークロード ID.

表 201. SNAPAPPL_INFO 管理ビューおよび SNAP_GET_APPL_INFO_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
IS_SYSTEM_APPL	SMALLINT	IS_SYSTEM_APPL の値は、アプリケーションが DB2 内部システム・アプリケーションかどうかを示します。 0 はユーザー・アプリケーションであることを示します。 1 はシステム・アプリケーションであることを示します。 DB2 システム・アプリケーションの例は DB2 イベント・モニターです。 一般に、DB2 システム・アプリケーションの名前は "db2" で始まります。例えば、db2stmm, db2taskd などです。

SNAPAPPL 管理ビューおよび SNAP_GET_APPL_V95 表関数 - appl 論理データ・グループのスナップショット情報の検索

652 ページの『SNAPAPPL 管理ビュー』および 653 ページの『SNAP_GET_APPL_V95 表関数』は、アプリケーション・スナップショットからアプリケーションに関する情報 (特に appl 論理データ・グループ) を戻します。

SNAPAPPL 管理ビュー

この管理ビューを使用して、現在接続中のデータベースに関する appl 論理データ・グループのスナップショット情報を取得することができます。

SNAPAPPL 管理ビューを SNAPAGENT、 SNAPAGENT_MEMORY_POOL、 SNAPAPPL_INFO、 SNAPSTMT、および SNAPSUBSECTION 管理ビューとともに使用すると、GET SNAPSHOT FOR APPLICATIONS ON database-alias CLP コマンドに相当する情報が提供されます。ただし、すべてのデータベース・パーティションからデータを取得します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 655 ページの表 172を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPAPPL 管理ビューに対する SELECT 特権
- SNAPAPPL 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_APPL_V95 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

接続中のデータベース内の各アプリケーションについて読み取りおよび書き込みが行われた行の詳細を取得します。

```
SELECT SUBSTR(DB_NAME,1,8) AS DB_NAME, AGENT_ID, ROWS_READ, ROWS_WRITTEN
   FROM SYSIBMADM.SNAPAPPL
```

以下はこの照会の出力例です。

```
DB_NAME  AGENT_ID             ROWS_READ            ROWS_WRITTEN
-------- -------------------- -------------------- --------------------
SAMPLE                     7                   25                    0

  1 record(s) selected.
```

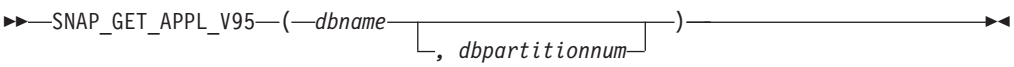
SNAP_GET_APPL_V95 表関数

SNAP_GET_APPL_V95 表関数は、SNAPAPPL 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションに関する情報を取得することができます。

SNAP_GET_APPL_V95 表関数を SNAP_GET_AGENT、SNAP_GET_AGENT_MEMORY_POOL、 SNAP_GET_APPL_INFO_V95、SNAP_GET_STMT、および SNAP_GET_SUBSECTION 表関数とともに使用すると、GET SNAPSHOT FOR ALL APPLICATIONS CLP コマンドに相当する情報が提供されます。ただし、すべてのデータベース・パーティションからデータを取得します。

戻される可能性のある情報の完全なリストは、 655 ページの表 172を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、空ストリングを指定します。現在接続されているデータベースと同じインスタンス内のすべてのデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプを持つファイルが存在しない場合、SNAP_GET_APPL_V95 表関数は、現在接続中のデータベースおよびデータベース・パーティション番号のスナップショットを取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_APPL_V95 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

すべてのアクティブ・データベースの各アプリケーションについて読み取りおよび書き込みが行われた行の詳細を取得します。

```
SELECT SUBSTR(DB_NAME,1,8) AS DB_NAME, AGENT_ID, ROWS_READ, ROWS_WRITTEN
   FROM TABLE (SNAP_GET_APPL_V95(CAST(NULL AS VARCHAR(128)),-1)) AS T
```

以下はこの照会の出力例です。

```
DB_NAME  AGENT_ID       ROWS_READ      ROWS_WRITTEN
-------- --------...--- ---------...-- ------------...-
WSDB                679              0                 0
WSDB                461              3                 0
WSDB                460              4                 0
TEST                680              4                 0
TEST                455              6                 0
TEST                454              0                 0
TEST                453             50                 0
```

戻される情報

表 202. SNAPAPPL 管理ビューおよび SNAP_GET_APPL_V95 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
DB_NAME	VARCHAR(128)	db_name - データベース名
AGENT_ID	BIGINT	agent_id - アプリケーション・ハンドル (エージェント ID)
UOW_LOG_SPACE_USED	BIGINT	uow_log_space_used - 作業単位ログ・スペース
ROWS_READ	BIGINT	rows_read - 読み取り行数
ROWS_WRITTEN	BIGINT	rows_written - 書き込み行数
INACT_STMTHIST_SZ	BIGINT	stmt_history_list_size - ステートメント履歴リストのサイズ
POOL_DATA_L_READS	BIGINT	pool_data_l_reads - バッファー・プール・データの論理読み取り
POOL_DATA_P_READS	BIGINT	pool_data_p_reads - バッファー・プール・データの物理読み取り
POOL_DATA_WRITES	BIGINT	pool_data_writes - バッファー・プールへのデータの書き込み
POOL_INDEX_L_READS	BIGINT	pool_index_l_reads - バッファー・プール索引の論理読み取り
POOL_INDEX_P_READS	BIGINT	pool_index_p_reads - バッファー・プール索引の物理読み取り
POOL_INDEX_WRITES	BIGINT	pool_index_writes - バッファー・プール索引の書き込み
POOL_TEMP_DATA_L_READS	BIGINT	pool_temp_data_l_reads - バッファー・プール一時データの論理読み取り
POOL_TEMP_DATA_P_READS	BIGINT	pool_temp_data_p_reads - バッファー・プール一時データの物理読み取り
POOL_TEMP_INDEX_L_READS	BIGINT	pool_temp_index_l_reads - バッファー・プール一時索引の論理読み取り
POOL_TEMP_INDEX_P_READS	BIGINT	pool_temp_index_p_reads - バッファー・プール一時索引の物理読み取り

表 202. SNAPAPPL 管理ビューおよび SNAP_GET_APPL_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
POOL_TEMP_XDA_L_READS	BIGINT	pool_temp_xda_l_reads - バッファー・プール一時 XDA データの論理読み取り
POOL_TEMP_XDA_P_READS	BIGINT	pool_temp_xda_p_reads - バッファー・プール一時 XDA データの物理読み取り : モニター・エレメント
POOL_XDA_L_READS	BIGINT	pool_xda_l_reads - バッファー・プール XDA データの論理読み取り
POOL_XDA_P_READS	BIGINT	pool_xda_p_reads - バッファー・プール XDA データの物理読み取り
POOL_XDA_WRITES	BIGINT	pool_xda_writes - バッファー・プール XDA データの書き込み
POOL_READ_TIME	BIGINT	pool_read_time - バッファー・プール物理読み取り時間の合計
POOL_WRITE_TIME	BIGINT	pool_write_time - バッファー・プール物理書き込み時間の合計
DIRECT_READS	BIGINT	direct_reads - データベースからの直接読み取り
DIRECT_WRITES	BIGINT	direct_writes - データベースへの直接書き込み
DIRECT_READ_REQS	BIGINT	direct_read_reqs - 直接読み取り要求
DIRECT_WRITE_REQS	BIGINT	direct_write_reqs - 直接書き込み要求
DIRECT_READ_TIME	BIGINT	direct_read_time - 直接読み取り時間
DIRECT_WRITE_TIME	BIGINT	direct_write_time - 直接書き込み時間
UNREAD_PREFETCH_PAGES	BIGINT	unread_prefetch_pages - 読み取り不能プリフェッチ・ページ
LOCKS_HELD	BIGINT	locks_held - ロック保持数
LOCK_WAITS	BIGINT	lock_waits - ロック待機数
LOCK_WAIT_TIME	BIGINT	lock_wait_time - ロック待機中の時間
LOCK_ESCALS	BIGINT	lock_escals - ロック・エスカレーション数
X_LOCK_ESCALS	BIGINT	x_lock_escals - 排他ロック・エスカレーション数
DEADLOCKS	BIGINT	deadlocks - デッドロック検出数
TOTAL_SORTS	BIGINT	total_sorts - ソート合計
TOTAL_SORT_TIME	BIGINT	total_sort_time - ソート時間合計

表 202. SNAPAPPL 管理ビューおよび SNAP_GET_APPL_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
SORT_OVERFLOWS	BIGINT	sort_overflows - ソート・オーバーフロー
COMMIT_SQL_STMTS	BIGINT	commit_sql_stmts - 試行されたコミット・ステートメント
ROLLBACK_SQL_STMTS	BIGINT	rollback_sql_stmts - 試行されたロールバック・ステートメント
DYNAMIC_SQL_STMTS	BIGINT	dynamic_sql_stmts - 試行された動的 SQL ステートメント
STATIC_SQL_STMTS	BIGINT	static_sql_stmts - 試行された静的 SQL ステートメント
FAILED_SQL_STMTS	BIGINT	failed_sql_stmts - 失敗したステートメント操作
SELECT_SQL_STMTS	BIGINT	select_sql_stmts - 実行された選択 SQL ステートメント
DDL_SQL_STMTS	BIGINT	ddl_sql_stmts - データ定義言語 (DDL) SQL ステートメント
UID_SQL_STMTS	BIGINT	uid_sql_stmts - 実行された更新/挿入/削除 SQL ステートメント
INT_AUTO_REBINDS	BIGINT	int_auto_rebinds - 内部自動再バインド
INT_ROWS_DELETED	BIGINT	int_rows_deleted - 削除された内部行数
INT_ROWS_UPDATED	BIGINT	int_rows_updated - 更新された内部行数
INT_COMMITS	BIGINT	int_commits - 内部コミット数
INT_ROLLBACKS	BIGINT	int_rollbacks - 内部ロールバック数
INT_DEADLOCK_ROLLBACKS	BIGINT	int_deadlock_rollbacks - デッドロックによる内部ロールバック数
ROWS_DELETED	BIGINT	rows_deleted - 削除行数
ROWS_INSERTED	BIGINT	rows_inserted - 挿入行数
ROWS_UPDATED	BIGINT	rows_updated - 更新行数
ROWS_SELECTED	BIGINT	rows_selected - 選択行数
BINDS_PRECOMPILES	BIGINT	binds_precompiles - 試行されたバインド/プリコンパイル
OPEN_REM_CURS	BIGINT	open_rem_curs - 開かれているリモート・カーソル
OPEN_REM_CURS_BLK	BIGINT	open_rem_curs_blk - 開かれているリモート・ブロック・カーソル
REJ_CURS_BLK	BIGINT	rej_curs_blk - リジェクトされたブロック・カーソル要求
ACC_CURS_BLK	BIGINT	acc_curs_blk - 受け入れられたブロック・カーソル要求

表 202. SNAPAPPL 管理ビューおよび SNAP_GET_APPL_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
SQL_REQS_SINCE_COMMIT	BIGINT	sql_reqs_since_commit - 最終コミット後の SQL 要求数
LOCK_TIMEOUTS	BIGINT	lock_timeouts - ロック・タイムアウト数
INT_ROWS_INSERTED	BIGINT	int_rows_inserted - 挿入された内部行数
OPEN_LOC_CURS	BIGINT	open_loc_curs - 開かれているローカル・カーソル
OPEN_LOC_CURS_BLK	BIGINT	open_loc_curs_blk - 開かれているローカル・ブロック・カーソル
PKG_CACHE_LOOKUPS	BIGINT	pkg_cache_lookups - パッケージ・キャッシュ参照
PKG_CACHE_INSERTS	BIGINT	pkg_cache_inserts - パッケージ・キャッシュ挿入
CAT_CACHE_LOOKUPS	BIGINT	cat_cache_lookups - カタログ・キャッシュ参照数
CAT_CACHE_INSERTS	BIGINT	cat_cache_inserts - カタログ・キャッシュ挿入数
CAT_CACHE_OVERFLOWS	BIGINT	cat_cache_overflows - カタログ・キャッシュ・オーバーフロー数
NUM_AGENTS	BIGINT	num_agents - ステートメントで作動しているエージェントの数
AGENTS_STOLEN	BIGINT	agents_stolen - スチールされたエージェント
ASSOCIATED_AGENTS_TOP	BIGINT	associated_agents_top - 関連エージェント最大数
APPL_PRIORITY	BIGINT	appl_priority - アプリケーション・エージェント優先順位
APPL_PRIORITY_TYPE	VARCHAR(16)	appl_priority_type - アプリケーション優先順位タイプ。このインターフェースは、`sqlmon.h` 内の定義に基づいてテキスト ID を戻します。それは、次のうちの 1 つです。 • DYNAMIC_PRIORITY • FIXED_PRIORITY
PREFETCH_WAIT_TIME	BIGINT	prefetch_wait_time - プリフェッチ待ち時間
APPL_SECTION_LOOKUPS	BIGINT	appl_section_lookups - セクションの参照回数
APPL_SECTION_INSERTS	BIGINT	appl_section_inserts - セクション挿入数

表 202. SNAPAPPL 管理ビューおよび SNAP_GET_APPL_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
LOCKS_WAITING	BIGINT	locks_waiting - ロックで待機中の現行エージェント
TOTAL_HASH_JOINS	BIGINT	total_hash_joins - ハッシュ結合の合計
TOTAL_HASH_LOOPS	BIGINT	total_hash_loops - ハッシュ・ループの合計
HASH_JOIN_OVERFLOWS	BIGINT	hash_join_overflows - ハッシュ結合のオーバーフロー
HASH_JOIN_SMALL_ OVERFLOWS	BIGINT	hash_join_small_overflows - ハッシュ結合の短精度オーバーフロー
APPL_IDLE_TIME	BIGINT	appl_idle_time - アプリケーション・アイドル時間
UOW_LOCK_WAIT_TIME	BIGINT	uow_lock_wait_time - ロック待機中の作業単位の合計時間
UOW_COMP_STATUS	VARCHAR(14)	uow_comp_status - 作業単位完了状況。このインターフェースは、sqlmon.h 内の定義に基づいてテキスト ID を戻します。それは、次のうちの 1 つです。 • APPL_END • UOWABEND • UOWCOMMIT • UOWDEADLOCK • UOWLOCKTIMEOUT • UOWROLLBACK • UOWUNKNOWN
AGENT_USR_CPU_TIME_S	BIGINT	agent_usr_cpu_time - エージェントが使用したユーザー CPU 時間 (秒単位)*
AGENT_USR_CPU_TIME_MS	BIGINT	agent_usr_cpu_time - エージェントが使用したユーザー CPU 時間 (小数部、マイクロ秒単位)*
AGENT_SYS_CPU_TIME_S	BIGINT	agent_sys_cpu_time - エージェントが使用したシステム CPU 時間 (秒単位)*
AGENT_SYS_CPU_TIME_MS	BIGINT	agent_sys_cpu_time - エージェントが使用したシステム CPU 時間 (小数部、マイクロ秒単位)*
APPL_CON_TIME	TIMESTAMP	appl_con_time - 接続要求開始タイム・スタンプ

表 202. SNAPAPPL 管理ビューおよび SNAP_GET_APPL_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
CONN_COMPLETE_TIME	TIMESTAMP	conn_complete_time - 接続要求完了タイム・スタンプ
LAST_RESET	TIMESTAMP	last_reset - 最後のリセット・タイム・スタンプ
UOW_START_TIME	TIMESTAMP	uow_start_time - 作業単位開始タイム・スタンプ
UOW_STOP_TIME	TIMESTAMP	uow_stop_time - 作業単位停止タイム・スタンプ
PREV_UOW_STOP_TIME	TIMESTAMP	prev_uow_stop_time - 直前の作業単位完了タイム・スタンプ
UOW_ELAPSED_TIME_S	BIGINT	uow_elapsed_time - 最新の作業単位の経過時間 (秒単位)*
UOW_ELAPSED_TIME_MS	BIGINT	uow_elapsed_time - 最新の作業単位の経過時間 (小数部、マイクロ秒単位)*
ELAPSED_EXEC_TIME_S	BIGINT	elapsed_exec_time - ステートメント実行経過時間 (秒単位)*
ELAPSED_EXEC_TIME_MS	BIGINT	elapsed_exec_time - ステートメント実行経過時間 (小数部、マイクロ秒単位)*
INBOUND_COMM_ADDRESS	VARCHAR(32)	inbound_comm_address - インバウンド通信アドレス
LOCK_TIMEOUT_VAL	BIGINT	lock_timeout_val - ロック・タイムアウト (秒)
PRIV_WORKSPACE_NUM_OVERFLOWS	BIGINT	priv_workspace_num_overflows - 専用ワークスペースのオーバーフロー回数
PRIV_WORKSPACE_SECTION_INSERTS	BIGINT	priv_workspace_section_inserts - 専用ワークスペース・セクション挿入
PRIV_WORKSPACE_SECTION_LOOKUPS	BIGINT	priv_workspace_section_lookups - 専用ワークスペース・セクションの参照
PRIV_WORKSPACE_SIZE_TOP	BIGINT	priv_workspace_size_top - 専用ワークスペースの最大サイズ
SHR_WORKSPACE_NUM_OVERFLOWS	BIGINT	shr_workspace_num_overflows - 共有ワークスペースのオーバーフロー回数
SHR_WORKSPACE_SECTION_INSERTS	BIGINT	shr_workspace_section_inserts - 共有ワークスペース・セクション挿入数

表 202. SNAPAPPL 管理ビューおよび SNAP_GET_APPL_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
SHR_WORKSPACE_SECTION_LOOKUPS	BIGINT	shr_workspace_section_lookups - 共有ワークスペース・セクションの参照回数
SHR_WORKSPACE_SIZE_TOP	BIGINT	shr_workspace_size_top - 最大共有ワークスペース・サイズ
DBPARTITIONNUM	SMALLINT	行のデータが検索されたデータベース・パーティション。
CAT_CACHE_SIZE_TOP	BIGINT	cat_cache_size_top - カタログ・キャッシュ最高水準点
TOTAL_OLAP_FUNCS	BIGINT	実行される OLAP 関数の合計数。
OLAP_FUNC_OVERFLOWS	BIGINT	OLAP 関数データが使用可能なソート・ヒープ・スペースを超えた回数。
* この列の元になるモニター・エレメントの合計消費時間を計算するには、このモニター・エレメントに関する _S で終わっている列で報告されている整数秒と、このモニター・エレメントに関する _MS で終わっている列に報告されている小数秒を、次の式を使用して加算しなければなりません: (*monitor-element-name*_S × 1,000,000 + *monitor-element-name*_MS) ÷ 1,000,000. 例えば、(ELAPSED_EXEC_TIME_S × 1,000,000 + ELAPSED_EXEC_TIME_MS) ÷ 1,000,000。		

SNAPBP 管理ビューおよび SNAP_GET_BP_V95 表関数 - bufferpool 論理グループのスナップショット情報の検索

SNAPBP 管理ビューおよび SNAP_GET_BP_V95 表関数は、bufferpool スナップショットから、特に bufferpool 論理データ・グループのバッファー・プール情報を戻します。

SNAPBP 管理ビュー

この管理ビューを使用して、現在接続中のデータベースに関する bufferpool 論理グループのスナップショット情報を取得することができます。

SNAPBP 管理ビューを SNAPBP_PART 管理ビューとともに使用すると、GET SNAPSHOT FOR BUFFERPOOLS ON database-alias CLP コマンドに相当するデータが提供されます。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 664 ページの表 173を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPBP 管理ビューに対する SELECT 特権
- SNAPBP 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_BP_V95 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続中のデータベースのすべてのバッファー・プールについて、データおよび索引の書き込みを取得します。

```
SELECT SUBSTR(DB_NAME,1,8) AS DB_NAME,SUBSTR(BP_NAME,1,15)
   AS BP_NAME,POOL_DATA_WRITES,POOL_INDEX_WRITES
   FROM SYSIBMADM.SNAPBP
```

以下はこの照会の出力例です。

```
DB_NAME  BP_NAME         POOL_DATA_WRITES     POOL_INDEX_WRITES
-------- --------------- -------------------- --------------------
TEST     IBMDEFAULTBP                       0                    0
TEST     IBMSYSTEMBP4K                      0                    0
TEST     IBMSYSTEMBP8K                      0                    0
TEST     IBMSYSTEMBP16K                     0                    0
TEST     IBMSYSTEMBP32K                     0                    0

5 record(s) selected
```

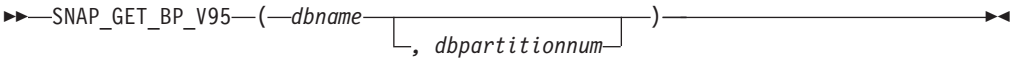
SNAP_GET_BP_V95 表関数

SNAP_GET_BP_V95 表関数は SNAPBP 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションに関する情報を取得することができます。

SNAP_GET_BP_V95 表関数を SNAP_GET_BP_PART 表関数とともに使用すると、GET SNAPSHOT FOR ALL BUFFERPOOLS CLP コマンドに相当するデータが提供されます。

戻される可能性のある情報の完全なリストは、664 ページの表 173を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、空ストリングを指定します。現在接続されているデータベースと同じインスタンス内のすべてのデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプを持つファイルが存在しない場合、SNAP_GET_BP_V95 表関数は、現在接続中のデータベースおよびデータベース・パーティション番号のスナップショットを取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_BP_V95 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT

- SYSADM

例

現在接続されているデータベース・パーティションのすべてのアクティブ・データベースのすべてのバッファー・プールについて、物理および論理読み取りの合計を取得します。

```
SELECT SUBSTR(T.DB_NAME,1,10) AS DB_NAME,
   SUBSTR(T.BP_NAME,1,20) AS BP_NAME,
   (T.POOL_DATA_L_READS+T.POOL_INDEX_L_READS) AS TOTAL_LOGICAL_READS,
   (T.POOL_DATA_P_READS+T.POOL_INDEX_P_READS) AS TOTAL_PHYSICAL_READS,
   T.DBPARTITIONNUM
   FROM TABLE(SNAP_GET_BP_V95(CAST(NULL AS VARCHAR(128)), -1)) AS T
```

以下はこの照会の出力例です。

```
DB_NAME    BP_NAME          TOTAL_LOGICAL_READS  ...
---------- ------------...- -------------------- ...
SAMPLE     IBMDEFAULTBP                       0 ...
TOOLSDB    IBMDEFAULTBP                       0 ...
TOOLSDB    BP32K0000                          0 ...

  3 record(s) selected.
```

この照会からの出力 (続き)。

```
... TOTAL_PHYSICAL_READS DBPARTITIONNUM
... -------------------- --------------
...                    0              0
...                    0              0
...                    0              0
```

戻される情報

表 203. SNAPBP 管理ビューおよび SNAP_GET_BP_V95 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
BP_NAME	VARCHAR(128)	bp_name - バッファー・プール名
DB_NAME	VARCHAR(128)	db_name - データベース名
DB_PATH	VARCHAR(1024)	db_path - データベース・パス
INPUT_DB_ALIAS	VARCHAR(128)	input_db_alias - 入力データベース別名
POOL_DATA_L_READS	BIGINT	pool_data_l_reads - バッファー・プール・データの論理読み取り
POOL_DATA_P_READS	BIGINT	pool_data_p_reads - バッファー・プール・データの物理読み取り
POOL_DATA_WRITES	BIGINT	pool_data_writes - バッファー・プールへのデータの書き込み
POOL_INDEX_L_READS	BIGINT	pool_index_l_reads - バッファー・プール索引の論理読み取り
POOL_INDEX_P_READS	BIGINT	pool_index_p_reads - バッファー・プール索引の物理読み取り

表 203. SNAPBP 管理ビューおよび SNAP_GET_BP_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
POOL_INDEX_WRITES	BIGINT	pool_index_writes - バッファー・プール索引の書き込み
POOL_XDA_L_READS	BIGINT	pool_xda_l_reads - バッファー・プール XDA データの論理読み取り
POOL_XDA_P_READS	BIGINT	pool_xda_p_reads - バッファー・プール XDA データの物理読み取り
POOL_XDA_WRITES	BIGINT	pool_xda_writes - バッファー・プール XDA データの書き込み
POOL_READ_TIME	BIGINT	pool_read_time - バッファー・プール物理読み取り時間の合計
POOL_WRITE_TIME	BIGINT	pool_write_time - バッファー・プール物理書き込み時間の合計
POOL_ASYNC_DATA_READS	BIGINT	pool_async_data_reads - バッファー・プール非同期データ読み取り
POOL_ASYNC_DATA_WRITES	BIGINT	pool_async_data_writes - バッファー・プール非同期データ書き込み
POOL_ASYNC_INDEX_READS	BIGINT	pool_async_index_reads - バッファー・プール非同期索引読み取り
POOL_ASYNC_INDEX_WRITES	BIGINT	pool_async_index_writes - バッファー・プール非同期索引書き込み
POOL_ASYNC_XDA_READS	BIGINT	pool_async_xda_reads - バッファー・プール非同期 XDA データ読み取り
POOL_ASYNC_XDA_WRITES	BIGINT	pool_async_xda_writes - バッファー・プール非同期 XDA データ書き込み
POOL_ASYNC_READ_TIME	BIGINT	pool_async_read_time - バッファー・プール非同期読み取り時間
POOL_ASYNC_WRITE_TIME	BIGINT	pool_async_write_time - バッファー・プール非同期書き込み時間
POOL_ASYNC_DATA_READ_REQS	BIGINT	pool_async_data_read_reqs - バッファー・プール非同期読み取り要求
POOL_ASYNC_INDEX_READ_REQS	BIGINT	pool_async_index_read_reqs - バッファー・プール非同期索引読み取り要求
POOL_ASYNC_XDA_READ_REQS	BIGINT	pool_async_xda_read_reqs - バッファー・プール非同期 XDA 読み取り要求
DIRECT_READS	BIGINT	direct_reads - データベースからの直接読み取り
DIRECT_WRITES	BIGINT	direct_writes - データベースへの直接書き込み
DIRECT_READ_REQS	BIGINT	direct_read_reqs - 直接読み取り要求

表 203. SNAPBP 管理ビューおよび SNAP_GET_BP_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
DIRECT_WRITE_REQS	BIGINT	direct_write_reqs - 直接書き込み要求
DIRECT_READ_TIME	BIGINT	direct_read_time - 直接読み取り時間
DIRECT_WRITE_TIME	BIGINT	direct_write_time - 直接書き込み時間
UNREAD_PREFETCH_PAGES	BIGINT	unread_prefetch_pages - 読み取り不能プリフェッチ・ページ
FILES_CLOSED	BIGINT	files_closed - 閉じられたデータベース・ファイル
POOL_TEMP_DATA_L_READS	BIGINT	pool_temp_data_l_reads - バッファー・プール一時データの論理読み取り
POOL_TEMP_DATA_P_READS	BIGINT	pool_temp_data_p_reads - バッファー・プール一時データの物理読み取り
POOL_TEMP_INDEX_L_READS	BIGINT	pool_temp_index_l_reads - バッファー・プール一時索引の論理読み取り
POOL_TEMP_INDEX_P_READS	BIGINT	pool_temp_index_p_reads - バッファー・プール一時索引の物理読み取り
POOL_TEMP_XDA_L_READS	BIGINT	pool_temp_xda_l_reads - バッファー・プール一時 XDA データの論理読み取り
POOL_TEMP_XDA_P_READS	BIGINT	pool_temp_xda_p_reads - バッファー・プール一時 XDA データの物理読み取り : モニター・エレメント
POOL_NO_VICTIM_BUFFER	BIGINT	pool_no_victim_buffer - バッファー・プールの非ビクティム・バッファー数
PAGES_FROM_BLOCK_IOS	BIGINT	pages_from_block_ios - ブロック入出力によって読み取られたページ数の合計
PAGES_FROM_VECTORED_IOS	BIGINT	pages_from_vectored_ios - ベクトル化入出力によって読み取られたページ数の合計
VECTORED_IOS	BIGINT	vectored_ios - ベクトル化入出力要求数
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

SNAPBP_PART 管理ビューおよび SNAP_GET_BP_PART 表関数 - bufferpool_nodeinfo 論理データ・グループのスナップショット情報の検索

SNAPBP_PART 管理ビューおよび SNAP_GET_BP_PART 表関数は、バッファー・プール・スナップショットからバッファー・プール情報について、特に bufferpool_nodeinfo 論理データ・グループのバッファー・プール情報を戻します。

SNAPBP_PART 管理ビュー

この管理ビューを使用して、現在接続中のデータベースに関する bufferpool_nodeinfo 論理データ・グループのスナップショット情報を取得することができます。

SNAPBP_PART 管理ビューを SNAPBP 管理ビューとともに使用すると、GET SNAPSHOT FOR BUFFERPOOLS ON database-alias CLP コマンドに相当するデータが提供されます。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、670 ページの表 174を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPBP_PART 管理ビューに対する SELECT 特権
- SNAPBP_PART 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_BP_PART 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

SAMPLE データベースへの接続中に、すべてのバッファー・プールのデータを取得します。

```
SELECT SUBSTR(DB_NAME,1,8) AS DB_NAME, SUBSTR(BP_NAME,1,15) AS BP_NAME,
   BP_CUR_BUFFSZ, BP_NEW_BUFFSZ, BP_PAGES_LEFT_TO_REMOVE, BP_TBSP_USE_COUNT
   FROM SYSIBMADM.SNAPBP_PART
```

以下はこの照会の出力例です。

```
DB_NAME  BP_NAME         BP_CUR_BUFFSZ        BP_NEW_BUFFSZ        ...
-------- --------------- -------------------- -------------------- ...
SAMPLE   IBMDEFAULTBP                   1000                 1000 ...
SAMPLE   IBMSYSTEMBP4K                    16                   16 ...
SAMPLE   IBMSYSTEMBP8K                    16                   16 ...
SAMPLE   IBMSYSTEMBP16K                   16                   16 ...
                                                                  ...
  4 record(s) selected.
```

この照会からの出力 (続き)。

```
... BP_PAGES_LEFT_TO_REMOVE BP_TBSP_USE_COUNT
... ----------------------- --------------------
...                       0                    3
...                       0                    0
...                       0                    0
...                       0                    0
...
```

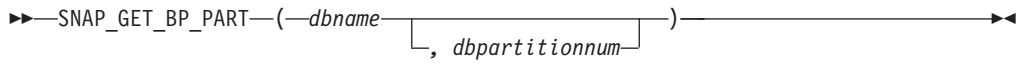
SNAP_GET_BP_PART 表関数

SNAP_GET_BP_PART 表関数は、SNAPBP_PART 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションに関する情報を取得することができます。

SNAP_GET_BP_PART 表関数を SNAP_GET_BP_V95 表関数とともに使用すると、GET SNAPSHOT FOR ALL BUFFERPOOLS CLP コマンドに相当するデータが提供されます。

戻される可能性のある情報の完全なリストは、 670 ページの表 174を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、空ストリングを指定します。現在接続されているデータベースと同じインスタンス内のすべてのデータベースにある、すべてのバッファー・プールのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。

dbname が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプを持つファイルが存在しない場合、SNAP_GET_BP_PART 表関数は、現在接続中のデータベースおよびデータベース・パーティション番号のスナップショットを取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_BP_PART 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

SAMPLE データベースへの接続中に、すべてのアクティブ・データベースのすべてのバッファー・プールのデータを取得します。

```
SELECT SUBSTR(DB_NAME,1,8) AS DB_NAME, SUBSTR(BP_NAME,1,15) AS BP_NAME,
   BP_CUR_BUFFSZ, BP_NEW_BUFFSZ, BP_PAGES_LEFT_TO_REMOVE, BP_TBSP_USE_COUNT
   FROM TABLE(SNAP_GET_BP_PART(CAST(NULL AS VARCHAR(128)),-1)) AS T
```

以下はこの照会の出力例です。

```
DB_NAME  BP_NAME         BP_CUR_BUFFSZ        BP_NEW_BUFFSZ        ...
-------- --------------- -------------------- -------------------- ...
SAMPLE   IBMDEFAULTBP                    250                  250 ...
SAMPLE   IBMSYSTEMBP4K                    16                   16 ...
SAMPLE   IBMSYSTEMBP8K                    16                   16 ...
SAMPLE   IBMSYSTEMBP16K                   16                   16 ...
SAMPLE   IBMSYSTEMBP32K                   16                   16 ...
TESTDB   IBMDEFAULTBP                    250                  250 ...
TESTDB   IBMSYSTEMBP4K                    16                   16 ...
TESTDB   IBMSYSTEMBP8K                    16                   16 ...
TESTDB   IBMSYSTEMBP16K                   16                   16 ...
TESTDB   IBMSYSTEMBP32K                   16                   16 ...

...
```

この照会からの出力 (続き)。

```
... BP_PAGES_LEFT_TO_REMOVE BP_TBSP_USE_COUNT
... ---------------------- --------------------
...                      0                    3
...                      0                    0
...                      0                    0
...                      0                    0
...                      0                    0
...                      0                    3
...                      0                    0
...                      0                    0
...                      0                    0
...                      0                    0

...
```

戻される情報

表 204. SNAPBP_PART 管理ビューおよび SNAP_GET_BP_PART 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
BP_NAME	VARCHAR(128)	bp_name - バッファー・プール名
DB_NAME	VARCHAR(128)	db_name - データベース名
BP_CUR_BUFFSZ	BIGINT	bp_cur_buffsz - バッファー・プールの現行サイズ
BP_NEW_BUFFSZ	BIGINT	bp_new_buffsz - 新規バッファー・プール・サイズ
BP_PAGES_LEFT_TO_REMOVE	BIGINT	bp_pages_left_to_remove - 除去残ページ数
BP_TBSP_USE_COUNT	BIGINT	bp_tbsp_use_count - バッファー・プールにマップされている表スペースの数
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

SNAPCONTAINER 管理ビューおよび SNAP_GET_CONTAINER_V91 表関数 - tablespace_container 論理データ・グループ・スナップショット情報の検索

SNAPCONTAINER 管理ビューおよび SNAP_GET_CONTAINER_V91 表関数は、tablespace_container 論理データ・グループからの表スペース・スナップショット情報を戻します。

SNAPCONTAINER 管理ビュー

この管理ビューでは、現在接続されているデータベースの tablespace_container 論理データ・グループ・スナップショット情報を検索できます。

SNAPTBSP、SNAPTBSP_PART、SNAPTBSP_QUIESCER、および SNAPTBSP_RANGE 管理ビューと共に使用すると、SNAPCONTAINER 管理ビューは、GET SNAPSHOT FOR TABLESPACES ON database-alias CLP コマンドと同等のデータを戻します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 674 ページの表 175を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPCONTAINER 管理ビューに対する SELECT 特権
- SNAPCONTAINER 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_CONTAINER_V91 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベースのすべてのデータベース・パーティションの表スペース・コンテナーの詳細を検索します。

```
SELECT SNAPSHOT_TIMESTAMP, SUBSTR(TBSP_NAME, 1, 15) AS TBSP_NAME,
   TBSP_ID, SUBSTR(CONTAINER_NAME, 1, 20) AS CONTAINER_NAME,
   CONTAINER_ID, CONTAINER_TYPE, ACCESSIBLE, DBPARTITIONNUM
   FROM SYSIBMADM.SNAPCONTAINER ORDER BY DBPARTITIONNUM
```

以下はこの照会の出力例です。

```
SNAPSHOT_TIMESTAMP         TBSP_NAME       TBSP_ID     ...
-------------------------- --------------- -------...- ...
2006-01-08-16.49.24.639945 SYSCATSPACE               0 ...
2006-01-08-16.49.24.639945 TEMPSPACE1                1 ...
2006-01-08-16.49.24.639945 USERSPACE1                2 ...
2006-01-08-16.49.24.639945 SYSTOOLSPACE              3 ...
2006-01-08-16.49.24.640747 TEMPSPACE1                1 ...
2006-01-08-16.49.24.640747 USERSPACE1                2 ...
2006-01-08-16.49.24.639981 TEMPSPACE1                1 ...
2006-01-08-16.49.24.639981 USERSPACE1                2 ...
                                                       ...
8 record(s) selected.
```

この照会からの出力 (続き)。

```
... CONTAINER_NAME       CONTAINER_ID     CONTAINER_TYPE   ...
... -------------------- -------------...- ---------------- ...
... /home/swalkty/swalkt                0 FILE_EXTENT_TAG  ...
... /home/swalkty/swalkt                0 PATH             ...
... /home/swalkty/swalkt                0 FILE_EXTENT_TAG  ...
... /home/swalkty/swalkt                0 FILE_EXTENT_TAG  ...
... /home/swalkty/swalkt                0 PATH             ...
... /home/swalkty/swalkt                0 FILE_EXTENT_TAG  ...
... /home/swalkty/swalkt                0 PATH             ...
... /home/swalkty/swalkt                0 FILE_EXTENT_TAG  ...
```

この照会からの出力 (続き)。

```
... ACCESSIBLE DBPARTITIONNUM
... ---------- --------------
...          1              0
...          1              0
...          1              0
...          1              0
...          1              1
...          1              1
...          1              2
...          1              2
```

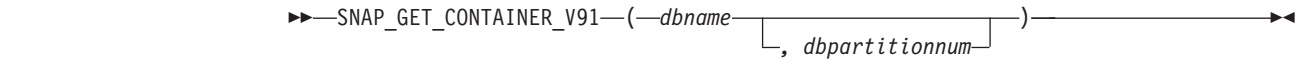
SNAP_GET_CONTAINER_V91 表関数

SNAP_GET_CONTAINER_V91 表関数は SNAPCONTAINER 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションのいずれかの特定のデータベースを対象とした情報を検索することができます。

SNAP_GET_TBSP_V91、 SNAP_GET_TBSP_PART_V91、 SNAP_GET_TBSP_QUIESCER、および SNAP_GET_TBSP_RANGE 表関数と共に使用すると、SNAP_GET_CONTAINER_V91 表関数は、GET SNAPSHOT FOR TABLESPACES ON database-alias CLP コマンドと同等のデータを戻します。

戻される可能性のある情報の完全なリストは、 674 ページの表 175を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL または空ストリングを指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプが含まれるファイルが存在しない場合、SNAP_GET_CONTAINER_V91 表関数は、現在接続されているデータベースのスナップショットとデータベース・パーティション番号を取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_CONTAINER_V91 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベース・パーティション上で現在接続されているデータベースの表スペース・コンテナーの詳細を検索します。

```
SELECT SNAPSHOT_TIMESTAMP, TBSP_NAME, TBSP_ID, CONTAINER_NAME,
   CONTAINER_ID, CONTAINER_TYPE, ACCESSIBLE
   FROM TABLE(SNAP_GET_CONTAINER_V91('',-1)) AS T
```

以下はこの照会の出力例です。

```
SNAPSHOT_TIMESTAMP         TBSP_NAME            TBSP_ID ...
-------------------------- -------------------- ------- ...
2005-04-25-14.42.10.899253 SYSCATSPACE                0 ...
2005-04-25-14.42.10.899253 TEMPSPACE1                 1 ...
2005-04-25-14.42.10.899253 USERSPACE1                 2 ...
2005-04-25-14.42.10.899253 SYSTOOLSPACE               3 ...
2005-04-25-14.42.10.899253 MYTEMP                     4 ...
2005-04-25-14.42.10.899253 WHATSNEWTEMPSPACE          5 ...
```

この照会からの出力 (続き)。

```
... CONTAINER_NAME                              CONTAINER_ID ...
... ------------------------------------------ ------------ ...
... D:¥DB2¥NODE0000¥SQL00002¥SQLT0000.0                   0 ...
... D:¥DB2¥NODE0000¥SQL00002¥SQLT0001.0                   0 ...
... D:¥DB2¥NODE0000¥SQL00002¥SQLT0002.0                   0 ...
... D:¥DB2¥NODE0000¥SQL00002¥SYSTOOLSPACE                 0 ...
... D:¥DB2¥NODE0000¥SQL003                                0 ...
... d:¥DGTTsWhatsNewContainer                             0 ...
```

この照会からの出力 (続き)。

```
... CONTAINER_TYPE ACCESSIBLE
... -------------- ----------
... CONT_PATH               1
... CONT_PATH               1
... CONT_PATH               1
... CONT_PATH               1
... CONT_PATH               1
... CONT_PATH               1
```

戻される情報

注: ファイル・システム情報を戻すためには、BUFFERPOOL データベース・マネージャーのモニター・スイッチをオンにする必要があります。

表 205. SNAPCONTAINER 管理ビューおよび SNAP_GET_CONTAINER_V91 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
TBSP_NAME	VARCHAR(128)	tablespace_name - 表スペース名
TBSP_ID	BIGINT	tablespace_id - 表スペース ID
CONTAINER_NAME	VARCHAR(256)	container_name - コンテナー名
CONTAINER_ID	BIGINT	container_id - コンテナー ID
CONTAINER_TYPE	VARCHAR(16)	container_type - コンテナー・タイプ。これは、sqlutil.h での定義を基にしたテキスト ID です。以下のいずれかとなります。 • DISK_EXTENT_TAG • DISK_PAGE_TAG • FILE_EXTENT_TAG • FILE_PAGE_TAG • PATH
TOTAL_PAGES	BIGINT	container_total_pages - コンテナー内の合計ページ数
USABLE_PAGES	BIGINT	container_usable_pages - コンテナー内の使用可能なページ数
ACCESSIBLE	SMALLINT	container_accessible - コンテナーのアクセス可能性
STRIPE_SET	BIGINT	container_stripe_set - ストライプ・セット

表 205. SNAPCONTAINER 管理ビューおよび SNAP_GET_CONTAINER_V91 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。
FS_ID	VARCHAR(22)	fs_id - 固有のファイル・システム識別番号
FS_TOTAL_SIZE	BIGINT	fs_total_size - ファイル・システムの合計サイズ
FS_USED_SIZE	BIGINT	fs_used_size - ファイル・システム上で使用されるスペースの量

SNAPDB 管理ビューおよび SNAP_GET_DB_V95 表関数 - dbase 論理グループからのスナップショット情報の検索

注: SNAP_GET_DB_V95 表関数は非推奨になり、 SNAP_GET_DB_V97 表関数 (dbase 論理グループからのスナップショット情報の取得) に置き換わりました。

『SNAPDB 管理ビュー』と 830 ページの『SNAP_GET_DB_V95 表関数』は、データベース (dbase) 論理グループからのスナップショット情報を戻します。

SNAPDB 管理ビュー

この管理ビューを使用すると、現在接続されているデータベースに関するスナップショット情報を dbase 論理グループから検索できます。

SNAPDB 管理ビューを SNAPDB_MEMORY_POOL、 SNAPDETAILLOG、 SNAPHADR、および SNAPSTORAGE_PATHS 管理ビューと併せて使用することにより、GET SNAPSHOT FOR DATABASE on database-alias CLP コマンドと同等の情報を戻します。

スキーマは SYSIBMADM です。

戻される情報の完全なリストは、 832 ページの表 206 を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPDB 管理ビューに対する SELECT 特権
- SNAPDB 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_DB_V95 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベースのすべてのデータベース・パーティションに関する状況、プラットフォーム、ロケーション、および接続時間を取り出します。

```
SELECT SUBSTR(DB_NAME, 1, 20) AS DB_NAME, DB_STATUS, SERVER_PLATFORM,
   DB_LOCATION, DB_CONN_TIME, DBPARTITIONNUM
   FROM SYSIBMADM.SNAPDB ORDER BY DBPARTITIONNUM
```

以下はこの照会の出力例です。

```
DB_NAME     DB_STATUS    SERVER_PLATFORM DB_LOCATION  ...
-------...- ------------ --------------- ------------ ...
TEST        ACTIVE       AIX64           LOCAL        ...
TEST        ACTIVE       AIX64           LOCAL        ...
TEST        ACTIVE       AIX64           LOCAL        ...

3 record(s) selected.
```

この照会からの出力 (続き)。

```
... DB_CONN_TIME               DBPARTITIONNUM
... -------------------------- --------------
... 2006-01-08-16.48.30.665477              0
... 2006-01-08-16.48.34.005328              1
... 2006-01-08-16.48.34.007937              2
```

このルーチンは、コマンド行で以下を呼び出すことにより使用できます。

```
SELECT TOTAL_OLAP_FUNCS, OLAP_FUNC_OVERFLOWS, ACTIVE_OLAP_FUNCS
   FROM SYSIBMADM.SNAPDB

TOTAL_OLAP_FUNCS     OLAP_FUNC_OVERFLOWS  ACTIVE_OLAP_FUNCS
-------------------- -------------------- -----------------
                   7                    2                 1

  1 record(s) selected.
```

ワークロードの実行後に、ユーザーは次の照会を使用できます。

```
SELECT STATS_CACHE_SIZE, STATS_FABRICATIONS, SYNC_RUNSTATS,
   ASYNC_RUNSTATS, STATS_FABRICATE_TIME, SYNC_RUNSTATS_TIME
   FROM SYSIBMADM.SNAPDB

STATS_CACHE_SIZE STATS_FABRICATIONS SYNC_RUNSTATS ASYNC_RUNSTATS ...
---------------- ------------------ ------------- -------------- ...
             128                  2             1              0 ...


... STATS_FABRICATE_TIME SYNC_RUNSTATS_TIME
... -------------------- ------------------
...                   10                100

1 record(s) selected.
```

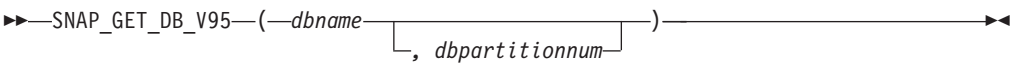
SNAP_GET_DB_V95 表関数

SNAP_GET_DB_V95 表関数は、SNAPDB 管理ビューと同じ情報を戻します。

SNAP_GET_DB_V95 表関数を SNAP_GET_DB_MEMORY_POOL、SNAP_GET_DETAILLOG_V91、 SNAP_GET_HADR、および SNAP_GET_STORAGE_PATHS 表関数と併せて使用することにより、GET SNAPSHOT FOR ALL DATABASES CLP コマンドと同等の情報を戻します。

戻される情報の完全なリストは、 832 ページの表 206 を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、空ストリングを指定します。現在接続されているデータベースと同じインスタンス内のすべてのデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプを持つファイルが存在しない場合には、現在接続されているデータベースのスナップショットとデータベース・パーティション番号を SNAP_GET_DB_V95 表関数が取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_DB_V95 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

例 1: 現在接続されているデータベースのすべてのデータベース・パーティションに渡る集約ビューとして、状況、プラットフォーム、ロケーション、および接続時間を取り出します。

```
SELECT SUBSTR(DB_NAME, 1, 20) AS DB_NAME, DB_STATUS, SERVER_PLATFORM,
   DB_LOCATION, DB_CONN_TIME FROM TABLE(SNAP_GET_DB_V95('', -2)) AS T
```

以下はこの照会の出力例です。

```
DB_NAME     DB_STATUS    SERVER_PLATFORM ...
-------...- ------------ --------------- ...
SAMPLE      ACTIVE       AIX64           ...

  1 record(s) selected.
```

この照会からの出力 (続き)。

```
... DB_LOCATION  DB_CONN_TIME
... ------------ --------------------------
... LOCAL        2005-07-24-22.09.22.013196
```

例 2: 現在接続されているデータベースを含む同じインスタンス内にあるすべてのアクティブ・データベースのすべてのデータベース・パーティションに渡る集約ビューとして、状況、プラットフォーム、ロケーション、および接続時間を取り出します。

```
SELECT SUBSTR(DB_NAME, 1, 20) AS DB_NAME, DB_STATUS, SERVER_PLATFORM,
   DB_LOCATION, DB_CONN_TIME
   FROM TABLE(SNAP_GET_DB_V95(CAST (NULL AS VARCHAR(128)), -2)) AS T
```

以下はこの照会の出力例です。

```
DB_NAME      DB_STATUS    SERVER_PLATFORM ...
--------...- ------------ --------------- ...
TOOLSDB      ACTIVE       AIX64           ...
SAMPLE       ACTIVE       AIX64           ...
```

この照会からの出力 (続き)。

```
... DB_LOCATION  DB_CONN_TIME
... ------------ --------------------------
... LOCAL        2005-07-24-22.26.54.396335
... LOCAL        2005-07-24-22.09.22.013196
```

例 3: このルーチンは、データベースへの接続時にコマンド行で以下を呼び出すことにより使用できます。

```
SELECT TOTAL_OLAP_FUNCS, OLAP_FUNC_OVERFLOWS, ACTIVE_OLAP_FUNCS
   FROM TABLE (SNAP_GET_DB_V95('', 0)) AS T
```

出力は次のようになります。

```
TOTAL_OLAP_FUNCS  OLAP_FUNC_OVERFLOWS  ACTIVE_OLAP_FUNCS
----------------  -------------------- --------------------
               7                    2                    1

1 record(s) selected.
```

例 4: ワークロードの実行後に、ユーザーは次の照会を表関数とともに使用できます。

```
 SELECT STATS_CACHE_SIZE, STATS_FABRICATIONS, SYNC_RUNSTATS,
   ASYNC_RUNSTATS, STATS_FABRICATE_TIME, SYNC_RUNSTATS_TIME
   FROM TABLE (SNAP_GET_DB_V95('mytestdb', -1)) AS SNAPDB

STATS_CACHE_SIZE STATS_FABRICATIONS SYNC_RUNSTATS ASYNC_RUNSTATS ...
---------------- ----------------- ------------- -------------- ...
             200                 1             2              0 ...

Continued

...STATS_FABRICATE_TIME  SYNC_RUNSTATS_TIME
...--------------------  ------------------
...                    2                  32

  1 record(s) selected.
```

SNAPDB 管理ビューおよび SNAP_GET_DB_V95 表関数のメタデータ

表 206. SNAPDB 管理ビューおよび SNAP_GET_DB_V95 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
DB_NAME	VARCHAR(128)	db_name - データベース名
DB_PATH	VARCHAR(1024)	db_path - データベース・パス
INPUT_DB_ALIAS	VARCHAR(128)	input_db_alias - 入力データベース別名
DB_STATUS	VARCHAR(12)	db_status - データベース状況。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • ACTIVE • QUIESCE_PEND • QUIESCED • ROLLFWD • ACTIVE_STANDBY • STANDBY
CATALOG_PARTITION	SMALLINT	catalog_node - カタログ・ノード番号
CATALOG_PARTITION_NAME	VARCHAR(128)	catalog_node_name - カタログ・ノード・ネットワーク名

表 206. SNAPDB 管理ビューおよび SNAP_GET_DB_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
SERVER_PLATFORM	VARCHAR(12)	server_platform - サーバーのオペレーティング・システム。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • AIX • AIX64 • AS400_DRDA • DOS • DYNIX • HP • HP64 • HPIA • HPIA64 • LINUX • LINUX390 • LINUXIA64 • LINUXPPC • LINUXPPC64 • LINUXX8664 • LINUXZ64 • MAC • MVS_DRDA • NT • NT64 • OS2 • OS390 • SCO • SGI • SNI • SUN • SUN64 • UNKNOWN • UNKNOWN_DRDA • VM_DRDA • VSE_DRDA • WINDOWS

表 206. SNAPDB 管理ビューおよび SNAP_GET_DB_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
DB_LOCATION	VARCHAR(12)	db_location - データベース・ロケーション。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • LOCAL • REMOTE
DB_CONN_TIME	TIMESTAMP	db_conn_time - データベース活動化タイム・スタンプ
LAST_RESET	TIMESTAMP	last_reset - 最後のリセット・タイム・スタンプ
LAST_BACKUP	TIMESTAMP	last_backup - 最終バックアップ・タイム・スタンプ
CONNECTIONS_TOP	BIGINT	connections_top - 同時接続の最大数
TOTAL_CONS	BIGINT	total_cons - データベース活動化以降の接続
TOTAL_SEC_CONS	BIGINT	total_sec_cons - 2 次接続
APPLS_CUR_CONS	BIGINT	appls_cur_cons - 現在接続されているアプリケーション
APPLS_IN_DB2	BIGINT	appls_in_db2 - データベースで現在実行中のアプリケーション
NUM_ASSOC_AGENTS	BIGINT	num_assoc_agents - 関連したエージェント数
AGENTS_TOP	BIGINT	agents_top - 作成されたエージェントの数
COORD_AGENTS_TOP	BIGINT	coord_agents_top - コーディネーター・エージェント最大数
LOCKS_HELD	BIGINT	locks_held - ロック保持数
LOCK_WAITS	BIGINT	lock_waits - ロック待機数
LOCK_WAIT_TIME	BIGINT	lock_wait_time - ロック待機中の時間
LOCK_LIST_IN_USE	BIGINT	lock_list_in_use - 使用中のロック・リスト・メモリーの合計
DEADLOCKS	BIGINT	deadlocks - デッドロック検出数
LOCK_ESCALS	BIGINT	lock_escals - ロック・エスカレーション数
X_LOCK_ESCALS	BIGINT	x_lock_escals - 排他ロック・エスカレーション数
LOCKS_WAITING	BIGINT	locks_waiting - ロックで待機中の現行エージェント
LOCK_TIMEOUTS	BIGINT	lock_timeouts - ロック・タイムアウト数
NUM_INDOUBT_TRANS	BIGINT	num_indoubt_trans - 未確定トランザクション数
SORT_HEAP_ALLOCATED	BIGINT	sort_heap_allocated - 割り振られたソート・ヒープの合計
SORT_SHRHEAP_ALLOCATED	BIGINT	sort_shrheap_allocated - 現在割り振られているソート共有ヒープ
SORT_SHRHEAP_TOP	BIGINT	sort_shrheap_top - ソート共有ヒープの最高水準点

表 206. SNAPDB 管理ビューおよび SNAP_GET_DB_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
POST_SHRTHRESHOLD_SORTS	BIGINT	post_shrthreshold_sorts - ポスト共有しきい値ソート
TOTAL_SORTS	BIGINT	total_sorts - ソート合計
TOTAL_SORT_TIME	BIGINT	total_sort_time - ソート時間合計
SORT_OVERFLOWS	BIGINT	sort_overflows - ソート・オーバーフロー
ACTIVE_SORTS	BIGINT	active_sorts - アクティブ・ソート
POOL_DATA_L_READS	BIGINT	pool_data_l_reads - バッファー・プール・データの論理読み取り
POOL_DATA_P_READS	BIGINT	pool_data_p_reads - バッファー・プール・データの物理読み取り
POOL_TEMP_DATA_L_READS	BIGINT	pool_temp_data_l_reads - バッファー・プール一時データの論理読み取り
POOL_TEMP_DATA_P_READS	BIGINT	pool_temp_data_p_reads - バッファー・プール一時データの物理読み取り
POOL_ASYNC_DATA_READS	BIGINT	pool_async_data_reads - バッファー・プール非同期データ読み取り
POOL_DATA_WRITES	BIGINT	pool_data_writes - バッファー・プールへのデータの書き込み
POOL_ASYNC_DATA_WRITES	BIGINT	pool_async_data_writes - バッファー・プール非同期データ書き込み
POOL_INDEX_L_READS	BIGINT	pool_index_l_reads - バッファー・プール索引の論理読み取り
POOL_INDEX_P_READS	BIGINT	pool_index_p_reads - バッファー・プール索引の物理読み取り
POOL_TEMP_INDEX_L_READS	BIGINT	pool_temp_index_l_reads - バッファー・プール一時索引の論理読み取り
POOL_TEMP_INDEX_P_READS	BIGINT	pool_temp_index_p_reads - バッファー・プール一時索引の物理読み取り
POOL_ASYNC_INDEX_READS	BIGINT	pool_async_index_reads - バッファー・プール非同期索引読み取り
POOL_INDEX_WRITES	BIGINT	pool_index_writes - バッファー・プール索引の書き込み
POOL_ASYNC_INDEX_WRITES	BIGINT	pool_async_index_writes - バッファー・プール非同期索引書き込み
POOL_XDA_P_READS	BIGINT	pool_xda_p_reads - バッファー・プール XDA データの物理読み取り
POOL_XDA_L_READS	BIGINT	pool_xda_l_reads - バッファー・プール XDA データの論理読み取り
POOL_XDA_WRITES	BIGINT	pool_xda_writes - バッファー・プール XDA データの書き込み
POOL_ASYNC_XDA_READS	BIGINT	pool_async_xda_reads - バッファー・プール非同期 XDA データ読み取り

表 206. SNAPDB 管理ビューおよび SNAP_GET_DB_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
POOL_ASYNC_XDA_WRITES	BIGINT	pool_async_xda_writes - バッファー・プール非同期 XDA データ書き込み
POOL_TEMP_XDA_P_READS	BIGINT	pool_temp_xda_p_reads - バッファー・プール一時 XDA データの物理読み取り : モニター・エレメント
POOL_TEMP_XDA_L_READS	BIGINT	pool_temp_xda_l_reads - バッファー・プール一時 XDA データの論理読み取り
POOL_READ_TIME	BIGINT	pool_read_time - バッファー・プール物理読み取り時間の合計
POOL_WRITE_TIME	BIGINT	pool_write_time - バッファー・プール物理書き込み時間の合計
POOL_ASYNC_READ_TIME	BIGINT	pool_async_read_time - バッファー・プール非同期読み取り時間
POOL_ASYNC_WRITE_TIME	BIGINT	pool_async_write_time - バッファー・プール非同期書き込み時間
POOL_ASYNC_DATA_ READ_REQS	BIGINT	pool_async_data_read_reqs - バッファー・プール非同期読み取り要求
POOL_ASYNC_INDEX_ READ_REQS	BIGINT	pool_async_index_read_reqs - バッファー・プール非同期索引読み取り要求
POOL_ASYNC_XDA_ READ_REQS	BIGINT	pool_async_xda_read_reqs - バッファー・プール非同期 XDA 読み取り要求
POOL_NO_VICTIM_BUFFER	BIGINT	pool_no_victim_buffer - バッファー・プールの非ビクティム・バッファー数
POOL_LSN_GAP_CLNS	BIGINT	pool_lsn_gap_clns - 起動されたバッファー・プール・ログ・スペース・クリーナー
POOL_DRTY_PG_STEAL_CLNS	BIGINT	pool_drty_pg_steal_clns - 起動されたバッファー・プール・ビクティム・ページ・クリーナー
POOL_DRTY_PG_THRSH_CLNS	BIGINT	pool_drty_pg_thrsh_clns - 起動されたバッファー・プールしきい値クリーナー
PREFETCH_WAIT_TIME	BIGINT	prefetch_wait_time - プリフェッチ待ち時間
UNREAD_PREFETCH_PAGES	BIGINT	unread_prefetch_pages - 読み取り不能プリフェッチ・ページ
DIRECT_READS	BIGINT	direct_reads - データベースからの直接読み取り
DIRECT_WRITES	BIGINT	direct_writes - データベースへの直接書き込み
DIRECT_READ_REQS	BIGINT	direct_read_reqs - 直接読み取り要求
DIRECT_WRITE_REQS	BIGINT	direct_write_reqs - 直接書き込み要求
DIRECT_READ_TIME	BIGINT	direct_read_time - 直接読み取り時間
DIRECT_WRITE_TIME	BIGINT	direct_write_time - 直接書き込み時間

表 206. SNAPDB 管理ビューおよび SNAP_GET_DB_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
FILES_CLOSED	BIGINT	files_closed - 閉じられたデータベース・ファイル
ELAPSED_EXEC_TIME_S	BIGINT	elapsed_exec_time - ステートメント実行経過時間 (秒単位)*
ELAPSED_EXEC_TIME_MS	BIGINT	elapsed_exec_time - ステートメント実行経過時間 (小数部、マイクロ秒単位)*
COMMIT_SQL_STMTS	BIGINT	commit_sql_stmts - 試行されたコミット・ステートメント
ROLLBACK_SQL_STMTS	BIGINT	rollback_sql_stmts - 試行されたロールバック・ステートメント
DYNAMIC_SQL_STMTS	BIGINT	dynamic_sql_stmts - 試行された動的 SQL ステートメント
STATIC_SQL_STMTS	BIGINT	static_sql_stmts - 試行された静的 SQL ステートメント
FAILED_SQL_STMTS	BIGINT	failed_sql_stmts - 失敗したステートメント操作
SELECT_SQL_STMTS	BIGINT	select_sql_stmts - 実行された選択 SQL ステートメント
UID_SQL_STMTS	BIGINT	uid_sql_stmts - 実行された更新/挿入/削除 SQL ステートメント
DDL_SQL_STMTS	BIGINT	ddl_sql_stmts - データ定義言語 (DDL) SQL ステートメント
INT_AUTO_REBINDS	BIGINT	int_auto_rebinds - 内部自動再バインド
INT_ROWS_DELETED	BIGINT	int_rows_deleted - 削除された内部行数
INT_ROWS_INSERTED	BIGINT	int_rows_inserted - 挿入された内部行数
INT_ROWS_UPDATED	BIGINT	int_rows_updated - 更新された内部行数
INT_COMMITS	BIGINT	int_commits - 内部コミット数
INT_ROLLBACKS	BIGINT	int_rollbacks - 内部ロールバック数
INT_DEADLOCK_ROLLBACKS	BIGINT	int_deadlock_rollbacks - デッドロックによる内部ロールバック数
ROWS_DELETED	BIGINT	rows_deleted - 削除行数
ROWS_INSERTED	BIGINT	rows_inserted - 挿入行数
ROWS_UPDATED	BIGINT	rows_updated - 更新行数
ROWS_SELECTED	BIGINT	rows_selected - 選択行数
ROWS_READ	BIGINT	rows_read - 読み取り行数
BINDS_PRECOMPILES	BIGINT	binds_precompiles - 試行されたバインド/プリコンパイル
TOTAL_LOG_AVAILABLE	BIGINT	total_log_available - 使用可能なログ合計
TOTAL_LOG_USED	BIGINT	total_log_used - 使用されているログ・スペースの合計
SEC_LOG_USED_TOP	BIGINT	sec_log_used_top - 使用された最大 2 次ログ・スペース

表 206. SNAPDB 管理ビューおよび SNAP_GET_DB_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
TOT_LOG_USED_TOP	BIGINT	tot_log_used_top - 使用された最大合計ログ・スペース
SEC_LOGS_ALLOCATED	BIGINT	sec_logs_allocated - 現在割り振られている 2 次ログ
LOG_READS	BIGINT	log_reads - 読み取られたログ・ページの数
LOG_READ_TIME_S	BIGINT	log_read_time - ログ読み取り時間 (秒単位)†
LOG_READ_TIME_NS	BIGINT	log_read_time - ログ読み取り時間 (小数部、ナノ秒単位)†
LOG_WRITES	BIGINT	log_writes - 書き込まれたログ・ページの数
LOG_WRITE_TIME_S	BIGINT	log_write_time - ログ書き込み時間 (秒単位)†
LOG_WRITE_TIME_NS	BIGINT	log_write_time - ログ書き込み時間 (小数部、ナノ秒単位)†
NUM_LOG_WRITE_IO	BIGINT	num_log_write_io - ログ書き込み数
NUM_LOG_READ_IO	BIGINT	num_log_read_io - ログ読み取り数
NUM_LOG_PART_PAGE_IO	BIGINT	num_log_part_page_io - 部分ログ・ページ書き込み数
NUM_LOG_BUFFER_FULL	BIGINT	num_log_buffer_full - フル・ログ・バッファーの回数
NUM_LOG_DATA_FOUND_IN_BUFFER	BIGINT	num_log_data_found_in_buffer - ログ・データがバッファーにある回数
APPL_ID_OLDEST_XACT	BIGINT	appl_id_oldest_xact - 最も古いトランザクションを使用するアプリケーション
LOG_TO_REDO_FOR_RECOVERY	BIGINT	log_to_redo_for_recovery - リカバリーの場合に再実行されるログの量
LOG_HELD_BY_DIRTY_PAGES	BIGINT	log_held_by_dirty_pages - ダーティー・ページ別に計算されるログ・スペースの量
PKG_CACHE_LOOKUPS	BIGINT	pkg_cache_lookups - パッケージ・キャッシュ参照
PKG_CACHE_INSERTS	BIGINT	pkg_cache_inserts - パッケージ・キャッシュ挿入
PKG_CACHE_NUM_OVERFLOWS	BIGINT	pkg_cache_num_overflows - パッケージ・キャッシュ・オーバーフロー数
PKG_CACHE_SIZE_TOP	BIGINT	pkg_cache_size_top - パッケージ・キャッシュ最高水準点
APPL_SECTION_LOOKUPS	BIGINT	appl_section_lookups - セクションの参照回数
APPL_SECTION_INSERTS	BIGINT	appl_section_inserts - セクション挿入数
CAT_CACHE_LOOKUPS	BIGINT	cat_cache_lookups - カタログ・キャッシュ参照数
CAT_CACHE_INSERTS	BIGINT	cat_cache_inserts - カタログ・キャッシュ挿入数

表 206. SNAPDB 管理ビューおよび SNAP_GET_DB_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
CAT_CACHE_OVERFLOWS	BIGINT	cat_cache_overflows - カタログ・キャッシュ・オーバーフロー数
CAT_CACHE_SIZE_TOP	BIGINT	cat_cache_size_top - カタログ・キャッシュ最高水準点
PRIV_WORKSPACE_SIZE_TOP	BIGINT	priv_workspace_size_top - 専用ワークスペースの最大サイズ
PRIV_WORKSPACE_NUM_OVERFLOWS	BIGINT	priv_workspace_num_overflows - 専用ワークスペースのオーバーフロー回数
PRIV_WORKSPACE_SECTION_INSERTS	BIGINT	priv_workspace_section_inserts - 専用ワークスペース・セクション挿入
PRIV_WORKSPACE_SECTION_LOOKUPS	BIGINT	priv_workspace_section_lookups - 専用ワークスペース・セクションの参照
SHR_WORKSPACE_SIZE_TOP	BIGINT	shr_workspace_size_top - 最大共有ワークスペース・サイズ
SHR_WORKSPACE_NUM_OVERFLOWS	BIGINT	shr_workspace_num_overflows - 共有ワークスペースのオーバーフロー回数
SHR_WORKSPACE_SECTION_INSERTS	BIGINT	shr_workspace_section_inserts - 共有ワークスペース・セクション挿入数
SHR_WORKSPACE_SECTION_LOOKUPS	BIGINT	shr_workspace_section_lookups - 共有ワークスペース・セクションの参照回数
TOTAL_HASH_JOINS	BIGINT	total_hash_joins - ハッシュ結合の合計
TOTAL_HASH_LOOPS	BIGINT	total_hash_loops - ハッシュ・ループの合計
HASH_JOIN_OVERFLOWS	BIGINT	hash_join_overflows - ハッシュ結合のオーバーフロー
HASH_JOIN_SMALL_OVERFLOWS	BIGINT	hash_join_small_overflows - ハッシュ結合の短精度オーバーフロー
POST_SHRTHRESHOLD_HASH_JOINS	BIGINT	post_shrthreshold_hash_joins - ポストしきい値ハッシュ結合
ACTIVE_HASH_JOINS	BIGINT	active_hash_joins - アクティブ・ハッシュ結合
NUM_DB_STORAGE_PATHS	BIGINT	num_db_storage_paths - 自動ストレージ・パスの数
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。
SMALLEST_LOG_AVAIL_NODE	INTEGER	smallest_log_avail_node - 使用可能なログ・スペースが最小のノード
TOTAL_OLAP_FUNCS	BIGINT	実行される OLAP 関数の合計数。

表 206. SNAPDB 管理ビューおよび SNAP_GET_DB_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
OLAP_FUNC_OVERFLOWS	BIGINT	OLAP 関数データが使用可能なソート・ヒープ・スペースを超えた回数。
ACTIVE_OLAP_FUNCS	BIGINT	現在実行中で、ソート・ヒープ・メモリーを消費している OLAP 関数の合計数。
STATS_CACHE_SIZE	BIGINT	統計キャッシュのサイズ (バイト)。
STATS_FABRICATIONS	BIGINT	表または索引のスキャンを実行しないでシステムが統計を作成するための statistics-collect アクティビティーの合計数。
SYNC_RUNSTATS	BIGINT	照会コンパイル中の同期 statistics-collect アクティビティーの合計数。
ASYNC_RUNSTATS	BIGINT	この列の出力は、成功した非同期 statistics-collect アクティビティーの合計数に変更されます。
STATS_FABRICATE_TIME	BIGINT	照会コンパイル中に表または索引のスキャンを実行しないでシステムが統計を作成するのに費やされる合計時間 (ミリ秒)。
SYNC_RUNSTATS_TIME	BIGINT	同期 statistics-collect アクティビティーに費やされる合計時間 (ミリ秒)。
NUM_THRESHOLD_VIOLATIONS	BIGINT	データベースで発生したしきい値違反の数。
* この列の元になるモニター・エレメントの合計消費時間を計算するには、このモニター・エレメントに関する _S で終わっている列で報告されている整数秒と、このモニター・エレメントに関する _MS で終わっている列に報告されている小数秒を、次の式を使用して加算しなければなりません: (*monitor-element-name*_S × 1,000,000 + *monitor-element-name*_MS) ÷ 1,000,000. 例えば、(ELAPSED_EXEC_TIME_S × 1,000,000 + ELAPSED_EXEC_TIME_MS) ÷ 1,000,000。 †このモニター・エレメントの合計経過時間を計算するには、このモニター・エレメントに関する _S で終わっている列で報告されている整数秒と、このモニター・エレメントに関する _NS で終わっている列に報告されている小数秒を、次の式を使用して加算しなければなりません: (*monitor-element-name*_S × 1,000,000,000 + *monitor-element-name*_NS) ÷ 1,000,000,000。例えば、(LOG_READ_TIME_S × 1,000,000,000 + LOG_READ_TIME_NS) ÷ 1,000,000,000。		

SNAPDB_MEMORY_POOL 管理ビューおよび SNAP_GET_DB_MEMORY_POOL 表関数 - データベース・レベルのメモリー使用量情報の検索

SNAPDB_MEMORY_POOL 管理ビューおよび SNAP_GET_DB_MEMORY_POOL 表関数は、データベース・レベルでのメモリー使用量についての情報を戻します (UNIX プラットフォームの場合のみ)。

SNAPDB_MEMORY_POOL 管理ビュー

この管理ビューを使用して、現在接続中のデータベースに関するデータベース・レベルのメモリー使用量情報を取得することができます。

SNAPDB_MEMORY_POOL 管理ビューを SNAPDB、 SNAPDETAILLOG、 SNAPHADR、および SNAPSTORAGE_PATHS 管理ビューとともに使用すると、 GET SNAPSHOT FOR DATABASE ON database-alias CLP コマンドに相当する情報が提供されます。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 691 ページの表 177を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPDB_MEMORY_POOL 管理ビューに対する SELECT 特権
- SNAPDB_MEMORY_POOL 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_DB_MEMORY_POOL 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続中のデータベースである SAMPLE のメモリー・プールおよびそれらの現在のサイズのリストを取得します。

```
SELECT POOL_ID, POOL_CUR_SIZE FROM SYSIBMADM.SNAPDB_MEMORY_POOL
```

以下はこの照会の出力例です。

```
POOL_ID       POOL_CUR_SIZE
------------- --------------------
UTILITY                      32768
PACKAGE_CACHE               475136
CAT_CACHE                    65536
BP                         2097152
BP                         1081344
BP                          540672
BP                          278528
BP                          147456
BP                           81920
LOCK_MGR                    294912
DATABASE                   3833856
OTHER                            0

  12 record(s) selected.
```

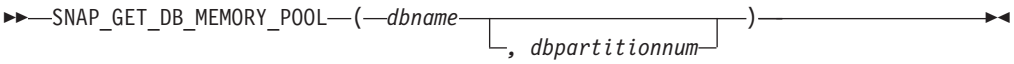
SNAP_GET_DB_MEMORY_POOL 表関数

SNAP_GET_DB_MEMORY_POOL 表関数は、SNAPDB_MEMORY_POOL 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションに関する情報を取得することができます。

SNAP_GET_DB_MEMORY_POOL 表関数を SNAP_GET_DB_V95、SNAP_GET_DETAILLOG_V91、 SNAP_GET_HADR、および SNAP_GET_STORAGE_PATHS 表関数とともに使用すると、GET SNAPSHOT FOR ALL DATABASES CLP コマンドに相当する情報が提供されます。

戻される可能性のある情報の完全なリストは、 691 ページの表 177を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、空ストリングを指定します。現在接続されているデータベースと同じインスタンス内のすべてのデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプを持つファイルが存在しない場合、SNAP_GET_DB_MEMORY_POOL 表関数は、現在接続中のデータベースおよびデー

タベース・パーティション番号のスナップショットを取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_DB_MEMORY_POOL 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

すべてのデータベースのメモリー・プールおよびそれらの現在のサイズのリストを取得します。

```
SELECT SUBSTR(DB_NAME,1,8) AS DB_NAME, POOL_ID, POOL_CUR_SIZE
   FROM TABLE(SNAPSHOT_GET_DB_MEMORY_POOL
   (CAST(NULL AS VARCHAR(128)), -1)) AS T
```

以下はこの照会の出力例です。

```
DB_NAME  POOL_ID        POOL_CUR_SIZE
-------- -------------- --------------------
TESTDB   UTILITY                  65536
TESTDB   PACKAGE_CACHE           851968
TESTDB   CAT_CACHE                65536
TESTDB   BP                    35913728
TESTDB   BP                      589824
TESTDB   BP                      327680
TESTDB   BP                      196608
TESTDB   BP                      131072
TESTDB   SHARED_SORT              65536
TESTDB   LOCK_MGR              10092544
TESTDB   DATABASE               4980736
TESTDB   OTHER                   196608
SAMPLE   UTILITY                  65536
SAMPLE   PACKAGE_CACHE           655360
SAMPLE   CAT_CACHE                131072
SAMPLE   BP                     4325376
SAMPLE   BP                      589824
SAMPLE   BP                      327680
SAMPLE   BP                      196608
SAMPLE   BP                      131072
SAMPLE   SHARED_SORT                  0
SAMPLE   LOCK_MGR                655360
SAMPLE   DATABASE               4653056
SAMPLE   OTHER                   196608

24 record(s) selected.
```

戻される情報

表 207. SNAPDB_MEMORY_POOL 管理ビューおよび SNAP_GET_DB_MEMORY_POOL 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
DB_NAME	VARCHAR(128)	db_name - データベース名
POOL_ID	VARCHAR(14)	pool_id - メモリー・プール ID。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • APP_GROUP • APPL_CONTROL • APPLICATION • BP • CAT_CACHE • DATABASE • DFM • FCMBP • IMPORT_POOL • LOCK_MGR • MONITOR • OTHER • PACKAGE_CACHE • QUERY • SHARED_SORT • SORT • STATEMENT • STATISTICS • UTILITY
POOL_SECONDARY_ID	VARCHAR(32)	pool_secondary_id - メモリー・プール 2 次 ID
POOL_CUR_SIZE	BIGINT	pool_cur_size - メモリー・プールの現行サイズ
POOL_WATERMARK	BIGINT	pool_watermark - メモリー・プール水準点
POOL_CONFIG_SIZE	BIGINT	pool_config_size - メモリー・プールの構成済みサイズ
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

SNAPDBM 管理ビューおよび SNAP_GET_DBM_V95 表関数 - dbm 論理グループ・スナップショット情報の検索

SNAPDBM 管理ビューおよび SNAP_GET_DBM_V95 表関数 は、スナップショット・モニターの DB2 データベース・マネージャー (dbm) 論理グループ情報を戻します。

SNAPDBM 管理ビュー

SNAPDBM_MEMORY_POOL、SNAPFCM、SNAPFCM_PART、および SNAPSWITCHES 管理ビューと共に使用すると、SNAPDBM 管理ビューは、GET SNAPSHOT FOR DBM コマンドと同等のデータを提供します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、695 ページの表 178を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPDBM 管理ビューに対する SELECT 特権
- SNAPDBM 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_DBM_V95 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

すべてのデータベース・パーティションにおけるデータベース・マネージャーの状況と接続情報を検索します。

```
SELECT DB2_STATUS, DB2START_TIME, LAST_RESET, LOCAL_CONS, REM_CONS_IN,
   (AGENTS_CREATED_EMPTY_POOL/AGENTS_FROM_POOL) AS AGENT_USAGE,
   DBPARTITIONNUM FROM SYSIBMADM.SNAPDBM ORDER BY DBPARTITIONNUM
```

以下はこの照会の出力例です。

```
DB2_STATUS   DB2START_TIME              LAST_RESET     ...
------------ -------------------------- ----------...- ...
ACTIVE       2006-01-06-14.59.59.059879              - ...
ACTIVE       2006-01-06-14.59.59.097605              - ...
```

```
ACTIVE       2006-01-06-14.59.59.062798             - ...

  3 record(s) selected.                                  ...
```

この照会からの出力 (続き)。

```
... LOCAL_CONS     REM_CONS_IN     AGENT_USAGE     DBPARTITIONNUM
... -----------...- -----------...- -----------...- --------------
...              1               1               0              0
...              0               0               0              1
...              0               0               0              2
```

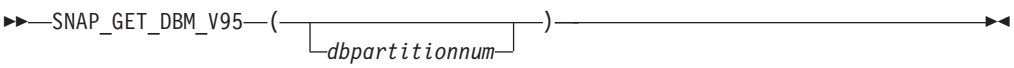
SNAP_GET_DBM_V95 表関数

SNAP_GET_DBM_V95 表関数は SNAPDBM 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションを対象とした情報を検索することができます。

SNAP_GET_DBM_MEMORY_POOL、SNAP_GET_FCM、SNAP_GET_FCM_PART、および SNAP_GET_SWITCHES 表関数と共に使用すると、SNAP_GET_DBM_V95 表関数は、GET SNAPSHOT FOR DBM コマンドと同等のデータを提供します。

戻される可能性のある情報の完全なリストは、 695 ページの表 178を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。この入力オプションが使用されない場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbpartitionnum が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプが含まれるファイルが存在しない場合、SNAP_GET_DBM_V95 表関数はメモリーからスナップショットを呼び出します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_DBM_V95 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

データベース・パーティション番号 2 の開始時刻と現行状況を検索します。

```
SELECT DB2START_TIME, DB2_STATUS FROM TABLE(SNAP_GET_DBM_V95(2)) AS T
```

以下はこの照会の出力例です。

```
DB2START_TIME              DB2_STATUS
-------------------------- ------------
2006-01-06-14.59.59.062798 ACTIVE
```

戻される情報

表 208. SNAPDBM 管理ビューおよび SNAP_GET_DBM_V95 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
SORT_HEAP_ALLOCATED	BIGINT	sort_heap_allocated - 割り振られたソート・ヒープの合計
POST_THRESHOLD_SORTS	BIGINT	post_threshold_sorts - ポストしきい値ソート
PIPED_SORTS_REQUESTED	BIGINT	piped_sorts_requested - 要求されたパイプ・ソート数
PIPED_SORTS_ACCEPTED	BIGINT	piped_sorts_accepted - 受け入れられたパイプ・ソート
REM_CONS_IN	BIGINT	rem_cons_in - データベース・マネージャーへのリモート接続
REM_CONS_IN_EXEC	BIGINT	rem_cons_in_exec - データベース・マネージャーで実行中のリモート接続 : モニター・エレメント
LOCAL_CONS	BIGINT	local_cons - ローカル接続
LOCAL_CONS_IN_EXEC	BIGINT	local_cons_in_exec - データベース・マネージャーで実行中のローカル接続 : モニター・エレメント
CON_LOCAL_DBASES	BIGINT	con_local_dbases - 現行接続を使用したローカル・データベース
AGENTS_REGISTERED	BIGINT	agents_registered - 登録済みエージェント

表 208. SNAPDBM 管理ビューおよび SNAP_GET_DBM_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
AGENTS_WAITING_ON_TOKEN	BIGINT	agents_waiting_on_token - トークン待ちエージェント
DB2_STATUS	VARCHAR(12)	db2_status - DB2 インスタンス状況。 このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • ACTIVE • QUIESCE_PEND • QUIESCED
AGENTS_REGISTERED_TOP	BIGINT	agents_registered_top - エージェント最大登録数
AGENTS_WAITING_TOP	BIGINT	agents_waiting_top - エージェント最大待機数
COMM_PRIVATE_MEM	BIGINT	comm_private_mem - コミット済み専用メモリー
IDLE_AGENTS	BIGINT	idle_agents - アイドル・エージェント数
AGENTS_FROM_POOL	BIGINT	agents_from_pool - プールから割り当てられたエージェント
AGENTS_CREATED_EMPTY_POOL	BIGINT	agents_created_empty_pool - エージェント・プールが空のために作成されたエージェント
COORD_AGENTS_TOP	BIGINT	coord_agents_top - コーディネーター・エージェント最大数
MAX_AGENT_OVERFLOWS	BIGINT	max_agent_overflows - 最大エージェント・オーバーフロー回数
AGENTS_STOLEN	BIGINT	agents_stolen - スチールされたエージェント
GW_TOTAL_CONS	BIGINT	gw_total_cons - DB2 Connect の接続試行合計回数
GW_CUR_CONS	BIGINT	gw_cur_cons - DB2 Connect の現在の接続数
GW_CONS_WAIT_HOST	BIGINT	gw_cons_wait_host - ホストの応答を待機している接続の数
GW_CONS_WAIT_CLIENT	BIGINT	gw_cons_wait_client - クライアントの要求送信を待機している接続の数
POST_THRESHOLD_ HASH_JOINS	BIGINT	post_threshold_hash_joins - ハッシュ結合のしきい値
NUM_GW_CONN_SWITCHES	BIGINT	num_gw_conn_switches - 接続切り替え回数
DB2START_TIME	TIMESTAMP	db2start_time - データベース・マネージャー開始タイム・スタンプ
LAST_RESET	TIMESTAMP	last_reset - 最後のリセット・タイム・スタンプ
NUM_NODES_IN_ DB2_INSTANCE	INTEGER	num_nodes_in_db2_instance - データベース・パーティション内のノード数
PRODUCT_NAME	VARCHAR(32)	product_name - 製品名
SERVICE_LEVEL	VARCHAR(18)	service_level - サービス・レベル

表 208. SNAPDBM 管理ビューおよび SNAP_GET_DBM_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
SORT_HEAP_TOP	BIGINT	sort_heap_top - ソート専用ヒープの最高水準点
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。
POST_THRESHOLD_OLAP_FUNCS	BIGINT	ソート・ヒープしきい値を超えた後でソート・ヒープを要求した OLAP 関数の数。 ソート、ハッシュ結合、および OLAP 関数は、ソート・ヒープを使用する操作の例です。通常の条件では、データベース・マネージャーは sortheap 構成パラメーターによって指定された値を使用して、ソート・ヒープを割り振ります。ソート・ヒープに割り振られたメモリーの量がソート・ヒープしきい値 (sheapthres 構成パラメーター) を超える場合、データベース・マネージャーは sortheap 構成パラメーターで指定されたものより小さい値を使用して、その後のソート・ヒープを割り振ります。 ソート・ヒープしきい値に達した後で開始する OLAP 関数は、実行に最適な量のメモリーを受け取ることができない場合があります。

SNAPDBM_MEMORY_POOL 管理ビューおよび SNAP_GET_DBM_MEMORY_POOL 表関数 - データベース・マネージャー・レベルのメモリー使用量情報の検索

SNAPDBM_MEMORY_POOL 管理ビューおよび SNAP_GET_DBM_MEMORY_POOL 表関数は、データベース・マネージャーでのメモリー使用量についての情報を戻します。

SNAPDBM_MEMORY_POOL 管理ビュー

SNAPDBM_MEMORY_POOL 管理ビューを SNAPDBM、SNAPFCM、SNAPFCM_PART、および SNAPSWITCHES 管理ビューとともに使用すると、GET SNAPSHOT FOR DBM コマンドに相当するデータが提供されます。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 700 ページの表 179を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPDBM_MEMORY_POOL 管理ビューに対する SELECT 特権
- SNAPDBM_MEMORY_POOL 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_DBM_MEMORY_POOL 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

接続中のデータベースのデータベース・マネージャーのメモリー・プールおよびそれらの現在のサイズのリストを取得します。

```
SELECT POOL_ID, POOL_CUR_SIZE FROM SNAPDBM_MEMORY_POOL
```

以下はこの照会の出力例です。

```
POOL_ID        POOL_CUR_SIZE
-------------- --------------------
MONITOR                   65536
OTHER                  29622272
FCMBP                  57606144
...
```

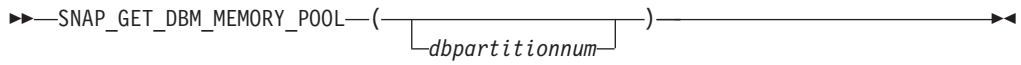
SNAP_GET_DBM_MEMORY_POOL 表関数

SNAP_GET_DBM_MEMORY_POOL 表関数は、SNAPDBM_MEMORY_POOL 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションに関する情報を取得することができます。

SNAP_GET_DBM_MEMORY_POOL 表関数を SNAP_GET_DBM_V95、SNAP_GET_FCM、 SNAP_GET_FCM_PART、および SNAP_GET_SWITCHES 表関数とともに使用すると、GET SNAPSHOT FOR DBM コマンドに相当するデータが提供されます。

戻される可能性のある情報の完全なリストは、 700 ページの表 179を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。この入力オプションが使用されない場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbpartitionnum が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプを持つファイルが存在しない場合、SNAP_GET_DBM_MEMORY_POOL 表関数は、現在接続中のデータベースおよびデータベース・パーティション番号のスナップショットを取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_DBM_MEMORY_POOL 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

接続中のデータベースのデータベース・マネージャーのすべてのデータベース・パーティションのメモリー・プールおよびそれらの現在のサイズのリストを取得します。

```
SELECT POOL_ID, POOL_CUR_SIZE, DBPARTITIONNUM
   FROM TABLE(SYSPROC.SNAP_GET_DBM_MEMORY_POOL())
   AS T ORDER BY DBPARTITIONNUM
```

以下はこの照会の出力例です。

```
POOL_ID        POOL_CUR_SIZE        DBPARTITIONNUM
-------------- -------------------- --------------
MONITOR                     65536              0
OTHER                    29622272              0
```

```
FCMBP                    57606144             0
MONITOR                     65536             1
OTHER                    29425664             1
FCMBP                    57606144             1
MONITOR                     65536             2
OTHER                    29425664             2
FCMBP                    57606144             2
```

戻される情報

表 209. SNAPDBM_MEMORY_POOL 管理ビューおよび SNAP_GET_DBM_MEMORY_POOL 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
POOL_ID	VARCHAR(14)	pool_id - メモリー・プール ID。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • APP_GROUP • APPL_CONTROL • APPLICATION • BP • CAT_CACHE • DATABASE • DFM • FCMBP • IMPORT_POOL • LOCK_MGR • MONITOR • OTHER • PACKAGE_CACHE • QUERY • SHARED_SORT • SORT • STATEMENT • STATISTICS • UTILITY
POOL_CUR_SIZE	BIGINT	pool_cur_size - メモリー・プールの現行サイズ
POOL_WATERMARK	BIGINT	pool_watermark - メモリー・プール水準点
POOL_CONFIG_SIZE	BIGINT	pool_config_size - メモリー・プールの構成済みサイズ
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

SNAPDETAILLOG 管理ビューおよび SNAP_GET_DETAILLOG_V91 表関数 - detail_log 論理データ・グループからのスナップショット情報の検索

SNAPDETAILLOG 管理ビューおよび SNAP_GET_DETAILLOG_V91 表関数は、detail_log 論理データ・グループからのスナップショット情報を戻します。

SNAPDETAILLOG 管理ビュー

この管理ビューを使用すると、現在接続されているデータベースに関するスナップショット情報を detail_log 論理データ・グループから検索できます。

SNAPDETAILLOG 管理ビューを SNAPDB、 SNAPDB_MEMORY_POOL、SNAPHADR、および SNAPSTORAGE_PATHS 管理ビューと併せて使用することにより、GET SNAPSHOT FOR DATABASE on database-alias CLP コマンドと同等の情報を戻します。

スキーマは SYSIBMADM です。

戻される情報の完全なリストは、 704 ページの表 180 を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPDETAILLOG 管理ビューに対する SELECT 特権
- SNAPDETAILLOG 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_DETAILLOG_V91 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

すべてのデータベース・パーティションに関して、現在接続されているデータベースのログ情報を検索します。

```
SELECT SUBSTR(DB_NAME, 1, 8) AS DB_NAME, FIRST_ACTIVE_LOG,
   LAST_ACTIVE_LOG, CURRENT_ACTIVE_LOG, CURRENT_ARCHIVE_LOG,
   DBPARTITIONNUM
   FROM SYSIBMADM.SNAPDETAILLOG ORDER BY DBPARTITIONNUM
```

以下はこの照会の出力例です。

```
DB_NAME  FIRST_ACTIVE_LOG     LAST_ACTIVE_LOG      ...
-------- -------------------- -------------------- ...
TEST                        0                    8 ...
TEST                        0                    8 ...
TEST                        0                    8 ...

  3 record(s) selected.
```

この照会からの出力 (続き)。

```
... CURRENT_ACTIVE_LOG   CURRENT_ARCHIVE_LOG  DBPARTITIONNUM
... -------------------- -------------------- --------------
...                    0                    -              0
...                    0                    -              1
...                    0                    -              2
```

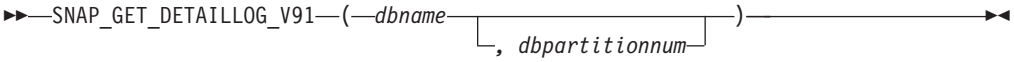
SNAP_GET_DETAILLOG_V91 表関数

SNAP_GET_DETAILLOG_V91 表関数は、SNAPDETAILLOG 管理ビューと同じ情報を戻します。

SNAP_GET_DETAILLOG 表関数を SNAP_GET_DB_V95、SNAP_GET_DB_MEMORY_POOL、 SNAP_GET_HADR、および SNAP_GET_STORAGE_PATHS 表関数と併せて使用することにより、GET SNAPSHOT FOR ALL DATABASES CLP コマンドと同等の情報を戻します。

戻される情報の完全なリストは、 704 ページの表 180 を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、空ストリングを指定します。現在接続されているデータベースと同じインスタンス内のすべてのデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブ

なデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプを持つファイルが存在しない場合には、現在接続されているデータベースのスナップショットとデータベース・パーティション番号を SNAP_GET_DETAILLOG_V91 表関数が取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_DETAILLOG_V91 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

データベース・パーティション 1 に関して、現在接続されているデータベースのログ情報を検索します。

```
SELECT SUBSTR(DB_NAME, 1, 8) AS DB_NAME, FIRST_ACTIVE_LOG,
   LAST_ACTIVE_LOG, CURRENT_ACTIVE_LOG, CURRENT_ARCHIVE_LOG
   FROM TABLE(SNAP_GET_DETAILLOG_V91('', 1)) AS T
```

以下はこの照会の出力例です。

```
DB_NAME  FIRST_ACTIVE_LOG     LAST_ACTIVE_LOG      ...
-------- -------------------- -------------------- ...
TEST                        0                    8 ...

  1 record(s) selected.
```

この照会からの出力 (続き)。

```
... CURRENT_ACTIVE_LOG   CURRENT_ARCHIVE_LOG
... -------------------- --------------------
...                    0                    -
```

SNAPDETAILLOG 管理ビューおよび SNAP_GET_DETAILLOG_V91 表関数のメタデータ

表 210. SNAPDETAILLOG 管理ビューおよび SNAP_GET_DETAILLOG_V91 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
DB_NAME	VARCHAR(128)	db_name - データベース名
FIRST_ACTIVE_LOG	BIGINT	first_active_log - 先頭アクティブ・ログ・ファイル番号
LAST_ACTIVE_LOG	BIGINT	last_active_log - 最終アクティブ・ログ・ファイル番号
CURRENT_ACTIVE_LOG	BIGINT	current_active_log - 現行アクティブ・ログ・ファイル番号
CURRENT_ARCHIVE_LOG	BIGINT	current_archive_log - 現行アーカイブ・ログ・ファイル番号
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

SNAPDYN_SQL 管理ビューおよび SNAP_GET_DYN_SQL_V95 表関数 - dynsql 論理グループのスナップショット情報の検索

704 ページの『SNAPDYN_SQL 管理ビュー』と 706 ページの『SNAP_GET_DYN_SQL_V95 表関数』は、dynsql 論理データ・グループからのスナップショット情報を戻します。

SNAPDYN_SQL 管理ビュー

この管理ビューを使用すると、現在接続されているデータベースに関する dynsql 論理グループのスナップショット情報を検索できます。

このビューは、GET SNAPSHOT FOR DYNAMIC SQL ON database-alias CLP コマンドと同等の情報を戻します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、708 ページの表 181を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPDYN_SQL 管理ビューに対する SELECT 特権
- SNAPDYN_SQL 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_DYN_SQL_V95 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベースのすべてのデータベース・パーティションで実行される動的 SQL のリスト (読み取られる行の番号順に並んでいるもの) を検索します。

```
SELECT PREP_TIME_WORST, NUM_COMPILATIONS, SUBSTR(STMT_TEXT, 1, 60)
   AS STMT_TEXT, DBPARTITIONNUM
   FROM SYSIBMADM.SNAPDYN_SQL ORDER BY ROWS_READ
```

以下はこの照会の出力例です。

```
PREP_TIME_WORST      NUM_COMPILATIONS     ...
-------------------- -------------------- ...
                  98                    1 ...
                   9                    1 ...
                   0                    0 ...
                   0                    1 ...
                   0                    1 ...
                   0                    1 ...
                   0                    1 ...
                   0                    1 ...
                  40                    1 ...

  9 record(s) selected.
```

この照会からの出力 (続き)。

```
... STMT_TEXT                                                    ...
... ------------------------------------------------------------ ...
... select prep_time_worst, num_compilations, substr(stmt_text,  ...
... select * from dbuser.employee                                ...
... SET CURRENT LOCALE LC_CTYPE = 'en_US'                        ...
... select prep_time_worst, num_compilations, substr(stmt_text,  ...
... select prep_time_worst, num_compilations, substr(stmt_text,  ...
... select * from dbuser.employee                                ...
... insert into dbuser.employee values(1)                        ...
... select * from dbuser.employee                                ...
... insert into dbuser.employee values(1)                        ...
```

この照会からの出力 (続き)。

```
... DBPARTITIONNUM
... --------------
...               0
...               0
...               0
...               2
...               1
```

```
...             2
...             2
...             1
...             0
```

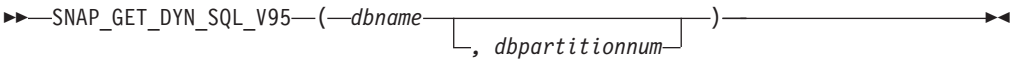
SNAP_GET_DYN_SQL_V95 表関数

SNAP_GET_DYN_SQL_V91 表関数は SNAPDYN_SQL 管理ビューと同じ情報を戻します。ただし、SNAP_GET_DYN_SQL_V95 表関数の場合は、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションについて、特定のデータベースの情報を取り出すことができます。

この表関数は、GET SNAPSHOT FOR DYNAMIC SQL ON database-alias CLP コマンドと同等の情報を戻します。

戻される可能性のある情報の完全なリストは、 708 ページの表 181を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL または空ストリングを指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプを持つファイルが存在しない場合には、現在接続されているデ

ータベースのスナップショットとデータベース・パーティション番号を SNAP_GET_DYN_SQL_V95 表関数が取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_DYN_SQL_V95 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベースの現在接続されているデータベース・パーティションで実行される動的 SQL のリスト (読み取られる行の番号順に並んでいるもの) を検索します。

```
SELECT PREP_TIME_WORST, NUM_COMPILATIONS, SUBSTR(STMT_TEXT, 1, 60)
   AS STMT_TEXT FROM TABLE(SNAP_GET_DYN_SQL_V95('',-1)) as T
    ORDER BY ROWS_READ
```

以下はこの照会の出力例です。

```
PREP_TIME_WORST      ...
-------------------- ...
                   0 ...
                   3 ...
                     ...
                   4 ...
                     ...
                   4 ...
                     ...
                   4 ...
                     ...
                   3 ...
                     ...
                   4 ...
                     ...
```

この照会からの出力 (続き)。

```
... NUM_COMPILATIONS     STMT_TEXT
... -------------------- ----------------------------------------...-
...                     0 SET CURRENT LOCALE LC_CTYPE = 'en_US'
...                     1 select rows_read, rows_written,
...                          substr(stmt_text, 1, 40) as
...                     1 select * from table
...                          (snap_get_dyn_sqlv9('',-1)) as t
...                     1 select * from table
...                          (snap_getdetaillog9('',-1)) as t
...                     1 select * from table
...                          (snap_get_hadr('',-1)) as t
...                     1 select prep_time_worst, num_compilations,
```

```
...                          substr(stmt_text,
...                    1 select prep_time_worst, num_compilations,
...                          substr(stmt_text,
```

ワークロードの実行後に、ユーザーは次の照会を表関数とともに使用できます。

```
SELECT STATS_FABRICATE_TIME,SYNC_RUNSTATS_TIME
   FROM TABLE (SNAP_GET_DYN_SQL_V95('mytestdb', -1))
   AS SNAPDB

STATS_FABRICATE_TIME   SYNC_RUNSTATS_TIME
---------------------- ------------------
                     2                  12
                     1                  30
```

この表関数に基づくビューの場合:

```
SELECT STATS_FABRICATE_TIME,SYNC_RUNSTATS_TIME
   FROM SYSIBMADM.SNAPDYN_SQL

STATS_FABRICATE_TIME   SYNC_RUNSTATS_TIME
---------------------- ------------------
                     5                  10
                     3                  20
  2 record(s) selected.
```

戻される情報

表 211. SNAPDYN_SQL 管理ビューおよび SNAP_GET_DYN_SQL_V95 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
NUM_EXECUTIONS	BIGINT	num_executions - ステートメント実行回数
NUM_COMPILATIONS	BIGINT	num_compilations - ステートメント・コンパイル数
PREP_TIME_WORST	BIGINT	prep_time_worst - ステートメント最長準備時間
PREP_TIME_BEST	BIGINT	prep_time_best - ステートメント最短準備時間
INT_ROWS_DELETED	BIGINT	int_rows_deleted - 削除された内部行数
INT_ROWS_INSERTED	BIGINT	int_rows_inserted - 挿入された内部行数
INT_ROWS_UPDATED	BIGINT	int_rows_updated - 更新された内部行数
ROWS_READ	BIGINT	rows_read - 読み取り行数
ROWS_WRITTEN	BIGINT	rows_written - 書き込み行数
STMT_SORTS	BIGINT	stmt_sorts - ステートメント・ソート回数
SORT_OVERFLOWS	BIGINT	sort_overflows - ソート・オーバーフロー
TOTAL_SORT_TIME	BIGINT	total_sort_time - ソート時間合計
POOL_DATA_L_READS	BIGINT	pool_data_l_reads - バッファー・プール・データの論理読み取り
POOL_DATA_P_READS	BIGINT	pool_data_p_reads - バッファー・プール・データの物理読み取り
POOL_TEMP_DATA_L_READS	BIGINT	pool_temp_data_l_reads - バッファー・プール一時データの論理読み取り
POOL_TEMP_DATA_P_READS	BIGINT	pool_temp_data_p_reads - バッファー・プール一時データの物理読み取り

表 211. SNAPDYN_SQL 管理ビューおよび SNAP_GET_DYN_SQL_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
POOL_INDEX_L_READS	BIGINT	pool_index_l_reads - バッファー・プール索引の論理読み取り
POOL_INDEX_P_READS	BIGINT	pool_index_p_reads - バッファー・プール索引の物理読み取り
POOL_TEMP_INDEX_L_READS	BIGINT	pool_temp_index_l_reads - バッファー・プール一時索引の論理読み取り
POOL_TEMP_INDEX_P_READS	BIGINT	pool_temp_index_p_reads - バッファー・プール一時索引の物理読み取り
POOL_XDA_L_READS	BIGINT	pool_xda_l_reads - バッファー・プール XDA データの論理読み取り
POOL_XDA_P_READS	BIGINT	pool_xda_p_reads - バッファー・プール XDA データの物理読み取り
POOL_TEMP_XDA_L_READS	BIGINT	pool_temp_xda_l_reads - バッファー・プール一時 XDA データの論理読み取り
POOL_TEMP_XDA_P_READS	BIGINT	pool_temp_xda_p_reads - バッファー・プール一時 XDA データの物理読み取り : モニター・エレメント
TOTAL_EXEC_TIME	BIGINT	total_exec_time - ステートメント実行の経過時間 (秒単位)*
TOTAL_EXEC_TIME_MS	BIGINT	total_exec_time - ステートメント実行の経過時間 (小数部、マイクロ秒単位)*
TOTAL_USR_CPU_TIME	BIGINT	total_usr_cpu_time - ステートメントのユーザー CPU の合計 (秒単位)*
TOTAL_USR_CPU_TIME_MS	BIGINT	total_usr_cpu_time - ステートメントのユーザー CPU の合計 (小数部、マイクロ秒単位)*
TOTAL_SYS_CPU_TIME	BIGINT	total_sys_cpu_time - ステートメントのシステム CPU の合計 (秒単位)*
TOTAL_SYS_CPU_TIME_MS	BIGINT	total_sys_cpu_time - ステートメントのシステム CPU の合計 (小数部、マイクロ秒単位)*
STMT_TEXT	CLOB(2 M)	stmt_text - SQL 動的ステートメント・テキスト
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。
STATS_FABRICATE_TIME	BIGINT	動的ステートメントの照会コンパイル中に表または索引のスキャンを実行しないで、システムが必要とされる統計を作成するのに費やす合計時間 (ミリ秒)。
SYNC_RUNSTATS_TIME	BIGINT	動的ステートメントの照会コンパイル中に同期 statistics-collect アクティビティーに費やされる合計時間 (ミリ秒)。
* この列の元になるモニター・エレメントの合計消費時間を計算するには、このモニター・エレメントに関する _S で終わっている列で報告されている整数秒と、このモニター・エレメントに関する _MS で終わっている列に報告されている小数秒を、次の式を使用して加算しなければなりません: (*monitor-element-name*_S × 1,000,000 + *monitor-element-name*_MS) ÷ 1,000,000. 例えば、(ELAPSED_EXEC_TIME_S × 1,000,000 + ELAPSED_EXEC_TIME_MS) ÷ 1,000,000。		

SNAPFCM 管理ビューおよび SNAP_GET_FCM 表関数 - fcm 論理データ・グループ・スナップショット情報の検索

SNAPFCM 管理ビューおよび SNAP_GET_FCM 表関数は、データベース・マネージャー・スナップショットから高速コミュニケーション・マネージャーについて、特に fcm 論理データ・グループについての情報を戻します。

SNAPFCM 管理ビュー

SNAPDBM、SNAPDBM_MEMORY_POOL、SNAPFCM_PART、および SNAPSWITCHES 管理ビューと共に使用すると、SNAPFCM 管理ビューは、GET SNAPSHOT FOR DBM コマンドと同等のデータを提供します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 712 ページの表 182を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPFCM 管理ビューに対する SELECT 特権
- SNAPFCM 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_FCM 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

すべてのデータベース・パーティションにおける高速コミュニケーション・マネージャーのメッセージ・バッファーについての情報を検索します。

```
SELECT BUFF_FREE, BUFF_FREE_BOTTOM, DBPARTITIONNUM
    FROM SYSIBMADM.SNAPFCM ORDER BY DBPARTITIONNUM
```

以下はこの照会の出力例です。

```
BUFF_FREE        BUFF_FREE_BOTTOM     DBPARTITIONNUM
----------...---- -------------------- --------------
            5120                 5100              0
            5120                 5100              1
            5120                 5100              2
```

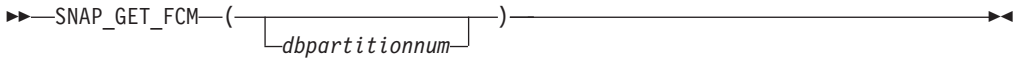
SNAP_GET_FCM 表関数

SNAP_GET_FCM 表関数は SNAPFCM 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションを対象とした情報を検索することができます。

SNAP_GET_DBM_V95、 SNAP_GET_DBM_MEMORY_POOL、SNAP_GET_FCM_PART、および SNAP_GET_SWITCHES 表関数と共に使用すると、SNAP_GET_FCM 表関数は、GET SNAPSHOT FOR DBM コマンドと同等のデータを提供します。

戻される可能性のある情報の完全なリストは、 712 ページの表 182を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。この入力オプションが使用されない場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbpartitionnum が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプが含まれるファイルが存在しない場合、SNAP_GET_FCM 表関数は、現在接続されているデータベースのスナップショットとデータベース・パーティション番号を取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_FCM 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

データベース・パーティション 1 における高速コミュニケーション・マネージャーのメッセージ・バッファーについての情報を検索します。

```
SELECT BUFF_FREE, BUFF_FREE_BOTTOM, DBPARTITIONNUM
   FROM TABLE(SYSPROC.SNAP_GET_FCM( 1 )) AS T
```

以下はこの照会の出力例です。

```
BUFF_FREE            BUFF_FREE_BOTTOM     DBPARTITIONNUM
-------------------- -------------------- --------------
                5120                 5100              1
```

戻される情報

表 212. SNAPFCM 管理ビューおよび SNAP_GET_FCM 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
BUFF_FREE	BIGINT	buff_free - 現在空いている FCM バッファー
BUFF_FREE_BOTTOM	BIGINT	buff_free_bottom - 空き FCM バッファーの最小数
CH_FREE	BIGINT	ch_free - 現在空いているチャネル
CH_FREE_BOTTOM	BIGINT	ch_free_bottom - 空いているチャネルの最小
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

SNAPFCM_PART 管理ビューおよび SNAP_GET_FCM_PART 表関数 - fcm_node 論理データ・グループ・スナップショット情報の検索

SNAPFCM_PART 管理ビューおよび SNAP_GET_FCM_PART 表関数は、データベース・マネージャー・スナップショットから、特に fcm_node 論理データ・グループの、高速コミュニケーション・マネージャー情報を戻します。

SNAPFCM_PART 管理ビュー

SNAPDBM、SNAPDBM_MEMORY_POOL、SNAPFCM、および SNAPSWITCHES 管理ビューと共に使用すると、SNAPFCM_PART 管理ビューは、GET SNAPSHOT FOR DBM コマンドと同等のデータを提供します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 715 ページの表 183を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPFCM_PART 管理ビューに対する SELECT 特権
- SNAPFCM_PART 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_FCM_PART 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

高速コミュニケーション・マネージャーのバッファー送受信情報を検索します。

```
SELECT CONNECTION_STATUS, TOTAL_BUFFERS_SENT, TOTAL_BUFFERS_RECEIVED
   FROM SYSIBMADM.SNAPFCM_PART WHERE DBPARTITIONNUM = 0
```

以下はこの照会の出力例です。

```
CONNECTION_STATUS    TOTAL_BUFFERS_SENT   TOTAL_BUFFERS_RCVD
-------------------- -------------------- --------------------
INACTIVE                               2                    1

  1 record(s) selected.
```

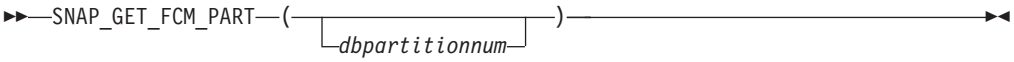
SNAP_GET_FCM_PART 表関数

SNAP_GET_FCM_PART 表関数は SNAPFCM_PART 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションを対象とした情報を検索することができます。

SNAP_GET_DBM_V95、 SNAP_GET_DBM_MEMORY_POOL、 SNAP_GET_FCM、および SNAP_GET_SWITCHES 表関数と共に使用すると、SNAP_GET_FCM_PART 表関数は、GET SNAPSHOT FOR DBM コマンドと同等のデータを提供します。

戻される可能性のある情報の完全なリストは、 715 ページの表 183を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のパーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。この入力オプションが使用されない場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbpartitionnum が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプが含まれるファイルが存在しない場合、SNAP_GET_FCM_PART 表関数は、現在接続されているデータベースのスナップショットとデータベース・パーティション番号を取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_FCM_PART 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

すべてのデータベース・パーティションにおける高速コミュニケーション・マネージャーのバッファー送受信情報を検索します。

```
SELECT FCM_DBPARTITIONNUM, TOTAL_BUFFERS_SENT, TOTAL_BUFFERS_RCVD,
   DBPARTITIONNUM FROM TABLE(SNAP_GET_FCM_PART()) AS T
   ORDER BY DBPARTITIONNUM
```

以下はこの照会の出力例です。

```
FCM_DBPARTITIONNUM TOTAL_BUFFERS_SENT   TOTAL_BUFFERS_RCVD   DBPARTITIONNUM
------------------ -------------------- -------------------- --------------
                 0                  305                  305              0
                 1                 5647                 1664              0
```

```
2                5661                1688                0
0                  19                  19                1
1                 305                 301                1
2                1688                5661                1
0                1664                5647                2
1                  10                  10                2
2                 301                 305                2
```

戻される情報

表 213. SNAPFCM_PART 管理ビューおよび SNAP_GET_FCM_PART 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
CONNECTION_STATUS	VARCHAR(10)	connection_status - 接続状況。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • INACTIVE • ACTIVE • CONGESTED
TOTAL_BUFFERS_SENT	BIGINT	total_buffers_sent - 送信された FCM バッファーの合計
TOTAL_BUFFERS_RCVD	BIGINT	total_buffers_rcvd - 受信された FCM バッファーの合計
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。
FCM_DBPARTITIONNUM	SMALLINT	データの送信先または受信元のデータベース・パーティション番号 (TOTAL_BUFFERS_SENT および TOTAL_BUFFERS_RCVD 列ごとに)。

SNAPHADR 管理ビューおよび SNAP_GET_HADR 表関数 - hadr 論理データ・グループのスナップショット情報の検索

SNAPHADR 管理ビューおよび SNAP_GET_HADR 表関数は、データベース・スナップショットから、特に hadr 論理データ・グループの高可用性災害時リカバリー情報を戻します。

SNAPHADR 管理ビュー

この管理ビューを使用して、現在接続中のデータベースに関する hadr 論理データ・グループのスナップショット情報を取得することができます。データベースが 1 次またはスタンバイ高可用性災害時リカバリー (HADR) データベースの場合にのみ、このビューによってデータが戻されます。

SNAPHADR 管理ビューを SNAPDB、 SNAPDB_MEMORY_POOL、 SNAPDETAILLOG、および SNAPSTORAGE_PATHS 管理ビューとともに使用すると、GET SNAPSHOT FOR DATABASE ON database-alias CLP コマンドに相当する情報が提供されます。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 718 ページの表 184を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPHADR 管理ビューに対する SELECT 特権
- SNAPHADR 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_HADR 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

1 次 HADR データベース上の HADR に関する構成および状況情報を取得します。

```
SELECT SUBSTR(DB_NAME, 1, 8) AS DBNAME, HADR_ROLE, HADR_STATE,
   HADR_SYNCMODE, HADR_CONNECT_STATUS
   FROM SYSIBMADM.SNAPHADR
```

以下はこの照会の出力例です。

```
DBNAME   HADR_ROLE HADR_STATE     HADR_SYNCMODE HADR_CONNECT_STATUS
-------- --------- -------------- ------------- ------------------
SAMPLE   PRIMARY   PEER           SYNC          CONNECTED

  1 record(s) selected.
```

SNAP_GET_HADR 表関数

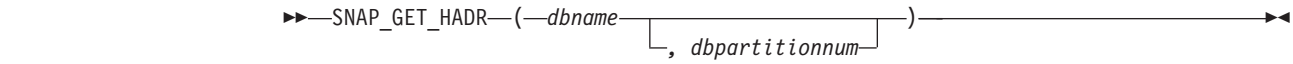
SNAP_GET_HADR 表関数は、SNAPHADR 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションに関する情報を取得することができます。

SNAP_GET_HADR 表関数を SNAP_GET_DB_V95、 SNAP_GET_DB_MEMORY_POOL、 SNAP_GET_DETAILLOG_V91、および

SNAP_GET_STORAGE_PATHS 表関数とともに使用すると、GET SNAPSHOT FOR ALL DATABASES CLP コマンドに相当する情報が提供されます。

戻される可能性のある情報の完全なリストは、 718 ページの表 184を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、空ストリングを指定します。現在接続されているデータベースと同じインスタンス内のすべてのデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。 *dbname* が *NULL* に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプを持つファイルが存在しない場合、SNAP_GET_HADR 表関数は、現在接続中のデータベースおよびデータベース・パーティション番号のスナップショットを取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_HADR 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

すべてのデータベースの HADR に関する構成および状況情報を取得します。

```
SELECT SUBSTR(DB_NAME, 1, 8) AS DBNAME, HADR_ROLE, HADR_STATE,
   HADR_SYNCMODE, HADR_CONNECT_STATUS
   FROM TABLE (SNAP_GET_HADR (CAST (NULL as VARCHAR(128)), 0)) as T
```

以下はこの照会の出力例です。

```
DBNAME   HADR_ROLE HADR_STATE     HADR_SYNCMODE HADR_CONNECT_STATUS
-------- --------- -------------- ------------- --------------------
SAMPLE   PRIMARY   PEER           SYNC          CONNECTED
TESTDB   PRIMARY   DISCONNECTED   NEARSYNC      DISCONNECTED

2 record(s) selected.
```

戻される情報

表 214. SNAPHADR 管理ビューおよび SNAP_GET_HADR 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
DB_NAME	VARCHAR(128)	db_name - データベース名
HADR_ROLE	VARCHAR(10)	hadr_role - HADR のロール。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • PRIMARY • STANDARD • STANDBY
HADR_STATE	VARCHAR(14)	hadr_state - HADR の状態。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • DISCONNECTED • LOCAL_CATCHUP • PEER • REM_CATCH_PEN • REM_CATCHUP

表 214. SNAPHADR 管理ビューおよび SNAP_GET_HADR 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
HADR_SYNCMODE	VARCHAR(10)	hadr_syncmode - HADR 同期モード。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • ASYNC • NEARSYNC • SYNC
HADR_CONNECT_STATUS	VARCHAR(12)	hadr_connect_status - HADR 接続状況。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • CONGESTED • CONNECTED • DISCONNECTED
HADR_CONNECT_TIME	TIMESTAMP	hadr_connect_time - HADR 接続時刻
HADR_HEARTBEAT	INTEGER	hadr_heartbeat - HADR ハートビート
HADR_LOCAL_HOST	VARCHAR(255)	hadr_local_host - HADR ローカル・ホスト
HADR_LOCAL_SERVICE	VARCHAR(40)	hadr_local_service - HADR ローカル・サービス
HADR_REMOTE_HOST	VARCHAR(255)	hadr_remote_host - HADR リモート・ホスト
HADR_REMOTE_SERVICE	VARCHAR(40)	hadr_remote_service - HADR リモート・サービス
HADR_REMOTE_INSTANCE	VARCHAR(128)	hadr_remote_instance - HADR リモート・インスタンス
HADR_TIMEOUT	BIGINT	hadr_timeout - HADR タイムアウト
HADR_PRIMARY_LOG_FILE	VARCHAR(255)	hadr_primary_log_file - HADR 1 次ログ・ファイル
HADR_PRIMARY_LOG_PAGE	BIGINT	hadr_primary_log_page - HADR 1 次ログ・ページ
HADR_PRIMARY_LOG_LSN	BIGINT	hadr_primary_log_lsn - HADR 1 次ログ LSN
HADR_STANDBY_LOG_FILE	VARCHAR(255)	hadr_standby_log_file - HADR スタンバイ・ログ・ファイル
HADR_STANDBY_LOG_PAGE	BIGINT	hadr_standby_log_page - HADR スタンバイ・ログ・ページ

表 214. SNAPHADR 管理ビューおよび SNAP_GET_HADR 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
HADR_STANDBY_LOG_LSN	BIGINT	hadr_standby_log_lsn - HADR スタンバイ・ログ LSN
HADR_LOG_GAP	BIGINT	hadr_log_gap - HADR ログ・ギャップ
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

SNAPLOCK 管理ビューおよび SNAP_GET_LOCK 表関数 - lock 論理データ・グループのスナップショット情報の検索

注: この管理ビューおよび表関数は非推奨になり、以下のものに置き換えられました。すなわち、 460 ページの『MON_GET_APPL_LOCKWAIT - アプリケーションが待機しているロックについての情報の収集』、 490 ページの『MON_GET_LOCKS - 現在接続されているデータベース内のすべてのロックのリスト』、および 425 ページの『MON_FORMAT_LOCK_NAME - 内部ロック名のフォーマット設定と詳細の出力』。

SNAPLOCK 管理ビューおよび SNAP_GET_LOCK 表関数は、特に lock 論理データ・グループの、ロック・スナップショット情報を戻します。

SNAPLOCK 管理ビュー

この管理ビューでは、現在接続されているデータベースの lock 論理データ・グループ・スナップショット情報を検索できます。

SNAPLOCKWAIT 管理ビューと共に使用すると、SNAPLOCK 管理ビューは、GET SNAPSHOT FOR LOCKS ON database-alias CLP コマンドと同等の情報を提供します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 722 ページの表 185を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPLOCK 管理ビューに対する SELECT 特権
- SNAPLOCK 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_LOCK 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベースのデータベース・パーティション 0 のロック情報を検索します。

```
SELECT AGENT_ID, LOCK_OBJECT_TYPE, LOCK_MODE, LOCK_STATUS
   FROM SYSIBMADM.SNAPLOCK WHERE DBPARTITIONNUM = 0
```

以下はこの照会の出力例です。

```
AGENT_ID             LOCK_OBJECT_TYPE LOCK_MODE LOCK_STATUS
-------------------- ---------------- --------- -----------
                   7 TABLE            IX        GRNT

  1 record(s) selected.
```

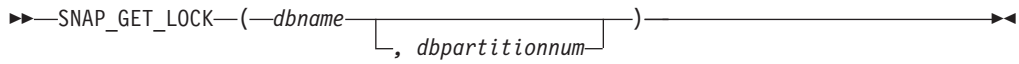
SNAP_GET_LOCK 表関数

SNAP_GET_LOCK 表関数は SNAPLOCK 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションのいずれかの特定のデータベースを対象とした情報を検索することができます。

SNAP_GET_LOCKWAIT 表関数と共に使用すると、SNAP_GET_LOCK 表関数は、GET SNAPSHOT FOR LOCKS ON database-alias CLP コマンドと同等の情報を提供します。

戻される可能性のある情報の完全なリストは、 722 ページの表 185を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL 値または空ストリングを指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプが含まれるファイルが存在しない場合、SNAP_GET_LOCK 表関数は、現在接続されているデータベースのスナップショットとデータベース・パーティション番号を取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_LOCK 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベースの現行データベース・パーティションのロック情報を検索します。

```
SELECT AGENT_ID, LOCK_OBJECT_TYPE, LOCK_MODE, LOCK_STATUS
   FROM TABLE(SNAP_GET_LOCK('',-1)) as T
```

以下はこの照会の出力例です。

```
AGENT_ID       LOCK_OBJECT_TYPE   LOCK_MODE  LOCK_STATUS
--------...--- ------------------ ---------- -----------
           680 INTERNALV_LOCK     S          GRNT
           680 INTERNALP_LOCK     S          GRNT

2 record(s) selected.
```

戻される情報

表 215. SNAPLOCK 管理ビューおよび SNAP_GET_LOCK 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
AGENT_ID	BIGINT	agent_id - アプリケーション・ハンドル (エージェント ID)
TAB_FILE_ID	BIGINT	table_file_id - 表ファイル ID
LOCK_OBJECT_TYPE	VARCHAR(18)	lock_object_type - 待機中のロック対象タイプ。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • AUTORESIZE_LOCK • AUTOSTORAGE_LOCK • BLOCK_LOCK • EOT_LOCK • INPLACE_REORG_LOCK • INTERNAL_LOCK • INTERNALB_LOCK • INTERNALC_LOCK • INTERNALJ_LOCK • INTERNALL_LOCK • INTERNALO_LOCK • INTERNALQ_LOCK • INTERNALP_LOCK • INTERNALS_LOCK • INTERNALT_LOCK • INTERNALV_LOCK • KEYVALUE_LOCK • ROW_LOCK • SYSBOOT_LOCK • TABLE_LOCK • TABLE_PART_LOCK • TABLESPACE_LOCK • XML_PATH_LOCK

表 215. SNAPLOCK 管理ビューおよび SNAP_GET_LOCK 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
LOCK_MODE	VARCHAR(10)	lock_mode - ロック・モード。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • IN • IS • IX • NON (ロックなしの場合) • NS • NW • S • SIX • U • X • Z
LOCK_STATUS	VARCHAR(10)	lock_status - ロック状況。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • CONV • GRNT
LOCK_ESCALATION	SMALLINT	lock_escalation - ロック・エスカレーション
TABNAME	VARCHAR(128)	table_name - 表名
TABSCHEMA	VARCHAR(128)	table_schema - 表スキーマ名
TBSP_NAME	VARCHAR(128)	tablespace_name - 表スペース名

表 215. SNAPLOCK 管理ビューおよび SNAP_GET_LOCK 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
LOCK_ATTRIBUTES	VARCHAR(128)	lock_attributes - ロック属性。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。ロックがない場合、テキスト ID は NONE となり、それ以外の場合、以下のいずれかの組み合わせを '+' 記号で区切ったものとなります。 • ALLOW_NEW • DELETE_IN_BLOCK • ESCALATED • INSERT • NEW_REQUEST • RR • RR_IN_BLOCK • UPDATE_DELETE • WAIT_FOR_AVAIL
LOCK_COUNT	BIGINT	lock_count - ロック・カウント
LOCK_CURRENT_MODE	VARCHAR(10)	lock_current_mode - 移行前の元のロック・モード。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • IN • IS • IX • NON (ロックなしの場合) • NS • NW • S • SIX • U • X • Z
LOCK_HOLD_COUNT	BIGINT	lock_hold_count - ロック保留カウント
LOCK_NAME	VARCHAR(32)	lock_name - ロック名
LOCK_RELEASE_FLAGS	BIGINT	lock_release_flags - ロック保留解除フラグ

表 215. SNAPLOCK 管理ビューおよび SNAP_GET_LOCK 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
DATA_PARTITION_ID	INTEGER	data_partition_id - データ・パーティション ID。非パーティション表では、このエレメントは NULL です。
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

SNAPLOCKWAIT 管理ビューおよび SNAP_GET_LOCKWAIT 表関数 - lockwait 論理データ・グループのスナップショット情報の検索

注: この管理ビューおよび表関数は非推奨になり、以下のものに置き換えられました。すなわち、 556 ページの『MON_LOCKWAITS 管理ビュー - ロックの取得を待機しているアプリケーションに関するメトリックの取得』、および 460 ページの『MON_GET_APPL_LOCKWAIT - アプリケーションが待機しているロックについての情報の収集』、 490 ページの『MON_GET_LOCKS - 現在接続されているデータベース内のすべてのロックのリスト』、 425 ページの『MON_FORMAT_LOCK_NAME - 内部ロック名のフォーマット設定と詳細の出力』。

SNAPLOCKWAIT 管理ビューおよび SNAP_GET_LOCKWAIT 表関数は、特に lockwait 論理データ・グループの、ロック待機スナップショット情報を戻します。

SNAPLOCKWAIT 管理ビュー

この管理ビューでは、現在接続されているデータベースの lockwait 論理データ・グループ・スナップショット情報を検索できます。

SNAPLOCK 管理ビューと共に使用すると、SNAPLOCKWAIT 管理ビューは、GET SNAPSHOT FOR LOCKS ON database-alias CLP コマンドと同等の情報を提供します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 729 ページの表 186を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPLOCKWAIT 管理ビューに対する SELECT 特権
- SNAPLOCKWAIT 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_LOCKWAIT 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベースのデータベース・パーティション 0 のロック待機情報を検索します。

```
SELECT AGENT_ID, LOCK_MODE, LOCK_OBJECT_TYPE, AGENT_ID_HOLDING_LK,
   LOCK_MODE_REQUESTED FROM SYSIBMADM.SNAPLOCKWAIT
   WHERE DBPARTITIONNUM = 0
```

以下はこの照会の出力例です。

```
AGENT_ID     LOCK_MODE LOCK_OBJECT_TYPE ...
---------...- --------- ---------------- ...
           7 IX        TABLE            ...

  1 record(s) selected.
```

この照会からの出力 (続き)。

```
... AGENT_ID_HOLDING_LK  LOCK_MODE_REQUESTED
... -------------------- --------------------
...                   12 IS
```

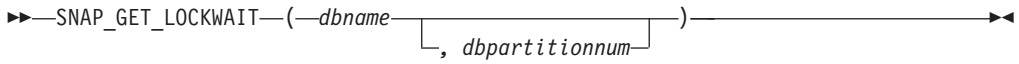
SNAP_GET_LOCKWAIT 表関数

SNAP_GET_LOCKWAIT 表関数は SNAPLOCKWAIT 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションのいずれかの特定のデータベースを対象とした情報を検索することができます。

SNAP_GET_LOCK 表関数と共に使用すると、SNAP_GET_LOCKWAIT 表関数は、GET SNAPSHOT FOR LOCKS ON database-alias CLP コマンドと同等の情報を提供します。

戻される可能性のある情報の完全なリストは、 729 ページの表 186を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL 値または空ストリングを指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプが含まれるファイルが存在しない場合、SNAP_GET_LOCKWAIT 表関数は、現在接続されているデータベースのスナップショットとデータベース・パーティション番号を取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_LOCKWAIT 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベースの現行データベース・パーティションのロック待機情報を検索します。

```
SELECT AGENT_ID, LOCK_MODE, LOCK_OBJECT_TYPE, AGENT_ID_HOLDING_LK,
   LOCK_MODE_REQUESTED FROM TABLE(SNAP_GET_LOCKWAIT('',-1)) AS T
```

以下はこの照会の出力例です。

```
AGENT_ID      LOCK_MODE  LOCK_OBJECT_TYPE   ...
---------...-- ---------- ------------------ ...
           12 X          ROW_LOCK           ...

1 record(s) selected.
```

この照会からの出力 (続き)。

```
... AGENT_ID_HOLDING_LK  LOCK_MODE_REQUESTED
... -------------------- ------------------
...                    7 X
```

使用上の注意

ロック待機情報を表示するには、まずデータベース・マネージャー構成でデフォルトの LOCK モニター・スイッチをオンにする必要があります。変更を即時に有効にするには、CLP を使用してインスタンスに明示的にアタッチし、次いで以下の CLP コマンドを発行します。

```
UPDATE DATABASE MANAGER CONFIGURATION CLP USING DFT_MON_LOCK ON
```

デフォルトの設定値も、ADMIN_CMD ストアード・プロシージャーによりオンにすることができます。以下に例を示します。

```
CALL SYSPROC.ADMIN_CMD('update dbm cfg using DFT_MON_LOCK ON')
```

ADMIN_CMD ストアード・プロシージャーを使用する場合、または事前にインスタンスにアタッチせずに CLP コマンドを使用する場合、インスタンスをリサイクルしなければ変更は有効になりません。

戻される情報

表 216. SNAPLOCKWAIT 管理ビューおよび SNAP_GET_LOCKWAIT 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
AGENT_ID	BIGINT	agent_id - アプリケーション・ハンドル (エージェント ID)
SUBSECTION_NUMBER	BIGINT	ss_number - サブセクション番号

表 216. SNAPLOCKWAIT 管理ビューおよび SNAP_GET_LOCKWAIT 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
LOCK_MODE	VARCHAR(10)	lock_mode - ロック・モード。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • IN • IS • IX • NON (ロックなしの場合) • NS • NW • S • SIX • U • X • Z

表 216. SNAPLOCKWAIT 管理ビューおよび SNAP_GET_LOCKWAIT 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
LOCK_OBJECT_TYPE	VARCHAR(18)	lock_object_type - 待機中のロック対象タイプ。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • AUTORESIZE_LOCK • AUTOSTORAGE_LOCK • BLOCK_LOCK • EOT_LOCK • INPLACE_REORG_LOCK • INTERNAL_LOCK • INTERNALB_LOCK • INTERNALC_LOCK • INTERNALJ_LOCK • INTERNALL_LOCK • INTERNALO_LOCK • INTERNALQ_LOCK • INTERNALP_LOCK • INTERNALS_LOCK • INTERNALT_LOCK • INTERNALV_LOCK • KEYVALUE_LOCK • ROW_LOCK • SYSBOOT_LOCK • TABLE_LOCK • TABLE_PART_LOCK • TABLESPACE_LOCK • XML_PATH_LOCK
AGENT_ID_HOLDING_LK	BIGINT	agent_id_holding_lock - ロックを保持しているエージェント ID
LOCK_WAIT_START_TIME	TIMESTAMP	lock_wait_start_time - ロック待機開始タイム・スタンプ

表 216. SNAPLOCKWAIT 管理ビューおよび SNAP_GET_LOCKWAIT 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
LOCK_MODE_REQUESTED	VARCHAR(10)	lock_mode_requested - 要求されているロック・モード。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • IN • IS • IX • NON (ロックなしの場合) • NS • NW • S • SIX • U • X • Z
LOCK_ESCALATION	SMALLINT	lock_escalation - ロック・エスカレーション
TABNAME	VARCHAR(128)	table_name - 表名
TABSCHEMA	VARCHAR(128)	table_schema - 表スキーマ名
TBSP_NAME	VARCHAR(128)	tablespace_name - 表スペース名
APPL_ID_HOLDING_LK	VARCHAR(128)	appl_id_holding_lk - ロックを保持しているアプリケーション ID
LOCK_ATTRIBUTES	VARCHAR(128)	lock_attributes - ロック属性。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。ロックがない場合、テキスト ID は NONE となり、それ以外の場合、以下のいずれかの組み合わせを '+' 記号で区切ったものとなります。 • ALLOW_NEW • DELETE_IN_BLOCK • ESCALATED • INSERT • NEW_REQUEST • RR • RR_IN_BLOCK • UPDATE_DELETE • WAIT_FOR_AVAIL

表 216. SNAPLOCKWAIT 管理ビューおよび SNAP_GET_LOCKWAIT 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
LOCK_CURRENT_MODE	VARCHAR(10)	lock_current_mode - 移行前の元のロック・モード。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • IN • IS • IX • NON (ロックなしの場合) • NS • NW • S • SIX • U • X • Z
LOCK_NAME	VARCHAR(32)	lock_name - ロック名
LOCK_RELEASE_FLAGS	BIGINT	lock_release_flags - ロック保留解除フラグ。
DATA_PARTITION_ID	INTEGER	data_partition_id - データ・パーティション ID。非パーティション表では、このエレメントは NULL です。
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

SNAPSTMT 管理ビューおよび SNAP_GET_STMT 表関数 - ステートメント・スナップショット情報の検索

SNAPSTMT 管理ビューおよび SNAP_GET_STMT 表関数は、アプリケーション・スナップショットから SQL または XQuery ステートメントについての情報を戻します。

SNAPSTMT 管理ビュー

この管理ビューでは、現在接続されているデータベースのステートメント・スナップショット情報を検索できます。

SNAPAGENT、 SNAPAGENT_MEMORY_POOL、 SNAPAPPL、 SNAPAPPL_INFO、および SNAPSUBSECTION 管理ビューと共に使用すると、SNAPSTMT 管理ビューは、GET SNAPSHOT FOR APPLICATIONS on

database-alias CLP コマンドと同等の情報を提供しますが、すべてのデータベース・パーティションからデータを検索します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 736 ページの表 187を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPSTMT 管理ビューに対する SELECT 特権
- SNAPSTMT 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_STMT 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されている単一パーティション・データベース上で実行されたステートメントのために読み取り、書き込み、および操作の実行が行われた行を検索します。

```
SELECT SUBSTR(STMT_TEXT,1,30) AS STMT_TEXT, ROWS_READ, ROWS_WRITTEN,
   STMT_OPERATION FROM SYSIBMADM.SNAPSTMT
```

以下はこの照会の出力例です。

```
STMT_TEXT     ROWS_READ     ROWS_WRITTEN     STMT_OPERATION
---------...- ---------...- -------------...- --------------------
-                         0                0 FETCH
-                         0                0 STATIC_COMMIT

  2 record(s) selected.
```

SNAP_GET_STMT 表関数

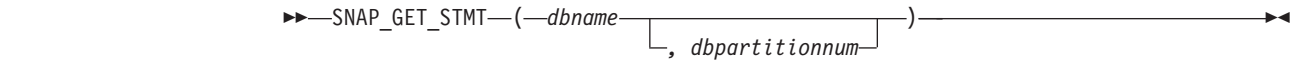
SNAP_GET_STMT 表関数は SNAPSTMT 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションのいずれかの特定のデータベースを対象とした情報を検索することができます。

SNAP_GET_AGENT、 SNAP_GET_AGENT_MEMORY_POOL、
SNAP_GET_APPL_V95、 SNAP_GET_APPL_INFO_V95、および
SNAP_GET_SUBSECTION 表関数と共に使用すると、SNAP_GET_STMT 表関数

は、GET SNAPSHOT FOR ALL APPLICATIONS CLP コマンドと同等の情報を提供しますが、すべてのデータベース・パーティションからデータを検索します。

戻される可能性のある情報の完全なリストは、 736 ページの表 187を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、空ストリングを指定します。現在接続されているデータベースと同じインスタンス内のすべてのデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプが含まれるファイルが存在しない場合、SNAP_GET_STMT 表関数は、現在接続されているデータベースのスナップショットとデータベース・パーティション番号を取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_STMT 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベースの現行データベース・パーティション上で実行されたステートメントのために読み取り、書き込み、および操作の実行が行われた行を検索します。

```
SELECT SUBSTR(STMT_TEXT,1,30) AS STMT_TEXT, ROWS_READ,
   ROWS_WRITTEN, STMT_OPERATION FROM TABLE(SNAP_GET_STMT('',-1)) AS T
```

以下はこの照会の出力例です。

```
STMT_TEXT                      ROWS_READ     ...
------------------------------ ---------...- ...
update t set a=3                           0 ...
SELECT SUBSTR(STMT_TEXT,1,30)              0 ...
-                                          0 ...
-                                          0 ...
update t set a=2                           9 ...
                                             ...
5 record(s) selected.                        ...
```

この照会からの出力 (続き)。

```
... ROWS_WRITTEN     STMT_OPERATION
... ------------...- --------------------
...                0 EXECUTE_IMMEDIATE
...                0 FETCH
...                0 NONE
...                0 NONE
...                1 EXECUTE_IMMEDIATE
...
```

戻される情報

表 217. SNAPSTMT 管理ビューおよび SNAP_GET_STMT 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
DB_NAME	VARCHAR(128)	db_name - データベース名
AGENT_ID	BIGINT	agent_id - アプリケーション・ハンドル (エージェント ID)
ROWS_READ	BIGINT	rows_read - 読み取り行数
ROWS_WRITTEN	BIGINT	rows_written - 書き込み行数
NUM_AGENTS	BIGINT	num_agents - ステートメントで作動しているエージェントの数
AGENTS_TOP	BIGINT	agents_top - 作成されたエージェントの数

表 217. SNAPSTMT 管理ビューおよび SNAP_GET_STMT 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
STMT_TYPE	VARCHAR(20)	stmt_type - ステートメント・タイプ。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • DYNAMIC • NON_STMT • STATIC • STMT_TYPE_UNKNOWN
STMT_OPERATION	VARCHAR(20)	stmt_operation/operation - ステートメント操作。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • CALL • CLOSE • COMPILE • DESCRIBE • EXECUTE • EXECUTE_IMMEDIATE • FETCH • FREE_LOCATOR • GETAA • GETNEXTCHUNK • GETTA • NONE • OPEN • PREP_COMMIT • PREP_EXEC • PREP_OPEN • PREPARE • REBIND • REDIST • REORG • RUNSTATS • SELECT • SET • STATIC_COMMIT • STATIC_ROLLBACK
SECTION_NUMBER	BIGINT	section_number - セクション番号

表 217. SNAPSTMT 管理ビューおよび SNAP_GET_STMT 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
QUERY_COST_ESTIMATE	BIGINT	query_cost_estimate - 照会コストの見積もり
QUERY_CARD_ESTIMATE	BIGINT	query_card_estimate - 照会行数の見積もり
DEGREE_PARALLELISM	BIGINT	degree_parallelism - 並列処理の度合い
STMT_SORTS	BIGINT	stmt_sorts - ステートメント・ソート回数
TOTAL_SORT_TIME	BIGINT	total_sort_time - ソート時間合計
SORT_OVERFLOWS	BIGINT	sort_overflows - ソート・オーバーフロー
INT_ROWS_DELETED	BIGINT	int_rows_deleted - 削除された内部行数
INT_ROWS_UPDATED	BIGINT	int_rows_updated - 更新された内部行数
INT_ROWS_INSERTED	BIGINT	int_rows_inserted - 挿入された内部行数
FETCH_COUNT	BIGINT	fetch_count - 成功した取り出しの数
STMT_START	TIMESTAMP	stmt_start - ステートメント操作開始タイム・スタンプ
STMT_STOP	TIMESTAMP	stmt_stop - ステートメント操作停止タイム・スタンプ
STMT_USR_CPU_TIME_S	BIGINT	stmt_usr_cpu_time - ステートメントに使用されたユーザー CPU 時間 (秒単位)*
STMT_USR_CPU_TIME_MS	BIGINT	stmt_usr_cpu_time - ステートメントに使用されたユーザー CPU 時間 (小数部、マイクロ秒単位)*
STMT_SYS_CPU_TIME_S	BIGINT	stmt_sys_cpu_time - ステートメントが使用したシステム CPU 時間 (秒単位)*
STMT_SYS_CPU_TIME_MS	BIGINT	stmt_sys_cpu_time - ステートメントが使用したシステム CPU 時間 (小数部、マイクロ秒単位)*
STMT_ELAPSED_TIME_S	BIGINT	stmt_elapsed_time - 最新のステートメント経過時間 (秒単位)*
STMT_ELAPSED_TIME_MS	BIGINT	stmt_elapsed_time - 最新のステートメント経過時間 (小数部、マイクロ秒単位)*
BLOCKING_CURSOR	SMALLINT	blocking_cursor - ブロック・カーソル

表 217. SNAPSTMT 管理ビューおよび SNAP_GET_STMT 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
STMT_NODE_NUMBER	SMALLINT	stmt_node_number - ステートメント・ノード
CURSOR_NAME	VARCHAR(128)	cursor_name - カーソル名
CREATOR	VARCHAR(128)	creator - アプリケーション作成者
PACKAGE_NAME	VARCHAR(128)	package_name - パッケージ名
STMT_TEXT	CLOB(16 M)	stmt_text - SQL 動的ステートメント・テキスト
CONSISTENCY_TOKEN	VARCHAR(128)	consistency_token - パッケージ整合性トークン
PACKAGE_VERSION_ID	VARCHAR(128)	package_version_id - パッケージ・バージョン
POOL_DATA_L_READS	BIGINT	pool_data_l_reads - バッファー・プール・データの論理読み取り
POOL_DATA_P_READS	BIGINT	pool_data_p_reads - バッファー・プール・データの物理読み取り
POOL_INDEX_L_READS	BIGINT	pool_index_l_reads - バッファー・プール索引の論理読み取り
POOL_INDEX_P_READS	BIGINT	pool_index_p_reads - バッファー・プール索引の物理読み取り
POOL_XDA_L_READS	BIGINT	pool_xda_l_reads - バッファー・プール XDA データの論理読み取り : モニター・エレメント
POOL_XDA_P_READS	BIGINT	pool_xda_p_reads - バッファー・プール XDA データの物理読み取り : モニター・エレメント
POOL_TEMP_DATA_L_READS	BIGINT	pool_temp_data_l_reads - バッファー・プール一時データの論理読み取り
POOL_TEMP_DATA_P_READS	BIGINT	pool_temp_data_p_reads - バッファー・プール一時データの物理読み取り
POOL_TEMP_INDEX_L_READS	BIGINT	pool_temp_index_l_reads - バッファー・プール一時索引の論理読み取り
POOL_TEMP_INDEX_P_READS	BIGINT	pool_temp_index_p_reads - バッファー・プール一時索引の物理読み取り
POOL_TEMP_XDA_L_READS	BIGINT	pool_temp_xda_l_reads - バッファー・プール一時 XDA データの論理読み取り
POOL_TEMP_XDA_P_READS	BIGINT	pool_temp_xda_p_reads - バッファー・プール一時 XDA データの物理読み取り : モニター・エレメント

表 217. SNAPSTMT 管理ビューおよび SNAP_GET_STMT 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。
* この列の元になるモニター・エレメントの合計消費時間を計算するには、このモニター・エレメントに関する _S で終わっている列で報告されている整数秒と、このモニター・エレメントに関する _MS で終わっている列に報告されている小数秒を、次の式を使用して加算しなければなりません: (*monitor-element-name*_S × 1,000,000 + *monitor-element-name*_MS) ÷ 1,000,000. 例えば、(ELAPSED_EXEC_TIME_S × 1,000,000 + ELAPSED_EXEC_TIME_MS) ÷ 1,000,000。		

SNAPSTORAGE_PATHS 管理ビューおよび SNAP_GET_STORAGE_PATHS 表関数 - 自動ストレージ・パスの情報の検索

注: この表関数は使用すべきではなく、 740 ページの『SNAPSTORAGE_PATHS 管理ビューおよび SNAP_GET_STORAGE_PATHS_V97 表関数 - 自動ストレージ・パスの情報の検索』 に置き換えられました。

SNAPSTORAGE_PATHS 管理ビューと SNAP_GET_STORAGE_PATHS 表関数は、データベースの自動ストレージ・パスのリストを戻します。これには、各ストレージ・パスのファイル・システムの情報、特に db_storage_group 論理データ・グループからの情報が含まれます。

SNAPSTORAGE_PATHS 管理ビュー

この管理ビューを使用すると、現在接続されているデータベースに関する自動ストレージ・パスの情報を検索できます。

SNAPSTORAGE_PATHS 管理ビューを SNAPDB、 SNAPDETAILLOG、SNAPHADR、および SNAPDB_MEMORY_POOL 管理ビューと併せて使用することにより、GET SNAPSHOT FOR DATABASE ON database-alias CLP コマンドと同等の情報を戻します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 895 ページの表 218を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPSTORAGE_PATHS 管理ビューに対する SELECT 特権
- SNAPSTORAGE_PATHS 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_STORAGE_PATHS 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されている単一パーティション・データベースのストレージ・パスを検索します。

```
SELECT SUBSTR(DB_NAME,1,8) AS DB_NAME, SUBSTR(DB_STORAGE_PATH,1,8)
   AS DB_STORAGE_PATH FROM SYSIBMADM.SNAPSTORAGE_PATHS
```

以下はこの照会の出力例です。

```
DB_NAME  DB_STORAGE_PATH
-------- ---------------
STOPATH  d:

  1 record(s) selected.
```

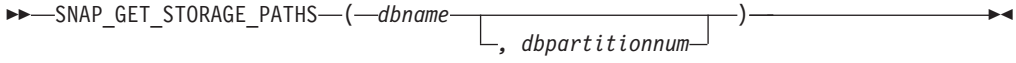
SNAP_GET_STORAGE_PATHS 表関数

SNAP_GET_STORAGE_PATHS 表関数は SNAPSTORAGE_PATHS 管理ビューと同じ情報を戻します。ただし、SNAP_GET_STORAGE_PATHS 表関数の場合は、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションについて、特定のデータベースの情報を検索することができます。

SNAP_GET_STORAGE_PATHS 表関数を SNAP_GET_DB_V95、SNAP_GET_DETAILLOG_V91、 SNAP_GET_HADR、および SNAP_GET_DB_MEMORY_POOL 表関数と併せて使用することにより、GET SNAPSHOT FOR ALL DATABASES CLP コマンドと同等の情報を戻します。

戻される可能性のある情報の完全なリストは、 895 ページの表 218を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、空ストリングを指定します。現在接続されているデータベースと同じインスタンス内のすべてのデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプを持つファイルが存在しない場合には、現在接続されているデータベースのスナップショットとデータベース・パーティション番号を SNAP_GET_STORAGE_PATHS 表関数が取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_STORAGE_PATHS 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

すべてのアクティブ・データベースに関するストレージ・パスの情報を取り出します。

```
SELECT SUBSTR(DB_NAME,1,8) AS DB_NAME, DB_STORAGE_PATH
   FROM TABLE(SNAP_GET_STORAGE_PATHS(CAST (NULL AS VARCHAR(128)), -1)) AS T
```

以下はこの照会の出力例です。

```
DB_NAME  DB_STORAGE_PATH
-------- -------------------...
STOPATH  /home/jessicae/sdb
MYDB     /home/jessicae/mdb

  2 record(s) selected
```

戻される情報

ファイル・システムの情報を戻すためには、BUFFERPOOL モニター・スイッチをオンにする必要があります。

表 218. SNAPSTORAGE_PATHS 管理ビューおよび SNAP_GET_STORAGE_PATHS 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
DB_NAME	VARCHAR(128)	db_name - データベース名
DB_STORAGE_PATH	VARCHAR(256)	db_storage_path - 自動ストレージ・パス
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。
FS_ID	VARCHAR(22)	fs_id - 固有のファイル・システム識別番号
FS_TOTAL_SIZE	BIGINT	fs_total_size - ファイル・システムの合計サイズ
FS_USED_SIZE	BIGINT	fs_used_size - ファイル・システム上で使用されるスペースの量
STO_PATH_FREE_SIZE	BIGINT	sto_path_free_sz - 自動ストレージ・パスのフリー・スペース

SNAPSUBSECTION 管理ビューおよび SNAP_GET_SUBSECTION 表関数 - subsection 論理モニター・グループ・スナップショット情報の検索

SNAPSUBSECTION 管理ビューおよび SNAP_GET_SUBSECTION 表関数は、アプリケーション・サブセクション情報として、subsection 論理モニター・グループの情報を戻します。

SNAPSUBSECTION 管理ビュー

この管理ビューでは、現在接続されているデータベースの subsection 論理モニター・グループ・スナップショット情報を検索できます。

SNAPAGENT、 SNAPAGENT_MEMORY_POOL、 SNAPAPPL、 SNAPAPPL_INFO、および SNAPSTMT 管理ビューと共に使用すると、SNAPSUBSECTION 管理ビューは、GET SNAPSHOT FOR APPLICATIONS on database-alias CLP コマンドと同等の情報を提供しますが、すべてのデータベース・パーティションからデータを検索します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 746 ページの表 189を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPSUBSECTION 管理ビューに対する SELECT 特権
- SNAPSUBSECTION 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_SUBSECTION 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

すべてのデータベース・パーティションで実行するサブセクションの状況を取得します。

```
SELECT DB_NAME, STMT_TEXT, SS_STATUS, DBPARTITIONNUM
   FROM SYSIBMADM.SNAPSUBSECTION
   ORDER BY DB_NAME, SS_STATUS, DBPARTITIONNUM
```

以下はこの照会の出力例です。

```
DB_NAME     STMT_TEXT                 SS_STATUS     DBPARTITIONNUM
--------...- ----------------------...- ---------...- --------------
SAMPLE      select * from EMPLOYEE    EXEC                       0
SAMPLE      select * from EMPLOYEE    EXEC                       1
```

SNAP_GET_SUBSECTION 表関数

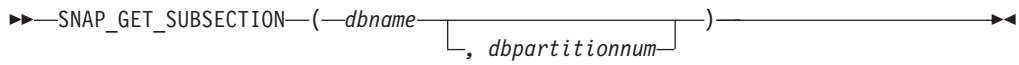
SNAP_GET_SUBSECTION 表関数は SNAPSUBSECTION 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションのいずれかの特定のデータベースを対象とした情報を検索することができます。

戻される可能性のある情報の完全なリストは、 746 ページの表 189を参照してください。

SNAP_GET_AGENT、 SNAP_GET_AGENT_MEMORY_POOL、 SNAP_GET_APPL_V95、 SNAP_GET_APPL_INFO_V95、および SNAP_GET_STMT 表関数と共に使用すると、SNAP_GET_SUBSECTION 表関数は、GET SNAPSHOT FOR ALL APPLICATIONS CLP コマンドと同等の情報を提供しますが、すべての

データベース・パーティションからデータを検索します。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、空ストリングを指定します。現在接続されているデータベースと同じインスタンス内のすべてのデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。 *dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプが含まれるファイルが存在しない場合、SNAP_GET_SUBSECTION 表関数は、現在接続されているデータベースのスナップショットとデータベース・パーティション番号を取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_SUBSECTION 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL

- SYSMAINT
- SYSADM

例

すべてのデータベース・パーティションで実行するサブセクションの状況を取得します。

```
SELECT DB_NAME, STMT_TEXT, SS_STATUS, DBPARTITIONNUM
   FROM TABLE(SYSPROC.SNAP_GET_SUBSECTION( '', 0 )) as T
   ORDER BY DB_NAME, SS_STATUS, DBPARTITIONNUM
```

以下はこの照会の出力例です。

```
DB_NAME     STMT_TEXT                 SS_STATUS     DBPARTITIONNUM
--------...- ----------------------...- ---------...- --------------
SAMPLE      select * from EMPLOYEE     EXEC                       0
SAMPLE      select * from EMPLOYEE     EXEC                       1
```

戻される情報

表 219. SNAPSUBSECTION 管理ビューおよび SNAP_GET_SUBSECTION 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
DB_NAME	VARCHAR(128)	db_name - データベース名
STMT_TEXT	CLOB(16 M)	stmt_text - SQL 動的ステートメント・テキスト
SS_EXEC_TIME	BIGINT	ss_exec_time - サブセクション実行経過時間
TQ_TOT_SEND_SPILLS	BIGINT	tq_tot_send_spills - オーバーフローした表キュー・バッファーの合計数
TQ_CUR_SEND_SPILLS	BIGINT	tq_cur_send_spills - オーバーフローした表キュー・バッファーの現在数
TQ_MAX_SEND_SPILLS	BIGINT	tq_max_send_spills - 表キュー・バッファー・オーバーフローの最大数
TQ_ROWS_READ	BIGINT	tq_rows_read - 表キューから読み取られた行数
TQ_ROWS_WRITTEN	BIGINT	tq_rows_written - 表キューに書き込まれた行数
ROWS_READ	BIGINT	rows_read - 読み取り行数
ROWS_WRITTEN	BIGINT	rows_written - 書き込み行数
SS_USR_CPU_TIME_S	BIGINT	ss_usr_cpu_time - サブセクションに使用されたユーザー CPU 時間 (秒単位)*
SS_USR_CPU_TIME_MS	BIGINT	ss_usr_cpu_time - サブセクションに使用されたユーザー CPU 時間 (小数部、マイクロ秒単位)*

表 219. SNAPSUBSECTION 管理ビューおよび SNAP_GET_SUBSECTION 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
SS_SYS_CPU_TIME_S	BIGINT	ss_sys_cpu_time - サブセクションに使用されたシステム CPU 時間 (秒単位)*
SS_SYS_CPU_TIME_MS	BIGINT	ss_sys_cpu_time - サブセクションに使用されたシステム CPU 時間 (小数部、マイクロ秒単位)*
SS_NUMBER	INTEGER	ss_number - サブセクション番号
SS_STATUS	VARCHAR(20)	ss_status - サブセクションの状況。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • EXEC • TQ_WAIT_TO_RCV • TQ_WAIT_TO_SEND • COMPLETED
SS_NODE_NUMBER	SMALLINT	ss_node_number - サブセクション・ノード番号
TQ_NODE_WAITED_FOR	SMALLINT	tq_node_waited_for - 表キュー上のノード待機
TQ_WAIT_FOR_ANY	INTEGER	tq_wait_for_any - 表キュー上のノード送信待機
TQ_ID_WAITING_ON	INTEGER	tq_id_waiting_on - ノード上の表キュー待機
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。
* この列の元になるモニター・エレメントの合計消費時間を計算するには、このモニター・エレメントに関する _S で終わっている列で報告されている整数秒と、このモニター・エレメントに関する _MS で終わっている列に報告されている小数秒を、次の式を使用して加算しなければなりません: (*monitor-element-name*_S × 1,000,000 + *monitor-element-name*_MS) ÷ 1,000,000. 例えば、(ELAPSED_EXEC_TIME_S × 1,000,000 + ELAPSED_EXEC_TIME_MS) ÷ 1,000,000。		

SNAPSWITCHES 管理ビューおよび SNAP_GET_SWITCHES 表関数 - データベース・スナップショットのスイッチ状態情報の検索

SNAPSWITCHES 管理ビューおよび SNAP_GET_SWITCHES 表関数は、データベース・スナップショットのスイッチ状態に関する情報を戻します。

SNAPSWITCHES 管理ビュー

このビューは、GET DBM MONITOR SWITCHES CLP コマンドと同等のデータを提供します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、750 ページの表 190を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPSWITCHES 管理ビューに対する SELECT 特権
- SNAPSWITCHES 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_SWITCHES 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

すべてのデータベース・パーティションにおける DBM モニターのスイッチ状態情報を検索します。

```
SELECT UOW_SW_STATE, STATEMENT_SW_STATE, TABLE_SW_STATE, BUFFPOOL_SW_STATE,
   LOCK_SW_STATE, SORT_SW_STATE, TIMESTAMP_SW_STATE,
   DBPARTITIONNUM FROM SYSIBMADM.SNAPSWITCHES
```

以下はこの照会の出力例です。

```
UOW_SW_STATE STATEMENT_SW_STATE TABLE_SW_STATE BUFFPOOL_SW_STATE ...
------------ ------------------ -------------- ----------------- ...
           0                  0              0                 0 ...
           0                  0              0                 0 ...
           0                  0              0                 0 ...
                                                                  ...
  3 record selected.
```

この照会からの出力 (続き)。

```
... LOCK_SW_STATE SORT_SW_STATE TIMESTAMP_SW_STATE DBPARTITIONNUM
... ------------- ------------- ------------------ --------------
...             1             0                  1              0
...             1             0                  1              1
...             1             0                  1              2
```

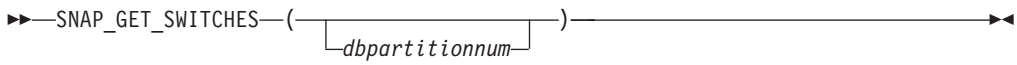
SNAP_GET_SWITCHES 表関数

SNAP_GET_SWITCHES 表関数は SNAPSWITCHES 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションを対象とした情報を検索することができます。

この表関数は、GET DBM MONITOR SWITCHES CLP コマンドと同等のデータを提供します。

戻される可能性のある情報の完全なリストは、 750 ページの表 190を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。この入力オプションが使用されない場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbpartitionnum が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプが含まれるファイルが存在しない場合、SNAP_GET_SWITCHES 表関数は、現在接続されているデータベースのスナップショットとデータベース・パーティション番号を取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_SWITCHES 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現行データベース・パーティションにおける DBM モニターのスイッチ状態情報を検索します。

```
SELECT UOW_SW_STATE, STATEMENT_SW_STATE, TABLE_SW_STATE,
   BUFFPOOL_SW_STATE,LOCK_SW_STATE, SORT_SW_STATE, TIMESTAMP_SW_STATE
   FROM TABLE(SNAP_GET_SWITCHES(-1)) AS T
```

以下はこの照会の出力例です。

```
UOW_SW_STATE STATEMENT_SW_STATE TABLE_SW_STATE...
------------ ----------------- --------------...
           1                 1              1...
                                             ...
1 record(s) selected.                        ...
```

この照会からの出力 (続き)。

```
... BUFFPOOL_SW_STATE LOCK_SW_STATE SORT_SW_STATE TIMESTAMP_SW_STATE
... ----------------- ------------- ------------- ------------------
...                 1             1             0                  1
```

戻される情報

表 220. SNAPSWITCHES 管理ビューおよび SNAP_GET_SWITCHES 表関数によって戻される情報

列名	データ・タイプ	説明
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
UOW_SW_STATE	SMALLINT	作業単位モニター記録スイッチの状態 (0 または 1)。
UOW_SW_TIME	TIMESTAMP	作業単位モニター記録スイッチがオンの場合、このスイッチがオンになった日時。
STATEMENT_SW_STATE	SMALLINT	SQL ステートメント・モニター記録スイッチの状態 (0 または 1)。
STATEMENT_SW_TIME	TIMESTAMP	SQL ステートメント・モニター記録スイッチがオンの場合、このスイッチがオンになった日時。
TABLE_SW_STATE	SMALLINT	表アクティビティー・モニター記録スイッチの状態 (0 または 1)。
TABLE_SW_TIME	TIMESTAMP	表アクティビティー・モニター記録スイッチがオンの場合、このスイッチがオンになった日時。
BUFFPOOL_SW_STATE	SMALLINT	バッファー・プール・アクティビティー・モニター記録スイッチの状態 (0 または 1)。
BUFFPOOL_SW_TIME	TIMESTAMP	バッファー・プール・アクティビティー・モニター記録スイッチがオンの場合、このスイッチがオンになった日時。
LOCK_SW_STATE	SMALLINT	ロック・モニター記録スイッチの状態 (0 または 1)。

表 220. SNAPSWITCHES 管理ビューおよび SNAP_GET_SWITCHES 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明
LOCK_SW_TIME	TIMESTAMP	ロック・モニター記録スイッチがオンの場合、このスイッチがオンになった日時。
SORT_SW_STATE	SMALLINT	ソート・モニター記録スイッチの状態 (0 または 1)。
SORT_SW_TIME	TIMESTAMP	ソート・モニター記録スイッチがオンの場合、このスイッチがオンになった日時。
TIMESTAMP_SW_STATE	SMALLINT	タイム・スタンプのモニター記録スイッチの状態 (0 または 1)。
TIMESTAMP_SW_TIME	TIMESTAMP	タイム・スタンプのモニター記録スイッチがオンの場合、このスイッチがオンになった日付と時刻。
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

SNAPTAB 管理ビューおよび SNAP_GET_TAB_V91 表関数 - table 論理データ・グループのスナップショット情報の検索

SNAPTAB 管理ビューおよび SNAP_GET_TAB_V91 表関数は、table 論理データ・グループからのスナップショット情報を戻します。

SNAPTAB 管理ビュー

この管理ビューを使用すると、現在接続されているデータベースに関する table 論理データ・グループのスナップショット情報を検索できます。

SNAPTAB_REORG 管理ビューと併せて使用することにより、SNAPTAB 管理ビューは GET SNAPSHOT FOR TABLES ON database-alias CLP コマンドと同等の情報を戻します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、753 ページの表 191を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPTAB 管理ビューに対する SELECT 特権
- SNAPTAB 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_TAB_V91 表関数に対する EXECUTE 特権

- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

すべてのアクティブな表のスキーマと名前を取り出します。

```
SELECT SUBSTR(TABSCHEMA,1,8), SUBSTR(TABNAME,1,15) AS TABNAME, TAB_TYPE,
   DBPARTITIONNUM FROM SYSIBMADM.SNAPTAB
```

以下はこの照会の出力例です。

```
TABSCHEMA TABNAME         TAB_TYPE     DBPARTITIONNUM
--------- --------------- ------------ --------------
SYSTOOLS  HMON_ATM_INFO   USER_TABLE                0

  1 record selected.
```

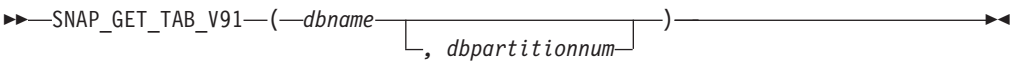
SNAP_GET_TAB_V91 表関数

SNAP_GET_TAB_V91 表関数は SNAPTAB 管理ビューと同じ情報を戻します。ただし、SNAP_GET_TAB_V91 表関数の場合は、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションについて、特定のデータベースの情報を検索することができます。

SNAP_GET_TAB_REORG 表関数と併せて使用することにより、SNAP_GET_TAB_V91 表関数は GET SNAPSHOT FOR TABLES ON database-alias CLP コマンドと同等の情報を戻します。

戻される可能性のある情報の完全なリストは、 753 ページの表 191を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できま

す。現在接続されているデータベースからのスナップショットを取得するには、NULL または空ストリングを指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプを持つファイルが存在しない場合には、現在接続されているデータベースのスナップショットとデータベース・パーティション番号を SNAP_GET_TAB_V91 表関数が取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_TAB_V91 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベースの集約ビューとして、アクティブな表のリストを検索します。

```
SELECT SUBSTR(TABSCHEMA,1,8) AS TABSCHEMA, SUBSTR(TABNAME,1,15) AS TABNAME,
   TAB_TYPE, DBPARTITIONNUM FROM TABLE(SNAP_GET_TAB('',-2)) AS T
```

以下はこの照会の出力例です。

```
TABSCHEMA TABNAME         TAB_TYPE      DBPARTITIONNUM
--------- --------------- ------------- --------------
SYSTOOLS  HMON_ATM_INFO   USER_TABLE                 -
JESSICAE  EMPLOYEE        USER_TABLE                 -
```

戻される情報

表 221. SNAPTAB 管理ビューおよび SNAP_GET_TAB_V91 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
TABSCHEMA	VARCHAR(128)	table_schema - 表スキーマ名
TABNAME	VARCHAR(128)	table_name - 表名
TAB_FILE_ID	BIGINT	table_file_id - 表ファイル ID
TAB_TYPE	VARCHAR(14)	table_type - 表タイプ。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • USER_TABLE • DROPPED_TABLE • TEMP_TABLE • CATALOG_TABLE • REORG_TABLE
DATA_OBJECT_PAGES	BIGINT	data_object_pages - データ・オブジェクト・ページ数
INDEX_OBJECT_PAGES	BIGINT	index_object_pages - 索引オブジェクト・ページ数
LOB_OBJECT_PAGES	BIGINT	lob_object_pages - LOB オブジェクト・ページ数
LONG_OBJECT_PAGES	BIGINT	long_object_pages - 長オブジェクト・ページ数
XDA_OBJECT_PAGES	BIGINT	xda_object_pages - XDA オブジェクト・ページ数
ROWS_READ	BIGINT	rows_read - 読み取り行数
ROWS_WRITTEN	BIGINT	rows_written - 書き込み行数
OVERFLOW_ACCESSES	BIGINT	overflow_accesses - オーバーフロー・レコードへのアクセス
PAGE_REORGS	BIGINT	page_reorgs - ページ再編成
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。
TBSP_ID	BIGINT	tablespace_id - 表スペース ID
DATA_PARTITION_ID	INTEGER	data_partition_id - データ・パーティション ID。非パーティション表では、このエレメントは NULL になります。

SNAPTAB_REORG 管理ビューおよび SNAP_GET_TAB_REORG 表関数 - 表再編成スナップショット情報の検索

SNAPTAB_REORG 管理ビューおよび SNAP_GET_TAB_REORG 表関数は、表再編成情報を戻します。再編成された表がない場合は、0 行が戻されます。データ・パーティション表が再編成されるとき、各データ・パーティションに対して 1 つのレコードが戻されます。データ・パーティション表の特定の 1 つのデータ・パーティションだけが再編成される場合、そのパーティションのレコードだけが戻されます。

SNAPTAB_REORG 管理ビュー

この管理ビューでは、現在接続されているデータベースの表再編成スナップショット情報を検索できます。

SNAPTAB 管理ビューと共に使用すると、SNAPTAB_REORG 管理ビューは、GET SNAPSHOT FOR TABLES ON database-alias CLP コマンドと同等のデータを提供します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 758 ページの表 192を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPTAB_REORG 管理ビューに対する SELECT 特権
- SNAPTAB_REORG 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_TAB_REORG 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベース上のすべてのデータベース・パーティションでの再編成操作の詳細を選択します。

```
SELECT SUBSTR(TABNAME, 1, 15) AS TAB_NAME, SUBSTR(TABSCHEMA, 1, 15)
   AS TAB_SCHEMA, REORG_PHASE, SUBSTR(REORG_TYPE, 1, 20) AS REORG_TYPE,
   REORG_STATUS, REORG_COMPLETION, DBPARTITIONNUM
   FROM SYSIBMADM.SNAPTAB_REORG ORDER BY DBPARTITIONNUM
```

以下はこの照会の出力例です。

```
TAB_NAME     TAB_SCHEMA     REORG_PHASE      ...
--------...- ----------...- ---------------- ...
EMPLOYEE     DBUSER         REPLACE          ...
EMPLOYEE     DBUSER         REPLACE          ...
EMPLOYEE     DBUSER         REPLACE          ...
                                             ...
3 record(s) selected.
```

この照会からの出力 (続き)。

```
... REORG_TYPE           REORG_STATUS REORG_COMPLETION DBPARTITIONNUM
... -------------------- ------------ ---------------- --------------
... RECLAIM+OFFLINE+ALLO COMPLETED    SUCCESS                       0
... RECLAIM+OFFLINE+ALLO COMPLETED    SUCCESS                       1
... RECLAIM+OFFLINE+ALLO COMPLETED    SUCCESS                       2
```

マルチディメンション・クラスタリング (MDC) 表からエクステントを再利用するための再編成操作に関するすべての情報を、SNAPTAB_REORG 管理ビューから選択します。

```
db2 -v "select * from sysibmadm.snaptab_reorg"

TABNAME  REORG_PHASE       REORG_MAX_PHASE   REORG_TYPE
-------- ----------------- ----------------- --------------------------
T1       RELEASE           3                 RECLAIM_EXTENTS+ALLOW_WRITE

REORG_STATUS REORG_COMPLETION REORG_START                REORG_END
------------ ---------------- -------------------------- --------------------------
COMPLETED    SUCCESS          2008-09-24-14.35.30.734741 2008-09-24-14.35.31.460674
```

SNAP_GET_TAB_REORG 表関数

SNAP_GET_TAB_REORG 表関数は SNAPTAB_REORG 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションのいずれかの特定のデータベースを対象とした情報を検索することができます。

SNAP_GET_TAB 表関数と共に使用すると、SNAP_GET_TAB_REORG 表関数は、GET SNAPSHOT FOR TABLES ON database-alias CLP コマンドと同等のデータを提供します。

戻される可能性のある情報の完全なリストは、 758 ページの表 192を参照してください。

構文

スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL または空ストリングを指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプが含まれるファイルが存在しない場合、SNAP_GET_TAB_REORG 表関数は、現在接続されているデータベースのスナップショットとデータベース・パーティション番号を取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_TAB_REORG 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベース上のデータベース・パーティション 1 での再編成操作の詳細を選択します。

```
SELECT SUBSTR(TABNAME, 1, 15) AS TAB_NAME, SUBSTR(TABSCHEMA, 1, 15)
   AS TAB_SCHEMA, REORG_PHASE, SUBSTR(REORG_TYPE, 1, 20) AS REORG_TYPE,
   REORG_STATUS, REORG_COMPLETION, DBPARTITIONNUM
   FROM TABLE( SNAP_GET_TAB_REORG('', 1)) AS T
```

以下はこの照会の出力例です。

```
TAB_NAME     TAB_SCHEMA     REORG_PHASE     REORG_TYPE           ...
--------...- ----------...- -----------...- -------------------- ...
EMPLOYEE     DBUSER         REPLACE         RECLAIM+OFFLINE+ALLO ...
                                                                 ...
  1 record(s) selected.                                          ...
```

この照会からの出力 (続き)。

```
... REORG_STATUS REORG_COMPLETION DBPARTITIONNUM
... ------------ ---------------- --------------
... COMPLETED    SUCCESS                     1
...
```

SNAP_GET_TAB_REORG 表関数を使用して、マルチディメンション・クラスタリング (MDC) 表からエクステントを再利用するための再編成操作に関するすべての情報を選択します。

```
db2 -v "select * from table(snap_get_tab_reorg(''))"

TABNAME  REORG_PHASE       REORG_MAX_PHASE   REORG_TYPE
-------- ----------------- ----------------- --------------------------
T1       RELEASE           3                 RECLAIM_EXTENTS+ALLOW_WRITE

REORG_STATUS REORG_COMPLETION REORG_START                REORG_END
------------ --------------- -------------------------- --------------------------
COMPLETED    SUCCESS         2008-09-24-14.35.30.734741 2008-09-24-14.35.31.460674
```

戻される情報

表 222. SNAPTAB_REORG 管理ビューおよび SNAP_GET_TAB_REORG 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
TABNAME	VARCHAR (128)	table_name - 表名
TABSCHEMA	VARCHAR (128)	table_schema - 表スキーマ名
PAGE_REORGS	BIGINT	page_reorgs - ページ再編成
REORG_PHASE	VARCHAR (16)	reorg_phase - 表再編成フェーズ。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • BUILD • DICT_SAMPLE • INDEX_RECREATE • REPLACE • SORT • SCAN • DRAIN • RELEASE または SORT+DICT_SAMPLE。
REORG_MAX_PHASE	INTEGER	reorg_max_phase - 表再編成の最大フェーズ数
REORG_CURRENT_COUNTER	BIGINT	reorg_current_counter - 表再編成の進行状況
REORG_MAX_COUNTER	BIGINT	reorg_max_counter - 表再編成の合計量

表 222. SNAPTAB_REORG 管理ビューおよび SNAP_GET_TAB_REORG 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント

REORG_TYPE	VARCHAR (128)	reorg_type - 表再編成の属性。このインターフェースは、以下の ID の組み合わせを '+' 記号で区切ったものを使用してテキスト ID を戻します。
		
		以下のいずれかが使用されます。
		• RECLAIM
		• RECLUSTER
		• RECLAIM_EXTS
		さらに以下のいずれかが使用されます。
		• +OFFLINE
		• +ONLINE
		アクセス・モードが指定されている場合、以下のいずれかが使用されます。
		• +ALLOW_NONE
		• +ALLOW_READ
		• +ALLOW_WRITE
		オフラインで RECLUSTER オプションが指定されている場合、以下のいずれかが使用されます。
		• +INDEXSCAN
		• +TABLESCAN
		オフラインの場合、以下のいずれかが使用されます。
		• +LONGLOB
		• +DATAONLY
		オフラインで、オプションが指定されている場合、以下の任意のものが使用されます。
		• +CHOOSE_TEMP
		• +KEEPDICTIONARY
		• +RESETDICTIONARY
		オンラインで、オプションが指定されている場合、以下が使用されます。
		• +NOTRUNCATE
		
		例 1: REORG TABLE TEST.EMPLOYEE が実行された場合、以下のように表示されます。
		`RECLAIM+OFFLINE+ALLOW_READ+DATAONLY +KEEPDICTIONARY`
		
		例 2: REORG TABLE TEST.EMPLOYEE INDEX EMPIDX INDEXSCAN が実行された場合、以下のように表示されます。
		`RECLUSTER+OFFLINE+ALLOW_READ+INDEXSCAN +DATAONLY+KEEPDICTIONARY`

表 222. SNAPTAB_REORG 管理ビューおよび SNAP_GET_TAB_REORG 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
REORG_STATUS	VARCHAR (10)	reorg_status - 表再編成の状況。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • COMPLETED • PAUSED • STARTED • STOPPED • TRUNCATE
REORG_COMPLETION	VARCHAR (10)	reorg_completion - 表再編成完了フラグ。このインターフェースは、sqlmon.h での定義を基にしてテキスト ID を戻します。以下のいずれかとなります。 • FAIL • SUCCESS
REORG_START	TIMESTAMP	reorg_start - 表再編成開始時刻
REORG_END	TIMESTAMP	reorg_end - 表再編成終了時刻
REORG_PHASE_START	TIMESTAMP	reorg_phase_start - 表再編成フェーズ開始時刻
REORG_INDEX_ID	BIGINT	reorg_index_id - 表の再編成に使用される索引
REORG_TBSPC_ID	BIGINT	reorg_tbspc_id - 表が再編成される表スペース
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。
DATA_PARTITION_ID	INTEGER	data_partition_id - データ・パーティション ID。非パーティション表では、このエレメントは NULL になります。
REORG_ ROWSCOMPRESSED	BIGINT	reorg_rows_compressed - 圧縮行数
REORG_ROWSREJECTED	BIGINT	reorg_rows_rejected_for_compression - 圧縮がリジェクトされる行
REORG_LONG_TBSPC_ID	BIGINT	reorg_long_tbspc_id - 長いオブジェクトが再編成される表スペース

SNAPTBSP 管理ビューおよび SNAP_GET_TBSP_V91 表関数 - 表スペース論理データ・グループのスナップショット情報の検索

SNAPTBSP 管理ビューおよび SNAP_GET_TBSP_V91 表関数は、tablespace 論理データ・グループからのスナップショット情報を戻します。

SNAPTBSP 管理ビュー

この管理ビューを使用すると、現在接続されているデータベースに関する table space 論理データ・グループのスナップショット情報を検索できます。

SNAPTBSP_PART、 SNAPTBSP_QUIESCER、 SNAPTBSP_RANGE、 SNAPCONTAINER 管理ビューと併せて使用することにより、SNAPTBSP 管理ビューは GET SNAPSHOT FOR TABLESPACES ON database-alias CLP コマンドと同等の情報を戻します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 763 ページの表 193を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPTBSP 管理ビューに対する SELECT 特権
- SNAPTBSP 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_TBSP_V91 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベースの、カタログ・データベース・パーティションの表スペースのリストを検索します。

```
SELECT SUBSTR(TBSP_NAME,1,30) AS TBSP_NAME, TBSP_ID, TBSP_TYPE,
   TBSP_CONTENT_TYPE FROM SYSIBMADM.SNAPTBSP WHERE DBPARTITIONNUM = 1
```

以下はこの照会の出力例です。

```
TBSP_NAME  TBSP_ID     TBSP_TYPE  TBSP_CONTENT_TYPE
---------- -------...- ---------- -----------------
TEMPSPACE1           1 SMS        SYSTEMP
USERSPACE1           2 DMS        LONG

2 record(s) selected.
```

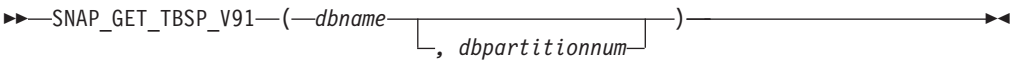
SNAP_GET_TBSP_V91 表関数

SNAP_GET_TBSP_V91 表関数は SNAPTBSP 管理ビューと同じ情報を戻します。ただし、SNAP_GET_TBSP_V91 表関数の場合は、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションについて、特定のデータベースに関する情報を検索することができます。

SNAP_GET_TBSP_PART_V91、 SNAP_GET_TBSP_QUIESCER、 SNAP_GET_TBSP_RANGE、 SNAP_GET_CONTAINER_V91 表関数と併せて使用することにより、SNAP_GET_TBSP_V91 表関数は GET SNAPSHOT FOR TABLESPACES ON database-alias CLP コマンドと同等の情報を戻します。

戻される可能性のある情報の完全なリストは、 763 ページの表 193を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL または空ストリングを指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。 *dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプを持つファイルが存在しない場合には、現在接続されているデータベースのスナップショットとデータベース・パーティション番号を SNAP_GET_TBSP_V91 表関数が取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_TBSP_V91 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベースの、すべてのデータベース・パーティションの表スペースのリストを検索します。

```
SELECT SUBSTR(TBSP_NAME,1,10) AS TBSP_NAME, TBSP_ID, TBSP_TYPE,
   TBSP_CONTENT_TYPE, DBPARTITIONNUM FROM TABLE(SNAP_GET_TBSP_V91('')) AS T
```

以下はこの照会の出力例です。

```
TBSP_NAME  TBSP_ID     TBSP_TYPE  TBSP_CONTENT_TYPE DBPARTITIONNUM
---------- -------...- ---------- ----------------- --------------
TEMPSPACE1           1 SMS        SYSTEMP                         1
USERSPACE1           2 DMS        LONG                            1
SYSCATSPAC           0 DMS        ANY                             0
TEMPSPACE1           1 SMS        SYSTEMP                         0
USERSPACE1           2 DMS        LONG                            0
SYSTOOLSPA           3 DMS        LONG                            0
TEMPSPACE1           1 SMS        SYSTEMP                         2
USERSPACE1           2 DMS        LONG                            2

  8 record(s) selected.
```

戻される情報

表 223. SNAPTBSP 管理ビューおよび SNAP_GET_TBSP_V91 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
TBSP_NAME	VARCHAR(128)	tablespace_name - 表スペース名
TBSP_ID	BIGINT	tablespace_id - 表スペース ID
TBSP_TYPE	VARCHAR(10)	tablespace_type - 表スペース・タイプ。このインターフェースは、sqlutil.h の定義に基づくテキスト ID を戻します。これは次のいずれかです。 • DMS • SMS

表 223. SNAPTBSP 管理ビューおよび SNAP_GET_TBSP_V91 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
TBSP_CONTENT_TYPE	VARCHAR(10)	tablespace_content_type - 表スペースのコンテンツ・タイプ。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • ANY • LARGE • SYSTEMP • USRTEMP
TBSP_PAGE_SIZE	BIGINT	tablespace_page_size - 表スペースのページ・サイズ
TBSP_EXTENT_SIZE	BIGINT	tablespace_extent_size - 表スペースのエクステント・サイズ
TBSP_PREFETCH_SIZE	BIGINT	tablespace_prefetch_size - 表スペースのプリフェッチ・サイズ
TBSP_CUR_POOL_ID	BIGINT	tablespace_cur_pool_id - 現在使用中のバッファー・プール
TBSP_NEXT_POOL_ID	BIGINT	tablespace_next_pool_id - 次の始動時に使用されるバッファー・プール
FS_CACHING	SMALLINT	fs_caching - ファイル・システム・キャッシング
POOL_DATA_L_READS	BIGINT	pool_data_l_reads - バッファー・プール・データの論理読み取り
POOL_DATA_P_READS	BIGINT	pool_data_p_reads - バッファー・プール・データの物理読み取り
POOL_TEMP_DATA_L_READS	BIGINT	pool_temp_data_l_reads - バッファー・プール一時データの論理読み取り
POOL_TEMP_DATA_P_READS	BIGINT	pool_temp_data_p_reads - バッファー・プール一時データの物理読み取り
POOL_ASYNC_DATA_READS	BIGINT	pool_async_data_reads - バッファー・プール非同期データ読み取り
POOL_DATA_WRITES	BIGINT	pool_data_writes - バッファー・プールへのデータの書き込み
POOL_ASYNC_DATA_WRITES	BIGINT	pool_async_data_writes - バッファー・プール非同期データ書き込み
POOL_INDEX_L_READS	BIGINT	pool_index_l_reads - バッファー・プール索引の論理読み取り
POOL_INDEX_P_READS	BIGINT	pool_index_p_reads - バッファー・プール索引の物理読み取り

表 223. SNAPTBSP 管理ビューおよび SNAP_GET_TBSP_V91 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
POOL_TEMP_INDEX_L_READS	BIGINT	pool_temp_index_l_reads - バッファー・プール一時索引の論理読み取り
POOL_TEMP_INDEX_P_READS	BIGINT	pool_temp_index_p_reads - バッファー・プール一時索引の物理読み取り
POOL_ASYNC_INDEX_READS	BIGINT	pool_async_index_reads - バッファー・プール非同期索引読み取り
POOL_INDEX_WRITES	BIGINT	pool_index_writes - バッファー・プール索引の書き込み
POOL_ASYNC_INDEX_WRITES	BIGINT	pool_async_index_writes - バッファー・プール非同期索引書き込み
POOL_XDA_L_READS	BIGINT	pool_xda_l_reads - バッファー・プール XDA データの論理読み取り
POOL_XDA_P_READS	BIGINT	pool_xda_p_reads - バッファー・プール XDA データの物理読み取り
POOL_XDA_WRITES	BIGINT	pool_xda_writes - バッファー・プール XDA データの書き込み
POOL_ASYNC_XDA_READS	BIGINT	pool_async_xda_reads - バッファー・プール非同期 XDA データ読み取り
POOL_ASYNC_XDA_WRITES	BIGINT	pool_async_xda_writes - バッファー・プール非同期 XDA データ書き込み
POOL_TEMP_XDA_L_READS	BIGINT	pool_temp_xda_l_reads - バッファー・プール一時 XDA データの論理読み取り
POOL_TEMP_XDA_P_READS	BIGINT	pool_temp_xda_p_reads - バッファー・プール一時 XDA データの物理読み取り : モニター・エレメント
POOL_READ_TIME	BIGINT	pool_read_time - バッファー・プール物理読み取り時間の合計
POOL_WRITE_TIME	BIGINT	pool_write_time - バッファー・プール物理書き込み時間の合計
POOL_ASYNC_READ_TIME	BIGINT	pool_async_read_time - バッファー・プール非同期読み取り時間
POOL_ASYNC_WRITE_TIME	BIGINT	pool_async_write_time - バッファー・プール非同期書き込み時間
POOL_ASYNC_DATA_ READ_REQS	BIGINT	pool_async_data_read_reqs - バッファー・プール非同期読み取り要求
POOL_ASYNC_INDEX_ READ_REQS	BIGINT	pool_async_index_read_reqs - バッファー・プール非同期索引読み取り要求

表 223. SNAPTBSP 管理ビューおよび SNAP_GET_TBSP_V91 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
POOL_ASYNC_XDA_ READ_REQS	BIGINT	pool_async_xda_read_reqs - バッファー・プール非同期 XDA 読み取り要求
POOL_NO_VICTIM_BUFFER	BIGINT	pool_no_victim_buffer - バッファー・プールの非ビクティム・バッファー数
DIRECT_READS	BIGINT	direct_reads - データベースからの直接読み取り
DIRECT_WRITES	BIGINT	direct_writes - データベースへの直接書き込み
DIRECT_READ_REQS	BIGINT	direct_read_reqs - 直接読み取り要求
DIRECT_WRITE_REQS	BIGINT	direct_write_reqs - 直接書き込み要求
DIRECT_READ_TIME	BIGINT	direct_read_time - 直接読み取り時間
DIRECT_WRITE_TIME	BIGINT	direct_write_time - 直接書き込み時間
FILES_CLOSED	BIGINT	files_closed - 閉じられたデータベース・ファイル
UNREAD_PREFETCH_PAGES	BIGINT	unread_prefetch_pages - 読み取り不能プリフェッチ・ページ
TBSP_REBALANCER_MODE	VARCHAR(10)	tablespace_rebalancer_mode - リバランサー・モード。このインターフェースは、`sqlmon.h` での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • NO_REBAL • FWD_REBAL • REV_REBAL
TBSP_USING_AUTO_STORAGE	SMALLINT	tablespace_using_auto_storage - 自動ストレージが使用可能な表スペース
TBSP_AUTO_RESIZE_ENABLED	SMALLINT	tablespace_auto_resize_enabled - 表スペースの自動サイズ変更可能
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

SNAPTBSP_PART 管理ビューおよび SNAP_GET_TBSP_PART_V91 表関数 - tablespace_nodeinfo 論理データ・グループのスナップショット情報の検索

注: この表関数は使用すべきではなく、 767 ページの『SNAPTBSP_PART 管理ビューおよび SNAP_GET_TBSP_PART_V97 表関数 - tablespace_nodeinfo 論理データ・グループのスナップショット情報の検索』 に置き換えられました。

SNAPTBSP_PART 管理ビューおよび SNAP_GET_TBSP_PART_V91 表関数は、tablespace_nodeinfo 論理データ・グループからのスナップショット情報を戻します。

SNAPTBSP_PART 管理ビュー

この管理ビューを使用すると、現在接続されているデータベースに関する tablespace_nodeinfo 論理データ・グループのスナップショット情報を検索することができます。

SNAPTBSP、SNAPTBSP_QUIESCER、SNAPTBSP_RANGE、SNAPCONTAINER 管理ビューと併せて使用することにより、SNAPTBSP_PART 管理ビューは GET SNAPSHOT FOR TABLESPACES ON database-alias CLP コマンドと同等の情報を戻します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 921 ページの表 224を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPTBSP_PART 管理ビューに対する SELECT 特権
- SNAPTBSP_PART 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_TBSP_PART_V91 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベースのすべてのデータベース・パーティションの表スペースとその状態のリストを検索します。

```
SELECT SUBSTR(TBSP_NAME,1,30) AS TBSP_NAME, TBSP_ID,
   SUBSTR(TBSP_STATE,1,30) AS TBSP_STATE, DBPARTITIONNUM
   FROM SYSIBMADM.SNAPTBSP_PART
```

以下はこの照会の出力例です。

```
TBSP_NAME       TBSP_ID     TBSP_STATE     DBPARTITIONNUM
-----------...- -------...- ----------...- --------------
SYSCATSPACE               0 NORMAL                       0
TEMPSPACE1                1 NORMAL                       0
USERSPACE1                2 NORMAL                       0
TEMPSPACE1                1 NORMAL                       1
USERSPACE1                2 NORMAL                       1

   5 record(s) selected.
```

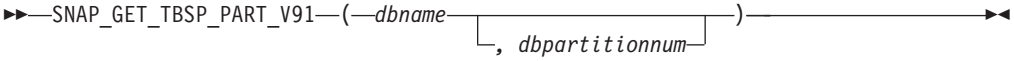
SNAP_GET_TBSP_PART_V91 表関数

SNAP_GET_TBSP_PART_V91 表関数は SNAPTBSP_PART 管理ビューと同じ情報を戻します。ただし、SNAP_GET_TBSP_PART_V91 表関数の場合は、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションについて、特定のデータベースに関する情報を検索することができます。

SNAP_GET_TBSP_V91、 SNAP_GET_TBSP_QUIESCER、 SNAP_GET_TBSP_RANGE、 SNAP_GET_CONTAINER_V91 表関数と併せて使用することにより、SNAP_GET_TBSP_PART_V91 表関数は GET SNAPSHOT FOR TABLESPACES ON database-alias CLP コマンドと同等の情報を戻します。

戻される可能性のある情報の完全なリストは、 921 ページの表 224を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL または空ストリングを指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。 *dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプション

が使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプを持つファイルが存在しない場合には、現在接続されているデータベースのスナップショットとデータベース・パーティション番号を SNAP_GET_TBSP_PART_V91 表関数が取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_TBSP_PART_V91 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

接続されているデータベースの接続されているデータベース・パーティションの表スペースとその状態のリストを検索します。

```
SELECT SUBSTR(TBSP_NAME,1,30) AS TBSP_NAME, TBSP_ID,
   SUBSTR(TBSP_STATE,1,30) AS TBSP_STATE
   FROM TABLE(SNAP_GET_TBSP_PART_V91(CAST(NULL AS VARCHAR(128)),-1)) AS T
```

以下はこの照会の出力例です。

```
TBSP_NAME                      TBSP_ID              TBSP_STATE
------------------------------ -------------------- ------------...-
SYSCATSPACE                                       0 NORMAL
TEMPSPACE1                                        1 NORMAL
USERSPACE1                                        2 NORMAL
SYSTOOLSPACE                                      3 NORMAL
SYSTOOLSTMPSPACE                                  4 NORMAL

  5 record(s) selected.
```

戻される情報

表 224. SNAPTBSP_PART 管理ビューおよび SNAP_GET_TBSP_PART_V91 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。

表 224. SNAPTBSP_PART 管理ビューおよび SNAP_GET_TBSP_PART_V91 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
TBSP_NAME	VARCHAR (128)	tablespace_name - 表スペース名
TBSP_ID	BIGINT	tablespace_id - 表スペース ID
TBSP_STATE	VARCHAR (256)	tablespace_state - 表スペースの状態。このインターフェースは、sqlutil.h での定義に基づくテキスト ID を戻します。これは次のものを「+」符号で分離して組み合わせたものになります。 • BACKUP_IN_PROGRESS • BACKUP_PENDING • DELETE_PENDING • DISABLE_PENDING • DROP_PENDING • LOAD_IN_PROGRESS • LOAD_PENDING • NORMAL • OFFLINE • PSTAT_CREATION • PSTAT_DELETION • QUIESCED_EXCLUSIVE • QUIESCED_SHARE • QUIESCED_UPDATE • REBAL_IN_PROGRESS • REORG_IN_PROGRESS • RESTORE_IN_PROGRESS • RESTORE_PENDING • ROLLFORWARD_IN_PROGRESS • ROLLFORWARD_PENDING • STORDEF_ALLOWED • STORDEF_CHANGED • STORDEF_FINAL_VERSION • STORDEF_PENDING • SUSPEND_WRITE
TBSP_PREFETCH_SIZE	BIGINT	tablespace_prefetch_size - 表スペースのプリフェッチ・サイズ
TBSP_NUM_QUIESCERS	BIGINT	tablespace_num_quiescers - 静止プログラム数
TBSP_STATE_CHANGE_OBJECT_ID	BIGINT	tablespace_state_change_object_id - 状態変更オブジェクト ID

表 224. SNAPTBSP_PART 管理ビューおよび SNAP_GET_TBSP_PART_V91 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
TBSP_STATE_CHANGE_TBSP_ID	BIGINT	tablespace_state_change_ts_id - 状態変更表スペース ID
TBSP_MIN_RECOVERY_TIME	TIMESTAMP	tablespace_min_recovery_time - ロールフォワードの最小リカバリー時間
TBSP_TOTAL_PAGES	BIGINT	tablespace_total_pages - 表スペース内の合計ページ数
TBSP_USABLE_PAGES	BIGINT	tablespace_usable_pages - 表スペース内の使用可能ページ数
TBSP_USED_PAGES	BIGINT	tablespace_used_pages - 表スペース内の使用されているページ数
TBSP_FREE_PAGES	BIGINT	tablespace_free_pages - 表スペース内のフリー・ページ数
TBSP_PENDING_FREE_PAGES	BIGINT	tablespace_pending_free_pages - 表スペース内のペンディング・フリー・ページ数
TBSP_PAGE_TOP	BIGINT	tablespace_page_top - 表スペース最高水準点
REBALANCER_MODE	VARCHAR (10)	tablespace_rebalancer_mode - リバランサー・モード。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • FWD_REBAL • NO_REBAL • REV_REBAL
REBALANCER_EXTENTS_REMAINING	BIGINT	tablespace_rebalancer_extents_remaining - リバランサーで処理されるエクステントの合計数
REBALANCER_EXTENTS_PROCESSED	BIGINT	tablespace_rebalancer_extents_processed - リバランサーで処理されたエクステントの数
REBALANCER_PRIORITY	BIGINT	tablespace_rebalancer_priority - 現行のリバランサー優先順位
REBALANCER_START_TIME	TIMESTAMP	tablespace_rebalancer_start_time - リバランサー開始時刻
REBALANCER_RESTART_TIME	TIMESTAMP	tablespace_rebalancer_restart_time - リバランサー再始動時刻
REBALANCER_LAST_EXTENT_MOVED	BIGINT	tablespace_rebalancer_last_extent_moved - リバランサーによって最後に移動されたエクステント
TBSP_NUM_RANGES	BIGINT	tablespace_num_ranges - 表スペース・マップ内の範囲数

表 224. SNAPTBSP_PART 管理ビューおよび SNAP_GET_TBSP_PART_V91 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
TBSP_NUM_CONTAINERS	BIGINT	tablespace_num_containers - 表スペース内のコンテナー数
TBSP_INITIAL_SIZE	BIGINT	tablespace_initial_size - 表スペースの初期サイズ
TBSP_CURRENT_SIZE	BIGINT	tablespace_current_size - 表スペースの現行サイズ
TBSP_MAX_SIZE	BIGINT	tablespace_max_size - 表スペースの最大サイズ
TBSP_INCREASE_SIZE	BIGINT	tablespace_increase_size - バイト単位のサイズの増加
TBSP_INCREASE_SIZE_PERCENT	SMALLINT	tablespace_increase_size_percent - パーセント単位のサイズの増加
TBSP_LAST_RESIZE_TIME	TIMESTAMP	tablespace_last_resize_time - 最後にサイズ変更が正常に行われた時刻
TBSP_LAST_RESIZE_FAILED	SMALLINT	tablespace_last_resize_failed - 失敗した最後のサイズ変更
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

SNAPTBSP_QUIESCER 管理ビューおよび SNAP_GET_TBSP_QUIESCER 表関数 - quiescer 表スペース・スナップショット情報の検索

SNAPTBSP_QUIESCER 管理ビューおよび SNAP_GET_TBSP_QUIESCER 表関数は、表スペース・スナップショットから、静止プログラムに関する情報を戻します。

SNAPTBSP_QUIESCER 管理ビュー

この管理ビューでは、現在接続されているデータベースの静止プログラム表スペース・スナップショット情報を検索できます。

SNAPTBSP、SNAPTBSP_PART、SNAPTBSP_RANGE、SNAPCONTAINER 管理ビューと共に使用すると、SNAPTBSP_QUIESCER 管理ビューは、GET SNAPSHOT FOR TABLESPACES ON database-alias CLP コマンドと同等の情報を提供します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、776 ページの表 195を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPTBSP_QUIESCER 管理ビューに対する SELECT 特権
- SNAPTBSP_QUIESCER 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_TBSP_QUIESCER 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベースのすべてのデータベース・パーティションの静止した表スペースの情報を検索します。

```
SELECT SUBSTR(TBSP_NAME, 1, 10) AS TBSP_NAME, QUIESCER_TS_ID,
   QUIESCER_OBJ_ID, QUIESCER_AUTH_ID, QUIESCER_AGENT_ID,
   QUIESCER_STATE, DBPARTITIONNUM
   FROM SYSIBMADM.SNAPTBSP_QUIESCER ORDER BY DBPARTITIONNUM
```

以下はこの照会の出力例です。

```
TBSP_NAME  QUIESCER_TS_ID     QUIESCER_OBJ_ID     QUIESCER_AUTH_ID     ..
---------- --------------...- ---------------...- ----------------...- ..
USERSPACE1                  2                   5 SWALKTY              ..
USERSPACE1                  2                   5 SWALKTY              ..

  2 record(s) selected.
```

この照会からの出力 (続き)。

```
... QUIESCER_AGENT_ID    QUIESCER_STATE DBPARTITIONNUM
... -------------------- -------------- --------------
...                    0 EXCLUSIVE                   0
...                65983 EXCLUSIVE                   1
```

例: 範囲パーティション表の名前の判別

表が範囲パーティション化されていて、静止状態に保たれている場合、表スペース ID および表 ID の値は、SYSCAT.TABLES 内のものとは異なる表記となります。これらの ID は、符号なしの短い表記となります。静止した表の名前を検索するには、QEUIESCER_TS_ID から 65536 (最大値) を減算した表スペース ID を計算することによって、まず符号付きの短い表記を検索する必要があります。それから、この表スペース ID を使用して、静止した表を特定します。(実際の表スペース ID は、表内の各範囲パーティションの SYSCAT.DATAPARTITIONS にあります。)

```
SELECT SUBSTR(TBSP_NAME, 1, 10) AS TBSP_NAME,
  CASE WHEN QUIESCER_TS_ID = 65530
    THEN QUIESCER_TS_ID - 65536
    ELSE QUIESCER_TS_ID END as tbspaceid,
  CASE WHEN QUIESCER_TS_ID = 65530
    THEN QUIESCER_OBJ_ID - 65536
    ELSE QUIESCER_OBJ_ID END as tableid
  FROM SYSIBMADM.SNAPTBSP_QUIESCER
  ORDER BY DBPARTITIONNUM
```

以下はこの照会の出力例です。

```
TBSP_NAME     TBSPACEID    TABLEID
------------- ------------ --------
TABDATA       -6           -32768
DATAMART      -6           -32765
SMALL         5            17

  3 record(s) selected.
```

上記の照会で得られた特定の TBSPACEID および TABLEID を使用して、SYSCAT.TABLES から表スキーマおよび表名を検索します。

```
SELECT CHAR(tabschema, 10)tabschema, CHAR(tabname,15)tabname
  FROM SYSCAT.TABLES
  WHERE tbspaceid = -6 AND tableid in (-32768,-32765)
```

以下はこの照会の出力例です。

```
TABSCHEMA    TABNAME
------------ --------
TPCD         ORDERS_RP
TPCD         ORDERS_DMART

  2 record(s) selected.
```

```
SELECT CHAR(tabschema, 10)tabschema, CHAR(tabname,15)tabname
  FROM SYSCAT.TABLES
  WHERE tbspaceid = 5 AND tableid = 17
```

以下はこの照会の出力例です。

```
TABSCHEMA    TABNAME
------------ --------
TPCD         NATION

  1 record(s) selected.
```

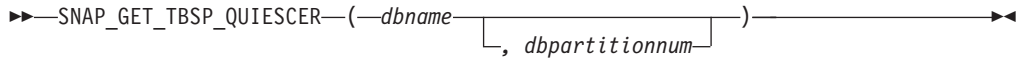
SNAP_GET_TBSP_QUIESCER 表関数

SNAP_GET_TBSP_QUIESCER 表関数は SNAPTBSP_QUIESCER 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションのいずれかの特定のデータベースを対象とした情報を検索することができます。

SNAP_GET_TBSP_V91、 SNAP_GET_TBSP_PART_V91、SNAP_GET_TBSP_RANGE、 SNAP_GET_CONTAINER_V91 表関数と共に使用すると、SNAP_GET_TBSP_QUIESCER 表関数は、GET SNAPSHOT FOR TABLESPACES ON database-alias CLP コマンドと同等の情報を提供します。

戻される可能性のある情報の完全なリストは、 776 ページの表 195を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL または空ストリングを指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプが含まれるファイルが存在しない場合、SNAP_GET_TBSP_QUIESCER 表関数は、現在接続されているデータベースのスナップショットとデータベース・パーティション番号を取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_TBSP_QUIESCER 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベースのデータベース・パーティション 1 の静止した表スペースの情報を検索します。

```
SELECT SUBSTR(TBSP_NAME, 1, 10) AS TBSP_NAME, QUIESCER_TS_ID,
   QUIESCER_OBJ_ID, QUIESCER_AUTH_ID, QUIESCER_AGENT_ID,
   QUIESCER_STATE, DBPARTITIONNUM
   FROM TABLE( SYSPROC.SNAP_GET_TBSP_QUIESCER( '', 1)) AS T
```

以下はこの照会の出力例です。

```
TBSP_NAME  QUIESCER_TS_ID     QUIESCER_OBJ_ID     QUIESCER_AUTH_ID     ...
---------- --------------...- ---------------...- ----------------...- ...
USERSPACE1                 2                   5 SWALKTY              ...

  1 record(s) selected.
```

この照会からの出力 (続き)。

```
... QUIESCER_AGENT_ID    QUIESCER_STATE DBPARTITIONNUM
... -------------------- -------------- --------------
...                65983 EXCLUSIVE                   1
```

戻される情報

表 225. SNAPTBSP_QUIESCER 管理ビューおよび SNAP_GET_TBSP_QUIESCER 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
TBSP_NAME	VARCHAR(128)	tablespace_name - 表スペース名
QUIESCER_TS_ID	BIGINT	quiescer_ts_id - 静止プログラム表スペース ID
QUIESCER_OBJ_ID	BIGINT	quiescer_obj_id - 静止プログラム・オブジェクト ID
QUIESCER_AUTH_ID	VARCHAR(128)	quiescer_auth_id - 静止プログラム・ユーザー許可 ID
QUIESCER_AGENT_ID	BIGINT	quiescer_agent_id - 静止プログラム・エージェント ID
QUIESCER_STATE	VARCHAR(14)	quiescer_state - 静止プログラムの状態。このインターフェースは、sqlutil.h での定義を基にしてテキスト ID を戻します。以下のいずれかとなります。 • EXCLUSIVE • UPDATE • SHARE
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

SNAPTBSP_RANGE 管理ビューおよび SNAP_GET_TBSP_RANGE 表関数 - 範囲スナップショット情報の検索

SNAPTBSP_RANGE 管理ビューおよび SNAP_GET_TBSP_RANGE 表関数は、範囲スナップショットから情報を戻します。

SNAPTBSP_RANGE 管理ビュー

この管理ビューでは、現在接続されているデータベースの範囲スナップショット情報を検索できます。

SNAPTBSP、SNAPTBSP_PART、SNAPTBSP_QUIESCER、および SNAPCONTAINER 管理ビューと共に使用すると、SNAPTBSP_RANGE 管理ビューは、GET SNAPSHOT FOR TABLESPACES ON database-alias CLP コマンドと同等の情報を提供します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 780 ページの表 196を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPTBSP_RANGE 管理ビューに対する SELECT 特権
- SNAPTBSP_RANGE 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_TBSP_RANGE 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベースのすべてのデータベース・パーティションの表スペース範囲についての情報を選択します。

```
SELECT TBSP_ID, SUBSTR(TBSP_NAME, 1, 15) AS TBSP_NAME, RANGE_NUMBER,
   RANGE_STRIPE_SET_NUMBER, RANGE_OFFSET, RANGE_MAX_PAGE,
   RANGE_MAX_EXTENT, RANGE_START_STRIPE, RANGE_END_STRIPE,
   RANGE_ADJUSTMENT, RANGE_NUM_CONTAINER, RANGE_CONTAINER_ID,
   DBPARTITIONNUM FROM SYSIBMADM.SNAPTBSP_RANGE
   ORDER BY DBPARTITIONNUM
```

以下はこの照会の出力例です。

```
TBSP_ID     TBSP_NAME       RANGE_NUMBER     RANGE_STRIPE_SET_NUMBER ...
-------...- --------------- ------------...- ----------------------- ...
          0 SYSCATSPACE                    0                       0 ...
          2 USERSPACE1                     0                       0 ...
          3 SYSTOOLSPACE                   0                       0 ...
          2 USERSPACE1                     0                       0 ...
          2 USERSPACE1                     0                       0 ...

  5 record(s) selected.
```

この照会からの出力 (続き)。

```
... RANGE_OFFSET     RANGE_MAX_PAGE       RANGE_MAX_EXTENT     ...
... ------------...- -------------------- -------------------- ...
...                0                11515                 2878 ...
...                0                  479                   14 ...
...                0                  251                   62 ...
...                0                  479                   14 ...
...                0                  479                   14 ...
```

この照会からの出力 (続き)。

```
... RANGE_START_STRIPE   RANGE_END_STRIPE     RANGE_ADJUSTMENT     ...
... -------------------- -------------------- -------------------- ...
...                    0                 2878                    0 ...
...                    0                   14                    0 ...
...                    0                   62                    0 ...
...                    0                   14                    0 ...
...                    0                   14                    0 ...
```

この照会からの出力 (続き)。

```
... RANGE_NUM_CONTAINER  RANGE_CONTAINER_ID   DBPARTITIONNUM
... -------------------- -------------------- --------------
...                    1                    0              0
...                    1                    0              0
...                    1                    0              0
...                    1                    0              1
...                    1                    0              2
```

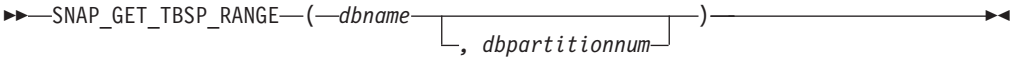
SNAP_GET_TBSP_RANGE 表関数

SNAP_GET_TBSP_RANGE 表関数は SNAPTBSP_RANGE 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションのいずれかの特定のデータベースを対象とした情報を検索することができます。

SNAP_GET_TBSP_V91、SNAP_GET_TBSP_PART_V91、SNAP_GET_TBSP_QUIESCER、および SNAP_GET_CONTAINER_V91 表関数と共に使用すると、SNAP_GET_TBSP_RANGE 表関数は、GET SNAPSHOT FOR TABLESPACES ON database-alias CLP コマンドと同等の情報を提供します。

戻される可能性のある情報の完全なリストは、 780 ページの表 196を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL または空ストリングを指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプが含まれるファイルが存在しない場合、SNAP_GET_TBSP_RANGE 表関数は、現在接続されているデータベースのスナップショットとデータベース・パーティション番号を取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_TBSP_RANGE 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベース・パーティション上で tbsp_id = 2 である表スペースの表スペース範囲の情報を選択します。

```
SELECT TBSP_ID, SUBSTR(TBSP_NAME, 1, 15) AS TBSP_NAME, RANGE_NUMBER,
   RANGE_STRIPE_SET_NUMBER, RANGE_OFFSET, RANGE_MAX_PAGE, RANGE_MAX_EXTENT,
   RANGE_START_STRIPE, RANGE_END_STRIPE, RANGE_ADJUSTMENT,
   RANGE_NUM_CONTAINER, RANGE_CONTAINER_ID
   FROM TABLE(SNAP_GET_TBSP_RANGE('',-1)) AS T WHERE TBSP_ID = 2
```

以下はこの照会の出力例です。

```
TBSP_ID     TBSP_NAME       RANGE_NUMBER     ...
-------...- --------------- ------------...- ...
          2 USERSPACE1                    0 ...

  1 record(s) selected.
```

この照会からの出力 (続き)。

```
... RANGE_STRIPE_SET_NUMBER RANGE_OFFSET     RANGE_MAX_PAGE        ...
... ----------------------- ------------...- --------------...---- ...
...                       0               0                  3967 ...
```

この照会からの出力 (続き)。

```
... RANGE_MAX_EXTENT     RANGE_START_STRIPE   RANGE_END_STRIPE     ...
... -------------------- -------------------- -------------------- ...
...                  123                    0                  123 ...
```

この照会からの出力 (続き)。

```
... RANGE_ADJUSTMENT     RANGE_NUM_CONTAINER RANGE_CONTAINER_ID
... -------------------- ------------------- --------------------
...                    0                   1                    0
```

戻される情報

表 226. SNAPTBSP_RANGE 管理ビューおよび SNAP_GET_TBSP_RANGE 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
TBSP_ID	BIGINT	tablespace_id - 表スペース ID
TBSP_NAME	VARCHAR(128)	tablespace_name - 表スペース名
RANGE_NUMBER	BIGINT	range_number - 範囲番号
RANGE_STRIPE_SET_NUMBER	BIGINT	range_stripe_set_number - ストライプ・セット番号
RANGE_OFFSET	BIGINT	range_offset - 範囲オフセット
RANGE_MAX_PAGE	BIGINT	range_max_page_number - 範囲内の最大ページ
RANGE_MAX_EXTENT	BIGINT	range_max_extent - 範囲内の最大エクステント
RANGE_START_STRIPE	BIGINT	range_start_stripe - 開始ストライプ
RANGE_END_STRIPE	BIGINT	range_end_stripe - 終了ストライプ
RANGE_ADJUSTMENT	BIGINT	range_adjustment - 範囲調整
RANGE_NUM_CONTAINER	BIGINT	range_num_containers - 範囲内コンテナーの数
RANGE_CONTAINER_ID	BIGINT	range_container_id - 範囲コンテナー

表 226. SNAPTBSP_RANGE 管理ビューおよび SNAP_GET_TBSP_RANGE 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

SNAPUTIL 管理ビューおよび SNAP_GET_UTIL 表関数 - utility_info 論理データ・グループ・スナップショット情報の検索

SNAPUTIL 管理ビューおよび SNAP_GET_UTIL 表関数は、utility_info 論理データ・グループからのユーティリティー・スナップショット情報を戻します。

SNAPUTIL 管理ビュー

SNAPUTIL_PROGRESS 管理ビューと組み合わせて使用すると、SNAPUTIL 管理ビューは、LIST UTILITIES SHOW DETAIL CLP コマンドと同じ情報を提供します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、783 ページの表 197を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPUTIL 管理ビューに対する SELECT 特権
- SNAPUTIL 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_UTIL 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

接続されているデータベースが含まれるインスタンス内のすべてのアクティブ・データベースのすべてのデータベース・パーティション上にある、ユーティリティーとその状態のリストを検索します。

```
SELECT UTILITY_TYPE, UTILITY_PRIORITY, SUBSTR(UTILITY_DESCRIPTION, 1, 72)
   AS UTILITY_DESCRIPTION, SUBSTR(UTILITY_DBNAME, 1, 17) AS
   UTILITY_DBNAME, UTILITY_STATE, UTILITY_INVOKER_TYPE, DBPARTITIONNUM
   FROM SYSIBMADM.SNAPUTIL ORDER BY DBPARTITIONNUM
```

以下はこの照会の出力例です。

```
UTILITY_TYPE     UTILITY_PRIORITY ...
------------...- ---------------- ...
LOAD                            - ...
LOAD                            - ...
LOAD                            - ...

3 record(s) selected.
```

この照会からの出力 (続き)。

```
... UTILITY_DESCRIPTION                                                    ...
... -----------------------------------------------------------------------...
... ONLINE LOAD DEL AUTOMATIC INDEXING INSERT COPY NO TEST .LOADTEST ...
... ONLINE LOAD DEL AUTOMATIC INDEXING INSERT COPY NO TEST .LOADTEST ...
... ONLINE LOAD DEL AUTOMATIC INDEXING INSERT COPY NO TEST .LOADTEST ...
```

この照会からの出力 (続き)。

```
... UTILITY_DBNAME    UTILITY_STATE UTILITY_INVOKER_TYPE DBPARTITIONNUM
... ----------------- ------------- -------------------- --------------
... SAMPLE            EXECUTE       USER                            0
... SAMPLE            EXECUTE       USER                            1
... SAMPLE            EXECUTE       USER                            2
```

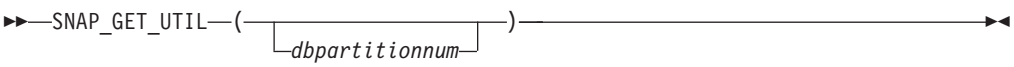
SNAP_GET_UTIL 表関数

SNAP_GET_UTIL 表関数は SNAPUTIL 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションを対象とした情報を検索することができます。

SNAP_GET_UTIL_PROGRESS 表関数と組み合わせて使用すると、SNAP_GET_UTIL 表関数は、LIST UTILITIES SHOW DETAIL CLP コマンドと同じ情報を提供します。

戻される可能性のある情報の完全なリストは、 783 ページの表 197を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。この入力オプションが使用されない場合、データはすべてのアクティブなデータベー

ス・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbpartitionnum が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプが含まれるファイルが存在しない場合、SNAP_GET_UTIL 表関数は、現在接続されているデータベースのスナップショットとデータベース・パーティション番号を取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_UTIL 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

データベース SAMPLE 上の現在接続されているデータベース・パーティションのユーティリティー ID (そのタイプと状態を含む) のリストを検索します。

```
SELECT UTILITY_ID, UTILITY_TYPE, STATE
   FROM TABLE(SNAP_GET_UTIL(-1)) AS T WHERE UTILITY_DBNAME='SAMPLE'
```

以下はこの照会の出力例です。

```
UTILITY_ID           UTILITY_TYPE               STATE
-------------------- -------------------------- --------
                   1 BACKUP                     EXECUTE

  1 record(s) selected.
```

戻される情報

表 227. SNAPUTIL 管理ビューおよび SNAP_GET_UTIL 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
UTILITY_ID	INTEGER	utility_id - ユーティリティー ID。データベース・パーティションに固有。

表 227. SNAPUTIL 管理ビューおよび SNAP_GET_UTIL 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
UTILITY_TYPE	VARCHAR(26)	utility_type - ユーティリティー・タイプ。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • ASYNC_INDEX_CLEANUP • BACKUP • CRASH_RECOVERY • LOAD • REBALANCE • REDISTRIBUTE • REORG • RESTART_RECREATE_INDEX • RESTORE • ROLLFORWARD_RECOVERY • RUNSTATS
UTILITY_PRIORITY	INTEGER	utility_priority - ユーティリティー優先度。ユーティリティーがスロットルをサポートする場合には優先順位、それ以外の場合は NULL。
UTILITY_DESCRIPTION	VARCHAR(2048)	utility_description - ユーティリティー記述。NULL にすることもできます。
UTILITY_DBNAME	VARCHAR(128)	utility_dbname - ユーティリティーで操作されるデータベース
UTILITY_START_TIME	TIMESTAMP	utility_start_time - ユーティリティー開始時刻
UTILITY_STATE	VARCHAR(10)	utility_state - ユーティリティー状態。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • ERROR • EXECUTE • WAIT
UTILITY_INVOKER_TYPE	VARCHAR(10)	utility_invoker_type - ユーティリティー呼び出し側タイプ。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • AUTO • USER

表 227. SNAPUTIL 管理ビューおよび SNAP_GET_UTIL 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。
PROGRESS_LIST_ATTR	VARCHAR(10)	progress_list_attr - 現在の進行リストの属性
PROGRESS_LIST_CUR_SEQ_NUM	INTEGER	progress_list_current_seq_num - 現行の進行リストのシーケンス番号

SNAPUTIL_PROGRESS 管理ビューおよび SNAP_GET_UTIL_PROGRESS 表関数 - progress 論理データ・グループ・スナップショット情報の検索

SNAPUTIL_PROGRESS 管理ビューおよび SNAP_GET_UTIL_PROGRESS 表関数は、特に progress 論理データ・グループの、ユーティリティー進行状況のスナップショット情報を戻します。

SNAPUTIL_PROGRESS 管理ビュー

SNAPUTIL 管理ビューと組み合わせて使用すると、SNAPUTIL_PROGRESS 管理ビューは、LIST UTILITIES SHOW DETAIL CLP コマンドと同じ情報を提供します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、787 ページの表 198を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPUTIL_PROGRESS 管理ビューに対する SELECT 特権
- SNAPUTIL_PROGRESS 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_UTIL_PROGRESS 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

ユーティリティー ID ごとの合計進行単位および完了した進行単位の詳細を検索します。

```
SELECT SELECT UTILITY_ID, PROGRESS_TOTAL_UNITS, PROGRESS_COMPLETED_UNITS,
   DBPARTITIONNUM FROM SYSIBMADM.SNAPUTIL_PROGRESS
```

以下はこの照会の出力例です。

```
UTILITY_ID PROGRESS_TOTAL_UNITS PROGRESS_COMPLETED_UNITS DBPARTITIONNU
---------- ------------------- ----------------------- ------------
         7                  10                       5            0
         9                  10                       5            1

  1 record(s) selected.
```

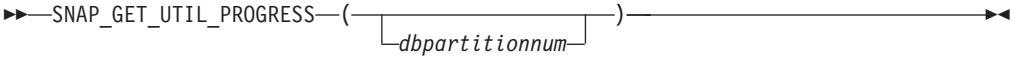
SNAP_GET_UTIL_PROGRESS 表関数

SNAP_GET_UTIL_PROGRESS 表関数は SNAPUTIL_PROGRESS 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションのいずれかの特定のデータベースを対象とした情報を検索することができます。

SNAP_GET_UTIL 表関数と組み合わせて使用すると、SNAP_GET_UTIL_PROGRESS 表関数は、LIST UTILITIES SHOW DETAIL CLP コマンドと同じ情報を提供します。

戻される可能性のある情報の完全なリストは、 787 ページの表 198を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。この入力オプションが使用されない場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbpartitionnum が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプが含まれるファイルが存在しない場合、SNAP_GET_UTIL_PROGRESS 表関数は、現在接続されてい

るデータベースのスナップショットとデータベース・パーティション番号を取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_UTIL_PROGRESS 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているパーティション上のユーティリティーの進行状況の詳細を検索します。

```
SELECT UTILITY_ID, PROGRESS_TOTAL_UNITS, PROGRESS_COMPLETED_UNITS,
   DBPARTITIONNUM FROM TABLE(SNAP_GET_UTIL_PROGRESS(-1)) as T
```

以下はこの照会の出力例です。

```
UTILITY_ID PROGRESS_TOTAL_UNITS PROGRESS_COMPLETED_UNITS DBPARTITIONNUM
---------- -------------------- ------------------------ --------------
         7                   10                        5              0

  1 record(s) selected.
```

戻される情報

表 228. SNAPUTIL_PROGRESS 管理ビューおよび SNAP_GET_UTIL_PROGRESS 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
UTILITY_ID	INTEGER	utility_id - ユーティリティー ID。データベース・パーティションに固有。
PROGRESS_SEQ_NUM	INTEGER	progress_seq_num - 進行シーケンス番号。逐次の場合、フェーズの数。並行の場合、NULL の場合があります。

表 228. SNAPUTIL_PROGRESS 管理ビューおよび SNAP_GET_UTIL_PROGRESS 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
UTILITY_STATE	VARCHAR(16)	utility_state - ユーティリティー状態。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • ERROR • EXECUTE • WAIT
PROGRESS_DESCRIPTION	VARCHAR(2048)	progress_description - 進行の記述
PROGRESS_START_TIME	TIMESTAMP	progress_start_time - 進行開始時刻。フェーズが開始済みの場合には開始時刻、それ以外の場合は NULL。
PROGRESS_WORK_METRIC	VARCHAR(16)	progress_work_metric - 進行作業メトリック。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • NOT_SUPPORT • BYTES • EXTENTS • INDEXES • PAGES • ROWS • TABLES
PROGRESS_TOTAL_UNITS	BIGINT	progress_total_units - 合計進行作業単位
PROGRESS_COMPLETED_UNITS	BIGINT	progress_completed_units - 完了した進行作業単位
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

SNAP_WRITE_FILE プロシージャー

SNAP_WRITE_FILE プロシージャーはシステム・スナップショット・データを、インスタンス・ディレクトリーの tmp サブディレクトリーにあるファイルに書き込みます。

構文

スキーマは SYSPROC です。

プロシージャー・パラメーター

requestType
有効なスナップショット要求タイプを指定する、タイプ VARCHAR(32) の入力引数。可能な要求タイプは、sqlmon.h での定義を基にしたテキスト ID です。以下のいずれかとなります。

- APPL_ALL
- BUFFERPOOLS_ALL
- DB2
- DBASE_ALL
- DBASE_LOCKS
- DBASE_TABLES
- DBASE_TABLESPACES
- DYNAMIC_SQL

dbname
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。"Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL または空ストリングを指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

プロシージャーを実行するには、ユーザーに SYSADM、SYSCTRL、SYSMON、または SYSMON 権限が必要です。保存されたスナップショットは、スナップショット表関数への入力として NULL 値を渡すことにより、SYSADM、 SYSCTRL、SYSMAINT、または SYSMON 権限のないユーザーでも読み取れます。

例

'DB2' の要求タイプ (SQLMA_DB2 に相当) を指定し、現在接続されているデータベースおよび現行データベース・パーティションをデフォルトにすることで、データベース・マネージャー情報のスナップショットをとります。

```
CALL SYSPROC.SNAP_WRITE_FILE ('DB2', '', -1)
```

この場合、スナップショット・データは、インスタンス一時ディレクトリー (UNIX オペレーティング・システムでは sqllib/tmp/SQLMA_DB2.dat、Windows オペレー

ティング・システムでは sqllib¥DB2¥tmp¥SQLMA_DB2.dat) に書き込まれます。

使用上の注意

未認識の入力パラメーターが指定された場合、「SQL2032N "REQUEST_TYPE" パラメーターが無効です」エラーが戻されます。

TBSP_UTILIZATION 管理ビュー - 表スペースの構成および使用率に関する情報の検索

TBSP_UTILIZATION 管理ビューは、表スペースの構成および使用率に関する情報を戻します。これは、単一パーティション・データベースに対する LIST TABLESPACES コマンドと同様のレポートを取得します。その情報は、SNAPTBSP、SNAPTBSP_PART 管理ビューと TABLESPACES カタログ・ビューに基づきます。

スキーマは SYSIBMADM です。

許可

以下のいずれかの権限が必要です。

- TBSP_UTILIZATION 管理ビューに対する SELECT 特権
- TBSP_UTILIZATION 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

単一パーティション・データベース上での LIST TABLESPACES コマンドと同じレポートを検索します。

```
SELECT TBSP_ID, SUBSTR(TBSP_NAME,1,20) as TBSP_NAME, TBSP_TYPE,
   TBSP_CONTENT_TYPE, TBSP_STATE FROM SYSIBMADM.TBSP_UTILIZATION
```

以下はこの照会の出力例です。

```
TBSP_ID     TBSP_NAME            TBSP_TYPE  ...
--------...- -------------------- ---------- ...
          0 SYSCATSPACE          SMS        ...
          1 TEMPSPACE1           SMS        ...
          2 USERSPACE1           SMS        ...
          3 SYSTOOLSPACE         SMS        ...
          4 SYSTOOLSTMPSPACE     SMS        ...
```

この照会の出力 (続き)。

```
... TBSP_CONTENT_TYPE TBSP_STATE
... ---------------- -----------
... ANY              NORMAL
... SYSTEMP          NORMAL
... ANY              NORMAL
... ANY              NORMAL
... USRTEMP          NORMAL
```

戻される情報

表 229. TBSP_UTILIZATION 管理ビューによって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
TBSP_ID	BIGINT	tablespace_id - 表スペース ID
TBSP_NAME	VARCHAR(128)	tablespace_name - 表スペース名
TBSP_TYPE	VARCHAR(10)	tablespace_type - 表スペース・タイプ
TBSP_CONTENT_TYPE	VARCHAR(10)	tablespace_content_type - 表スペースのコンテンツ・タイプ
TBSP_CREATE_TIME	TIMESTAMP	表スペースの作成時刻。
TBSP_STATE	VARCHAR(256)	tablespace_state - 表スペースの状態
TBSP_TOTAL_SIZE_KB	BIGINT	表スペースの合計サイズ (KB)。計算式は total_pages*pagesize/1024。
TBSP_USABLE_SIZE_KB	BIGINT	表スペースの合計使用可能サイズ (KB)。計算式は usable_pages*pagesize/1024。
TBSP_USED_SIZE_KB	BIGINT	表スペースの合計使用済みサイズ (KB)。計算式は used_pages*pagesize/1024。
TBSP_FREE_SIZE_KB	BIGINT	表スペースの合計利用可能サイズ (KB)。計算式は free_pages*pagesize/1024。
TBSP_UTILIZATION_PERCENT	BIGINT	表スペースの使用率 (パーセンテージ)。 usable_pages が使用可能な場合、計算式は (used_pages/ usable_pages)*100。使用可能でない場合には -1 が表示されます。
TBSP_TOTAL_PAGES	BIGINT	tablespace_total_pages - 表スペース内の合計ページ数
TBSP_USABLE_PAGES	BIGINT	tablespace_usable_pages - 表スペース内の使用可能ページ数
TBSP_USED_PAGES	BIGINT	tablespace_used_pages - 表スペース内の使用されているページ数
TBSP_FREE_PAGES	BIGINT	tablespace_free_pages - 表スペース内のフリー・ページ数
TBSP_PAGE_TOP	BIGINT	tablespace_page_top - 表スペース最高水準点

表 229. TBSP_UTILIZATION 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
TBSP_PAGE_SIZE	INTEGER	tablespace_page_size - 表スペースのページ・サイズ
TBSP_EXTENT_SIZE	INTEGER	tablespace_extent_size - 表スペースのエクステント・サイズ
TBSP_PREFETCH_SIZE	BIGINT	tablespace_prefetch_size - 表スペースのプリフェッチ・サイズ
TBSP_MAX_SIZE	BIGINT	tablespace_max_size - 表スペースの最大サイズ
TBSP_INCREASE_SIZE	BIGINT	tablespace_increase_size - バイト単位のサイズの増加
TBSP_INCREASE_SIZE_PERCENT	SMALLINT	tablespace_increase_size_percent - パーセント単位のサイズの増加
TBSP_LAST_RESIZE_TIME	TIMESTAMP	tablespace_last_resize_time - 最後にサイズ変更が正常に行われた時刻
TBSP_LAST_RESIZE_FAILED	SMALLINT	tablespace_last_resize_failed - 失敗した最後のサイズ変更
TBSP_USING_AUTO_STORAGE	SMALLINT	tablespace_using_auto_storage - 自動ストレージが使用可能な表スペース
TBSP_AUTO_RESIZE_ENABLED	SMALLINT	tablespace_auto_resize_enabled - 表スペースの自動サイズ変更可能
DBPGNAME	VARCHAR(128)	表スペースのデータベース・パーティション・グループの名前。
TBSP_NUM_CONTAINERS	BIGINT	tablespace_num_containers - 表スペース内のコンテナー数
REMARKS	VARCHAR(254)	ユーザーが入力したコメント。
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

TOP_DYNAMIC_SQL 管理ビュー - 上位動的 SQL ステートメントに関する情報の検索

TOP_DYNAMIC_SQL 管理ビューは、実行数、平均実行時間、ソート数、またはステートメントあたりのソートによってソートできる動的 SQL ステートメントのうち、上位のものを戻します。これは適切な調整を行うために注目すべき照会です。

スキーマは SYSIBMADM です。

許可

以下のいずれかの権限が必要です。

- TOP_DYNAMIC_SQL 管理ビューに対する SELECT 特権
- TOP_DYNAMIC_SQL 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

実行頻度の高い上位 5 つの SQL を識別します。

```
SELECT NUM_EXECUTIONS, AVERAGE_EXECUTION_TIME_S, STMT_SORTS,
   SORTS_PER_EXECUTION, SUBSTR(STMT_TEXT,1,60) AS STMT_TEXT
   FROM SYSIBMADM.TOP_DYNAMIC_SQL
   ORDER BY NUM_EXECUTIONS DESC FETCH FIRST 5 ROWS ONLY
```

以下はこの照会の出力例です。

```
NUM_EXECUTIONS       AVERAGE_EXECUTION_TIME_S STMT_SORTS           ...
-------------------- ------------------------ -------------------- ...
                 148                        0                    0 ...
                 123                        0                    0 ...
                   2                        0                    0 ...
                   1                        0                    0 ...
                   1                        0                    0 ...

  5 record(s) selected.
```

この照会の出力 (続き)。

```
... SORTS_PER_EXECUTION  ...
... -------------------- ...
...                    0 ...
...                    0 ...
...                    0 ...
...                    0 ...
...                    0 ...
```

この照会の出力 (続き)。

```
... STMT_TEXT
... ------------------------------------------------------------
... SELECT A.ID, B.EMPNO, B.FIRSTNME, B.LASTNAME, A.DEPT FROM E
... SELECT A.EMPNO, A.FIRSTNME, A.LASTNAME, B.LOCATION, B.MGRNO
... SELECT A.EMPNO, A.FIRSTNME, A.LASTNAME, B.DEPTNAME FROM EMP
... SELECT ATM.SCHEMA, ATM.NAME, ATM.CREATE_TIME, ATM.LAST_WAIT,
... SELECT * FROM JESSICAE.EMP_RESUME
```

戻される情報

表 230. TOP_DYNAMIC_SQL 管理ビューによって戻される情報

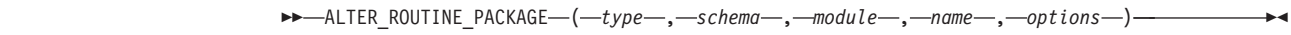
列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	レポートのタイム・スタンプ。
NUM_EXECUTIONS	BIGINT	num_compilations - ステートメント・コンパイル数
AVERAGE_EXECUTION_TIME_S	BIGINT	平均実行時間。
STMT_SORTS	BIGINT	stmt_sorts - ステートメント・ソート回数

表 230. TOP_DYNAMIC_SQL 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
SORTS_PER_EXECUTION	BIGINT	ステートメントの実行ごとに行われるソートの数。
STMT_TEXT	CLOB(2 M)	stmt_text - SQL 動的ステートメント・テキスト
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

第 16 章 SQL プロシージャー・ルーチン

ALTER_ROUTINE_PACKAGE プロシージャー



スキーマは SYSPROC です。

このプロシージャーは、コンパイル済み SQL ルーチンまたはコンパイル済みトリガーに関連付けられたパッケージに使用される値を、再バインドを必要とせずに変更します。機能的には ALTER PACKAGE コマンドと同等ですが、パッケージ名の代わりにオブジェクト名を引数として使用します。 ALTER_ROUTINE_PACKAGE プロシージャーは、コマンド行またはアプリケーションから呼び出すことができます。

type
以下のいずれかの値を使用してルーチンまたはコンパイル・トリガーのタイプを指定する、タイプ CHAR(2) の入力引数。

- 'P' - プロシージャー
- 'SP' - プロシージャーの特定名
- 'F' - コンパイル関数
- 'SF' - コンパイル関数の特定名
- 'T' - コンパイル・トリガー

schema
ルーチンまたはトリガーのスキーマを指定する、タイプ VARCHAR(128) のオプションの入力引数。スキーマが指定されない場合、値はデフォルトで CURRENT SCHEMA 特殊レジスターの値になります。このパラメーターには、大文字と小文字の区別があります。

module
ルーチンがあるモジュールの名前を指定する、タイプ VARCHAR(128) のオプションの入力引数。トリガーの場合、このパラメーターを指定することはできません。このパラメーターを指定しない場合、モジュール・ルーチンは無視されます。このパラメーターには、大文字と小文字の区別があります。

name
ルーチンまたはトリガーの名前を指定する、タイプ VARCHAR(128) の入力引数。このパラメーターには、大文字と小文字の区別があります。

options
ALTER PACKAGE ステートメントがサポートしているオプションのリストを指定する、タイプ VARCHAR(1024) の入力引数。 *options* パラメーター内に、ALTER PACKAGE 節を少なくとも 1 つ指定する必要があります。

© Copyright IBM Corp. 2006, 2010

例

UPDATE_EMPLOYEE という名前の既存のストアード・プロシージャーの基礎パッケージを変更します。

```
CALL SYSPROC.ALTER_ROUTINE_PACKAGE ('P','','','UPDATE_EMPLOYEE',
   'ACCESS PLAN REUSE YES OPTIMIZATION PROFILE AYYANG.INDEXHINTS')
```

DRICARD スキーマの MIN_SALARY というコンパイル済みトリガーのパッケージを変更します。

```
CALL SYSPROC.ALTER_ROUTINE_PACKAGE ('T','DRICARD','','MIN_SALARY',
   'OPTIMIZATION PROFILE AYYANG.INDEXHINTS')
```

3 部構成の名前を使用して、コンパイル済み関数のパッケージを変更します。

```
CALL SYSPROC.ALTER_ROUTINE_PACKAGE ('F','DRICARD','MODULE','FUNCTION','APREUSE YES')
```

GET_ROUTINE_OPTS

スキーマは SYSPROC です。

GET_ROUTINE_OPTS 関数は、現行のセッションで SQL プロシージャーを作成するのに使用されるオプションの文字ストリング値を戻します。

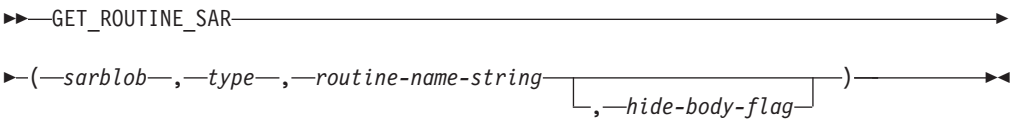
この関数の結果は、長さ属性が 1024 である、可変長文字ストリング (VARCHAR) 値です。

例:

照会の結果として SQL プロシージャーを作成するのに使用されるオプションを戻します。

```
SELECT GET_ROUTINE_OPTS()
  FROM SYSIBM.SYSDUMMY1
```

GET_ROUTINE_SAR



スキーマは SYSFUN です。

GET_ROUTINE_SAR プロシージャーは、同じオペレーティング・システムで同じレベルを実行している別のデータベース・サーバーで、同じルーチンをインストールするために必要な情報を検索します。情報は、SQL アーカイブ・ファイルを表す単一 BLOB ストリングへ取り出されます。

許可

GET_ROUTINE_SAR プロシージャーに対する EXECUTE 特権。

sarblob
ルーチン SAR ファイル・コンテンツを含む、タイプ BLOB(3M) の出力引数。

type
以下のいずれかの値を使用してルーチンのタイプを指定する、タイプ CHAR(2) の入力引数。

- 'P' - プロシージャー
- 'SP' - プロシージャーの特定名

routine-name-string
ルーチンの修飾名を指定する、タイプ VARCHAR(257) の入力引数。スキーマ名が指定されていない場合、ルーチンが処理されるときデフォルトは CURRENT SCHEMA になります。 *routine-name-string* に二重引用符 (") を組み込むことはできません。

hide-body-flag
カタログからルーチンのテキストを抽出する際にルーチンの本文を隠すかどうかを指定する、タイプ INTEGER の入力引数 (次のいずれかの値を使用します)。有効な値は以下のとおりです。

0 ルーチンのテキストをそのまま残す。これはデフォルト値です。

1 カタログからルーチンのテキストを抽出する際に、ルーチンの本文を空の本文に置き換える。

ルーチンの修飾名は、検索するルーチンを決定するために使用されます。検出されるルーチンは、SQL ルーチンでなければなりません。また、特定の名前を使用せずに検索を行うと、複数のルーチンが検出されて、エラー (SQLSTATE 42725) になる場合があります。このような場合は、必ず、検索したいルーチンの固有の名前を使用してください。

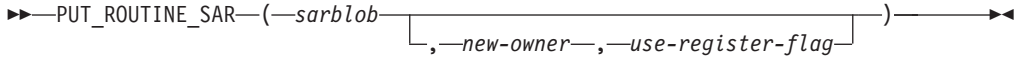
SAR ファイルは、サーバーで使用可能ではない可能性のあるバインド・ファイルを組み込む必要があります。バインド・ファイルが見つからず、SAR ファイルに保管できない場合、エラーが起こります (SQLSTATE 55045)。

PUT_ROUTINE_SAR

PUT_ROUTINE_SAR プロシージャーは、サーバーで SQL ルーチンを作成するために必要なファイルを渡し、ルーチンを定義します。

許可

DBADM



スキーマは SYSFUN です。

sarblob
ルーチン SAR ファイル・コンテンツを含む、タイプ BLOB(3M) の入力引数。

new-owner
ルーチンの許可検査に使用される許可名を含む、タイプ VARCHAR(128) の入力引数。 *new-owner* は、定義されるルーチンに必要な権限を持っていなければなりません。 *new-owner* が指定されない場合、オリジナル・ルーチン定義者の許可名が使用されます。

use-register-flag
CURRENT SCHEMA および CURRENT PATH 特殊レジスターがルーチンの定義に使用されるかどうかを指示する、タイプ INTEGER の入力引数。特殊レジスターが使用されない場合、ルーチンがはじめに定義されるときはデフォルト・スキーマと SQL パスの設定が使用されます。 *use-register-flag* に考えられる値は以下のとおりです。

0 現行環境の特殊レジスターを使用しません。

1 CURRENT SCHEMA および CURRENT PATH 特殊レジスターを使用します。

値が 1 の場合、ルーチン定義 (ルーチンの名前を含む) の非修飾オブジェクト名に CURRENT SCHEMA が使用され、ルーチン定義の非修飾ルーチンとデータ・タイプを解決するために CURRENT PATH が使用されます。
use-registers-flag が指定されない場合、振る舞いは 0 が指定されたときと同じです。

sarblob に含まれる識別情報がチェックされ、入力が環境に対して適切であるかどうかが確認されます。入力が適切でない場合は、エラーが起こります (SQLSTATE 55046)。 PUT_ROUTINE_SAR プロシージャーは次に、*sarblob* の内容を使用してサーバーでルーチンを定義します。

sarblob 引数の内容は SQL アーカイブ・ファイルを構成する個々のファイルへ抽出されます。共有ライブラリーとバインド・ファイルは、一時ディレクトリーのファイルに書き込まれます。環境は、コンパイルおよびリンクが不要であること、共有ライブラリーとバインド・ファイルのロケーションが使用可能であることを、ルーチン定義ステートメント処理に認識させるように設定されます。 DDL ファイルの内容が、ルーチン定義ステートメントを動的に実行するために使用されます。

指定のスキーマの下で、複数のプロシージャーが並行してインストールされることはありません。

このステートメントの処理は、他のインターフェースを使用してルーチン定義ステートメントを実行するのと同じエラーを起こす可能性があります。ルーチン定義処理中に、共有ライブラリーとバインド・ファイルの存在が通知され、プリコンパイル、コンパイル、およびリンクのステップがスキップされます。バインド・ファイルはバインド処理中に使用され、両方のファイルの内容が SQL ルーチンの通常ディレクトリーにコピーされます。

GET ROUTINE または PUT ROUTINE 操作 (またはそれに対応するプロシージャー) が正常に実行できない場合、エラー (SQLSTATE 38000)、および失敗の原因に関する情報を示す診断テキストを毎回戻します。例えば、GET ROUTINE に指定さ

れたプロシージャー名が SQL プロシージャーを識別しない場合、 "-204, 42704" という診断テキストが戻されます。 "-204" は SQLCODE、"42704" は SQLSTATE で、それぞれ問題の原因を示します。この例の SQLCODE と SQLSTATE は、GET ROUTINE コマンドで指定されたプロシージャー名が未定義であることを示します。

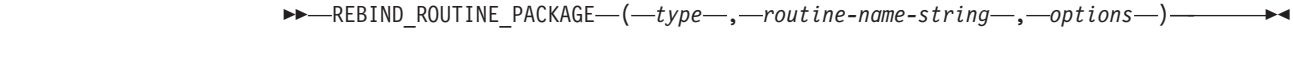
REBIND_ROUTINE_PACKAGE プロシージャー - パッケージの再バインド

REBIND_ROUTINE_PACKAGE プロシージャーは、SQL プロシージャー、ルーチン、コンパイル済み関数、またはトリガーに関連付けられたパッケージを再バインドします。これは機能的には REBIND コマンドと同じですが、これは、パッケージ名の代わりにプロシージャー名を引数として使用します。
REBIND_ROUTINE_PACKAGE プロシージャーは、コマンド行またはアプリケーションから呼び出すことができます。

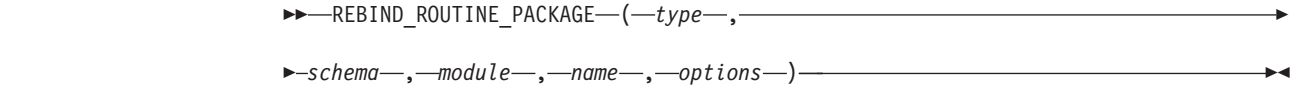
構文

REBIND_ROUTINE_PACKAGE を呼び出すには、いずれも同じように有効な 2 つの方法があります。 2 つの呼び出しの違いは、ルーチン名を指定する方法だけです。 1 つ目の例では、*routine-name-string* 変数はピリオドで区切られた ID 名で構成されます。 2 つ目の方法では、ルーチンは *schema*、*module*、および *name* 値という別個の値で識別されます。

方法 1:



方法 2:



スキーマは SYSPROC です。

プロシージャー・パラメーター

type
以下のいずれかの値を使用してルーチンまたはコンパイル・トリガーのタイプを指定する、タイプ CHAR(2) の入力引数。

- 'P' - プロシージャー
- 'SP' - プロシージャーの特定名
- 'F' - コンパイル関数
- 'SF' - コンパイル関数の特定名
- 'T' - コンパイル・トリガー

routine-name-string **(方法 1 のみ)**
ルーチンまたはトリガーの名前を指定する、タイプ VARCHAR(386) の入力引数。トリガー名はピリオドで区切られた 2 つの部分で構成されていて、

schema.trigger という形式です。スキーマはオプションです。ルーチン名はピリオドで区切られた 3 つの名前部分で構成されていて、schema.module.routine という形式です。スキーマとモジュールはオプションです。スキーマを指定しないと、この値はデフォルトの CURRENT SCHEMA 特殊レジスターの値になります。 2 部構成の名前が指定されると、最初の部分はスキーマ名として当初解釈されます。そのスキーマ名の下にルーチンが見つからないと、最初の部分はモジュール名として解釈され、CURRENT SCHEMA の下のその名前のモジュール内でルーチンを検出しようとします。スキーマ、モジュール、またはオブジェクト名に、二重引用符 (") またはピリオド (.) を含めることはできません。

schema **(方法 2 のみ)**
ルーチンまたはトリガーのスキーマを指定する、タイプ VARCHAR(128) のオプションの入力引数。スキーマが指定されない場合、値はデフォルトで CURRENT SCHEMA 特殊レジスターの値になります。このパラメーターには、大文字と小文字の区別があります。

module **(方法 2 のみ)**
ルーチンがあるモジュールの名前を指定する、タイプ VARCHAR (128) のオプションの入力引数。このパラメーターをトリガーに指定しないでください。このパラメーターを指定しない場合は、モジュール・ルーチンは無視されます。このパラメーターには、大文字と小文字の区別があります。

name **(方法 2 のみ)**
ルーチンまたはトリガーの名前を指定する、タイプ VARCHAR (128) の入力引数。このパラメーターには、大文字と小文字の区別があります。

options
REBIND コマンド構文に従うすべての再バインド・オプションのリストを指定する、タイプ VARCHAR(1024) のオプションの入力引数。後方互換性の目的で「ANY」または「CONSERVATIVE」という単一値もサポートされており、RESOLVE 再バインド・オプションの値として解釈されます。

ルーチンの修飾名は、検索するルーチンを決定するために使用されます。検出されるルーチンは、SQL ルーチンでなければなりません。それ以外は、エラーが返されます。(SQLSTATE 428F7) 特定名が使用されない場合は、複数のルーチンが検出される可能性があり、エラーが返されます。(SQLSTATE 42725) このような場合は、必ず、検索したいルーチンの固有の名前を使用してください。

例

例 1: RESOLVE、REOPT、および APREUSE オプションを使用して、ルーチン UPDATE_EMPLOYEE のパッケージを再バインドします。

```
方法 1:
CALL SYSPROC.REBIND_ROUTINE_PACKAGE (
  'P','UPDATE_EMPLOYEE','RESOLVE ANY REOPT ONCE APREUSE YES')
方法 2:
CALL SYSPROC.REBIND_ROUTINE_PACKAGE (
  'P','','','UPDATE_EMPLOYEE','RESOLVE ANY REOPT ONCE APREUSE YES')
```

例 2: オプションを使用せずに、ルーチン UPDATE_EMPLOYEE のパッケージを再バインドします。

```
方法 1:
CALL SYSPROC.REBIND_ROUTINE_PACKAGE (
  'P','UPDATE_EMPLOYEE','')
方法 2:
CALL SYSPROC.REBIND_ROUTINE_PACKAGE (
  'P','','','UPDATE_EMPLOYEE','')
```

例 3: コンパイル済みトリガーのパッケージを再バインドします。

```
方法 1:
CALL SYSPROC.REBIND_ROUTINE_PACKAGE (
  'T','DRICARD.MIN_SALARY','REOPT ALWAYS')
方法 2:
CALL SYSPROC.REBIND_ROUTINE_PACKAGE (
  'T','DRICARD','','MIN_SALARY','REOPT ALWAYS')
```

例 4: 3 つの部分に分かれた名前を使用して、コンパイル済み関数のパッケージを再バインドします。

```
方法 1
CALL SYSPROC.REBIND_ROUTINE_PACKAGE (
  'F','DRICARD.MODULE.FUNCTION','REOPT ALWAYS')
方法 2
CALL SYSPROC.REBIND_ROUTINE_PACKAGE (
  'F','DRICARD','MODULE','FUNCTION','REOPT ALWAYS')
```

SET_ROUTINE_OPTS

スキーマは SYSPROC です。

SET_ROUTINE_OPTS プロシージャーは、現行セッションで SQL プロシージャーを作成するのに使用されるオプションを設定します。この設定は、DB2_SQLROUTINE_PREPOPTS レジストリー変数で指定された、インスタンス全体の設定をオーバーライドします。

character-expression
現行セッションのオプション設定を指定する、タイプ VARCHAR(1024) の入力引数。

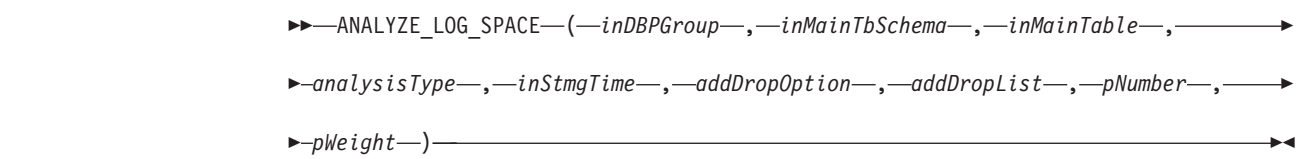
指定されたオプションは、セッション期間に有効です。引数として NULL 値が指定される場合、DB2_SQLROUTINE_PREPOPTS レジストリー変数の値は、現行セッションのデフォルトのオプション設定としてリストアされます。許可されたオプションのリストは、『照会コンパイラー変数』の下にある、DB2_SQLROUTINE_PREPOPTS レジストリー変数の説明を参照してください。

第 17 章 段階的な再配分ルーチン

ANALYZE_LOG_SPACE プロシージャー - ログ・スペース分析情報の検索

ANALYZE_LOG_SPACE プロシージャーは、指定されたデータベース・パーティション・グループのデータベース・パーティションそれぞれのログ・スペース分析結果を戻します。

構文



スキーマは SYSPROC です。

プロシージャー・パラメーター

inDBPGroup
データベース・パーティション・グループ名を指定する、タイプ VARCHAR (128) の入力引数。

inMainTbSchema
メイン表のスキーマを指定する、タイプ VARCHAR (128) の入力引数。

inMainTable
データベース・パーティション・グループ内のメイン表を指定する、タイプ VARCHAR (128) の入力引数。この表は通常、データベース・パーティション・グループ内で最大の表です。

analysisType
分析タイプの標識を指定する、タイプ SMALLINT の入力引数。以下の標識があります。

- SWRD_USE_STMG_TABLE (1): データベース・パーティションあたりの表の行数を検出するのに、ストレージ管理表内の情報が使用されることを表します。このタイプは、ストレージ管理表がセットアップされていて、再配分の対象のデータベース・パーティション・グループのストレージ・スナップショットが少なくとも 1 つ取られている場合のみ、使用してください。
- SWRD_USE_REALTIME_ANALYSIS (2): データベース・パーティションあたりの表の行数を検出するのに、SELECT 照会が使用されることを表します。

inStmgTime
ストレージ管理レコードのタイム・スタンプを指定する、タイプ VARCHAR (26) の入力引数。 *analysisType* を SWRD_USE_REALTIME_ANALYSIS に設定した場合、このパラメーターは無視されます。

© Copyright IBM Corp. 2006, 2010

addDropOption
追加またはドロップされるデータベース・パーティションを指定する、タイプ CHAR (1) の入力引数。以下の値を使用できます。

- 'A': データベース・パーティションを追加
- 'D': データベース・パーティションをドロップ
- 'N': 追加もドロップもしない

addDropList
追加またはドロップされるデータベース・パーティションを指定する、タイプ VARCHAR (6000) の入力引数。このデータベース・パーティション番号は、コンマで区切られたストリング・フォーマットで指定します。ストリング内でスペースは使用できません。

pNumber
すべてのデータベース・パーティション番号を指定するタイプ VARCHAR (6000) の入力引数で、データベース・パーティションの重みに対応します。各データベース・パーティション番号は、0 から 999 までの範囲の数値です。また、データベース・パーティション番号は、コンマで区切られたストリングで指定し、ストリング内にスペースは使用できません。

pWeight
すべてのデータベース・パーティションのユーザー指定の重みを指定するタイプ VARCHAR (6000) の入力引数で、*pNumber* ストリング内のデータベース・パーティション番号に対応します。各データベース・パーティションの重みは、0 から 32767 までの範囲の数値です。また、データベース・パーティションの重みは、コンマで区切られたストリングで指定し、ストリング内にスペースは使用できません。

許可

- SYSADM、SYSMON、SYSCTRL、または SYSMAINT
- ANALYZE_LOG_SPACE プロシージャーに対する EXECUTE 特権

例

データベース・パーティションを追加した場合の影響を、変更を適用しないで分析します。ここでは、データベース・パーティション・グループにデータベース・パーティション 40、50、60 を追加し、データベース・パーティション 10、20、30、40、50、60 の個別ターゲット率を 1:2:1:2:1:2 にした場合を仮説とします。この例では、データベース・パーティション・グループに実際に存在するのは、パーティション 10、20、30 だけであることに注意してください。

```
CALL SYSPROC.ANALYZE_LOG_SPACE('IBMDEFAULTGROUP', 'TEST',
   'EMP', 2, ' ', 'A', '40,50,60', '10,20,30,40,50,60',
   '1,2,1,2,1,2')
```

データベース・パーティションをドロップした場合の影響を、変更を適用しないで分析します。ここでは、データベース・パーティション・グループからデータベース・パーティション 30 をドロップし、データベース・パーティション 10 および 20 に、個別ターゲット率 1 : 1 でデータを再配分した場合を仮説とします。この例では、データベース・パーティション 10、20、30 がすべてデータベース・パーティション・グループに存在していなければならないことに注意してください。

```
CALL SYSPROC.ANALYZE_LOG_SPACE('IBMDEFAULTGROUP', 'TEST',
   'EMP', 2, ' ', 'D', '30', '10,20','1,1')
```

使用上の注意

パラメーターの値を取得できない場合は、その出力値として -1 が使用されます。

再配分に関するストアード・プロシージャーおよび関数は、各表の分散キーが定義されているパーティション・データベース環境でのみ動作します。

戻される情報

ANALYZE_LOG_SPACE プロシージャーは、ログ・スペース分析結果の結果セット (オープン・カーソル) を戻します。この結果セットには、指定されたデータベース・パーティション・グループのデータベース・パーティションごとに、以下のフィールドが含まれます。

表 231. ANALYZE_LOG_SPACE プロシージャーによって戻される情報

列名	列タイプ	説明
PARTITION_NUM	SMALLINT	ログ・スペース分析のデータベース・パーティション番号。
TOTAL_LOG_SIZE	BIGINT	割り振られるログ・スペースの合計 (バイト単位)。-1 はサイズの制限がないことを表します。
AVAIL_LOG_SPACE	BIGINT	フリーであり再配分処理で使用できるログ・スペース量 (バイト単位)。
DATA_SKEW	BIGINT	ターゲット・レベルから逸脱するデータのサイズの絶対値 (バイト単位)。
REQ_LOG_SPACE	BIGINT	望ましいデータ配分に到達するのに必要なスペース量 (バイト単位)。
NUM_OF_STEPS	SMALLINT	データ・スキューをゼロまで減らすのに必要なステップ数。
MAX_STEP_SIZE	BIGINT	ログ・フル・エラーにならずに一度に移動できるデータ最大量 (バイト単位)。

GENERATE_DISTFILE プロシージャー - データ配分ファイルの生成

GENERATE_DISTFILE プロシージャーは、指定された表のデータ配分ファイルを生成し、指定されたファイル名で保存します。

構文

スキーマは SYSPROC です。

プロシージャー・パラメーター

inTbSchema
表スキーマ名を指定する、タイプ VARCHAR (128) の入力引数。

inTbName
表名を指定する、タイプ VARCHAR (128) の入力引数。

fileName
データ配分ファイル名を指定する、タイプ VARCHAR (255) の入力または出力引数。指定したファイル名が単なるファイル名だった場合、ファイルはインスタンス・ディレクトリー下の tmp サブディレクトリーに保存され、このパラメーターに完全ファイル・パス名が戻されます。

許可

- GENERATE_DISTFILE プロシージャーに対する EXECUTE 特権。
- SYSCAT.TABLES、SYSCAT.COLUMNS、および指定した表に対する SELECT 特権。

加えて、fenced ユーザー ID は、インスタンス・ディレクトリー下の tmp サブディレクトリーにファイルを作成できなければなりません。

例

再配分処理で使用されるデータ配分ファイルを生成します。

```
CALL SYSPROC.GENERATE_DISTFILE('TEST', 'EMP',
   '$HOME/sqllib/function/SAMPLE.IBMDEFAULTGROUP_swrdData.dst')"
```

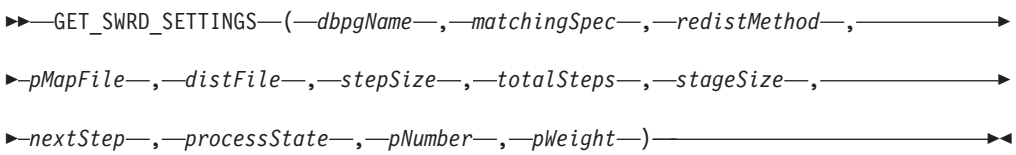
使用上の注意

再配分に関するストアード・プロシージャーおよび関数は、各表の分散キーが定義されているパーティション・データベース環境でのみ動作します。

GET_SWRD_SETTINGS プロシージャー - 再配分情報の検索

GET_SWRD_SETTINGS プロシージャーは、指定されたデータベース・パーティション・グループの既存の再配分レジストリー・レコードを読み取ります。

構文



スキーマは SYSPROC です。

プロシージャー・パラメーター

dbpgName
再配分処理の実行の対象となるデータベース・パーティション・グループ名を指定する、タイプ VARCHAR(128) の入力引数。

matchingSpec

表 232 に示すビット単位フィールド ID を指定する、タイプ SMALLINT の入力引数。出力パラメーターによって戻されるターゲット・フィールドを、これで表します。必須でない出力パラメーターは NULL 設定できます。

例えば *matchingSpec* を (REDIST_STAGE_SIZE | REDIST_NEXT_STEP) の整数値である 96 に設定した場合、この関数の呼び出し側は、値を受け取るためには *stageSize* と *nextStep* を指定するだけでよく、残りの出力パラメーターは NULL にすることができます。

表 232. ビット単位フィールド ID

フィールド名	16 進値	10 進値
REDIST_METHOD	0x0001<<0	1
REDIST_PMAP_FILE	0x0001<<1	2
REDIST_DIST_FILE	0x0001<<2	4
REDIST_STEP_SIZE	0x0001<<3	8
REDIST_NUM_STEPS	0x0001<<4	16
REDIST_STAGE_SIZE	0x0001<<5	32
REDIST_NEXT_STEP	0x0001<<6	64
REDIST_PROCESS_STATE	0x0001<<7	128
REDIST_PWEIGHT_START_NODE	0x0001<<8	256
REDIST_PWEIGHT	0x0001<<9	512

redistMethod

再配分が実行される際に、データ分散ファイルが使用されるか、またはターゲット分散マップが使用されるかを指定する、タイプ SMALLINT の出力引数。戻り値として次の 2 つがあります。

- 2: 再配分処理の入力としてデータ配分ファイルが使用されることを表します。
- 3: 再配分処理の入力としてターゲット分散マップが使用されることを表します。

pMapFile

データベース・サーバー上のターゲット分散マップの絶対パス・ファイル名を指定する、タイプ VARCHAR (255) の出力引数。

distFile

データベース・サーバー上のデータ配分ファイルの絶対パス・ファイル名を指定する、タイプ VARCHAR (255) の出力引数。

stepSize

ログ・フル状態になるのを防止するためのコミット呼び出しの前に移動可能な最大行数を指定する、タイプ BIGINT の出力引数。この数は、各再配分ステップ内で変わることがあります。

totalSteps

このデータベース・パーティション・グループを完全に再配分するために必要なステップ数を指定する、タイプ SMALLINT の出力引数。

stageSize
連続して実行されるステップ数を指定する、タイプ SMALLINT の出力引数。

nextStep
どのステップが完了していて、まだ何を実行する必要があるかを区分する索引を指定する、タイプ SMALLINT の出力引数。

processState
再配分処理が次のチェックポイントで停止するかどうかを指定する、タイプ SMALLINT の出力引数。チェックポイントは、各再配分ステップの先頭に置かれています。この引数が 1 に設定されている場合、ステップは開始されません。値が 0 の場合、ステップは進行します。

pNumber
データベース・パーティション番号がコンマで区切られたストリング・フォーマットのリスト (該当する場合) を戻す、タイプ VARCHAR (6000) の出力引数。これらのパーティション番号は、データベース・パーティション・グループが現在使用しているデータベース・パーティションか、あるいは追加またはドロップされるデータベース・パーティションです。これらのパーティション番号の順序と数は、*pWeight* 変数によって戻されるターゲット・パーティションの重みに対応します。

pWeight
ターゲット・データベース・パーティションの重み数値がコンマで区切られたリスト (該当する場合) を戻す、タイプ VARCHAR (6000) の出力引数。これらのパーティション重みの順序と数は、*pNumber* 変数によって戻されるパーティション番号に対応します。

許可

GET_SWRD_SETTINGS プロシージャーに対する EXECUTE 特権。

例

指定されたデータベース・パーティション・グループのステップ単位の再配分プランの内容を報告します。

```
CALL SYSPROC.GET_SWRD_SETTINGS
   ('IBMDEFAULTGROUP', 255, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)
```

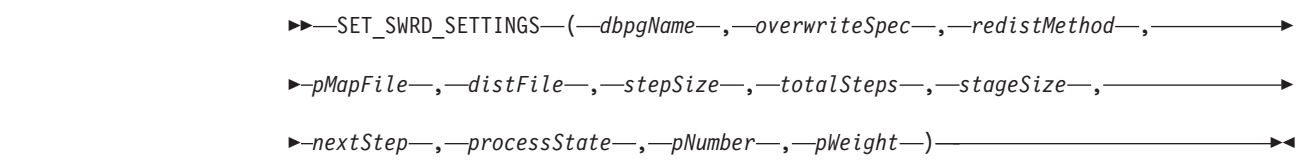
使用上の注意

再配分に関するストアード・プロシージャーおよび関数は、各表の分散キーが定義されているパーティション・データベース環境でのみ動作します。

SET_SWRD_SETTINGS プロシージャー - 再配分レジストリーの作成または変更

SET_SWRD_SETTINGS プロシージャーは、再配分レジストリーを作成または変更します。レジストリーが存在しない場合は、レジストリーを作成してそれにレコードを追加します。レジストリーがすでに存在する場合は、どのフィールド値に上書きする必要があるかを *overwriteSpec* を使用して識別します。 *overwriteSpec* フィールドを指定することにより、更新する必要のないフィールドについては、この関数の入力に NULL を使用できます。

構文



スキーマは SYSPROC です。

プロシージャー・パラメーター

dbpgName
再配分処理の実行の対象となるデータベース・パーティション・グループ名を指定する、タイプ VARCHAR(128) の入力引数。

overwriteSpec
表 233 に示すビット単位フィールド ID。これで、再配分設定レジストリーに書き込むか上書きするターゲット・フィールドを表します。

表 233. ビット単位フィールド ID

フィールド名	16 進値	10 進値
REDIST_METHOD	0x0001<<0	1
REDIST_PMAP_FILE	0x0001<<1	2
REDIST_DIST_FILE	0x0001<<2	4
REDIST_STEP_SIZE	0x0001<<3	8
REDIST_NUM_STEPS	0x0001<<4	16
REDIST_STAGE_SIZE	0x0001<<5	32
REDIST_NEXT_STEP	0x0001<<6	64
REDIST_PROCESS_STATE	0x0001<<7	128
REDIST_PWEIGHT_START_NODE	0x0001<<8	256
REDIST_PWEIGHT	0x0001<<9	512

redistMethod
再配分を実行する際に、データ分散ファイルを使用するか、またはターゲット分散マップを使用するかを指定する、タイプ SMALLINT の入力引数。有効な入力値は、次の 2 つです。

- 2: 再配分処理の入力としてデータ配分ファイルを使用することを表します。

- 3: 再配分処理の入力としてターゲット分散マップを使用することを表します。

pMapFile
データベース・サーバー上のターゲット分散マップの絶対パス・ファイル名を指定する、タイプ VARCHAR (255) の入力引数。

distFile
データベース・サーバー上のデータ配分ファイルの絶対パス・ファイル名を指定する、タイプ VARCHAR (255) の入力引数。

stepSize
ログ・フル状態になるのを防止するためのコミット呼び出しの前に移動可能な最大行数を指定する、タイプ BIGINT の入力引数。この数は、各再配分ステップ内で変わることがあります。 *stepSize* の値に -2 を使用することにより、この数が無制限であることを表すことができます。

totalSteps
このデータベース・パーティション・グループを完全に再配分するために必要なステップ数を指定する、タイプ SMALLINT の入力引数。 *totalSteps* の値に -2 を使用することにより、この数が無制限であることを表すことができます。

stageSize
連続して実行するステップ数を指定する、タイプ SMALLINT の入力引数。

nextStep
どのステップが完了していて、まだ何を実行する必要があるかを区分する索引を指定する、タイプ SMALLINT の入力引数。

processState
再配分処理を次のチェックポイントで停止させるかどうかを指定する、タイプ SMALLINT の入力引数。チェックポイントは、各再配分ステップの先頭に置かれています。この引数が 1 に設定されている場合、ステップは開始されません。値が 0 の場合、ステップは進行します。

pNumber
データベース・パーティション番号がコンマで区切られたストリング・フォーマットのリストを含めることのできる、タイプ VARCHAR (6000) の入力引数。これらのパーティション番号は、データベース・パーティション・グループが現在使用しているデータベース・パーティションか、あるいは追加またはドロップされるデータベース・パーティションです。これらのパーティション番号の順序と数は、*pWeight* 変数によって戻されるターゲット・パーティションの重みに対応します。各データベース・パーティション番号は、0 から 999 までの範囲の数値です。ストリング内でスペースは使用できません。

pWeight
すべてのデータベース・パーティションのユーザー指定の重みがコンマで区切られたストリングを含めることのできるタイプ VARCHAR (6000) の入力引数で、*pNumber* ストリング内のデータベース・パーティション番号に対応します。各データベース・パーティションの重みは、0 から 32767 までの範囲の数値です。ストリング内でスペースは使用できません。

許可

SET_SWRD_SETTINGS プロシージャーに対する EXECUTE 特権。

例

ステップ単位の再配分プランをレジストリーに書き込みます。 *processState* を 1 に設定すると、現在実行中のステップ単位の再配分ストアード・プロシージャーが現行ステップを完了して停止する場合があります。その場合は、このパラメーターを 0 にリセットして再配分ストアード・プロシージャーを再度呼び出すまで停止したままになります。

```
CALL SYSPROC.SET_SWRD_SETTINGS('IBMDEFAULTGROUP', 255, 0, ' ',
   '$HOME/sqllib/function/TEST.IBMDEFAULTGROUP_swrdData.dst', 1000,
   12, 2, 1, 0, '10,20,30', '50,50,50')
```

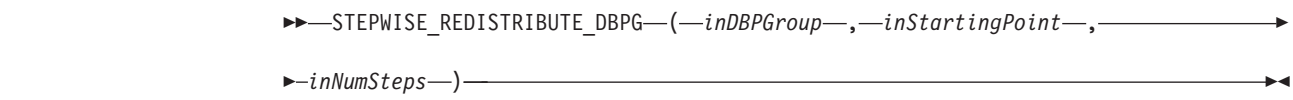
使用上の注意

再配分に関するストアード・プロシージャーおよび関数は、各表の分散キーが定義されているパーティション・データベース環境でのみ動作します。

STEPWISE_REDISTRIBUTE_DBPG プロシージャー - データベース・パーティション・グループの一部の再配分

STEPWISE_REDISTRIBUTE_DBPG プロシージャーは、このプロシージャーに指定された入力と、SET_SWRD_SETTINGS プロシージャーによって作成または更新された設定ファイルに従って、データベース・パーティション・グループの一部を再配分します。

構文



スキーマは SYSPROC です。

プロシージャー・パラメーター

inDBPGroup
ターゲット・データベース・パーティション・グループの名前を指定する、タイプ VARCHAR (128) の入力引数。

inStartingPoint
使用する開始点を指定する、タイプ SMALLINT の入力引数。このパラメーターが NULL 以外の正の整数に設定されると、STEPWISE_REDISTRIBUTE_DBPG プロシージャーは、設定ファイルで指定された *nextStep* 値を使用しないで、この値を使用します。このオプションは、STEPWISE_REDISTRIBUTE_DBPG プロシージャーを特定のステップから再実行するときに役立ちます。このパラメーターを NULL に設定した場合は、*nextStep* 値が使用されます。

inNumSteps
実行するステップ数を指定する、タイプ SMALLINT の入力引数。このパラメ

ーターが NULL 以外の正の整数に設定されると、STEPWISE_REDISTRIBUTE_DBPG プロシージャーは、設定ファイルで指定された *stageSize* 値を使用しないで、この値を使用します。このオプションは、設定で指定された内容とは異なるステップ数を指定して STEPWISE_REDISTRIBUTE_DBPG プロシージャーを再実行するときに役立ちます。例えば、スケジュール済みステージに 5 つのステップがあり、再配分処理がステップ 3 で失敗した場合、エラー状態が訂正されたら STEPWISE_REDISTRIBUTE_DBPG プロシージャーを呼び出すことによって、残りの 3 つのステップを実行することができます。このパラメーターを NULL に設定した場合は、*stageSize* 値が使用されます。このプロシージャーに値 -2 を使用することにより、この数が無制限であることを表すことができます。

注: REDISTRIBUTE DATABASE PARTITION GROUP コマンドに **NOT ROLLFORWARD RECOVERABLE** オプションと同等のものを指定するパラメーターはありません。 STEPWISE_REDISTRIBUTE_DBPG プロシージャーの使用時には、行データ再配布に対してロギングが必ず実行されます。

許可

- STEPWISE_REDISTRIBUTE_DBPG プロシージャーに対する EXECUTE 特権
- SYSADM、SYSCTRL、または DBADM

例

SET_SWRD_SETTINGS プロシージャーによってレジストリーに保管された再配分プランに従って、データベース・パーティション・グループ「IBMDEFAULTGROUP」を再配分します。データの再配分をステップ 3 から開始して、再配分プランの 2 つのステップを完了します。

```
CALL SYSPROC.STEPWISE_REDISTRIBUTE_DBPG('IBMDEFAULTGROUP', 3, 2)
```

使用上の注意

STEPWISE_REDISTRIBUTE_DBPG プロシージャーの実行開始後に SET_SWRD_SETTINGS プロシージャーを使用して *processState* のレジストリー値を 1 に更新すると、処理は次のステップの先頭で停止し、警告メッセージが戻されます。

再配分処理で SQL COMMIT ステートメントが呼び出されるため、タイプ 2 接続での再配分処理の実行はサポートされていません。

第 18 章 ストレージ管理ツール・ルーチン

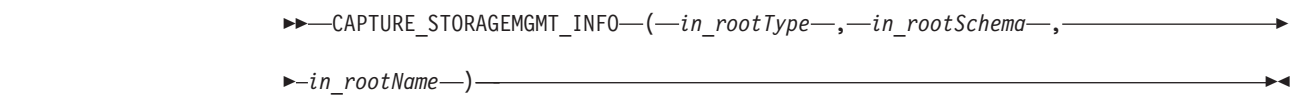
CAPTURE_STORAGEMGMT_INFO プロシージャー - 特定ルート・オブジェクトのストレージ関連情報の検索

CAPTURE_STORAGEMGMT_INFO プロシージャーは、指定されたルート・オブジェクト、およびその有効範囲内の定義済みストレージ・オブジェクトについて、ストレージ関連情報の収集を試みます。すべてのストレージ・オブジェクトは SYSTOOLS.STMG_OBJECT_TYPE 表で特定されています。

表 234. STMG_OBJECT_TYPE 表

列名	データ・タイプ	NULL 可能	説明
OBJ_TYPE	INTEGER	N	ストレージ・オブジェクトのタイプに対応する整数値 • 0 - データベース • 1 - データベース・パーティション・グループ • 2 - 表スペース • 3 - 表スペース・コンテナー • 4 - 表 • 5 - 索引
TYPE_NAME	VARCHAR	N	ストレージ・オブジェクト・タイプの記述名 • STMG_DATABASE • STMG_DBPGROUP • STMG_TABLESPACE • STMG_CONTAINER • STMG_TABLE • STMG_INDEX

構文



スキーマは SYSPROC です。

プロシージャー・パラメーター

in_rootType
タイプ SMALLINT の入力引数。有効なオプション・タイプは、以下のとおりです。

- 0 - データベース

© Copyright IBM Corp. 2006, 2010

- 1 - データベース・パーティション・グループ
- 2 - 表スペース
- 4 - 表
- 5 - 索引

この入力引数を NULL にすることはできません。 NULL 値を指定すると、SQL0443 エラー (SQLSTATE 38553) と、トークン DBA7617 が戻されます。

in_rootSchema
ストレージ・スナップショットのルート・オブジェクトのスキーマ名を指定する、タイプ VARCHAR (128) の入力引数。

in_rootName
ルート・オブジェクトの名前を指定する、タイプ VARCHAR (128) の入力引数。この入力引数を NULL にすることはできません。 NULL 値を指定すると、SQL0443 エラー (SQLSTATE 38553) と、トークン DBA7617 が戻されます。

許可

- CAPTURE_STORAGEMGMT_INFO プロシージャーに対する EXECUTE 特権。
- SYSPROC.DB_PARTITIONS、 SYSPROC.SNAP_GET_CONTAINER、SYSPROC.SNAPSHOT_CNTRFS 表関数に対する EXECUTE 特権。
- SYSCAT.TABLES、 SYSCAT.TABLESPACES、 SYSCAT.NODEGROUPDEF、SYSCAT.DATABASEPARTITIONS、 SYSCAT.DATAPARTITIONEXPRESSION、SYSCAT.INDEXES、 SYSCAT.COLUMNS に対する SELECT 特権。

使用上の注意:

1. 以下のストアード・プロシージャーを使用してストレージ管理表を作成する必要があります。

   create_storagemgmt_tables(TABLESPACE_NAME) (この 'TABLESPACE' は、ストレージ管理表の作成場所となる表スペースの名前です)。

   既存のストレージ管理表で問題が発生した場合、下記のストアード・プロシージャーを使ってこれをドロップした後、上記のストアード・プロシージャーを使って再作成することができます。

   drop_storagemgmt_tables(0 または 1) (ここで '0' はエラー検出時に「停止」、'1 ' は「続行」をそれぞれ示します。)
2. 以下のコマンドを使用して、詳細を取得する対象となるストレージ・オブジェクトに関する統計を実行する必要があります。

   RUNSTATS ON TABLE (TABLESCHEMA.TABLENAME) ON KEY COLUMNS AND INDEXES ALL
3. 以下のコマンドを使用して、ストレージ管理表にデータを入れる必要があります。

   ストレージ管理表にデータを入れるために 'capture_storagemgnt_info()' ストアード・プロシージャーを実行します。場合によっては、CAPTURE_STORAGEMGMT_INFO プロシージャーを 2 度実行する必要が生じ

ることがあります。最初の実行では、ストレージ表に表スペースの詳細を取り込むために CAPTURE_STORAGEMGMT_INFO プロシージャーを使用します。例えば、以下のようにします。

```
db2 "call capture_storagemgmt_info(0,(SCHEMA_NAME),(DATABASE_NAME))"
```

2 度目は、実際のオブジェクトのストレージに関する詳細をストレージ表に追加するために CAPTURE_STORAGEMGMT_INFO プロシージャーを使用します。例えば、以下の例は、索引タイプのオブジェクトの詳細を追加します (*in_rootType* 引数は 5 に設定されています)。

```
db2 "call capture_storagemgmt_info(5,(SCHEMA_NAME),
(SCHEMA_NAME.INDEX_NAME))"
```

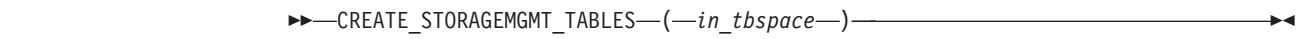
4. 必要なストレージ管理表に対する SELECT 照会を実行して、ストレージ・オブジェクトの詳細を表示します。例えば INDEX オブジェクトの場合には次のようになります。

```
db2 "SELECT * FROM SYSTOOLS.STMG_INDEX"
```

CREATE_STORAGEMGMT_TABLES プロシージャー - ストレージ管理表の作成

CREATE_STORAGEMGMT_TABLES プロシージャーは、入力で指定された表スペースに、「DB2TOOLS」固定スキーマ下のすべてのストレージ管理表を作成します。

構文



スキーマは SYSPROC です。

プロシージャー・パラメーター

in_tbspace
表スペース名を指定する、タイプ VARCHAR(128) の入力引数。この入力引数を NULL にすることはできません。 NULL 値を指定すると、SQL0443 エラー (SQLSTATE 38553) と、トークン DBA7617 が戻されます。

許可

CREATE_STORAGEMGMT_TABLES プロシージャーに対する EXECUTE 特権。

この他に、データベースに対する CREATETAB 特権および表スペースに対する USE 特権と、以下のいずれかが必要です。

- データベースに対する IMPLICIT_SCHEMA 権限 (暗黙または明示のスキーマ名 DB2TOOLS が存在しない場合)
- スキーマに対する CREATEIN 特権 (表のスキーマ名が存在する場合)
- DBADM 権限

使用上の注意

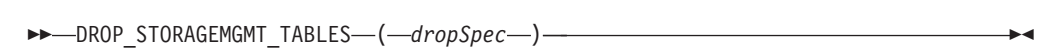
以下の表が DB2TOOLS スキーマで作成されます。

- STMG_CONTAINER
- STMG_CURR_THRESHOLD
- STMG_DATABASE
- STMG_DBPARTITION
- STMG_DBPGROUP
- STMG_HIST_THRESHOLD
- STMG_INDEX
- STMG_OBJECT
- STMG_OBJECT_TYPE
- STMG_ROOT_OBJECT
- STMG_TABLE
- STMG_TABLESPACE
- STMG_TBPARTITION
- STMG_THRESHOLD_REGISTRY

DROP_STORAGEMGMT_TABLES プロシージャー - すべてのストレージ管理表のドロップ

DROP_STORAGEMGMT_TABLES プロシージャーは、すべてのストレージ管理表のドロップを試みます。

構文



スキーマは SYSPROC です。

プロシージャー・パラメーター

dropSpec
タイプ SMALLINT の入力引数。 *dropSpec* を 0 に設定した場合、エラーが検出されると処理を停止します。*dropSpec* を 1 に設定した場合、検出したエラーを無視して処理を続行します。この入力引数を NULL にすることはできません。 NULL 値を指定すると、SQL0443 エラー (SQLSTATE 38553) と、トークン DBA7617 が戻されます。

許可

DROP_STORAGEMGMT_TABLES プロシージャーに対する EXECUTE 特権。

データベース接続を確立するユーザー ID は、ストレージ管理表の定義者 (SYSCAT.TABLES の DEFINER 列に記録されている) であるか、または以下の特権の少なくとも 1 つを持っていなければなりません。

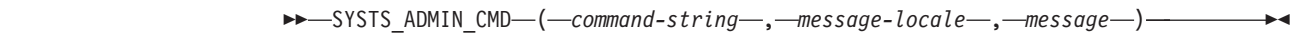
- DBADM 権限
- これらの表のスキーマに対する DROPIN 特権
- これらの表に対する CONTROL 特権

第 19 章 テキスト検索ルーチン

SYSTS_ADMIN_CMD ストアード・プロシージャー - テキスト検索管理コマンドの実行

SYSTS_ADMIN_CMD プロシージャーは、SQL CALL ステートメントを使用してテキスト検索管理コマンドを実行するアプリケーションで使用されます。

構文



スキーマは SYSPROC です。

プロシージャー・パラメーター

command-string
実行される単一のテキスト検索索引管理コマンドを指定する、タイプ VARCHAR (32K) の入力引数。コマンド構文は、接続オプションを除き、DB2 テキスト検索コマンドと同じです。接続オプションは、このプロシージャーではサポートされません。このプロシージャーから実行されるコマンドは、現行接続を使用します。

message-locale
戻されるすべてのエラー・メッセージ・テキスト用に使用される言語を指定する、タイプ VARCHAR(33) の入力引数。引数が NULL または空ストリングであるか、指定したロケールのメッセージ・ファイルがサーバーで使用できない場合には、'en_US' が使用されます。

message
正常と見なされる操作に対する警告または通知メッセージを指定する、タイプ VARCHAR(32K) の出力引数。

許可

SYSTS_ADMIN_CMD プロシージャーに対する EXECUTE 特権。

このプロシージャーは現在、以下の DB2 テキスト検索コマンドをサポートします。

- ALTER INDEX
- CLEAR COMMAND LOCKS
- CLEAR EVENTS
- CREATE INDEX
- DISABLE DATABASE
- DROP INDEX
- ENABLE DATABASE

© Copyright IBM Corp. 2006, 2010

- UPDATE INDEX

例

スキーマ DB2TS のテキスト検索索引 MYTEXTINDEX を更新し、すべてのエラー・メッセージを英語で戻します。

```
CALL SYSPROC.SYSTS_ADMIN_CMD
 ('UPDATE INDEX DB2TS.MYTEXTINDEX FOR TEXT','en_US', ?)";
```

以下はこの照会の出力例です。

```
Value of output parameters
--------------------------
Parameter Name  : MESSAGE
Parameter Value : CIE00001 Operation completed successfully.

Return Status = 0
```

使用上の注意

- コマンドの実行が失敗した場合、SQLCODE -20427 および SQLSTATE 38H14 が、テキスト検索固有のエラー・メッセージとともに戻されます。例えば、索引 MYTEXTINDEX が既に存在しており、以下のステートメントが発行された場合:

  ```
  CALL SYSPROC.SYSTS_ADMIN_CMD ('CREATE INDEX MYTEXTINDEX FOR TEXT
    ON DB2TS.TEXTBOOKS (STORY)', 'en_US', ?)
  ```

  索引作成は失敗し、以下のエラー・メッセージが出されます。

  ```
  SQL20427N An error occurred during a text search administration
  procedure or  command. The error message is "CIE00201 Text search
  index "DB2TS  "."MYTEXTINDEX" already exists. ". SQLSTATE=38H14
  ```

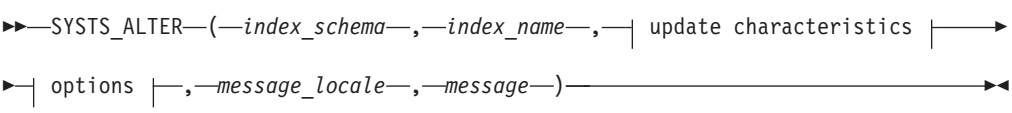
- SQLCODE がプロシージャーにより戻される場合、メッセージは切り捨てられる可能性があります。詳細なメッセージ情報は db2diag ログ・ファイル内にあります。

SYSTS_ALTER プロシージャー - 索引の更新特性の変更

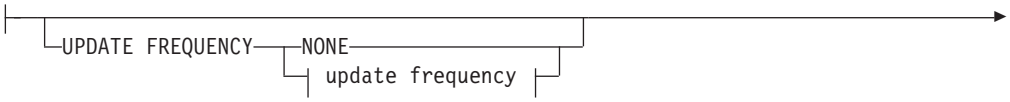
このプロシージャーは、索引の更新特性を変更します。

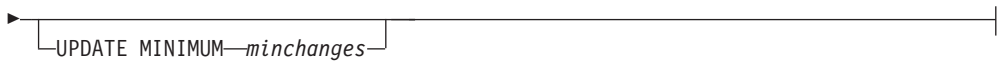
このプロシージャーは ALTER INDEX テキスト検索管理コマンドをデータベース・サーバー上で発行します。

構文

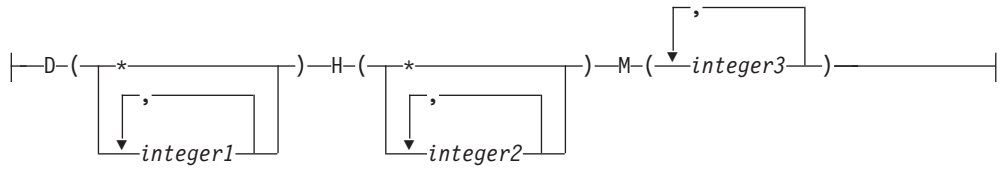


update characteristics:

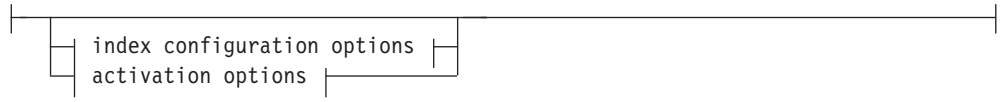




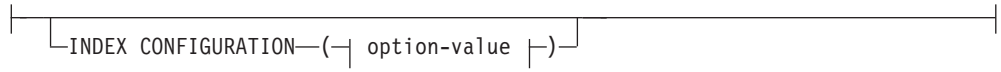
update frequency:



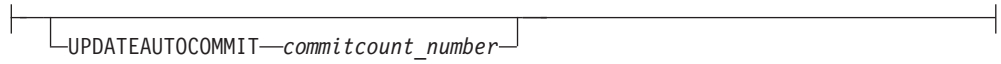
options:



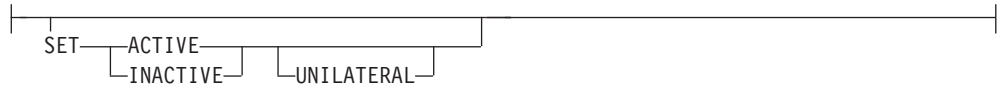
index configuration options:



option-value:



activation options:



スキーマは SYSPROC です。

プロシージャー・パラメーター

index_schema
テキスト検索索引のスキーマを指定する、タイプ VARCHAR(128) の入力引数。 *index_schema* は、DB2 スキーマ名の命名上の制約に従う必要があります。引数が NULL または空ストリングである場合、値 CURRENT SCHEMA が使用されます。 *index_schema* には、大文字小文字の区別があります。

index_name
索引の名前を指定する、タイプ VARCHAR (128) の入力引数。 *index_schema* とともに使用すると、データベースのテキスト検索索引を一意的に識別します。 *index_name* には、大文字小文字の区別があります。

update characteristics
変更オプションを指定する、タイプ VARCHAR(32K) の入力引数。指定可能な変更オプションは以下のとおりです。

UPDATE FREQUENCY
索引の更新が行われる頻度を指定します。変更回数が UPDATE MINIMUM に設定された値以上である場合、索引は更新されます。更新頻度 NONE は、索引の更新はそれ以降行われないことを意味します。これは、変更されないデータを持つ表のテキスト列に役立ちます。さらに、これはユーザーが (UPDATE INDEX コマンドを使用して) 手動で索引を更新する予定の場合にも役立ちます。 START FOR TEXT コマンドが実行されていて、DB2 Text Search インスタンス・サービスが実行中の場合にのみ、自動更新が行われます。

デフォルトの頻度値はビュー SYSIBMTS.TSDEFAULTS から取られ、DEFAULTNAME='UPDATEFREQUENCY' となります。

NONE
自動更新は、テキスト索引に適用されません。以降のすべての索引更新は手動で開始する必要があります。

D 索引が更新される曜日。

* 毎日。

integer1
日曜日から土曜日の特定の曜日: 0 から 6

H 索引が更新される指定の日の時間 (複数指定可)。

* 毎時。

integer2
午前 0 時から午後 11 時までの特定の時: 0 から 23

M 索引が更新される指定の時間の分 (複数指定可)。

integer3
正時 (0) として、または正時の後に 5 分の増分の倍数として指定: 0、5、10、15、20、25、30、35、40、45、50、または 55

UPDATE FREQUENCY オプションを指定しない場合、頻度の設定は変更されないままとなります。

UPDATE MINIMUM *minchanges*
索引が増分的に更新される前に生じなければならない、テキスト文書への変更の最小数を指定します。同じテキスト文書への複数の変更は、別個の変更として扱われます。 UPDATE MINIMUM オプションを指定しない場合、設定は変更されないままとなります。

INDEX CONFIGURATION (*option-value***)**
バージョン 9.7 フィックスパック 3 以降のフィックスパックでは、これは、テキスト索引構成設定を変更できる VARCHAR(32K) タイプのオプションの入力引数です。以下のオプションがサポートされています。

表 235. オプション値の仕様

オプション	値	データ・タイプ	説明
UPDATEAUTOCOMMIT	*commitcount_number*	整数	ここで指定された回数の索引更新が行われると、コミットが実行され、イニシャル更新またはインクリメンタル更新のためにそれまでの作業が自動的に保存されます。 • イニシャル更新では、データ更新をキャプチャーするトリガーがアクティブ化された後の基本表からの文書のバッチを、索引更新が処理します。文書更新の回数が COMMITCOUNT に達すると、サーバーは暫定コミットを実行します。処理されていない文書により生成されたログ項目は、ステージング表から除去されます。イニシャル・テキスト索引更新に UPDATEAUTOCOMMIT オプションを使用すると、実行時間が大幅に増加します。 • インクリメンタル更新では、処理されているログ項目は、各暫定コミットで対応するステージング表から除去されます。COMMITCOUNT は更新される文書の数であり、ステージング表項目の数ではありません。

activation options

バージョン 9.7 フィックスパック 3 以降のフィックスパックでは、この整数タイプの入力引数はテキスト索引の状況を設定します。

ACTIVE
テキスト索引状況をアクティブに設定します

INACTIVE
テキスト索引状況を非アクティブに設定します

UNILATERAL
DB2 Text Search 索引の状況にのみ影響する UNILATERAL 変更を指定します。この引数が指定された場合、DB2 Text Search 索引の状況だけがアクティブまたは非アクティブに変更されます。UNILATERAL 引数がない場合、DB2 Text Search 索引および DB2 Net Search Extender 索引のアクティブ化の状況はともに切り替えられ、テキスト索引の一方だけがアクティブ化されます。

message_locale

戻されるすべてのエラー・メッセージに使用されるロケールを指定する、タイプ VARCHAR(33) の入力引数。引数が NULL または空ストリングであるか、指定したロケールのメッセージ・ファイルがサーバーで使用できない場合には、'en_US' が使用されます。

message
正常に完了した操作の警告または通知メッセージを指定する、タイプ VARCHAR(32K) の出力引数。

許可

ステートメントの許可 ID によって保持されている特権には、少なくとも以下のいずれかが含まれていなければなりません。

- そのテキスト索引の定義されている表に対する CONTROL 特権。
- DBADM 権限

例

例 1: 以下の例では、テキスト検索索引の更新特性が変更されます。この索引は、最初は *index_schema* `'db2ts'` および *index_name* `'myTextIndex'` で作成されています。関連する表列に変更が加えられないことが予想されるため、テキスト検索索引にそれ以上の更新をしないという意図で UPDATE FREQUENCY NONE を使用します。すべてのエラー・メッセージは英語で戻すことが要求されます。このプロシージャーが成功すると、操作の成功を示す出力パラメーター・メッセージが呼び出し元に戻されます。

```
CALL SYSPROC.SYSTS_ALTER('db2ts', 'myTextIndex',
 'UPDATE FREQUENCY NONE', 'en_US', ?)
```

以下はこの照会の出力例です。

```
Value of output parameters
--------------------------
Parameter Name  : MESSAGE
Parameter Value : Operation completed successfully.

Return Status = 0
```

例 2: 以下の例では、SYSTS_ALTER ストアード・プロシージャーが呼び出され、*index_schema* `'db2ts'` および *index_name* `'myTextIndex'` があるテキスト検索索引の更新特性が変更されます。この意図は、索引への更新が毎時行われるということです。ただし、この索引は存在せず、結果としてエラーを戻します。

```
CALL SYSPROC.SYSTS_ALTER('db2ts', 'myTextIndex',
 'update frequency D(*) H(*) M(0)', 'en_US', ?)
```

以下はこの照会の出力例です。

```
SQL20427N An error occurred during a text search administration
procedure or command. The error message is "CIE00316 Text search
index "db2ts"."myTextIndex" does not exist. ". SQLSTATE 38H14
```

使用上の注意

- テキスト検索管理プロシージャーは、データベースへの既存の接続を使用します。現行トランザクションは、プロシージャーの完了結果に応じてコミットされるかまたはロールバックされる場合があります。そのため、そのようなコミットまたはロールバックからの予期しない影響を避けるために、すべてのトランザクション変更をコミットすることができます。これを行う 1 つの方法は、接続の AUTOCOMMIT をオンにすることです。

- 複数のプロシージャーまたはコマンドは、競合する可能性がある場合は、テキスト検索索引に対して同時に実行することはできません。競合するプロシージャーおよびコマンドの一部は以下のとおりです。
  - SYSTS_ALTER プロシージャーまたは ALTER INDEX db2ts コマンド
  - SYSTS_CLEAR_EVENTS プロシージャーまたは CLEAR EVENTS FOR INDEX db2ts コマンド
  - SYSTS_DISABLE プロシージャーまたは DISABLE DATABASE FOR TEXT db2ts コマンド
  - SYSTS_DROP プロシージャーまたは DROP INDEX db2ts コマンド
  - STOP FOR TEXT db2ts コマンド
  - SYSTS_UPDATE プロシージャーまたは UPDATE INDEX db2ts コマンド

  競合がある場合、プロシージャーは SQLCODE -20426 および SQLSTATE 38H13 を戻します。
- このプロシージャーが実行されると、
  - DB2 テキスト検索ビュー SYSIBMTS.TSLOCKS の内容は更新されます。
  - テキスト検索索引データ・ファイル内の索引項目が更新されます。このファイルには、インスタンスの各索引の更新スケジュール (空のものも含む) の永続表現が含まれています。
- 索引をアクティブにした結果は、元の索引の状況により異なります。次の表で結果を説明します。

表 236. 無効索引無しの状況変更:

イニシャルの DB2 Text Search または Net Search Extender の状況	ACTIVE 要求	ACTIVE UNILATERAL 要求	INACTIVE 要求	INACTIVE UNILATERAL 要求
アクティブ / 非アクティブ	変更なし	変更なし	非アクティブ / アクティブ	非アクティブ / 非アクティブ
非アクティブ / アクティブ	アクティブ / 非アクティブ	エラー	変更なし	変更なし
非アクティブ / 非アクティブ	アクティブ / 非アクティブ	アクティブ / 非アクティブ	非アクティブ / アクティブ	変更なし

SQL20427N および CIE0379E エラー・メッセージが、索引のアクティブ化競合の場合に戻されます。

SYSTS_CLEAR_COMMANDLOCKS プロシージャー - テキスト検索索引のコマンド・ロックの削除

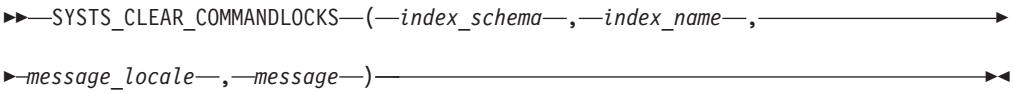
このプロシージャーは、データベース内の特定のテキスト検索索引またはすべてのテキスト検索索引のすべてのコマンド・ロックを解除します。

コマンド・ロックはテキスト検索索引コマンドの開始時に作成され、コマンドの完了時に破棄されます。それにより、異なるコマンド間の望ましくない競合が避けられます。

活動状態でなくなったプロセスと関連するすべてのロックのクリーンアップは、自動的に実行されます。これは、テキスト検索索引が新規の検索要求からアクセス可能になるために行われます。まれなケースとして、予期しないシステム動作によってロックがそのまま残り、明示的にクリーンアップする必要があるときに、このプロシージャーの使用が必要とされます。

このプロシージャーは、データベース・サーバーで CLEAR COMMAND LOCKS テキスト検索管理コマンドを発行します。

構文



スキーマは SYSPROC です。

プロシージャー・パラメーター

index_schema
テキスト索引のスキーマを指定する、タイプ VARCHAR(128) の入力引数。 *index_schema* は、DB2 スキーマ名の命名上の制約に従う必要があります。引数が NULL または空ストリングである場合、値 CURRENT SCHEMA が使用されます。 *index_schema* には、大文字小文字の区別があります。

index_name
索引の名前を指定する、タイプ VARCHAR (128) の入力引数。 *index_schema* とともに使用すると、データベースのテキスト検索索引を一意的に識別します。引数が NULL または空ストリングである場合、このプロシージャーはデータベース内のすべてのテキスト検索索引のコマンド・ロックを削除します。 *index_name* には、大文字小文字の区別があります。

message_locale
戻されるすべてのエラー・メッセージに使用されるロケールを指定する、タイプ VARCHAR(33) の入力引数。引数が NULL または空ストリングであるか、指定したロケールのメッセージ・ファイルがサーバーで使用できない場合には、'en_US' が使用されます。

message
正常に完了した操作の警告または通知メッセージを指定する、タイプ VARCHAR(32K) の出力引数。

許可

索引名が指定されていない場合、データベース接続用の *username* は DBADM 権限を持っている必要があります。特定の索引でコマンド・ロックをクリアする場合、データベース接続用の *username* が、テキスト検索索引が作成された表に対する CONTROL 特権を持っている必要があります。

例

例 1: 以下の例では、SYSTS_CLEAR_COMMANDLOCKS が、*index_schema* `'db2ts'` および *index_name* `'myTextIndex'` があるテキスト検索索引に対して発行

されます。エラー・メッセージは英語で戻すことが要求されます。このプロシージャーが成功すると、操作の成功を示す出力パラメーター・メッセージが呼び出し元に戻されます。

```
CALL SYSPROC.SYSTS_CLEAR_COMMANDLOCKS('db2ts', 'myTextIndex', 'en_US', ?)
```

以下はこの照会の出力例です。

```
Value of output parameters
--------------------------
Parameter Name  : MESSAGE
Parameter Value : Operation completed successfully.

Return Status = 0
```

例 2: 以下の例では、SYSTS_CLEAR_COMMANDLOCKS が呼び出され、*index_schema* 'db2ts' および *index_name* 'myTextIndex' があるテキスト検索索引のコマンド・ロックがクリアされます。この索引は存在せず、プロシージャーはエラー・メッセージを戻します。

```
CALL SYSPROC.SYSTS_CLEAR_COMMANDLOCKS('db2ts', 'myTextIndex', 'en_US', ?)
```

以下はこの照会の出力例です。

```
SQL20427N An error occurred during a text search administration
procedure or command. The error message is "CIE00316 Text search
index "db2ts"."myTextIndex" does not exist. ". SQLSTATE 38H14
```

使用上の注意

- テキスト検索管理プロシージャーは、データベースへの既存の接続を使用します。現行トランザクションは、プロシージャーの完了結果に応じてコミットされるかまたはロールバックされる場合があります。そのため、そのようなコミットまたはロールバックからの予期しない影響を避けるために、すべてのトランザクション変更をコミットすることができます。これを行う 1 つの方法は、AUTOCOMMIT をオンにすることです。
- ビュー SYSIBMTS.TSLOCKS のプロセスおよびスレッド情報は、ロックを保持するスレッドまたはプロセスがまだ存在しているかをチェックするために使用されます。実行中のテキスト検索管理プロシージャーまたはコマンド (例えば、SYSTS_UPDATE または UPDATE INDEX) に属する既存のプロセスに対するロックは、クリアしてはなりません。
- コマンド・ロックを所有するプロセスが非活動であるため、このプロシージャーを呼び出す場合があります。このケースでは、(ロックにより表される) コマンドが完了していない場合や、索引が操作できない場合があります。適切なアクションを取る必要があります。例えば、DROP INDEX コマンドを実行している処理が突然停止したとします。いくつかの索引データは削除されましたが、すべてのカタログおよびコレクション情報が削除された訳ではありません。コマンド・ロックはそのまま残されています。 DROP INDEX コマンド・ロックをクリアした後に、SYSTS_DROP プロシージャーを再実行することができます。別の例として、SYSTS_CREATE プロシージャーを実行するプロセスが突然非活動になったとします。いくつかの索引カタログおよびコレクション情報は作成されましたが、すべてではありません。コマンド・ロックは、そのまま残されています。コマンド・ロックをクリアした後に、SYSTS_DROP および SYSTS_CREATE プロシージャーを実行することができます。

- このプロシージャーを実行すると、DB2 テキスト検索ビュー SYSIBMTS.TSLOCKS の内容は更新されます。

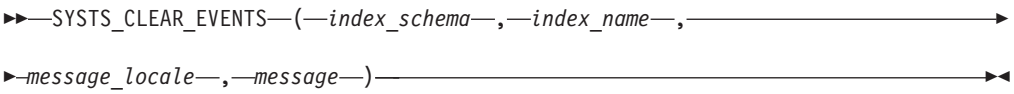
SYSTS_CLEAR_EVENTS プロシージャー - 索引のイベント表からの索引付けイベントの削除

このプロシージャーは、管理に使用される索引のイベント表から、索引付けイベントを削除します。

イベント表の名前は、列 EVENTVIEWNAME のビュー SYSIBMTS.TSINDEXES にあります。少なくとも 1 つの文書を処理する索引の更新操作が行われるごとに、通知項目と、場合によってはエラー項目がイベント表に生成されます。自動更新の場合、イベント表は定期的に検査する必要があります。文書固有のエラーは、文書内容を変更することで訂正する必要があります。エラーを訂正した後、イベントをクリアすることができます (スペースを消費しすぎないために、クリアすべきです)。

このプロシージャーは CLEAR EVENTS FOR INDEX テキスト検索管理コマンドをデータベース・サーバー上で発行します。

構文



スキーマは SYSPROC です。

プロシージャー・パラメーター

index_schema
テキスト検索索引のスキーマを指定する、タイプ VARCHAR(128) の入力引数。 *index_schema* は、DB2 スキーマ名の命名上の制約に従う必要があります。引数が NULL または空ストリングである場合、値 CURRENT SCHEMA が使用されます。 *index_schema* には、大文字小文字の区別があります。

index_name
索引の名前を指定する、タイプ VARCHAR (128) の入力引数。 *index_schema* とともに使用すると、データベースのテキスト検索索引を一意的に識別します。 *index_name* には、大文字小文字の区別があります。

message_locale
戻されるすべてのエラー・メッセージに使用されるロケールを指定する、タイプ VARCHAR(33) の入力引数。引数が NULL または空ストリングであるか、指定したロケールのメッセージ・ファイルがサーバーで使用できない場合には、'en_US' が使用されます。

message
正常に完了した操作の警告または通知メッセージを指定する、タイプ VARCHAR(32K) の出力引数。

許可

ステートメントの許可 ID によって保持されている特権には、少なくとも以下のいずれかが含まれていなければなりません。

- その索引の定義されている表に対する CONTROL 特権。
- DBADM 権限

例

例 1: 以下の例では、*index_schema* 'db2ts' および *index_name* 'myTextIndex' で作成されたテキスト検索索引に対して SYSTS_CLEAR_EVENTS が呼び出されます。すべてのエラー・メッセージは英語で戻すことが要求されます。このプロシージャーが成功すると、操作の成功を示す出力パラメーター・メッセージが呼び出し元に戻されます。

```
CALL SYSPROC.SYSTS_CLEAR_EVENTS('db2ts', 'myTextIndex', 'en_US', ?)
```

以下はこの照会の出力例です。

```
Value of output parameters
--------------------------
Parameter Name  : MESSAGE
Parameter Value : Operation completed successfully.

Return Status = 0
```

例 2: 以下の例では、SYSTS_CLEAR_EVENTS が呼び出され、*index_schema* 'db2ts' および *index_name* 'myTextIndex' があるテキスト検索索引のイベント表項目がクリアされます。この索引は存在せず、結果としてエラーを戻します。

```
CALL SYSPROC.SYSTS_CLEAR_EVENTS('db2ts', 'myTextIndex', 'en_US', ?)
```

以下はこの照会の出力例です。

```
SQL20427N An error occurred during a text search administration
procedure or command. The error message is "CIE00316 Text search
index "db2ts"."myTextIndex" does not exist. ". SQLSTATE 38H14
```

使用上の注意

- テキスト検索管理プロシージャーは、データベースへの既存の接続を使用します。現行トランザクションは、プロシージャーの完了結果に応じてコミットされるかまたはロールバックされる場合があります。そのため、そのようなコミットまたはロールバックからの予期しない影響を避けるために、すべてのトランザクション変更をコミットすることができます。これを行う 1 つの方法は、AUTOCOMMIT をオンにすることです。
- 複数のプロシージャーまたはコマンドは、競合する可能性がある場合は、テキスト検索索引に対して同時に実行することはできません。競合するプロシージャーおよびコマンドの一部は以下のとおりです。
  - SYSTS_ALTER プロシージャーまたは ALTER INDEX db2ts コマンド
  - SYSTS_DISABLE プロシージャーまたは DISABLE DATABASE FOR TEXT db2ts コマンド
  - SYSTS_DROP プロシージャーまたは DROP INDEX db2ts コマンド
  - STOP FOR TEXT db2ts コマンド
  - SYSTS_UPDATE プロシージャーまたは UPDATE INDEX db2ts コマンド

競合がある場合、プロシージャーは SQLCODE -20426 および SQLSTATE 38H13 を戻します。

- 定期的な更新がスケジュールされている場合 (SYSTS_CREATE または SYSTS_ALTER プロシージャーの UPDATE FREQUENCY オプションを参照)、イベント表を定期的に検査する必要があります。
- テキスト検索索引の DB2 テキスト検索イベントをクリーンアップするには、イベントの理由を検査し、エラーの原因を除去した後に、SYSTS_CLEAR_EVENTS プロシージャーまたは CLEAR EVENTS FOR INDEX db2ts コマンドを使用します。
- イベント表で参照されるすべての行に変更が加えられていることを確認します。ユーザー表内の行を変更することで、SYSTS_UPDATE プロシージャーまたは UPDATE INDEX db2ts コマンドを再度実行するときに、エラーがある文書に索引付けが再試行されます。
- このコマンドの発行時に、イベント表はクリアされます。

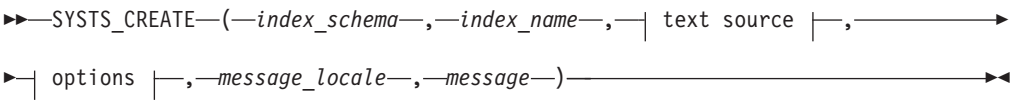
SYSTS_CREATE プロシージャー - 列のテキスト検索索引の作成

このプロシージャーは、テキスト検索関数を使用して列データを検索可能にする、テキスト列のテキスト検索索引を作成します。

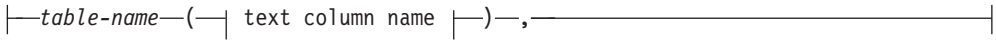
テキスト検索索引を作成すると、照会でテキスト検索関数を使用して列を検索できます。テキスト検索 UPDATE INDEX コマンドまたは SYSTS_UPDATE プロシージャーがユーザーにより明示的に実行されるまで、または索引に定義された更新頻度に従ってテキスト検索インスタンス・レベル・サービスにより暗黙に実行されるまで、索引にデータが入ることはありません。

このプロシージャーは CREATE INDEX テキスト検索管理コマンドをデータベース・サーバー上で発行します。

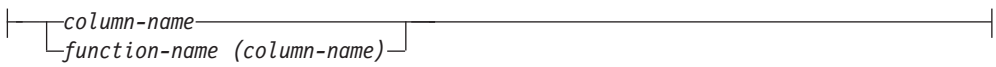
構文



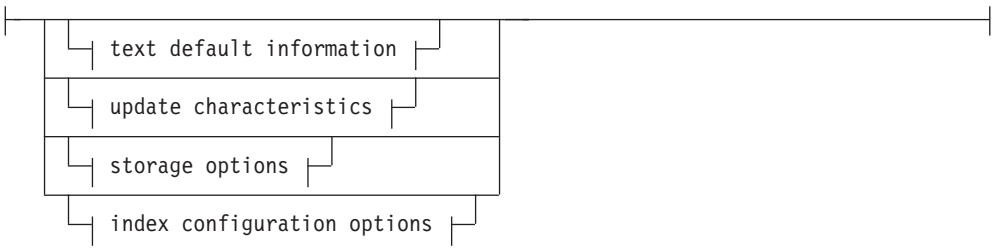
text source:



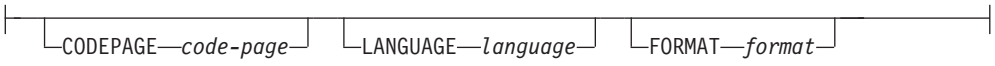
text column name:



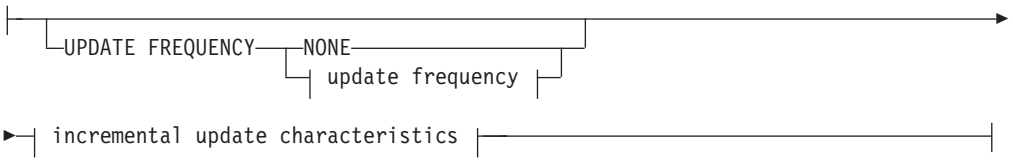
options:



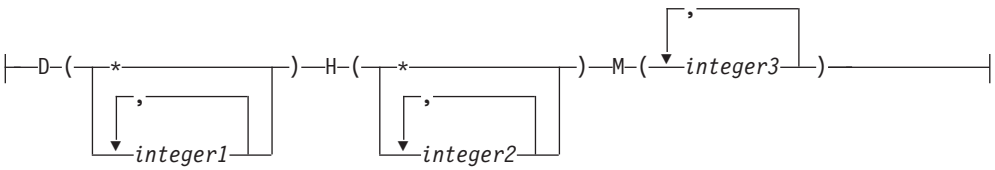
text default information:



update characteristics:

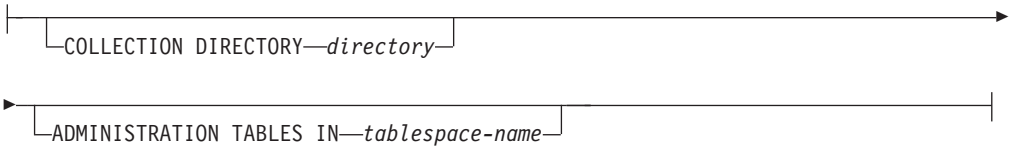


update frequency:

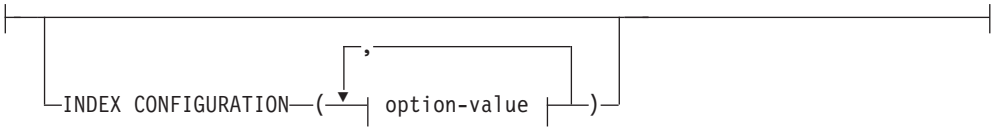


incremental update characteristics:

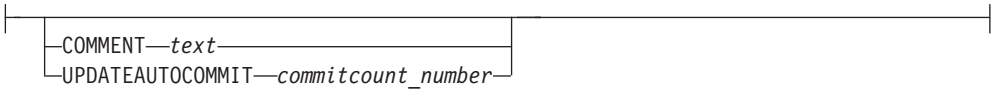
storage options:



index configuration options:



option-value:



スキーマは SYSPROC です。

プロシージャー・パラメーター

index_schema
テキスト検索索引のスキーマを指定する、タイプ VARCHAR(128) の入力引数。 *index_schema* は、DB2 スキーマ名の命名上の制約に従う必要があります。引数が NULL または空ストリングである場合、値 CURRENT SCHEMA が使用されます。 *index_schema* には、大文字小文字の区別があります。

index_name
索引の名前を指定する、タイプ VARCHAR (128) の入力引数。 *index_schema* とともに使用すると、データベースのテキスト検索索引を一意的に識別します。 *index_name* には、大文字小文字の区別があります。

text source
索引付けされた列の名前を指定する、タイプ VARCHAR (1024) の入力引数。オプションは以下のとおりです。

table-name
テキスト列を含む表名。テキスト検索索引を、以下の表に作成することはできません。

- 範囲パーティション表
- フェデレーテッド表
- マテリアライズ照会表
- ビュー

table-name には、大文字小文字の区別があります。

text column name
索引を付ける列の列名。

column-name
列は、CHAR、 VARCHAR、 LONG VARCHAR、 CLOB、 DBCLOB、 BLOB、 GRAPHIC、 VARGRAPHIC、 LONG VARGRAPHIC、または XML のデータ・タイプのいずれかでなければなりません。列のデータ・タイプがこれらのいずれでもない場合、 *function-schema.function-name* で指定された変換関数を使用して、列タイプを有効なタイプのいずれかに変換します。構文および詳細については、*function-name (column-name)* を参照してください。あるいは、索引付けされるテキスト文書にアクセスするユーザー定義の外部関数を指定することができます。1 つのテキスト検索索引のみが、列に作成されます。 *column-name* には、大文字小文字の区別があります。

function-name (column-name)
テキスト検索でサポートされないタイプの列のテキスト文書にアクセスする外部スカラー関数のスキーマ修飾名を指定します (DB2 命名規則に

従います)。その値のデータ・タイプ変換を実行し、テキスト検索のサポートされるデータ・タイプの 1 つで値を戻します。このタスクは、列タイプの変換を実行します。この関数は、1 つのパラメーターのみを取り、1 つの値のみを戻す必要があります。*function-name (column-name)* には、大文字小文字の区別があります。

options
使用するオプションを指定する、タイプ VARCHAR(32K) の入力引数。オプションを必要としない場合は、引数を NULL にするか、または空ストリングにすることができます。使用可能なオプションは以下のとおりです。

CODEPAGE *code-page*
テキスト文書を索引付けするときに使用される DB2 コード・ページ (CODEPAGE) を指定します。デフォルト値はビュー SYSIBMTS.TSDEFAULTS にある値によって指定されます。ここで、DEFAULTNAME='CODEPAGE' (ここではデータベースのコード・ページ) です。この引数はバイナリー・データ・タイプにのみ適用されます。つまり列タイプまたは変換関数からの戻りの型は、BLOB または文字タイプの FOR BIT DATA でなければなりません。

LANGUAGE *language*
索引付け中に、文書の言語固有の処理のために DB2 Text Search によって使用される言語を指定します。ロケールを指定しない場合、データベース・テリトリーを使用して LANGUAGE のデフォルト設定が判別されます。ロケールを判別するために文書を自動的にスキャンさせる場合は、ロケールを AUTO と指定してください。

FORMAT *format*
列内のテキスト文書のフォーマットを指定します。サポートされているフォーマットには TEXT、XML、および HTML が含まれます。文書を索引付けするときに、DB2 Text Search はこの情報を必要とします。フォーマットを指定しない場合、デフォルト値が使用されます。デフォルト値はビュー SYSIBMTS.TSDEFAULTS にあります。ここで、DEFAULTNAME='FORMAT' です。データ・タイプ XML の列の場合は、DEFAULTNAME の値に関係なく、デフォルトのフォーマット 'XML' が使用されます。

UPDATE FREQUENCY
索引の更新が行われる頻度を指定します。変更の数が少なくとも UPDATE MINIMUM に設定された値である場合に、索引が更新されます。更新頻度 NONE は、索引の更新はそれ以降行われないことを意味します。これは、変更されないデータを持つ表のテキスト列に役立ちます。さらに、これはユーザーが (UPDATE INDEX コマンドを使用して) 手動で索引を更新する予定の場合にも役立ちます。 START FOR TEXT コマンドが実行されていて、DB2 Text Search インスタンス・サービスが実行中の場合にのみ、自動更新が行われます。

デフォルトの頻度値はビュー SYSIBMTS.TSDEFAULTS から取られ、DEFAULTNAME='UPDATEFREQUENCY' となります。

NONE
これ以上の索引更新は行われません。更新を手動で開始する必要があります。

D 索引が更新される曜日。

* 毎日。

integer1
日曜日から土曜日の特定の曜日: 0 から 6

H 索引が更新される指定の日の時間 (複数指定可)。

* 毎時。

integer2
午前 0 時から午後 11 時までの特定の時: 0 から 23

M 索引が更新される指定の時間の分 (複数指定可)。

integer3
正時 (0) として、または正時の後に 5 分の増分の倍数として指定: 0、5、10、15、20、25、30、35、40、45、50、または 55

UPDATE MINIMUM *minchanges*
UPDATE FREQUENCY で指定された時間に索引が増分的に更新される前に、テキスト文書に加えられる変更の最小数を指定します。正整数値だけを指定できます。デフォルト値はビュー SYSIBMTS.TSDEFAULTS から取られます。ここで、DEFAULTNAME='UPDATEMINIMUM' です。

注: この値は、UPDATE INDEX コマンドの実行中は無視されます (USING UPDATE MINIMUM オプションがそこで使用されていない場合)。小さい値は、表列とテキスト検索索引の間の整合性を増します。しかし、それはパフォーマンス・オーバーヘッドがより大きくなる原因ともなります。

COLLECTION DIRECTORY *directory*
テキスト検索索引が保管されるディレクトリー。デフォルトでは、コレクション・データは `DBPATH/db2collections` に置かれます。ここで、`DBPATH` の値は、データベースを作成するのに使用したファイル・パスです。絶対パスを指定する必要があります。絶対パス名の最大長は 215 文字です。さまざまな索引が `COLLECTION DIRECTORY` の下の *index identifier* という名前のサブディレクトリーに編成されます。ここで、index identifier はシステムが生成する識別子です。

ADMINISTRATION TABLES IN *tablespace-name*
索引用に作成された管理表の既存の REGULAR 表スペースの名前を指定します。指定しない場合、索引が作成されている基本表の表スペースが使用されます。

INDEX CONFIGURATION (*option-value*)
追加の索引関連値をオプション値のストリングのペアとして指定します。以下の値がサポートされています。

表 237. オプション値の仕様

オプション	値	データ・タイプ	説明
COMMENT	*text*	512 バイトより少ないストリング値	DB2 Text Search カタログ・ビュー `TSINDEXES` にある `REMARKS` 列にストリング・コメント値を追加します。ストリング・コメント値をコレクションの説明としても追加します。

表 237. オプション値の仕様 (続き)

オプション	値	データ・タイプ	説明
UPDATEAUTOCOMMIT	*commitcount_number*	整数	バージョン 9.7 フィックスパック 3 以降のフィックスパックでは、ここで指定された回数の索引更新が行われると、コミットが実行され、イニシャル更新またはインクリメンタル更新のためにそれまでの作業が自動的に保存されます。 • イニシャル更新では、データ更新をキャプチャーするトリガーがアクティブ化された後の基本表からの文書のバッチを、索引更新が処理します。文書更新の回数が COMMITCOUNT に達すると、サーバーは暫定コミットを実行します。処理されていない文書により生成されたログ項目は、ステージング表から除去されます。イニシャル・テキスト索引更新に UPDATEAUTOCOMMIT オプションを使用すると、実行時間が大幅に増加します。 • インクリメンタル更新では、処理されているログ項目は、各暫定コミットで対応するステージング表から除去されます。COMMITCOUNT は更新される文書の数であり、ステージング表項目の数ではありません。

要確認: 非数値は単一引用符で囲む必要があります。ストリング値内の単一引用符の文字は、2 つの連続した単一引用符によって表記される必要があります。

例: `INDEX CONFIGURATION (COMMENT 'Index on User''s Guide column')`

message_locale
戻されるすべてのエラー・メッセージに使用されるロケールを指定する、タイプ VARCHAR(33) の入力引数。引数が NULL または空ストリングであるか、指定したロケールのメッセージ・ファイルがサーバーで使用できない場合には、'en_US' が使用されます。

message
正常に完了した操作の警告または通知メッセージを指定する、タイプ VARCHAR(32K) の出力引数。

許可

ステートメントの許可 ID によって保持されている特権には、少なくとも以下のいずれかが含まれていなければなりません。

以下のいずれか

- その索引の定義されている表に対する CONTROL 特権。

- その索引の定義されている表に対する INDEX 特権。

  および以下のいずれか

  – データベースに対する IMPLICIT_SCHEMA 権限 (索引の暗黙または明示のスキーマ名が存在しない場合)
  – スキーマに対する CREATEIN 特権 (索引のスキーマ名が既存のスキーマを指している場合)
- DBADM 権限

例

例 1: 以下の例では、*index_schema* 'db2ts' および *index_name* 'myTextIndex' があるテキスト検索索引が、SYSTS_CREATE プロシージャーを使用して作成されます。オプション 'UPDATE MINIMUM 10' は、索引に関連付けられたテキスト文書に少なくとも 10 の変更が加えられてから、索引のインクリメンタル更新が実行されることを指定します。すべてのエラー・メッセージは英語で戻すことが要求されます。基本となるテキスト検索コマンドが正常に実行された場合は、出力パラメーター message がコマンド実行の状況を示すように設定されます。

```
CALL SYSPROC.SYSTS_CREATE('db2ts', 'myTextIndex',
 'myUserSchema.myBaseTable (myTextColumn)', 'UPDATE MINIMUM 10',
 'en_US', ?)
```

以下はこの照会の出力例です。

```
Value of output parameters
--------------------------
Parameter Name  : MESSAGE
Parameter Value : Operation completed successfully.
Return Status = 0
```

例 2: 以下の例では、SYSTS_CREATE が呼び出され、*index_schema* 'db2ts' および *index_name* 'myTextIndex' があるテキスト検索索引が作成されます。オプションが指定されていません。この例では、索引が既に存在しており、結果としてエラー・メッセージが呼び出し元に戻されます。

```
CALL SYSPROC.SYSTS_CREATE('db2ts', 'myTextIndex',
 'myUserSchema.myBaseTable (myTextColumn)', '', 'en_US', ?)
```

以下はこの照会の出力例です。

```
SQL20427N An error occurred during a text search administration
procedure or command. The error message is "CIE00201 Text search
index "db2ts"."myTextIndex" already exists. ".
```

使用上の注意

- テキスト検索管理プロシージャーは、データベースへの既存の接続を使用します。現行トランザクションは、プロシージャーの完了結果に応じてコミットされるかまたはロールバックされる場合があります。そのため、そのようなコミットまたはロールバックからの予期しない影響を避けるために、すべてのトランザクション変更をコミットすることができます。これを行う 1 つの方法は、接続の AUTOCOMMIT をオンにすることです。
- 複数のプロシージャーまたはコマンドは、競合する可能性がある場合は、テキスト検索索引に対して同時に実行することはできません。競合するプロシージャーおよびコマンドの一部は以下のとおりです。

- SYSTS_ALTER プロシージャーまたは ALTER INDEX db2ts コマンド
- SYSTS_CLEAR_EVENTS プロシージャーまたは CLEAR EVENTS FOR INDEX db2ts コマンド
- SYSTS_DISABLE プロシージャーまたは DISABLE DATABASE FOR TEXT db2ts コマンド
- STOP FOR TEXT db2ts コマンド
- SYSTS_UPDATE プロシージャーまたは UPDATE INDEX db2ts コマンド

競合がある場合、プロシージャーは SQLCODE -20426 および SQLSTATE 38H13 を戻します。

- CREATE INDEX コマンドが正常に実行されると、以下のようになります。
  - DB2 Text Search のサーバー・データが更新されます。以下の例のように、*instance_database-name_index-identifier_number* という名前のコレクションが作成されます。

    ```
    tigertail_MYTSDB_TS250517_0000
    ```

    コレクション名は SYSIBMTS.TSCOLLECTIONNAMES ビュー (列 COLLECTIONNAME) から取られます。
  - DB2 Text Search のカタログ情報が更新されます。索引のステージング表が、適切な DB2 索引のある指定された表スペースに作成されます。さらに、索引のイベント表が指定された表スペースに作成されます。
  - DB2 Text Search と DB2 Net Search Extender が共に存在し、テーブル列のアクティブ Net Search Extender 索引が既に存在している場合、新規テキスト索引は非アクティブに設定されます。
  - 新規に作成されたテキスト検索索引には、自動的にはデータが追加されません。テキスト検索索引にデータを追加するには、SYSTS_UPDATE プロシージャーまたは UPDATE INDEX コマンドを手動で、または (UPDATE FREQUENCY オプションの指定により索引に対して定義されている更新スケジュールの結果として) 自動で実行する必要があります。
  - DB2 データベース・サーバー上の Text Search の索引データ・ファイルが更新されます。スケジュールされた更新情報は、インスタンス内の各索引ごとに記録されます。

使用上の制限:

- 主キーが表に対して定義されている必要があります。DB2 Text Search では、複数列の DB2 主キーをタイプの制限なしで使用できます。主キー列の数は、DB2 によって許可されている主キー列の数より 2 列少ない数に制限されています。
- DB2 Text Search 索引のある表のすべての主キー列の合計長は、DB2 によって許可されている主キー列の最大合計長より 15 バイト少ない長さに制限されています。 DB2 CREATE INDEX ステートメントの DB2 制約事項を参照してください。

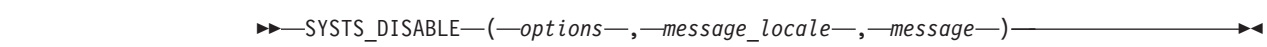
SYSTS_DISABLE プロシージャー - テキスト検索の現行データベースを使用不可にする

このプロシージャーは、現行データベースの DB2 テキスト検索を使用不可にします。

テキスト検索機能を使用不可にすると、テキスト検索索引およびコマンドは、データベースで使用できなくなります。

このプロシージャーは DISABLE DATABASE FOR TEXT テキスト検索管理コマンドをデータベース・サーバー上で発行します。

構文



スキーマは SYSPROC です。

プロシージャー・パラメーター

options
データベースが使用不可の場合に使用するオプションを指定する、タイプ VARCHAR(128) の入力引数。引数は FORCE に設定できます。この値を指定した場合、すべての索引はドロップされ、テキスト検索機能は強制的に使用不可にされます。テキスト検索索引は保存されず、エラー・メッセージまたは警告は戻されません。引数が NULL または空ストリングである場合、データベースのテキスト検索機能を使用不可にすることが試行されます。

message_locale
戻されるすべてのエラー・メッセージに使用されるロケールを指定する、タイプ VARCHAR(33) の入力引数。引数が NULL または空ストリングであるか、指定したロケールのメッセージ・ファイルがサーバーで使用できない場合には、'en_US' が使用されます。

message
正常に完了した操作の警告または通知メッセージを指定する、タイプ VARCHAR(32K) の出力引数。

許可

このステートメントの許可 ID が持つ特権には、DBADM 権限が含まれている必要があります。

例

例 1: 以下の例では、テキスト検索は、SYSTS_DISABLE プロシージャーを使用してデータベースに対して使用不可にされます。 FORCE オプションを指定すると、テキスト検索索引が引き続きデータベース内の表に存在しているとしても、この機能を確実に使用不可にできます。エラー・メッセージは英語で戻すことが要求されます。 *message* 出力パラメーターは、情報メッセージ・ストリングに設定されます。

```
CALL SYSPROC.SYSTS_DISABLE('FORCE', 'en_US', ?)
```

以下はこの照会の出力例です。

```
Value of output parameters
--------------------------
Parameter Name  : MESSAGE
Parameter Value : Operation completed successfully.

Return Status = 0
```

例 2: 以下の例では、FORCE オプションを指定せずに SYSTS_DISABLE プロシージャーを使用して、テキスト検索を既存のテキスト検索索引があるデータベースに対して使用不可にします。これは結果としてエラー・メッセージを呼び出し元に戻します。テキスト検索機能を使用不可にするか、またはその代わりに *options* 入力パラメーター値に FORCE オプションを指定する前に、すべての既存のテキスト検索索引をドロップすることをお勧めします。

```
CALL SYSPROC.SYSTS_DISABLE('', 'en_US', ?)
```

以下はこの照会の出力例です。

```
SQL20427N An error occurred during a text search administration
procedure or command. The error message is "CIE00326 Text search
index active in specified or default database. ". SQLSTATE 38H14
```

使用上の注意

- テキスト検索管理プロシージャーは、データベースへの既存の接続を使用します。現行トランザクションは、プロシージャーの完了結果に応じてコミットされるかまたはロールバックされる場合があります。そのため、そのようなコミットまたはロールバックからの予期しない影響を避けるために、すべてのトランザクション変更をコミットすることができます。これを行う 1 つの方法は、AUTOCOMMIT をオンにすることです。
- 複数のプロシージャーまたはコマンドは、競合する可能性がある場合は、テキスト検索索引に対して同時に実行することはできません。競合するプロシージャーおよびコマンドの一部は以下のとおりです。
  - SYSTS_ALTER プロシージャーまたは ALTER INDEX db2ts コマンド
  - SYSTS_CLEAR_EVENTS プロシージャーまたは CLEAR EVENTS FOR INDEX db2ts コマンド
  - SYSTS_DISABLE プロシージャーまたは DISABLE DATABASE FOR TEXT db2ts コマンド
  - STOP FOR TEXT db2ts コマンド
  - SYSTS_UPDATE プロシージャーまたは UPDATE INDEX db2ts コマンド

  競合がある場合、プロシージャーは SQLCODE -20426 および SQLSTATE 38H13 を戻します。
- このプロシージャーが実行されると、
  - DB2 Text Search のカタログ情報が更新されます。索引ログおよびイベント表がドロップされます。ユーザー・テキスト表のトリガーが削除されます。
  - FORCE オプションを指定した場合、すべてのテキスト索引情報はデータベースから削除され、すべての関連する集合も削除されます。さらに、テキスト・サービスが更新され、残りの更新スケジュール情報がすべて除去されます。詳しくは、『db2ts DROP INDEX コマンド』または『SYSTS_DROP プロシージャー』を参照してください。

- このプロシージャーは、データベースの DB2 Net Search Extender 使用可能化状況には影響を与えません。これは、SYSTS_ENABLE プロシージャーまたは ENABLE FOR TEXT コマンドにより作成された DB2 テキスト検索カタログ表およびビューを削除します。
- テキスト検索索引定義がある DB2 データベースをドロップする前に、このプロシージャーを実行して、テキスト索引および集合が正常に削除されていることを確認してください。
- FORCE オプションを使用して、いくつかの索引が削除できなかった場合、コレクション名が db2diag ログ・ファイルに書き込まれます。テキスト検索索引プロシージャー SYSTS_DISABLE またはコマンド DISABLE DATABASE FOR TEXT が、CLP コマンド DROP DATABASE より先に実行されない場合、テキスト検索索引サービスには CLEANUP FOR TEXT コマンドを使用したクリーンアップも必要になります。索引のドロップについて詳しくは、SYSTS_DROP プロシージャーまたは DROP INDEX コマンドを参照してください。テキスト検索集合およびそのテキスト検索索引に対する関係については、CLEANUP FOR TEXT コマンドを参照してください。

  注: 結果がオーファン集合 (つまり残された集合がテキスト検索サーバー上に定義された状態であるが、DB2 には使用されない) となる使用法は推奨されていません。以下は、孤立したコレクションの原因となる可能性のあるいくつかのケースです。
  - SYSTS_DISABLE プロシージャーまたは DISABLE DATABASE FOR TEXT コマンドを実行せずに、DROP DATABASE CLP コマンドまたは DROP TABLE ステートメントが実行される場合。
  - FORCE オプションを使用して、SYSTS_DISABLE プロシージャーまたは DISABLE DATABASE FOR TEXT コマンドが実行される場合。
  - その他のいくつかのエラー条件。CLEANUP FOR TEXT コマンドは、いくつかのシナリオで使用することができます。

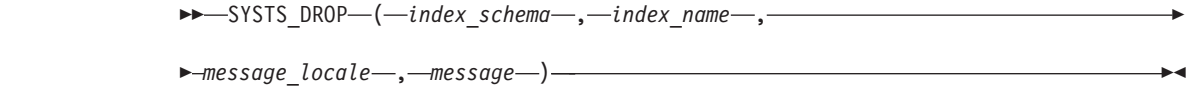
SYSTS_DROP プロシージャー - テキスト検索索引のドロップ

このプロシージャーは、表列と関連付けられた既存のテキスト検索索引をドロップします。

このプロシージャーを正常に実行した後は、テキスト検索照会をその列に対して実行することはできません。

このプロシージャーは DROP INDEX テキスト検索管理コマンドをデータベース・サーバー上で発行します。

構文



スキーマは SYSPROC です。

プロシージャー・パラメーター

index_schema
テキスト検索索引のスキーマを指定する、タイプ VARCHAR(128) の入力引数。 *index_schema* は、DB2 スキーマ名の命名上の制約に従う必要があります。引数が NULL または空ストリングである場合、値 CURRENT SCHEMA が使用されます。 *index_schema* には、大文字小文字の区別があります。

index_name
索引の名前を指定する、タイプ VARCHAR (128) の入力引数。 *index_schema* とともに使用すると、データベースのテキスト検索索引を一意的に識別します。 *index_name* には、大文字小文字の区別があります。

message_locale
戻されるすべてのエラー・メッセージに使用されるロケールを指定する、タイプ VARCHAR(33) の入力引数。引数が NULL または空ストリングであるか、指定したロケールのメッセージ・ファイルがサーバーで使用できない場合には、'en_US' が使用されます。

message
正常に完了した操作の警告または通知メッセージを指定する、タイプ VARCHAR(32K) の出力引数。

許可

ステートメントの許可 ID によって保持されている特権には、少なくとも以下のいずれかが含まれていなければなりません。

- その索引の定義されている表に対する CONTROL 特権。
- DBADM 権限

例

例 1: 以下の例では、*index_schema* `'db2ts'` および *index_name* `'myTextIndex'` で作成されたテキスト検索索引がドロップされます。すべてのエラー・メッセージは英語で戻すことが要求されます。このプロシージャーが成功すると、操作の成功を示す出力パラメーター・メッセージが呼び出し元に戻されます。

```
CALL SYSPROC.SYSTS_DROP('db2ts', 'myTextIndex', 'en_US', ?)
```

以下はこの照会の出力例です。

```
Value of output parameters
--------------------------
Parameter Name  : MESSAGE
Parameter Value : Operation completed successfully.

Return Status = 0
```

例 2: 以下の例では、SYSTS_DROP が呼び出され、*index_schema* `'db2ts'` および *index_name* `'myTextIndex'` があるテキスト検索索引がドロップされます。この索引は存在せず、結果としてエラーを戻します。

```
CALL SYSPROC.SYSTS_DROP('db2ts', 'myTextIndex', 'en_US', ?)
```

以下はこの照会の出力例です。

```
SQL20427N An error occurred during a text search administration
procedure or command. The error message is "CIE00316 Text search
index "db2ts"."myTextIndex" does not exist. ". SQLSTATE 38H14
```

使用上の注意

- テキスト検索管理プロシージャーは、データベースへの既存の接続を使用します。現行トランザクションは、プロシージャーの完了結果に応じてコミットされるかまたはロールバックされる場合があります。そのため、そのようなコミットまたはロールバックからの予期しない影響を避けるために、すべてのトランザクション変更をコミットすることができます。これを行う 1 つの方法は、AUTOCOMMIT をオンにすることです。
- 複数のプロシージャーまたはコマンドは、競合する可能性がある場合は、テキスト検索索引に対して同時に実行することはできません。競合するプロシージャーおよびコマンドの一部は以下のとおりです。
  - SYSTS_ALTER プロシージャーまたは ALTER INDEX db2ts コマンド
  - SYSTS_CLEAR_EVENTS プロシージャーまたは CLEAR EVENTS FOR INDEX db2ts コマンド
  - SYSTS_DISABLE プロシージャーまたは DISABLE DATABASE FOR TEXT db2ts コマンド
  - STOP FOR TEXT db2ts コマンド
  - SYSTS_UPDATE プロシージャーまたは UPDATE INDEX db2ts コマンド

  競合がある場合、プロシージャーは SQLCODE -20426 および SQLSTATE 38H13 を戻します。
- DB2 でユーザー表をドロップしても、索引のドロップは行われません。索引は表をドロップする前後に手動でドロップする必要があります。
- このプロシージャーが実行されると、
  - テキスト検索のカタログ情報が更新されます。索引ステージング表およびイベント表がドロップされます。ユーザー表のトリガーが削除されます。
  - テキスト検索索引データ・ファイル内の索引項目が削除されます。このファイルには、インスタンスの各索引の更新スケジュール (空のものも含む) の永続表現が含まれています。
  - テキスト検索索引定義と関連付けられた集合は削除されます。
- テキスト検索索引をドロップした後に、同じテキスト列に新しいものを作成する予定の場合、新しいテキスト検索索引を作成する前に、まずデータベースから切断し、次いで再接続する必要があります。

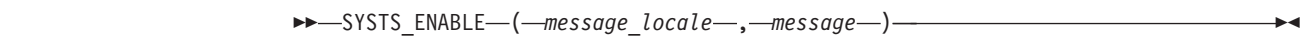
SYSTS_ENABLE プロシージャー - テキスト検索の現行データベースを使用可能にする

このプロシージャーは、現行データベースの DB2 テキスト検索を使用可能にします。

このプロシージャーは、データベース内の表の列に対してテキスト検索索引を作成する前に、正常に実行する必要があります。

このプロシージャーは ENABLE DATABASE FOR TEXT テキスト検索管理コマンドをデータベース・サーバー上で発行します。

構文



スキーマは SYSPROC です。

プロシージャー・パラメーター

message_locale
戻されるすべてのエラー・メッセージに使用されるロケールを指定する、タイプ VARCHAR(33) の入力引数。引数が NULL または空ストリングであるか、指定したロケールのメッセージ・ファイルがサーバーで使用できない場合には、'en_US' が使用されます。

message
正常に完了した操作の警告または通知メッセージを指定する、タイプ VARCHAR(32K) の出力引数。

許可

ENABLE DATABASE コマンドを実行するためには、ユーザーが DBADM 特権を持っていなければなりません。

例

例 1: データベースをテキスト検索に使用できるようにし、すべてのエラー・メッセージを英語で戻します。

```
CALL SYSPROC.SYSTS_ENABLE('en_US', ?)
```

以下はこの照会の出力例です。

```
Value of output parameters
--------------------------
Parameter Name  : MESSAGE
Parameter Value : Operation completed successfully.

Return Status = 0
```

例 2: 以下の例では、SYSTS_ENABLE がテキスト検索用に既に使用可能にされているデータベース上で呼び出されます。これは結果としてエラー・メッセージを呼び出し元に戻します。

```
CALL SYSPROC.SYSTS_ENABLE('en_US', ?)
```

以下はこの照会の出力例です。

```
SQL20427N An error occurred during a text search administration
procedure or command. The error message from the text search
product is "CIE00322 Specified or default database already
enabled for text. ". SQLSTATE 38H14
```

使用上の注意

- テキスト検索管理プロシージャーは、データベースへの既存の接続を使用します。現行トランザクションは、プロシージャーの完了結果に応じてコミットされるかまたはロールバックされる場合があります。そのため、そのようなコミットまたはロールバックからの予期しない影響を避けるために、すべてのトランザクション変更をコミットすることができます。これを行う 1 つの方法は、AUTOCOMMIT をオンにすることです。
- このプロシージャーが実行されると、
  - このプロシージャーは、スキーマ SYSIBMTS にテキスト検索管理カタログ表およびビューなどのデータベース・オブジェクトを作成します。このオブジェクトは、データベースのデフォルトの表スペース (IBMDEFAULTGROUP) に置かれます。
  - テキスト検索索引の設定済みデータベース・デフォルトは、ビュー SYSIBMTS.TSDEFAULTS で入手可能です。
  - コマンドが正常に完了したら、テキスト検索カタログ表およびビューが作成され、使用可能になります。

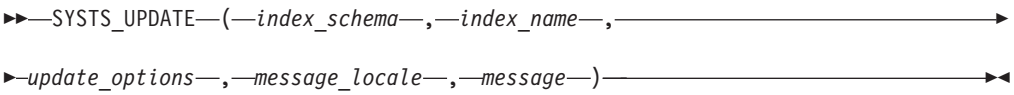
SYSTS_UPDATE プロシージャー - テキスト検索索引の更新

このプロシージャーは、索引が関連付けられるテキスト列の現行の内容を反映するテキスト検索索引を更新します。

更新が実行されている間も、検索は可能です。更新が完了するまでは、検索は部分的に更新された索引に対して実行されます。

このプロシージャーは UPDATE INDEX テキスト検索管理コマンドをデータベース・サーバー上で発行します。

構文



スキーマは SYSPROC です。

プロシージャー・パラメーター

index_schema
テキスト検索索引のスキーマを指定する、タイプ VARCHAR(128) の入力引数。 *index_schema* は、DB2 スキーマ名の命名上の制約に従う必要があります。引数が NULL または空ストリングである場合、値 CURRENT SCHEMA が使用されます。 *index_schema* には、大文字小文字の区別があります。

index_name
索引の名前を指定する、タイプ VARCHAR (128) の入力引数。 *index_schema* とともに使用すると、データベースのテキスト検索索引を一意的に識別します。*index_name* には、大文字小文字の区別があります。

update_options
更新オプションを指定する、タイプ VARCHAR(32K) の入力引数。可能な値は次のとおりです。

- USING UPDATE MINIMUM: この設定は、CREATE INDEX テキスト検索管理コマンドおよび SYSTS_CREATE プロシージャーからの UPDATE MINIMUM 設定を受け入れます。
- UPDATEAUTOCOMMIT: この設定は、この更新実行期間中のテキスト索引の commitcount 定義をオーバーライドします。
- NULL または空ストリング (''): プロシージャーの呼び出し時に更新は無条件に開始されます。

message_locale
戻されるすべてのエラー・メッセージに使用されるロケールを指定する、タイプ VARCHAR(33) の入力引数。引数が NULL または空ストリングであるか、指定したロケールのメッセージ・ファイルがサーバーで使用できない場合には、'en_US' が使用されます。

message
正常に完了した操作の警告または通知メッセージを指定する、タイプ VARCHAR(32K) の出力引数。

許可

ステートメントの許可 ID によって保持されている特権には、少なくとも以下のいずれかが含まれていなければなりません。

- そのテキスト索引の定義されている表に対する CONTROL 特権。
- DATAACCESS 権限

例

例 1: 以下の例では、*index_schema* 'db2ts' および *index_name* 'myTextIndex' で作成されたテキスト検索索引が更新されます。 *update_options* が NULL 値であることは、ストアード・プロシージャーの呼び出し時に更新が無条件で開始されることを意味します。すべてのエラー・メッセージは英語で戻すことが要求されます。このプロシージャーが成功すると、操作の成功を示す出力パラメーター・メッセージが呼び出し元に戻されます。

```
CALL SYSPROC.SYSTS_UPDATE('db2ts', 'myTextIndex', '', 'en_US', ?)
```

以下はこの照会の出力例です。

```
Value of output parameters
--------------------------
Parameter Name  : MESSAGE
Parameter Value : Operation completed successfully.

Return Status = 0
```

例 2: 以下の例では、SYSTS_UPDATE が呼び出され、*index_schema* 'db2ts' および *index_name* 'myTextIndex' があるテキスト検索索引が更新されます。この索引は存在せず、結果としてエラーを戻します。

```
CALL SYSPROC.SYSTS_UPDATE('db2ts', 'myTextIndex', 'USING UPDATE MINIMUM',
'en_US', ?)
```

以下はこの照会の出力例です。

```
SQL20427N An error occurred during a text search administration
procedure or command. The error message is "CIE00316 Text search
index "db2ts"."myTextIndex" does not exist. ". SQLSTATE 38H14
```

使用上の注意

- テキスト検索管理プロシージャーは、データベースへの既存の接続を使用します。現行トランザクションは、プロシージャーの完了結果に応じてコミットされるかまたはロールバックされる場合があります。そのため、そのようなコミットまたはロールバックからの予期しない影響を避けるために、すべてのトランザクション変更をコミットすることができます。これを行う 1 つの方法は、AUTOCOMMIT をオンにすることです。
- 複数のプロシージャーまたはコマンドは、競合する可能性がある場合は、テキスト検索索引に対して同時に実行することはできません。競合するプロシージャーおよびコマンドの一部は以下のとおりです。
  - SYSTS_ALTER プロシージャーまたは ALTER INDEX db2ts コマンド
  - SYSTS_CLEAR_EVENTS プロシージャーまたは CLEAR EVENTS FOR INDEX db2ts コマンド
  - SYSTS_DISABLE プロシージャーまたは DISABLE DATABASE FOR TEXT db2ts コマンド
  - SYSTS_DROP プロシージャーまたは DROP INDEX db2ts コマンド
  - STOP FOR TEXT db2ts コマンド
  - SYSTS_UPDATE プロシージャーまたは UPDATE INDEX db2ts コマンド

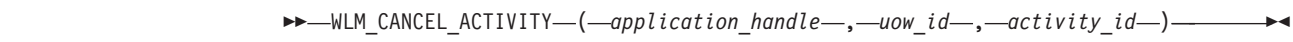
  競合がある場合、プロシージャーは SQLCODE -20426 および SQLSTATE 38H13 を戻します。
- このプロシージャーは、すべての索引更新処理が完了するまで戻されません。この期間は、これから索引付けされる文書の数および既に索引付けされている文書の数に応じて異なります。索引のコレクション名は SYSIBMTS.TSCOLLECTIONNAMES ビュー (列 COLLECTIONNAME) から取られます。
- 個々の文書にエラーがある場合、文書を訂正する必要があります。エラーがある文書の主キーは、索引のイベント表で参照できます。ユーザー表内の対応する行を変更することで、SYSTS_UPDATE への次の呼び出しはこれらの文書を再処理します。
- このプロシージャーが実行されると、
  - 行がイベント表に挿入されます (パーサー・エラー情報を含む)。インクリメンタル更新の場合、情報は索引ステージング表から削除されます。最初の更新の前に、これはユーザー表にトリガーを作成します。
  - 集合が更新されます。新規文書または変更済み文書は構文解析され、索引付けされます。削除された文書は索引から廃棄されます。

第 20 章 ワークロード管理ルーチン

WLM_CANCEL_ACTIVITY - アクティビティーのキャンセル

このプロシージャーは、指定されたアクティビティーをキャンセルします。キャンセルが行われる場合、キャンセルされたアクティビティーをサブミットしたアプリケーションにエラー・メッセージが戻されます。

構文



スキーマは SYSPROC です。

プロシージャー・パラメーター

application_handle
アクティビティーがキャンセルされるアプリケーション・ハンドルを指定する、タイプ BIGINT の入力引数。引数が NULL の場合、アクティビティーは検出されず、SQLSTATE 5U035 の SQL4702N が戻されます。

uow_id
キャンセルされるアクティビティーの作業単位 ID を指定する、タイプ INTEGER の入力引数。引数が NULL の場合、アクティビティーは検出されず、SQLSTATE 5U035 の SQL4702N が戻されます。

activity_id
キャンセルされる作業単位内のアクティビティーを一意的に識別するアクティビティー ID を指定する、タイプ INTEGER の入力引数。引数が NULL の場合、アクティビティーは検出されず、SQLSTATE 5U035 の SQL4702N が戻されます。

許可

WLM_CANCEL_ACTIVITY プロシージャーに対する EXECUTE 特権。

例

管理者は WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES 表関数を使用して、アクティビティーのアプリケーション・ハンドル、作業単位 ID、およびアクティビティー ID を検索できます。アプリケーション・ハンドル 1、作業単位 ID 2、およびアクティビティー ID 3 のアクティビティーをキャンセルするには、次のようにします。

```
CALL WLM_CANCEL_ACTIVITY(1, 2, 3)
```

使用上の注意

- アクティビティーが見つからない場合、SQLSTATE 5U035 の SQL4702N が戻されます。

© Copyright IBM Corp. 2006, 2010

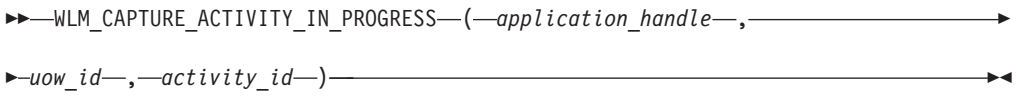
- アクティビティーが正しい状態でない (初期化されていない) ためにキャンセルできない場合、SQLSTATE 5U016 の SQL4703N (理由コード 1) が戻されます。
- アクティビティーが正常にキャンセルされた場合、SQLSTATE 57014 の SQL4725N がキャンセルされたアプリケーションに戻されます。
- キャンセル時に、コーディネーターが別のアクティビティーの要求を処理しているかまたはアイドル状態である場合、アクティビティーは CANCEL_PENDING 状態になり、コーディネーターが次の要求を処理するとキャンセルされます。

WLM_CAPTURE_ACTIVITY_IN_PROGRESS - アクティビティー・イベント・モニターのアクティビティー情報の収集

WLM_CAPTURE_ACTIVITY_IN_PROGRESS プロシージャーは、指定されたアクティビティーに関する情報を収集し、その情報をアクティブなアクティビティー・イベント・モニターに書き込みます。

子アクティビティーを持つアクティビティーにこのプロシージャーを適用する場合、プロシージャーはそれぞれの子アクティビティーのレコードを再帰的に生成します。この情報は、プロシージャーを呼び出すときに収集されて送信されます。プロシージャーは、親アクティビティーによる実行の完了を待機しません。イベント・モニター内のアクティビティーのレコードは部分レコードとしてマークが付けられます。

構文



スキーマは SYSPROC です。

プロシージャー・パラメーター

以下のパラメーターがすべて指定されないと、アクティビティーは検出されず、SQL4702N が SQLSTATE 5U035 とともに返されます。

application_handle
そのアクティビティー情報がキャプチャーされるアプリケーションのハンドルを指定する、タイプ BIGINT の入力引数。

uow_id
その情報がキャプチャーされるアクティビティーの作業単位 ID を指定する、タイプ INTEGER の入力引数。

activity_id
その情報がキャプチャーされる作業単位内のアクティビティーを一意的に識別するアクティビティー ID を指定する、タイプ INTEGER の入力引数。

許可

WLM_CAPTURE_ACTIVITY_IN_PROGRESS プロシージャーに対する EXECUTE 特権。

例

ストアード・プロシージャー MYSCHEMA.MYSLOWSTP の実行がいつもより遅いように感じる、とユーザーが苦情を言うとします。管理者はスローダウンの原因の調査に乗り出します。ストアード・プロシージャーを実行しながらの調査は実際的とは言えないので、管理者はストアード・プロシージャー・アクティビティーおよびその中にネストされたすべてのアクティビティーに関する情報をキャプチャーすることにします。

DB2ACTIVITIES という名前の DB2 アクティビティーのイベント・モニターがアクティブ化されています。管理者は WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES 関数を使用して、このストアード・プロシージャーの呼び出しに関するアプリケーション・ハンドル、作業単位 ID、およびアクティビティー ID を取得します。ここで、管理者がアクティビティーがアプリケーション・ハンドル 1、作業単位 ID 2、およびアクティビティー ID 3 で識別されていると想定し、WLM_CAPTURE_ACTIVITY_IN_PROGRESS への呼び出しを次のように発行できます。

```
CALL WLM_CAPTURE_ACTIVITY_IN_PROGRESS(1,2,3)
```

プロシージャーが完了した後、管理者は次の表関数を使用してアクティビティーが時間を要した場所を発見することができます。関数は、DB2ACTIVITIES イベント・モニターから情報を取得します。

```
CREATE FUNCTION SHOWCAPTUREDACTIVITY(APPHNDL BIGINT,
                                     UOWID INTEGER,
                                     ACTIVITYID INTEGER)
  RETURNS TABLE (UOW_ID INTEGER, ACTIVITY_ID INTEGER, STMT_TEXT VARCHAR(40),
    LIFE_TIME DOUBLE)
  LANGUAGE SQL
  READS SQL DATA
  NO EXTERNAL ACTION
  DETERMINISTIC
  RETURN WITH RAH (LEVEL, APPL_ID, PARENT_UOW_ID, PARENT_ACTIVITY_ID,
          UOW_ID, ACTIVITY_ID, STMT_TEXT, ACT_EXEC_TIME) AS
  (SELECT 1, ROOT.APPL_ID, ROOT.PARENT_UOW_ID,
          ROOT.PARENT_ACTIVITY_ID, ROOT.UOW_ID, ROOT.ACTIVITY_ID,
          ROOTSTMT.STMT_TEXT, ACT_EXEC_TIME
   FROM ACTIVITY_DB2ACTIVITIES ROOT, ACTIVITYSTMT_DB2ACTIVITIES ROOTSTMT
   WHERE ROOT.APPL_ID = ROOTSTMT.APPL_ID AND ROOT.AGENT_ID = APPHNDL
     AND ROOT.UOW_ID = ROOTSTMT.UOW_ID AND ROOT.UOW_ID = UOWID
     AND ROOT.ACTIVITY_ID = ROOTSTMT.ACTIVITY_ID AND ROOT.ACTIVITY_ID = ACTIVITYID
  UNION ALL
   SELECT PARENT.LEVEL +1, CHILD.APPL_ID, CHILD.PARENT_UOW_ID,
          CHILD.PARENT_ACTIVITY_ID, CHILD.UOW_ID,
          CHILD.ACTIVITY_ID, CHILDSTMT.STMT_TEXT, CHILD.ACT_EXEC_TIME
   FROM RAH PARENT, ACTIVITY_DB2ACTIVITIES CHILD,
        ACTIVITYSTMT_DB2ACTIVITIES CHILDSTMT
   WHERE PARENT.APPL_ID = CHILD.APPL_ID AND
         CHILD.APPL_ID = CHILDSTMT.APPL_ID AND
         PARENT.UOW_ID = CHILD.PARENT_UOW_ID AND
         CHILD.UOW_ID = CHILDSTMT.UOW_ID AND
         PARENT.ACTIVITY_ID = CHILD.PARENT_ACTIVITY_ID AND
         CHILD.ACTIVITY_ID = CHILDSTMT.ACTIVITY_ID AND
         PARENT.LEVEL < 64
  )
SELECT UOW_ID, ACTIVITY_ID, SUBSTR(STMT_TEXT,1,40),
               ACT_EXEC_TIME AS
               LIFE_TIME
FROM RAH
```

以下のサンプル照会では、表関数を使用します。

```
SELECT * FROM TABLE(SHOWCAPTUREDACTIVITY(1, 2, 3))
  AS ACTS ORDER BY UOW_ID, ACTIVITY_ID
```

使用上の注意

アクティブなアクティビティー・イベント・モニターがない場合、SQLSTATE 01H53 の SQL1633W が戻されます。

アクティビティー情報は、アクティビティーのコーディネーター・パーティションでのみ収集されます。

WLM_COLLECT_STATS - ワークロード管理統計の収集およびリセット

WLM_COLLECT_STATS プロシージャーは、サービス・クラス、ワークロード、作業クラス、およびしきい値キューの統計を収集し、統計イベント・モニターに書き込みます。また、このプロシージャーは、サービス・クラス、ワークロード、作業クラス、およびしきい値キューの統計のリセットも行います。アクティブな統計イベント・モニターがない場合、プロシージャーは統計のリセットのみを行います。

構文

スキーマは SYSPROC です。

許可

WLM_COLLECT_STATS プロシージャーに対する EXECUTE 特権。

例

例 1: WLM_COLLECT_STATS を呼び出して、統計を収集およびリセットする。

```
CALL WLM_COLLECT_STATS()
```

以下はこの照会の出力例です。

```
Return Status = 0
```

例 2: 別の呼び出しが進行中に、WLM_COLLECT_STATS を呼び出して、統計を収集およびリセットする。

```
CALL WLM_COLLECT_STATS()
```

以下はこの照会の出力例です。

```
SQL1632W The collect and reset statistics request was ignored because
another collect and reset statistics request is already in progress.
```

使用上の注意

WLM_COLLECT_STATS プロシージャーは、**wlm_collect_int** データベース構成パラメーターで定義された間隔で自動的に行われるものと同じ収集操作 (アクティブな統計イベント・モニターへの統計の送信) およびリセット操作を実行します。

別の収集およびリセット要求の進行中 (例えばプロシージャーの別の呼び出しの実行中または自動収集の発生中) にプロシージャーを呼び出すと、SQL1632W が SQLSTATE 01H53 とともに返され、新しい要求は無視されます。

WLM_COLLECT_STATS プロシージャーは、収集およびリセット・プロセスのみ開始します。このプロシージャーは、すべての統計がアクティブな統計イベント・モニターに書き込まれる前に呼び出し元に戻る場合があります。統計の収集およびリセットが発生する頻度に応じて、WLM_COLLECT_STATS プロシージャーの呼び出し (これ自体がアクティビティー) は統計において、前の収集間隔または直前に開始された新規の収集間隔のいずれかでカウントされます。

WLM_GET_CONN_ENV - 接続のアクティビティー・データ収集の設定の取得

WLM_GET_CONN_ENV 表関数は、特定の接続の、アクティビティー・データおよびセクション実行時統計の収集を制御する設定の値を戻します。この表関数を使用することで、WLM_SET_CONN_ENV ストアード・プロシージャーによって適用される設定の現行値を検査できます。

構文

パラメーター

application_handle
情報が戻される接続のアプリケーション・ハンドルを指定するタイプ BIGINT の入力引数。 NULL 値を使用することで、プロシージャーが呼び出された接続を示すことができます。

許可

WLM_GET_CONN_ENV 表関数に対する EXECUTE 特権。

例

以下の照会は、現行接続のアクティビティーが収集されているかどうかを検査します。

```
SELECT application_handle,
       xmlparse(document details preserve whitespace)
FROM TABLE (
   WLM_GET_CONN_ENV(
      cast(NULL as bigint))
   ) connenv
```

以下はこの照会の出力例です。

```
APPLICATION_HANDLE   DETAILS
-------------------- ------------------------------------------------------
                   7 <wlm_conn_env
                     xmlns=http://www.ibm.com/xmlns/prod/db2/mon
                     release="9070100">
```

```
<collectactdata>NONE</collectactdata>
<collectactpartition>COORDINATOR</collectactpartition>
<collectsectionactuals>NONE</collectsectionactuals>
</wlm_conn_env>
```

使用上の注意

WLM_GET_CONN_ENV 表関数は、接続の WLM 環境情報を 1 つの XML 文書として戻すため、出力フォーマットを最大限柔軟性のあるものにします。 出力は XML パーサーで直接解析でき、XMLTABLE 関数でリレーショナル形式に変換することもできます。

DETAILS 列に戻される XML 文書のスキーマは、ファイル sqllib/misc/DB2MonRoutines.xsd で入手できます。詳細は、ファイル sqllib/misc/DB2MonCommon.xsd 内にあります。

戻される情報

表 238. WLM_GET_CONN_ENV について戻される情報

列名	データ・タイプ	説明
APPLICATION_HANDLE	BIGINT	application_handle - アプリケーション・ハンドル
DETAILS	BLOB(8M)	接続環境の詳細を含む XML 文書。エレメントの説明については、本書の表 239を参照してください。

戻される詳細設定

表 239. WLM_GET_CONN_ENV について戻される詳細メトリック

エレメント名	データ・タイプ	説明または対応するモニター・エレメント
collectactdata	xs:string(255)	収集されているアクティビティー・データの種類を指定します (存在する場合)。可能な値は以下のとおりです。 • NONE • WITHOUT DETAILS • WITH DETAILS • WITH DETAILS, SECTION • WITH DETAILS, SECTION AND VALUES • WITH DETAILS AND VALUES これらのオプションについて詳しくは、WLM_SET_CONN_ENV プロシージャーに関する情報を参照してください。
collectactpartition	xs:string(255)	どこでアクティビティー・データを収集するかを指定します。可能な値は以下のとおりです。 • COORDINATOR • ALL これらのオプションについて詳しくは、WLM_SET_CONN_ENV プロシージャーに関する情報を参照してください。

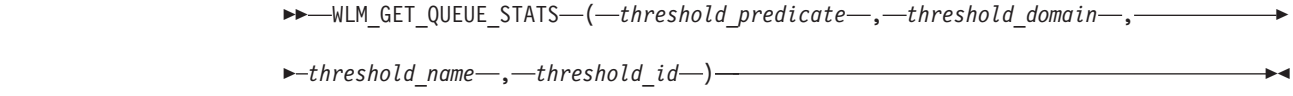
表 239. WLM_GET_CONN_ENV について戻される詳細メトリック (続き)

エレメント名	データ・タイプ	説明または対応するモニター・エレメント
collectsectionactuals	xs:string(255)	セクション実行時統計を収集するかどうかを指定します。使用できる値は以下のとおりです。 • NONE • BASE これらのオプションについて詳しくは、WLM_SET_CONN_ENV プロシージャーに関する情報を参照してください。

WLM_GET_QUEUE_STATS 表関数 - しきい値キュー統計を戻す

WLM_GET_QUEUE_STATS 関数は、すべてのアクティブ・パーティション上の 1 つ以上のしきい値キューの基本統計を戻します。この関数は、しきい値キューごとに 1 行の統計を戻します。

構文



スキーマは SYSPROC です。

表関数パラメーター

threshold_predicate
しきい値述部を指定する、タイプ VARCHAR(27) の入力引数。可能な値は、以下のとおりです。

CONCDBC
並行データベース・コーディネーター・アクティビティーしきい値

DBCONN
データベース・パーティション接続合計しきい値

SCCONN
サービス・クラス・パーティション接続合計しきい値

引数が NULL または空ストリングである場合、他の基準を満たすすべてのしきい値についてデータが戻されます。

threshold_predicate の値は、SYSCAT.THRESHOLDS ビューの THRESHOLDPREDICATE 列の値と一致します。

threshold_domain
しきい値ドメインを指定する、タイプ VARCHAR(18) の入力引数。可能な値は、以下のとおりです。

DB データベース

SB サービス・サブクラス

SP　サービス・スーパークラス

WA　作業アクション・セット

引数が NULL または空ストリングである場合、他の基準を満たすすべてのしきい値についてデータが戻されます。

threshold_domain の値は、SYSCAT.THRESHOLDS ビューの DOMAIN 列の値と一致します。

threshold_name
しきい値の名前を指定する、タイプ VARCHAR(128) の入力引数。引数が NULL または空ストリングである場合、他の基準を満たすすべてのしきい値についてデータが戻されます。 *threshold_name* の値は、SYSCAT.THRESHOLDS ビューの THRESHOLDNAME 列の値と一致します。

threshold_id
しきい値 ID を指定する、タイプ INTEGER の入力引数。引数が NULL または -1 である場合、他の基準を満たすすべてのしきい値についてデータが戻されます。 *threshold_id* の値は、SYSCAT.THRESHOLDS ビューの THRESHOLDID 列の値と一致します。

許可

WLM_GET_QUEUE_STATS 関数に対する EXECUTE 特権。

例

以下の照会は、システム上のすべてのキューの基本統計をすべてのパーティションの間で表示します。

```
SELECT substr(THRESHOLD_NAME, 1, 6) THRESHNAME,
     THRESHOLD_PREDICATE,
     THRESHOLD_DOMAIN,
     DBPARTITIONNUM PART,
     QUEUE_SIZE_TOP,
     QUEUE_TIME_TOTAL,
     QUEUE_ASSIGNMENTS_TOTAL QUEUE_ASSIGN
FROM table(WLM_GET_QUEUE_STATS('', '', '', -1)) as QSTATS
```

出力例を以下に示します。

```
THRESHNAME THRESHOLD_PREDICATE         THRESHOLD_DOMAIN   ...
---------- --------------------------- ------------------ ...
LIMIT1     CONCDBC                     DB                 ...
LIMIT2     SCCONN                      SP                 ...
LIMIT3     DBCONN                      DB                 ...

... PART QUEUE_SIZE_TOP QUEUE_TIME_TOTAL QUEUE_ASSIGN
... ---- -------------- ---------------- ------------
... 0             12          1238540          734
... 0              4           741249           24
... 0              7           412785          128
```

使用上の注意

この関数は、(パーティション上の) キュー全体または (1 つ以上のキューの) パーティション全体のデータ集約は行いません。ただし、上の例で示された SQL 照会を使用すると、データを集約することができます。

戻される情報

表 240. WLM_GET_QUEUE_STATS について戻される情報

列名	データ・タイプ	説明
THRESHOLD_PREDICATE	VARCHAR(27)	このキューを担当するしきい値のしきい値述部。可能な値は、以下のとおりです。 *CONCDBC* 並行データベース・コーディネーター・アクティビティーしきい値 *DBCONN* データベース・パーティション接続合計しきい値 *SCCONN* サービス・クラス・パーティション接続合計しきい値 しきい値述部の値は、SYSCAT.THRESHOLDS ビューの THRESHOLDPREDICATE 列の値と一致します。
THRESHOLD_DOMAIN	VARCHAR(18)	このキューを担当するしきい値のドメイン。可能な値は、以下のとおりです。 *DB* データベース *SB* サービス・サブクラス *SP* サービス・スーパークラス *WA* 作業アクション・セット しきい値ドメインの値は、SYSCAT.THRESHOLDS ビューの DOMAIN 列の値と一致します。
THRESHOLD_NAME	VARCHAR(128)	このキューを担当するしきい値の固有の名前。しきい値名前の値は、SYSCAT.THRESHOLDS ビューの THRESHOLDNAME 列の値と一致します。
THRESHOLD_ID	INTEGER	このキューを担当するしきい値のユニーク ID。しきい値 ID の値は、SYSCAT.THRESHOLDS ビューの THRESHOLDID 列の値と一致します。
DBPARTITIONNUM	SMALLINT	このレコードが収集されたパーティション番号。

表 240. WLM_GET_QUEUE_STATS について戻される情報 (続き)

列名	データ・タイプ	説明
SERVICE_SUPERCLASS_NAME	VARCHAR(128)	このキューを担当するしきい値のドメインであるサービス・スーパークラスの名前。しきい値のドメインがサービス・スーパークラスまたはサービス・サブクラスでない場合、列の値は NULL です。
SERVICE_SUBCLASS_NAME	VARCHAR(128)	このキューを担当するしきい値のドメインであるサービス・サブクラスの名前。しきい値のドメインがサービス・サブクラスでない場合、列の値は NULL です。
WORK_ACTION_SET_NAME	VARCHAR(128)	このキューを担当するしきい値のドメインである作業アクション・セットの名前。しきい値のドメインが作業アクション・セットでない場合、列の値は NULL です。
WORK_CLASS_NAME	VARCHAR(128)	その作業アクションがこのキューを担当するしきい値のドメインである作業アクション・セットに属する作業クラスの名前。しきい値のドメインが作業アクション・セットでない場合、列の値は NULL です。
WORKLOAD_NAME	VARCHAR(128)	このキューを担当するしきい値のドメインであるワークロードの名前。しきい値のドメインがワークロードでない場合、列の値は NULL です。
LAST_RESET	TIMESTAMP	統計が最後にリセットされた時刻。統計のリセットを起動するイベントは 4 つあります。 • WLM_COLLECT_STATS プロシージャーの呼び出し。 • **wlm_collect_int** 構成パラメーター (これにより、収集とリセットが発生します)。 • データベースの再活動化。 • キュー統計が報告されているしきい値の変更、およびその変更のコミット。 LAST_RESET タイム・スタンプが現地時間である。
QUEUE_SIZE_TOP	INTEGER	最後のリセット以降の、キュー内の接続またはアクティビティーの最大数。
QUEUE_TIME_TOTAL	BIGINT	最後のリセット以降、キューに置かれているすべての接続またはアクティビティーについて、このキューで費やされた時間の合計。単位はミリ秒です。

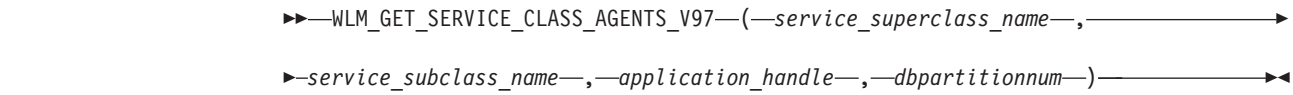
表 240. WLM_GET_QUEUE_STATS について戻される情報 (続き)

列名	データ・タイプ	説明
QUEUE_ASSIGNMENTS_TOTAL	BIGINT	最後のリセット以降、このキューに割り当てられた接続またはアクティビティーの数。
QUEUE_SIZE_CURRENT	INTEGER	キュー内の接続またはアクティビティーの数。
QUEUE_TIME_LATEST	BIGINT	最後の接続またはアクティビティーがキューをそのままにしておくためにキューで費やした時間。単位はミリ秒です。
QUEUE_EXIT_TIME_LATEST	TIMESTAMP	最後の接続またはアクティビティーがキューをそのままにしておいた時間。
THRESHOLD_CURRENT_CONCURRENCY	INTEGER	しきい値に従って現在実行中の接続またはアクティビティーの数。
THRESHOLD_MAX_CONCURRENCY	INTEGER	しきい値によって現在実行中の接続またはアクティビティーの最大数。

WLM_GET_SERVICE_CLASS_AGENTS_V97 表関数 - サービス・クラスで実行中のエージェントのリスト

WLM_GET_SERVICE_CLASS_AGENTS_V97 関数は、指定されたサービス・クラスで実行しているか、または指定されたアプリケーションの代わりに実行している、指定されたパーティションにあるエージェント、fenced モード・プロセス (db2fmp プロセス)、およびシステム・エンティティーのリストを戻します。システム・エンティティーは、非エージェント・スレッドおよびプロセス (ページ・クリーナーおよびプリフェッチャーなど) です。

構文



スキーマは SYSPROC です。

表関数パラメーター

service_superclass_name
現在接続されているデータベースのサービス・スーパークラスの名前を指定する、タイプ VARCHAR(128) の入力引数。引数が NULL または空ストリングである場合、データベース内のすべてのスーパークラスについてデータが取得されます。

service_subclass_name
スーパークラス内の特定のサブクラスを参照する、タイプ VARCHAR(128) の入力引数。引数が NULL または空ストリングである場合、データベース内のすべてのサブクラスについてデータが取得されます。

application_handle
エージェント情報が戻されるアプリケーション・ハンドルを指定する、タイプ BIGINT の入力引数。引数が NULL である場合、データベース内のすべてのアプリケーションについてデータが取得されます。アプリケーション・ハンドルが 0 の場合、システム・エンティティーのみ戻されます。

dbpartitionnum
現在接続されているデータベースと同じインスタンス内のパーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションには -1、すべてのデータベース・パーティションには -2 を指定します。NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

WLM_GET_SERVICE_CLASS_AGENTS_V97 関数に対する EXECUTE 特権。

例

例 1

以下の照会は、すべてのデータベース・パーティションについてアプリケーション・ハンドル 1 に関連付けられたエージェントのリストを戻します。 LIST APPLICATIONS コマンドまたは WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES_V97 表関数を使用して、アプリケーション・ハンドルを判別することができます。

```
SELECT SUBSTR(CHAR(APPLICATION_HANDLE),1,7) AS APPHANDLE,
  SUBSTR(CHAR(DBPARTITIONNUM),1,4) AS PART,
  SUBSTR(CHAR(AGENT_TID),1,9) AS AGENT_TID,
  SUBSTR(AGENT_TYPE,1,11) AS AGENTTYPE,
  SUBSTR(AGENT_STATE,1,10) AS AGENTSTATE,
  SUBSTR(REQUEST_TYPE,1,12) AS REQTYPE,
  SUBSTR(CHAR(UOW_ID),1,6) AS UOW_ID,
  SUBSTR(CHAR(ACTIVITY_ID),1,6) AS ACT_ID
FROM TABLE(WLM_GET_SERVICE_CLASS_AGENTS_V97(CAST(NULL AS VARCHAR(128)),
  CAST(NULL AS VARCHAR(128)), 1, -2)) AS SCDETAILS
ORDER BY APPHANDLE, PART, AGENT_TID
```

出力例を以下に示します。

```
APPHANDLE PART AGENT_TID AGENTTYPE   AGENTSTATE REQTYPE        UOW_ID ACT_ID
--------- ---- --------- ----------- ---------- -------------- ------ ------
1         0    3         COORDINATOR ACTIVE     FETCH          1      5
1         0    4         SUBAGENT    ACTIVE     SUBSECTION:1   1      5
1         1    2         SUBAGENT    ACTIVE     SUBSECTION:2   1      5
```

この出力は、UOW ID 1 およびアクティビティー ID 5 のアクティビティーの代わりに作動している、パーティション 0 上のコーディネーター・エージェントとサブエージェント、およびパーティション 1 上のサブエージェントを示しています。値 COORDINATOR を持つ AGENTTYPE 列では、REQTYPE 列に FETCH の値があります (これは、メインまたは初期要求タイプを示しています)。これは、要求のタイプがコーディネーター・エージェントに対するフェッチ要求であることを意味しています。

例 2

以下の照会は、エージェントがどのロックを待機しているかを判別します。

```
db2 select event_object, event_type, event_state, varchar(event_object_name, 30)
  as event_object_name
  from table(wlm_get_service_class_agents_v97('','',cast(NULL as bigint), -1)) as t
```

出力例を以下に示します。

```
EVENT_OBJECT    EVENT_TYPE        EVENT_STATE         EVENT_OBJECT_NAME
--------------- ----------------- ------------------- --------------------------
REQUEST         PROCESS           EXECUTING           -
REQUEST         PROCESS           EXECUTING           -
REQUEST         PROCESS           EXECUTING           -
REQUEST         PROCESS           EXECUTING           -
REQUEST         PROCESS           EXECUTING           -
REQUEST         PROCESS           EXECUTING           -
REQUEST         PROCESS           EXECUTING           -
REQUEST         PROCESS           EXECUTING           -
REQUEST         WAIT              IDLE                -
LOCK            ACQUIRE           IDLE                02000500000000000000000054
ROUTINE         PROCESS           EXECUTING           -
REQUEST         PROCESS           EXECUTING           -
REQUEST         PROCESS           EXECUTING           -
REQUEST         PROCESS           EXECUTING           -
REQUEST         PROCESS           EXECUTING           -
REQUEST         PROCESS           EXECUTING           -
REQUEST         PROCESS           EXECUTING           -
REQUEST         PROCESS           EXECUTING           -
REQUEST         PROCESS           EXECUTING           -
REQUEST         PROCESS           EXECUTING           -

  21 record(s) selected.
```

後で同じ照会を使用すると、WLM しきい値によってエージェントがキューに入れられたことが示されます。

```
EVENT_OBJECT    EVENT_TYPE        EVENT_STATE         EVENT_OBJECT_NAME
--------------- ----------------- ------------------- --------------------------
REQUEST         PROCESS           EXECUTING           -
REQUEST         PROCESS           EXECUTING           -
REQUEST         PROCESS           EXECUTING           -
REQUEST         PROCESS           EXECUTING           -
REQUEST         PROCESS           EXECUTING           -
REQUEST         PROCESS           EXECUTING           -
REQUEST         PROCESS           EXECUTING           -
REQUEST         PROCESS           EXECUTING           -
REQUEST         PROCESS           EXECUTING           -
WLM_QUEUE       WAIT              IDLE                MYCONCDBCOORDTH
ROUTINE         PROCESS           EXECUTING           -
REQUEST         PROCESS           EXECUTING           -
REQUEST         PROCESS           EXECUTING           -
REQUEST         PROCESS           EXECUTING           -
REQUEST         PROCESS           EXECUTING           -
REQUEST         PROCESS           EXECUTING           -
REQUEST         PROCESS           EXECUTING           -
REQUEST         PROCESS           EXECUTING           -
REQUEST         PROCESS           EXECUTING           -
REQUEST         PROCESS           EXECUTING           -

  21 record(s) selected.
```

使用上の注意

これらのパラメーターの作用については、論理積 (AND) が取られます。つまり、矛盾する入力パラメーターを指定する (例えば、サービス・スーパークラス SUP_A

とサブクラス SUB_B を、SUB_B が SUP_A のサブクラスにならないように指定する) 場合、行は戻されません。

戻される情報

表 241. WLM_GET_SERVICE_CLASS_AGENTS_V97 によって戻される情報

列名	データ・タイプ	説明
SERVICE_SUPERCLASS_NAME	VARCHAR (128)	このレコードが収集されたサービス・スーパークラスの名前。
SERVICE_SUBCLASS_NAME	VARCHAR (128)	このレコードが収集されたサービス・サブクラスの名前。
APPLICATION_HANDLE	BIGINT	アプリケーションのシステム全体のユニーク ID。単一パーティション・データベースの場合、この ID は 16 ビットのカウンターで構成されます。複数パーティション・データベースの場合、この ID はコーディネーター・パーティション番号と 16 ビットのカウンターを連結したもので構成されます。さらに、この ID はアプリケーションが 2 次接続を行うすべてのパーティションにおいて同じです。
DBPARTITIONNUM	SMALLINT	このレコードが収集されたパーティション番号。
ENTITY	VARCHAR (32)	以下の値のいずれか。 • エンティティーのタイプがエージェントである場合、値は db2agent です。 • エンティティーのタイプが fenced モード・プロセスである場合、値は db2fmp (*pid*) です。ここで、*pid* は fenced モード・プロセスのプロセス ID です。 • それ以外の場合、値はシステム・エンティティーの名前です。
WORKLOAD_NAME	VARCHAR (128)	このレコードが収集されたワークロードの名前。
WORKLOAD_OCCURRENCE_ID	INTEGER	ワークロード・オカレンスの ID。ワークロード・オカレンスがコーディネーター・データベース・パーティション番号およびワークロード名と結合されていなければ、この ID はワークロード・オカレンスを一意的に識別しません。
UOW_ID	INTEGER	このアクティビティーが開始された作業単位のユニーク ID。
ACTIVITY_ID	INTEGER	作業単位内の固有のアクティビティー ID。
PARENT_UOW_ID	INTEGER	アクティビティーの親アクティビティーが開始された作業単位のユニーク ID。このアクティビティーに親がない場合、この列の値は NULL です。
PARENT_ACTIVITY_ID	INTEGER	親のアクティビティー ID が activity_id と同じである、作業単位内の固有のアクティビティー ID。このアクティビティーに親がない場合、この列の値は NULL です。
AGENT_TID	BIGINT	エージェントまたはシステム・エンティティーのスレッド ID。この ID が使用できない場合、列の値は NULL です。

表 241. WLM_GET_SERVICE_CLASS_AGENTS_V97 によって戻される情報 (続き)

列名	データ・タイプ	説明
AGENT_TYPE	VARCHAR (32)	エージェント・タイプ。エージェント・タイプは以下のとおりです。 • COORDINATOR • OTHER • PDBSUBAGENT • SMPSUBAGENT 値が COORDINATOR である場合、エージェント ID はコンセントレーター環境で変わることがあります。
SMP_COORDINATOR	INTEGER	エージェントが SMP コーディネーターかどうかを示します。「はい」の場合は 1、「いいえ」の場合は 0。
AGENT_SUBTYPE	VARCHAR (32)	エージェント・サブタイプ。可能なサブタイプは以下のとおりです。 • DSS • OTHER • RPC • SMP
AGENT_STATE	VARCHAR (32)	エージェントが関連付けられているか、アクティブであるかを示します。可能な値は次のとおりです。 • ASSOCIATED • ACTIVE
EVENT_TYPE	VARCHAR (32)	このエージェントによって最後に処理されたイベントのタイプ。可能な値は、以下のとおりです。 • ACQUIRE • PROCESS • WAIT この列に使用できる値についての詳細は、 1016 ページの表 242 を参照してください。
EVENT_OBJECT	VARCHAR (32)	このエージェントによって最後に処理されたイベントのオブジェクト。可能な値は、以下のとおりです。 • COMPRESSION_DICTIONARY_BUILD • IMPLICIT_REBIND • INDEX_RECREATE • LOCK • LOCK_ESCALATION • QP_QUEUE • REMOTE_REQUEST • REQUEST • ROUTINE • WLM_QUEUE この列に使用できる値についての詳細は、 1016 ページの表 242 を参照してください。

表 241. *WLM_GET_SERVICE_CLASS_AGENTS_V97* によって戻される情報 (続き)

列名	データ・タイプ	説明
EVENT_STATE	VARCHAR (32)	このエージェントによって最後に処理されたイベントの状態。可能な値は、以下のとおりです。 • EXECUTING • IDLE この列に使用できる値についての詳細は、 1016 ページの表 242 を参照してください。
REQUEST_ID	VARCHAR (64)	要求 ID。この値は、*application_handle* の値と組み合わせて指定される場合のみ固有です。この組み合わせを使用して、長い時間を要する 1 つの要求と複数の要求とを区別することができます。例えば、1 つの長いフェッチと複数のフェッチを区別するなどです。
REQUEST_TYPE	VARCHAR (32)	要求のタイプ。可能な値は、以下のとおりです。 • コーディネーター・エージェントの場合: – CLOSE – COMMIT – COMPILE – DESCRIBE – EXCSQLSET – EXECIMMD – EXECUTE – FETCH – INTERNAL *number*。ここで、*number* は内部定数の値です。 – OPEN – PREPARE – REBIND – REDISTRIBUTE – REORG – ROLLBACK – RUNSTATS • AGENT_SUBTYPE が DSS または SMP であるサブエージェントの場合: – サブセクション番号がゼロ以外の場合は、SUBSECTION:*subsection number* の形式のサブセクション番号。そうでない場合は NULL を戻します。

表 241. WLM_GET_SERVICE_CLASS_AGENTS_V97 によって戻される情報 (続き)

列名	データ・タイプ	説明
REQUEST_TYPE (続く)	VARCHAR (32)	• AGENT_SUBTYPE が RPC であるサブエージェントの場合: – ABP – CATALOG – INTERNAL – REORG – RUNSTATS – WLM • SUBTYPE が OTHER であるサブエージェントの場合: – ABP – APP_RBSVPT – APP_RELSVPT – BACKUP – CLOSE – EXTERNAL_RBSVPT – EVMON – FORCE – FORCE_ALL – INTERNAL *number*。ここで、*number* は内部定数の値です。 – INTERRUPT – NOOP (要求がない場合) – QP – REDISTRIBUTE – STMT_RBSVPT – STOP_USING – UPDATE_DBM_CFG – WLM
NESTING_LEVEL	INTEGER	ID が activity_id であるアクティビティーのネスト・レベル。ネスト・レベルは、このアクティビティーが一番上の親アクティビティー内でネストされる深さです。
INVOCATION_ID	INTEGER	1 つのルーチン呼び出しを、作業単位内の同じネスト・レベルにある他の呼び出しと区別するための ID。これは作業単位内の特定のネスト・レベルにおいて固有です。
ROUTINE_ID	INTEGER	ルーチンのユニーク ID。アクティビティーがルーチンの一部でない場合には、この列の値は NULL です。

表 241. WLM_GET_SERVICE_CLASS_AGENTS_V97 によって戻される情報 (続き)

列名	データ・タイプ	説明
EVENT_OBJECT_NAME	VARCHAR (1024)	イベント・オブジェクト名。 EVENT_OBJECT の値が LOCK である場合、この列の値は、エージェントが待機するロックの名前です。 EVENT_OBJECT の値が WLM_QUEUE である場合、この列の値は、エージェントがキューに入れられている WLM しきい値の名前です。それ以外の場合、値は NULL です。 この列に使用できる値についての詳細は、表 242 を参照してください。
APPLICATION_NAME	VARCHAR (128)	appl_name - アプリケーション名
APPLICATION_ID	VARCHAR (128)	appl_id - アプリケーション ID
CLIENT_PID	BIGINT	client_pid - クライアント・プロセス ID
SESSION_AUTH_ID	VARCHAR (128)	session_auth_id - セッション許可 ID
REQUEST_START_TIME	TIMESTAMP	エージェントか現在処理中の要求の処理を開始した時刻
AGENT_STATE_LAST _UPDATE_TIME	TIMESTAMP	エージェントによって処理されているイベントが最後に変更された時刻。エージェントによって現在処理されているイベントは、EVENT_TYPE、EVENT_OBJECT、および EVENT_STATE 列で示されます。
EXECUTABLE_ID	VARCHAR (32) FOR BIT DATA	エージェントが作業中であるセクションを一意的に識別する、データ・サーバーで生成されたバイナリー・トークン。実行可能 ID を異なるモニター・インターフェースへの入力として使用し、セクションについてのデータを取得できます。エージェントがセクションで作業中でない場合は、NULL 値が戻されます。

注: EVENT_STATE、EVENT_TYPE、EVENT_OBJECT および EVENT_OBJECT_NAME 列値の可能な組み合わせを、以下の表にリストします。

表 242. EVENT_STATE、EVENT_TYPE、EVENT_OBJECT および EVENT_OBJECT_NAME 列値の可能な組み合わせ

イベント記述	EVENT_STATE 値	EVENT_TYPE 値	EVENT_OBJECT 値	EVENT_OBJECT_NAME 値
ロックの獲得	IDLE	ACQUIRE	LOCK	ロック名
ロックのエスカレート	EXECUTING	PROCESS	LOCK_ESCALATION	NULL
要求の処理	EXECUTING	PROCESS	REQUEST	NULL
新規要求の待機	IDLE	WAIT	REQUEST	NULL
リモート・パーティションで処理される要求の待機	IDLE	WAIT	REMOTE_REQUEST	NULL
Query Patroller キューの待機	IDLE	WAIT	QP_QUEUE	NULL
WLM threshold キューの待機	IDLE	WAIT	WLM_QUEUE	しきい値名

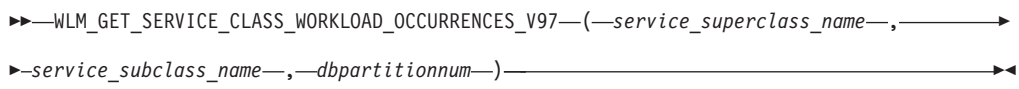
表 242. EVENT_STATE、EVENT_TYPE、EVENT_OBJECT および EVENT_OBJECT_NAME 列値の可能な組み合わせ (続き)

イベント記述	EVENT_STATE 値	EVENT_TYPE 値	EVENT_OBJECT 値	EVENT_OBJECT_NAME 値
ルーチンの処理	EXECUTING	PROCESS	ROUTINE	NULL
索引の再作成	EXECUTING	PROCESS	INDEX_RECREATE	NULL
コンプレッション・ディクショナリーの作成	EXECUTING	PROCESS	COMP_DICT_BLD	NULL
暗黙的な再バインド	EXECUTING	PROCESS	IMPLICIT_REBIND	NULL

WLM_GET_SERVICE_CLASS_WORKLOAD _OCCURRENCES_V97 - ワークロード・オカレンスのリスト

WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES_V97 関数は、特定のパーティション上の指定されたサービス・クラスで実行しているすべてのワークロード・オカレンスのリストを戻します。ワークロード・オカレンスとは、属性がワークロードの定義と一致しており、そのためにワークロードに関連付けられた、またはワークロードに割り当てられた特定のデータベース接続です。

構文



スキーマは SYSPROC です。

表関数パラメーター

service_superclass_name
現在接続されているデータベースのサービス・スーパークラスの名前を指定する、タイプ VARCHAR(128) の入力引数。引数が NULL または空ストリングである場合、他のパラメーターの値と一致する、データベース内のすべてのスーパークラスについてデータが取得されます。

service_subclass_name
ワークロード・オカレンスのターゲット・サービス・サブクラス。このワークロード・オカレンスによってサブミットされる作業は、別のサブクラスにマップまたは再マップされるアクティビティーを除いて、すべてターゲット・サービス・スーパークラスの下のこのサービス・サブクラスで実行されます。

dbpartitionnum
現在接続されているデータベースと同じインスタンス内のパーティションの番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションには -1、すべてのデータベース・パーティションには -2 を指定します。NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES_V97 関数に対する EXECUTE 特権。

例

管理者がどのワークロード・オカレンスがシステム上で全体として実行しているかを調べる場合、*service_superclass_name* および *service_subclass_name* に NULL 値または空ストリングを指定し、 *dbpartitionnum* に -2 を指定した WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES_V97 関数を呼び出すことができます。

```
SELECT SUBSTR(SERVICE_SUPERCLASS_NAME,1,19) AS SUPERCLASS_NAME,
       SUBSTR(SERVICE_SUBCLASS_NAME,1,18) AS SUBCLASS_NAME,
       SUBSTR(CHAR(DBPARTITIONNUM),1,4) AS PART,
       SUBSTR(CHAR(COORD_PARTITION_NUM),1,4) AS COORDPART,
       SUBSTR(CHAR(APPLICATION_HANDLE),1,7) AS APPHNDL,
       SUBSTR(WORKLOAD_NAME,1,22) AS WORKLOAD_NAME,
       SUBSTR(CHAR(WORKLOAD_OCCURRENCE_ID),1,6) AS WLO_ID
FROM TABLE(WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES_V97
      (CAST(NULL AS VARCHAR(128)), CAST(NULL AS VARCHAR(128)), -2))
       AS SCINFO
ORDER BY SUPERCLASS_NAME, SUBCLASS_NAME, PART, APPHNDL,
       WORKLOAD_NAME, WLO_ID
```

システムに 4 つのデータベース・パーティションがあり、現時点で 2 つのワークロードを実行していると想定すると、上記の照会は以下のような結果を生成します。

```
SUPERCLASS_NAME     SUBCLASS_NAME      PART COORDPART ...
------------------- ------------------ ---- --------- ...
SYSDEFAULTMAINTENAN SYSDEFAULTSUBCLASS 0    0         ...
SYSDEFAULTSYSTEMCLA SYSDEFAULTSUBCLASS 0    0         ...
SYSDEFAULTUSERCLASS SYSDEFAULTSUBCLASS 0    0         ...
SYSDEFAULTUSERCLASS SYSDEFAULTSUBCLASS 0    0         ...
SYSDEFAULTUSERCLASS SYSDEFAULTSUBCLASS 1    0         ...
SYSDEFAULTUSERCLASS SYSDEFAULTSUBCLASS 1    0         ...
SYSDEFAULTUSERCLASS SYSDEFAULTSUBCLASS 2    0         ...
SYSDEFAULTUSERCLASS SYSDEFAULTSUBCLASS 2    0         ...
SYSDEFAULTUSERCLASS SYSDEFAULTSUBCLASS 3    0         ...
SYSDEFAULTUSERCLASS SYSDEFAULTSUBCLASS 3    0         ...

... APPHNDL WORKLOAD_NAME          WLO_ID
... ------- ---------------------- ------
... -       -                      -
... -       -                      -
... 1       SYSDEFAULTUSERWORKLOAD 1
... 2       SYSDEFAULTUSERWORKLOAD 2
... 1       SYSDEFAULTUSERWORKLOAD 1
... 2       SYSDEFAULTUSERWORKLOAD 2
... 1       SYSDEFAULTUSERWORKLOAD 1
... 2       SYSDEFAULTUSERWORKLOAD 2
... 1       SYSDEFAULTUSERWORKLOAD 1
... 2       SYSDEFAULTUSERWORKLOAD 2
```

使用上の注意

これらのパラメーターの作用については、論理積 (AND) が取られます。つまり、矛盾する入力パラメーターを指定する (例えば、サービス・スーパークラス SUP_A とサブクラス SUB_B を、SUB_B が SUP_A のサブクラスにならないように指定する) 場合、行は戻されません。

注: ワークロード・オカレンスについて報告される統計 (例えば、coord_act_completed_total) が、対応するワークロード統計と結合されると、ワークロード・オカレンスについて報告される統計が各作業単位の初めにリセットされます。

戻される情報

表 243. WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES_V97 について戻される情報

列名	データ・タイプ	説明
SERVICE_SUPERCLASS_NAME	VARCHAR(128)	このレコードが収集されたサービス・スーパークラスの名前。
SERVICE_SUBCLASS_NAME	VARCHAR(128)	このレコードが収集されたサービス・サブクラスの名前。
DBPARTITIONNUM	SMALLINT	このレコードが収集されたパーティション番号。
COORD_PARTITION_NUM	SMALLINT	指定されたワークロード・オカレンスのコーディネーター・パーティションのパーティション番号。
APPLICATION_HANDLE	BIGINT	システム全体におけるアプリケーションのユニーク ID。単一パーティション・データベースの場合、この ID は 16 ビットのカウンターで構成されます。複数パーティション・データベースの場合、この ID はコーディネーター・パーティション番号と 16 ビットのカウンターを連結したもので構成されます。さらに、この ID はアプリケーションが 2 次接続を行うすべてのパーティションにおいて同じです。
WORKLOAD_NAME	VARCHAR(128)	このレコードが収集されたワークロードの名前。
WORKLOAD_OCCURRENCE_ID	INTEGER	ワークロード・オカレンスの ID。ワークロード・オカレンスがコーディネーター・データベース・パーティション番号およびワークロード名と結合されていなければ、この ID はワークロード・オカレンスを一意的に識別しません。

表 243. WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES_V97 について戻される情報 (続き)

列名	データ・タイプ	説明
WORKLOAD_OCCURRENCE_STATE	VARCHAR(32)	ワークロード・オカレンスの状態。値は、以下のとおりです。 *DECOUPLED* ワークロード・オカレンスには割り当てられたコーディネーター・エージェントがありません (コンセントレーターの場合)。 *DISCONNECTPEND* ワークロード・オカレンスはデータベースから切断中です。 *FORCED* ワークロード・オカレンスはデータベースから強制的に切断されました。 *INTERRUPTED* ワークロード・オカレンスが中断されました。 *QUEUED* ワークロード・オカレンス・コーディネーター・エージェントが、Query Patroller またはワークロード管理キューイングしきい値によってキューに入れられています。パーティション・データベース環境では、この状態は、コーディネーター・エージェントがしきい値チケットを取得するためにカタログ・パーティションに対して RPC を行ったものの、まだ応答を受け取っていないことを示している可能性があります。 *TRANSIENT* ワークロード・オカレンスがまだサービス・スーパークラスにマップされていません。 *UOWEXEC* ワークロード・オカレンスは要求を処理中です。 *UOWWAIT* ワークロード・オカレンスはクライアントからの要求を待機中です。
UOW_ID	INTEGER	このワークロード・オカレンスが開始した作業単位のユニーク ID。
SYSTEM_AUTH_ID	VARCHAR(128)	ワークロード・オカレンスがシステムに挿入されるときに使用したシステム許可 ID。
SESSION_AUTH_ID	VARCHAR(128)	ワークロード・オカレンスがシステムに挿入されるときに使用したセッション許可 ID。

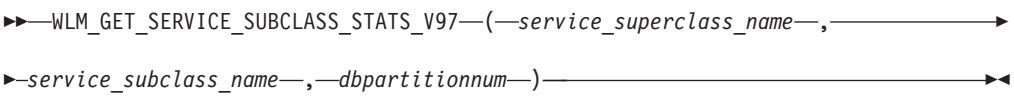
表 243. WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES_V97 について戻される情報 (続き)

列名	データ・タイプ	説明
APPLICATION_NAME	VARCHAR(128)	このワークロード・オカレンスを作成したアプリケーションの名前。
CLIENT_WRKSTNNAME	VARCHAR(255)	このワークロード・オカレンスの CLIENT_WRKSTNNAME 特殊レジスターの現行値。
CLIENT_ACCTNG	VARCHAR(255)	このワークロード・オカレンスの CLIENT_ACCTNG 特殊レジスターの現行値。
CLIENT_USER	VARCHAR(255)	このワークロード・オカレンスの CLIENT_USERID 特殊レジスターの現行値。
CLIENT_APPLNAME	VARCHAR(255)	このワークロード・オカレンスの CLIENT_APPLNAME 特殊レジスターの現行値。
COORD_ACT_COMPLETED_TOTAL	INTEGER	このワークロード・オカレンスの現在の作業単位でこれまでに完了した、任意のネスト・レベルのコーディネーター・アクティビティーの数。この統計は、このワークロード・オカレンスのアクティビティーが完了し、各作業単位の初めにリセットされるたびに更新されます。
COORD_ACT_ABORTED_TOTAL	INTEGER	このワークロード・オカレンスの現在の作業単位でこれまでにアボートしたコーディネーター・アクティビティーの数。この統計は、このワークロード・オカレンスのアクティビティーがアボートされ、各作業単位の初めにリセットされるたびに更新されます。
COORD_ACT_REJECTED_TOTAL	INTEGER	このワークロード・オカレンスの現在の作業単位でこれまでにリジェクトしたコーディネーター・アクティビティーの数。アクティビティーが実行抑制作業アクションまたは予測しきい値のいずれかによって実行を妨げられている場合、そのアクティビティーはリジェクトとしてカウントされます。この統計は、このワークロード・オカレンスのアクティビティーがリジェクトされ、各作業単位の初めにリセットされるたびに更新されます。
CONCURRENT_ACT_TOP	INTEGER	現在の作業単位でこのワークロード・オカレンスについて到達している、実行状態 (アイドルおよび待機中を含む) またはキューに入れられた状態の任意のネスト・レベルの並行アクティビティーの最大数。この統計は、各作業単位の初めにリセットされます。
ADDRESS	VARCHAR(255)	このワークロード・オカレンスを作成した IP アドレスまたはセキュア・ドメイン・ネーム。セキュア・ドメイン・ネームは、IP アドレスに変換されて示されます。

WLM_GET_SERVICE_SUBCLASS_STATS_V97 表関数 - サービス・サブクラスの統計を戻す

WLM_GET_SERVICE_SUBCLASS_STATS_V97 関数は、1 つ以上のサービス・サブクラスの基本統計を戻します。

構文



スキーマは SYSPROC です。

表関数パラメーター

service_superclass_name
現在接続されているデータベースのサービス・スーパークラスの名前を指定する、タイプ VARCHAR(128) の入力引数。引数が NULL または空ストリングである場合、データベース内のすべてのスーパークラスについてデータが取得されます。

service_subclass_name
現在接続されているデータベースのサービス・サブクラスの名前を指定する、タイプ VARCHAR(128) の入力引数。引数が NULL または空ストリングである場合、データベース内のすべてのサブクラスについてデータが取得されます。

dbpartitionnum
現在接続されているデータベースと同じインスタンス内の有効なパーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションには -1、すべてのデータベース・パーティションには -2 を指定します。 NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

WLM_GET_SERVICE_SUBCLASS_STATS_V97 関数に対する EXECUTE 特権。

例

例 1: すべてのアクティビティーは実行前に DB2 サービス・クラスに対してマップされる必要があるため、サービス・クラス統計表関数を使用し、すべてのパーティション上のすべてのサービス・クラスを照会して、システムの全体的な状態をモニターできます。以下の例では、NULL 値が *service_superclass_name* と *service_subclass_name* に渡されてすべてのサービス・クラスの統計を戻し、*dbpartitionnum* には値 -2 が指定されてすべてのパーティションの統計を戻します。

```
SELECT SUBSTR(SERVICE_SUPERCLASS_NAME,1,19) AS SUPERCLASS_NAME,
       SUBSTR(SERVICE_SUBCLASS_NAME,1,18) AS SUBCLASS_NAME,
       SUBSTR(CHAR(DBPARTITIONNUM),1,4) AS PART,
       CAST(COORD_ACT_LIFETIME_AVG / 1000 AS DECIMAL(9,3))
         AS AVGLIFETIME,
       CAST(COORD_ACT_LIFETIME_STDDEV / 1000 AS DECIMAL(9,3))
         AS STDDEVLIFETIME,
       SUBSTR(CAST(LAST_RESET AS VARCHAR(30)),1,16) AS LAST_RESET
```

```
  FROM TABLE(WLM_GET_SERVICE_SUBCLASS_STATS_V97(CAST(NULL AS VARCHAR(128)),
       CAST(NULL AS VARCHAR(128)), -2)) AS SCSTATS
  ORDER BY SUPERCLASS_NAME, SUBCLASS_NAME, PART
```

ステートメントは、以下の出力例で示されているように、アクティビティー存続期間の平均および標準偏差などのサービス・クラス統計を秒単位で戻します。

```
SUPERCLASS_NAME     SUBCLASS_NAME      PART ...
------------------- ------------------ ---- ...
SYSDEFAULTUSERCLASS SYSDEFAULTSUBCLASS 0    ...
SYSDEFAULTUSERCLASS SYSDEFAULTSUBCLASS 1    ...
SYSDEFAULTUSERCLASS SYSDEFAULTSUBCLASS 2    ...
SYSDEFAULTUSERCLASS SYSDEFAULTSUBCLASS 3    ...

... AVGLIFETIME STDDEVLIFETIME LAST_RESET
... ----------- -------------- ----------------
...     691.242         34.322 2006-07-24-11.44
...     644.740         22.124 2006-07-24-11.44
...     612.431         43.347 2006-07-24-11.44
...     593.451         28.329 2006-07-24-11.44
```

例 2: また、同じ表関数が、各パーティション上のサービス・クラスで実行しているコーディネーター・アクティビティーの平均並行性の最高値を示すこともできます。

```
  SELECT SUBSTR(SERVICE_SUPERCLASS_NAME,1,19) AS SUPERCLASS_NAME,
         SUBSTR(SERVICE_SUBCLASS_NAME,1,18) AS SUBCLASS_NAME,
         SUBSTR(CHAR(DBPARTITIONNUM),1,4) AS PART,
         CONCURRENT_ACT_TOP AS ACTTOP,
         CONCURRENT_WLO_TOP AS CONNTOP
  FROM TABLE(WLM_GET_SERVICE_SUBCLASS_STATS_V97(CAST(NULL AS VARCHAR(128)),
         CAST(NULL AS VARCHAR(128)), -2)) AS SCSTATS
  ORDER BY SUPERCLASS_NAME, SUBCLASS_NAME, PART
```

出力例を以下に示します。

```
SUPERCLASS_NAME     SUBCLASS_NAME      PART ACTTOP    CONNTOP
------------------- ------------------ ---- --------- ---------
SYSDEFAULTUSERCLASS SYSDEFAULTSUBCLASS 0           10         7
SYSDEFAULTUSERCLASS SYSDEFAULTSUBCLASS 1            0         0
SYSDEFAULTUSERCLASS SYSDEFAULTSUBCLASS 2            0         0
SYSDEFAULTUSERCLASS SYSDEFAULTSUBCLASS 3            0         0
```

この表関数の出力内のアクティビティーの平均実行時間および回数を調べることで、特定のデータベースの各パーティションの負荷の概要を、的確に知ることができます。この表関数によって戻される高水準ゲージが大きく変わった場合は、システムの負荷の変化を示していることがあります。

例 3: アクティビティーが REMAP ACTIVITY TO アクションでしきい値を使用する場合、そのアクティビティーは、その存続時間中に複数のサービス・クラスで時間を費やすことがあります。以下の例で示されているように、ACTIVITIES_MAPPED_IN 列と ACTIVITIES_MAPPED_OUT 列を調べると、一連のサービス・クラスを通過したアクティビティーの数を判別できます。

```
SELECT SUBSTR(SERVICE_SUPERCLASS_NAME,1,19) AS SUPERCLASS_NAME,
       SUBSTR(SERVICE_SUBCLASS_NAME,1,18) AS SUBCLASS_NAME,
       ACTIVITIES_MAPPED_IN AS MAPPED_IN,
       ACTIVITIES_MAPPED_OUT AS MAPPED_OUT
FROM TABLE(WLM_GET_SERVICE_SUBCLASS_STATS_V97(CAST(NULL AS VARCHAR(128)),
       CAST(NULL AS VARCHAR(128)), -2)) AS SCSTATS
ORDER BY SUPERCLASS_NAME, SUBCLASS_NAME
```

出力例を以下に示します。

```
SUPERCLASS_NAME     SUBCLASS_NAME      MAPPED_IN MAPPED_OUT
------------------- ------------------ --------- ----------
SYSDEFAULTUSERCLASS SYSDEFAULTSUBCLASS         0          0
SUPERCLASS1         SYSDEFAULTSUBCLASS         0          0
SUPERCLASS1         SUBCLASS1                  0          7
SUPERCLASS1         SUBCLASS2                  7          0
```

使用上の注意

一部の統計は、対応するサービス・サブクラスの COLLECT AGGREGATE ACTIVITY DATA および COLLECT AGGREGATE REQUEST DATA パラメーターを NONE 以外の値に設定した場合のみ、戻されます。

WLM_GET_SERVICE_SUBCLASS_STATS_V97 表関数は、サービス・サブクラスごとおよびパーティションごとに 1 行のデータを戻します。この関数は、(パーティション上の) サービス・クラス全体または (1 つ以上のサービス・クラスの) パーティション全体のデータ集約は行いません。ただし、SQL 照会を使用すると、データを集約することができます。

これらのパラメーターの作用については、論理積 (AND) が取られます。つまり、例えばスーパークラス SUPA と、SUPA のサブクラスでないサブクラス SUBB などの競合する入力パラメーターを指定した場合、行は戻されません。

戻される情報

表 244. WLM_GET_SERVICE_SUBCLASS_STATS_V97 について戻される情報

列名	データ・タイプ	説明
SERVICE_SUPERCLASS_NAME	VARCHAR(128)	このレコードが収集されたサービス・スーパークラスの名前。
SERVICE_SUBCLASS_NAME	VARCHAR(128)	このレコードが収集されたサービス・サブクラスの名前。
DBPARTITIONNUM	SMALLINT	このレコードが収集されたパーティション番号。
LAST_RESET	TIMESTAMP	統計が最後にリセットされた時刻。統計のリセットを起動するイベントは 4 つあります。 • WLM_COLLECT_STATS プロシージャーの呼び出し。 • **wlm_collect_int** 構成パラメーター (これにより、収集とリセットが発生します)。 • データベースの再活動化。 • 統計が報告されているサービス・サブクラスの変更、およびその変更のコミット。 LAST_RESET タイム・スタンプが現地時間である。

表 244. WLM_GET_SERVICE_SUBCLASS_STATS_V97 について戻される情報 (続き)

列名	データ・タイプ	説明
COORD_ACT_COMPLETED_TOTAL	BIGINT	最後のリセット以降にサブミットされ、正常に完了したコーディネーター・アクティビティーの合計数。この数は、アクティビティーが完了するたびに更新されます。 異なるサービス・サブクラスにアクティビティーを再マップする場合、そのアクティビティーのカウント対象となるのは、それが完了したサブクラスの合計だけです。
COORD_ACT_ABORTED_TOTAL	BIGINT	最後のリセット以降にサブミットされ、エラーを出して完了したコーディネーター・アクティビティーの合計数。この数は、アクティビティーが異常終了するたびに更新されます。 異なるサービス・サブクラスにアクティビティーを再マップする場合、そのアクティビティーのカウント対象となるのは、それが打ち切ったサブクラスの合計だけです。
COORD_ACT_REJECTED_TOTAL	BIGINT	最後のリセット以降にサブミットされ、実行前にリジェクトされたコーディネーター・アクティビティーの合計数。アクティビティーが実行抑制作業アクションまたは予測しきい値のいずれかによって実行を妨げられている場合、そのアクティビティーはリジェクトとしてカウントされます。この数は、アクティビティーがリジェクトされるたびに更新されます。
CONCURRENT_ACT_TOP	INTEGER	このサービス・サブクラスについて到達している、実行状態 (アイドルおよび待機中を含む) の任意のネスト・レベルの並行アクティビティーの最高数。

表 244. WLM_GET_SERVICE_SUBCLASS_STATS_V97 について戻される情報 (続き)

列名	データ・タイプ	説明
COORD_ACT_LIFETIME_TOP	BIGINT	すべてのネスト・レベルで評価される、コーディネーター・アクティビティー存続期間の最高水準点。サービス・クラスの COLLECT AGGREGATE ACTIVITY DATA パラメーターが NONE に設定される場合、列の値は NULL です。単位はミリ秒です。 サービス・クラスに再マップされるサブクラスが含まれている場合にこの統計を効率的に使用するには、サービス・サブクラスの COORD_ACT_LIFETIME_TOP 最高水準点を、同じ再マップしきい値によって影響を受ける他のサブクラスのこの最高水準点と集約する必要があります。こうした値を集約する必要があるのは、アクティビティーを完了するにはその前に別のサービス・サブクラスにサブクラスを再マップしておく必要があるためです。再マップされる前に他のサービス・サブクラスでアクティビティーが費やす時間はアクティビティーが完了したサービス・クラスでのみカウント対象となります。
COORD_ACT_LIFETIME_AVG	DOUBLE	最後のリセット以降、このサービス・サブクラスに関連付けられたネスト・レベル 0 のコーディネーター・アクティビティーの存続期間の算術平均。内部的に追跡された平均がオーバーフローした場合、値 -2 が戻されます。サービス・クラスの COLLECT AGGREGATE ACTIVITY DATA パラメーターが NONE に設定される場合、列の値は NULL です。単位はミリ秒です。 サービス・サブクラスの COORD_ACT_LIFETIME_AVG 値は、完了前にサブクラスを通過したものの別のサブクラスに再マップされるアクティビティーの影響は受けません。

表 244. WLM_GET_SERVICE_SUBCLASS_STATS_V97 について戻される情報 (続き)

列名	データ・タイプ	説明
COORD_ACT_LIFETIME_STDDEV	DOUBLE	最後のリセット以降、このサービス・サブクラスに関連付けられたネスト・レベル 0 のコーディネーター・アクティビティーの存続期間の標準偏差。サービス・クラスの COLLECT AGGREGATE ACTIVITY DATA パラメーターが NONE に設定される場合、列の値は NULL です。単位はミリ秒です。 この標準偏差はコーディネーター・アクティビティーの存続期間ヒストグラムから計算され、ヒストグラムのサイズがデータに合わせて正しく設定されていない場合は不正確になることがあります。値が最後のヒストグラム bin に入る場合、値 -1 が戻されます。 サービス・サブクラスの COORD_ACT_LIFETIME_STDDEV 値は、完了前にサービス・サブクラスを通過したものの別のサブクラスに再マップされるアクティビティーの影響は受けません。
COORD_ACT_EXEC_TIME_AVG	DOUBLE	最後のリセット以降、このサービス・サブクラスに関連付けられたネスト・レベル 0 のコーディネーター・アクティビティーの実行時間の算術平均。内部的に追跡された平均がオーバーフローした場合、値 -2 が戻されます。サービス・クラスの COLLECT AGGREGATE ACTIVITY DATA パラメーターが NONE に設定される場合、列の値は NULL です。単位はミリ秒です。 サービス・サブクラスの実行時間平均は、完了前にサブクラスを通過したものの別のサブクラスに再マップされるアクティビティーの影響は受けません。

表 244. WLM_GET_SERVICE_SUBCLASS_STATS_V97 について戻される情報 (続き)

列名	データ・タイプ	説明
COORD_ACT_EXEC_TIME_STDDEV	DOUBLE	最後のリセット以降、このサービス・サブクラスに関連付けられたネスト・レベル 0 のコーディネーター・アクティビティーの実行時間の標準偏差。単位はミリ秒です。 この標準偏差はコーディネーター・アクティビティーの実行時間ヒストグラムから計算され、ヒストグラムのサイズがデータに合わせて正しく設定されていない場合は不正確になることがあります。値が最後のヒストグラム bin に入る場合、値 -1 が戻されます。 サービス・サブクラスの実行時間標準偏差は、完了前にサブクラスを通過したものの別のサブクラスに再マップされるアクティビティーの影響は受けません。
COORD_ACT_QUEUE_TIME_AVG	DOUBLE	最後のリセット以降、このサービス・サブクラスに関連付けられたネスト・レベル 0 のコーディネーター・アクティビティーのキュー時間の算術平均。内部的に追跡された平均がオーバーフローした場合、値 -2 が戻されます。サービス・クラスの COLLECT AGGREGATE ACTIVITY DATA パラメーターが NONE に設定される場合、列の値は NULL です。単位はミリ秒です。 キュー時間平均のカウント対象は、アクティビティーがキューに入れられたサービス・サブクラスだけです。
COORD_ACT_QUEUE_TIME_STDDEV	DOUBLE	最後のリセット以降、このサービス・サブクラスに関連付けられたネスト・レベル 0 のコーディネーター・アクティビティーのキュー時間の標準偏差。サービス・クラスの COLLECT AGGREGATE ACTIVITY DATA パラメーターが NONE に設定される場合、列の値は NULL です。単位はミリ秒です。 この標準偏差はコーディネーター・アクティビティーのキュー時間ヒストグラムから計算され、ヒストグラムのサイズがデータに合わせて正しく設定されていない場合は不正確になることがあります。値が最後のヒストグラム bin に入る場合、値 -1 が戻されます。 キュー時間標準偏差のカウント対象は、アクティビティーがキューに入れられたサービス・サブクラスだけです。

表 244. WLM_GET_SERVICE_SUBCLASS_STATS_V97 について戻される情報 (続き)

列名	データ・タイプ	説明
NUM_REQUESTS_ACTIVE	BIGINT	この表関数の実行時にサービス・サブクラスで実行している要求の数。
NUM_REQUESTS_TOTAL	BIGINT	最後のリセット以降、このサービス・サブクラスで実行を終了する要求の数。この終了状態は、アクティビティー内の要求のメンバーシップに関係なく、任意の要求に適用されます。サービス・クラスの COLLECT AGGREGATE ACTIVITY DATA パラメーターが NONE に設定される場合、列の値は NULL です。 サービス・サブクラスの NUM_REQUESTS_TOTAL 値は、サービス・サブクラスを通過したものの、その中で完了しない要求の影響は受けません。
REQUEST_EXEC_TIME_AVG	DOUBLE	最後のリセット以降、このサービス・サブクラスに関連付けられた要求の実行時間の算術平均。単位はミリ秒です。内部的に追跡された平均がオーバーフローした場合、値 -2 が戻されます。このサービス・クラスの COLLECT AGGREGATE REQUEST DATA パラメーターが NONE に設定される場合、この列の値は NULL です。 サービス・サブクラスの実行時間平均は、サブクラスを通過したものの、その中で完了しない要求の影響は受けません。
REQUEST_EXEC_TIME_STDDEV	DOUBLE	最後のリセット以降、このサービス・サブクラスに関連付けられた要求の実行時間の標準偏差。単位はミリ秒です。サービス・クラスの COLLECT AGGREGATE REQUEST DATA パラメーターが NONE に設定される場合、この列の値は NULL です。 この標準偏差は要求実行時間ヒストグラムから計算され、ヒストグラムのサイズがデータに合わせて正しく設定されていない場合は不正確になることがあります。値が最後のヒストグラム bin に入る場合、値 -1 が戻されます。 サービス・サブクラスの実行時間標準偏差は、サブクラスを通過したものの、その中で完了しなかった要求の影響は受けません。

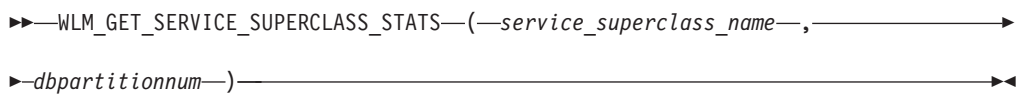
表 244. WLM_GET_SERVICE_SUBCLASS_STATS_V97 について戻される情報 (続き)

列名	データ・タイプ	説明
REQUEST_EXEC_TIME_TOTAL	BIGINT	最後のリセット以降、このサービス・サブクラスに関連付けられた要求の実行時間の合計。単位はミリ秒です。サービス・クラスの COLLECT AGGREGATE REQUEST DATA パラメーターが NONE に設定される場合、この列の値は NULL です。 この合計は要求実行時間ヒストグラムから計算され、ヒストグラムのサイズがデータに合わせて正しく設定されていない場合は不正確になることがあります。値が最後のヒストグラム bin に入る場合、値 -1 が戻されます。 サービス・サブクラスの実行時間合計は、サブクラスを通過したものの、その中で完了しない要求の影響は受けません。
ACT_REMAPPED_IN	BIGINT	最後のリセット以降にしきい値 REMAP ACTIVITY アクションによってこのサービス・サブクラスに再マップされたアクティビティー数。
ACT_REMAPPED_OUT	BIGINT	最後のリセット以降にしきい値 REMAP ACTIVITY アクションによってこのサービス・サブクラスから出て再マップされたアクティビティー数。
CONCURRENT_WLO_TOP	INTEGER	最後のリセット以降、このパーティション上の指定されたワークロードの並行オカレンスの最大数。
UOW_TOTAL_TIME_TOP	BIGINT	作業単位の存続期間の最高水準点 (ミリ秒)。 サービス・クラスの COLLECT AGGREGATE ACTIVITY DATA が NONE に設定されている場合は -1 を戻します。 **注:** この最高水準点の尺度は、ワークロードによって割り当てられたサービス・クラスに関して計算されます。アクティビティーのサービス・クラスを変更する作業アクション・セットによるマッピングは、この最高水準点に影響を与えません。

WLM_GET_SERVICE_SUPERCLASS_STATS - サービス・スーパークラスの統計を戻す

WLM_GET_SERVICE_SUPERCLASS_STATS 関数は、1 つ以上のサービス・スーパークラスの基本統計を戻します。

構文



スキーマは SYSPROC です。

表関数パラメーター

service_superclass_name
現在接続されているデータベースのサービス・スーパークラスの名前を指定する、タイプ VARCHAR(128) の入力引数。引数が NULL または空ストリングである場合、データベース内のすべてのスーパークラスについてデータが取得されます。

dbpartitionnum
現在接続されているデータベースと同じインスタンス内の有効なパーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションには -1、すべてのデータベース・パーティションには -2 を指定します。 NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

WLM_GET_SERVICE_SUPERCLASS_STATS 関数に対する EXECUTE 特権。

例

以下の照会は、システム上のすべてのサービス・スーパークラスのすべての基本統計をすべてのデータベース・パーティションの間で表示します。

```
SELECT SUBSTR(SERVICE_SUPERCLASS_NAME, 1, 26) SERVICE_SUPERCLASS_NAME,
       DBPARTITIONNUM,
       LAST_RESET,
       CONCURRENT_CONNECTION_TOP CONCURRENT_CONN_TOP
FROM TABLE(WLM_GET_SERVICE_SUPERCLASS_STATS('', -2)) as SCSTATS
```

出力例を以下に示します。

```
SERVICE_SUPERCLASS_NAME    DBPARTITIONNUM ...
-------------------------- -------------- ...
SYSDEFAULTSYSTEMCLASS                   0 ...
SYSDEFAULTMAINTENANCECLASS              0 ...
SYSDEFAULTUSERCLASS                     0 ...

... LAST_RESET                 CONCURRENT_CONN_TOP
... -------------------------- -------------------
... 2006-09-05-09.38.44.396788                   0
... 2006-09-05-09.38.44.396795                   0
... 2006-09-05-09.38.44.396796                   1
```

使用上の注意

WLM_GET_SERVICE_SUPERCLASS_STATS 表関数は、サービス・スーパークラスおよびパーティションごとに 1 行のデータを戻します。この関数は、(パーティション上の) サービス・スーパークラス全体または (1 つ以上のサービス・スーパークラスの) パーティション全体のデータ集約は行いません。ただし、上の例で示された SQL 照会を使用すると、データを集約することができます。

戻される情報

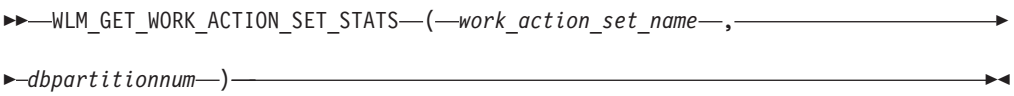
表 245. WLM_GET_SERVICE_SUPERCLASS_STATS について戻される情報

列名	データ・タイプ	説明
SERVICE_SUPERCLASS_NAME	VARCHAR(128)	このレコードが収集されたサービス・スーパークラスの名前。
DBPARTITIONNUM	SMALLINT	このレコードが収集されたパーティション番号。
LAST_RESET	TIMESTAMP	統計が最後にリセットされた時刻。統計のリセットを起動するイベントは 4 つあります。 • WLM_COLLECT_STATS プロシージャーの呼び出し。 • **wlm_collect_int** 構成パラメーター (これにより、収集とリセットが発生します)。 • データベースの再活動化。 • 統計が報告されているサービス・スーパークラスの変更、およびその変更のコミット。 LAST_RESET タイム・スタンプが現地時間である。
CONCURRENT_CONNECTION_TOP	INTEGER	最後のリセット以降のこのクラスの並行コーディネーター接続の最大数。

WLM_GET_WORK_ACTION_SET_STATS - 作業アクション・セット統計を戻す

WLM_GET_WORK_ACTION_SET_STATS 関数は、作業アクション・セットの統計を戻します。

構文



スキーマは SYSPROC です。

表関数パラメーター

work_action_set_name
統計を戻す作業アクション・セットを指定する、タイプ VARCHAR(128) の入力引数。引数が NULL または空ストリングである場合、すべての作業アクション・セットについて統計が戻されます。

dbpartitionnum
現在接続されているデータベースと同じインスタンス内の有効なパーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションには -1、すべてのデータベース・パーティションには -2 を指定します。 NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

WLM_GET_WORK_ACTION_SET_STATS 関数に対する EXECUTE 特権。

例

3 つの作業クラス、ReadClass、WriteClass、および LoadClass があると想定します。 ReadClass に関連した作業アクションと LoadClass に関連した作業アクションはありますが、WriteClass に関連した作業アクションはありません。パーティション 0 上で現在実行されている、またはキューに入れられているアクティビティーの数は次のとおりです。

- ReadClass クラス: 8
- WriteClass クラス: 4
- LoadClass クラス: 2
- 未割り当て: 3

```
SELECT SUBSTR(WORK_ACTION_SET_NAME,1,18) AS WORK_ACTION_SET_NAME,
       SUBSTR(CHAR(DBPARTITIONNUM),1,4) AS PART,
       SUBSTR(WORK_CLASS_NAME,1,15) AS WORK_CLASS_NAME,
       LAST_RESET,
       SUBSTR(CHAR(ACT_TOTAL),1,14) AS ACT_TOTAL
  FROM TABLE(WLM_GET_WORK_ACTION_SET_STATS
       (CAST(NULL AS VARCHAR(128)), -2)) AS WASSTATS
  ORDER BY WORK_ACTION_SET_NAME, WORK_CLASS_NAME, PART
```

出力例を以下に示します。 WriteClass 作業クラスに関連した作業アクションはないため、その作業クラスが該当する 4 つのアクティビティーは、出力でアスタリスク (*) によって示される人工的なクラスでカウントされます。どの作業クラスにも割り当てられなかった 3 つのアクティビティーも、人工的なクラスに組み込まれています。

```
WORK_ACTION_SET_NAME PART WORK_CLASS_NAME LAST_RESET                ACT_TOTAL
-------------------- ---- --------------- ------------------------- --------------
AdminActionSet       0    ReadClass       2005-11-25-18.52.49.343000 8
AdminActionSet       1    ReadClass       2005-11-25-18.52.50.478000 0
AdminActionSet       0    LoadClass       2005-11-25-18.52.49.343000 2
AdminActionSet       1    LoadClass       2005-11-25-18.52.50.478000 0
AdminActionSet       0    *               2005-11-25-18.52.49.343000 7
AdminActionSet       1    *               2005-11-25-18.52.50.478000 0
```

戻される情報

表 246. WLM_GET_WORK_ACTION_SET_STATS について戻される情報

列名	データ・タイプ	説明
WORK_ACTION_SET_NAME	VARCHAR(128)	作業アクション・セットの名前。名前は、作業アクション・セットを使用可能にする場合のみ返されます。
DBPARTITIONNUM	SMALLINT	このレコードが収集されたパーティション番号。

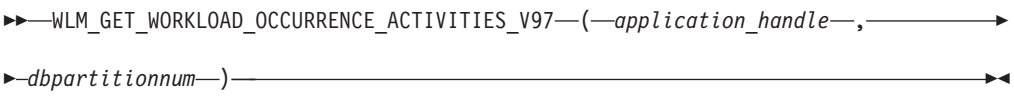
表 246. WLM_GET_WORK_ACTION_SET_STATS について戻される情報 (続き)

列名	データ・タイプ	説明
LAST_RESET	TIMESTAMP	統計が最後にリセットされた時刻。統計のリセットを起動するイベントは 4 つあります。 • WLM_COLLECT_STATS プロシージャーの呼び出し。 • **wlm_collect_int** 構成パラメーター (これにより、収集とリセットが発生します)。 • データベースの再活動化。 • 統計が報告されている作業アクション・セットの変更、およびその変更のコミット。 LAST_RESET タイム・スタンプが現地時間である。
WORK_CLASS_NAME	VARCHAR(128)	指定された作業アクション・セットに関連した作業クラスの名前。作業クラス名は、作業クラスに関連した作業アクションを使用可能にする場合のみ返されます。アスタリスク (*) は、1 つ以上の作業アクションを関連付けたその他の作業クラスに属さない、すべてのアクティビティーをカウントするために作成された人工的な作業クラスを表します。
ACT_TOTAL	BIGINT	WORK_CLASS_NAME で指定される作業クラスに割り当てられた、ネスト・レベルのアクティビティーの数。

WLM_GET_WORKLOAD_OCCURRENCE _ACTIVITIES_V97 - アクティビティーのリストを戻す

WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES_V97 関数は、指定されたパーティション上の指定されたアプリケーションによってサブミットされ、また完了していないすべてのアクティビティーのリストを戻します。

構文



スキーマは SYSPROC です。

表関数パラメーター

application_handle
アクティビティーのリストが戻されるアプリケーション・ハンドルを指定する、タイプ BIGINT の入力引数。引数が NULL である場合、データベース内のすべてのアプリケーションについてデータが取得されます。

dbpartitionnum
現在接続されているデータベースと同じインスタンス内の有効なパーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションには -1、すべてのデータベース・パーティションには -2 を指定します。 NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES_V97 関数に対する EXECUTE 特権。

例

例 1: 認識済みのアプリケーション・ハンドルで現在実行中のアクティビティー

アプリケーション・ハンドルを識別した後、このアプリケーションで現在実行中のすべてのアクティビティーを検索できます。 LIST APPLICATIONS コマンドを使用して判別されるアプリケーション・ハンドルが 1 であるアプリケーションのアクティビティーをリストすることを管理者が望んでいるという場面を、例として考えてみましょう。管理者は以下の照会を実行します。

```
SELECT SUBSTR(CHAR(COORD_PARTITION_NUM),1,5) AS COORD,
     SUBSTR(CHAR(DBPARTITIONNUM),1,4) AS PART,
     SUBSTR(CHAR(UOW_ID),1,5) AS UOWID,
     SUBSTR(CHAR(ACTIVITY_ID),1,5) AS ACTID,
     SUBSTR(CHAR(PARENT_UOW_ID),1,8) AS PARUOWID,
     SUBSTR(CHAR(PARENT_ACTIVITY_ID),1,8) AS PARACTID,
            ACTIVITY_TYPE AS ACTTYPE,
     SUBSTR(CHAR(NESTING_LEVEL),1,7) AS NESTING
FROM TABLE(WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES_V97(1, -2)) AS WLOACTS
ORDER BY PART, UOWID, ACTID
```

照会からの出力例は、次のようになります。

```
COORD PART UOWID ACTID PARUOWID PARACTID ACTTYPE  NESTING
----- ---- ----- ----- -------- -------- -------- -------
0     0    2     3     -        -        CALL     0
0     0    2     5     2        3        READ_DML 1
```

例 2: システムで現在実行中のアクティビティー

以下の照会では、EXECUTABLE_ID で WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES_V97 出力と MON_GET_PKG_CACHE_STMT 出力を結合し、システム上で現在実行中のすべての SQL アクティビティーにステートメント・テキストを提供します。

```
SELECT t.application_handle,
       t.uow_id,
       t.activity_id,
       varchar(p.stmt_text, 256) as stmt_text
FROM table(wlm_get_workload_occurrence_activities_v97(NULL, -1)) as t,
     table(mon_get_pkg_cache_stmt(NULL, NULL, NULL, -1)) as p
WHERE t.executable_id = p.executable_id
```

出力例を以下に示します。

```
APPLICATION_HANDLE   UOW_ID      ACTIVITY_ID    STMT_TEXT
-------------------- ----------- -------------- ------------------------------
1                    1           1              SELECT * FROM SYSCAT.TABLES
47                   1           36             INSERT INTO T1 VALUES(123)
```

戻される情報

表 247. WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES_V97 によって戻される情報

列名	データ・タイプ	説明
APPLICATION_HANDLE	BIGINT	システム全体におけるアプリケーションのユニーク ID。単一パーティション・データベースの場合、この ID は 16 ビットのカウンターで構成されます。複数パーティション・データベースの場合、この ID はコーディネーター・パーティション番号と 16 ビットのカウンターを連結したもので構成されます。さらに、この ID はアプリケーションが 2 次接続を行うすべてのパーティションにおいて同じです。
DBPARTITIONNUM	SMALLINT	このレコードが収集されたパーティション番号。
COORD_PARTITION_NUM	SMALLINT	アクティビティーのコーディネーター・パーティション番号。
LOCAL_START_TIME	TIMESTAMP	アクティビティーがパーティションで作業を開始した現地時間。アクティビティーがシステムに入ったが、キューに入れられており、実行を開始していない場合は、この列の値は NULL です。
UOW_ID	INTEGER	このアクティビティーが開始された元の作業単位のユニーク ID。
ACTIVITY_ID	INTEGER	作業単位内の固有のアクティビティー ID。
PARENT_UOW_ID	INTEGER	アクティビティーの親アクティビティーが開始された元の作業単位のユニーク ID。アクティビティーに親アクティビティーがない場合、またはそれがリモート・パーティションにある場合は、この列の値は NULL です。
PARENT_ACTIVITY_ID	INTEGER	親のアクティビティー ID が ACTIVITY_ID によって指定される、作業単位内のアクティビティーのユニーク ID。アクティビティーに親アクティビティーがない場合、またはそれがリモート・パーティションにある場合は、この列の値は NULL です。

表 247. WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES_V97 によって戻される情報 (続き)

列名	データ・タイプ	説明
ACTIVITY_STATE	VARCHAR(32)	アクティビティーの状態。可能な値は、以下のとおりです。 *CANCEL_PENDING* アクティビティーの要求をアクティブに実行しているエージェントがないため、アクティビティーは取り消されました。次に要求がアクティビティーの一部としてサブミットされるときに、そのアクティビティーは取り消され、SQL4725N エラーが生成されます。 *EXECUTING* エージェントはアクティビティーの要求をアクティブに実行しています。 *IDLE* アクティビティーの要求をアクティブに処理しているエージェントがありません。 *INITIALIZING* アクティビティーはサブミットされましたが、まだ実行を開始していません。初期化状態中に、予測しきい値がアクティビティーに適用され、アクティビティーが実行を許可されるかどうかを判別します。

表 247. *WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES_V97* によって戻される情報 (続き)

列名	データ・タイプ	説明
ACTIVITY_STATE (続き)	VARCHAR(32)	アクティビティーの状態。可能な値は、以下のとおりです。 *QP_CANCEL_PENDING* アクティビティーが WLM_CANCEL_ACTIVITY プロシージャーではなく、Query Patroller によって取り消された点を除いて、この状態は、CANCEL_PENDING 状態と同じです。 *QP_QUEUED* アクティビティーは Query Patroller によってキューに入れられます。 *QUEUED* アクティビティーが、ワークロード管理キューイングしきい値によってキューに入れられています。パーティション・データベース環境では、この状態は、コーディネーター・エージェントがカタログ・パーティションに対する RPC を行ってしきい値チケットを取得したものの、まだ応答を受け取っていないことを示す場合があります。この状態は、アクティビティーがワークロード管理キューイングしきい値によってキューに入れられているか、それほど時間が経過していない場合はアクティビティーがそのチケットを取得する処理中であることを示すことがあります。アクティビティーがキューに入れられているかどうかについての詳細を知るために、どのエージェントがアクティビティーで作業しているかを判別し、カタログ・パーティションにあるオブジェクトの EVENT_OBJECT の値が WLM_QUEUE の値であるかどうかを調べてください。 *TERMINATING* アクティビティーは実行を完了し、システムから除去されています。

表 247. WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES_V97 によって戻される情報 (続き)

列名	データ・タイプ	説明
ACTIVITY_TYPE	VARCHAR(32)	アクティビティー・タイプ。可能な値は、以下のとおりです。 • CALL • DDL • LOAD • OTHER • READ_DML • WRITE_DML 各アクティビティー・タイプに関連付けられた SQL ステートメントのさまざまなタイプの説明については、「ワークロード・マネージャー *ガイドおよびリファレンス* 」の『ワーク・クラスによる作業タイプの識別』を参照してください。
NESTING_LEVEL	INTEGER	このアクティビティーが一番上の親アクティビティー内でネストされる深さ。
INVOCATION_ID	INTEGER	1 つのルーチン呼び出しを、作業単位内の同じネスト・レベルにある他の呼び出しと区別するための ID。これは作業単位内の特定のネスト・レベルにおいて固有です。
ROUTINE_ID	INTEGER	ルーチンのユニーク ID。
UTILITY_ID	INTEGER	以下の値のいずれか。 • アクティビティーがユーティリティーの場合、値はユーティリティーの ID です。 • アクティビティーがユーティリティーでない場合、値は NULL です。
SERVICE_CLASS_ID	INTEGER	このアクティビティーが属するサービス・クラスのユニーク ID。
DATABASE_WORK_ACTION_SET_ID	INTEGER	以下の値のいずれか。 • このアクティビティーがデータベース有効範囲の作業クラスに分類されている場合、値はこの作業クラスがメンバーとなっている作業クラス・セットの ID です。 • このアクティビティーがデータベース有効範囲の作業クラスに分類されていない場合、値は NULL です。
DATABASE_WORK_CLASS_ID	INTEGER	以下の値のいずれか。 • このアクティビティーがデータベース有効範囲の作業クラスに分類されている場合、値は作業クラスの ID です。 • このアクティビティーがデータベース有効範囲の作業クラスに分類されていない場合、値は NULL です。

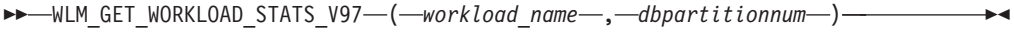
表 247. WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES_V97 によって戻される情報 (続き)

列名	データ・タイプ	説明
SERVICE_CLASS_WORK_ACTION_SET_ID	INTEGER	以下の値のいずれか。 • このアクティビティーがサービス・クラス有効範囲の作業クラスに分類されている場合、値は作業クラスが属する作業クラス・セットに関連付けられた作業アクション・セットの ID です。 • このアクティビティーがサービス・クラス有効範囲の作業クラスに分類されていない場合、値は NULL です。
SERVICE_CLASS_WORK_CLASS_ID	INTEGER	以下の値のいずれか。 • このアクティビティーがサービス・クラス有効範囲の作業クラスに分類されている場合、この値はこのアクティビティーに割り当てられた作業クラスの ID です。 • このアクティビティーがサービス・クラス有効範囲の作業クラスに分類されていない場合、値は NULL です。
EXECUTABLE_ID	VARCHAR(32) FOR BIT DATA	データ・サーバー上で生成される不透明なバイナリー・トークンで、セクションを固有に識別します。実行可能 ID を異なるモニター・インターフェースへの入力として使用し、セクションについてのデータを取得できます。 LOAD など、非 SQL アクティビティーの場合には、NULL 値が戻されます。
TOTAL_CPU_TIME	BIGINT	total_cpu_time - 合計 CPU 時間
ROWS_READ	BIGINT	rows_read - 読み取り行数
ROWS_RETURNED	BIGINT	rows_returned - 戻り行数
QUERY_COST_ESTIMATE	BIGINT	query_cost_estimate - 照会コストの見積もり
DIRECT_READS	BIGINT	direct_reads - データベースからの直接読み取り
DIRECT_WRITES	BIGINT	direct_writes - データベースへの直接書き込み

WLM_GET_WORKLOAD_STATS_V97 表関数 - ワークロード統計を戻す

WLM_GET_WORKLOAD_STATS_V97 関数は、ワークロード名とデータベース・パーティション番号のすべての組み合わせについてワークロード統計の 1 行を返します。

構文



スキーマは SYSPROC です。

表関数パラメーター

workload_name
統計が戻されるワークロードを指定する、タイプ VARCHAR(128) の入力引数。引数が NULL または空ストリングである場合、すべてのワークロードについて統計が戻されます。

dbpartitionnum
現在接続されているデータベースと同じインスタンス内のパーティションの番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションには -1、すべてのデータベース・パーティションには -2 を指定します。NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

WLM_GET_WORKLOAD_STATS_V97 関数に対する EXECUTE 特権。

例

以下の照会は、ワークロードの統計を表示します。

```
SELECT SUBSTR(WORKLOAD_NAME,1,18) AS WL_DEF_NAME,
       SUBSTR(CHAR(DBPARTITIONNUM),1,4) AS PART,
       COORD_ACT_LIFETIME_TOP,
       COORD_ACT_LIFETIME_AVG,
       COORD_ACT_LIFETIME_STDDEV
FROM TABLE(WLM_GET_WORKLOAD_STATS_V97(CAST(NULL AS VARCHAR(128)), -2)) AS WLSTATS
ORDER BY WL_DEF_NAME, PART
```

照会からの出力例は、次のようになります。

```
WL_DEF_NAME        PART COORD_ACT_LIFETIME_TOP ...
------------------ ---- ---------------------- ...
SYSDEFAULTADMWORKL 0                       -1 ...
SYSDEFAULTUSERWORK 0                       -1 ...
WL1                0                        2 ...

... COORD_ACT_LIFETIME_AVG   COORD_ACT_LIFETIME_STDDEV
... ------------------------ ------------------------
... -1.00000000000000E+000    -1.00000000000000E+000
... -1.00000000000000E+000    -1.00000000000000E+000
... +2.56000000000000E+000    +6.00000000000001E-002
```

使用上の注意

この関数は、ワークロード、パーティション、またはサービス・クラスを越えたデータ集約は行いません。ただし、SQL 照会を使用すると、データを集約することができます。

戻される情報

表 248. WLM_GET_WORKLOAD_STATS_V97 によって戻される情報

列名	データ・タイプ	説明
WORKLOAD_NAME	VARCHAR(128)	このレコードが収集されたワークロードの名前。
DBPARTITIONNUM	SMALLINT	このレコードが収集されたパーティション番号。

表 248. *WLM_GET_WORKLOAD_STATS_V97* によって戻される情報 (続き)

列名	データ・タイプ	説明
LAST_RESET	TIMESTAMP	統計が最後にリセットされた時刻。統計のリセットを起動するイベントは 4 つあります。 • WLM_COLLECT_STATS プロシージャーの呼び出し。 • **wlm_collect_int** 構成パラメーター (これにより、収集とリセットが発生します)。 • データベースの再活動化。 • 統計が報告されているワークロードの変更、およびその変更のコミット。 LAST_RESET タイム・スタンプがローカル時刻である。
CONCURRENT_WLO_TOP	INTEGER	最後のリセット以降、このパーティション上の指定されたワークロードの並行オカレンスの最大数。
CONCURRENT_WLO_ACT_TOP	INTEGER	最後のリセット以降、このワークロードのいずれかのオカレンスで到達した、実行状態 (アイドルおよび待機中を含む) またはキューに入れられた状態の並行アクティビティー (コーディネーターとネストの両方) の最大数。列の値は、各ワークロード・オカレンスによってその作業単位の終わりに更新されます。
COORD_ACT_COMPLETED_TOTAL	BIGINT	最後のリセット以降に完了したこのワークロードのいずれかのオカレンスに割り当てられた、任意のネスト・レベルのコーディネーター・アクティビティーの合計数。この列の値は、各ワークロード・オカレンスによってその作業単位の終わりに更新されます。
COORD_ACT_ABORTED_TOTAL	BIGINT	最後のリセット以降で完了前にアボートされたこのワークロードのいずれかのオカレンスに割り当てられた、任意のネスト・レベルのコーディネーター・アクティビティーの合計数。この列の値は、各ワークロード・オカレンスによってその作業単位の終わりに更新されます。

表 248. WLM_GET_WORKLOAD_STATS_V97 によって戻される情報 (続き)

列名	データ・タイプ	説明
COORD_ACT_REJECTED_TOTAL	BIGINT	最後のリセット以降で実行前にリジェクトされたこのワークロードのいずれかのオカレンスに割り当てられた、任意のネスト・レベルのコーディネーター・アクティビティーの合計数。この列の値は、各ワークロード・オカレンスによってその作業単位の終わりに更新されます。 アクティビティーが実行抑制作業アクションまたは予測しきい値のいずれかによって実行を妨げられている場合、そのアクティビティーはリジェクトとしてカウントされます。WLM_GET_SERVICE_SUBCLASS_STATS_V97 関数の同じ名前の列とは異なり、この WLM_GET_WORKLOAD_STATS_V97 列にはアクティビティーがサービス・クラスに割り当てられる前に発生するリジェクトの数も含まれます。例えば、アクティビティーが ConcurrentWorkloadOccurrences しきい値に違反すると、そうしたリジェクトが発生します。
WLO_COMPLETED_TOTAL	BIGINT	最後のリセット以降、完了するワークロード・オカレンスの数。
COORD_ACT_LIFETIME_TOP	BIGINT	すべてのネスト・レベルで収集される、コーディネーター・アクティビティー存続期間の最高水準点。単位はミリ秒です。サービス・クラスの COLLECT AGGREGATE ACTIVITY DATA パラメーターが NONE に設定される場合、列の値は NULL です。
COORD_ACT_LIFETIME_AVG	DOUBLE	このワークロードに関連付けられたネスト・レベル 0 の完了または打ち切られたコーディネーター・アクティビティーの存続期間の算術平均。単位はミリ秒です。内部的に追跡された平均がオーバーフローした場合、値 -2 が戻されます。 ワークロードの COLLECT AGGREGATE ACTIVITY DATA パラメーターが NONE に設定される場合、列の値は NULL です。
COORD_ACT_LIFETIME_STDDEV	DOUBLE	このワークロードに関連付けられたネスト・レベル 0 の完了または打ち切られたコーディネーター・アクティビティーの存続期間の標準偏差。単位はミリ秒です。ワークロードの COLLECT AGGREGATE ACTIVITY DATA パラメーターが NONE に設定される場合、列の値は NULL です。この標準偏差はコーディネーター・アクティビティーの存続期間ヒストグラムから計算され、ヒストグラムのサイズがデータに合わせて正しく設定されていない場合は不正確になることがあります。値が最後のヒストグラム bin に入る場合、値 -1 が戻されます。

表 248. WLM_GET_WORKLOAD_STATS_V97 によって戻される情報 (続き)

列名	データ・タイプ	説明
COORD_ACT_EXEC_TIME_AVG	DOUBLE	このワークロードに関連付けられたネスト・レベル 0 の完了または打ち切られたコーディネーター・アクティビティーの実行時間の算術平均。単位はミリ秒です。内部的に追跡された平均がオーバーフローした場合、値 -2 が戻されます。 ワークロードの COLLECT AGGREGATE ACTIVITY DATA パラメーターが NONE に設定される場合、列の値は NULL です。
COORD_ACT_EXEC_TIME_STDDEV	DOUBLE	このワークロードに関連付けられたネスト・レベル 0 の完了または打ち切られたコーディネーター・アクティビティーの実行時間の標準偏差。単位はミリ秒です。この標準偏差はコーディネーター・アクティビティーの実行時間ヒストグラムから計算され、ヒストグラムのサイズがデータに合わせて正しく設定されていない場合は不正確になることがあります。 値が最後のヒストグラム bin に入る場合、値 -1 が戻されます。 ワークロードの COLLECT AGGREGATE ACTIVITY DATA パラメーターが NONE に設定される場合、列の値は NULL です。
COORD_ACT_QUEUE_TIME_AVG	DOUBLE	このワークロードに関連付けられたネスト・レベル 0 の完了または打ち切られたコーディネーター・アクティビティーのキュー時間の算術平均。単位はミリ秒です。内部的に追跡された平均がオーバーフローした場合、値 -2 が戻されます。サービス・クラスの COLLECT AGGREGATE ACTIVITY DATA パラメーターが NONE に設定される場合、列の値は NULL です。
COORD_ACT_QUEUE_TIME_STDDEV	DOUBLE	このワークロードに関連付けられたネスト・レベル 0 の完了または打ち切られたコーディネーター・アクティビティーのキュー時間の標準偏差。単位はミリ秒です。ワークロードの COLLECT AGGREGATE ACTIVITY DATA パラメーターが NONE に設定される場合、列の値は NULL です。この標準偏差はコーディネーター・アクティビティーのキュー時間ヒストグラムから計算され、ヒストグラムのサイズがデータに合わせて正しく設定されていない場合は不正確になることがあります。値が最後のヒストグラム bin に入る場合、値 -1 が戻されます。
UOW_TOTAL_TIME_TOP	BIGINT	作業単位の存続期間の最高水準点 (ミリ秒)。 ワークロードに関する COLLECT AGGREGATE ACTIVITY DATA が NONE に設定されている場合は -1 を戻します。

WLM_SET_CLIENT_INFO プロシージャー - クライアント情報設定

WLM_SET_CLIENT_INFO プロシージャーは、DB2 サーバーでの現行接続に関連付けられたクライアント情報を設定します。

このプロシージャーを使用することにより、クライアントのユーザー ID、アプリケーション名、ワークステーション名、アカウンティング情報、またはワークロード情報を DB2 サーバーで設定できます。このプロシージャーを呼び出すと、この接続に関する、関連するトランザクション・プロセッサー (TP) モニターのクライアント情報フィールドおよび特殊レジスター設定の保管された値が変更されます。

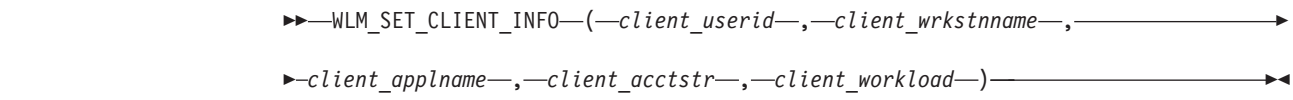
クライアント情報フィールドは、現在接続を使用しているアプリケーションまたはユーザーの ID を判別するために、DB2 サーバーで使用されます。接続用のクライアント情報フィールドは、DB2 のワークロード評価中に考慮され、この接続用に生成される DB2 監査レコードまたはアプリケーション・スナップショットにも表示されます。

sqleseti API とは異なり、このプロシージャーは、クライアントでクライアント情報を設定するのではなく、対応するクライアント属性を DB2 サーバーで設定します。そのため、このプロシージャーを使用して DB2 サーバーに設定されたクライアント情報を照会するために sqleqry API を使用することはできません。

このプロシージャーで提供されたデータ値は、関連した TP モニターのフィールドまたは特殊レジスターに保管される前に、適切なデータベースのコード・ページに変換されます。データベースのコード・ページへの変換後に最大サポート・サイズを超えるデータ値は、サーバーに保管される前に切り捨てられます。切り捨てられた値は、それら保管された値が照会されるときに TP モニターのフィールドおよび特殊レジスターの両方によって返されます。

WLM_SET_CLIENT_INFO プロシージャーはトランザクションの制御下にはなく、このプロシージャーにより加えられたクライアント情報変更は、作業単位のコミットまたはロールバックとは無関係です。ただし、各アプリケーションの次の作業単位の開始時にワークロードの再評価が行われるので、クライアント情報変更を有効にするには、COMMIT または ROLLBACK のいずれかのステートメントを発行する必要があります。

構文



スキーマは SYSPROC です。

プロシージャー・パラメーター

client_userid
クライアントのユーザー ID を指定する、タイプ VARCHAR(255) の入力引数。 NULL を指定する場合、値は変わりません。空ストリング (デフォルト値) を指定する場合、クライアントのユーザー ID はデフォルト値であるブランクにリセットされます。

client_wrkstnname
クライアントのワークステーション名を指定する、タイプ VARCHAR(255) の入力引数。 NULL を指定する場合、値は変わりません。空ストリング (デフォルト値) を指定する場合、クライアントのワークステーション名はデフォルト値であるブランクにリセットされます。

client_applname
クライアントのアプリケーション名を指定する、タイプ VARCHAR(255) の入力引数。 NULL を指定する場合、値は変わりません。空ストリング (デフォルト値) を指定する場合、クライアントのアプリケーション名はデフォルト値であるブランクにリセットされます。

client_acctstr
クライアント・アカウンティング・ストリングを指定する、タイプ VARCHAR(255) の入力引数。 NULL を指定する場合、値は変わりません。空ストリング (デフォルト値) を指定する場合、クライアント・アカウンティング・ストリングはデフォルト値であるブランクにリセットされます。

client_workload
クライアントのワークロードの割り当てを指定する、タイプ VARCHAR(255) の入力引数。 NULL を指定する場合、値は変わりません。値は、以下のとおりです。

SYSDEFAULTADMWORKLOAD
データベース接続が SYSDEFAULTADMWORKLOAD に割り当てられることを指定します。これにより、ACCESSCTRL、DATAACCESS、DBADM、SECADM、または WLMADM 権限を持つユーザーは通常のワークロード評価を迂回することができます。

AUTOMATIC
サーバーが自動的に実行するワークロード計算に選ばれているワークロードに、データベース接続を割り当てるよう指定します。

注: *client_workload* 引数は大文字小文字を区別します。

許可

WLM_SET_CLIENT_INFO プロシージャーに対する EXECUTE 特権。

例

以下のプロシージャー呼び出しは、クライアントのユーザー ID、ワークステーション名、アプリケーション名、アカウンティング・ストリング、およびワークロード割り当てモードを設定します。

```
CALL SYSPROC.WLM_SET_CLIENT_INFO('db2user', 'machine.torolab.ibm.com',
   'auditor', 'Accounting department', 'AUTOMATIC')
```

以下のプロシージャー呼び出しは、クライアントのユーザー ID を db2user2 に設定し、他のクライアント属性は設定しません。

```
CALL SYSPROC.WLM_SET_CLIENT_INFO('db2user2', NULL, NULL, NULL, NULL)
```

以下のプロシージャー呼び出しは、クライアントのユーザー ID をブランクにリセットし、他のクライアント属性の値を変更しません。

```
CALL SYSPROC.WLM_SET_CLIENT_INFO('', NULL, NULL, NULL, NULL)
```

WLM_SET_CONN_ENV - アクティビティー・データの収集とセクション実行時統計の測定の有効化

WLM_SET_CONN_ENV プロシージャーは、特定の接続でのアクティビティー・データの収集とセクション実行時統計 (セクションの実行中に測定されるランタイム統計) の測定を有効にします。

WLM_SET_CONN_ENV プロシージャーによる設定が適用されると、別の WLM_SET_CONN_ENV プロシージャーの呼び出しによって明示的に上書きされるか、接続が閉じられるまで適用され続けます。接続が閉じられた後に同じアプリケーション・ハンドルを再利用する新規接続がある場合、その接続でそのアプリケーション・ハンドルを使用するために、前の接続の設定が継承されることはありません。

注: フィックスパック 2 より前に作成されたバージョン 9.7 のデータベースを使用している場合、このルーチンを実行するには、既に db2updv97 コマンドを実行済みでなければなりません。バージョン 9.7 より前に作成されたデータベースを使用している場合は、db2updv97 コマンドを実行する必要はありません (データベースのアップグレードによって自動的にカタログ更新が実行されるため)。バージョン 9.7 にダウングレードすると、このルーチンは機能しなくなります。

スキーマは SYSPROC です。

許可

WLM_SET_CONN_ENV プロシージャーに対する EXECUTE 特権。

パラメーター

application_handle
接続環境が変更されるアプリケーション・ハンドルを指定する、タイプ BIGINT の入力引数。指定されるアプリケーション・ハンドルは、既存のアプリケーションを参照する必要があります (参照しない場合、SQLSTATE 5U002 が戻されます)。 NULL 値を使用することで、変更される環境を持つ接続が、プロシージャーが呼び出された接続であることを示すことができます。

settings

1 つ以上のモニター設定の指定を可能にする、タイプ CLOB(8K) の入力引数。設定は、次の形式の名前と値の対として指定されます。

<setting name tag>value</setting name tag>

設定は、それぞれ最大で 1 回指定できます。設定名には、大/小文字の区別があります。設定の変更は、次回ステートメントを実行するときに有効になります。既に進行中のステートメントに対しては有効になりません。

使用可能な設定名タグは次のとおりです。

- '<collectactdata>*value*</collectactdata>'

  アクティビティー・イベント・モニターによって収集するアクティビティー・データを指定します。有効な値は以下のとおりです (単語の間のスペースのバリエーションはサポートされています)。

値	説明
NONE	アクティビティー・データを収集しません。
WITHOUT DETAILS	各アクティビティーに関するデータを、アクティビティーの実行が完了したときに、任意のアクティブなアクティビティー・イベント・モニターに送信します。ステートメント、コンパイル環境、およびセクション環境のデータに関する詳細は送信されません。
WITH DETAILS	任意のアクティブなアクティビティーにステートメントおよびコンパイル環境のデータが含まれる場合、それを該当するアクティビティーのイベント・モニターへ送信します。セクションの環境データは送信されません。
WITH DETAILS, SECTION	ステートメント、コンパイル環境、セクション環境データ、セクション実行時統計を、それらが含まれるアクティビティー用のアクティブなアクティビティー・イベント・モニターに送信します。 収集されるセクション実行時統計で、*collectsectionactuals* を BASE に設定するか、**section_actuals** データベース構成パラメーターを BASE に設定する必要があります。セクション実行時統計は、アクティビティー・データが収集されるすべてのパーティションで収集されます。
WITH DETAILS, SECTION AND VALUES	ステートメント、コンパイル環境、セクション環境データ、セクション実行時統計、および入力データ値を、それらが含まれるアクティビティー用のアクティブなアクティビティー・イベント・モニターに送信します。 収集されるセクション実行時統計で、*collectsectionactuals* を BASE に設定するか、**section_actuals** データベース構成パラメーターを BASE に設定する必要があります。セクション実行時統計は、アクティビティー・データが収集されるすべてのパーティションで収集されます。

値	説明
WITH DETAILS AND VALUES	ステートメント、コンパイル環境、および入力データ値を、それらが含まれるアクティビティー用のアクティブなアクティビティー・イベント・モニターに送信します。セクションの環境データは送信されません。

- '<collectactpartition>COORDINATOR</collectactpartition>' または '<collectactpartition>ALL</collectactpartition>'

  アクティビティー・データが収集される場所 (コーディネーター・パーティションのみか、またはすべてのパーティションか) を指定します。*collectactpartition* を指定しない場合、接続は *collectactpartition* のこれまでの値 (デフォルトは COORDINATOR) を維持します。
- '<collectsectionactuals>NONE</collectsectionactuals>' または '<collectsectionactuals>BASE</collectsectionactuals>'

  collectsectionactuals が BASE に設定される場合、セクション実行時統計は収集されます。

例

以下の例はどちらも、コーディネーター・パーティションで、現行接続の、詳細を伴わないアクティビティーの収集を有効にします。

```
CALL WLM_SET_CONN_ENV(NULL,'<collectactdata>WITHOUT DETAILS</collectactdata>')

CALL WLM_SET_CONN_ENV(NULL,'<collectactdata>WITHOUT
   DETAILS</collectactdata><collectactpartition>COORDINATOR
   </collectactpartition>')
```

次の例は、すべてのパーティションで、現行接続の、セクションの環境データとセクション実行時統計は伴うがデータ値は伴わないアクティビティー・データの収集を有効にします。

```
CALL WLM_SET_CONN_ENV(NULL, '<collectactdata>WITH DETAILS, SECTION
   </collectactdata><collectactpartition>ALL</collectactpartition>')
```

以下の例は、現行接続のアクティビティー・データの収集を無効にします。

```
CALL WLM_SET_CONN_ENV(NULL, '<collectactdata>NONE</collectactdata>')
```

使用上の注意

collectactdata 設定は、アクティビティー・データの収集を接続レベルでのみ制御します。アクティビティーには、複数のアクティビティー・データの収集制御が適用されていることがあります。例えば、接続は COLLECT ACTIVITY DATA 節が適用されたサービス・クラスにマップされる可能性があります。複数のアクティビティー・データの収集制御が適用されている場合、すべての設定を組み合わせたものが実際の設定になります。例えば、

1. 接続レベルの制御は、詳細を伴わない (WITHOUT DETAILS) アクティビティー・データである。
2. ワークロードの制御は、なし (NONE) である。

3. サービス・クラスの制御は、詳細および値を伴う (WITH DETAILS AND VALUES) アクティビティー・データである。
4. アクティビティーが実行を完了したときに、アクティビティーおよびデータ値に関する詳細な情報がすべてのアクティブなイベント・モニターに送信される。

設定が WLM_SET_CONN_ENV プロシージャーの入力で指定されない場合、接続環境で変更されません。

セクション実行時統計の収集の有効な設定は、*collectsectionactuals* 設定と **section_actuals** データベース構成パラメーターを組み合わせたものです。例えば、*collectsectionactuals* が BASE に設定され、**section_actuals** データベース構成パラメーターの値が NONE に設定された場合、セクション実行時統計の収集の有効な設定は、BASE になります (逆も当てはまります)。*collectsectionactuals* が BASE に設定される場合には、自動統計プロファイル (**auto_stats_prof** データベース構成パラメーターを使って有効にされる) を使用しないでください (使用すると、警告 SQLSTATE 01HN2 が返されます)。

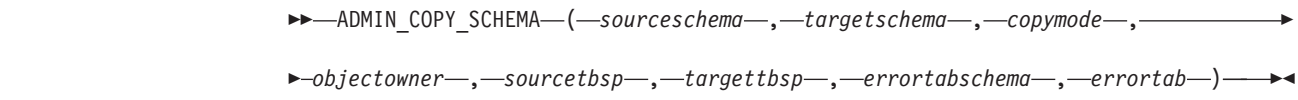
アクティビティー・データおよびセクション実行時統計が収集されているとき (*collectactdata* が NONE 以外の値に設定されているとき) に、接続に対して自動クライアント・リルートを実行することはできません。

第 21 章 その他のルーチンおよびビュー

ADMIN_COPY_SCHEMA プロシージャー - 特定のスキーマとそのオブジェクトのコピー

ADMIN_COPY_SCHEMA プロシージャーは、特定のスキーマと、その中に含まれているすべてのオブジェクトをコピーするために使用されます。新しいターゲット・スキーマ・オブジェクトは、ソース・スキーマ内のオブジェクトと同じオブジェクト名を使って作成されますが、ターゲット・スキーマの修飾子が付きます。ADMIN_COPY_SCHEMA プロシージャーは表をコピーするために使用することができます。元の表のデータは、含めることも除くことも可能です。

構文



スキーマは SYSPROC です。

プロシージャー・パラメーター

sourceschema
コピーされるオブジェクトが属しているスキーマの名前を指定する、タイプ VARCHAR(128) の入力引数。この名前は大文字小文字が区別されます。

targetschema
コピーされたオブジェクトの作成先となる固有のスキーマ名を指定する、タイプ VARCHAR(128) の入力引数。この名前は大文字小文字が区別されます。そのスキーマ名が既に存在する場合、プロシージャー呼び出しは失敗し、プロシージャーを呼び出す前にそのスキーマを削除する必要があることを示すメッセージが戻されます。

copymode
コピー操作のモードを指定する、タイプ VARCHAR(128) の入力引数。有効なオプションは以下のとおりです。

- 'DDL': ソース・スキーマの、サポートされているすべてのオブジェクトの空のコピーを作成します。
- 'COPY': ソース・スキーマのすべてのオブジェクトの空のコピーを作成し、それから各ターゲット・スキーマ表にデータをロードします。ロードは 'NONRECOVERABLE' モードで行われます。ADMIN_COPY_SCHEMA を呼び出した後でバックアップを取る必要があります。そうしなければ、リカバリーの後、コピーされた表がアクセス不能になります。
- 'COPYNO': ソース・スキーマのすべてのオブジェクトの空のコピーを作成し、それから各ターゲット・スキーマ表にデータをロードします。ロードは 'COPYNO' モードで行われます。

© Copyright IBM Corp. 2006, 2010

注: *copymode* が「COPY」または「COPYNO」の場合、完全修飾ファイル名 (たとえば「COPYNO /home/mckeough/loadoutput」) を *copymode* パラメーター値とともに指定できます。パスが渡されると、指定のファイルにロード・メッセージが記録されます。ファイル名は、インスタンスでの fenced ルーチン呼び出しで使用されるユーザー ID が書き込めるものでなければなりません。パスが指定されないと、ロード・メッセージ・ファイルは廃棄されます (デフォルトの振る舞い)。

objectowner
コピーされたオブジェクトの所有者として使用される許可 ID を指定する、タイプ VARCHAR(128) の入力引数。NULL の場合、所有者は、コピー操作を実行するユーザーの許可 ID になります。

sourcetbsp
コピー用のソース表スペースのコンマ区切りリストを指定する、タイプ CLOB(2 M) の入力引数。区切り文字で区切られている表スペース名がサポートされます。作成される表ごとに、このリストに含まれているいずれかの表スペースと表定義が、*targettbsp* リストの n 番目の項目に変換されます。このパラメーターに NULL が指定されている場合、新規オブジェクトは、ソース・オブジェクトが使用するのと同じ表スペースを使って作成されます。

targettbsp
コピー用のターゲット表スペースのコンマ区切りリストを指定する、タイプ CLOB(2 M) の入力引数。区切り文字で区切られている表スペース名がサポートされます。表スペースのリスト *sourcetbsp* の各項目ごとに 1 つの表スペースが指定されていなければなりません。DDL の再生中、*sourcetbsp* リストの n 番目の表スペースは、*targettbsp* リストの n 番目の表スペースにマップされます。'SYS_ANY' を最後の表スペース (ソース・リストのどの名前とも対応しない、追加の表スペース名) として指定することが可能です。'SYS_ANY' が検出されると、オブジェクト作成時にデフォルトの表スペース選択アルゴリズムが使用されます (選択アルゴリズムについて詳しくは、CREATE TABLE ステートメントの資料の IN *tablespace-name1* オプションを参照してください)。このパラメーターに NULL が指定されている場合、新規オブジェクトは、ソース・オブジェクトが使用するのと同じ表スペースを使って作成されます。

errortabschema
コピーできなかったオブジェクトのエラー情報を入れる表のスキーマ名を指定する、タイプ VARCHAR(128) の入出力引数。ADMIN_COPY_SCHEMA プロシージャーがユーザーのためにこの表を SYSTOOLSPACE 表スペースに作成します。エラーが生じなかった場合、このパラメーターの出力は NULL です。

errortab
コピーできなかったオブジェクトのエラー情報を入れる表の名前を指定する、タイプ VARCHAR(128) の入出力引数。ADMIN_COPY_SCHEMA プロシージャーがユーザーのためにこの表を SYSTOOLSPACE 表スペースに作成します。この表は、プロシージャーを呼び出したユーザー ID によって所有されます。エラーが生じなかった場合、このパラメーターの出力は NULL です。表が作成できなかった場合、または既に存在する場合には、プロシージャー操作は失敗し、エラー・メッセージが戻されます。ADMIN_COPY_SCHEMA プロシージャーを呼

び出した後、ユーザーは表をクリーンアップしなければなりません。つまり、SYSTOOLSPACE で表が占めているスペースを取り戻すためには、表をドロップする必要があります。

表 249. ADMIN_COPY_SCHEMA errortab の形式

列名	データ・タイプ	説明
OBJECT_SCHEMA	VARCHAR(128)	コピー・コマンドが失敗したオブジェクトのスキーマ名。
OBJECT_NAME	VARCHAR(128)	コピー・コマンドが失敗したオブジェクトの名前。
OBJECT_TYPE	VARCHAR(30)	オブジェクトのタイプ。
SQLCODE	INTEGER	エラー SQLCODE。
SQLSTATE	CHAR(5)	エラー SQLSTATE。
ERROR_TIMESTAMP	TIMESTAMP	失敗した操作の失敗時刻。
STATEMENT	CLOB(2 M)	失敗したオブジェクトの DDL。ターゲット表へのデータのロード中に障害が生じた場合、このフィールドには失敗したロード・コマンドに対応するテキストが含まれます。
DIAGTEXT	CLOB(2 K)	失敗した操作のエラー・メッセージ・テキスト。

許可

スキーマが正常にコピーされるためには、ユーザーは CREATE_SCHEMA 特権および DB2 オブジェクト固有の特権を持っている必要があります。

例: ADMIN_COPY_SCHEMA コマンドで表をコピーするために必要な CREATE_TABLE 特権および索引をコピーするために必要な CREATE_INDEX 特権が必要です。

ソース・スキーマ内の表がラベル・ベースのアクセス制御 (LBAC) で保護されている場合、ユーザー ID は、ターゲット表にもその同じ保護を作成できるようにするための LBAC 信用証明情報を必要とします。データもコピーする場合、ユーザー ID は、ソース表からのデータの読み取りとそのデータのターゲット表への書き込みの両方を許可する LBAC 信用証明情報を必要とします。

ADMIN_COPY_SCHEMA プロシージャーに対する EXECUTE 特権も必要です。

例

```
CALL SYSPROC.ADMIN_COPY_SCHEMA('SOURCE_SCHEMA', 'TARGET_SCHEMA',
   'COPY', NULL, 'SOURCETS1 , SOURCETS2', 'TARGETTS1, TARGETTS2,
   SYS_ANY', 'ERRORSCHEMA', 'ERRORNAME')
```

制約事項

- HADR を構成するデータベースでは、DDL *copymode* のみがサポートされています。

- COPY または COPY NO を指定した XML はサポートされません。
- COPYNO オプションを指定した ADMIN_COPY_SCHEMA プロシージャーを使用して、ターゲット・データベース・オブジェクトが常駐している表スペースを、バックアップ・ペンディング状態にします。ロード操作の完了後、ターゲット・スキーマ表は SET INTEGRITY ペンディング状態になります。その後、ADMIN_COPY_SCHEMA プロシージャーは、SET INTEGRITY ステートメントを発行して、表をこの状態から解除します。表スペースは既にバックアップ・ペンディング状態にあるので、SET INTEGRITY ステートメントは失敗します。この問題の解決方法について詳しくは、『スキーマのコピー』を参照してください。

使用上の注意

- コピーされるオブジェクト内の完全修飾されたオブジェクトへの参照は変更されません。ADMIN_COPY_SCHEMA プロシージャーは、作成されるオブジェクトの修飾スキーマのみを変更し、それらのオブジェクトの SQL 式内に現れるスキーマ名は変更しません。これには生成された列やトリガー本体などのオブジェクトが含まれます。
- このプロシージャーは、以下のオブジェクトのコピーはサポートしません。
  - 索引拡張
  - ニックネーム
  - パッケージ
  - 型付き表
  - 配列タイプ
  - ユーザー定義の構造化タイプ (およびそれらのトランスフォーム関数)
  - 型付きビュー
  - JAR (Java ルーチン・アーカイブ)
  - ステージング表
  - 同じソース・スキーマに属さない基本オブジェクトの別名
- コピーされるスキーマ内に上記のオブジェクトのいずれかが存在する場合、そのオブジェクトはコピーされず、オブジェクトがコピーされなかったことを示す項目がエラー表に追加されます。
- 複製された表をコピーする場合、表の新しいコピーではサブスクリプションが使用可能になりません。表は、基本表として再作成されるに過ぎません。
- このプロシージャーを操作するには、SYSTOOLSPACE 表スペースが存在していることが必要です。この表スペースは、ADMIN_COPY_SCHEMA プロシージャーが使用するメタデータと、このプロシージャーによって戻されるエラー表を保管するために使用されます。表スペースが存在しない場合は、エラーが戻されます。
- ターゲット・スキーマ内のオブジェクトの統計はデフォルトに設定されています。
- 表に生成された ID 列があり、かつ *copymode* が 'COPY' または 'COPYNO' のいずれかである場合、ソース表からのデータ値はロード中に保存されます。
- 各外部ルーチンごとに、元のソース・ルーチンのバイナリーを参照する新しいカタログ項目が 1 つ作成されます。

- コピー操作の開始時に表が SET INTEGRITY ペンディング状態にあった場合、データはターゲット表にロードされず、その表に関してデータがロードされなかったことを示す項目が *errortab* に記録されます。
- ロードまたは DDL 操作が失敗した場合、作成されなかったオブジェクトに関する項目が *errortab* に記録されます。正常に作成されたオブジェクトすべてはそのまま残ります。リカバリーするには、手動ロードを開始するか、ADMIN_DROP_SCHEMA プロシージャーを使用して新しいスキーマをドロップしてから、ADMIN_COPY_SCHEMA プロシージャーを再び呼び出すことができます。
- ターゲット・スキーマがソース・スキーマと一致する場合、DDL 再生中に、デフォルト・スキーマはターゲット・スキーマにオーバーライドされます。
- トリガー、ビュー、または SQL 関数のコンパイルに使用される関数パスは、ソース・オブジェクトの作成に使用されたパスです。ただしこれには例外があり、オブジェクトの関数パスにソース・スキーマ名が含まれている場合、DDL 再生中にパス内のこの項目がターゲット・スキーマ名に変更されます。
- 複数の ADMIN_COPY_SCHEMA プロシージャーを実行すると、デッドロックが生じます。一度に 1 つの ADMIN_COPY_SCHEMA プロシージャー呼び出しだけを発行する必要があります。コピー処理中にソース・スキーマ内で表を変更すると、ターゲット・スキーマ内のデータがコピー操作の後には等しくないことを示す場合があります。
- 単一パーティションのデータベース・パーティション・グループ内の表スペースから、複数パーティションのデータベース・パーティション・グループ内の表スペースに、表を備えたスキーマをコピーするときは、慎重に考慮する必要があります。分散キーの自動選択が設定済みでない限り、スキーマのコピー操作を実行する前に、分散キーを表で定義する必要があります。分散キーの変更は、表スペースが単一パーティション・データベース・パーティション・グループと関連している表に対してのみ行うことができます。

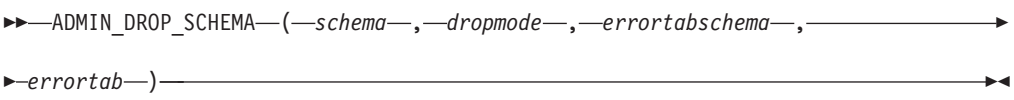
トランザクションの考慮事項

- ADMIN_COPY_SCHEMA プロシージャーが、その処理中にデッドロックまたはロックのタイムアウトのためにロールバックを強制された場合、ADMIN_COPY_SCHEMA プロシージャーを呼び出した作業単位内で行われた作業もロールバックされます。
- コピーの DDL 段階で障害が生じた場合、ターゲット・スキーマになされた変更はすべてセーブポイントにロールバックされます。
- *copymode* が 'COPY' または 'COPYNO' に設定されている場合、いったんコピーの DDL 段階が完了すると ADMIN_COPY_SCHEMA プロシージャーがコミットし、そのプロシージャーを呼び出した作業単位でなされた作業もコミットします。

ADMIN_DROP_SCHEMA プロシージャー - 特定のスキーマとそのオブジェクトのドロップ

ADMIN_DROP_SCHEMA プロシージャーは、特定のスキーマと、その中に含まれているすべてのオブジェクトをドロップするために使用されます。

構文



スキーマは SYSPROC です。

プロシージャー・パラメーター

schema
ドロップされるスキーマの名前を指定する、タイプ VARCHAR(128) の入力引数。名前は、大文字で指定しなければなりません。

dropmode
将来の使用のために予約済みであり、NULL に設定する必要があります。

errortabschema
ドロップできなかったオブジェクトのエラー情報を入れる表のスキーマ名を指定する、タイプ VARCHAR(128) の入出力引数。この名前は大文字小文字が区別されます。ADMIN_DROP_SCHEMA プロシージャーがユーザーのためにこの表を SYSTOOLSPACE 表スペースに作成します。エラーが生じなかった場合、このパラメーターの出力は NULL です。

errortab
ドロップできなかったオブジェクトのエラー情報を入れる表の名前を指定する、タイプ VARCHAR(128) の入出力引数。この名前は大文字小文字が区別されます。ADMIN_DROP_SCHEMA プロシージャーがユーザーのためにこの表を SYSTOOLSPACE 表スペースに作成します。この表は、プロシージャーを呼び出したユーザー ID によって所有されます。エラーが生じなかった場合、このパラメーターの出力は NULL です。表が作成できなかった場合、または既に存在する場合には、プロシージャー操作は失敗し、エラー・メッセージが戻されます。ADMIN_DROP_SCHEMA を呼び出した後、ユーザーは表をクリーンアップしなければなりません。つまり、SYSTOOLSPACE で表が占めているスペースを取り戻すためには、表をドロップする必要があります。

表 250. ADMIN_DROP_SCHEMA errortab の形式

列名	データ・タイプ	説明
OBJECT_SCHEMA	VARCHAR(128)	ドロップ・コマンドが失敗したオブジェクトのスキーマ名。
OBJECT_NAME	VARCHAR(128)	ドロップ・コマンドが失敗したオブジェクトの名前。
OBJECT_TYPE	VARCHAR(30)	オブジェクトのタイプ。
SQLCODE	INTEGER	エラー SQLCODE。
SQLSTATE	CHAR(5)	エラー SQLSTATE。
ERROR_TIMESTAMP	TIMESTAMP	ドロップ・コマンドが失敗した時刻。
STATEMENT	CLOB(2 M)	失敗したオブジェクトの DDL。

表 250. ADMIN_DROP_SCHEMA errortab の形式 (続き)

列名	データ・タイプ	説明
DIAGTEXT	CLOB(2 K)	失敗したドロップ・コマンドのエラー・メッセージ・テキスト。

許可

このプロシージャーを呼び出すユーザーには、削除されるすべてのオブジェクトに対するドロップ権限が必要です。

ADMIN_DROP_SCHEMA プロシージャーに対する EXECUTE 特権も必要です。

例

```
CALL SYSPROC.ADMIN_DROP_SCHEMA('SCHNAME', NULL, 'ERRORSCHEMA', 'ERRORTABLE')
```

以下はこのプロシージャーの出力例です。

```
Value of output parameters
--------------------------
Parameter Name : ERRORTABSCHEMA
Parameter Value : ERRORSCHEMA <-- error!

Parameter Name : ERRORTAB
Parameter Value : ERRORTABLE <-- error!

Return Status = 0
```

戻り状況は、内部エラーが検出された場合 (SYSTOOLSPACE が存在しないなど) のみ、0 ではありません。

エラーは、エラー表の照会によってチェックできます。

```
SELECT * FROM ERRORSCHEMA.ERRORTABLE
```

使用上の注意

- ドロップするオブジェクトに別のスキーマ内のオブジェクトが従属している場合、デフォルトの DROP ステートメントのセマンティクスが適用されます。
- このプロシージャーは、以下のオブジェクトのドロップはサポートしません。
  - 索引拡張
  - ニックネーム
  - パッケージ
  - 型付き表
  - 配列タイプ
  - ユーザー定義の構造化タイプ (およびそれらのトランスフォーム関数)
  - 型付きビュー
  - JAR (Java ルーチン・アーカイブ)
  - ステージング表
- ドロップされるスキーマ内に上記のオブジェクトのいずれかが存在する場合、そのオブジェクトもそのスキーマもドロップされず、オブジェクトがドロップされなかったことを示す項目がエラー表に追加されます。

- このプロシージャーを操作するには、SYSTOOLSPACE 表スペースが存在していることが必要です。この表スペースは、ADMIN_DROP_SCHEMA プロシージャーが使用するメタデータと、このプロシージャーによって戻されるエラー表を保管するために使用されます。表スペースが存在しない場合は、エラーが戻されます。

ADMIN_MOVE_TABLE プロシージャー - オンライン表の移動

ADMIN_MOVE_TABLE ストアード・プロシージャーは、アクティブな表のデータを、同じ名前の新しい表オブジェクトに移動します。データはオンラインのままなので、引き続きアクセス可能です。このストアード・プロシージャーは、移動対象の表に関連する状況情報と構成オプションを内容とする行から成るプロトコル表を作成します。このプロシージャーからの戻りセットは、移動対象の表に関連するこのプロトコル表の行です。

このストアード・プロシージャーでは、以下の用語を使用します。

ソース表
: ストアード・プロシージャーにパラメーターとして渡される元の表の名前。移動対象の表です。

ターゲット表
: ストアード・プロシージャーで渡された表定義を使用して、ストアード・プロシージャーによって作成される表。この表にソース表のすべてのデータがコピーされ、ソース表と同じ名前に名前変更されます。

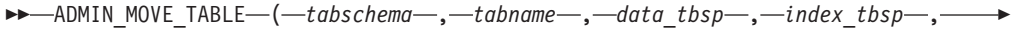
ステージング表
: ストアード・プロシージャーによって作成される表。表移動の実行中にソース表に対して行われる更新、削除、または挿入による変更はすべて、ステージング表に保管されます。この表は、移動が完了するとドロップされます。

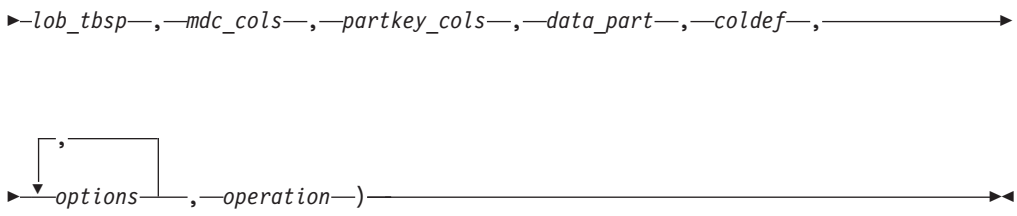
構文

ADMIN_MOVE_TABLE を呼び出す場合、2 つの同等に有効な方法があります。 1 つ目の方法では、ターゲット表の表定義の特定の部分のみを変更できます。例えば、表定義がかなり大きい (数 KB) 場合、表の表スペースのみを変更したいときは、ソース表の再作成に必要な CREATE TABLE ステートメント全体を決定しなくても、その変更を行うことができます。 data_tbsp、index_tbsp、および lob_tbsp パラメーターを設定するだけでよく、他のオプション・パラメーターはブランクのままにしておくことができます。

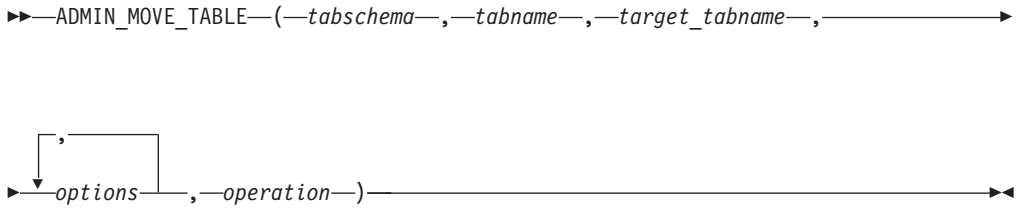
2 番目の方法は、ターゲット表をストアード・プロシージャーに作成させるのではなく、あらかじめ作成しておけるので、制御と柔軟性の面でまさっています。この方法を使用すると、最初の方式では不可能なターゲット表を作成できます。

方法 1:





方法 2:



両方式ともスキーマは SYSPROC です。

プロシージャー・パラメーター

tabschema
この入力パラメーターは、移動対象の表が含まれるスキーマの名前を指定します。このパラメーターには大/小文字の区別があり、データ・タイプは VARCHAR(128) です。

tabname
この入力パラメーターは、移動対象の表の名前を指定します。このパラメーターには大/小文字の区別があり、データ・タイプは VARCHAR(128) です。

data_tbsp
この入力パラメーターは、ターゲット表の新しいデータ表スペースを指定します。値を指定する場合は、*index_tbsp* および *lob_tbsp* パラメーターが必須となります。値を指定しない場合は、ソース表のデータ表スペースが使用されます。このパラメーターには大/小文字の区別があり、データ・タイプは VARCHAR(128) です。このパラメーターを NULL または空ストリングにすることができます。

index_tbsp
この入力パラメーターは、ターゲット表の新しい索引表スペースを指定します。値を指定する場合は、*data_tbsp* および *lob_tbsp* パラメーターが必須となります。値を指定しない場合は、ソース表の索引表スペースが使用されます。このパラメーターには大/小文字の区別があり、データ・タイプは VARCHAR(128) です。このパラメーターを NULL または空ストリングにすることができます。

lob_tbsp
この入力パラメーターは、ターゲット表の新しい LOB 表スペースを指定します。値を指定する場合は、*data_tbsp* および *index_tbsp* パラメーターが必須となります。値を指定しない場合は、ソース表の LOB 表スペースが使用されます。このパラメーターには大/小文字の区別があり、データ・タイプは VARCHAR(128) です。このパラメーターを NULL または空ストリングにすることができます。

mdc_cols
この入力パラメーターは、ターゲット表のマルチディメンション列 (MDC) 仕様を指定します。ターゲット表のデータをマルチディメンションに従ってクラスター化するために使用する列を決定し、それらをコンマ区切りリストにして値を入力します。値として NULL または「-」を指定した場合、ORGANIZE BY DIMENSIONS 節は使用されません。空ストリングまたは単一ブランクを指定すると、プロシージャーはソース表に MDC 仕様があるかどうかを検査し、見つかった場合はその仕様を使用します。このパラメーターは VARCHAR(32672) データ・タイプで、CREATE TABLE ステートメントの ORGANIZE BY DIMENSIONS 節と同じ形式です。このパラメーターを NULL、空ストリング、または単一ブランクにすることができます。

例: 'C1, C4, (C3,C1), C2'

partkey_cols
この入力パラメーターは、ターゲット表のパーティション・キー列仕様を指定します。複数のデータベース・パーティションにデータをどのように分散させるかを指定するキー列を決定し、それらをコンマ区切りリストにして値を入力します。値として NULL または「-」を指定した場合、PARTITIONING KEY 節は使用されません。空ストリングまたは単一ブランクを指定すると、プロシージャーはソース表にパーティション・キー列仕様があるかどうかを検査し、見つかった場合はその仕様を使用します。このパラメーターは VARCHAR(32672) データ・タイプで、CREATE TABLE ステートメントの DISTRIBUTE BY HASH 節と同じ形式です。

例: 'C1, C3'

data_part
この入力パラメーターは、ターゲット表のデータ・パーティション仕様を指定します。このステートメントは、表データを複数のストレージ・オブジェクト (データ・パーティションと言う) にどのように分割するかを、表の 1 つ以上の列の値に基づいて定義します。値として NULL または「-」を指定した場合、PARTITION BY RANGE 節は使用されません。空ストリングまたは単一ブランクを指定すると、プロシージャーはソース表にデータ・パーティション・スキームがあるかどうかを検査し、見つかった場合はその情報 (パーティション名を含む) を使用します。このパラメーターは VARCHAR(32672) データ・タイプで、CREATE TABLE ステートメントの PARTITION BY RANGE 節と同じ形式です。

例: '(C1) (STARTING FROM (1) EXCLUSIVE ENDING AT (1000) EVERY (100))'

coldef
この入力パラメーターは、ターゲット表の新しい列定義を指定します。列タイプの変更は互換性のある限り可能ですが、列名は同じままでなければなりません。

新しい列を追加したり、既存の列をドロップしたりすることもできます。列を追加する場合は、NULL 可能と定義するか、またはデフォルト値を設定する必要があります。また、表にユニーク索引または 1 次索引があり、ドロップされる列がそのユニーク索引または 1 次索引の一部でない場合のみ、列をドロップできます。このパラメーターのデータ・タイプは VARCHAR(32672) です。このパラメーターを NULL または空ストリングにすることができます。

例: 'C1 INT, C2 INT DEFAULT 0'

target_tabname
この入力パラメーターは、移動中にターゲット表として使用する既存の表の名前を指定します。渡されるターゲット表に対して、以下の変更を加えることができます。

- データ、索引、LOB の表スペースを変更できます。
- マルチディメンション列 (MDC) 仕様を追加または変更できます。
- パーティション・キー列仕様を追加または変更できます。
- データ・パーティション仕様を追加または変更できます。
- データ圧縮を追加または除去できます。
- 新しい列定義を指定できます。ただし、*coldef* パラメーターを指定する場合と同じ制限が適用されます。

指定する表には、以下の制限が適用されます。

- ソース表と同じスキーマに表が存在しなければなりません。
- 表は空でなければなりません。
- 型付き表、マテリアライズ照会表 (MQT)、ステージング表、リモート表、クラスター表は許可されません。

このパラメーターを NULL または空ストリングに設定した場合、ストアード・プロシージャーはソース表と同じ定義を使用します。このパラメーターには大/小文字の区別があり、データ・タイプは VARCHAR(128) です。

options
このコンマ区切り入力パラメーターのセットは、ストアード・プロシージャーが使用するオプションを定義します。

- KEEP: このオプションを設定すると、元のソース表のコピーが異なる名前で保持されます。ソース表の名前が T1 の場合、移動後にこの表は自動的に T1AAAAVxo のような名前に変更されます。ソース表の正確な名前は、戻されるプロトコル表の ORIGINAL キーのフィールドに示されます。このオプションは、SWAP フェーズまでの (SWAP フェーズを含む) 任意の時点で設定できます。
- COPY_USE_LOAD: このオプションを設定すると、ソース表からターゲット表へのデータのコピーに、リカバリー不能 db2Load API が使用されます。このオプションは、COPY フェーズまでの (COPY フェーズを含む) 任意の時点で設定できます。 DB2 バージョン 9.7 フィックスパック 2 より前のリリースでは、COPY_USE_LOAD を使用する場合は FORCE オプションを指定しなければなりません。
- COPY_WITH_INDEXES: このオプションを設定すると、ソース表のコピー前に索引が作成されます。一方デフォルトでは、ソース表のコピー後に索引が作成されます。このオプションの利点は、コピー後の索引作成の場合は索引ごとに表全体スキャンが必要になるということ、および索引作成はアクティブ・ログ・スペースを必要とするトランザクションであるということです。LOGINDEXREBUILD データベース構成パラメーターがオンの場合、索引を短い時間枠で作成するためには、かなりのログ・スペースが必要になります。このオプションの欠点の 1 つは、索引をターゲット表で維持する必要があるために、コピー・パフォーマンスが低下するという点です。また、結果

の索引は「疑似」削除済みキーを多く含んでおり、コピー後に索引が作成された場合ほど索引のバランスが取れていません。 COPY_WITH_INDEXES オプションは、COPY フェーズまでの (COPY フェーズを含む) 任意の時点で設定できます。

- FORCE: この強制オプションを設定すると、ソース表の表定義が変わったかどうかが SWAP フェーズで検査されません。 DB2 バージョン 9.7 フィックスパック 2 より前のリリースでは、COPY_USE_LOAD を使用する場合は FORCE オプションを指定しなければなりません。このオプションは、SWAP フェーズまでの (SWAP フェーズを含む) 任意の時点で設定できます。
- NO_STATS: このオプションを設定すると、ターゲット表での RUNSTATS または統計コピーは開始されません。 AUTO_RUNSTATS または AUTO_STMT_STATS データベース構成パラメーターを使用すると、その後 DB2 が自動的に新しい統計を作成します。後方互換性のために STATS_NO も受け入れます。 NO_STATS オプションは、SWAP フェーズまでの (SWAP フェーズを含む) 任意の時点で設定できます。
- COPY_STATS: このオプションを設定すると、スワップ実行前にソース表からターゲット表に統計がコピーされます。これにより、ページ・サイズが変わる場合は特に、物理統計が不正確になる可能性があります。しかし、このオプションを設定すると、新しい統計を計算するために RUNSTATS が呼び出されることがないので、計算時間を節約できます。また、同じ統計になるので、オプティマイザーが同じアクセス・プランを選択する可能性があります。後方互換性のために STATS_COPY も受け入れます。 STATS_COPY オプションは、SWAP フェーズまでの (SWAP フェーズを含む) 任意の時点で設定できます。
- NO_AUTO_REVAL: このオプションを設定すると、表に関する自動再有効化は行われません。代わりに、すべてのトリガーとビューが再作成されます。NO_AUTO_REVAL オプションは INIT フェーズでのみ設定できます。
- REORG: このオプションを設定すると、スワップ実行前にターゲット表に対する補足的なオフライン REORG がセットアップされます。このオプションを使用してコンプレッション・ディクショナリーを改良することも考えられますが、最適なコンプレッション・ディクショナリーを作成する方法としては、デフォルトのサンプリング・アプローチの方が適していると言えます。ただし、最適な XML コンプレッション・ディクショナリーを必要とする場合は、REORG が唯一の方法です。 REORG オプションは、SWAP フェーズまでの (SWAP フェーズを含む) 任意の時点で設定できます。
- NO_TARGET_LOCKSIZE_TABLE: このオプションは、COPY および SWAP フェーズ中にターゲット表でロック・サイズ表を保持しません。デフォルトでは、ソース表でユニーク索引が指定されていない場合、ロック・オーバーヘッドを防ぐためにターゲット表でロック・サイズ表を保持します。このオプションは、バージョン 9.7 フィックスパック 1 以降のフィックスパックで使用可能です。
- CLUSTER: このオプションは、ソース表にクラスター索引が存在する場合、またはコピー索引が指定されている場合に、ORDER BY 節を使ってソース表からデータを読み取ります。このオプションは、バージョン 9.7 フィックスパック 1 以降のフィックスパックで使用可能です。

- NON_CLUSTER: このオプションは、クラスター索引またはコピー索引が指定されているかどうかにかかわらず、ORDER BY 節を使用せずにソース表からデータを読み取ります。注: CLUSTER オプションと NON_CLUSTER オプションのどちらも指定しない場合には、ソース表にクラスター索引が存在する場合に限り、ORDER BY 節を使ってソース表からデータを読み取ります。このオプションは、バージョン 9.7 フィックスパック 1 以降のフィックスパックで使用可能です。
- LOAD_MSGPATH <path>: このオプションは、COPY_USE_LOAD オプションが指定された際に、ロード・メッセージ・ファイルのパスを定義するのに使用します。LOAD_MSGPATH オプションを指定しない場合は、デフォルト・パスとして diagpath が使用されます。このオプションは、DB2 バージョン 9.7 フィックスパック 2 以降で使用可能です。

このオプション・リストには大/小文字の区別がなく、データ・タイプは VARCHAR(128) です。リスト値を NULL または空ストリングにすることができます。

operation

この入力パラメーターは、ストアード・プロシージャーが実行する操作を指定します。ストアード・プロシージャーを呼び出す方法は 2 通りあります。MOVE コマンドを使用してすべての操作を一度に実行する方法と、個々のコマンドを使用して表移動を 1 ステップずつ実行する方法です。この 2 番目の方法の主な利点は、SWAP フェーズを実際にいつ行うかを制御できるので、表をいつ一時的にオフラインにするかを決定できるという点です。これにより、システム・アクティビティーが低い間に移動を行うことができます。個々のコマンドを使用する場合は、INIT、COPY、REPLAY、VERIFY (オプション)、SWAP の順に呼び出す必要があります。

- MOVE: 表移動全体 (INIT、COPY、REPLAY、および SWAP 操作) を 1 ステップで実行します。
- INIT: 表移動を行えることを検査し、表移動の過程で必要なすべてのデータ (ターゲット表、ステージング表、およびソース表のトリガー) を初期化します。
- COPY: ソース表の内容をターゲット表にコピーします。その間ソース表で行われる更新、削除、挿入はキャプチャーされ、ステージング表に保管されます。 COPY_WITH_INDEXES オプションが選択されていなければ、COPY フェーズの最後に新しい索引が作成されます。パフォーマンスを上げるために必要であれば、REPLAY フェーズでソース表とターゲット表の副次索引も作成されます。 COPY を使用できるのは、INIT フェーズが完了した後に限られます。
- REPLAY: COPY フェーズの開始以降ソース表で変更された行を、ターゲット表にコピーします。 REPLAY を使用できるのは、COPY フェーズが完了した後に限られます。
- VERIFY: (オプション) ソース表とターゲット表の表の内容が等しいかどうかを検査します。このプロセスでは、まずソース表とターゲット表の共用ロックが獲得されます。次にソース表で行われた変更が再生された後に、比較が行われます。表にユニーク索引がある場合、このコマンドは両方の表にある列どうしですべての値を比較します。そうでない場合、このコマンドは両方の表にある列 (LONG、LOB、または XML 列を除く) どうしで すべての値

を比較します。これは高コストの操作になるので、移動に有用かどうかを慎重に判断する必要があります。 VERIFY を使用できるのは、COPY または REPLAY フェーズが完了した後に限られます。

- SWAP: ステージング表の最後のスキャン中に適用された変更の数が、プロトコル表に保管されている REPLAY_THRESHOLD 値より小さくなるまで、REPLAY フェーズを実行します。その後ソース表が一時的にオフラインになって最終 REPLAY が完了します。次にこのコマンドはソース表とターゲット表をスワップし、表をオンラインに戻します。 SWAP を使用できるのは COPY フェーズが完了した後ですが、理想的には REPLAY フェーズが呼び出された後です。
- CLEANUP: ステージング表と、ストアード・プロシージャーによってソース表に対して作成された非ユニーク索引またはトリガーをドロップします。さらに、KEEP オプションが設定されていない場合は、ソース表もドロップします。 CLEANUP を呼び出すことができるのは、SWAP フェーズでコマンドが失敗した場合です。
- CANCEL: 複数ステップの表移動をフェーズとフェーズの間で取り消します。または、失敗した表移動操作を取り消します。このコマンドを実行するためには、操作状況が COMPLETED 状態でも CLEANUP 状態でもないことが必要です。 CANCEL は、すべての中間データ (索引、ステージング表、ターゲット表、およびソース表のトリガー) をクリアします。

このパラメーターには大/小文字の区別がなく、データ・タイプは VARCHAR(128) です。

許可

ADMIN_MOVE_TABLE ストアード・プロシージャーを呼び出すためには、SQLADM または DBADM 権限のいずれかが必要です。また、SELECT ステートメントをソース表に発行したり、ターゲット表に INSERT ステートメントを発行したりする権限を含め、適切なオブジェクト作成権限もなければなりません。

例

この例は、最初の方法でストアード・プロシージャーを呼び出し、スキーマ「SVALENTI」にある T1 という名前の表を移動します。ここでは、ターゲット表がプロシージャー内で定義されます。

```
CALL SYSPROC.ADMIN_MOVE_TABLE(
'SVALENTI',
'T1',
'ACCOUNTING',
'ACCOUNT_IDX',
'ACCOUNT_LONG',
'',
'',
'',
'CUSTOMER VARCHAR(80), REGION CHAR(5), YEAR INTEGER, CONTENTS CLOB',
'',
'MOVE')
```

以下はこの照会の出力例です

```
Result set 1
------------
```

```
KEY                         VALUE
------------------------    -----------------------------------
AUTHID                      SVALENTI
CLEANUP_END                 2009-02-13-11.34.07.609575
CLEANUP_START               2009-02-13-11.34.07.369331
COPY_END                    2009-02-13-11.34.05.148018
COPY_OPTS                   BY_KEY,OVER_INDEX
COPY_START                  2009-02-13-11.34.04.841292
COPY_TOTAL_ROWS             100
INDEXNAME                   T1_INDEX
INDEXSCHEMA                 SVALENTI
INDEX_CREATION_TOTAL_TIME   0
INIT_END                    2009-02-13-11.34.04.552875
INIT_START                  2009-02-13-11.34.03.013563
PAR_COLDEF                  CUSTOMER VARCHAR(80), REGION CHAR(5),
                                YEAR INTEGER, CONTENTS CLOB
REPLAY_END                  2009-02-13-11.34.06.198369
REPLAY_START                2009-02-13-11.34.05.164582
REPLAY_TOTAL_ROWS           100
REPLAY_TOTAL_TIME           5
STATUS                      COMPLETE
SWAP_END                    2009-02-12-11.34.07.214447
SWAP_RETRIES                0
SWAP_START                  2009-02-13-11.34.06.244506
VERSION                     09.07.0000

22 record(s) selected.

Return Status = 0
```

この例は、2 番目の方法でストアード・プロシージャーを呼び出し、先ほどと同じ表を移動します。ここでは、ターゲット表がプロシージャー外で作成され、その後 *target_tabname* パラメーター内で名前が付けられます。

最初のステップとして、手動で表を作成します。

```
CREATE TABLE SVALENTI.T1_TARGET (
   CUSTOMER VARCHAR(80),
   REGION CHAR(5),
   YEAR INTEGER,
   CONTENTS CLOB)
IN ACCOUNTING
INDEX IN ACCOUNT_IDX
LONG IN ACCOUNT_LONG'
```

次に、ストアード・プロシージャーを呼び出し、ターゲット表の名前を渡します。

```
CALL SYSPROC.ADMIN_MOVE_TABLE(
'SVALENTI',
'T1',
'T1_TARGET',
'',
'MOVE')
```

以下はこの照会の出力例です

```
Result set 1
------------

KEY                         VALUE
------------------------    -----------------------------------------
AUTHID                      SVALENTI
CLEANUP_END                 2009-02-13-11.37.49.283090
CLEANUP_START               2009-02-13-11.37.49.125786
COPY_END                    2009-02-13-11.37.47.806060
COPY_OPTS                   BY_KEY,OVER_INDEX
```

```
COPY_START                  2009-02-13-11.37.47.446616
COPY_TOTAL_ROWS             0
INDEXNAME                   T1_INDEX
INDEXSCHEMA                 SVALENTI
INDEX_CREATION_TOTAL_TIME   1
INIT_END                    2009-02-13-11.37.47.287703
INIT_START                  2009-02-13-11.37.46.052952
PAR_COLDEF                  using a supplied target table so COLDEF
                                could be different
REPLAY_END                  2009-02-13-11.37.48.785503
REPLAY_START                2009-02-13-11.37.47.822109
REPLAY_TOTAL_ROWS           0
REPLAY_TOTAL_TIME           0
STATUS                      COMPLETE
SWAP_END                    2009-02-13-11.37.48.977745
SWAP_RETRIES                0
SWAP_START                  2009-02-13-11.37.48.825228
VERSION                     09.07.0000
22 record(s) selected.

Return Status = 0
```

使用上の注意

このプロシージャーを使用する場合に最良の結果を得るためのヒント:

- 同一表スペースに複数の移動を同時に行うことは避けてください。ターゲット表スペースでのフラグメント化の妨げになります。
- 表に対するアクティビティーが低いときに、このプロシージャーを実行してください。並列読み取りアクセスが問題にならないように、データの一括ロードや一括削除は避けてください。
- 複数ステップの移動操作を使用してください。 INIT フェーズと COPY フェーズはいつでも呼び出すことができます。ステージング表のサイズを小さくしておくためには、REPLAY フェーズを複数回実行します。その後、表に対するアクティビティーが低いときに SWAP を発行してください。
- ユニーク索引のない表を検討する場合や索引のない表の場合は特に、表移動の方式としてオフライン方式を選択した方が良くないか確認してください。

ソース表に対する制限された操作

ストアード・プロシージャーは、ソース表の何らかの変更点をキャプチャーするためにトリガーに依存します。ソース表に影響を与える可能性がある操作の中には、トリガーを起動しないものもあります。その結果、ストアード・プロシージャーによって簡単に検出できないような不整合がソース表とターゲット表の間に発生する可能性があります。次のような操作がこれに該当します。

- TRUNCATE TABLE (トリガー削除時の制限なし)
- IMPORT ... REPLACE INTO ...
- LOAD TABLE
- ALTER TABLE
- REORG (オンラインおよびオフライン)

ソース表に対するこれらの操作は、新しい表レベルの状態フラグを使って制限されます。このフラグは INIT フェーズ中に設定され、CLEANUP または CANCEL フェーズ中に消去されます。制限された操作は失敗して、SQL0668N 理由コード 10 (sqlstate 57016) が出されます。

表の移動操作に影響を与える操作

移動操作中にストアード・プロシージャーの失敗の原因となり得る操作があります。次のような操作がこれに該当します。

- SYSTOOLSPACE 表スペースのドロップ
- ソース表のドロップ/名前変更
- INIT フェーズで OTM によって作成されたいずれかの一時オブジェクトのドロップ/名前変更 (ターゲット表、ステージング表、ソース表のトリガー、プロトコル表)
- ユーザー構成可能としてリストされていない、プロトコル表内の値の変更

一時オブジェクトの命名規則

一時オブジェクトを作成する際の名前の競合を防ぐために、以下のような命名規則が使用されます。

- 接尾部
  - "t" (ターゲットを示す)
  - "s" (ステージングを示す)
  - "o" (「元の」を示す)
  - "g" (「生成された」を示す)
  - "i" (挿入トリガーを示す)
  - "d" (削除トリガーを示す)
  - "u" (更新前トリガーを示す)
  - "v" (更新後トリガーを示す)
- 名前は、<オブジェクト名から取られた文字><オブジェクト名に対する Base64 エンコード・ハッシュ・キー><接尾部> という構成で作成されます。
- 名前の長さがオブジェクトの長さ (128 バイト) を超える場合には、<オブジェクト名から取られた文字> が短くなります。
- ハッシュ値はオブジェクト名から計算されて、Base64 エンコード方式と同様にエンコードされます。

サンプル:

```
Name of object: T1
Staging object: T1AAAAVxs
Target object: T1AAAAVxt
Original object: T1AAAAVxo
Generated index: T1AAAAVxg (if table has no index)
Insert trigger: T1AAAAVxi
Delete trigger: T1AAAAVxd
Before update trigger: T1AAAAVxu
After update trigger: T1AAAAVxv
```

圧縮およびディクショナリー作成を伴うオンライン表移動

オンライン表移動を使ってデータ・コンプレッション・ディクショナリーを作成する方法は、いくつかあります。ソース表で圧縮が使用可能に設定されるか、または新しい表定義 (提供される場合) で圧縮がアクティブと指定される必要があります。

サンプリングを使用したディクショナリーの作成は、オンライン表移動を介したディクショナリー作成のデフォルト方式です。表の圧縮がオンに設定されている場合、COPY 操作の実行前に、ソース表のデータのベルヌーイ・サンプリングがターゲット表に挿入されます。サンプリングされるデータの量はプロトコル表の DEEPCOMPRESSION_SAMPLE フィールドで指定されます。その後、このランダム・サンプルに基づいてコンプレッション・ディクショナリーが作成され、結果として最適なコンプレッション・ディクショナリーになります。

サンプリング方式では XML コンプレッション・ディクショナリーが作成されないことに注意してください。これはコンプレッション・ディクショナリーの作成に db2Inspect が使用されるためであり、現在、db2Inspect には XML コンプレッション・ディクショナリーを作成する機能がありません。 XML コンプレッション・ディクショナリーは自動ディクショナリー作成 (ADC) を介して作成されます。

自動ディクショナリー作成 (ADC) によるディクショナリー作成は、DB2 の表でディクショナリーを作成する標準的な方法です。表の圧縮をオンに設定するだけで、データが表に挿入されるときに DB2 はディクショナリーを自動作成します。その結果として、非最適コンプレッション・ディクショナリーが生成されます。ストアード・プロシージャーがより最適なコンプレッション・ディクショナリーの作成を試みるのを防止するには、プロトコル表の DEEPCOMPRESSION_SAMPLE フィールドを 0 に設定する必要があることに注意してください。

ディクショナリー作成の REORG メソッドを使ってディクショナリーを作成すると、COPY フェーズ中にソース表で発生したすべてのアクティビティーを反映するディクショナリーが結果として作成されます。これは、RESETDICTIONARY オプションを設定して SWAP フェーズの前に REORG を実行することによって行われます。最適なディクショナリーが作成されますが、表のサイズによっては REORG に長い時間がかかる可能性があります。さらに、最適な XML ディクショナリーが必要な場合、それを生成する唯一の方法は REORG です。サンプリング方式を使ってディクショナリーを作成することが推奨されます。

オンライン表移動と表の統計

統計が収集される表に対して表移動を実行する場合、デフォルト動作として、表に対する RUNSTATS が SWAP フェーズ中に実行されます。統計プロファイルが見つかった場合、その統計プロファイルを使用して RUNSTATS が呼び出されます。そうでない場合は、"WITH DISTRIBUTION ON COLUMNS (...) AND SAMPLE DETAILED INDEXES ALL" オプションを使って RUNSTATS が呼び出されます。

COPY_STATS オプションが設定されている場合、スワップの実行前にソース表から統計がターゲット表にコピーされます。統計をコピーすると、特にページ・サイズが変更される場合には、物理統計が不正確になる可能性があります。ただし、新しい統計を計算するために RUNSTATS を呼び出す必要がないため、コンピューティング時間が節約されます。また、統計が同じであるため、オプティマイザーは同じアクセス・プランを選択することができます (プランの安定性)。コピーされる統計は、SYSSTAT.TABLES、 SYSSTAT.COLUMNS、 SYSSTAT.COLDIST、 SYSSTAT.INDEXES、および SYSSTAT.COLGROUPS カタログ・ビューの中にあります。

NO_STATS オプションが設定されている場合、ストアード・プロシージャーは RUNSTATS や統計コピー操作をターゲット表に対して実行しません。AUTO_RUNSTATS または AUTO_STMT_STATS を使用すると、DB2 は新しい統計を自動的に作成します。

COPY での LOAD 使用を伴うオンライン表移動

COPY_USE_LOAD オプションを使用する場合、リカバリー可能性を確保するには、SWAP フェーズの前にターゲット表スペースのバックアップを実行する必要があります。次のようなステートメントを発行することで、バックアップを作成できます。

```
BACKUP DB dbname TABLESPACE targetDataTablespace, targetIndexTablespace
    ONLINE TO <destination>
```

DB2 バージョン 9.7 フィックスパック 2 より前のリリースでは、COPY_USE_LOAD を使用する場合は FORCE オプションを指定しなければなりません。そうしない場合、SWAP フェーズは実行されずエラーを受け取ることになります。

生成済み列を伴うオンライン表移動

表移動ストアード・プロシージャーは、ソース表の中の生成済み列を特別な方法で扱います。以下に、さまざまな種類の生成済み列がどのように扱われるかを説明します。

行変更タイム・スタンプ列は、行が最後に変更された時間を表すタイム・スタンプを保持する列です。

ソース表で行変更タイム・スタンプ列が見つかった場合、表移動操作が完了した後のこの列の値は、表移動操作の前と同じではありません。表移動後のこの列の値は、新しい表オブジェクトで行が挿入/更新された時間を表すようになります。これは、行が実際に変更されようとしており、行変更タイム・スタンプ列の値はこれらの変更を反映する必要があるためです。

新しい表定義が提供される場合、ソース表で列が行変更タイム・スタンプ列と定義されていても、新しい表定義でそのように定義されていなければ、その列は行変更タイム・スタンプ列になりません。

ID 列は、表に行が挿入されるときに列に関する値を自動生成する列です。

ソース表で ID 列が見つかった場合、表移動操作が完了した後のこの列の値は、表移動操作の前に存在した値と同じになります。ただし、ソース表での ID 列の「前の/次の」値を判別する方法はありません。このため、ターゲット表で ID 列を作成する際、次の「未変更」値から値の生成が始まるように設定されます。これはデータベース再始動 (停止/開始) のときに発生する動作と同じです。この動作は、インフォメーション・センターの『ALTER TABLE』という項目の『identity-alteration』セクションの『SET NO CACHE または CACHE integer-constant』という見出しで説明されています。

列はターゲット表で最初に通常の列として作成された後、SWAP フェーズの短いオフライン期間中に ID 列に変更されます。こうする理由は、列が "GENERATED

ALWAYS" として作成された可能性があるためです。そのような場合、ストアード・プロシージャーはソース表の実際の値をターゲット表の列に正確に挿入できません。

新しい表定義が指定される場合、列が新しい表定義で ID 列として指定されていれば、ストアード・プロシージャーは ID 列の定義がソース表の列の定義と一致するかどうか検査します。一致する場合には、ストアード・プロシージャーは上記の説明のような処理を続けます。一致しない場合、ストアード・プロシージャーは新しい ID 列定義を使用します。この場合、ID 列カウンターは指定された開始値から再び開始されることに注意してください。ただし、列にある行の現在の値は引き続き同じです。

新しい表定義が指定され、ソース表で ID 列と指定されている列が新しい表定義では ID 列と指定されていない場合には、ストアード・プロシージャーはその列を引き続き ID 列としてターゲット表に作成します (その際、ソース表で見つかった仕様と同じものが使われます)。これにより、ユーザーは既存の ID 列の定義を検索して新しい表定義にそれを再び入力する必要がなくなります。ユーザーがこの列を ID 列として保持したくない場合には、ストアード・プロシージャーの呼び出し後にターゲット表を変更して、列の ID 仕様を除去することができます。

式列は、表に行が挿入されるときに、式に基づいて列の値を自動生成する列です。

ソース表で式列が見つかった場合、表移動操作が完了した後のこの列の値は、表移動操作の前に存在した値と同じになります。

列はターゲット表で最初に通常の列として作成された後、SWAP フェーズの短いオフライン期間中に式列に変更されます。こうする理由は、式列が "GENERATED ALWAYS" として作成されて、その列への挿入が禁止されるためです。ただし、ターゲット表の列を式列に変更するために、ターゲット表の保全性の設定が一時的にオフになります。 ALTER ステートメントが実行された後、"GENERATED COLUMN IMMEDIATE UNCHECKED" オプションを使って保全性が元どおりオンに設定されます。

表名を含む列式 (例えば表 T1 での式 (T1.C *5)) は、ソース表とターゲット表のどちらの場合も、ストアード・プロシージャーによってサポートされません。この解決策として、ユーザーは列を修正して、表名を含まないように式を変更することができます。

新しい表定義が指定される場合、列が新しい表定義で式列として指定されていれば、ストアード・プロシージャーは基本的なストリング比較を実行することにより、式列の定義がソース表の列の定義と一致するかどうか検査します。一致する場合には、ストアード・プロシージャーは上記の説明のような処理を続けます。一致しない場合、ストアード・プロシージャーは新しい式列の定義を使用します。列にある行の現在の値は引き続き同じであることに注意してください。

新しい表定義が指定され、ソース表で式列と指定されている列が新しい表定義では式列と指定されていない場合には、ストアード・プロシージャーはその列を引き続き式列としてターゲット表に作成します (その際、ソース表で見つかった仕様と同じものが使われます)。これにより、ユーザーは既存の式列の定義を検索して新しい表定義にそれを再び入力する必要がなくなります。ユーザーがこの列を式列として

保持したくない場合には、ストアード・プロシージャーの呼び出し後にターゲット表を変更して、列の式仕様を除去することができます。

オンライン表移動と、保持されるオブジェクトおよび特権

表移動が実行されるとき、ストアード・プロシージャーは以下のオブジェクトを保持します。

ビュー SWAP フェーズでの短いオフライン期間中に、ビューはソース表からドロップされてターゲット表に再作成されます。

また、ビューの所有権を元の所有者に戻すために、所有権の転送も実行されます。

トリガー

SWAP フェーズでの短いオフライン期間中に、トリガーはソース表からドロップされてターゲット表に再作成されます。

また、トリガーの所有権を元の所有者に戻すために、所有権の転送も実行されます。

索引 表移動の操作中には、ターゲット表に索引が何度か作成されます。まず、索引は COPY フェーズの終わりに作成されます。ただし COPY_WITH_INDEXES オプションが設定されている場合には、COPY フェーズの開始時に索引が最初に作成されます。さらに、その後の REPLAY および SWAP フェーズの開始時に、ストアード・プロシージャーは新しく作成される索引があるかどうか検索します。その際、索引名だけから判断します。新しい索引が見つかった場合、それが作成されます。ただし、ストアード・プロシージャーはソース表からいずれかの索引が削除されたかどうかを確認しません。

ユーザー作成の索引の場合、索引名はソース表と同じになります。ただし、システム作成の索引が同じ名前になるかどうかは保証されません。

保持される索引のタイプは、'REG'、'CLUST'、および 'XVIL' です。

ターゲット表でドロップされる列を参照するユーザー作成の索引は、保持されません。

ソース・パーティション表からターゲット・パーティション表に移動する場合、索引のパーティション属性は保持されます。パーティション化されたソース表から非パーティション化されたターゲット表に移動する場合、またはその逆の場合には、データベースのデフォルト動作によってパーティション属性が決まります。

制約 (参照制約を除く) 制約は、同じ制約名を使ってターゲット表で再作成されます。ただし、ユニーク制約と主制約の場合、基礎となる索引名はソース表の索引名と異なる可能性があります。

表フラグ

ソース表の表フラグは、INIT フェーズでターゲット表が作成されると直ちにターゲット表に作成されます。該当するフラグは 'append_mode'、'locksize'、 'volatile'、 'compression'、 'datacapture'、 'pctfree'、'logindexbuild'、 'owner'、 'clustered'、および 'droprule' です。その後、これ

らのフラグは COPY フェーズの終わりと SWAP フェーズ中に検査されます。フラグの内容が変更されている場合は、ターゲット表でそれが更新されます。

付与/取り消し

SWAP フェーズ中に、ストアード・プロシージャーは SYSCAT.TABAUTH 内の項目を調べて、表に関するユーザー/グループ/ロールへの特権付与を複製します。

ストアード・プロシージャーの呼び出し元が ACCESSCTRL 権限または SECADM 権限をどちらも持っていない場合、CONTROL 特権を付与することはできません。 CONTROL 特権を付与されなかったすべてのユーザー/グループ/ロールのリストは、プロトコル表の WARNINGS キーの部分にあります。

データベースの auto_revalidation が有効で、しかも USE_AUTO_REVAL オプション (auto_revalidation が有効である場合のデフォルト) が設定されている場合、ビューは上記のようにドロップされないことに注意してください。代わりに、ビューは保持されて、auto_revalidation によって再検証されます。現在、サブジェクトとして定義されたトリガーを持つ表の名前変更に関して制限があるため、トリガーがストアード・プロシージャーによってドロップされて、再作成されます。

索引によるクラスタリングを伴うオンライン表移動

ターゲット表を索引によってクラスター化することが可能です。ソース表にクラスター索引が存在する場合、デフォルトでは、その索引によってクラスター化されます。 INIT フェーズ後にこのデフォルトを変更することができます (これはフェーズ全体にわたるオンライン表移動の実行を意味します)。クラスター索引が存在しない 1 つの MOVE フェーズでオンライン表移動を呼び出した場合、ストアード・プロシージャーはユニーク/主索引を使ってターゲット表をクラスター化します。クラスター索引が存在する場合、ストアード・プロシージャーはクラスター索引を使ってターゲット表をクラスター化します。

ソース表にクラスター索引が存在する場合、ターゲット表をクラスター索引でクラスター化しないことも可能です。そうするには、複数ステップから成る移動を実行して INIT フェーズ後にキー項目 "COPY_INDEXSCHEMA" および "COPY_INDEXNAME" をプロトコル表から削除します。

任意の副次索引によってターゲット表をクラスター化できます。そうするには、複数ステップから成る移動を実行し、適切な索引を使ってプロトコル表のキー項目 "COPY_INDEXSCHEMA" および "COPY_INDEXNAME" を挿入/更新することで、ターゲット表をクラスター化します。

索引属性の変更

既存のいずれかの属性を変更する必要が生じた場合 (例えば索引のクラスタリング、索引の圧縮、グローバル索引からローカル索引への変更、またはその逆への変更)、ユーザーは複数ステップから成る移動操作中に手動で属性を変更できます。

これを行うには、複数ステップから成る移動で INIT および COPY 移動フェーズを実行します。その後、ターゲット表の索引を手動で変更します。ターゲット表の名

前はプロトコル表の中にあります。変更が完了した後、REPLAY および SWAP フェーズから再開します。

制約事項

ADMIN_MOVE_TABLE ストアード・プロシージャーには、以下の制限が適用されます。

- ソース表としてサポートされるのは単純な表のみです。マテリアライズ照会表、型付き表、クラスター表、システム表、ビュー、ニックネーム、別名は許可されません。
- イベント・モニターが現在アクティブになっている表を移動することはできません。
- 外部キー (参照制約) は、親も子もサポートされていません。外部キーを持つ表を移動する場合には、db2look コマンドを使用して外部キーをキャプチャーしてから、外部キーをドロップし、移動操作を実行し、それからキーを再作成することができます。
- ユニーク索引のない表は、複雑で高コストになる可能性がある再生フェーズの対象となります。
- 表に LOB、XML、または LONG 列がある場合は、ユニーク索引が必要になります。
- 生成された列を MDC 仕様の一部にすることはできません。
- テキスト検索索引に対するサポートはありません。
- ディスク・スペース所要量が大きいことに気を付けてください。このプロシージャーは表と索引の 2 種類のコピーを作成するほか、ステージング表とログ・スペースも作成します。
- データのほとんどが「選択から挿入」の形式を使用して新しい表に移動されるので、コピー・パフォーマンスが問題になる可能性があります。
- ユニーク索引のない表に対する VERIFY 操作は、LOB のある表では機能しません。
- DB2 バージョン 9.7 フィックスパック 2 より前のリリースでは、*DB2_SKIPDELETED* レジストリー変数を ON に設定できません。
- SYSTOOLSPACE 表スペースが作成されていて、「PUBLIC」でアクセスできなければなりません。
- ソース表に対する長時間の実行トランザクションのために、COPY フェーズでロック・タイムアウトになる可能性があります。
- SWAP フェーズでデッドロックが発生する可能性があります。
- ソース表に非ユニーク索引があり、更新処理がいくつも行われると、デッドロックが発生する可能性があります。
- VARCHAR2 サポートが有効になっている場合、データベースは空ストリングと NULL を等価の値として扱いますが、単一ブランクは別個の値です。VARCHAR2 サポートが有効になっている場合、*mdc_cols*、*partkey_cols*、および *data_part* パラメーターは、単一ブランクを空ストリングおよび NULL とは異なるものとして使用することになります。
- SET INTEGRITY PENDING 状態の表は、移動できません。

戻される情報

表 251. ADMIN_MOVE_TABLE ストアード・プロシージャーによって戻される情報

列名	データ・タイプ	説明
TABSCHEMA	VARCHAR(128)	移動対象の表のスキーマ。システム規模のデフォルトでは空ストリング。
TABNAME	VARCHAR(128)	移動対象の表の表名。システム規模のデフォルトでは空ストリング。
KEY	VARCHAR(32)	属性の名前。
VALUE	CLOB(10M)	属性の値。

結果セットで戻されるキーと値のペアを、表 252に示しています。結果セット内のユーザー構成可能キーを変更するには、ADMIN_MOVE_TABLE_UTIL ストアード・プロシージャーを使用します。

表 252. ADMIN_MOVE_TABLE ストアード・プロシージャーによって戻されるキーと値のペア

キー	戻り値	ユーザー構成可能
VERSION	ストアード・プロシージャーのバージョンを表示します。	いいえ
AUTHID	ストアード・プロシージャーを呼び出したユーザーの許可 ID を表示します。	いいえ
LOCK	別のオンライン表移動ストアード・プロシージャー呼び出しがアクティブの場合に、ロック開始時刻を表示します。それ以外の場合は空です。	いいえ
STATUS	オンライン表移動の現在の状況を表示します。 • INIT: INIT が進行中。 • COPY: COPY が進行中であるか COPY が見込まれる。 • REPLAY: REPLAY が進行中であるか REPLAY と SWAP が見込まれる。 • CLEANUP: MOVE が完了しているが、CLEANUP が完了していないか CLEANUP が見込まれる。 • COMPLETE: MOVE と CLEANUP が完了している。 • COMPLETE_WITH_WARNINGS: MOVE と CLEANUP が完了しているが、警告がある (WARNINGS キーのフィールドにリストされる)。	いいえ
STAGING	ステージング表の名前を表示します。	いいえ
TARGET	ターゲット表の名前を表示します。	いいえ
ORIGINAL	スワップ後の元の表の名前を表示します。	いいえ
INDEXSCHEMA	索引のスキーマを表示します。表に索引がない場合は空ストリングになります。	いいえ
INDEXNAME	索引の名前を表示します。表に索引がない場合は空ストリングになります。	いいえ

表 252. *ADMIN_MOVE_TABLE* ストアード・プロシージャーによって戻されるキーと値のペア *(続き)*

キー	戻り値	ユーザー構成可能
COMMIT_AFTER_N_ROWS	この数の行がコピーされると COPY フェーズでコミットが実行されます。 0 は COPY 中にコミットが行われないことを意味します。デフォルト値は 10000 です。	はい
DEEPCOMPRESSION_SAMPLE	このフィールドは、ソース表の圧縮が有効になっている場合に、圧縮用ディクショナリーの作成時にサンプリングするデータ量 (KB 単位) を指定します。 0 はサンプリングが行われないことを意味します。デフォルト値は 20MB (20480 KB) です。	はい
COPY_ARRAY_SIZE	COPY_ARRAY_INSERT の配列サイズを指定します。 0 以下の値は、COPY_ARRAY_INSERT を使用しないことを意味します。デフォルト値は 100 です。	はい
COPY_OPTS	COPY フェーズで使用されるコピー・オプション。	いいえ
COPY_INDEXSCHEMA	COPY フェーズでターゲット表のデータのクラスター化に使用される索引のスキーマ。この値は、COPY フェーズの前に設定する必要があります。ソース表のクラスター索引が存在する場合は、そのスキーマ名がデフォルト・スキーマになります。存在しない場合は、ソース表のユニーク索引または 1 次索引のスキーマ名がデフォルト・スキーマになります。	はい
COPY_INDEXNAME	COPY フェーズでターゲット表のデータのクラスター化に使用される索引の名前。この値は、COPY フェーズの前に設定する必要があります。ソース表のクラスター索引が存在する場合は、その名前がデフォルトの名前になります。存在しない場合は、ソース表のユニーク索引または 1 次索引の名前がデフォルトの名前になります。	はい
INDEX_CREATION_TOTAL_TIME	副次索引の作成に要した合計時間を表示します。	いいえ
INIT_START	INIT フェーズの開始時刻を表示します。	いいえ
INIT_END	INIT フェーズの終了時刻を表示します。	いいえ
COPY_START	COPY フェーズの開始時刻を表示します。	いいえ
COPY_END	COPY フェーズの終了時刻を表示します。	いいえ
COPY_TOTAL_ROWS	COPY フェーズでコピーされた行の総数を表示します。	いいえ
REPLAY_START	REPLAY フェーズの開始時刻を表示します。	いいえ
REPLAY_END	REPLAY フェーズの終了時刻を表示します。	いいえ
REPLAY_TOTAL_ROWS	再生された行の累積数を表示します。	いいえ
REPLAY_TOTAL_TIME	行の再生に使用された累積時間 (秒単位) を表示します。	いいえ
REPLAY_MAX_ERR_RETRIES	REPLAY フェーズでエラー (ロック・タイムアウトまたはデッドロック) が発生した場合の最大再試行回数を指定します。デフォルト値は 100 です。	はい

表 252. *ADMIN_MOVE_TABLE* ストアード・プロシージャーによって戻されるキーと値のペア *(続き)*

キー	戻り値	ユーザー構成可能
REPLAY_THRESHOLD	REPLAY フェーズの単一反復について、ステージング表に適用される行数がこの値より小さければ、REPLAY は停止します。この動作は、その間に新しい項目が作成されたとしても変わりません。デフォルト値は 100 です。	はい
REORG_USE_TEMPSPACE	REORG オプションを呼び出す場合は、REORG コマンドの USE 節に TEMPORARY 表スペースを指定することもできます。この値が指定されない場合、REORG コマンドは再編成される表と同じ表スペースを使用します。	はい
VERIFY_START	検査の開始時刻を表示します。	いいえ
VERIFY_END	検査の終了時刻を表示します。	いいえ
SWAP_START	SWAP フェーズの開始時刻を表示します。	いいえ
SWAP_END	SWAP フェーズの終了時刻を表示します。	いいえ
SWAP_MAX_RETRIES	SWAP フェーズにおける (ロック・タイムアウトまたはデッドロックが発生した場合の) 再試行の許容最大回数を指定します。デフォルト値は 10 です。	はい
SWAP_RETRIES	SWAP フェーズで行われた再試行の回数を表示します。	いいえ
CLEANUP_START	CLEANUP フェーズの開始時刻を表示します。	いいえ
CLEANUP_END	CLEANUP フェーズの終了時刻を表示します。	いいえ
WARNINGS	ユーザーに伝える警告を表示します。この警告には、以下の種類があります。 • 失敗したすべてのオブジェクトの再確認。 • ユーザー、グループ、またはロールに制御権を付与できなかった。 • 索引が参照する列がもう存在しないために、索引が作成されなかった。	いいえ

ADMIN_MOVE_TABLE_UTIL プロシージャー - オンライン表移動の変更プロシージャー

ADMIN_MOVE_TABLE_UTIL プロシージャーは、アクティブ表データの移動時に、SYSPROC.ADMIN_MOVE_TABLE ストアード・プロシージャーと連動します。このストアード・プロシージャーは、ADMIN_MOVE_TABLE プロシージャーによって作成および使用される ADMIN_MOVE_TABLE プロトコル表のユーザー定義可能値を変更するための手段として使用できます。

このプロシージャーが ADMIN_MOVE_TABLE プロトコル表の値を変更するのは、TABSCHEMA および TABNAME パラメーターが参照する表の表移動が既に進行中であり、プロシージャーの呼び出し元の許可 ID が表移動を実行しているユーザーと同じである場合のみです。

構文

このストアード・プロシージャーのスキーマは SYSPROC です。

プロシージャー・パラメーター

tabschema
この入力パラメーターは、移動中の表が含まれるスキーマの名前を指定します。この名前には、大/小文字の区別があります。データ・タイプは VARCHAR(128) です。

tabname
この入力パラメーターは、移動中の表の名前を指定します。このパラメーターには大/小文字の区別があり、データ・タイプは VARCHAR(128) です。

action
この入力パラメーターは、プロシージャーの実行アクションを指定します。

有効な値は以下のとおりです。

- UPSERT: 指定した TABSCHEMA.TABNAME.KEY が ADMIN_MOVE_TABLE プロトコル表に存在する場合は、対応する値が新しい *value* パラメーターで更新されます。その他の場合は、ADMIN_MOVE_TABLE プロトコル表にキーと値のペアが挿入されます。
- DELETE: 指定した TABSCHEMA.TABNAME.KEY が ADMIN_MOVE_TABLE プロトコル表に存在する場合は、指定したキーと値のペアが ADMIN_MOVE_TABLE プロトコル表から削除されます。

このパラメーターのデータ・タイプは VARCHAR(128) です。

key
この入力パラメーターは、ADMIN_MOVE_TABLE プロトコル表で「UPSERT」または削除を行うためのキーを指定します。

有効な値は以下のとおりです。

- COMMIT_AFTER_N_ROWS: この数の行がコピーされると COPY フェーズでコミットが実行されます。値 0 は、COPY 中にコミットを実行しないことを意味します。
- DEEPCOMPRESSION_SAMPLE: このフィールドは、ソース表の圧縮が有効になっている場合に、圧縮用ディクショナリーの作成時にサンプリングするデータ量 (KB 単位) を指定します。値 0 は、サンプリングを行わないことを意味します。
- COPY_ARRAY_SIZE: COPY_ARRAY_INSERT の配列サイズを指定します。0 以下の値は、COPY_ARRAY_INSERT を使用しないことを意味します。
- COPY_INDEXSCHEMA: COPY フェーズでターゲット表のデータのクラスター化に使用する索引のスキーマ。
- COPY_INDEXNAME: COPY フェーズでターゲット表のデータのクラスター化に使用する索引の名前。
- REPLAY_MAX_ERR_RETRIES: REPLAY フェーズでエラー (ロック・タイムアウトまたはデッドロック) が発生した場合の最大再試行回数を指定します。

- REPLAY_THRESHOLD: REPLAY フェーズの単一反復について、ステージング表に適用される行数がこの値より小さければ、REPLAY は停止します。この動作は、その間に新しい項目が作成されたとしても変わりません。
- REORG_USE_TEMPSPACE: 表移動で REORG オプションを呼び出す場合は、REORG コマンドの USE 節に TEMPORARY 表スペースを指定することもできます。この値が指定されない場合、REORG コマンドは再編成される表と同じ表スペースを使用します。
- SWAP_MAX_RETRIES: SWAP フェーズにおける (ロック・タイムアウトまたはデッドロックが発生した場合の) 再試行の許容最大回数を指定します。

このパラメーターのデータ・タイプは VARCHAR(128) です。

value
この入力パラメーターは、ADMIN_MOVE_TABLE プロトコル表に「UPSERT」する値を指定します。このパラメーターのデータ・タイプは CLOB(10M) です。このパラメーターを NULL または空ストリングにすることができます。

許可

このストアード・プロシージャーを呼び出すための明示的な権限は不要です。ただし、使用する許可 ID は、ADMIN_MOVE_TABLE ストアード・プロシージャーの呼び出しに使用されたものと同じでなければなりません。

例

この例では、このストアード・プロシージャーの基本的な呼び出しを取り上げています。ここでは、圧縮の値を更新し、ターゲット表のコピー処理に使用される特定の索引情報を除去するために呼び出します。

まず、ADMIN_MOVE_TABLE プロシージャーを呼び出して表移動プロセスを開始します。その後で、ADMIN_MOVE_TABLE プロトコル表の値を更新または削除するために、ADMIN_MOVE_TABLE_UTIL プロシージャーを呼び出します。

```
CALL SYSPROC.ADMIN_MOVE_TABLE('SVALENTI','T1','','','','','','','','','INIT')
```

次に、DEEP_COMPRESSION_SAMPLE 値を 30720 KB に更新します。

```
CALL SYSPROC.ADMIN_MOVE_TABLE_UTIL('SVALENTI','T1','UPSERT',
     'DEEPCOMPRESSION_SAMPLE','30720')
```

ここで、COPY_INDEXSCHEMA と COPY_INDEXNAME の値を削除します。

```
CALL SYSPROC.ADMIN_MOVE_TABLE_UTIL('SVALENTI','T1','DELETE','COPY_INDEXSCHEMA','')
CALL SYSPROC.ADMIN_MOVE_TABLE_UTIL('SVALENTI','T1','DELETE','COPY_INDEXNAME','')
```

これらの変更後、メタ・テーブル内の新しい値を使用して ADMIN_MOVE_TABLE プロシージャーを続行します。

```
CALL SYSPROC.ADMIN_MOVE_TABLE('SVALENTI','T1','','','','','','','','','COPY')
CALL SYSPROC.ADMIN_MOVE_TABLE('SVALENTI','T1','','','','','','','','','REPLAY')
CALL SYSPROC.ADMIN_MOVE_TABLE('SVALENTI','T1','','','','','','','','','SWAP')
```

使用上の注意

ADMIN_MOVE_TABLE プロトコル表の変更可能キー値についての詳細が、ADMIN_MOVE_TABLE プロシージャーの『使用上の注意』のセクションに記載してあります。

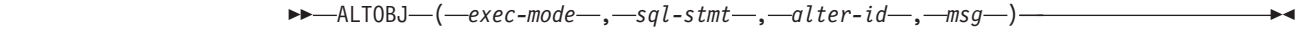
ALTOBJ

ALTOBJ プロシージャーは、変更される既存の表のターゲット・データ定義言語 (DDL) として機能する、入力 CREATE TABLE ステートメントを解析します。このプロシージャーは、変更される表のデータのバックアップを取ってから、元の表をドロップし、DDL ステートメントを使用して新しいバージョンを作成します。最終ステップでは、保管データを新しい表にロードして戻します。

このプロシージャーでは、次の表変更操作がサポートされていて、リカバリー可能な従属関係が維持されています。

- 列の名前変更
- 列のサイズの増減
- 列タイプの変更と DB2 スカラー関数を使用した既存データのトランスフォーム
- 10 進数値の精度または位取りの変更
- 列のデフォルト値の変更
- 列の NULL 可能属性の NULL 可能への変更
- 列のドロップ

構文



スキーマは SYSPROC です。

プロシージャー・パラメーター

exec-mode
次の実行モードのいずれかを指定する、タイプ VARCHAR(30) の入力引数。

'GENERATE'
VALIDATE、APPLY、および UNDO モードで必要なすべてのスクリプトを生成することを指定します。

'VALIDATE'
ステートメント構文が検査されることを指定します。このオプションにより、変更される表の関連オブジェクトおよびリレーションシップの処理を管理するスクリプトも生成されます。

'APPLY_CONTINUE_ON_ERROR' または 'APPLY_STOP_ON_ERROR'
変更される表の関連オブジェクトおよびリレーションシップの処理を管理するスクリプトを生成することを指定します。元の表のデータは、新しい表を取り込むために、エクスポートされ、トランスフォームされて使用されます。

'UNDO'
生じた可能性のあるエラーをロールバック操作でリカバリーできない場合に、表変更操作によって加えられた変更を取り消すことを指定します。このモードは、元の表および生成されたスクリプトが削除されていない場合にのみ有効です。

'FINISH'
名前変更された元の表をドロップすることを指定します。

sql-stmt
既存の表を変更するときのテンプレートとして使用される CREATE TABLE ステートメントを指定する、タイプ VARCHAR(2048) の入力引数。 *exec-mode* が 'GENERATE' である場合、*sql-stmt* は NULL 値であってはなりません。それ以外の場合、*sql-stmt* を NULL 値にすることはできますが、 *alter-id* が -1 ではない場合だけです。

alter-id
この呼び出しによって生成されるすべてのステートメントを識別する、タイプ INTEGER の入力および出力引数。 -1 が指定される場合、新しい ID が生成されて呼び出し元に戻されます。指定した整数で識別される既存のステートメントがあれば、それらは上書きされます。

msg
指定した実行モード下で、表変更プロセス用に生成されるか表変更プロセスで使用されるすべての SQL ステートメントを表示するために実行できる SQL 照会を含む、タイプ VARCHAR(2048) の出力引数。

許可

ALTOBJ プロシージャーに対する EXECUTE 特権

LOAD 権限を備えた DBADM。SETSESSIONUSER も必要です。

例

例 1: ALTOBJ プロシージャーを実行して、表 T1 の列 CL2 をタイプ INTEGER からタイプ BIGINT に変更します。表 T1 の元のデータ定義言語は、次のとおりです。

```
CREATE TABLE T1 (CL1 VARCHAR(5), CL2 INTEGER)
```

列データを変更するための ALTOBJ プロシージャー呼び出しは、次のとおりです。

```
CALL SYSPROC.ALTOBJ('APPLY_CONTINUE_ON_ERROR',
   'CREATE TABLE T1 (CL1 VARCHAR(5), CL2 BIGINT)', -1, ?)
```

注: 次のエラーが表示された場合は、APPLHEAPSZ パラメーターの値を増やしてください。
SQL0443N ルーチン "SYSPROC.ALTOBJ" (特定名 "ALTOBJ") が、診断テキスト "SQL0954" とともにエラー SQLSTATE を返しました。SQLSTATE=38553

例 2: ALTOBJ プロシージャーを、*alter-id* 入力と共に VALIDATE モードで実行します。

```
CALL SYSPROC.ALTOBJ('VALIDATE', CAST (NULL AS VARCHAR(2048)), 123, ?)
```

使用上の注意

このプロシージャーは表をドロップして再作成しますが、元の表を作成したユーザーは表の定義者として残ります。しかし、監査では、このプロシージャーを実行しているユーザーが表をドロップして再作成したと示されます。

このプロシージャーは、以下の表変更操作はサポートしません。

- マテリアライズ照会表 (MQT) の変更はサポートされていません。 MQT の入っている表の変更はサポートされます。
- 型付き表の変更はサポートされていません。
- ニックネームを使用したリモート表の変更は、サポートされていません。
- 列のシーケンスを再配列することはできません。
- 列の追加と除去、または名前変更と除去を、プロシージャーの 1 回の呼び出しで行うことはサポートされていませんが、列の追加と名前変更はサポートされています。その理由は、表の変更方法を指示する唯一の方法が、列一致情報ではなく、ターゲット DDL を使用する方法であるからです。既存の表から変更後の表へデータを変換するときには、ALTOBJ プロシージャーは次のような規則に従います。
  1. 既存の表中の列数が、変更後の表と同じである場合、列の追加または除去は行われないとみなされます。この場合、列の名前変更のみが可能であり、列索引によって突き合わせられます。
  2. 既存の表中の列数が、変更後の表より少ない場合、列の追加が行われるとみなされます。列の名前変更が可能であり、新規の列が末尾に追加されます。既存の列は、索引によって突き合わせられます。
  3. 既存の表中の列数が、変更後の表より多い場合、列の除去が行われるとみなされます。列は、名前変更不能になり、名前で突き合わせられます。ドロップする対象の列は、表中の任意の既存の列でかまいません。
- 構造化タイプ UDT および参照タイプの UDT は、サポートされていません。
- 変更される基本表上に定義された MQT が、表の変更プロセス中にデータを追加されることはありません。

ALTOBJ プロシージャーを使用して表を変更した場合に、その表に MQT が定義されていると、MQT は作成されますが、そこにはデータは取り込まれません。

ALTOBJ プロシージャーを使用して表を変更した場合に、その表に MQT が定義されていると、変更された表での選択結果に属さないすべての列は失われてしまいます。なぜなら、MQT の内容は、新規の基本表から再作成されるからです。

オブジェクトの定義は、ALTOBJ プロシージャーを次に呼び出したときには変わっていることがあります。セッションからセッションへ持続するオブジェクト・ロックはないからです。

表に関連付けられた表プロファイル (RUNSTATS プロファイルなど) は、このような広範囲にわたるプロセスを実行する過程で失われてしまいます。

SYSTOOLSPACE をルーチンの操作表に使用してメタデータを保管します。つまり、データベース・オブジェクトとその操作の記述に使用されるデータです。

APPLICATION_ID

APPLICATION_ID 関数は、現行接続のアプリケーション ID を戻します。この結果のデータ・タイプは VARCHAR(128) です。

この関数から戻される値は 100 年のインターバル内で固有であり、関数を呼び出す前に確立された接続の期間中だけ有効です。

構文

スキーマは SYSFUN です。

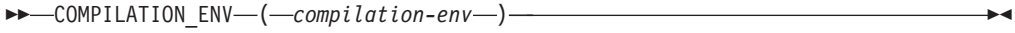
例

```
SELECT APPLICATION_ID() AS APPL_ID FROM SYSIBM.SYSDUMMY1
```

COMPILATION_ENV 表関数 - コンパイル環境のエレメントの検索

COMPILATION_ENV 表関数は、コンパイル環境のエレメントを戻します。

構文



スキーマは SYSPROC です。

表関数パラメーター

compilation-env
comp_env_desc (コンパイル環境) モニター・エレメントから得られる、コンパイル環境が入った、タイプ BLOB(2M) の入力引数。

関数は、NAME VARCHAR(256) および VALUE VARCHAR(1024) という 2 つの列 (1083 ページの表 253を参照) がある表を戻します。コンパイル環境のエレメント名として可能な値を、 1084 ページの表 254で説明します。

エレメント値の提供元は主に、SQL ステートメントが動的に発行されるか、またはパッケージの一部としてバインドされるかに応じて異なります。

コンパイル環境の項目の数とタイプは、DB2 データベース・マネージャーの機能の追加に応じ、時の経過と共に変わる可能性があります。コンパイル環境が、この関数が実行されているのとは異なる DB2 データベース・マネージャー・レベルのものである場合、そのレベルの関数によって認識されるエレメントだけが戻されます。エレメントの記述は、リリースごとに異なる場合もあります。

例

例 1: デッドロック・イベント・モニターによって以前にキャプチャーされた特定のコンパイル環境のすべてのエレメントを要求します。 WITH DETAILS HISTORY

オプションを指定して作成されたデッドロック・イベント・モニターは、動的 SQL ステートメントのコンパイル環境をキャプチャーします。このキャプチャー環境は、表関数への入力として受け入れられるものです。

```
SELECT NAME, VALUE
  FROM TABLE(SYSPROC.COMPILATION_ENV(:hv1)) AS t
```

例 2: コンパイル環境の特定のエレメント (デフォルトのスキーマ) を要求します。

```
SELECT NAME, VALUE
  FROM TABLE(SYSPROC.COMPILATION_ENV(:hv1)) AS t
  WHERE NAME = 'SCHEMA'
```

例 3: パッケージ・キャッシュ内の特定のステートメントのコンパイル環境を表示します。

1. 実行可能 ID を取得します。これは、以下のステートメントを使って、対象のステートメントを識別するために使用されます。

   ```
   SELECT EXECUTABLE_ID, VARCHAR{STMT_TEXT, 100)
     FROM TABLE(MON_GET_PKG_CACHE_STMT(NULL,NULL,NULL,-1)) AS t
   ```

   以下は、上記ステートメントを実行した後の出力例です。

```
EXECUTABLE_ID                                                       2
------------------------------------------------------------------- --------------------------------------------------
x'0100000000000000010000000000000000000000020020090914151405241700' select count(*) from syscat.tables
...
```

2. ステートメント (実行可能 ID を使って識別される) のコンパイル環境を調べ、COMPILATION_ENV 表関数を使ってコンパイル環境をフォーマットします。以下のステートメントは、これを行う方法を示す例です。

   ```
   SELECT VARCHAR(NAME, 30), VARCHAR(VALUE, 50)
     FROM TABLE(COMPILATION_ENV((SELECT COMP_ENV_DESC FROM TABLE
       (MON_GET_PKG_CACHE_STMT(NULL,
        x'0100000000000000010000000000000000000000020020090914151405241700',
        NULL, -1)) AS t))) AS s
   ```

   以下は、上記ステートメントを実行した後の出力例です。

   ```
   1                              2
   ------------------------------ --------------------------------------------------
   ISOLATION                      CS
   QUERY_OPTIMIZATION             5
   MIN_DEC_DIV_3                  NO
   DEGREE                         1
   SQLRULES                       DB2
   REFRESH_AGE                    +00000000000000.000000
   RESOLUTION_TIMESTAMP           2009-09-14-15.14.05.000000
   FEDERATED_ASYNCHRONY           0
   PATH                           "SYSIBM","SYSFUN","SYSPROC","SYSIBMADM","SWALKTY"
   MAINTAINED_TABLE_TYPE          SYSTEM

     10 record(s) selected.
   ```

戻される情報

表 253. COMPILATION 表関数によって戻される情報

列名	データ・タイプ	説明
NAME	VARCHAR(256)	コンパイル環境のエレメント。詳しくは、 1084 ページの表 254を参照してください。

表 253. COMPILATION 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明
VALUE	VARCHAR(1024)	エレメントの値。

表 254. COMPILATION_ENV 表関数によって戻されるコンパイル環境のエレメント

エレメント名	説明
ISOLATION	SQL コンパイラーに渡される分離レベル。この値は、現行のパッケージの CURRENT ISOLATION 特殊レジスターまたは ISOLATION BIND オプションから取得されます。
QUERY_OPTIMIZATION	SQL コンパイラーに渡される照会最適化レベル。この値は、現行のパッケージの CURRENT QUERY OPTIMIZATION 特殊レジスターまたは QUERYOPT BIND オプションから取得されます。
MIN_DEC_DIV_3	SQL コンパイラーに渡される、要求された 10 進計算スケール。この値は、*min_dec_div_3* データベース構成パラメーターから取得されます。
DEGREE	SQL コンパイラーに渡される、要求された内部並列処理の度合い。この値は、現行のパッケージの CURRENT DEGREE 特殊レジスターまたは DEGREE BIND オプションから取得されます。
SQLRULES	SQL コンパイラーに渡される、要求された SQL ステートメントの動作。この値は、現行のパッケージの LANGLVL BIND オプションの設定から導出されます。可能な値は「DB2」または「SQL92」です。
REFRESH_AGE	SQL コンパイラーに渡される、許容データ待ち時間。この値は、現行のパッケージの CURRENT REFRESH AGE 特殊レジスターまたは REFRESHAGE BIND オプションから取得されます。
SCHEMA	SQL コンパイラーに渡されるデフォルトのスキーマ。この値は、現行のパッケージの CURRENT SCHEMA 特殊レジスターまたは QUALIFIER BIND オプションから取得されます。
PATH	SQL コンパイラーに渡される関数パス。この値は、現行のパッケージの CURRENT PATH 特殊レジスターまたは FUNC_PATH BIND オプションから取得されます。
TRANSFORM_GROUP	SQL コンパイラーに渡されるトランスフォーム・グループ情報。この値は、CURRENT DEFAULT TRANSFORM GROUP 特殊レジスターまたは TRANSFORMGROUP パッケージ BIND オプションから取得されます。
MAINTAINED_TABLE _TYPE	SQL コンパイラーに渡される、どの表タイプを最適化の検討対象にできるかを示すインディケーター。この値は、CURRENT MAINTAINED TABLE TYPES FOR OPTIMIZATION 特殊レジスターから取得されます。
RESOLUTION _TIMESTAMP	SQL ステートメント内の関数やデータ・タイプ参照などの項目の解決のために、SQL コンパイラーによって使用されるタイム・スタンプ。このタイム・スタンプは、現行のタイム・スタンプか、または現行のパッケージの最新の明示的なバインド操作のタイム・スタンプです。

表 254. COMPILATION_ENV 表関数によって戻されるコンパイル環境のエレメント (続き)

エレメント名	説明
FEDERATED _ASYNCHRONY	SQL コンパイラーに渡される、要求されたフェデレーテッド非同期並列処理の度合い。この値は、現行のパッケージの CURRENT FEDERATED ASYNCHRONY 特殊レジスターまたは FEDERATED_ASYNCHRONY BIND オプションから取得されます。

CONTACTGROUPS 管理ビュー - 連絡先グループのリストの検索

CONTACTGROUPS 管理ビューは連絡先グループのリストを戻します。これはシステム上にローカルに定義されている場合も、グローバルなリストに定義されている場合もあります。Database Administration Server (DAS) の CONTACT_HOST 構成パラメーターの設定により、リストがローカルかグローバルかが決まります。

スキーマは SYSIBMADM です。

許可

以下のいずれかの権限が必要です。

- CONTACTGROUPS 管理ビューに対する SELECT 特権
- CONTACTGROUPS 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- ADMIN_GET_CONTACTGROUPS 表関数に対する EXECUTE 特権
- DATAACCESS 権限

例

すべての連絡先グループのリストを検索します。

```
SELECT * FROM SYSIBMADM.CONTACTGROUPS
```

以下はこの照会の出力例です。

```
NAME          DESCRIPTION                    MEMBERNAME        MEMBERTYPE
-------...--- -----------------------...--- -----------...--- ----------
group1        DBA Group1 Contact List        name1             CONTACT
group1        DBA Group1 Contact List        name9             CONTACT
group2        DBA Group2 List                name2             CONTACT
group3                                       group2            GROUP
group5        DBA Group5                     group2            GROUP
group6        DBA Group6                     group3            GROUP
group7                                       name1             CONTACT

  7 record(s) selected.
```

使用上の注意

DAS が作成されていて実行中でなければなりません。

戻される情報

表 255. CONTACTGROUPS 管理ビューによって戻される情報

列名	データ・タイプ	説明
NAME	VARCHAR(128)	連絡先グループの名前。
DESCRIPTION	VARCHAR(128)	連絡先グループの説明。
MEMBERNAME	VARCHAR(128)	連絡先グループのメンバーの名前。この名前は 1 つの連絡先を指す場合もあれば、別の連絡先グループを指す場合もあります。
MEMBERTYPE	VARCHAR(7)	連絡先グループのメンバーのタイプ。タイプは CONTACT または GROUP のいずれかです。

CONTACTS 管理ビュー - 連絡先のリストの検索

CONTACTS 管理ビューは、データベース・サーバーで定義されている連絡先のリストを戻します。Database Administration Server (DAS) の CONTACT_HOST 構成パラメーターの設定により、リストがローカルかグローバルかが決まります。

スキーマは SYSIBMADM です。

許可

以下のいずれかの権限が必要です。

- CONTACTS 管理ビューに対する SELECT 特権
- CONTACTS 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- ADMIN_GET_CONTACTS 表関数に対する EXECUTE 特権
- DATAACCESS 権限

例

すべての連絡先を検索します。

```
SELECT * FROM SYSIBMADM.CONTACTS
```

以下はこの照会の出力例です。

```
NAME      TYPE  ADDRESS              MAX_PAGE_LENGTH DESCRIPTION
------...- ----- -----------------...- --------------- ------------------...-
user1     EMAIL user3@ca.ibm.com                   - DBA Extraordinaire
user2     EMAIL user2@ca.ibm.com                   - DBA on Email
user3     PAGE  user3@ca.ibm.com                 128 DBA on Page
user5     EMAIL user2@ca.ibm.com                   - DBA Extraordinaire

  4 record(s) selected.
```

使用上の注意

DAS が作成されていて実行中でなければなりません。

戻される情報

表 256. CONTACTS 管理ビューによって戻される情報

列名	データ・タイプ	説明
NAME	VARCHAR(128)	連絡先の名前。
TYPE	VARCHAR(5)	連絡先のタイプ • 'EMAIL' • 'PAGE'
ADDRESS	VARCHAR(128)	受信側の SMTP メールボックス・アドレス。たとえば、joe@somewhere.org などです。
MAX_PAGE_LENGTH	INTEGER	メッセージの最大長。例えば、ポケットベル呼び出しサービスにメッセージ長の制限がある場合に使用されます。
DESCRIPTION	VARCHAR(128)	連絡先の説明。

DB_HISTORY 管理ビュー - 履歴ファイル情報の検索

DB_HISTORY 管理ビューは、すべてのデータベース・パーティションの履歴ファイルからの情報を戻します。

データベース・パーティションに PRUNE HISTORY コマンドを使用して、DB_HISTORY ビューで戻される情報量を削減できます。また、LIST HISTORY コマンドを使用して、選択したデータベース・パーティションの履歴情報を取り出すこともできます。

スキーマは SYSIBMADM です。

許可

以下のいずれかの権限が必要です。

- DB_HISTORY 管理ビューに対する SELECT 特権
- DB_HISTORY 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- ADMIN_LIST_HIST 表関数に対する EXECUTE 特権
- DATAACCESS 権限

使用上の注意

データ・パーティション表が再編成されるとき、再編成対象のそれぞれのデータ・パーティションに対して 1 つのレコードが戻されます。データ・パーティション表

の特定の 1 つのデータ・パーティションだけが再編成される場合、そのパーティションのレコードだけが戻されます。

例

データベース・パーティション番号、項目 ID、操作、開始時刻、および状況情報を、クライアントが現在接続しているデータベース・パーティションすべてのデータベース履歴ファイルから選択します。

```
SELECT DBPARTITIONNUM, EID, OPERATION, START_TIME, ENTRY_STATUS
   FROM SYSIBMADM.DB_HISTORY
```

以下はこの照会の出力例です。

```
DBPARTITIONNUM EID                   OPERATION START_TIME     ENTRY_STATUS
-------------- -------------------- --------- -------------- ------------
             0                    1 A         20051109185510 A

  1 record(s) selected.
```

戻される情報

表 257. DB_HISTORY 管理ビューによって戻される情報

列名	データ・タイプ	説明
DBPARTITIONNUM	SMALLINT	データベース・パーティション番号。
EID	BIGINT	履歴ファイル内の項目を一意的に識別する番号。
START_TIME	VARCHAR(14)	ログに記録されたイベントの開始を示すタイム・スタンプ。
SEQNUM	SMALLINT	シーケンス番号。
END_TIME	VARCHAR(14)	ログに記録されたイベントの終了を示すタイム・スタンプ。
FIRSTLOG	VARCHAR(254)	イベントに関連した最も初期のトランザクション・ログの名前。
LASTLOG	VARCHAR(254)	イベントに関連した最新のトランザクション・ログの名前。
BACKUP_ID	VARCHAR(24)	バックアップ ID または固有の表 ID。
TABSCHEMA	VARCHAR(128)	表のスキーマ。
TABNAME	VARCHAR(128)	表名。
COMMENT	VARCHAR(254)	ログに記録されたイベントに関連する、システム生成のコメント・テキスト。
CMD_TEXT	CLOB(2 M)	ログに記録されたイベントに関連するデータ定義言語。
NUM_TBSPS	INTEGER	ログに記録されたイベントに関連する表スペースの数。

表 257. DB_HISTORY 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明
TBSPNAMES	CLOB(5 M)	ログに記録されたイベントに関連する表スペースの名前。
OPERATION	CHAR(1)	操作 ID。考えられる値については、 1091 ページの表 258 を参照してください。
OPERATIONTYPE	CHAR(1)	操作のアクション ID。考えられる値については、 1091 ページの表 258を参照してください。
OBJECTTYPE	CHAR(1)	操作のターゲット・オブジェクトの ID。使用可能な値は、D (全データベースの場合)、P (表スペースの場合)、T (表の場合)。
LOCATION	VARCHAR(255)	ログに記録されたイベントに関連する、バックアップ・イメージやロード入力ファイルなどのファイルの絶対パス名。
DEVICETYPE	CHAR(1)	ログに記録されたイベントに関連するデバイス・タイプ ID。このフィールドにより、LOCATION フィールドを解釈する方法が決まります。可能な値は、A (TSM)、C (クライアント)、D (ディスク)、F (スナップショット・バックアップ)、K (ディスケット)、L (ローカル)、N (DB2 により内部生成される)、O (その他)(他のベンダー・デバイスのサポート用)、P (パイプ)、Q (カーソル)、R (リモート・フェッチ・データ)、S (サーバー)、T (テープ)、U (ユーザー出口)、および X (X/Open XBSA インターフェース) です。
ENTRY_STATUS	CHAR(1)	履歴ファイル内の項目の状況 ID。可能な値は、A (アクティブ)、D (削除済み (将来の利用))、E (期限切れ)、I (非アクティブ)、N (まだコミットされていない)、Y (コミット済みまたはアクティブ) です。

表 257. DB_HISTORY 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明
SQLCAID	VARCHAR(8)	SQL 連絡域 (SQLCA) の SQLCAID フィールドに現れるとおりの、'SQLCA' の入ったストレージ・ダンプの「目印」。
SQLCABC	INTEGER	SQLCA の SQLCABC フィールドに現れるとおりの SQLCA の長さ。
SQLCODE	INTEGER	SQLCA の SQLCODE フィールドに現れるとおりの SQL 戻りコード。
SQLERRML	SMALLINT	SQLCA の SQLERRML フィールドに現れるとおりの SQLERRMC の長さ標識。
SQLERRMC	VARCHAR(70)	SQLCA の SQLERRMC フィールドに現れるとおり、X'FF' で区切られた 1 つ以上のトークンが入ります。これらのトークンは、エラー条件の説明の中の変数を置き換えます。
SQLERRP	VARCHAR(8)	SQLCA の SQLERRP フィールドに現れるとおり、製品を示す 3 文字の ID の後に、製品のバージョン、リリース、および修正レベルを示す 5 文字の英数字が続きます。
SQLERRD1	INTEGER	「*SQL リファレンス 第 1 巻*」の『SQLCA (SQL 連絡域)』を参照。
SQLERRD2	INTEGER	「*SQL リファレンス 第 1 巻*」の『SQLCA (SQL 連絡域)』を参照。
SQLERRD3	INTEGER	「*SQL リファレンス 第 1 巻*」の『SQLCA (SQL 連絡域)』を参照。
SQLERRD4	INTEGER	「*SQL リファレンス 第 1 巻*」の『SQLCA (SQL 連絡域)』を参照。
SQLERRD5	INTEGER	「*SQL リファレンス 第 1 巻*」の『SQLCA (SQL 連絡域)』を参照。
SQLERRD6	INTEGER	「*SQL リファレンス 第 1 巻*」の『SQLCA (SQL 連絡域)』を参照。

表 257. *DB_HISTORY* 管理ビューによって戻される情報 (続き)

列名	データ・タイプ	説明
SQLWARN	VARCHAR(11)	一連の警告標識で、それぞれの標識はブランクか 'W' です。「*SQL リファレンス 第 1 巻*」の『SQLCA (SQL 連絡域)』を参照。
SQLSTATE	VARCHAR(5)	SQLCA の SQLSTATE フィールドに現れるとおりの、直前に実行された SQL ステートメントの結果を示す戻りコード。

表 258. *OPERATION* および *OPERATIONTYPE* 値

操作値	操作値の説明	操作タイプ
A	表スペースの追加	なし
B	バックアップ	操作タイプは、以下のとおりです。 • D = 差分オフライン • E = 差分オンライン • F = オフライン • I = 増分オフライン • N = オンライン • O = 増分オンライン
C	ロード・コピー	なし
D	ドロップ済み表	なし
F	ロールフォワード	操作タイプは、以下のとおりです。 • E = ログの終わり • P = ポイント・イン・タイム
G	表の再編成	操作タイプは、以下のとおりです。 • F = オフライン • N = オンライン
L	ロード	操作タイプは、以下のとおりです。 • I = 挿入 • R = 置換
N	表スペースの名前変更	なし
O	表スペースのドロップ	なし

表 258. *OPERATION* および *OPERATIONTYPE* 値 (続き)

操作値	操作値の説明	操作タイプ
Q	静止	操作タイプは、以下のとおりです。 • S = 静止共用 • U = 静止更新 • X = 静止排他 • Z = 静止リセット
R	リストア	操作タイプは、以下のとおりです。 • F = オフライン • I = 増分オフライン • N = オンライン • O = 増分オンライン • R = 再ビルド
T	表スペースの変更	操作タイプは、以下のとおりです。 • C = コンテナーの追加 • R = リバランス
U	アンロード	なし
X	アーカイブ・ログ	操作タイプは、以下のとおりです。 • F = 障害アーカイブ・パス • M = ミラー・ログ・パス • N = ARCHIVE LOG コマンドを介しての強制切り捨て • P = 1 次ログ・パス • 1 = 第 1 ログ・アーカイブ方法 • 2 = 第 2 ログ・アーカイブ方法

DBPATHS 管理ビュー - データベース・パスの検索

DBPATHS 管理ビューは、分割ミラー・バックアップなどのタスクに必要なデータベース・パスの値を戻します。

スキーマは SYSIBMADM です。

許可

以下のいずれかの権限が必要です。

- DBPATHS 管理ビューに対する SELECT 特権
- DBPATHS 管理ビューに対する CONTROL 特権

- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- ADMIN_LIST_DB_PATHS 表関数に対する EXECUTE 特権
- DATAACCESS 権限

例

すべてのデータベース・パスを検索します。

```
SELECT * FROM SYSIBMADM.DBPATHS
```

以下はこの照会の出力例です。

```
DBPARTITIONNUM TYPE                    ...
-------------- ----------------------...
             0 LOGPATH                 ...
             0 MIRRORLOGPATH           ...
             0 DB_STORAGE_PATH         ...
             0 DB_STORAGE_PATH         ...
             0 TBSP_CONTAINER          ...
             0 TBSP_CONTAINER          ...
             0 TBSP_CONTAINER          ...
             0 TBSP_DIRECTORY          ...
             0 TBSP_DIRECTORY          ...
             0 LOCAL_DB_DIRECTORY      ...
             0 DBPATH                  ...

  11 record(s) selected.
```

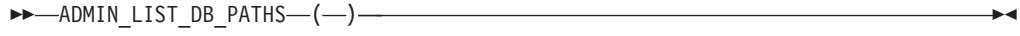
この照会の出力 (続き)。

```
... PATH
... -------------------------------------------------
... S:¥dbfiles¥INST5¥NODE0000¥SQL00001¥SQLOGDIR¥
... S:¥mirrorlogs¥NODE0000¥
... S:¥dbfiles¥
... S:¥dbfile2¥
... S:¥dbfiles¥INST5¥NODE0000¥SQL00001¥TS3
... S:¥dbfiles¥INST5¥NODE0000¥SQL00001¥long3
... S:¥dbfiles¥INST5¥NODE0000¥SQL00001¥regular05
... S:¥dbfiles¥INST5¥NODE0000¥SQL00001¥usertemp3¥
... S:¥dbfiles¥INST5¥NODE0000¥SQL00001¥systemp3¥
... S:¥dbfiles¥INST5¥NODE0000¥SQLDBDIR¥
... S:¥dbfiles¥INST5¥NODE0000¥SQL00001¥
```

ADMIN_LIST_DB_PATHS 表関数

ADMIN_LIST_DB_PATHS 表関数は、分割ミラー・バックアップなどのバックアップ・メカニズムに必要なファイルのリストを戻します。

構文



スキーマは SYSPROC です。

許可

以下のいずれかの権限が必要です。

- ADMIN_LIST_DB_PATHS 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

次のようにして ADMIN_LIST_DB_PATHS 表関数を呼び出すことができます。

```
SELECT DBPARTITIONNUM, TYPE, PATH FROM TABLE(ADMIN_LIST_DB_PATHS()) AS FILES
```

以下はこの照会の出力例です。

```
DBPARTITIONNUM    TYPE             PATH
------------------------------------------------------------------------------------------
0                 TBSP_CONTAINER   C:¥tablespaces¥dms¥dms1
0                 TBSP_CONTAINER   C:¥tablespaces¥dms¥dms2
1                 TBSP_CONTAINER   C:¥tablespaces¥dms¥dms3
1                 TBSP_DIRECTORY   D:¥tablespaces¥sms¥sms1¥
2                 TBSP_DIRECTORY   D:¥tablespaces¥sms¥sms2¥
2                 TBSP_DIRECTORY   D:¥tablespaces¥sms¥sms3¥
0                 LOGPATH          C:¥DB2¥NODE0000¥SQL00004¥SQLOGDIR¥
0                 DBPATH           C:¥DB2¥NODE0000¥SQL00004¥
1                 LOGPATH          C:¥DB2¥NODE0001¥SQL00004¥SQLOGDIR¥
1                 DBPATH           C:¥DB2¥NODE0001¥SQL00004¥
2                 LOGPATH          C:¥DB2¥NODE0002¥SQL00004¥SQLOGDIR¥
2                 DBPATH           C:¥DB2¥NODE0002¥SQL00004¥
```

分割ミラー操作を実行するストレージ・ライブラリーが、通常のファイル・システムとは異なる方法でロー・デバイス上のファイルとディレクトリーを扱う場合、以下の 2 つの照会を使用して、最初にロー・デバイス上のすべての場所のリストを取得することができます。

```
SELECT DBPARTITIONNUM, TYPE, PATH FROM TABLE(ADMIN_LIST_DB_PATHS()) AS FILES
   WHERE TYPE LIKE '%_DEVICE%'
```

次に、通常のファイル・システム上のファイルとディレクトリーのリストを以下のようにして取得します。

```
SELECT DBPARTITIONNUM, TYPE, PATH FROM TABLE(ADMIN_LIST_DB_PATHS()) AS FILES
    WHERE TYPE NOT LIKE '%_DEVICE%'
```

戻される情報

表 259. DBPATHS 管理ビューおよび ADMIN_LIST_DB_PATHS 表関数によって戻される情報

列名	データ・タイプ	説明
DBPARTITIONNUM	SMALLINT	データベース・パーティション番号。

表 259. DBPATHS 管理ビューおよび ADMIN_LIST_DB_PATHS 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明
TYPE	VARCHAR(64)	パスが属するデータベース・オブジェクトのタイプを記述します。例えば、LOGPATH データベース構成パラメーターが示すログ・ディレクトリーへのパスは、この列で LOGPATH と表示されます。考えられる戻り値のリストについては、表 260を参照してください。
PATH	VARCHAR(5000)	データベース・マネージャーのファイルまたはディレクトリーが見つかった場所へのパス。パスがファイル・システム区切り文字 (UNIX 環境では '/'、Windows 環境では '¥') で終わる場合、パスはディレクトリーを指します。

表 260. TYPE 列の値

タイプ値	説明
TBSP_DEVICE	データベース管理スペース (DMS) 表スペースのロー・デバイス。
TBSP_CONTAINER	DMS 表スペースのファイル・コンテナー。
TBSP_DIRECTORY	システム管理スペース (SMS) 表スペースのディレクトリー。
LOGPATH	1 次ログ・パス。
LOGPATH_DEVICE	1 次ログ・パスのロー・デバイス。
MIRRORLOGPATH	データベース構成ミラー・ログ・パス。
DB_STORAGE_PATH	自動ストレージ・パス。
DBPATH	データベース・ディレクトリー・パス。
LOCAL_DB_DIRECTORY	ローカル・データベース・ディレクトリーへのパス。

- 自動ストレージを使用する表スペースの場合、使用済みおよび未使用のストレージ・パスの両方が戻されます。未使用の自動ストレージ・パスは、分割ミラー・バックアップがリストアされる場合に必要になります。次の例を考慮してください。実動システムで分割ミラー・バックアップを取ります。バックアップ完了後に、バックアップ前は未使用だった自動ストレージ・パスが実動で使用されるようになります。ここで、分割ミラー・バックアップをリストアする必要が生じるとします。この時点で、実動データベースからログをロールフォワードする必要があります。ログをロールフォワードするには、自動ストレージ・パスがすべて必要です。現在、すべての自動ストレージ・パスが使用中だからです。

- 自動ストレージの管理下にある表スペース・コンテナーは、個別には戻されません。それらは自動ストレージ・パス列に反映されます。
- 自動ストレージ・パスはデータベース・パーティションごとに 1 回ずつ戻されます。
- LOGPATH および MIRRORLOGPATH に戻される値は、メモリー内に保管されている値です。ディスクに保管されている、変更された値 (データベースの再始動後に初めて適用される) は、戻されません。
- SELECT * FROM SYSIBMADM.DBPATHS からの出力を使用して db2relocatedb 構成ファイル (データベースの再配置に必要な構成情報を記載したファイル) を作成する場合、それに応じて DBPATH 出力を修正してから、構成ファイル内で使用する必要があります。

  例えば、以下のような DBPATH 出力があるとします。

  ```
  /storage/svtdbm3/svtdbm3/NODE0000/SQL00001/
  ```

  上記を使用して、次のように、db2relocatedb 構成ファイル内に DB_PATH パラメーターを指定することができます。

  ```
  DB_PATH=/storage/svtdbm3,/storage_copy2/svtdbm3
  ```
- LOCAL_DB_DIRECTORY パスには、複数のデータベースに属する情報を入れることができます。同じディレクトリー内に作成された複数のデータベースの場合、sqldbdir は分離されないので、ファイルのコピー先のターゲット・システムに、そのパスにすでに存在するデータベースがないことを確認してください。
- 複数のデータベースが、少なくとも 1 つの自動ストレージ・パスを共有する場合、それらのデータベースのいずれかに対して分割ミラー操作を実行すると、複数のデータベースがその影響を受け、それによって、分割の予定のなかったデータベースで入出力上の問題が起きることがあります。

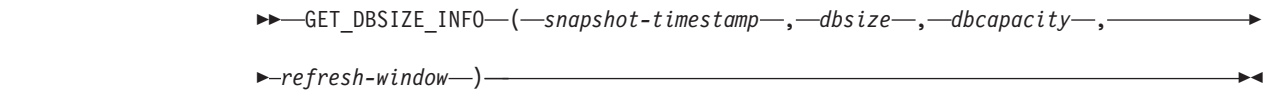
制約事項

データベースが WRITE SUSPEND モードのときにはこの管理ビューを呼び出せません。データベース管理者は、ビューを呼び出すときと WRITE SUSPEND モードでアクティブ化するときでデータベースの物理レイアウトが変わらないようにしなければなりません。これは分割ミラー操作を実行するために必要です。例えばその時に表スペースのレイアウトが変更されると、分割ミラー・バックアップ・イメージは正しくリストアされない可能性があります。

GET_DBSIZE_INFO

GET_DBSIZE_INFO プロシージャーは、データベース・サイズと最大容量を計算します。

構文



スキーマは SYSPROC です。

プロシージャー・パラメーター

snapshot-timestamp
dbsize および *dbcapacity* が計算された時刻を戻す、タイプ TIMESTAMP の出力パラメーター。このタイム・スタンプは、*refresh-window* の値と共に、SYSTOOLS.STMG_DBSIZE_INFO 表のキャッシュ値を更新しなければならない時刻を判別するのに使用されます。

dbsize
データベースのサイズを (バイト単位で) 戻すタイプ BIGINT の出力パラメーター。データベースのサイズは、各表スペース (SMS および DMS) ごとに、次のように計算されます。*dbsize* = sum (使用されているページ数 * ページ・サイズ)。

dbcapacity
データベース容量を (バイト単位で) 戻すタイプ BIGINT の出力パラメーター。この値は、パーティション・データベース・システムでは使用できません。データベースの容量は、次のように計算されます。*dbcapacity* = SUM (DMS 使用可能ページ数 * ページ・サイズ) + SUM (SMS コンテナー・サイズ + コンテナーあたりのファイル・システムの空きサイズ)。同じファイル・システムに複数の SMS コンテナーが定義されている場合、容量計算にファイル・システムの空きサイズが含められるのは 1 回だけです。

refresh-window
データベース・サイズおよび容量のキャッシュ値を更新するまでの分数を指定する、タイプ INTEGER の入力引数。 30 分のデフォルト更新枠の場合は -1 を指定します。更新枠が 0 であると、キャッシュ値の更新が即時に強制実行されます。

許可

- SYSMON 権限
- GET_DBSIZE_INFO プロシージャーに対する EXECUTE 特権。

例

例 1: 30 分のデフォルト更新枠を使用して、データベース・サイズおよび容量を入手します。データベース・サイズおよび容量は、キャッシュ・データが 30 分よりも前のものであると再計算されます。

```
CALL GET_DBSIZE_INFO(?, ?, ?, -1)
```

プロシージャーは、次を戻します。

```
 Value of output parameters
 --------------------------
 Parameter Name  : SNAPSHOTTIMESTAMP
 Parameter Value : 2004-02-29-18.31.55.178000

 Parameter Name  : DATABASESIZE
 Parameter Value : 22302720

 Parameter Name  : DATABASECAPACITY
 Parameter Value : 4684793856

 Return Status = 0
```

例 2: 0 分の更新枠を使用して、データベース・サイズおよび容量を入手します。データベース・サイズおよび容量がすぐに再計算されます。

```
CALL GET_DBSIZE_INFO(?, ?, ?, 0)
```

プロシージャーは、次を戻します。

```
Value of output parameters
--------------------------
Parameter Name  : SNAPSHOTTIMESTAMP
Parameter Value : 2004-02-29-18.33.34.561000

Parameter Name  : DATABASESIZE
Parameter Value : 22302720

Parameter Name  : DATABASECAPACITY
Parameter Value : 4684859392

 Return Status = 0
```

例 3: 24 時間の更新枠を使用して、データベース・サイズおよび容量を入手します。データベース・サイズおよび容量は、キャッシュ・データが 1440 分よりも前のものであると再計算されます。

```
CALL GET_DBSIZE_INFO(?, ?, ?, 1440)
```

プロシージャーは、次を戻します。

```
 Value of output parameters
 --------------------------
 Parameter Name  : SNAPSHOTTIMESTAMP
 Parameter Value : 2004-02-29-18.33.34.561000

 Parameter Name  : DATABASESIZE
 Parameter Value : 22302720

 Parameter Name  : DATABASECAPACITY
 Parameter Value : 4684859392

 Return Status = 0
```

使用上の注意

計算値は、プロシージャー出力パラメーターとして戻され、SYSTOOLS.STMG_DBSIZE_INFO 表にキャッシュされます。プロシージャーがこの値をキャッシュに入れるのは、計算にコストがかかるためです。SYSTOOLS.STMG_DBSIZE_INFO 表は、初めてプロシージャーを実行すると自動的に作成されます。 SYSTOOLS.STMG_DBSIZE_INFO 表にキャッシュされた値が存在し、それが最新であれば、 *snapshot-timestamp* および *refresh-window* 値で決定されたように、それらのキャッシュに入れられた値が戻されます。キャッシュに入れられた値が最新でない場合、キャッシュに入れられた新しい値が計算され、SYSTOOLS.STMG_DBSIZE_INFO 表に挿入されて戻され、*snapshot-timestamp* 値が更新されます。

グローバル表スペース・スナップショットのデータがすべてのパーティションから必ず戻されるようにするには、データベースをアクティブにしておく必要があります。

SYSTOOLSPACE をルーチンの操作表に使用してメタデータを保管します。つまり、データベース・オブジェクトとその操作の記述に使用されるデータです。

NOTIFICATIONLIST 管理ビュー - ヘルス通知の連絡先リストの検索

NOTIFICATIONLIST 管理ビューは、インスタンスの状況が通知される連絡先および連絡先グループのリストを戻します。

スキーマは SYSIBMADM です。

許可

以下のいずれかの権限が必要です。

- NOTIFICATIONLIST 管理ビューに対する SELECT 特権
- NOTIFICATIONLIST 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- HEALTH_GET_NOTIFICATION_LIST 表関数に対する EXECUTE 特権
- DATAACCESS 権限

例

ヘルス・アラートの通知を受信するすべての連絡先を検索します。

```
SELECT * FROM SYSIBMADM.NOTIFICATIONLIST
```

以下はこの照会の出力例です。

```
NAME                     TYPE
------------...-------- -------
group3                   GROUP
user4                    CONTACT
group3                   GROUP

  3 record(s) selected.
```

戻される情報

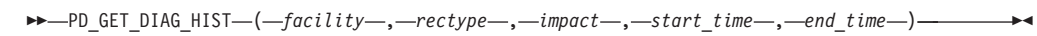
表 261. NOTIFICATIONLIST 管理ビューによって戻される情報

列名	データ・タイプ	説明
NAME	VARCHAR(128)	連絡先の名前。
TYPE	VARCHAR(7)	連絡先のタイプ • 'CONTACT' • 'GROUP'

PD_GET_DIAG_HIST - 指定された機能からレコードを戻す

PD_GET_DIAG_HIST 関数は、指定された機能からログ・レコード、イベント・レコード、および通知レコードを戻します。また、レコードのタイプ、レコードの顧客影響値、および from-until タイム・スタンプに基づいてフィルター操作を行うためのオプションもサポートされています。

構文



スキーマは SYSPROC です。

表関数パラメーター

facility

レコードが戻される機能を指定する、タイプ VARCHAR(20) のオプション入力引数。機能とは、レコードが関連付けられる論理グループです。可能な値は次のとおりです。

- ALL: すべての機能からレコードを戻します。
- MAIN: DB2 一般診断ログからレコードを戻します。現時点では、db2diag ログ・ファイル、管理通知ログ、および循環イベント・ログのことです。
- OPTSTATS: オプティマイザー統計に関連したレコードを戻します。

このパラメーターが NULL または空ストリング ('') の場合、「ALL」がデフォルトです。

rectype

戻すレコード・タイプを指定する、タイプ VARCHAR(30) のオプション入力引数。 '+' で区切られた組み合わせのタイプがサポートされます。例えば、'D + EI' などです。可能な値は次のとおりです。

- 「ALL」: すべてのレコード・タイプを戻します。
- 「D」: すべての診断レコードを戻します。
- 「E」: すべてのイベント・レコードを戻します。
- 「DI」: 内部診断レコード。これらは、診断状況で IBM サポートによって使用される翻訳されていない診断レコードです。
- 「DX」: 外部診断レコード。これらは、ユーザーに役立つ翻訳済みの診断です。これらのレコードは通知レコードです。
- 「EI」: 内部イベント・レコード。これらは、診断状況で IBM サポートによって使用されるイベント・レコードです。
- 「EX」: 外部イベント・レコード。これらは、ユーザーに役立つ診断レコードです。

このパラメーターが NULL または空ストリング ('') の場合、すべてのレコードが戻されます。

impact

戻されるレコードの最小の顧客影響レベルを指定する、タイプ VARCHAR(18) のオプション入力引数。可能な値は次のとおりです。

- 'NONE'
- 'UNLIKELY'
- 'POTENTIAL'
- 'IMMEDIATE'
- 'CRITICAL'

このパラメーターが NULL または空ストリング (") の場合、すべてのレコードが戻されます。

start_time
有効なタイム・スタンプを指定する、タイプ TIMESTAMP のオプション入力引数。項目のタイム・スタンプがこの値より新しい場合は、それらの項目が戻されます。このパラメーターが NULL または空ストリング (") の場合、項目がどのくらい古いかに関係なく、レコードが戻されます。

end_time
有効なタイム・スタンプを指定する、タイプ TIMESTAMP のオプション入力引数。項目のタイム・スタンプがこの値より古い場合は、それらの項目が戻されます。このパラメーターが NULL または空ストリング (") の場合、項目がどのくらい新しいかに関係なく、レコードが戻されます。

許可

PD_GET_DIAG_HIST 表関数に対する EXECUTE 特権。

例

```
SELECT FACILITY, RECTYPE, TIMESTAMP, IMPACT, SUBSTR(MSG,1, 50) AS MSG
FROM TABLE (PD_GET_DIAG_HIST( 'MAIN', 'E', '', CAST (NULL AS TIMESTAMP),
  CAST (NULL AS TIMESTAMP) ) ) AS T
WHERE T.PROCESS_NAME = 'db2star2' OR T.PROCESS_NAME = 'db2stop2'
```

以下はこの照会の出力例です。

```
FACILITY             RECTYPE TIMESTAMP                  ...
-------------------- ------- -------------------------- ...
MAIN                 EX      2007-06-25-11.34.05.756171 ...
MAIN                 EX      2007-06-25-11.34.25.946646 ...

  2 record(s) selected.
```

この照会からの出力 (続き)。

```
... IMPACT             MSG
... ------------------ --------------------------------------------------
... -                  ADM7514W  Database manager has stopped.
... -                  ADM7513W  Database manager has started.
```

戻される情報

表 262. PD_GET_DIAG_HIST 表関数によって戻される情報

列名	データ・タイプ	説明
FACILITY	VARCHAR(20)	機能とは、レコードが関連付けられる論理グループです。可能な値は次のとおりです。 • ALL: すべての機能からレコードを戻します。 • MAIN: DB2 一般診断ログからレコードを戻します。現時点では、db2diag ログ・ファイル、管理通知ログ、および循環イベント・ログのことです。 • OPTSTATS: オプティマイザー統計に関連したレコードを戻します。

表 262. PD_GET_DIAG_HIST 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明
RECTYPE	VARCHAR(3)	レコードのタイプ。可能な値は次のとおりです。 • 「DI」: 内部診断レコード • 「DX」: 外部診断レコード • 「EI」: 内部イベント・レコード • 「EX」: 外部イベント・レコード
TIMESTAMP	TIMESTAMP	メッセージが作成された時刻。
TIMEZONE	INTEGER	協定世界時 (UCT) との時差 (分単位)。例えば、-300 は米東部標準時です。
INSTANCENAME	VARCHAR(128)	メッセージが作成されたインスタンスの名前。
DBPARTITIONNUM	SMALLINT	メッセージが作成されたパーティションのパーティション番号。非パーティション・データベースの場合、0 が戻されます。
LEVEL	CHAR(1)	レコードの重大度レベル。可能な値は次のとおりです。 • 「C」: クリティカル • 「E」: エラー • 「I」: 通知 • 「S」: 重大 • 「W」: 警告
IMPACT	VARCHAR(18)	ユーザーの視点からこのメッセージの影響を限定します。これは、DB2 が構成要素となっているビジネス・プロセスに関するメッセージの影響を明確にします。可能な値は次のとおりです。 • 'CRITICAL' • 'IMMEDIATE' • 'NONE' • 'POTENTIAL' • 'UNLIKELY'
DBNAME	VARCHAR(128)	このメッセージの作成中にアクセスされているデータベースの名前。
EDU_ID	BIGINT	このメッセージを作成したエンジン・ディスパッチ可能単位 ID。
EDUNAME	VARCHAR(64)	このメッセージを作成したエンジン・ディスパッチ可能単位の名前。
PID	BIGINT	このメッセージを作成したオペレーティング・システム・プロセス ID。
PROCESS_NAME	VARCHAR(255)	このメッセージを作成したオペレーティング・システム・プロセス名。
TID	BIGINT	このメッセージを作成したスレッド数値 ID。
APPLNAME	VARCHAR(255)	接続を開始したクライアント・アプリケーションの名前 (使用可能な場合)。
APPL_ID	VARCHAR(64)	接続を開始したアプリケーション ID (使用可能な場合)。例えば、'G91A3955.F33A.02DD18143340' などです。
APPLHANDLE	VARCHAR(9)	使用可能な場合に接続を開始したアプリケーションの、システム全体におけるユニーク ID。これはエージェント ID と同義です。 ID は、コーディネーター・パーティション番号と 16 ビットのカウンターを '-' で区切ったもので構成されます。形式は、次のとおりです。'nnn-xxxxx'
AUTH_ID	VARCHAR(30)	プロセスのシステム許可 ID。

表 262. PD_GET_DIAG_HIST 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明
PRODUCT	VARCHAR(50)	メッセージを作成したプロダクトの名前。例えば、「DB2 Common」などです。
COMPONENT	VARCHAR(255)	メッセージを作成したコンポーネントの名前。
FUNCTION	VARCHAR(255)	メッセージを生成した関数の名前。
PROBE	INTEGER	メッセージが関数で生成された場所を識別するために使用されるプローブ・ポイント番号。
CALLEDPRODUCT	VARCHAR(50)	エラーのソースにあるプロダクトの名前。エラーのソースがメッセージが作成された場所ではない場合にこれが使用されます。
CALLEDCOMPONENT	VARCHAR(255)	エラーのソースにあるコンポーネントの名前。エラーのソースがメッセージが作成された場所ではない場合にこれが使用されます。
CALLEDFUNCTION	VARCHAR(255)	エラーのソースにある関数の名前。エラーのソースがメッセージが作成された場所ではない場合にこれが使用されます。
OSERR	INTEGER	オペレーティング・システム・エラー番号。
RETCODE	INTEGER	プロダクト固有の戻りコード。
MSGNUM	INTEGER	関連メッセージの数値メッセージ番号 (使用可能な場合)。例えば、これは ADM7513W の数値部分です。
MSGTYPE	CHAR(3)	メッセージ ID に関連したタイプ (使用可能な場合)。例えば、ADM は管理通知ログ・メッセージに使用されます。
MSG	CLOB(16KB)	このレコードの簡略説明テキスト。これは、翻訳済みメッセージの MSGNUM、および MSGTYPE に対応する翻訳済みメッセージ・テキストです。翻訳済みでないメッセージの場合、これは簡略説明です。例えば、「Bringing down all db2fmp processes as part of db2stop」のようになります。

表 262. PD_GET_DIAG_HIST 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明
OBJTYPE	VARCHAR(64)	イベントが適用されるオブジェクトのタイプ (使用可能な場合)。可能な値は次のとおりです。 • 'APM' • 'CATALOG CACHE ENTRY' • 'CFG' • 'CLI' • 'CLP' • 'CONTAINER' • 'COUNTER' • 'DAS' • 'DB2AGENT' • 'DB PART MAP ID' • 'DB PART NUM' • 'DBA' • 'DBM' • 'DMS' • 'DPS' • 'EDU' • 'EVALUATION' • 'EXTENDER' • 'FCM' • 'HISTOGRAM TEMPLATE' • 'INDEX STATS' • 'INITIAL SAMPLING' • 'REDIST DB PART GROUP' • 'REDIST TABLE' • 'RDS' • 'SAMPLING TEST' • 'SERVICE CLASS' • 'STATS' • 'STATS DAEMON' • 'TABLE' • 'TABLE STATS' • 'TABLE AND INDEX STATS' • 'THRESHOLD' • 'UDF' • 'WORK ACTION SET' • 'WORK CLASS SET' • 'WORKLOAD'
OBJNAME	VARCHAR(255)	イベントが関連するオブジェクトの名前 (使用可能な場合)。
OBJNAME_QUALIFIER	VARCHAR(255)	オブジェクトの追加情報 (使用可能な場合)。

表 262. PD_GET_DIAG_HIST 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明
EVENTTYPE	VARCHAR(24)	イベント・タイプは、このイベントに関連したアクションまたは verb です。可能な値は次のとおりです。 • 'ACCEPT' • 'ACCESS' • 'ADD' • 'ALTER' • 'ASSOCIATE' • 'AVAILABLE' • 'BRINGDOWN' • 'CHANGE' • 'CHANGECFG' • 'CLOSE' • 'COLLECT' • 'CONNECT' • 'CREATE' • 'DEPENDENCY' • 'DESTROY' • 'DISASSOCIATE' • 'DISCONNECT' • 'DISPATCH' • 'DROP' • 'FINI' • 'FREE' • 'GET' • 'INIT' • 'INTERRUPT' • 'OPEN','READ' • 'RECV' • 'REPLY' • 'REPORT' • 'REQUEST' • 'RESET' • 'SEND' • 'START' • 'STARTUP' • 'STOP' • 'SWITCH' • 'TERMINATE' • 'TRANSFER' • 'WAIT' • 'WORK' • 'WRITE'
EVENTDESC	VARCHAR(256)	このイベントのキー・フィールドの簡略表現。

表 262. *PD_GET_DIAG_HIST* 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明
FIRST_EVENTQUALIFIERTYPE	VARCHAR(64)	最初のイベント修飾子のタイプ。イベント修飾子は、イベントの影響を受けたものを説明するために使用されます。可能な値は次のとおりです。 • 'AT' • 'BY' • 'CONTEXT' • 'DUE TO' • 'FOR' • 'FROM' • 'ON' • 'TO' *facility* が OPTSTATS の場合、値は 'AT' のみです。
FIRST_EVENTQUALIFIER	CLOB(16K)	イベントの最初の修飾子。 *facility* が OPTSTATS の場合、これは統計収集が行われた時を示すタイム・スタンプになります。
SECOND_EVENTQUALIFIERTYPE	VARCHAR(64)	2 番目のイベント修飾子のタイプ。*facility* が OPTSTATS の場合、値は 'BY' です。
SECOND_EVENTQUALIFIER	CLOB(16K)	イベントの 2 番目の修飾子。 *facility* が OPTSTATS の場合、使用できる値は次のとおりです。 • Asynchronous • FABRICATE • FABRICATE PARTIAL • SYNCHRONOUS • SYNCHRONOUS SAMPLED • USER
THIRD_EVENTQUALIFIERTYPE	VARCHAR(64)	3 番目のイベント修飾子のタイプ。*facility* が OPTSTATS の場合、値は 'DUE TO' です。
THIRD_EVENTQUALIFIER	CLOB(16K)	イベントの 3 番目の修飾子。 *facility* が OPTSTATS の場合、使用できる値は次のとおりです。 • 競合 • エラー • 使用不可のオブジェクト • RUNSTATS エラー • タイムアウト
EVENTSTATE	VARCHAR(255)	イベントの結果としてのオブジェクトまたはアクションの状態。また、これにはイベントの進捗を示すパーセントも含めることができます。

表 262. PD_GET_DIAG_HIST 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明
EVENTATTRIBUTE	VARCHAR(255)	イベント属性。これはイベントに関連した属性のリストです。複数の属性が使用される場合、リストは '+' 文字で区切られます。例えば、'CACHED + LOGICAL + AUTO' などです。可能な値は次のとおりです。 • 'ASYNC' • 'AUTO' • 'CACHED' • 'DIRECT' • 'EXTERNAL' • 'INDIRECT' • 'INTERNAL' • 'LOGICAL' • 'PERMANENT' • 'PHYSICAL' • 'SYNC' • 'TEMPORARY'
EVENTSTACK	CLOB(16K)	該当する場合に、レコードが記録された地点の論理イベント・スタック。
CALLSTACK	CLOB(16K)	該当する場合に、このレコードを生成したスレッドのオペレーティング・システム・スタック・ダンプ。
DUMPFILE	CLOB(5000)	該当する場合に、ログ・レコードに関連した 2 次ダンプ・ファイルの名前。これはファイルへの絶対パス、またはメッセージに関連した追加情報を検索できるディレクトリーです。
FULLREC	CLOB(16K)	レコード全体のフォーマット済みテキスト・バージョン。このセクションには追加 DATA フィールドも含まれます。

PDLOGMSGS_LAST24HOURS 管理ビューおよび PD_GET_LOG_MSGS 表関数 - 問題判別メッセージの検索

PDLOGMSGS_LAST24HOURS 管理ビューおよび PD_GET_LOG_MSGS 表関数は、DB2 通知ログに記録された問題判別ログ・メッセージを戻します。この情報は、データベース管理者とシステム管理者が使用するためのものです。

PDLOGMSGS_LAST24HOURS 管理ビュー

PDLOGMSGS_LAST24HOURS 管理ビューは、DB2 通知ログに過去 24 時間に記録された問題判別ログ・メッセージを戻します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 1113 ページの表 263を参照してください。

許可

以下のいずれかの権限が必要です。

- PDLOGMSGS_LAST24HOURS 管理ビューに対する SELECT 特権

- PDLOGMSGS_LAST24HOURS 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- PD_GET_LOG_MSGS 表関数に対する EXECUTE 特権
- DATAACCESS 権限

例

過去 24 時間に記録されたすべての重大ログ・メッセージを、最新のものから順番に取得します。

```
SELECT * FROM SYSIBMADM.PDLOGMSGS_LAST24HOURS
   WHERE MSGSEVERITY = 'C' ORDER BY TIMESTAMP DESC
```

以下はこの照会の出力例です。

```
TIMESTAMP                  TIMEZONE    INSTANCENAME      ...
-------------------------- ----------- ----------------- ...
2005-11-23-21.56.41.240066        -300 svtdbm4           ...
                                                         ...
                                                         ...
                                                         ...
                                                         ...
                                                         ...
                                                         ...
2005-11-23-21.56.39.150597        -300 svtdbm4           ...
2005-11-23-21.56.37.363384        -300 svtdbm4           ...
                                                         ...
                                                         ...
                                                         ...
2005-11-23-21.56.35.880314        -300 svtdbm4           ...
                                                         ...


  4 record(s) selected.
```

この照会からの出力 (続き)。

```
...  DBPARTITIONNUM DBNAME        PID         PROCESSNAME          ...
...  -------------- ------------ ---...----- -------------------- ...
...               0 CAPTAIN           4239374 db2agent (CAPTAIN) 0 ...
...                                                                ...
...                                                                ...
...                                                                ...
...                                                                ...
...                                                                ...
...                                                                ...
...               0 CAPTAIN           4239374 db2agent (CAPTAIN) 0 ...
...               0 CAPTAIN           4239374 db2agent (CAPTAIN) 0 ...
...                                                                ...
...                                                                ...
...                                                                ...
...               0 CAPTAIN           4239374 db2agent (CAPTAIN) 0 ...
...                                                                ...
...                                                                ...
```

この照会からの出力 (続き)。

```
...TID APPL_ID                          COMPONENT            ...
...--- ------------------------------ -------------------- ...
...  1 9.26.15.148.36942.051124025612  oper system services  ...
...                                                         ...
...                                                         ...
```

```
...                                                                    ...
...                                                                    ...
...                                                                    ...
...                                                                    ...
...  1 9.26.15.148.36942.051124025612  base sys utilities              ...
...  1 9.26.15.148.36942.051124025612  relation data serv              ...
...                                                                    ...
...                                                                    ...
...                                                                    ...
...  1 9.26.15.148.36942.051124025612  relation data serv              ...
...                                                                    ...
...                                                                    ...
```

この照会からの出力 (続き)。

```
... FUNCTION           PROBE  MSGNUM      MSGTYPE ...
... ------------------ ------ ----------- ------- ...
... sqloSleepInstance      38         504 ADM     ...
...                                               ...
...                                               ...
...                                               ...
...                                               ...
...                                               ...
...                                               ...
... sqleMarkDBad           10        7518 ADM     ...
... sqlrr_dump_ffdc        10           1 ADM     ...
...                                               ...
...                                               ...
...                                               ...
... sqlrr_dump_ffdc        10           1 ADM     ...
...                                               ...
```

この照会からの出力 (続き)。

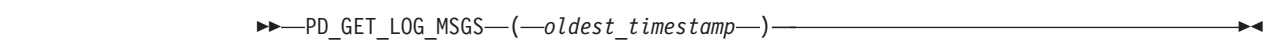
```
...  MSGSEVERITY MSG
...  ----------- -------------------------------------
...  C           ADM0504C  An unexpected internal
...              processing error has occurred.  ALL
...              DB2 PROCESSES ASSOCIATED WITH THIS
...              INSTANCE HAVE BEEN SUSPENDED.
...              Diagnostic information has been
...              recorded.  Contact IBM Support
...              for further assistance.
...  C           ADM7518C  "CAPTAIN " marked bad.
...  C           ADM0001C  A severe error has occurred.
...              Examine the administration notification
...              log and contact IBM Support if
...              necessary.
...  C           ADM0001C  A severe error has occurred.
...              Examine the administration notification
...              log and contact IBM Support if necessary.
```

PD_GET_LOG_MSGS 表関数

PD_GET_LOG_MSGS 表関数は PDLOGMSGS_LAST24HOURS 管理ビューと同じ情報を戻しますが、この関数では過去 24 時間に限らず、特定の時間枠を指定することが可能です。

戻される可能性のある情報の完全なリストは、1113 ページの表 263を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

oldest_timestamp
有効なタイム・スタンプを指定する、タイプ TIMESTAMP の入力引数。最新のタイム・スタンプの項目を先頭に、この入力引数に指定したタイム・スタンプまでのログ項目が戻されます。NULL 値が指定された場合、すべてのログ項目が戻されます。

許可

PD_GET_LOG_MSGS 表関数に対する EXECUTE 特権。

例

例 1: DB2 というインスタンス上のデータベース SAMPLE のすべてのデータベース・パーティションに関する、過去 1 週間にログに記録されたすべての通知メッセージを検索します。メッセージを日時順にレポートします。

```
SELECT TIMESTAMP, APPL_ID, DBPARTITIONNUM, MSG
   FROM TABLE ( PD_GET_LOG_MSGS( CURRENT_TIMESTAMP - 7 DAYS)) AS T
   WHERE INSTANCENAME = 'DB2' AND DBNAME = 'SAMPLE'
   ORDER BY TIMESTAMP ASC
```

以下はこの照会の出力例です。

```
TIMESTAMP                  APPL_ID                    DBPARTITIONNUM ...
-------------------------- -------------------------- -------------- ...
2005-11-13-12.51.37.772000 *LOCAL.DB2.050324175005                0 ...
                                                                    ...
2005-11-13-12.51.37.772001 *LOCAL.DB2.050324175005                0 ...
                                                                    ...
2005-11-13-12.51.37.781000 *LOCAL.DB2.050324175005                0 ...
                                                                    ...
2005-11-13-12.51.37.781001 *LOCAL.DB2.050324175005                0 ...
                                                                      ...
                                                                      ...
2005-11-17-14.12.39.036001 *LOCAL.DB2.041117191249                0 ...
2005-11-17-14.12.39.056000 *LOCAL.DB2.041117191249                0 ...
2005-11-17-14.13.04.450000 *LOCAL.DB2.041117191307                0 ...
2005-11-17-14.13.04.460000 *LOCAL.DB2.041117191307                0 ...
2005-11-17-14.18.29.042000 *LOCAL.DB2.041117190824                0 ...
...
...
...
```

この照会からの出力 (続き)。

```
... MSG
... ------------------------------------------------------------...--
... ADM5502W  The escalation of "143" locks on table
...    "SYSIBM  .SYSINDEXAUTH" to lock intent "X" was successful.
... ADM5502W  The escalation of "144" locks on table
...    "SYSIBM  .SYSINDEXES" to lock intent "X" was successful.
... ADM5502W  The escalation of "416" locks on table
...    "SYSIBM  .SYSINDEXCOLUSE" tolock intent "X" was successful.
```

```
... ADM5500W  DB2 is performing lock escalation.  The total
...    number of locks currently held is "1129", and the target
...    number of locks to hold is "564".
... ADM7506W  Database quiesce has been requested.
... ADM7507W  Database quiesce request has completed successfully.
... ADM7510W  Database unquiesce  has been requested.
... ADM7509W  Database unquiesce request has completed successfully.
... ADM4500W  A package cache overflow condition has occurred.  There
...    is no error but this indicates that the package cache has
...    exceeded the configured maximum size.  If this condition persists,
...    you may want to adjust the PCKCACHESZ DB configuration parameter.
```

例 2: DB2 というインスタンスのデータベース・パーティション 0 に関する、過去 1 日間にログに記録されたすべてのクリティカル・エラーを、最新のものから順にソートして検索します。

```
SELECT TIMESTAMP, DBNAME, MSG
   FROM TABLE (PD_GET_LOG_MSGS(CURRENT_TIMESTAMP - 1 DAYS)) AS T
   WHERE MSGSEVERITY = 'C' AND INSTANCENAME = 'DB2' AND
   DBPARTITIONNUM = 0
   ORDER BY TIMESTAMP DESC
```

以下はこの照会の出力例です。

```
TIMESTAMP                  DBNAME      MSG
-------------------------- ----------- ------------------------
2004-11-04-13.49.17.022000 TESTSBCS    ADM0503C  An unexpected
                                       internal processing error
                                       has occurred.  ALL DB2
                                       PROCESSES ASSOCIATED WITH
                                       THIS INSTANCE HAVE BEEN
                                       SHUTDOWN.  Diagnostic
                                       information has been
                                       recorded.  Contact IBM
                                       Support for further
                                       assistance.
2004-11-04-11.32.26.760000 SAMPLE      ADM0503C  An unexpected
                                       internal processing error
                                       has occurred.  ALL DB2
                                       PROCESSES ASSOCIATED WITH
                                       THIS INSTANCE HAVE BEEN
                                       SHUTDOWN.  Diagnostic
                                       information has been
                                       recorded.  Contact IBM
                                       Support for further
                                       assistance.

  2 record(s) selected.
```

例 3: *LOCAL.DB2.050927195337 というアプリケーション ID の DB2 プロセスによって過去 1 日の間に書き込まれたメッセージを検索します。

```
SELECT TIMESTAMP, MSG
   FROM TABLE (PD_GET_LOG_MSGS(CURRENT_TIMESTAMP - 1 DAYS)) AS T
   WHERE APPL_ID = '*LOCAL.DB2.050927195337'
```

以下はこの照会の出力例です。

```
TIMESTAMP                  MSG
-------------------------- ----------------------------------------

2005-06-27-21.17.12.389000 ADM4500W  A package cache overflow
                           condition has occurred.  There is no error
                           but this indicates that the package cache
                           has exceeded the configured maximum
                           size.  If this condition persists, you
```

```
                          may want to adjust the PCKCACHESZ DB
                          configuration parameter.
2005-06-27-18.41.22.248000 ADM4500W  A package cache overflow
                          condition has occurred.  There is no error
                          but this indicates that the package cache
                          has exceeded the configured maximum
                          size.  If this condition persists, you
                          may want to adjust the PCKCACHESZ DB
                          configuration parameter.
2005-06-27-12.51.37.772001 ADM5502W  The escalation of "143" locks
                          on table "SYSIBM  .SYSINDEXAUTH" to
                          lock intent "X" was successful.
2005-06-27-12.51.37.772000 ADM5502W  The escalation of "144" locks
                          on table "SYSIBM  .SYSINDEXES" to lock
                          intent "X" was successful.
2005-06-27-12.51.37.761001 ADM5502W  The escalation of "416" locks
                          on table "SYSIBM  .SYSINDEXCOLUSE" to
                          lock intent "X" was successful.

...
```

例 4: 通知ログ中のメッセージ ADM0504C のすべてのインスタンスを検索します。考慮の対象となるメッセージがタイム・スタンプで限定されていないことに注意してください。通知ログ・ファイルが非常に大きい場合、これはコストのかかる操作になる可能性があります。

```
SELECT TIMESTAMP, DBPARTITIONNUM, DBNAME, MSG
   FROM TABLE (PD_GET_LOG_MSGS(CAST(NULL AS TIMESTAMP))) AS T
   WHERE MSGNUM = 504 AND MSGTYPE = 'ADM' AND MSGSEVERITY = 'C'
```

以下はこの照会の出力例です。

```
TIMESTAMP                   DBPARTITIONNUM DBNAME       ...
--------------------------  -------------- -------------...
2005-11-23-21.56.41.240066               0 CAPTAIN      ...
...
...
...
...
...
...
...
...
...
```

この照会からの出力 (続き)。

```
... APPL_ID                          MSG
... ------------------------------- ------------------------
... 9.26.15.148.36942.051124025612   ADM0504C  An unexpected
...                                  internal processing error
...                                  has occurred.  ALL DB2
...                                  PROCESSES ASSOCIATED WITH
...                                  THIS INSTANCE HAVE BEEN
...                                  SUSPENDED.  Diagnostic
...                                  information has been
...                                  recorded.  Contact IBM
...                                  Support for further
...                                  assistance.
```

戻される情報

注: パーティション・データベース環境では、ログ・メッセージが戻される順序は保証できません。ログ・レコードの順序が重要である場合は、結果をタイム・スタンプでソートする必要があります。

表 263. PDLOGMSGS_LAST24HOURS 管理ビューおよび PD_GET_LOG_MSGS 表関数によって戻される情報

列名	データ・タイプ	説明
TIMESTAMP	TIMESTAMP	項目がログに記録された時刻。
TIMEZONE	INTEGER	協定世界時 (UCT) との時差 (分単位)。例えば、-300 は米東部標準時です。
INSTANCENAME	VARCHAR(128)	メッセージを生成したインスタンスの名前。
DBPARTITIONNUM	SMALLINT	メッセージを生成したデータベース・パーティション。非パーティション・データベース環境の場合、0 が戻されます。
DBNAME	VARCHAR(128)	エラーまたはイベントが発生したデータベース。
PID	BIGINT	メッセージを生成したプロセスのプロセス ID。
PROCESSNAME	VARCHAR(255)	メッセージを生成したプロセスの名前。
TID	BIGINT	メッセージを生成したプロセス内のスレッドの ID。
APPL_ID	VARCHAR(64)	プロセスが作業する対象のアプリケーションの ID。
COMPONENT	VARCHAR(255)	メッセージを提供している DB2 コンポーネントの名前。ユーザー・アプリケーションが db2AdminMsgWrite API を使用して書き込んだメッセージの場合、「User Application」が戻されます。
FUNCTION	VARCHAR(255)	メッセージを提供している DB2 関数の名前。ユーザー・アプリケーションが db2AdminMsgWrite API を使用して書き込んだメッセージの場合、「User Function」が戻されます。

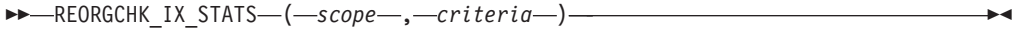
表 263. PDLOGMSGS_LAST24HOURS 管理ビューおよび PD_GET_LOG_MSGS 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明
PROBE	INTEGER	DB2 お客様サポートや開発部門がメッセージの生成された DB2 ソース・コード内のポイントを見つけられるようにするための、固有の内部 ID。
MSGNUM	INTEGER	エラーまたはイベントの数値のメッセージ番号。
MSGTYPE	CHAR(3)	メッセージ・タイプを示します。ADM (管理通知ログに書き込まれたメッセージの場合)、または NULL (メッセージ・タイプが判別できない場合)。
MSGSEVERITY	CHAR(1)	メッセージ重大度: C (重大)、E (エラー)、W (警告)、I (通知)、または NULL (メッセージ重大度が判別できない場合)。
MSG	CLOB(16 KB)	通知ログ・メッセージ・テキスト。

REORGCHK_IX_STATS プロシージャー - 再編成の評価用の索引統計の検索

REORGCHK_IX_STATS プロシージャーは、再編成が必要かどうかを示す索引統計を含む、結果セットを戻します。

構文



スキーマは SYSPROC です。

プロシージャー・パラメーター

scope
評価する表の有効範囲を指定するタイプ CHAR(1) の入力引数であり、以下のいずれかの値を使用します。

'T' 表

'S' スキーマ

criteria
タイプ VARCHAR(259) の入力引数。 *scope* の値が 'T' である場合、完全修飾

表名を指定するか、 ALL、USER、または SYSTEM のいずれかの値を受け入れます。 *scope* の値が 'S' である場合、スキーマ名を指定します。

許可

- カタログ表に対する SELECT 特権。
- REORGCHK_IX_STATS プロシージャーに対する EXECUTE 特権。

例

```
CALL SYSPROC.REORGCHK_IX_STATS('T','JESCOTT.EMPLOYEE')
```

使用上の注意

プロシージャーは、SYSTOOLSTMPSPACE 表スペースを使用します。
SYSTOOLSTMPSPACE がまだ存在しない場合、このプロシージャーは、この表スペースを作成します。

戻される情報

表 264. REORGCHK_IX_STATS プロシージャーによって戻される情報

列名	データ・タイプ	説明
TABLE_SCHEMA	VARCHAR(128)	スキーマ名。
TABLE_NAME	VARCHAR(128)	表名。
DATAPARTITIONNAME	VARCHAR(128)	データ・パーティションの名前。非パーティション表の場合は NULL。
INDEX_SCHEMA	VARCHAR(128)	索引スキーマ名。
INDEX_NAME	VARCHAR(128)	索引名。
INDCARD	BIGINT	索引内の索引項目の数。一部の索引の場合、これは、表のカーディナリティーとは異なることがあります。たとえば、XML 列に対する索引カーディナリティーは、表のカーディナリティーより大きいことがあります。
NLEAF	BIGINT	索引リーフ・ページの合計数。
NUM_EMPTY_LEAFS	BIGINT	疑似的な空の索引リーフ・ページの数。
NLEVELS	INTEGER	索引レベルの数。
NUMRIDS_DELETED	BIGINT	疑似的な削除済み RID の数。
FULLKEYCARD	BIGINT	削除マークの付いていないユニーク索引項目の数。
LEAF_RECSIZE	BIGINT	リーフ・ページ上の索引項目のレコード・サイズ。これは、すべてのオーバーヘッドを除外した索引項目の平均サイズであり、索引に関与するすべての列から得た列の平均の長さから計算されます。

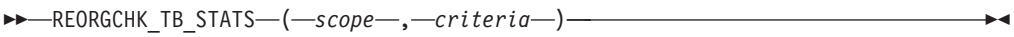
表 264. *REORGCHK_IX_STATS* プロシージャーによって戻される情報 (続き)

列名	データ・タイプ	説明
NONLEAF_RECSIZE	BIGINT	非リーフ・ページ上の索引項目のレコード・サイズ。これは、すべてのオーバーヘッドを除外した索引項目の平均サイズであり、索引に関与するすべての列 (INCLUDE 列を除く) から得た列の平均の長さから計算されます。
LEAF_PAGE_OVERHEAD	BIGINT	内部使用に備えて予約されている索引リーフ・ページ上のスペース。
NONLEAF_PAGE_OVERHEAD	BIGINT	内部使用に備えて索引の非リーフ・ページ上で予約されているスペース。
PCT_PAGES_SAVED	SMALLINT	索引圧縮を使用して節約されるページのパーセント。ゼロ以外の数値は、索引が圧縮されることを意味します。
F4	INTEGER	F4 公式値。
F5	INTEGER	F5 公式値。
F6	INTEGER	F6 公式値。
F7	INTEGER	F7 公式値。
F8	INTEGER	F8 公式値。
REORG	CHAR(5)	5 文字のフィールド。各文字は、F4、F5、F6、F7、および F8 の 5 つの公式のいずれかにマッピングされます。ダッシュが示される場合、公式値が推奨範囲に収まっていることを示します。アスタリスクが示される場合、公式値が推奨範囲から出ているので、再編成が必要であることを示します。

REORGCHK_TB_STATS プロシージャー - 再編成の評価用の表統計の検索

REORGCHK_TB_STATS プロシージャーは、再編成が必要かどうかを示す表統計を含む、結果セットを戻します。

構文



スキーマは SYSPROC です。

プロシージャー・パラメーター

scope
評価する表の有効範囲を指定するタイプ CHAR(1) の入力引数であり、以下のいずれかの値を使用します。

'T' 表

'S' スキーマ

criteria
タイプ VARCHAR(259) の入力引数。 *scope* の値が 'T' である場合、完全修飾表名を指定するか、 ALL、USER、または SYSTEM のいずれかの値を受け入れます。 *scope* の値が 'S' である場合、スキーマ名を指定します。

許可

- カタログ表に対する SELECT 特権。
- REORGCHK_TB_STATS プロシージャーに対する EXECUTE 特権。

例

```
CALL SYSPROC.REORGCHK_TB_STATS('T','JESCOTT.EMPLOYEE')
```

使用上の注意

プロシージャーは、SYSTOOLSTMPSPACE 表スペースを使用します。SYSTOOLSTMPSPACE がまだ存在しない場合、このプロシージャーは、この表スペースを作成します。

戻される情報

表 265. REORGCHK_TB_STATS プロシージャーによって戻される情報

列名	データ・タイプ	説明
TABLE_SCHEMA	VARCHAR(128)	スキーマ名。
TABLE_NAME	VARCHAR(128)	表名。
DATAPARTITIONNAME	VARCHAR(128)	データ・パーティションの名前。非パーティション表の場合は NULL。
CARD	BIGINT	カーディナリティー (表の中の行の数)。
OVERFLOW	BIGINT	オーバーフローした行数。
NPAGES	BIGINT	表の行が存在しているページの総数。ビューまたは別名の場合、または統計が収集されていない場合は -1。副表および階層表の場合は -2。
FPAGES	BIGINT	ページの総数。ビューまたは別名の場合、または統計が収集されていない場合は -1。副表および階層表の場合は -2。

表 265. REORGCHK_TB_STATS プロシージャーによって戻される情報 (続き)

列名	データ・タイプ	説明
ACTIVE_BLOCKS	BIGINT	マルチディメンション・クラスタリング (MDC) 表のアクティブ・ブロックの合計数。このフィールドは、ORGANIZE BY 節を使用して定義された表に対してのみ適用できます。これは、データを収めた表のブロック数を示します。
TSIZE	BIGINT	表のサイズ。
F1	INTEGER	F1 公式値。
F2	INTEGER	F2 公式値。
F3	INTEGER	F3 公式値。
REORG	CHAR(3)	3 文字のフィールド。各文字は、F1、F2、および F3 の 3 つの公式のいずれかにマッピングされます。ダッシュが示される場合、公式値が推奨範囲に収まっていることを示し、アスタリスクが示される場合、公式値が推奨範囲から出ているので、再編成が必要なことを示します。

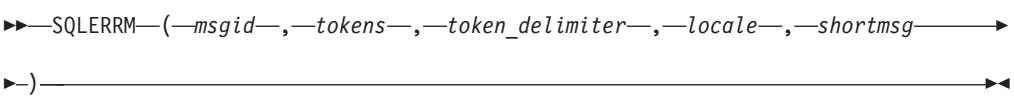
SQLERRM スカラー関数 - エラー・メッセージ情報の検索

SQLERRM スカラー関数には 2 つのバージョンがあります。1 つは、メッセージ・トークンの使用や言語選択などを含む、十分に柔軟性をもたせたメッセージ検索を提供します。もう 1 つは、SQLCODE のみを入力パラメーターとし、簡略メッセージを英語で戻します。

SQLERRM スカラー関数

この SQLERRM スカラー関数は、メッセージ ID、ロケール、およびトークンの入力を取り、タイプ VARCHAR(32672) の簡略メッセージまたは詳細メッセージを、指定のロケールで戻します。入力ロケールをサーバーがサポートしていない場合、メッセージは英語で戻されます。

構文



スキーマは SYSPROC です。

スカラー関数パラメーター

msgid
情報を検索するメッセージ番号を表す、タイプ VARCHAR(9) の入力引数。メ

ッセージ番号は、アプリケーション戻りコードに、接頭部 'SQL'、'DBA'、または 'CLI' を付けたものです。例えば、'SQL551'、'CLI0001' などです。メッセージ番号は、SQLSTATE である場合もあります。例えば、'42829' などです。

tokens
エラー・メッセージのトークン・リストを表す、タイプ VARCHAR(70) の入力引数。トークンのないメッセージもあります。このパラメーターが NULL の場合は、戻されるメッセージでトークンの置き換えはなされません。トークンの置き換えは、デフォルトの簡略メッセージを戻す場合にのみ生じます。詳細メッセージのオプションが選択されている場合、トークンの置き換えは生じません。

token_delimiter
トークンの区切り文字を表す、タイプ VARCHAR(1) の入力引数。この区切り文字は固有でなければならず、スカラー関数に渡されるトークンに含まれていてはなりません。区切り文字が指定されない場合、使用されるデフォルトの区切り文字はセミコロンです。

locale
ロケールを表すタイプ VARCHAR(33) の入力引数。その言語のエラー・メッセージを検索するためにサーバーに渡します。ロケールが指定されていない場合、またはサーバーがそのロケールをサポートしない場合、メッセージは英語で戻され、警告が戻されます。

shortmsg
デフォルトの簡略メッセージの代わりに詳細メッセージを戻すかどうかを示すために使用される、タイプ INTEGER の入力引数。詳細メッセージを戻すには、この値は 0 または CAST(NULL as INTEGER) に設定しなければなりません。

許可

SQLERRM スカラー関数に対する EXECUTE 特権

例

例 1: SQL0551N の英語の簡略メッセージを、トークン "AYYANG"、"UPDATE"、および "SYSCAT.TABLES" の含まれた状態で検索します。

```
VALUES (SYSPROC.SQLERRM
   ('SQL551', 'AYYANG;UPDATE;SYSCAT.TABLES', ';', 'en_US', 1))
```

以下は戻される出力の例です。

```
1
-------------------------------------------------------------...--
SQL0551N "AYYANG" does not have the privilege to perform operation
   "UPDATE" on object "SYSCAT.TABLES"
```

例 2: SQLSTATE 42501 に関連した英語のエラー・メッセージを検索します。

```
VALUES (SYSPROC.SQLERRM ('42501', '', '', 'en_US', 1))
```

以下は戻される出力の例です。

```
1
-------------------------------------------------------------...--
SQLSTATE 42501: The authorization ID does not have the privilege to
   perform the specified operation on the identified object.
```

例 3: SQL1001N の英語の長いエラー・メッセージを検索します。

```
VALUES (SYSPROC.SQLERRM ('SQL1001', '', '', 'en_US', 0))
```

以下は戻される出力の例です。

```
1
-------------------------------------------------------------...--
SQL1001N "<name>" is not a valid database name.

Explanation:

The syntax of the database name specified in the command is not
valid.  The database name must contain 1 to 8 characters and all
the characters must be from the database manager base character
set.

 The command cannot be processed.

User Response:

Resubmit the command with the correct database name.

 sqlcode :  -1001

 sqlstate :  2E000
```

SQLERRM スカラー関数

この SQLERRM スカラー関数は SQLCODE を唯一の入力データとして取り、指定の SQLCODE に対するタイプ VARCHAR(32672) の簡略メッセージを英語で戻します。

構文

スキーマは SYSPROC です。

スカラー関数パラメーター

sqlcode
SQLCODE を表す、タイプ INTEGER の入力引数。

許可

SQLERRM スカラー関数に対する EXECUTE 特権

例

SQLCODE SQL0551N の簡略メッセージを検索します。

```
VALUES (SYSPROC.SQLERRM (551))
```

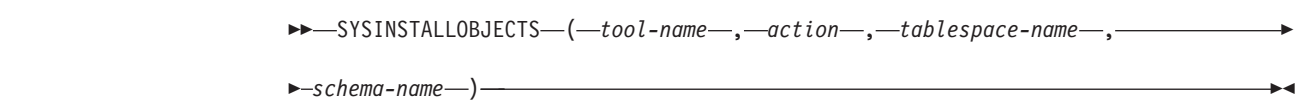
以下は戻される出力の例です。

```
1
-------------------------------------------------------------...--
SQL0551N  "" does not have the privilege to perform operation
   "" on object "".
```

SYSINSTALLOBJECTS

SYSINSTALLOBJECTS プロシージャーは、特定のツールで必要なデータベース・オブジェクトを作成またはドロップします。

構文



スキーマは SYSPROC です。

プロシージャー・パラメーター

tool-name
以下のいずれかの値を使用して、ロードされるツールの名前を指定する、タイプ VARCHAR(128) の入力引数。

- 'AM' (アクティビティー・モニター・オブジェクトを作成する場合)
- 'DB2AC' (自立走行式コンピューティングの場合 (ヘルス・モニター))
- 'STMG_DBSIZE_INFO' (ストレージ管理の場合)
- 'OPT_PROFILES' (最適化プロファイル表を作成する場合)
- 'POLICY' (ポリシーの場合 (表およびトリガー))
- 'EXPLAIN' (Explain 表を作成またはマイグレーションする場合)

action
実行予定のアクションを指定する、タイプ CHAR(1) 入力引数。有効な値は以下のとおりです。

C
オブジェクトを作成します。

D
オブジェクトをドロップします。

V
オブジェクトを検証します。

M
オブジェクトをマイグレーションします。 M オプションは、ツール名 EXPLAIN と共に使用する場合にのみ有効です。このオプションは、バージョン 9.5 からバージョン 9.7 までの間で作成された Explain 表をバージョン 9.7 フィックスパック 1 互換にマイグレーションします。バージョン 9.7 フィックスパック 1 以降で作成された Explain 表は変更されません。

tablespace-name
オブジェクトを作成するときの表スペースの名前を指定する、タイプ VARCHAR(128) の入力引数。値を指定しないか、値が空またはブランク・ストリングである場合には、ツール名が AM であればデフォルトのユーザー・スペースが使用されます。ツール名が EXPLAIN でアクションが M である場合には、入力表スペース名は無視されて、マイグレーション対象の Explain 表が作成された表スペースが使用されます。それ以外の場合は、SYSTOOLSPACE 表スペースが使用されます。 SYSTOOLSPACE がまだ存在しない場合は、作成されます。

schema-name

'EXPLAIN' tool-name オプションを除く、入力パラメーターとして渡される schema-name に関係なく、SYSTOOLS が必ずスキーマとして使用されます。

'EXPLAIN' tool-name オプションに対しては、入力の schema-name を渡すことができ、指定されたその schema-name で表が作成されます。 schema-name が入力パラメーターとして渡されない場合は、SYSTOOLS スキーマが使用されます。

例

```
CALL SYSPROC.SYSINSTALLOBJECTS('AM', 'C', CAST (NULL AS VARCHAR(128)),
   CAST (NULL AS VARCHAR(128)))
```

第 22 章 使用すべきでない SQL 管理ルーチンおよびその置換ルーチンまたはビュー

DB2 for Linux, UNIX, and Windows バージョン 9.7 の拡張サポートを既存の管理ルーチンに提供するために、一部の DB2 バージョン 9.5 ルーチンは、新しく、より包括的なルーチンまたはビューに置き換えられました。

DB2 for Linux, UNIX, and Windows バージョン 9.7 の表関数を使用するアプリケーションは、新しい関数または管理ビューを使用するために変更する必要があります。新しい表関数は、元の関数と同じベース名を持ち、それが追加された製品のバージョンの '_Vxx' (_V97 など) という接尾部が付けられます。たいていの場合、新しい表関数および管理ビューは追加の情報を戻します。管理ビューは必ず最新のバージョンの表関数を基にするので、アプリケーションの移植性は向上します。列はリリースごとに変わる可能性があるので (つまり追加されたり削除されたりするものがあるので)、管理ビューから特定の列を選択するか、または SELECT * ステートメントがアプリケーションにより使用される場合は結果セットを記述することをお勧めします。

© Copyright IBM Corp. 2006, 2010

表 266. DB2 バージョン 9.7 FP1 で推奨されない SQL 管理ルーチンまたはビュー、およびそれらに代わるルーチンまたはビュー

DB2 バージョン 9.7 で推奨されない関数	**新しい DB2 for Linux, UNIX, and Windows バージョン 9.7 FP1 関数またはビュー**
624 ページの『LOCKS_HELD 管理ビュー - 保持されているロックに関する情報の検索』	• **460 ページの『MON_GET_APPL_LOCKWAIT - アプリケーションが待機しているロックについての情報の収集』** • **490 ページの『MON_GET_LOCKS - 現在接続されているデータベース内のすべてのロックのリスト』** • **425 ページの『MON_FORMAT_LOCK_NAME - 内部ロック名のフォーマット設定と詳細の出力』** • **556 ページの『MON_LOCKWAITS 管理ビュー - ロックの取得を待機しているアプリケーションに関するメトリックの取得』**
627 ページの『LOCKWAITS 管理ビュー - 現行のロック待機情報の検索』	• **460 ページの『MON_GET_APPL_LOCKWAIT - アプリケーションが待機しているロックについての情報の収集』** • **490 ページの『MON_GET_LOCKS - 現在接続されているデータベース内のすべてのロックのリスト』** • **425 ページの『MON_FORMAT_LOCK_NAME - 内部ロック名のフォーマット設定と詳細の出力』** • **556 ページの『MON_LOCKWAITS 管理ビュー - ロックの取得を待機しているアプリケーションに関するメトリックの取得』**
720 ページの『SNAPLOCK 管理ビューおよび SNAP_GET_LOCK 表関数 - lock 論理データ・グループのスナップショット情報の検索』	• 460 ページの『MON_GET_APPL_LOCKWAIT - アプリケーションが待機しているロックについての情報の収集』 • 490 ページの『MON_GET_LOCKS - 現在接続されているデータベース内のすべてのロックのリスト』 • 425 ページの『MON_FORMAT_LOCK_NAME - 内部ロック名のフォーマット設定と詳細の出力』 • 556 ページの『MON_LOCKWAITS 管理ビュー - ロックの取得を待機しているアプリケーションに関するメトリックの取得』
726 ページの『SNAPLOCKWAIT 管理ビューおよび SNAP_GET_LOCKWAIT 表関数 - lockwait 論理データ・グループのスナップショット情報の検索』	• 460 ページの『MON_GET_APPL_LOCKWAIT - アプリケーションが待機しているロックについての情報の収集』 • 490 ページの『MON_GET_LOCKS - 現在接続されているデータベース内のすべてのロックのリスト』 • 425 ページの『MON_FORMAT_LOCK_NAME - 内部ロック名のフォーマット設定と詳細の出力』 • 556 ページの『MON_LOCKWAITS 管理ビュー - ロックの取得を待機しているアプリケーションに関するメトリックの取得』
828 ページの『SNAPDB 管理ビューおよび SNAP_GET_DB_V95 表関数 - dbase 論理グループからのスナップショット情報の検索』	675 ページの『SNAPDB 管理ビューおよび SNAP_GET_DB_V97 表関数 - dbase 論理グループからのスナップショット情報の検索』

表 267. DB2 バージョン 9.7 の使用すべきでない SQL 管理ルーチンおよびその置換ルーチンまたはビュー

DB2 バージョンの 9.5 使用すべきでない関数	新しい DB2 for Linux, UNIX, and Windows バージョン 9.7 関数またはビュー
1134 ページの『ADMINTABCOMPRESSINFO ビューおよび ADMIN_GET_TAB_COMPRESS_INFO』	259 ページの『ADMINTABCOMPRESSINFO 管理ビューおよび ADMIN_GET_TAB_COMPRESS_INFO_V97 表関数 - 圧縮された情報を返す』
1127 ページの『ADMIN_GET_TAB_INFO 表関数 - 表のサイズおよび状態に関する情報の検索』	268 ページの『ADMINTABINFO 管理ビューおよび ADMIN_GET_TAB_INFO_V97 表関数 - 表のサイズおよび状態に関する情報の検索』
892 ページの『SNAPSTORAGE_PATHS 管理ビューおよび SNAP_GET_STORAGE_PATHS 表関数 - 自動ストレージ・パスの情報の検索』	740 ページの『SNAPSTORAGE_PATHS 管理ビューおよび SNAP_GET_STORAGE_PATHS_V97 表関数 - 自動ストレージ・パスの情報の検索』
919 ページの『SNAPTBSP_PART 管理ビューおよび SNAP_GET_TBSP_PART_V91 表関数 - tablespace_nodeinfo 論理データ・グループのスナップショット情報の検索』	767 ページの『SNAPTBSP_PART 管理ビューおよび SNAP_GET_TBSP_PART_V97 表関数 - tablespace_nodeinfo 論理データ・グループのスナップショット情報の検索』
1316 ページの『WLM_GET_ACTIVITY_DETAILS - 特定のアクティビティーに関する詳細情報を戻す』	449 ページの『MON_GET_ACTIVITY_DETAILS 表関数 - 完全なアクティビティー詳細の取得』
1334 ページの『WLM_GET_SERVICE_SUBCLASS_STATS - サービス・サブクラスの統計を戻す』	1022 ページの『WLM_GET_SERVICE_SUBCLASS_STATS_V97 表関数 - サービス・サブクラスの統計を戻す』
1347 ページの『WLM_GET_WORKLOAD_STATS - ワークロード統計を戻す』	1040 ページの『WLM_GET_WORKLOAD_STATS_V97 表関数 - ワークロード統計を戻す』
1342 ページの『WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES - アクティビティーのリストを戻す』	1034 ページの『WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES_V97 - アクティビティーのリストを戻す』
1330 ページの『WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES - ワークロード・オカレンスのリスト』	1017 ページの『WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES_V97 - ワークロード・オカレンスのリスト』
1324 ページの『WLM_GET_SERVICE_CLASS_AGENTS - サービス・クラスで実行中のエージェントのリスト』	1009 ページの『WLM_GET_SERVICE_CLASS_AGENTS_V97 表関数 - サービス・クラスで実行中のエージェントのリスト』

DB2 for Linux, UNIX, and Windows バージョン 9.7 では、ヘルス・モニターは使用すべきでなくなりました。バージョン 9.7 では、使用すべきでないヘルス・モニター・インターフェースは引き続きサポートされています。DB2 for Linux, UNIX, and Windows のデータおよびデータ中心アプリケーションの管理用の新しい一揃いの GUI ツールが使用可能で、コントロール・センター・ツールの代わりに使用できます。詳しくは、データベース管理およびアプリケーション開発ツールを参照してください。

表 268. 使用すべきでないヘルス・モニター・ルーチン

1142 ページの『HEALTH_CONT_HI』
1143 ページの『HEALTH_CONT_HI_HIS』
1145 ページの『HEALTH_CONT_INFO』
1147 ページの『HEALTH_DB_HI』
1151 ページの『HEALTH_DB_HI_HIS』
1155 ページの『HEALTH_DB_HIC』
1157 ページの『HEALTH_DB_HIC_HIS』
1160 ページの『HEALTH_DB_INFO』
1162 ページの『HEALTH_DBM_HI』
1163 ページの『HEALTH_DBM_HI_HIS』
1166 ページの『HEALTH_DBM_INFO』
1167 ページの『HEALTH_GET_ALERT_ACTION_CFG』
1171 ページの『HEALTH_GET_ALERT_CFG』
1174 ページの『HEALTH_GET_IND_DEFINITION』
1176 ページの『HEALTH_HI_REC』
1178 ページの『HEALTH_TBS_HI』
1181 ページの『HEALTH_TBS_HI_HIS』
1185 ページの『HEALTH_TBS_INFO』

前のリリース、DB2 バージョン 9.5 でも、DB2 バージョン 9.1 関数と取り替えられた新規関数がありました。

表 269. DB2 バージョン 9.5 の使用すべきでない SQL 管理ルーチンおよびその置換ルーチンまたはビュー

DB2 バージョンの 9.1 使用すべきでない関数	**DB2 バージョン 9.5 の新規の関数またはビュー**
1187 ページの『SNAP_GET_APPL 表関数 - appl 論理データ・グループのスナップショット情報の検索』	652 ページの『SNAPAPPL 管理ビューおよび SNAP_GET_APPL_V95 表関数 - appl 論理データ・グループのスナップショット情報の検索』
1195 ページの『SNAP_GET_APPL_INFO 表関数 - appl_info 論理データ・グループのスナップショット情報の検索』	644 ページの『SNAPAPPL_INFO 管理ビューおよび SNAP_GET_APPL_INFO_V95 表関数 - appl_info 論理データ・グループのスナップショット情報の検索』
1203 ページの『SNAP_GET_BP 表関数 - bufferpool 論理グループのスナップショット情報の検索』	661 ページの『SNAPBP 管理ビューおよび SNAP_GET_BP_V95 表関数 - bufferpool 論理グループのスナップショット情報の検索』
1219 ページの『SNAP_GET_DB_V91 table function - dbase 論理グループからのスナップショット情報の検索』	828 ページの『SNAPDB 管理ビューおよび SNAP_GET_DB_V95 表関数 - dbase 論理グループからのスナップショット情報の検索』
1216 ページの『SNAP_GET_DBM 表関数 - dbm 論理グループ・スナップショット情報の検索』	693 ページの『SNAPDBM 管理ビューおよび SNAP_GET_DBM_V95 表関数 - dbm 論理グループ・スナップショット情報の検索』
1243 ページの『SNAP_GET_DYN_SQL_V91 表関数 - dynsql 論理グループのスナップショット情報の検索』	704 ページの『SNAPDYN_SQL 管理ビューおよび SNAP_GET_DYN_SQL_V95 表関数 - dynsql 論理グループのスナップショット情報の検索』

ADMIN_GET_TAB_INFO 表関数 - 表のサイズおよび状態に関する情報の検索

注: この表関数は使用すべきではなく、 268 ページの『ADMINTABINFO 管理ビューおよび ADMIN_GET_TAB_INFO_V97 表関数 - 表のサイズおよび状態に関する情報の検索』 に置き換えられました。

ADMIN_GET_TAB_INFO 表関数は、現在カタログ・ビューで使用できない表のサイズおよび状態に関する情報を検索するメソッドを提供します。

戻される可能性のある情報の完全なリストは、ADMIN_GET_TAB_INFO 表関数のメタデータ表を参照してください。

構文

スキーマは SYSPROC です。

表関数パラメーター

tabschema
スキーマ名を指定する、タイプ VARCHAR(128) の入力引数。

tabname
表、マテリアライズ照会表、または階層表それぞれの名前を指定する、タイプ VARCHAR(128) の入力引数。

許可

ADMIN_GET_TAB_INFO 表関数に対する EXECUTE 特権。

例

例 1: 表 DBUSER1.EMPLOYEE のサイズおよび状態に関する情報を検索します。

```
SELECT * FROM TABLE (SYSPROC.ADMIN_GET_TAB_INFO('DBUSER1', 'EMPLOYEE'))
   AS T
```

例 2: 非パーティション表 (DBUSER1.EMPLOYEE) が存在し、関連オブジェクト (たとえば索引や LOB など) がすべて 1 つの表スペースに保管されていると仮定します。 表が表スペース内のどのくらいの物理スペースを使用しているかを計算します。

```
SELECT (data_object_p_size + index_object_p_size + long_object_p_size +
   lob_object_p_size + xml_object_p_size) as total_p_size
   FROM TABLE( SYSPROC.ADMIN_GET_TAB_INFO( 'DBUSER1', 'EMPLOYEE' )) AS T
```

表が別の表スペースに移動されるときにどのくらいのスペースが必要になるかを計算します。ここでは、新規の表スペースには元の表スペースと同じページ・サイズおよびエクステント・サイズがあるとします。

```
SELECT (data_object_l_size + index_object_l_size + long_object_l_size +
   lob_object_l_size + xml_object_l_size) as total_l_size
   FROM TABLE( SYSPROC.ADMIN_GET_TAB_INFO( 'DBUSER1', 'EMPLOYEE' )) AS T
```

使用上の注意

- *tabschema* と *tabname* の両方が指定される場合、その特定の表の情報のみが戻されます。
- *tabschema* が指定され、*tabname* が空 (") または NULL の場合、指定したスキーマのすべての表に関する情報が戻されます。
- *tabschema* が空 (") または NULL で、*tabname* が指定される場合、エラーは戻されます。 特定の表の情報を取り出すには、その表がスキーマと表名の両方によって識別されることが必要です。
- *tabschema* と *tabname* の両方が空 (") または NULL の場合、すべての表の情報が戻されます。
- *tabschema* または *tabname* が存在しないか、あるいは *tabname* が表名 (タイプ T)、マテリアライズ照会表名 (タイプ S)、または階層表名 (タイプ H) と一致しない場合、空の結果セットが戻されます。
- ADMIN_GET_TAB_INFO 表関数が指定の表のデータを検索するとき、この表関数は SYSTABLES の対応する行に対する共用ロックを獲得します。これは、戻されるデータの整合性を確保するための動作です (たとえば、情報の検索中に、検索されている表がドロップされないようにするなど)。ロックが保持されるのは、表関数の呼び出し期間中ではなく、表のサイズおよび状態に関する情報を検索する間だけです。
- SMS 表スペースの表の物理サイズが報告されますが、このサイズは論理サイズと同じです。
- 表で INPLACE の REORG がアクティブになっていると、データ・オブジェクトの物理サイズ (DATA_OBJECT_P_SIZE) は計算されません。論理サイズだけが戻されます。 INPLACE の REORG が表でアクティブになっているかどうかは、INPLACE_REORG_STATUS 出力の列を見ると分かります。
- DB2 UDB バージョン 8 より前に作成された LOB オブジェクトの論理サイズが報告されますが、オブジェクトの再編成が行われていないと、この論理サイズは物理サイズよりも大きくなっている場合があります。

ADMIN_GET_TAB_INFO 表関数のメタデータ

表 270. ADMIN_GET_TAB_INFO 表関数のメタデータ

列名	データ・タイプ	説明
TABSCHEMA	VARCHAR(128)	スキーマ名。
TABNAME	VARCHAR(128)	表名。
TABTYPE	CHAR(1)	表タイプ: • 「H」= 階層表 • 「S」= マテリアライズ照会表 • 「T」= 表
DBPARTITIONNUM	SMALLINT	データベース・パーティション番号。
DATA_PARTITION_ID	INTEGER	データ・パーティション番号。

表 270. ADMIN_GET_TAB_INFO 表関数のメタデータ (続き)

列名	データ・タイプ	説明
AVAILABLE	CHAR(1)	表の状態: • 「N」= 表は使用不可。表を使用できない場合、サイズおよび状態に関係する他の出力列はすべて NULL になります。 • 「Y」= 表は使用可能。 **注:** リカバリー不能ロードでのロールフォワードを行うと、表の状態が使用不可になります。
DATA_OBJECT_L_SIZE	BIGINT	データ・オブジェクトの論理サイズ。表に対して論理的に割り振られるディスク・スペースの量 (KB 単位で報告)。論理サイズとは、表が認識するスペースの量のことです。このサイズは表に対して物理的に割り振られるスペースの量より小さくなることもあります (たとえば論理表の切り捨ての場合)。多次元クラスタリング (MDC) 表の場合、このサイズにはブロック・マップ・オブジェクトの論理サイズが含まれます。戻されるサイズは、表に対して論理的に割り振られるすべてのエクステントを考慮に入れ、DMS 表スペースで作成されるオブジェクトの場合は Extent Map Page (EMP) エクステントの見積もりを考慮に入れます。このサイズは、基本表のみの論理サイズを表します。 LOB データ、長形式データ、索引、および XML オブジェクトが消費するスペースは別の列で報告されます。
DATA_OBJECT_P_SIZE	BIGINT	データ・オブジェクトの物理サイズ。表に対して物理的に割り振られるディスク・スペースの量 (KB 単位で報告)。 MDC 表の場合、このサイズにはブロック・マップ・オブジェクトのサイズが含まれます。戻されるサイズは、表に割り振られるすべてのエクステントを考慮に入れ、DMS 表スペースで作成されるオブジェクトの EMP エクステントが含まれます。このサイズは、基本表のみの物理サイズを表します。 LOB データ、長形式データ、索引、および XML オブジェクトが消費するスペースは別の列で報告されます。

表 270. ADMIN_GET_TAB_INFO 表関数のメタデータ (続き)

列名	データ・タイプ	説明
INDEX_OBJECT_L_SIZE	BIGINT	索引オブジェクトの論理サイズ。表で定義される索引に対して論理的に割り振られるディスク・スペースの量 (KB 単位で報告)。論理サイズとは、表が認識するスペースの量のことです。このサイズは表の索引データを保持するために物理的に割り振られるスペースの量より小さくなることもあります (たとえば論理表の切り捨ての場合)。戻されるサイズは、索引に対して論理的に割り振られるすべてのエクステントを考慮に入れ、DMS 表スペースで作成される索引の場合は EMP エクステントの見積もりを考慮に入れます。この値は非パーティション表の場合にのみ報告されます。パーティション表の場合、この値は 0 になります。
INDEX_OBJECT_P_SIZE	BIGINT	索引オブジェクトの物理サイズ。表で定義される索引に対して物理的に割り振られるディスク・スペースの量 (KB 単位で報告)。戻されるサイズは、索引に割り振られるすべてのエクステントを考慮に入れ、DMS 表スペースで作成される索引の EMP エクステントが含まれます。この値は非パーティション表の場合にのみ報告されます。パーティション表の場合、この値は 0 になります。
LONG_OBJECT_L_SIZE	BIGINT	長形式オブジェクトの論理サイズ。表の長形式フィールド・データに対して論理的に割り振られるディスク・スペースの量 (KB 単位で報告)。論理サイズとは、表が認識するスペースの量のことです。このサイズは表の長形式フィールド・データを保持するために物理的に割り振られるスペースの量より小さくなることもあります (たとえば論理表の切り捨ての場合)。戻されるサイズは、長形式フィールド・データに対して論理的に割り振られるすべてのエクステントを考慮に入れ、DMS 表スペースで作成される長形式フィールド・データの場合は EMP エクステントの見積もりを考慮に入れます。

表 270. ADMIN_GET_TAB_INFO 表関数のメタデータ (続き)

列名	データ・タイプ	説明
LONG_OBJECT_P_SIZE	BIGINT	長形式オブジェクトの物理サイズ。表の長形式フィールド・データに対して物理的に割り振られるディスク・スペースの量 (KB 単位で報告)。戻されるサイズは、長形式フィールド・データに割り振られるすべてのエクステントを考慮に入れ、DMS 表スペースで作成される長形式フィールド・データの EMP エクステントが含まれます。
LOB_OBJECT_L_SIZE	BIGINT	LOB オブジェクトの論理サイズ。表の LOB データに対して論理的に割り振られるディスク・スペースの量 (KB 単位で報告)。論理サイズとは、表が認識するスペースの量のことです。このサイズは表の LOB データを保持するために物理的に割り振られるスペースの量より小さくなることもあります (たとえば論理表の切り捨ての場合)。サイズには LOB 割り振りオブジェクトに対して論理的に割り振られるスペースが含まれます。戻されるサイズは、LOB データに対して論理的に割り振られるすべてのエクステントを考慮に入れ、DMS 表スペースで作成される LOB データの場合は EMP エクステントの見積もりを考慮に入れます。
LOB_OBJECT_P_SIZE	BIGINT	LOB オブジェクトの物理サイズ。表の LOB データに対して物理的に割り振られるディスク・スペースの量 (KB 単位で報告)。サイズには LOB 割り振りオブジェクトに対して割り振られるスペースが含まれます。戻されるサイズは、LOB データに割り振られるすべてのエクステントを考慮に入れ、DMS 表スペースで作成される LOB データの EMP エクステントが含まれます。

表 270. ADMIN_GET_TAB_INFO 表関数のメタデータ (続き)

列名	データ・タイプ	説明
XML_OBJECT_L_SIZE	BIGINT	XML オブジェクトの論理サイズ。表の XML データに対して論理的に割り振られるディスク・スペースの量 (KB 単位で報告)。論理サイズとは、表が認識するスペースの量のことです。このサイズは表の XML データを保持するために物理的に割り振られるスペースの量より小さくなることもあります (たとえば論理表の切り捨ての場合)。戻されるサイズは、XML データに対して論理的に割り振られるすべてのエクステントを考慮に入れ、DMS 表スペースで作成される XML データの場合は EMP エクステントの見積もりを考慮に入れます。
XML_OBJECT_P_SIZE	BIGINT	XML オブジェクトの物理サイズ。表の XML データに対して物理的に割り振られるディスク・スペースの量 (KB 単位で報告)。戻されるサイズは、XML データに割り振られるすべてのエクステントを考慮に入れ、DMS 表スペースで作成される XML データの EMP エクステントが含まれます。
INDEX_TYPE	SMALLINT	現在表で使用されている索引のタイプを示します。これは、以下のものを戻します。 • タイプ 1 索引が使用されている場合は 1。 • タイプ 2 索引が使用されている場合は 2。
REORG_PENDING	CHAR(1)	「Y」は、REORG 推奨の変更がすでに表に適用されており、クラシック (オフライン) REORG が必要であることを示しています。それ以外の場合は「N」が戻されます。
INPLACE_REORG_STATUS	VARCHAR(10)	表に対する表のインプレース再編成の現在の状況。状況値は以下のいずれかになります。 • ABORTED (PAUSED 状態にあるが、RESUME は不可。STOP は必須) • EXECUTING • NULL (表に対して INPLACE の REORG が実行されていない場合) • PAUSED

表 270. ADMIN_GET_TAB_INFO 表関数のメタデータ (続き)

列名	データ・タイプ	説明
LOAD_STATUS	VARCHAR(12)	表に対するロード操作の現在の状況。状況値は以下のいずれかになります。 • IN_PROGRESS • NULL (表でロードが進行しておらず、表がロード・ペンディング状態になっていない場合) • PENDING
READ_ACCESS_ONLY	CHAR(1)	表が「読み取りアクセス専用」の状態になっていれば「Y」、そうでなければ「N」になります。「N」の値を、表が完全にアクセス可能であるという意味に解釈するべきではありません。ロードが進行中またはペンディング状態の場合、「Y」の値は表データが読み取りアクセス可能であることを意味し、「N」の値は表がアクセス不能であることを意味します。同様に、表の状況が SET INTEGRITY ペンディングである場合 (SYSCAT.TABLES STATUS 列を参照)、「N」の値は表がアクセス不能であることを意味します。
NO_LOAD_RESTART	CHAR(1)	「Y」の値は、表が部分的にロードされている状態になっていることを示します。この場合、ロードを再始動することができません。この状態になっていなければ「N」の値が戻されます。
NUM_REORG_REC_ALTERS	SMALLINT	最後に再編成が行われてからこの表に対して実行された REORG 推奨の変更操作 (たとえば直後に再編成を必要とする変更操作) の回数。
INDEXES_REQUIRE_ REBUILD	CHAR(1)	表で定義される索引のいずれかが再ビルドを必要とする場合は「Y」、必要としない場合は「N」。表で索引が定義されていない場合にも「N」が戻されます。再ビルドを必要とする索引が存在しないからです。

表 270. ADMIN_GET_TAB_INFO 表関数のメタデータ (続き)

列名	データ・タイプ	説明
LARGE_RIDS	CHAR(1)	表がラージ行 ID (RID) を使用しているかどうかを示します (4 バイトのページ番号と 2 バイトのスロット番号)。「Y」の値は表がラージ RID を使用していることを示し、「N」は使用していないことを示します。表がラージ RID をサポートしている (つまり表が LARGE 表スペースにある) ものの、少なくとも表の索引の 1 つがまだ再編成されていないかまたは再ビルドされていない場合、「P」(保留) の値が戻されます。これは表が 4 バイトの RID を使用しているためです (これは表または索引を変換するためのアクションを取る必要があることを意味します)。
LARGE_SLOTS	CHAR(1)	表がラージ・スロット (これは 1 ページにつき 255 を超える行が可能です) を使用しているかどうかを示します。「Y」の値は表がラージ・スロットを使用していることを示し、「N」は使用していないことを示します。表がラージ・スロットをサポートしている (つまり表が LARGE 表スペースにある) ものの、表に対してまだオフラインの表の再編成または表の切り捨て操作が実行されていない場合、「P」(保留) の値が戻されます。これは、表が 1 ページにつき最大 255 行のラージ・スロットを使用しているためです。
DICTIONARY_SIZE	BIGINT	ディクショナリーのサイズ (バイト)。表に行コンプレッション・ディクショナリーが存在する場合に行の圧縮で使用されます。

ADMINTABCOMPRESSINFO ビューおよび ADMIN_GET_TAB_COMPRESS_INFO

注: この表関数は使用すべきではなく、 259 ページの 『ADMINTABCOMPRESSINFO 管理ビューおよび ADMIN_GET_TAB_COMPRESS_INFO_V97 表関数 - 圧縮された情報を返す』 に置き換えられました。

ADMINTABCOMPRESSINFO 管理ビューおよび ADMIN_GET_TAB_COMPRESS_INFO 表関数は、表、マテリアライズ照会表 (MQT)、階層表の圧縮された情報を戻します。

ADMINTABCOMPRESSINFO 管理ビュー

ADMINTABCOMPRESSINFO 管理ビューは、表、マテリアライズ照会表 (MQT)、および階層表だけの圧縮された情報を戻します。これらの表タイプは SYSCAT.TABLES カタログ・ビューで、T (表)、S (マテリアライズ照会表)、および H (階層表) として報告されます。情報は表のデータ・パーティション・レベルとデータベース・パーティション・レベルの両方のものが戻されます。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、ADMINTABCOMPRESSINFO 管理ビューおよび ADMIN_GET_TAB_COMPRESS_INFO 表関数のメタデータ表を参照してください。

許可

以下のいずれかの権限が必要です。

- ADMINTABCOMPRESSINFO 管理ビューに対する SELECT 特権
- ADMINTABCOMPRESSINFO 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- ADMIN_GET_TAB_COMPRESS_INFO 表関数に対する EXECUTE 特権
- DATAACCESS 権限

例

例 1: すべての表に関するすべての圧縮された情報を検索します。

```
SELECT * FROM SYSIBMADM.ADMINTABCOMPRESSINFO
```

以下はこの照会の出力例です。

```
TABSCHEMA     TABNAME         DBPARTITIONNUM DATA_PARTITION_ID COMPRESS_ATTR DICT_BUILDER      DICT_BUILD_TIMESTAMP
---------...- -------------...- -------------- ----------------- ------------- -------------...- --------------------------
SYSIBM        SYSTABLES                       0                 0 N             NOT BUILT     -
SYSIBM        SYSCOLUMNS                      0                 0 N             NOT BUILT     -
...
SIMAP2        STAFF                           0                 0 Y             REORG         2006-08-27-19.07.36.000000
SIMAP2        PARTTAB                         0                 0 Y             REORG         2006-08-27-22.07.17.000000
...

  156 record(s) selected.
```

この照会からの出力 (続き):

```
COMPRESS_DICT_SIZE   EXPAND_DICT_SIZE      ROWS_SAMPLED PAGES_SAVED_PERCENT BYTES_SAVED_PERCENT AVG_COMPRESS_REC_LENGTH
-------------------- --------------------  ------------ -------------------- -------------------- ----------------------
                   0                    0            0                    0                    0                      0
                   0                    0            0                    0                    0                      0
...
               13312                 5312           35                   65                   84                    100
                5760                 4248           45                   76                   79                     98
...
```

例 2: すべての表に関するディクショナリー作成アクションおよびディクショナリー作成時間を決定します。

```
SELECT TABSCHEMA, TABNAME, DBPARTITIONNUM, DATA_PARTITION_ID, DICT_BUILDER, DICT_BUILD_TIMESTAMP
FROM SYSIBMADM.ADMINTABCOMPRESSINFO
```

以下はこの照会の出力例です。

```
TABSCHEMA       TABNAME         DBPARTITIONNUM DATA_PARTITION_ID DICT_BUILDER               DICT_BUILD_TIMESTAMP
--------------- --------------- -------------- ----------------- -------------------------- --------------------------
SYSIBM          SYSTABLES                   0                 0 NOT BUILT                  -
SYSIBM          SYSCOLUMNS                  0                 0 NOT BUILT                  -
...
SIMAP2          STAFF                       0                 0 REORG                      2006-08-27-19.07.36.000000
SIMAP2          SALES                       0                 0 NOT BUILT                  -
SIMAP2          CATALOG                     0                 0 NOT BUILT                  -
...

  156 record(s) selected.
```

ADMIN_GET_TAB_COMPRESS_INFO 表関数

ADMIN_GET_TAB_COMPRESS_INFO 表関数は ADMINTABCOMPRESSINFO 管理ビューと同じ情報を戻しますが、この関数ではスキーマ、表名、および実行モードを指定することが可能です。

戻される可能性のある情報の完全なリストは、ADMINTABCOMPRESSINFO 管理ビューおよび ADMIN_GET_TAB_COMPRESS_INFO 表関数のメタデータ表を参照してください。

注: この表関数は使用すべきではなく、 259 ページの『ADMINTABCOMPRESSINFO 管理ビューおよび ADMIN_GET_TAB_COMPRESS_INFO_V97 表関数 - 圧縮された情報を返す』 に置き換えられました。

構文

スキーマは SYSPROC です。

表関数パラメーター

tabschema
スキーマ名を指定する、タイプ VARCHAR(128) の入力引数。

tabname
表、マテリアライズ照会表、または階層表それぞれの名前を指定する、タイプ VARCHAR(128) の入力引数。

execmode
実行モードを指定する、タイプ VARCHAR(30) の入力引数。実行モードは以下のいずれかになります。

- 'REPORT' -- 最後の生成の時点での圧縮情報を報告します。これはデフォルト値です。
- 'ESTIMATE' -- 現在の表に基づいて新規の圧縮情報を生成します。

許可

ADMIN_GET_TAB_COMPRESS_INFO 関数に対する EXECUTE 特権。

例

例 1: 表 SIMAP2.STAFF に関する既存の圧縮情報を検索します。

```
SELECT * FROM TABLE (SYSPROC.ADMIN_GET_TAB_COMPRESS_INFO('SIMAP2', 'STAFF', 'REPORT'))
AS T
```

以下はこの照会の出力からの例です。

```
TABSCHEMA      TABNAME     DBPARTITIONNUM DATA_PARTITION_ID COMPRESS_ATTR DICT_BUILDER     DICT_BUILD_TIMESTAMP
---------...- -------...- -------------- ----------------- ------------- ------------...- --------------------------
SIMAP2         STAFF                   0                 0 Y             REORG           2006-08-27-19.07.36.000000

  1 record(s) selected.
```

この照会からの出力 (続き):

```
COMPRESS_DICT_SIZE   EXPAND_DICT_SIZE     ROWS_SAMPLED PAGES_SAVED_PERCENT BYTES_SAVED_PERCENT AVG_COMPRESS_REC_LENGTH
-------------------- -------------------- ------------ -------------------- -------------------- ----------------------
               13312                 5312           35                   65                   84                    100
```

例 2: 現時点での表 SIMAP2.STAFF に関する推定圧縮圧縮情報を検索します。

```
SELECT * FROM TABLE (SYSPROC.ADMIN_GET_TAB_COMPRESS_INFO('SIMAP2', 'STAFF', 'ESTIMATE'))
AS T
```

以下はこの照会の出力からの例です。

```
TABSCHEMA      TABNAME     DBPARTITIONNUM DATA_PARTITION_ID COMPRESS_ATTR DICT_BUILDER         DICT_BUILD_TIMESTAMP
---------...- -------...- -------------- ----------------- ------------- ----------------...- --------------------------
SIMAP2         STAFF                   0                 0 Y             TABLE FUNCTION      2006-08-28-19.18.13.000000

  1 record(s) selected.
```

この照会からの出力 (続き):

```
COMPRESS_DICT_SIZE   EXPAND_DICT_SIZE     ROWS_SAMPLED PAGES_SAVED_PERCENT BYTES_SAVED_PERCENT AVG_COMPRESS_REC_LENGTH
-------------------- -------------------- ------------ -------------------- -------------------- ----------------------
               13508                 6314           68                   72                   89                     98
```

例 3: スキーマ SIMAP2 でのすべての表に関する合計ディクショナリー・サイズを決定します。

```
SELECT TABSCHEMA, TABNAME, DICT_BUILDER,
 (COMPRESS_DICT_SIZE+EXPAND_DICT_SIZE) AS TOTAL_DICT_SIZE,
 DBPARTITIONNUM, DATA_PARTITION_ID
FROM TABLE (SYSPROC.ADMIN_GET_TAB_COMPRESS_INFO('SIMAP2', '', 'REPORT')) AS T
```

この照会からの出力:

```
TABSCHEMA       TABNAME         DICT_BUILDER                  TOTAL_DICT_SIZE      DBPARTITIONNUM DATA_PARTITION_ID
--------------- --------------- ----------------------------- -------------------- -------------- -----------------
SIMAP2          ACT             NOT BUILT                                        0              0                 0
SIMAP2          ADEFUSR         NOT BUILT                                        0              0                 0
...
SIMAP2          INVENTORY       NOT BUILT                                        0              0                 0
SIMAP2          ORG             NOT BUILT                                        0              0                 0
SIMAP2          PARTTAB         REORG                                        10008              0                 0
SIMAP2          PARTTAB         REORG                                         5464              0                 1
SIMAP2          PARTTAB         REORG                                         8456              0                 2
SIMAP2          PARTTAB         REORG                                         6960              0                 3
SIMAP2          PARTTAB         REORG                                         7136              0                 4
...
SIMAP2          STAFF           REORG                                        18624              0                 0
```

```
SIMAP2          SUPPLIERS       NOT BUILT                                       0             0               0
SIMAP2          TESTTABLE       NOT BUILT                                       0             0               0

  28 record(s) selected.
```

例 4: SIMAP2 スキーマの表のディクショナリー情報のレポートを表示します。

```
SELECT * FROM TABLE (SYSPROC.ADMIN_GET_TAB_COMPRESS_INFO('SIMAP2', '', 'REPORT'))
AS T
```

この照会からの出力:

```
TABSCHEMA      TABNAME     DBPARTITIONNUM DATA_PARTITION_ID COMPRESS_ATTR DICT_BUILDER         DICT_BUILD_TIMESTAMP
---------...- -------...- -------------- ----------------- ------------- -----------------...- --------------------------
SIMAP2         T1                      0                 0 Y             NOT BUILT           -
SIMAP2         T2                      0                 0 N             REORG               2007-02-03-17.35.28.000000
SIMAP2         T3                      0                 0 Y             INSPECT             2007-02-03-17.35.44.000000
SIMAP2         T4                      0                 0 N             NOT BUILT           -

  4 record(s) selected.
```

この照会からの出力 (続き):

```
COMPRESS_DICT_SIZE   EXPAND_DICT_SIZE     ROWS_SAMPLED PAGES_SAVED_PERCENT BYTES_SAVED_PERCENT AVG_COMPRESS_REC_LENGTH
-------------------- -------------------- ------------ -------------------- -------------------- ------------------------
                   0                    0            0                    0                    0                        0
                1280                 2562            -                    -                    -                        -
                1340                 2232            -                    -                    -                        -
                   0                    0            0                    0                    0                        0
```

使用上の注意

- *tabschema* と *tabname* の両方が指定される場合、その特定の表の情報のみが戻されます。
- *tabschema* が指定され、*tabname* が空 (") または NULL の場合、指定したスキーマのすべての表に関する情報が戻されます。
- *tabschema* が空 (") または NULL で、*tabname* が指定される場合、エラーは戻されます。 特定の表の情報を取り出すには、その表がスキーマと表名の両方によって識別されることが必要です。
- *tabschema* と *tabname* の両方が空 (") または NULL の場合、すべての表の情報が戻されます。
- *tabschema* または *tabname* が存在しないか、あるいは *tabname* が表名 (タイプ T)、マテリアライズ照会表名 (タイプ S)、または階層表名 (タイプ H) と一致しない場合、空の結果セットが戻されます。
- ADMIN_GET_TAB_COMPRESS_INFO 表関数が指定の表のデータを検索するとき、この表関数は SYSTABLES の対応する行に対する共用ロックを獲得します。これは、戻されるデータの整合性を確保するための動作です (たとえば、情報の検索中に、検索されている表が変更されないようにするなど)。ロックが保持されるのは、表関数の呼び出し期間中ではなく、表の圧縮情報を検索する間だけです。

ADMINTABCOMPRESSINFO 管理ビューおよび ADMIN_GET_TAB_COMPRESS_INFO 表関数のメタデータ

表 271. ADMINTABCOMPRESSINFO 管理ビューおよび ADMIN_GET_TAB_COMPRESS_INFO 表関数のメタデータ

列名	データ・タイプ	説明
TABSCHEMA	VARCHAR(128)	スキーマ名
TABNAME	VARCHAR(128)	表名

表 271. ADMINTABCOMPRESSINFO 管理ビューおよび ADMIN_GET_TAB_COMPRESS_INFO 表関数のメタデータ (続き)

列名	データ・タイプ	説明
DBPARTITIONNUM	SMALLINT	データベース・パーティション番号
DATA_PARTITION_ID	INTEGER	データ・パーティション番号
COMPRESS_ATTR	CHAR(1)	表の COMPRESS 属性の状態。以下のいずれかになります。 • 「Y」= 行圧縮は yes に設定されます。 • 「N」= 行圧縮は no に設定されます。
DICT_BUILDER	VARCHAR(30)	ディクショナリーを作成するために取られるコード・パス。以下のいずれかになります。 • 'INSPECT' = INSPECT ROWCOMPESTIMATE • 'LOAD' = LOAD INSERT/REPLACE • 'NOT BUILT' = 使用可能なディクショナリーがありません • 'REDISTRIBUTE' = REDISTRIBUTE • 'REORG' = REORG RESETDICTIONARY • 'TABLE GROWTH' = INSERT
DICT_BUILD_TIMESTAMP	TIMESTAMP	ディクショナリーが作成された時刻のタイム・スタンプ。タイム・スタンプの細分性は秒単位です。使用可能なディクショナリーがない場合、タイム・スタンプは NULL です。
COMPRESS_DICT_SIZE	BIGINT	バイト単位で測定されたコンプレッション・ディクショナリーのサイズ。
EXPAND_DICT_SIZE	BIGINT	バイト単位で測定されたエクスパンション・ディクショナリーのサイズ。履歴ディクショナリーが存在する場合、この値は現在のディクショナリー・サイズと履歴ディクショナリー・サイズの合計になります。
ROWS_SAMPLED	INTEGER	ディクショナリーの作成に役立つレコードの数。コンプレッション・ディクショナリーを含むマイグレーション済みの表はこの列で NULL を戻します。
PAGES_SAVED_PERCENT	SMALLINT	圧縮により削減されるページのパーセンテージこの情報は、サンプル・バッファー内のレコード・データのみに基づきます。コンプレッション・ディクショナリーを含むマイグレーション済みの表はこの列で NULL を戻します。
BYTES_SAVED_PERCENT	SMALLINT	圧縮により削減されるバイトのパーセンテージこの情報は、サンプル・バッファー内のレコード・データのみに基づきます。コンプレッション・ディクショナリーを含むマイグレーション済みの表はこの列で NULL を戻します。
AVG_COMPRESS_REC_LENGTH	SMALLINT	ディクショナリーの作成に役立っているレコードの平均圧縮レコード長。コンプレッション・ディクショナリーを含むマイグレーション済みの表はこの列で NULL を戻します。

GET_DB_CONFIG

注: このプロシージャーは使用すべきではなく、 356 ページの『DBCFG 管理ビュー - データベース構成パラメーター情報の検索』によって置き換えられました。

スキーマは SYSPROC です。

GET_DB_CONFIG プロシージャーは、データベース構成情報を戻します。このプロシージャーは引数を取りません。

このプロシージャーは、パラメーターごとに 1 つの列を備えた 2 つの行を示す単一の結果セットを戻します。最初の列は、以下に示すように DBCONFIG_TYPE という名前です。

表 272. GET_DB_CONFIG プロシージャーによって戻される情報

列名	データ・タイプ	説明
DBCONFIG_TYPE	INTEGER	この列の値が 0 の行には、ディスクに保管されたデータベース構成パラメーターの値が入ります。この列の値が 1 の行には、メモリーに保管されたデータベース構成パラメーターの現行値が入ります。

このプロシージャーは、結果セットを保管する DB_CONFIG という名前のグローバル一時表を作成するために使用する、USER TEMPORARY 表スペースを必要とします。

例

コマンド行プロセッサー (CLP) を使用して、*logretain* および *userexit* データベース構成パラメーターの値を変更します。オリジナルの (ディスク上の) 値と、更新済みの (メモリー内の) 値を GET_DB_CONFIG プロシージャーを呼び出して検索し、次いで結果のグローバル一時表 (DB_CONFIG) を照会します。

```
CONNECT TO SAMPLE

CREATE BUFFERPOOL MY8KPOOL SIZE 250 PAGESIZE 8K

CREATE USER TEMPORARY TABLESPACE MYTSP2 PAGESIZE
   8K MANAGED BY SYSTEM USING ( 'TSC2' ) BUFFERPOOL MY8KPOOL

UPDATE DB CFG USING LOGRETAIN RECOVERY USEREXIT ON

CALL SYSPROC.GET_DB_CONFIG()

SELECT DBCONFIG_TYPE, LOGRETAIN, USEREXIT
   FROM SESSION.DB_CONFIG

CONNECT RESET
```

以下はこの照会の出力例です。

```
DBCONFIG_TYPE LOGRETAIN   USEREXIT
------------- ----------- -----------
            0           1           1
            1           0           0

  2 record(s) selected.
```

GET_DBM_CONFIG

注: この表関数は使用すべきではなく、 358 ページの『DBMCFG 管理ビュー - データベース・マネージャー構成パラメーター情報の検索』 に置き換えられました。

スキーマは SYSFUN です。

GET_DBM_CONFIG 関数は、データベース・マネージャー構成情報を戻します。この関数は引数を取りません。

この関数は、パラメーターごとに 1 つの列を備えた 2 つの行を示す表を戻します。最初の列は、以下に示すように DBMCONFIG_TYPE という名前です。

表 273. GET_DBM_CONFIG 表関数によって戻される情報

列名	データ・タイプ	説明
DBMCONFIG_TYPE	INTEGER	この列の値が 0 の行には、ディスクに保管されたデータベース・マネージャー構成パラメーターの値が入ります。この列の値が 1 の行には、メモリーに保管されたデータベース・マネージャー構成パラメーターの現行値が入ります。

例

コマンド行プロセッサー (CLP) を使用し、*numdb* および *diaglevel* データベース・マネージャー構成パラメーターの値を変更し、次いでオリジナルの (ディスク上の) 値と、更新された (メモリー内の) 値を検索します。

```
UPDATE DBM CFG USING NUMDB 32 DIAGLEVEL 4

CONNECT TO SAMPLE

SELECT DBMCONFIG_TYPE, NUMDB, DIAGLEVEL
  FROM TABLE(SYSFUN.GET_DBM_CONFIG()) AS DBMCFG

CONNECT RESET
```

以下はこの照会の出力例です。

```
DBMCONFIG_TYPE NUMDB       DIAGLEVEL
-------------- ----------- -----------
             0          32           4
             1           8           3

  2 record(s) selected.
```

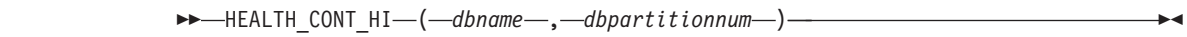
ヘルス・スナップショット・ルーチン

HEALTH_CONT_HI

HEALTH_CONT_HI 表関数は、表スペース・コンテナーのヘルス・インディケーター情報を、データベースの表スペースのヘルス・スナップショットから戻します。

重要: バージョン 9.7 でヘルス・モニターが非推奨になったため、この表関数は推奨されておらず、将来のリリースで除去される可能性があります。 詳しくは、「*DB2 バージョン 9.7 の新機能*」のトピック『ヘルス・モニターが推奨されなくなった』を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(255) の入力引数。"Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

HEALTH_CONT_HI 表関数に対する EXECUTE 特権。

例

```
SELECT * FROM TABLE(HEALTH_CONT_HI('',-1)) AS T
```

以下はこの照会の出力例です。

```
SNAPSHOT_TIMESTAMP         CONTAINER_NAME                                  ...
-------------------------- ----------------------------------------------...- ...
2006-02-13-12.30.40.759542 D:¥DB2¥NODE0000¥SAMPLE¥T0000000¥C0000000.CAT    ...
2006-02-13-12.30.40.759542 D:¥DB2¥NODE0000¥SAMPLE¥T0000003¥C0000000.LRG    ...
2006-02-13-12.30.40.759542 D:¥DB2¥NODE0000¥SAMPLE¥T0000004¥C0000000.UTM    ...
2006-02-13-12.30.40.759542 D:¥DB2¥NODE0000¥SAMPLE¥T0000001¥C0000000.TMP    ...
2006-02-13-12.30.40.759542 D:¥DB2¥NODE0000¥SAMPLE¥T0000002¥C0000000.LRG    ...

  5 record(s) selected.
```

この照会からの出力 (続き)。

```
... NODE_NUMBER HI_ID                HI_VALUE HI_TIMESTAMP                ...
... ----------- -------------------- -------- --------------------------- ...
...           -                 3001        1 2006-02-13-12.26.26.158000 ...
...           -                 3001        1 2006-02-13-12.26.26.158000 ...
...           -                 3001        1 2006-02-13-12.26.26.158000 ...
...           -                 3001        1 2006-02-13-12.26.26.158000 ...
...           -                 3001        1 2006-02-13-12.26.26.158000 ...
```

この照会からの出力 (続き)。

```
... HI_ALERT_STATE       HI_ALERT_STATE_DETAIL HI_FORMULA      HI_ADDITIONAL_INFO
... -------------------- --------------------- -----------...- ------------------
...                    1 Normal                1                                -
...                    1 Normal                1                                -
...                    1 Normal                1                                -
...                    1 Normal                1                                -
...                    1 Normal                1                                -
```

戻される情報

表 274. HEALTH_CONT_HI 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
CONTAINER_NAME	VARCHAR(256)	container_name - コンテナー名
NODE_NUMBER	INTEGER	node_number - ノード番号
HI_ID	BIGINT	スナップショット・データ・ストリーム内のヘルス・インディケーターを固有に識別する番号。
HI_VALUE	SMALLINT	ヘルス・インディケーターの値。
HI_TIMESTAMP	TIMESTAMP	アラートが生成された日時。
HI_ALERT_STATE	BIGINT	アラートの重大度。
HI_ALERT_STATE_DETAIL	VARCHAR(20)	HI_ALERT_STATE 列のテキスト記述。
HI_FORMULA	VARCHAR(2048)	ヘルス・インディケーターを計算するのに使用する公式。
HI_ADDITIONAL_INFO	VARCHAR(4096)	ヘルス・インディケーターに関する追加情報。

HEALTH_CONT_HI_HIS

データベースのヘルス・スナップショットからコンテナーに関するヘルス・インディケーター履歴情報を戻します。

重要: バージョン 9.7 でヘルス・モニターが非推奨になったため、この表関数は推奨されておらず、将来のリリースで除去される可能性があります。 詳しくは、「*DB2 バージョン 9.7 の新機能*」のトピック『ヘルス・モニターが推奨されなくなった』を参照してください。

構文

スキーマは SYSPROC です。

表関数パラメーター

dbname
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(255) の入力引数。"Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

HEALTH_CONT_HI_HIS 表関数に対する EXECUTE 特権。

例

```
SELECT * FROM TABLE(HEALTH_CONT_HI_HIS('',-1)) AS T
```

以下はこの照会の出力例です。

```
SNAPSHOT_TIMESTAMP         CONTAINER_NAME                                  ...
-------------------------- ----------------------------------------...- ...
2006-02-13-12.30.41.915646 D:¥DB2¥NODE0000¥SAMPLE¥T0000000¥C0000000.CAT ...
2006-02-13-12.30.41.915646 D:¥DB2¥NODE0000¥SAMPLE¥T0000000¥C0000000.CAT ...
2006-02-13-12.30.41.915646 D:¥DB2¥NODE0000¥SAMPLE¥T0000003¥C0000000.LRG ...
2006-02-13-12.30.41.915646 D:¥DB2¥NODE0000¥SAMPLE¥T0000003¥C0000000.LRG ...
2006-02-13-12.30.41.915646 D:¥DB2¥NODE0000¥SAMPLE¥T0000004¥C0000000.UTM ...
2006-02-13-12.30.41.915646 D:¥DB2¥NODE0000¥SAMPLE¥T0000004¥C0000000.UTM ...
2006-02-13-12.30.41.915646 D:¥DB2¥NODE0000¥SAMPLE¥T0000001¥C0000000.TMP ...
2006-02-13-12.30.41.915646 D:¥DB2¥NODE0000¥SAMPLE¥T0000001¥C0000000.TMP ...
2006-02-13-12.30.41.915646 D:¥DB2¥NODE0000¥SAMPLE¥T0000002¥C0000000.LRG ...
2006-02-13-12.30.41.915646 D:¥DB2¥NODE0000¥SAMPLE¥T0000002¥C0000000.LRG ...

  10 record(s) selected.
```

この照会からの出力 (続き)。

```
... NODE_NUMBER HI_ID     HI_TIMESTAMP               HI_VALUE HI_ALERT_STATE ...
... ----------- -----...- -------------------------- -------- -------------- ...
...           -      3001 2006-02-13-12.16.25.911000        1              1 ...
...           -      3001 2006-02-13-12.06.26.168000        1              1 ...
...           -      3001 2006-02-13-12.16.25.911000        1              1 ...
...           -      3001 2006-02-13-12.06.26.168000        1              1 ...
...           -      3001 2006-02-13-12.16.25.911000        1              1 ...
...           -      3001 2006-02-13-12.06.26.168000        1              1 ...
```

```
...          -      3001 2006-02-13-12.16.25.911000        1              1 ...
...          -      3001 2006-02-13-12.06.26.168000        1              1 ...
...          -      3001 2006-02-13-12.16.25.911000        1              1 ...
...          -      3001 2006-02-13-12.06.26.168000        1              1 ...
```

この照会からの出力 (続き)。

```
... HI_ALERT_STATE_DETAIL HI_FORMULA      HI_ADDITIONAL_INFO
... --------------------- ------------...- ------------------
... Normal                1                                 -
... Normal                1                                 -
... Normal                1                                 -
... Normal                1                                 -
... Normal                1                                 -
... Normal                1                                 -
... Normal                1                                 -
... Normal                1                                 -
... Normal                1                                 -
... Normal                1                                 -
```

戻される情報

表 275. HEALTH_CONT_HI_HIS 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
CONTAINER_NAME	VARCHAR(256)	**container_name** - コンテナー名
NODE_NUMBER	INTEGER	**node_number** - ノード番号
HI_ID	BIGINT	スナップショット・データ・ストリーム内のヘルス・インディケーターを固有に識別する番号。
HI_TIMESTAMP	TIMESTAMP	アラートが生成された日時。
HI_VALUE	SMALLINT	ヘルス・インディケーターの値。
HI_ALERT_STATE	BIGINT	アラートの重大度。
HI_ALERT_STATE_DETAIL	VARCHAR(20)	HI_ALERT_STATE 列のテキスト記述。
HI_FORMULA	VARCHAR(2048)	ヘルス・インディケーターを計算するのに使用する公式。
HI_ADDITIONAL_INFO	VARCHAR(4096)	ヘルス・インディケーターに関する追加情報。

HEALTH_CONT_INFO

HEALTH_CONT_INFO 表関数は、コンテナー情報をデータベースのヘルス・スナップショットから戻します。

重要: バージョン 9.7 でヘルス・モニターが非推奨になったため、この表関数は推奨されておらず、将来のリリースで除去される可能性があります。 詳しくは、「*DB2 バージョン 9.7 の新機能*」のトピック『ヘルス・モニターが推奨されなくなった』を参照してください。

構文

スキーマは SYSPROC です。

表関数パラメーター

dbname
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(255) の入力引数。"Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

HEALTH_CONT_INFO 表関数に対する EXECUTE 特権。

例

```
SELECT * FROM TABLE(HEALTH_CONT_INFO('',-1)) AS T
```

以下はこの照会の出力例です。

```
SNAPSHOT_TIMESTAMP         CONTAINER_NAME                                  ...
-------------------------- ----------------------------------------...- ...
2006-02-13-12.30.40.541209 D:¥DB2¥NODE0000¥SAMPLE¥T0000000¥C0000000.CAT  ...
2006-02-13-12.30.40.541209 D:¥DB2¥NODE0000¥SAMPLE¥T0000003¥C0000000.LRG  ...
2006-02-13-12.30.40.541209 D:¥DB2¥NODE0000¥SAMPLE¥T0000004¥C0000000.UTM  ...
2006-02-13-12.30.40.541209 D:¥DB2¥NODE0000¥SAMPLE¥T0000001¥C0000000.TMP  ...
2006-02-13-12.30.40.541209 D:¥DB2¥NODE0000¥SAMPLE¥T0000002¥C0000000.LRG  ...

  5 record(s) selected.
```

この照会からの出力 (続き)。

```
... TABLESPACE_NAME     NODE_NUMBER ...
... ---------------...- ----------- ...
... SYSCATSPACE                   - ...
```

```
... SYSTOOLSPACE                - ...
... SYSTOOLSTMPSPACE            - ...
... TEMPSPACE1                  - ...
... USERSPACE1                  - ...
```

この照会からの出力 (続き)。

```
... ROLLED_UP_ALERT_STATE ROLLED_UP_ALERT_STATE_DETAIL
... -------------------- --------------------------
...                    1 Normal
...                    1 Normal
...                    1 Normal
...                    1 Normal
...                    1 Normal
```

戻される情報

表 276. HEALTH_CONT_INFO 表関数によって戻される情報

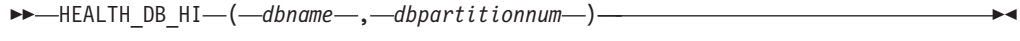
列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
CONTAINER_NAME	VARCHAR(256)	container_name - コンテナー名
TABLESPACE_NAME	VARCHAR(128)	tablespace_name - 表スペース名
NODE_NUMBER	INTEGER	node_number - ノード番号
ROLLED_UP_ALERT_STATE	BIGINT	このスナップショットでキャプチャーされた最も重大なアラート状態。
ROLLED_UP_ALERT_STATE_DETAIL	VARCHAR(20)	ROLLED_UP_ALERT_STATE 列のテキスト記述。

HEALTH_DB_HI

HEALTH_DB_HI 表関数は、データベースのヘルス・スナップショットからヘルス・インディケーターの情報を戻します。

重要: バージョン 9.7 でヘルス・モニターが非推奨になったため、この表関数は推奨されておらず、将来のリリースで除去される可能性があります。 詳しくは、「*DB2 バージョン 9.7 の新機能*」のトピック『ヘルス・モニターが推奨されなくなった』を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(255) の入力引数。"Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。データベース・インスタンス下のすべてのデータベースからスナップショットを取る場合は、 NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

HEALTH_DB_HI 表関数に対する EXECUTE 特権。

例

```
SELECT * FROM TABLE(HEALTH_DB_HI('',-1)) AS T
```

以下はこの照会の出力例です。

```
SNAPSHOT_TIMESTAMP         HI_ID     DB_NAME     HI_VALUE ...
-------------------------- -----...- -------...- -------- ...
2006-02-13-12.30.23.949888      1001 SAMPLE             0 ...
2006-02-13-12.30.23.949888      1002 SAMPLE             0 ...
2006-02-13-12.30.23.949888      1003 SAMPLE             0 ...
2006-02-13-12.30.23.949888      1005 SAMPLE             6 ...
2006-02-13-12.30.23.949888      1006 SAMPLE            53 ...
2006-02-13-12.30.23.949888      1008 SAMPLE             3 ...
2006-02-13-12.30.23.949888      1010 SAMPLE             0 ...
2006-02-13-12.30.23.949888      1014 SAMPLE            74 ...
2006-02-13-12.30.23.949888      1015 SAMPLE             1 ...
2006-02-13-12.30.23.949888      1018 SAMPLE             1 ...
2006-02-13-12.30.23.949888      1022 SAMPLE             1 ...

  11 record(s) selected.
```

この照会からの出力 (続き)。

```
... HI_TIMESTAMP              HI_ALERT_STATE HI_ALERT_STATE_DETAIL ...
... -------------------------- -------------- -------------------- ...
... 2006-02-13-12.26.26.158000              1 Normal               ...
... 2006-02-13-12.26.26.158000              1 Normal               ...
... 2006-02-13-12.26.26.158000              1 Normal               ...
... 2006-02-13-12.26.26.158000              1 Normal               ...
... 2006-02-13-12.26.26.158000              1 Normal               ...
... 2006-02-13-12.26.26.158000              1 Normal               ...
... 2006-02-13-12.26.26.158000              1 Normal               ...
... 2006-02-13-12.26.26.158000              1 Normal               ...
... 2006-02-13-12.30.25.640000              2 Attention            ...
... 2006-02-13-12.30.25.640000              2 Attention            ...
... 2006-02-13-12.29.25.281000              2 Attention            ...
```

この照会からの出力 (続き)。

```
... HI_FORMULA                                    ...
... ----------------------------------------...- ...
... 0                                             ...
... ((0 / 5000) * 100)                            ...
...                                               ...
...                                               ...
...                                               ...
...                                               ...
...                                               ...
...                                               ...
...                                               ...
... (((0 - 0) / ((118 - 0) + 1)) * 100)           ...
...                                               ...
...                                               ...
...                                               ...
...                                               ...
...                                               ...
... ((1170384 / (1170384 + 19229616)) * 100)      ...
...                                               ...
...                                               ...
...                                               ...
...                                               ...
...                                               ...
... ((11155116032 / 21138935808) * 100)           ...
...                                               ...
...                                               ...
...                                               ...
...                                               ...
...                                               ...
...                                               ...
...                                               ...
... ((5264 / (50 * 4096)) * 100)                  ...
... ((0 / 5) * 100)                               ...
... ((4587520 / 6160384) * 100)                   ...
... -                                             ...
...                                               ...
...                                               ...
...                                               ...
...                                               ...
...                                               ...
...                                               ...
... -                                             ...
...                                               ...
...                                               ...
...                                               ...
...                                               ...
...                                               ...
...                                               ...
...                                               ...
... -                                             ...
...                                               ...
...                                               ...
```

この照会からの出力 (続き)。

```
... HI_ADDITIONAL_INFO
... ------------------------------------
... -
... The high watermark for shared sort
...   memory is "57". "99"% of the time
...   the sort heap allocation is less
...   than or equal to "246". The sort
...   heap (sortheap) database
...   configuration parameter is set
...   to "256". The high watermark for
...   private sort memory is "0".
```

```
... The sort heap (sortheap) database
...   configuration parameter is set to
...   "256". The high watermark for
...   private sort memory is "57". The
...   high watermark for shared sort
...   memory is "0"
... The following are the related
...   database configuration parameter
...   settings: logprimary is "3",
...   logsecond is "2", and logfilsiz
...   is "1000". The application with
...   the oldest transaction is "712".
... The following are the related
...   database configuration parameter
...   settings: logprimary is "3",
...   logsecond is "2", and logfilsiz
...   is "1000", blk_log_dsk_ful is
...   "NO", userexit is "NO",
...   logarchmeth1 is "OFF" and
...   logarchmeth2 is "OFF".
... -
... -
... -
... The scope setting in the reorganization
...   policy is "TABSCHEMA NOT LIKE 'SYS%'".
...   Automatic reorganization (AUTO_REORG)
...   for this database is set to "OFF".
...   The longest estimated reorganization
...   time is "N/A".
... The last successful backup was taken
...   at "N/A". The log space consumed since
...   this last backup has been "N/A" 4KB
...   pages. Automation for database backup
...   is set to "OFF". The last automated
...   backup returned with SQLCODE = "N/A".
...   The longest estimated backup time
...   is "N/A".
... The scope is "N¥A". Automatic
...   statistics collection (AUTO_RUNSTATS)
...   is set to "OFF".
```

戻される情報

表 277. HEALTH_DB_HI 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
HI_ID	BIGINT	スナップショット・データ・ストリーム内のヘルス・インディケーターを固有に識別する番号。
DB_NAME	VARCHAR(128)	db_name - データベース名
HI_VALUE	SMALLINT	ヘルス・インディケーターの値。
HI_TIMESTAMP	TIMESTAMP	アラートが生成された日時。
HI_ALERT_STATE	BIGINT	アラートの重大度。
HI_ALERT_STATE_DETAIL	VARCHAR(20)	HI_ALERT_STATE 列のテキスト記述。

表 277. HEALTH_DB_HI 表関数によって戻される情報 (続き)

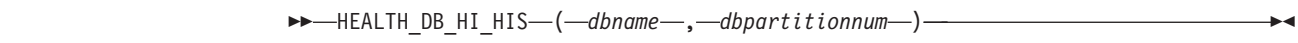
列名	データ・タイプ	説明または対応するモニター・エレメント
HI_FORMULA	VARCHAR(2048)	ヘルス・インディケーターを計算するのに使用する公式。
HI_ADDITIONAL_INFO	VARCHAR(4096)	ヘルス・インディケーターに関する追加情報。

HEALTH_DB_HI_HIS

HEALTH_DB_HI_HIS 表関数は、データベースのヘルス・スナップショットからヘルス・インディケーターの履歴情報を戻します。

重要: バージョン 9.7 でヘルス・モニターが非推奨になったため、この表関数は推奨されておらず、将来のリリースで除去される可能性があります。 詳しくは、「*DB2 バージョン 9.7 の新機能*」のトピック『ヘルス・モニターが推奨されなくなった』を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(255) の入力引数。"Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。データベース・インスタンス下のすべてのデータベースからスナップショットを取る場合は、 NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

HEALTH_DB_HI_HIS 表関数に対する EXECUTE 特権。

例

```
SELECT * FROM TABLE(HEALTH_DB_HI_HIS('',-1)) AS T
```

以下はこの照会の出力例です。

```
SNAPSHOT_TIMESTAMP         HI_ID     DB_NAME     HI_VALUE ...
-------------------------- -----...- -------...- -------- ...
2006-02-13-12.30.26.325627      1001 SAMPLE             0 ...
...                                                       ...
2006-02-13-12.30.26.325627      1002 SAMPLE             0 ...
...                                                       ...
2006-02-13-12.30.26.325627      1003 SAMPLE             0 ...
...                                                       ...
2006-02-13-12.30.26.325627      1005 SAMPLE             3 ...
...                                                       ...
2006-02-13-12.30.26.325627      1008 SAMPLE             2 ...
...                                                       ...
2006-02-13-12.30.26.325627      1010 SAMPLE             0 ...
...                                                       ...
2006-02-13-12.30.26.325627      1014 SAMPLE            73 ...
...                                                       ...
2006-02-13-12.30.26.325627      1015 SAMPLE             1 ...
...                                                       ...
2006-02-13-12.30.26.325627      1018 SAMPLE             1 ...
...                                                       ...
2006-02-13-12.30.26.325627      1022 SAMPLE             1 ...
...
```

この照会からの出力 (続き)。

```
... HI_TIMESTAMP              HI_ALERT_STATE       HI_ALERT_STATE_DETAIL ...
... -------------------------- -------------------- --------------------- ...
... 2006-02-13-12.21.25.649000                   1 Normal                ...
... ...                                                                  ...
... 2006-02-13-12.21.25.649000                   1 Normal                ...
... ...                                                                  ...
... 2006-02-13-12.20.25.182000                   1 Normal                ...
... ...                                                                  ...
... 2006-02-13-12.16.25.911000                   1 Normal                ...
... ...                                                                  ...
... 2006-02-13-12.16.25.911000                   1 Normal                ...
... ...                                                                  ...
... 2006-02-13-12.16.25.911000                   1 Normal                ...
... ...                                                                  ...
... 2006-02-13-12.21.25.649000                   1 Normal                ...
... ...                                                                  ...
... 2006-02-13-12.29.55.461000                   2 Attention             ...
... ...                                                                  ...
... 2006-02-13-12.29.25.281000                   2 Attention             ...
... ...                                                                  ...
... 2006-02-13-12.27.55.743000                   2 Attention             ...
... ...
```

この照会からの出力 (続き)。

```
... HI_FORMULA                                ...
... ------------------------------------------ ...
... 0                                         ...
... ...                                       ...
... ((0 / 5000) * 100)                        ...
...                                           ...
...                                           ...
...                                           ...
...                                           ...
...                                           ...
...                                           ...
```

```
...                                                 ...
... ...                                             ...
... (((0 - 0) / ((68 - 0) + 1)) * 100)              ...
...                                                 ...
...                                                 ...
...                                                 ...
...                                                 ...
...                                                 ...
... ...                                             ...
... ((698410 / (698410 + 19701590)) * 100)          ...
...                                                 ...
...                                                 ...
...                                                 ...
...                                                 ...
...                                                 ...
... ...                                             ...
... ((3920 / (50 * 4096)) * 100)                    ...
... ...                                             ...
... ((0 / 4) * 100)                                 ...
... ...                                             ...
... ((4521984 / 6160384) * 100)                     ...
... ...                                             ...
... -                                               ...
...                                                 ...
...                                                 ...
...                                                 ...
...                                                 ...
...                                                 ...
...                                                 ...
...                                                 ...
... ...                                             ...
... -                                               ...
...                                                 ...
...                                                 ...
...                                                 ...
...                                                 ...
...                                                 ...
...                                                 ...
...                                                 ...
... ...                                             ...
... -                                               ...
...                                                 ...
...                                                 ...
... ...                                             ...
```

この照会からの出力 (続き)。

```
... HI_ADDITIONAL_INFO
... ------------------------------------
... -
...
... The high watermark for shared sort
...   memory is "15". "99"% of the time
...   the sort heap allocation is less
...   than or equal to "246". The sort
...   heap (sortheap) database
...   configuration parameter is set
...   to "256". The high watermark
...   for private sort memory is "0".
...
... The sort heap (sortheap) database
...   configuration parameter is set
...   to "256". The high watermark for
...   private sort memory is "15". The
...   high watermark for shared sort
...   memory is "0"
...
```

```
... The following are the related
...   database configuration parameter
...   settings: logprimary is "3",
...   logsecond is "2", and logfilsiz
...   is "1000". The application with
...   the oldest transaction is "712".
...
... -
...
... -
...
... -
...
... The scope setting in the
...   reorganization policy is
...   "TABSCHEMA NOT LIKE 'SYS%'".
...   Automatic reorganization
...   (AUTO_REORG) for this database
...   is set to "OFF". The longest
...   estimated reorganization time
...   is "N/A".
...
... The last successful backup was taken
...   at "N/A". The log space consumed
...   since this last backup has been
...   "N/A" 4KB pages. Automation for
...   database backup is set to "OFF". The
...   last automated backup returned with
...   SQLCODE = "N/A". The longest
...   estimated backup time is "N/A".
...
... The scope is "N¥A". Automatic
...   statistics collection
...   (AUTO_RUNSTATS) is set to "OFF".
...
```

戻される情報

表 278. HEALTH_DB_HI_HIS 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
HI_ID	BIGINT	スナップショット・データ・ストリーム内のヘルス・インディケーターを固有に識別する番号。
DB_NAME	VARCHAR(128)	db_name - データベース名
HI_VALUE	SMALLINT	ヘルス・インディケーターの値。
HI_TIMESTAMP	TIMESTAMP	アラートが生成された日時。
HI_ALERT_STATE	BIGINT	アラートの重大度。
HI_ALERT_STATE_DETAIL	VARCHAR(20)	HI_ALERT_STATE 列のテキスト記述。
HI_FORMULA	VARCHAR(2048)	ヘルス・インディケーターを計算するのに使用する公式。

表 278. HEALTH_DB_HI_HIS 表関数によって戻される情報 (続き)

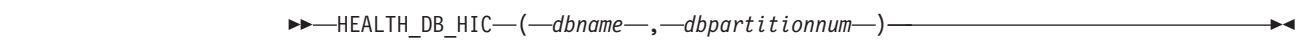
列名	データ・タイプ	説明または対応するモニター・エレメント
HI_ADDITIONAL_INFO	VARCHAR(4096)	ヘルス・インディケーターに関する追加情報。

HEALTH_DB_HIC

HEALTH_DB_HIC 関数は、データベースのヘルス・スナップショットからコレクション・ヘルス・インディケーターの情報を戻します。

重要: バージョン 9.7 でヘルス・モニターが非推奨になったため、この表関数は推奨されておらず、将来のリリースで除去される可能性があります。 詳しくは、「*DB2 バージョン 9.7 の新機能*」のトピック『ヘルス・モニターが推奨されなくなった』を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(255) の入力引数。"Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。データベース・インスタンス下のすべてのデータベースからスナップショットを取る場合は、 NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションに -1、またはすべてのアクティブなデータベース・パーティションに -2 を指定します。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

HEALTH_DB_HIC 表関数に対する EXECUTE 特権。

例

```
SELECT * FROM TABLE(HEALTH_DB_HIC('',-1)) AS T
```

以下はこの照会の出力例です。

```
SNAPSHOT_TIMESTAMP         HI_ID     DB_NAME     ...
-------------------------- -----...- -------...- ...
2006-02-13-12.30.33.870959      1015 SAMPLE      ...
2006-02-13-12.30.33.870959      1022 SAMPLE      ...

  2 record(s) selected.
```

この照会からの出力 (続き)。

```
... HI_OBJ_NAME                              HI_OBJ_DETAIL     ...
... -------------------------------...-- -------------...- ...
... "JESSICAE"."EMPLOYEE"                    REORG TABLE       ...
... "SYSIBM"."SYSDATAPARTITIONEXPRESSION"  RUNSTATS          ...
```

この照会からの出力 (続き)。

```
... HI_OBJ_STATE HI_OBJ_STATE_DETAIL  HI_TIMESTAMP
... ------------ -------------------- --------------------------
...            2 Attention            2006-02-13-12.24.27.000000
...            2 Attention            2006-02-13-12.29.26.000000
```

戻される情報

表 279. HEALTH_DB_HIC 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
HI_ID	BIGINT	スナップショット・データ・ストリーム内のヘルス・インディケーターを固有に識別する番号。
DB_NAME	VARCHAR(128)	**db_name** - データベース名
HI_OBJ_NAME	VARCHAR(256)	コレクション内のオブジェクトを固有に識別する名前。
HI_OBJ_DETAIL	VARCHAR(4096)	オブジェクトがコレクションに追加された理由を記述したテキスト。

表 279. HEALTH_DB_HIC 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
HI_OBJ_STATE	BIGINT	オブジェクトの状態。有効な状態 (sqlmon.h に定義されています) は次のとおりです。 • NORMAL (1)。このオブジェクトに対してアクションは必要ありません。 • ATTENTION (2)。このヘルス・インディケーターに対して自動化は有効になっていません。手動でアクションをとる必要があります。 • AUTOMATED (5)。このヘルス・インディケーターに対して自動化が有効になっています。アクションは自動的に開始します。 • AUTOMATE_FAILED (6)。このヘルス・インディケーターに対して自動化が有効になっています。アクションは開始しましたが、正常に完了できませんでした。現時点で、手操作による介入が必要です。
HI_OBJ_STATE_DETAIL	VARCHAR(20)	HI_OBJ_STATE 列の値を変換したストリングのバージョン。
HI_TIMESTAMP	TIMESTAMP	アラートが生成された日時。

HEALTH_DB_HIC_HIS

データベースのヘルス・スナップショットからコレクション・ヘルス・インディケーターの履歴情報を戻します。

重要: バージョン 9.7 でヘルス・モニターが非推奨になったため、この表関数は推奨されておらず、将来のリリースで除去される可能性があります。 詳しくは、「*DB2 バージョン 9.7 の新機能*」のトピック『ヘルス・モニターが推奨されなくなった』を参照してください。

構文

スキーマは SYSPROC です。

表関数パラメーター

dbname
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(255) の入力引数。"Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。データベース・インスタンス下のすべてのデータベースからスナップショットを取る場合は、 NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションに -1、またはすべてのアクティブなデータベース・パーティションに -2 を指定します。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

HEALTH_DB_HIC_HIS 表関数に対する EXECUTE 特権。

例

```
SELECT * FROM TABLE(HEALTH_DB_HIC_HIS('',-1)) AS T
```

以下はこの照会の出力例です。

```
HI_HIS_ENTRY_NUM SNAPSHOT_TIMESTAMP         HI_ID                ...
---------------- -------------------------- -------------------- ...
               1 2006-02-13-12.30.34.496720                 1015 ...
               2 2006-02-13-12.30.34.496720                 1022 ...
               3 2006-02-13-12.30.34.496720                 1022 ...
               4 2006-02-13-12.30.34.496720                 1022 ...
               5 2006-02-13-12.30.34.496720                 1022 ...
               6 2006-02-13-12.30.34.496720                 1022 ...
               7 2006-02-13-12.30.34.496720                 1022 ...
               8 2006-02-13-12.30.34.496720                 1022 ...
               9 2006-02-13-12.30.34.496720                 1022 ...
              10 2006-02-13-12.30.34.496720                 1022 ...

  10 record(s) selected.
```

この照会からの出力 (続き)。

```
... DB_NAME     HI_OBJ_NAME                                HI_OBJ_STATE ...
... -------...- ---------------------------------...- ------------ ...
... SAMPLE      "JESSICAE"."EMPLOYEE"                             2 ...
... SAMPLE      "SYSIBM"."SYSDATAPARTITIONEXPRESSION"             2 ...
... SAMPLE      "SYSIBM"."SYSDATAPARTITIONEXPRESSION"             2 ...
... SAMPLE      "SYSIBM"."SYSDATAPARTITIONEXPRESSION"             2 ...
... SAMPLE      "SYSIBM"."SYSDATAPARTITIONEXPRESSION"             1 ...
... SAMPLE      "SYSIBM"."SYSDATAPARTITIONEXPRESSION"             1 ...
... SAMPLE      "SYSIBM"."SYSDATAPARTITIONEXPRESSION"             1 ...
... SAMPLE      "SYSIBM"."SYSDATAPARTITIONEXPRESSION"             1 ...
... SAMPLE      "SYSIBM"."SYSDATAPARTITIONEXPRESSION"             1 ...
... SAMPLE      "SYSIBM"."SYSDATAPARTITIONEXPRESSION"             1 ...
```

この照会からの出力 (続き)。

```
... HI_OBJ_STATE_DETAIL  HI_TIMESTAMP
... -------------------- --------------------------
... Attention            2006-02-10-09.04.57.000000
... Attention            2006-02-13-12.27.56.000000
... Attention            2006-02-13-12.26.27.000000
... Attention            2006-02-13-12.24.56.000000
... Normal               2006-02-13-12.23.28.000000
... Normal               2006-02-13-12.21.56.000000
... Normal               2006-02-13-12.20.26.000000
... Normal               2006-02-13-12.18.57.000000
... Normal               2006-02-13-12.17.27.000000
... Normal               2006-02-13-12.15.56.000000
```

戻される情報

表 280. HEALTH_DB_HIC_HIS 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
HI_HIS_ENTRY_NUM	INTEGER	履歴項目を固有に識別する番号。
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
HI_ID	BIGINT	スナップショット・データ・ストリーム内のヘルス・インディケーターを固有に識別する番号。
DB_NAME	VARCHAR(128)	**db_name** - データベース名
HI_OBJ_NAME	VARCHAR(256)	コレクション内のオブジェクトを固有に識別する名前。

表 280. HEALTH_DB_HIC_HIS 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
HI_OBJ_STATE	BIGINT	オブジェクトの状態。有効な状態 (sqlmon.h に定義されています) は次のとおりです。 • NORMAL (1)。このオブジェクトに対してアクションは必要ありません。 • ATTENTION (2)。このヘルス・インディケーターに対して自動化は有効になっていません。手動でアクションをとる必要があります。 • AUTOMATED (5)。このヘルス・インディケーターに対して自動化が有効になっています。アクションは自動的に開始します。 • AUTOMATE_FAILED (6)。このヘルス・インディケーターに対して自動化が有効になっています。アクションは開始しましたが、正常に完了できませんでした。現時点で、手操作による介入が必要です。
HI_OBJ_STATE_DETAIL	VARCHAR(20)	HI_OBJ_STATE 列の値を変換したストリングのバージョン。
HI_TIMESTAMP	TIMESTAMP	アラートが生成された日時。

HEALTH_DB_INFO

HEALTH_DB_INFO 表関数は、データベースのヘルス・スナップショットからの情報を戻します。

重要: バージョン 9.7 でヘルス・モニターが非推奨になったため、この表関数は推奨されておらず、将来のリリースで除去される可能性があります。 詳しくは、「*DB2 バージョン 9.7 の新機能*」のトピック『ヘルス・モニターが推奨されなくなった』を参照してください。

構文

スキーマは SYSPROC です。

表関数パラメーター

dbname
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(255) の入力引数。"Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。データベース・インスタンス下のすべてのデータベースからスナップショットを取る場合は、 NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

HEALTH_DB_INFO 表関数に対する EXECUTE 特権。

例

```
SELECT * FROM TABLE(HEALTH_DB_INFO('',-1)) AS T
```

以下はこの照会の出力例です。

```
SNAPSHOT_TIMESTAMP         DB_NAME     INPUT_DB_ALIAS      ...
-------------------------- -------...- ---------------...- ...
2006-02-13-12.30.23.340081 SAMPLE      SAMPLE              ...

  1 record(s) selected.
```

この照会からの出力 (続き)。

```
... DB_PATH                       DB_LOCATION SERVER_PLATFORM ...
... ------------------------...- ----------- --------------- ...
... D:¥DB2¥NODE0000¥SQL00003¥              1               5 ...
```

この照会からの出力 (続き)。

```
... ROLLED_UP_ALERT_STATE ROLLED_UP_ALERT_STATE_DETAIL
... --------------------- ----------------------------
...                     4 Alarm
```

戻される情報

表 281. HEALTH_DB_INFO 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
DB_NAME	VARCHAR(128)	db_name - データベース名

表 281. HEALTH_DB_INFO 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
INPUT_DB_ALIAS	VARCHAR(128)	input_db_alias - 入力データベース別名
DB_PATH	VARCHAR(1024)	db_path - データベース・パス
DB_LOCATION	INTEGER	db_location - データベース・ロケーション
SERVER_PLATFORM	INTEGER	server_platform - サーバーのオペレーティング・システム
ROLLED_UP_ALERT_STATE	BIGINT	このスナップショットでキャプチャーされた最も重大なアラート状態。
ROLLED_UP_ALERT_STATE_DETAIL	VARCHAR(20)	ROLLED_UP_ALERT_STATE 列のテキスト記述。

HEALTH_DBM_HI

HEALTH_DBM_HI 表関数は、DB2 データベース・マネージャーのヘルス・スナップショットからヘルス・インディケーターの情報を戻します。

重要: バージョン 9.7 でヘルス・モニターが非推奨になったため、この表関数は推奨されておらず、将来のリリースで除去される可能性があります。 詳しくは、「*DB2 バージョン 9.7 の新機能*」のトピック『ヘルス・モニターが推奨されなくなった』を参照してください。

構文

スキーマは SYSPROC です。

表関数パラメーター

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

HEALTH_DBM_HI 表関数に対する EXECUTE 特権。

例

```
SELECT * FROM TABLE(HEALTH_DBM_HI(-1)) AS T
```

以下はこの照会の出力例です。

```
SNAPSHOT_TIMESTAMP         HI_ID     SERVER_INSTANCE_NAME     ...
-------------------------- -----...- --------------------...- ...
2006-02-13-12.30.19.773632         1 DB2                      ...
2006-02-13-12.30.19.773632         4 DB2                      ...

  2 record(s) selected.
```

この照会からの出力 (続き)。

```
... HI_VALUE HI_TIMESTAMP               HI_ALERT_STATE HI_ALERT_STATE_DETAIL ...
... -------- -------------------------- -------------- --------------------- ...
...        0 2006-02-13-12.26.26.158000              1 Normal                ...
...      100 2006-02-13-12.26.26.158000              4 Alarm                 ...
```

この照会からの出力 (続き)。

```
... HI_FORMULA                 HI_ADDITIONAL_INFO
... ---------------------...-  ------------------
... 0                                           -
... ((327680 / 327680) * 100)                   -
```

表 282. HEALTH_DBM_HI 表関数によって戻される情報

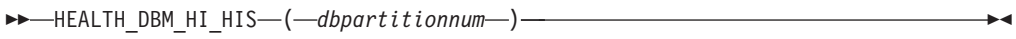
列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
HI_ID	BIGINT	スナップショット・データ・ストリーム内のヘルス・インディケーターを固有に識別する番号。
SERVER_INSTANCE_NAME	VARCHAR(128)	server_instance_name - サーバー・インスタンス名
HI_VALUE	SMALLINT	ヘルス・インディケーターの値。
HI_TIMESTAMP	TIMESTAMP	アラートが生成された日時。
HI_ALERT_STATE	BIGINT	アラートの重大度。
HI_ALERT_STATE_DETAIL	VARCHAR(20)	HI_ALERT_STATE 列のテキスト記述。
HI_FORMULA	VARCHAR(2048)	ヘルス・インディケーターを計算するのに使用する公式。
HI_ADDITIONAL_INFO	VARCHAR(4096)	ヘルス・インディケーターに関する追加情報。

HEALTH_DBM_HI_HIS

HEALTH_DBM_HI_HIS 表関数は、DB2 データベース・マネージャーのヘルス・スナップショットからヘルス・インディケーターの履歴情報を戻します。

重要: バージョン 9.7 でヘルス・モニターが非推奨になったため、この表関数は推奨されておらず、将来のリリースで除去される可能性があります。 詳しくは、「*DB2 バージョン 9.7 の新機能*」のトピック『ヘルス・モニターが推奨されなくなった』を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

HEALTH_DBM_HI_HIS 表関数に対する EXECUTE 特権。

例

```
SELECT * FROM TABLE(HEALTH_DBM_HI_HIS(-1)) AS T
```

以下はこの照会の出力例です。

```
SNAPSHOT_TIMESTAMP         HI_ID     SERVER_INSTANCE_NAME     HI_VALUE ...
-------------------------- -----...- --------------------...- -------- ...
2006-02-13-12.30.20.460905         1 DB2                             0 ...
2006-02-13-12.30.20.460905         1 DB2                             0 ...
2006-02-13-12.30.20.460905         1 DB2                             0 ...
2006-02-13-12.30.20.460905         1 DB2                             0 ...
2006-02-13-12.30.20.460905         1 DB2                             0 ...
2006-02-13-12.30.20.460905         1 DB2                             0 ...
2006-02-13-12.30.20.460905         1 DB2                             0 ...
2006-02-13-12.30.20.460905         1 DB2                             0 ...
2006-02-13-12.30.20.460905         1 DB2                             0 ...
2006-02-13-12.30.20.460905         4 DB2                           100 ...
2006-02-13-12.30.20.460905         4 DB2                           100 ...
2006-02-13-12.30.20.460905         4 DB2                           100 ...
2006-02-13-12.30.20.460905         4 DB2                           100 ...
2006-02-13-12.30.20.460905         4 DB2                            60 ...
2006-02-13-12.30.20.460905         4 DB2                            60 ...
2006-02-13-12.30.20.460905         4 DB2                            60 ...
2006-02-13-12.30.20.460905         4 DB2                            60 ...
2006-02-13-12.30.20.460905         4 DB2                            60 ...

  18 record(s) selected.
```

この照会の出力 (続き)。

```
... HI_TIMESTAMP              HI_ALERT_STATE HI_ALERT_STATE_DETAIL ...
... -------------------------- ------------- -------------------- ...
... 2006-02-13-12.21.25.649000             1 Normal               ...
```

```
... 2006-02-13-12.16.25.911000             1 Normal              ...
... 2006-02-13-12.11.25.377000             1 Normal              ...
... 2006-02-13-12.06.26.168000             1 Normal              ...
... 2006-02-13-12.01.25.165000             1 Normal              ...
... 2006-02-13-11.56.25.927000             1 Normal              ...
... 2006-02-13-11.51.25.452000             1 Normal              ...
... 2006-02-13-11.46.25.211000             1 Normal              ...
... 2006-02-13-11.41.25.972000             1 Normal              ...
... 2006-02-13-12.21.25.649000             4 Alarm               ...
... 2006-02-13-12.16.25.911000             4 Alarm               ...
... 2006-02-13-12.11.25.377000             4 Alarm               ...
... 2006-02-13-12.06.26.168000             4 Alarm               ...
... 2006-02-13-12.01.25.165000             1 Normal              ...
... 2006-02-13-11.56.25.927000             1 Normal              ...
... 2006-02-13-11.51.25.452000             1 Normal              ...
... 2006-02-13-11.46.25.211000             1 Normal              ...
... 2006-02-13-11.41.25.972000             1 Normal              ...
```

この照会の出力 (続き)。

```
... HI_FORMULA                   HI_ADDITIONAL_INFO
... -------------------------...- ------------------
... 0                                               -
... 0                                               -
... 0                                               -
... 0                                               -
... 0                                               -
... 0                                               -
... 0                                               -
... 0                                               -
... 0                                               -
... ((327680 / 327680) * 100)                       -
... ((327680 / 327680) * 100)                       -
... ((327680 / 327680) * 100)                       -
... ((327680 / 327680) * 100)                       -
... ((196608 / 327680) * 100)                       -
... ((196608 / 327680) * 100)                       -
... ((196608 / 327680) * 100)                       -
... ((196608 / 327680) * 100)                       -
... ((196608 / 327680) * 100)                       -
```

戻される情報

表 283. HEALTH_DBM_HI_HIS 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
HI_ID	BIGINT	スナップショット・データ・ストリーム内のヘルス・インディケーターを固有に識別する番号。
SERVER_INSTANCE_NAME	VARCHAR(128)	server_instance_name - サーバー・インスタンス名
HI_VALUE	SMALLINT	ヘルス・インディケーターの値。
HI_TIMESTAMP	TIMESTAMP	アラートが生成された日時。
HI_ALERT_STATE	BIGINT	アラートの重大度。

表 283. HEALTH_DBM_HI_HIS 表関数によって戻される情報 (続き)

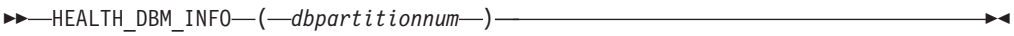
列名	データ・タイプ	説明または対応するモニター・エレメント
HI_ALERT_STATE_DETAIL	VARCHAR(20)	HI_ALERT_STATE 列のテキスト記述。
HI_FORMULA	VARCHAR(2048)	ヘルス・インディケーターを計算するのに使用する公式。
HI_ADDITIONAL_INFO	VARCHAR(4096)	ヘルス・インディケーターに関する追加情報。

HEALTH_DBM_INFO

HEALTH_DBM_INFO 関数は、DB2 データベース・マネージャーのヘルス・スナップショットからの情報を戻します。

重要: バージョン 9.7 でヘルス・モニターが非推奨になったため、この表関数は推奨されておらず、将来のリリースで除去される可能性があります。 詳しくは、「*DB2 バージョン 9.7 の新機能*」のトピック『ヘルス・モニターが推奨されなくなった』を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

HEALTH_DBM_INFO 表関数に対する EXECUTE 特権。

例

```
SELECT * FROM TABLE(HEALTH_DBM_INFO(-1)) AS T
```

以下はこの照会の出力例です。

```
SNAPSHOT_TIMESTAMP         SERVER_INSTANCE_NAME       ROLLED_UP_ALERT_STATE  ...
-------------------------- --------------------...--- ---------------------  ...
2006-02-13-12.30.19.663924 DB2                                            4  ...

  1 record(s) selected.
```

この照会からの出力 (続き)。

```
... ROLLED_UP_ALERT_STATE_DETAIL DB2START_TIME             ...
... --------------------------- -------------------------- ...
... Alarm                       2006-02-09-10.56.18.126182 ...
```

この照会からの出力 (続き)。

```
... LAST_RESET        NUM_NODES_IN_DB2_INSTANCE
... ----------...--- ------------------------
... -                                       1
```

戻される情報

表 284. HEALTH_DBM_INFO 表関数によって戻される情報

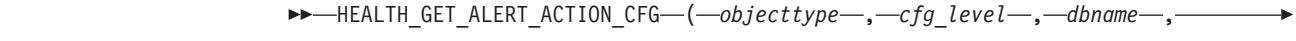
列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
SERVER_INSTANCE_NAME	VARCHAR(128)	server_instance_name - サーバー・インスタンス名
ROLLED_UP_ALERT_STATE	BIGINT	このスナップショットでキャプチャーされた最も重大なアラート状態。
ROLLED_UP_ALERT_STATE_DETAIL	VARCHAR(20)	ROLLED_UP_ALERT_STATE 列のテキスト記述。
DB2START_TIME	TIMESTAMP	db2start_time - データベース・マネージャー開始タイム・スタンプ
LAST_RESET	TIMESTAMP	last_reset - 最後のリセット・タイム・スタンプ
NUM_NODES_IN_DB2_INSTANCE	INTEGER	num_nodes_in_db2_instance - データベース・パーティション内のノード数

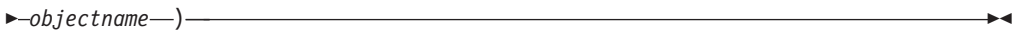
HEALTH_GET_ALERT_ACTION_CFG

さまざまなオブジェクト・タイプ (データベース・マネージャー、データベース、表スペース、表スペース・コンテナー)、およびさまざまな構成レベル (インストール・デフォルト、インスタンス、グローバル、およびオブジェクト) のヘルス・アラート・アクション構成設定値を戻します。

重要: バージョン 9.7 でヘルス・モニターが非推奨になったため、この表関数は推奨されておらず、将来のリリースで除去される可能性があります。 詳しくは、「*DB2 バージョン 9.7 の新機能*」のトピック『ヘルス・モニターが推奨されなくなった』を参照してください。

構文





スキーマは SYSPROC です。

表関数パラメーター

objecttype
オブジェクト・タイプを示す、タイプ VARCHAR(3) の入力引数。値は以下のいずれかでなければならず、大/小文字を区別しません。

- データベース・マネージャーは、'DBM'
- データベースは、'DB'
- 表スペースは、'TS'
- 表スペース・コンテナーは、'TSC'

注: 前後のスペースがあっても、それは無視されます。

cfg_level
構成レベルを表す、タイプ VARCHAR(1) の入力引数。値は以下のいずれかでなければならず、大/小文字を区別しません。

- *objecttype* が 'DBM' の場合: インストール・デフォルトは 'D'、インスタンス・レベルは 'G' または 'O'。
- *objecttype* が 'DBM' でない場合: インストール・デフォルトは 'D'、グローバル・レベルは 'G'、オブジェクト・レベルは 'O'。

dbname
データベース名を示す、タイプ VARCHAR(128) の入力引数。*objecttype* が 'DB'、'TS'、または 'TSC' で、かつ *cfg_level* が 'O' である場合、データベース名を指定しなければなりません。*objecttype* と *cfg_level* の組み合わせがそれ以外のものである場合、*dbname* パラメーターは NULL (または空ストリング) である必要があります。

objectname
オブジェクト名を示す、タイプ VARCHAR(1024) の入力引数。例えば、<table space name> または <table space name>.<container name> など。*objecttype* が 'TS' または 'TSC' で、かつ *cfg_level* が 'O' である場合、オブジェクト名を指定しなければなりません。*objecttype* と *cfg_level* の組み合わせがそれ以外のものである場合、*objectname* パラメーターは NULL (または空ストリング) である必要があります。

許可

HEALTH_GET_ALERT_ACTION_CFG 表関数に対する EXECUTE 特権。

例

例 1: データベース SAMPLE に関する、ヘルス・インディケーター ID 1004 のオブジェクト・レベルのアラート・アクション構成設定値を検索します。

```
SELECT OBJECTTYPE, CFG_LEVEL, SUBSTR(DBNAME,1,8) AS DBNAME,
   SUBSTR(OBJECTNAME,1,8) AS OBJECTNAME, ID, IS_DEFAULT,
   SUBSTR(CONDITION,1,10) AS CONDITION, ACTIONTYPE,
   SUBSTR(ACTIONNAME,1,30) AS ACTIONNAME, SUBSTR(USERID,1,8) AS USERID,
   SUBSTR(HOSTNAME,1,10) AS HOSTNAME, SCRIPT_TYPE,
```

```
   SUBSTR(WORKING_DIR,1,10) AS WORKING_DIR, TERMINATION_CHAR,
   SUBSTR(PARAMETERS,1,10) AS PARAMETERS
FROM TABLE(HEALTH_GET_ALERT_ACTION_CFG('DB','O','SAMPLE','')) AS ACTION_CFG
WHERE ID = 1004
```

以下はこの照会の出力例です。

```
OBJECTTYPE CFG_LEVEL DBNAME   OBJECTNAME ID        IS_DEFAULT CONDITION
---------- --------- -------- ---------- --...---- ---------- ----------
DB         O         SAMPLE                1004          1 ALARM
DB         O         SAMPLE                1004          1 ALARM

  2 record(s) selected.
```

この照会の出力 (続き)。

```
... ACTIONTYPE ACTIONNAME                    USERID   HOSTNAME
... ---------- ------------------------------ -------- ----------
... S          ~/health_center/script/scrpn6  uid1     -
... T          00.0005                        uid1     HOST3
```

この照会の出力 (続き)。

```
... SCRIPT_TYPE WORKING_DIR TERMINATION_CHAR PARAMETERS
... ----------- ----------- ---------------- ----------
... O           ~/health_c  -                -
... -           -           -                -
```

例 2: ヘルス・インディケーター ID 1004 の、データベース SAMPLE に関する状態、アクション・タイプ、アクション名、ホスト名、およびスクリプト・タイプを検索します。

```
SELECT CONDITION, ACTIONTYPE, SUBSTR(ACTIONNAME,1,35) AS ACTIONNAME,
   SUBSTR(USERID,1,8) AS USERID, SUBSTR(HOSTNAME,1,10) AS HOSTNAME, SCRIPT_TYPE
FROM TABLE(HEALTH_GET_ALERT_ACTION_CFG('DB','O','SAMPLE','')) AS ALERT_ACTION_CFG
WHERE ID=1004
```

以下はこの照会の出力例です。

```
CONDITION       ACTIONTYPE ACTIONNAME                          ...
---------...--- ---------- ------------------------------ ...
ALARM           S          ~/health_center/script/scrpn6    ...
ALARM           T          00.0005                          ...

  2 record(s) selected.
```

この照会の出力 (続き)。

```
... USERID   HOSTNAME   SCRIPT_TYPE
... -------- ---------- -----------
... uid1     -          O
... uid1     HOST3      -
```

使用上の注意

HEALTH_GET_IND_DEFINITION 表関数は、ヘルス・インディケーター ID をヘルス・インディケーター名にマップするために使用できます。

戻される情報

表 285. HEALTH_GET_ALERT_ACTION_CFG 表関数によって戻される情報

列名	データ・タイプ	説明
OBJECTTYPE	VARCHAR(3)	オブジェクト・タイプ。

表 285. HEALTH_GET_ALERT_ACTION_CFG 表関数によって戻される情報 (続き)

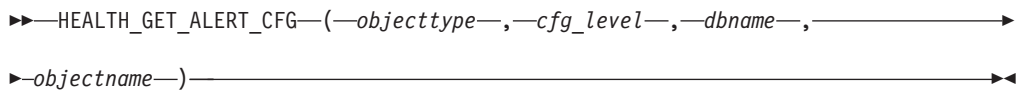
列名	データ・タイプ	説明
CFG_LEVEL	CHAR(1)	構成レベル。
DBNAME	VARCHAR(128)	データベース名。
OBJECTNAME	VARCHAR(512)	オブジェクト名。
ID	BIGINT	ヘルス・インディケーター ID。
IS_DEFAULT	SMALLINT	設定がデフォルトであるかどうか。デフォルトである場合は 1、デフォルトでない場合は 0、適用外である場合は NULL。
CONDITION	VARCHAR(512)	アクションが起動するアラート状態。
ACTIONTYPE	CHAR(1)	アクション・タイプ。スクリプト・アクションは 'S'、タスク・アクションは 'T'。
ACTIONNAME	VARCHAR(5000)	ACTIONTYPE が 'S' である場合、これはスクリプト・パス名です。ACTIONTYPE が 'T' である場合、これはタスク ID です。
USERID	VARCHAR(1024)	その下でアクションが実行されるユーザー名。
HOSTNAME	VARCHAR(255)	ホスト・システム名。
SCRIPT_TYPE	CHAR(1)	スクリプト・タイプ。ACTIONTYPE が 'S' の場合、オペレーティング・システムのコマンド・スクリプトは 'O'、DB2 コマンド・スクリプトは 'D'。ACTIONTYPE が 'T' の場合、NULL。
WORKING_DIR	VARCHAR(5000)	ACTIONTYPE が 'S' の場合、スクリプトの作業ディレクトリー。ACTIONTYPE が 'T' の場合、NULL。
TERMINATION_CHAR	VARCHAR(4)	DB2 コマンド・スクリプト・アクションである場合、ステートメント終了文字。その他の場合は NULL。
PARAMETERS	VARCHAR(200)	オペレーティング・システムのコマンド・スクリプト・アクションである場合、コマンド行パラメーター。

HEALTH_GET_ALERT_CFG

さまざまなオブジェクト・タイプ (データベース・マネージャー、データベース、表スペース、表スペース・コンテナー)、およびさまざまな構成レベル (インストール・デフォルト、グローバル、オブジェクト) のヘルス・アラート構成設定値を戻します。

重要: バージョン 9.7 でヘルス・モニターが非推奨になったため、この表関数は推奨されておらず、将来のリリースで除去される可能性があります。 詳しくは、「*DB2 バージョン 9.7 の新機能*」のトピック『ヘルス・モニターが推奨されなくなった』を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

objecttype
オブジェクト・タイプを示す、タイプ VARCHAR(3) の入力引数。値は以下のいずれかでなければならず、大/小文字を区別しません。

- データベース・マネージャーは、'DBM'
- データベースは、'DB'
- 表スペースは、'TS'
- 表スペース・コンテナーは、'TSC'

注: 前後のスペースがあっても、それは無視されます。

cfg_level
構成レベルを表す、タイプ VARCHAR(1) の入力引数。値は以下のいずれかでなければならず、大/小文字を区別しません。

- *objecttype* が 'DBM' の場合: インストール・デフォルトは 'D'、インスタンス・レベルは 'G' または 'O'。
- *objecttype* が 'DBM' でない場合: インストール・デフォルトは 'D'、グローバル・レベルは 'G'、オブジェクト・レベルは 'O'。

dbname
データベース名を示す、タイプ VARCHAR(128) の入力引数。*objecttype* が 'DB'、'TS'、または 'TSC' で、かつ *cfg_level* が 'O' である場合、データベース名を指定しなければなりません。*objecttype* と *cfg_level* の組み合わせがそれ以外のものである場合、*dbname* パラメーターは NULL (または空ストリング) である必要があります。

objectname
オブジェクト名を示す、タイプ VARCHAR(1024) の入力引数。例えば、<table space name> または <table space name>.<container name> など。*objecttype* が 'TS' または 'TSC' で、かつ *cfg_level* が 'O' である場合、オブジェクト名を指定

しなければなりません。*objecttype* と *cfg_level* の組み合わせがそれ以外のものである場合、*objectname* パラメーターは NULL (または空ストリング) である必要があります。

許可

HEALTH_GET_ALERT_CFG 表関数に対する EXECUTE 特権。

例

例 1: データベース SAMPLE のオブジェクト・レベルのアラート構成設定値を検索します。

```
SELECT * FROM TABLE(SYSPROC.HEALTH_GET_ALERT_CFG('DB','O','SAMPLE',''))
   AS ALERT_CFG
```

以下はこの照会の出力例です。

```
OBJECTTYPE CFG_LEVEL DBNAME           OBJECTNAME             ...
---------- --------- --------...------ -------------...------ ...
DB         0         SAMPLE                                  ...
DB         0         SAMPLE                                  ...
DB         0         SAMPLE                                  ...
DB         0         SAMPLE                                  ...
DB         0         SAMPLE                                  ...
DB         0         SAMPLE                                  ...
DB         0         SAMPLE                                  ...
DB         0         SAMPLE                                  ...
DB         0         SAMPLE                                  ...
DB         0         SAMPLE                                  ...
DB         0         SAMPLE                                  ...
DB         0         SAMPLE                                  ...
DB         0         SAMPLE                                  ...
DB         0         SAMPLE                                  ...
...                                                          ...
```

この照会の出力 (続き)。

```
... ID                   IS_DEFAULT WARNING_THRESHOLD     ...
... -------------------- ---------- --------------------  ...
...                1001          0                   0  ...
...                1018          0                   0  ...
...                1015          0                   0  ...
...                1022          0                   0  ...
...                1002          1                  95  ...
...                1003          1                  30  ...
...                1004          1                  60  ...
...                1005          1                  75  ...
...                1006          1                  75  ...
...                1007          1                   5  ...
...                1008          1                  75  ...
...                1009          1                   5  ...
...                1010          1                  50  ...
...                1011          1                  80  ...
```

この照会の出力 (続き)。

```
... ALARM_THRESHOLD      SENSITIVITY          EVALUATE ACTION_ENABLED
... -------------------- -------------------- -------- --------------
...                    0                    0        0              0
...                    0                    0        1              0
...                    0                    0        1              0
...                    0                    0        1              0
...                  100                    0        0              0
...                   50                    0        1              0
```

```
...                  30                   0       1            0
...                  85                   0       1            0
...                  85                   0       1            0
...                  10                   0       1            0
...                  85                   0       1            0
...                  10                   0       1            0
...                  70                   0       1            0
...                  70                   0       0            0
```

例 2: データベース SAMPLE の表スペース USERSPACE1 に関する、ヘルス・インディケーター ID '2002' の警告およびアラームのしきい値を検索します。

```
SELECT WARNING_THRESHOLD, ALARM_THRESHOLD
   FROM TABLE(SYSPROC.HEALTH_GET_ALERT_CFG('TS','O','SAMPLE','USERSPACE1'))
   AS T WHERE ID = 2002
```

以下はこの照会の出力例です。

```
WARNING_THRESHOLD    ALARM_THRESHOLD
-------------------- --------------------
                  80                   90
SQL22004N  Cannot find the requested configuration for the given object.
Returning default configuration for "tablespaces".


  1 record(s) selected with 1 warning messages printed.
```

使用上の注意

HEALTH_GET_IND_DEFINITION 表関数は、ヘルス・インディケーター ID をヘルス・インディケーター名にマップするために使用できます。

例: データベース SAMPLE の表スペース USERSPACE1 に関する、ヘルス・インディケーター「表スペース使用率」(ts.ts_util) の警告およびアラームのしきい値を検索します。

```
WITH HINAME(ID) AS (SELECT ID FROM TABLE(SYSPROC.HEALTH_GET_IND_DEFINITION('')) AS W
   WHERE NAME = 'ts.ts_util')
SELECT WARNING_THRESHOLD, ALARM_THRESHOLD
   FROM TABLE(SYSPROC.HEALTH_GET_ALERT_CFG('TS','O','SAMPLE','USERSPACE1')) AS T,
   HINAME AS H
   WHERE T.ID = H.ID
```

以下はこの照会の出力例です。

```
WARNING_THRESHOLD    ALARM_THRESHOLD
-------------------- --------------------
                  80                   90
SQL22004N  Cannot find the requested configuration for the given object.
Returning default configuration for "tablespaces".


  1 record(s) selected with 1 warning messages printed.
```

戻される情報

表 286. HEALTH_GET_ALERT_CFG 表関数によって戻される情報

列名	データ・タイプ	説明
OBJECTTYPE	VARCHAR(3)	オブジェクト・タイプ。
CFG_LEVEL	VARCHAR(1)	構成レベル。
DBNAME	VARCHAR(128)	データベース名。

表 286. HEALTH_GET_ALERT_CFG 表関数によって戻される情報 (続き)

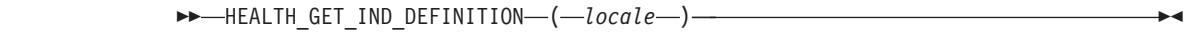
列名	データ・タイプ	説明
OBJECTNAME	VARCHAR(512)	オブジェクト名。
ID	BIGINT	ヘルス・インディケーター ID。
IS_DEFAULT	SMALLINT	設定がデフォルトであるかどうか。デフォルトである場合は 1、デフォルトでない場合は 0、適用外である場合は NULL。
WARNING_THRESHOLD	BIGINT	警告しきい値。適用外の場合は NULL 値。
ALARM_THRESHOLD	BIGINT	アラームしきい値。適用外の場合は NULL 値。
SENSITIVITY	BIGINT	ヘルス・インディケーターの感度。
EVALUATE	SMALLINT	このヘルス・インディケーターが評価されている場合は 1、評価されていない場合は 0。
ACTION_ENABLED	SMALLINT	アラートの発生時にアクションの実行が可能な場合は 1、アラートの発生時にアクションの実行が可能でない場合は 0。

HEALTH_GET_IND_DEFINITION

ヘルス・インディケーター定義を戻します。

重要: バージョン 9.7 でヘルス・モニターが非推奨になったため、この表関数は推奨されておらず、将来のリリースで除去される可能性があります。 詳しくは、「*DB2 バージョン 9.7 の新機能*」のトピック『ヘルス・モニターが推奨されなくなった』を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

locale
戻される翻訳可能出力のロケールを示す、タイプ VARCHAR(33) の入力引数。入力ロケールをデータベース・サーバーがサポートしていない場合、SQL 警告メッセージが発行され、デフォルト言語 (英語) が使用されます。入力ロケールが指定されていない場合、つまりその値が NULL (または空ストリング) の場合、デフォルト言語が使用されます。

許可

HEALTH_GET_IND_DEFINITION 表関数に対する EXECUTE 特権。

例

例 1: ヘルス・インディケーター db.db_op_status のタイプと簡略説明をフランス語で検索します。

```
SELECT TYPE, SHORT_DESCRIPTION
   FROM TABLE(SYSPROC.HEALTH_GET_IND_DEFINITION('fr_FR'))
   AS IND_DEFINITION WHERE NAME = 'db.db_op_status'
```

以下はこの照会の出力例です。

```
TYPE            SHORT_DESCRIPTION
--------------- ----------------------------------------...---------
STATE           Etat opérationnel de la base de données

  1 record(s) selected.
```

例 2: ヘルス・インディケーター ID 1001 の簡略説明を英語で検索します。

```
SELECT SHORT_DESCRIPTION FROM TABLE(SYSPROC.HEALTH_GET_IND_DEFINITION('en_US')
   AS IND_DEFINITION WHERE ID = 1001
```

以下はこの照会の出力例です。

```
SHORT_DESCRIPTION
----------------------------...-------------
Database Operational State
```

例 3: すべてのヘルス・インディケーター ID および名前を検索します。

```
SELECT ID, NAME FROM TABLE(HEALTH_GET_IND_DEFINITION('')) AS T
```

以下はこの照会の出力例です。

```
ID                  NAME
------------------- ----------------------------
                  1 db2.db2_op_status
                  2 db2.sort_privmem_util
                  4 db2.mon_heap_util
               1001 db.db_op_status
               1002 db.sort_shrmem_util
...
               2001 ts.ts_op_status
               2002 ts.ts_util
...
               3002 tsc.tscont_util
               1015 db.tb_reorg_req
...
```

戻される情報

表 287. HEALTH_GET_IND_DEFINITION 表関数によって戻される情報

列名	データ・タイプ	説明
ID	BIGINT	ヘルス・インディケーター ID。
NAME	VARCHAR(128)	ヘルス・インディケーター名。

表 287. HEALTH_GET_IND_DEFINITION 表関数によって戻される情報 (続き)

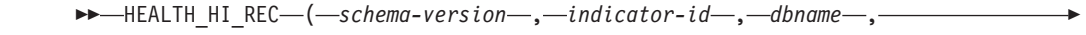
列名	データ・タイプ	説明
SHORT_DESCRIPTION	VARCHAR(1024)	ヘルス・インディケーター簡略説明。
LONG_DESCRIPTION	VARCHAR(32672)	ヘルス・インディケーター詳細説明。
TYPE	VARCHAR(16)	ヘルス・インディケーター・タイプ。可能な値は以下のとおりです。 • 'THRESHOLD_UPPER': 上限のしきい値ベースのヘルス・インディケーター。 • 'THRESHOLD_LOWER': 下限のしきい値ベースのヘルス・インディケーター。 • 'STATE': 状態ベースのヘルス・インディケーター。 • 'COLLECTION_STATE': コレクション状態ベースのヘルス・インディケーター。
UNIT	VARCHAR(1024)	ヘルス・インディケーターの値およびしきい値の単位。または適用外の場合は NULL。
CATEGORY	VARCHAR(1024)	ヘルス・インディケーターのカテゴリー。
FORMULA	VARCHAR(512)	ヘルス・インディケーターの式。
REFRESH_INTERVAL	BIGINT	ヘルス・インディケーターの評価の間隔 (秒単位)。

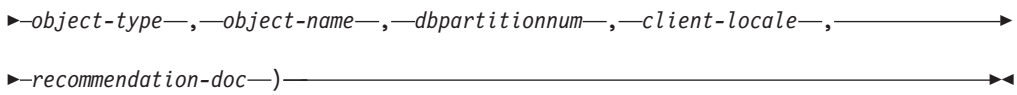
HEALTH_HI_REC

特定の DB2 オブジェクト上のアラート状態にあるヘルス・インディケーターに関連した一連の推奨事項を取り出します。推奨事項は、アラート状態を解決するために実行できるアクション (例えば、実行可能なスクリプト) についての情報を収めた XML 文書の中に戻されます。

重要: バージョン 9.7 でヘルス・モニターが非推奨になっているため、このプロシージャーは非推奨になり、今後のリリースで除去される可能性があります。詳しくは、「*バージョン 9.7 の新機能*」のトピック『ヘルス・モニターが推奨されなくなった』を参照してください。

構文





スキーマは SYSPROC です。

このプロシージャーで戻されるスクリプトがある場合、ヘルス・インディケーターがアラート状態になったインスタンスから呼び出す必要があります。

識別されたオブジェクト上の指定ヘルス・インディケーターがアラート状態ではない場合、エラーが戻されます (SQLSTATE 5U0ZZ)。

プロシージャー・パラメーター

schema-version
XML 文書を表すのに使用されるスキーマのバージョン番号を指定する、タイプ INTEGER の入力引数。推奨事項文書には、該当スキーマ・バージョンに定義されたエレメントと属性だけが含まれています。有効なスキーマ・バージョンは、sqllib ディレクトリーの include サブディレクトリーにある、 db2ApiDf.h で定義されます。

indicator-id
推奨事項が要求されているヘルス・インディケーターの数値 ID を指定する、タイプ INTEGER の入力引数。有効なヘルス・インディケーター ID は、sqllib ディレクトリーの include サブディレクトリーにある、 sqlmon.h で定義されます。

dbname
ヘルス・インディケーターがアラート状態となった対象のデータベースで、オブジェクト・タイプが DB2HEALTH_OBJTYPE_TS_CONTAINER、DB2HEALTH_OBJTYPE_TABLESPACE、または DB2HEALTH_OBJTYPE_DATABASE のいずれかであるとき、そのデータベースの別名を指定する、タイプ VARCHAR(255) の入力引数。それ以外の場合、NULL を指定します。

object-type
ヘルス・インディケーターがアラート状態になったオブジェクトのタイプを指定する、タイプ INTEGER の入力引数。有効なオブジェクト・タイプは、sqllib ディレクトリーの include サブディレクトリーにある、 sqlmon.h で定義されます。

object-name
オブジェクト・タイプが DB2HEALTH_OBJTYPE_TABLESPACE か DB2HEALTH_OBJTYPE_TS_CONTAINER に設定されるときの、表スペースまたは表スペース・コンテナーの名前を指定する、タイプ VARCHAR(255) の入力引数。オブジェクト・タイプが DB2HEALTH_OBJTYPE_DATABASE または DB2HEALTH_OBJTYPE_DATABASE_MANAGER である場合は、NULL を指定してください。表スペース・コンテナーの場合、オブジェクト名は、*table_space_name.container_name* と指定されます。

dbpartitionnum
ヘルス・インディケーターがアラート状態になったデータベース・パーティション数を指定する INTEGER のタイプの入力引数。有効な値は、0 から 999、-1 (現在接続されているデータベース・パーティションを指定)、および -2 (すべて

のデータベース・パーティションを指定) です。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

client-locale
クライアントの言語 ID を指定する、タイプ VARCHAR(33) の入力引数。このパラメーターは、推奨事項が戻される言語を指定するときに使用します。値が指定されていない場合は、'En_US' (英語) が使用されます。指定したロケールのメッセージ・ファイルがサーバーで使用できない場合には、 'En_US' がデフォルトとして使用されるので注意してください。

recommendation-doc
DB2 Health Recommendation スキーマ定義 (sqllib ディレクトリーの misc サブディレクトリーにある、XML スキーマ DB2RecommendationSchema.xsd を参照) に従ってフォーマットされた推奨事項文書 (XML) を含む、タイプ BLOB(2M) の出力引数。メッセージがターゲット・インスタンスでの呼び出し元のロケールで使用できない場合、 XML 文書は UTF-8 でエンコードされ、その文書のテキストは呼び出し元のロケールまたは英語になります。

HEALTH_TBS_HI

データベースの表スペースのヘルス・スナップショットから、表スペースのヘルス・インディケーターの情報を戻します。

重要: バージョン 9.7 でヘルス・モニターが非推奨になったため、この表関数は推奨されておらず、将来のリリースで除去される可能性があります。 詳しくは、「*DB2 バージョン 9.7 の新機能*」のトピック『ヘルス・モニターが推奨されなくなった』を参照してください。

構文

スキーマは SYSPROC です。

表関数パラメーター

dbname
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(255) の入力引数。"Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

HEALTH_TBS_HI 表関数に対する EXECUTE 特権。

例

```
SELECT * FROM TABLE(HEALTH_TBS_HI('',-1)) AS T
```

以下はこの照会の出力例です。

```
SNAPSHOT_TIMESTAMP         TABLESPACE_NAME     HI_ID     HI_VALUE ...
-------------------------- ---------------...- -----...- -------- ...
2006-02-13-12.30.35.229196 SYSCATSPACE              2001        0 ...
2006-02-13-12.30.35.229196 SYSCATSPACE              2002       99 ...
2006-02-13-12.30.35.229196 SYSCATSPACE              2003        0 ...
2006-02-13-12.30.35.229196 SYSTOOLSPACE             2001        0 ...
2006-02-13-12.30.35.229196 SYSTOOLSPACE             2002       62 ...
2006-02-13-12.30.35.229196 SYSTOOLSPACE             2003        0 ...
2006-02-13-12.30.35.229196 SYSTOOLSTMPSPACE         2001        0 ...
2006-02-13-12.30.35.229196 TEMPSPACE1               2001        0 ...
2006-02-13-12.30.35.229196 USERSPACE1               2001        0 ...
2006-02-13-12.30.35.229196 USERSPACE1               2002      100 ...
2006-02-13-12.30.35.229196 USERSPACE1               2003        0 ...

  11 record(s) selected.
```

この照会からの出力 (続き)。

```
... HI_TIMESTAMP               HI_ALERT_STATE  HI_ALERT_STATE_DETAIL ...
... -------------------------- --------------- --------------------- ...
... 2006-02-13-12.26.26.158000               1 Normal                ...
... 2006-02-13-12.26.26.158000               4 Alarm                 ...
... 2006-02-13-12.26.26.158000               1 Normal                ...
... 2006-02-13-12.26.26.158000               1 Normal                ...
... 2006-02-13-12.26.26.158000               1 Normal                ...
... 2006-02-13-12.26.26.158000               1 Normal                ...
... 2006-02-13-12.26.26.158000               1 Normal                ...
... 2006-02-13-12.26.26.158000               1 Normal                ...
... 2006-02-13-12.26.26.158000               4 Alarm                 ...
... 2006-02-13-12.26.26.158000               1 Normal                ...
```

この照会からの出力 (続き)。

```
... HI_FORMULA                HI_ADDITIONAL_INFO
... ---------------------...- ------------------------------------------------
... 0                         -
... ((9376 / 9468) * 100)     The short term table space growth rate
                                from "02/13/2006 11:26:26.000158" to
                                "02/13/2006 12:26:26.000158" is "N/A"
                                bytes per second and the long term growth
                                rate from "02/12/2006 12:26:26.000158"
                                to "02/13/2006 12:26:26.000158" is "N/A"
                                bytes per second. Time to fullness is
                                projected to be "N/A" and "N/A"
                                respectively. The table space is defined
                                with automatic storage set to "YES" and
                                automatic resize enabled set to "YES".
... 0                         The table space is defined with automatic
                                storage set to "YES" and automatic resize
                                enabled set to "YES". The following are
                                the automatic resize settings: increase
                                size (bytes) "-1", increase size (percent)
                                "N/A", maximum size (bytes) "-1". The
```

```
                              current table space size (bytes) is
                              "38797312".
... 0                         -
... ((156 / 252) * 100)       The short term table space growth rate
                              from "02/13/2006 11:26:26.000158" to
                              "02/13/2006 12:26:26.000158" is "N/A"
                              bytes per second and the long term growth
                              rate from "02/12/2006 12:26:26.000158"
                              to "02/13/2006 12:26:26.000158" is "N/A"
                              bytes per second. Time to fullness is
                              projected to be "N/A" and "N/A"
                              respectively. The table space is defined
                              with automatic storage set to "YES" and
                              automatic resize enabled set to "YES".
... 0                         The table space is defined with automatic
                              storage set to "YES" and automatic resize
                              enabled set to "YES". The following are
                              the automatic resize settings: increase
                              size (bytes) "-1", increase size (percent)
                              "N/A", maximum size (bytes) "-1". The
                              current table space size (bytes) is
                              "1048576".
... 0                         -
... 0                         -
... 0                         -
... ((1504 / 1504) * 100)     The short term table space growth rate from
                              "02/13/2006 11:26:26.000158" to
                              "02/13/2006 12:26:26.000158" is "N/A"
                              bytes per second and the long term growth
                              rate from "02/12/2006 12:26:26.000158" to
                              "02/13/2006 12:26:26.000158" is "N/A" bytes
                              per second. Time to fullness is projected
                              to be "N/A" and "N/A" respectively. The
                              table space is defined with automatic storage
                              set to "YES" and automatic resize enabled
                              set to "YES".
... 0                         The table space is defined with automatic
                              storage set to "YES" and automatic resize
                              enabled set to "YES". The following are
                              the automatic resize settings: increase
                              size (bytes) "-1", increase size (percent)
                              "N/A", maximum size (bytes) "-1". The
                              current table space size (bytes) is
                              "6291456".
```

戻される情報

表 288. HEALTH_TBS_HI 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
TABLESPACE_NAME	VARCHAR(128)	**tablespace_name** - 表スペース名
HI_ID	BIGINT	スナップショット・データ・ストリーム内のヘルス・インディケーターを固有に識別する番号。
HI_VALUE	SMALLINT	ヘルス・インディケーターの値。

表 288. HEALTH_TBS_HI 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
HI_TIMESTAMP	TIMESTAMP	アラートが生成された日時。
HI_ALERT_STATE	BIGINT	アラートの重大度。
HI_ALERT_STATE_DETAIL	VARCHAR(20)	HI_ALERT_STATE 列のテキスト記述。
HI_FORMULA	VARCHAR(2048)	ヘルス・インディケーターを計算するのに使用する公式。
HI_ADDITIONAL_INFO	VARCHAR(4096)	ヘルス・インディケーターに関する追加情報。

HEALTH_TBS_HI_HIS

HEALTH_TBS_HI_HIS 表関数は、データベースのヘルス・スナップショットから表スペースのヘルス・インディケーターの履歴情報を戻します。

重要: バージョン 9.7 でヘルス・モニターが非推奨になったため、この表関数は推奨されておらず、将来のリリースで除去される可能性があります。 詳しくは、「*DB2 バージョン 9.7 の新機能*」のトピック『ヘルス・モニターが推奨されなくなった』を参照してください。

構文

スキーマは SYSPROC です。

表関数パラメーター

dbname
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(255) の入力引数。"Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

HEALTH_TBS_HI_HIS 表関数に対する EXECUTE 特権。

例

```
SELECT * FROM TABLE(HEALTH_TBS_HI_HIS('',-1)) AS T
```

以下はこの照会の出力例です。

```
SNAPSHOT_TIMESTAMP         TABLESPACE_NAME     HI_ID     ...
-------------------------- --------------...- -----...- ...
2006-02-13-12.30.37.181478 SYSCATSPACE              2001 ...
2006-02-13-12.30.37.181478 SYSCATSPACE              2001 ...
2006-02-13-12.30.37.181478 SYSCATSPACE              2002 ...
2006-02-13-12.30.37.181478 SYSCATSPACE              2002 ...
2006-02-13-12.30.37.181478 SYSCATSPACE              2003 ...
2006-02-13-12.30.37.181478 SYSCATSPACE              2003 ...
2006-02-13-12.30.37.181478 SYSTOOLSPACE             2001 ...
2006-02-13-12.30.37.181478 SYSTOOLSPACE             2001 ...
2006-02-13-12.30.37.181478 SYSTOOLSPACE             2002 ...
2006-02-13-12.30.37.181478 SYSTOOLSPACE             2002 ...
2006-02-13-12.30.37.181478 SYSTOOLSPACE             2003 ...
2006-02-13-12.30.37.181478 SYSTOOLSPACE             2003 ...
2006-02-13-12.30.37.181478 SYSTOOLSTMPSPACE         2001 ...
2006-02-13-12.30.37.181478 SYSTOOLSTMPSPACE         2001 ...
2006-02-13-12.30.37.181478 TEMPSPACE1               2001 ...
2006-02-13-12.30.37.181478 TEMPSPACE1               2001 ...
2006-02-13-12.30.37.181478 USERSPACE1               2001 ...
2006-02-13-12.30.37.181478 USERSPACE1               2001 ...
2006-02-13-12.30.37.181478 USERSPACE1               2002 ...
2006-02-13-12.30.37.181478 USERSPACE1               2002 ...
2006-02-13-12.30.37.181478 USERSPACE1               2003 ...
2006-02-13-12.30.37.181478 USERSPACE1               2003 ...

  22 record(s) selected.
```

この照会からの出力 (続き)。

```
... HI_TIMESTAMP              HI_VALUE HI_ALERT_STATE HI_ALERT_STATE_DETAIL ...
... -------------------------- -------- -------------- --------------------- ...
... 2006-02-13-12.16.25.911000        0              1 Normal                ...
... 2006-02-13-12.06.26.168000        0              1 Normal                ...
... 2006-02-13-12.16.25.911000       99              4 Alarm                 ...
... 2006-02-13-12.06.26.168000       99              4 Alarm                 ...
... 2006-02-13-12.16.25.911000        0              1 Normal                ...
... 2006-02-13-12.06.26.168000        0              1 Normal                ...
... 2006-02-13-12.16.25.911000        0              1 Normal                ...
... 2006-02-13-12.06.26.168000        0              1 Normal                ...
... 2006-02-13-12.16.25.911000       62              1 Normal                ...
... 2006-02-13-12.06.26.168000       62              1 Normal                ...
... 2006-02-13-12.16.25.911000        0              1 Normal                ...
... 2006-02-13-12.06.26.168000        0              1 Normal                ...
... 2006-02-13-12.16.25.911000        0              1 Normal                ...
... 2006-02-13-12.06.26.168000        0              1 Normal                ...
... 2006-02-13-12.16.25.911000        0              1 Normal                ...
... 2006-02-13-12.06.26.168000        0              1 Normal                ...
... 2006-02-13-12.16.25.911000        0              1 Normal                ...
... 2006-02-13-12.06.26.168000        0              1 Normal                ...
... 2006-02-13-12.16.25.911000      100              4 Alarm                 ...
... 2006-02-13-12.06.26.168000      100              4 Alarm                 ...
... 2006-02-13-12.16.25.911000        0              1 Normal                ...
... 2006-02-13-12.06.26.168000        0              1 Normal                ...
```

この照会からの出力 (続き)。

```
... HI_FORMULA              HI_ADDITIONAL_INFO
... --------------------...- -------------------...---
... 0                       -
... 0                       -
... ((9376 / 9468) * 100)   The short term table space growth rate from
```

	"02/13/2006 11:16:25.000911" to "02/13/2006 12:16:25.000911" is "N/A" bytes per second and the long term growth rate from "02/12/2006 12:16:25.000911" to "02/13/2006 12:16:25.000911" is "N/A" bytes per second. Time to fullness is projected to be "N/A" and "N/A" respectively. The table space is defined with automatic storage set to "YES" and automatic resize enabled set to "YES".
... ((9376 / 9468) * 100)	The short term table space growth rate from "02/13/2006 11:06:26.000168" to "02/13/2006 12:06:26.000168" is "N/A" bytes per second and the long term growth rate from "02/12/2006 12:06:26.000168" to "02/13/2006 12:06:26.000168" is "N/A" bytes per second. Time to fullness is projected to be "N/A" and "N/A" respectively. The table space is defined with automatic storage set to "YES" and automatic resize enabled set to "YES".
... 0	The table space is defined with automatic storage set to "YES" and automatic resize enabled set to "YES". The following are the automatic resize settings: increase size (bytes) "-1", increase size (percent) "N/A", maximum size (bytes) "-1". The current table space size (bytes) is "38797312".
... 0	The table space is defined with automatic storage set to "YES" and automatic resize enabled set to "YES". The following are the automatic resize settings: increase size (bytes) "-1", increase size (percent) "N/A", maximum size (bytes) "-1". The current table space size (bytes) is "38797312".
... 0	-
... 0	-
... ((156 / 252) * 100)	The short term table space growth rate from "02/13/2006 11:16:25.000911" to "02/13/2006 12:16:25.000911" is "N/A" bytes per second and the long term growth rate from "02/12/2006 12:16:25.000911" to "02/13/2006 12:16:25.000911" is "N/A" bytes per second. Time to fullness is projected to be "N/A" and "N/A" respectively. The table space is defined with automatic storage set to "YES" and automatic resize enabled set to "YES".
... ((156 / 252) * 100)	The short term table space growth rate from "02/13/2006 11:06:26.000168" to "02/13/2006 12:06:26.000168" is "N/A" bytes per second and the long term growth rate from "02/12/2006 12:06:26.000168" to "02/13/2006 12:06:26.000168" is "N/A" bytes per second. Time to fullness is projected to be "N/A" and "N/A" respectively. The table space is defined with automatic storage set to "YES" and automatic resize enabled set to "YES".
... 0	The table space is defined with automatic storage set to "YES" and automatic resize enabled set to "YES". The following are the automatic resize settings: increase size (bytes) "-1", increase size (percent) "N/A", maximum size (bytes) "-1". The

```
                              current table space size (bytes) is
                              "1048576".
... 0                         The table space is defined with automatic
                              storage set to "YES" and automatic resize
                              enabled set to "YES". The following are
                              the automatic resize settings: increase
                              size (bytes) "-1", increase size (percent)
                              "N/A", maximum size (bytes) "-1". The
                              current table space size (bytes) is
                              "1048576".
... 0                         -
... 0                         -
... 0                         -
... 0                         -
... 0                         -
... 0                         -
... ((1504 / 1504) * 100)     The short term table space growth rate
                              from "02/13/2006 11:16:25.000911" to
                              "02/13/2006 12:16:25.000911" is "N/A"
                              bytes per second and the long term growth
                              rate from "02/12/2006 12:16:25.000911"
                              to "02/13/2006 12:16:25.000911" is "N/A"
                              bytes per second. Time to fullness is
                              projected to be "N/A" and "N/A"
                              respectively. The table space is defined
                              with automatic storage set to "YES" and
                              automatic resize enabled set to "YES".
... ((1504 / 1504) * 100)     The short term table space growth rate
                              from "02/13/2006 11:06:26.000168" to
                              "02/13/2006 12:06:26.000168" is "N/A"
                              bytes per second and the long term growth
                              rate from "02/12/2006 12:06:26.000168"
                              to "02/13/2006 12:06:26.000168" is "N/A"
                              bytes per second. Time to fullness is
                              projected to be "N/A" and "N/A"
                              respectively. The table space is defined
                              with automatic storage set to "YES" and
                              automatic resize enabled set to "YES".
... 0                         The table space is defined with automatic
                              storage set to "YES" and automatic
                              resize enabled set to "YES". The
                              following are the automatic resize
                              settings: increase size (bytes) "-1",
                              increase size (percent) "N/A", maximum
                              size (bytes) "-1". The current table
                              space size (bytes) is "6291456".
... 0                         The table space is defined with automatic
                              storage set to "YES" and automatic
                              resize enabled set to "YES". The
                              following are the automatic resize
                              settings: increase size (bytes) "-1",
                              increase size (percent) "N/A", maximum
                              size (bytes) "-1". The current table
                              space size (bytes) is "6291456".
```

戻される情報

表 289. HEALTH_TBS_HI_HIS 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。

表 289. HEALTH_TBS_HI_HIS 表関数によって戻される情報 (続き)

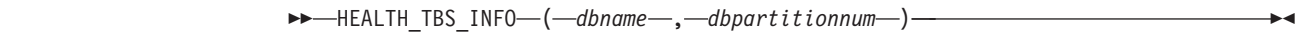
列名	データ・タイプ	説明または対応するモニター・エレメント
TABLESPACE_NAME	VARCHAR(128)	tablespace_name - 表スペース名
HI_ID	BIGINT	スナップショット・データ・ストリーム内のヘルス・インディケーターを固有に識別する番号。
HI_TIMESTAMP	TIMESTAMP	アラートが生成された日時。
HI_VALUE	SMALLINT	ヘルス・インディケーターの値。
HI_ALERT_STATE	BIGINT	アラートの重大度。
HI_ALERT_STATE_DETAIL	VARCHAR(20)	HI_ALERT_STATE 列のテキスト記述。
HI_FORMULA	VARCHAR(2048)	ヘルス・インディケーターを計算するのに使用する公式。
HI_ADDITIONAL_INFO	VARCHAR(4096)	ヘルス・インディケーターに関する追加情報。

HEALTH_TBS_INFO

データベースのヘルス・スナップショットから表スペースの情報を戻します。

重要: バージョン 9.7 でヘルス・モニターが非推奨になったため、この表関数は推奨されておらず、将来のリリースで除去される可能性があります。 詳しくは、「*DB2 バージョン 9.7 の新機能*」のトピック『ヘルス・モニターが推奨されなくなった』を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(255) の入力引数。"Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。アクティブなデー

タベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

HEALTH_TBS_INFO 表関数に対する EXECUTE 特権。

例

```
SELECT * FROM TABLE(HEALTH_TBS_INFO('',-1)) AS T
```

以下はこの照会の出力例です。

```
SNAPSHOT_TIMESTAMP         TABLESPACE_NAME      ...
-------------------------- ----------------...- ...
2006-02-13-12.30.35.027383 SYSCATSPACE          ...
2006-02-13-12.30.35.027383 SYSTOOLSPACE         ...
2006-02-13-12.30.35.027383 SYSTOOLSTMPSPACE     ...
2006-02-13-12.30.35.027383 TEMPSPACE1           ...
2006-02-13-12.30.35.027383 USERSPACE1           ...

  5 record(s) selected.
```

この照会からの出力 (続き)。

```
... ROLLED_UP_ALERT_STATE ROLLED_UP_ALERT_STATE_DETAIL
... --------------------- ----------------------------
...                     4 Alarm
...                     1 Normal
...                     1 Normal
...                     1 Normal
...                     4 Alarm
```

戻される情報

表 290. HEALTH_TBS_INFO 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
TABLESPACE_NAME	VARCHAR(128)	**tablespace_name** - 表スペース名
ROLLED_UP_ALERT_STATE	BIGINT	このスナップショットでキャプチャーされた最も重大なアラート状態。
ROLLED_UP_ALERT_STATE_DETAIL	VARCHAR(20)	ROLLED_UP_ALERT_STATE 列のテキスト記述。

SNAP_GET_APPL 表関数 - appl 論理データ・グループのスナップショット情報の検索

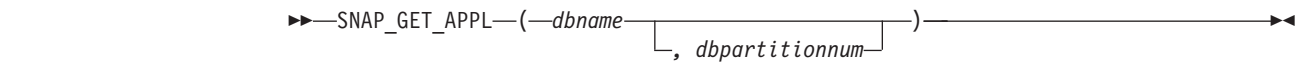
注: この表関数は使用すべきではなく、 652 ページの『SNAPAPPL 管理ビューおよび SNAP_GET_APPL_V95 表関数 - appl 論理データ・グループのスナップショット情報の検索』 に置き換えられました。

SNAP_GET_APPL 表関数は、アプリケーション・スナップショットから、特に appl 論理データ・グループのアプリケーション情報を戻します。

SNAP_GET_APPL 表関数を SNAP_GET_AGENT、SNAP_GET_AGENT_MEMORY_POOL、 SNAP_GET_APPL_INFO、SNAP_GET_STMT、および SNAP_GET_SUBSECTION 表関数とともに使用すると、GET SNAPSHOT FOR ALL APPLICATIONS CLP コマンドに相当する情報が提供されます。ただし、すべてのデータベース・パーティションからデータを取得します。

戻される可能性のある情報の完全なリストは、 1188 ページの表 291を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、空ストリングを指定します。現在接続されているデータベースと同じインスタンス内のすべてのデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプを持つファイルが存在しない場合、SNAP_GET_APPL 表関数は、現在接続中のデータベースおよびデータベース・パーティション番号のスナップショットを取得します。

許可

- SYSMON 権限
- SNAP_GET_APPL 表関数に対する EXECUTE 特権。

例

すべてのアクティブ・データベースの各アプリケーションについて読み取りおよび書き込みが行われた行の詳細を取得します。

```
SELECT SUBSTR(DB_NAME,1,8) AS DB_NAME, AGENT_ID, ROWS_READ, ROWS_WRITTEN
   FROM TABLE (SNAP_GET_APPL(CAST(NULL AS VARCHAR(128)),-1)) AS T
```

以下はこの照会の出力例です。

```
DB_NAME  AGENT_ID       ROWS_READ      ROWS_WRITTEN
-------- --------...--- ---------...-- ------------...-
WSDB               679              0                 0
WSDB               461              3                 0
WSDB               460              4                 0
TEST               680              4                 0
TEST               455              6                 0
TEST               454              0                 0
TEST               453             50                 0
```

戻される情報

表 291. SNAP_GET_APPL 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
DB_NAME	VARCHAR(128)	db_name - データベース名
AGENT_ID	BIGINT	agent_id - アプリケーション・ハンドル (エージェント ID)
UOW_LOG_SPACE_USED	BIGINT	uow_log_space_used - 作業単位ログ・スペース
ROWS_READ	BIGINT	rows_read - 読み取り行数
ROWS_WRITTEN	BIGINT	rows_written - 書き込み行数
INACT_STMTHIST_SZ	BIGINT	stmt_history_list_size - ステートメント履歴リストのサイズ
POOL_DATA_L_READS	BIGINT	pool_data_l_reads - バッファー・プール・データの論理読み取り
POOL_DATA_P_READS	BIGINT	pool_data_p_reads - バッファー・プール・データの物理読み取り

表 291. SNAP_GET_APPL 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
POOL_DATA_WRITES	BIGINT	pool_data_writes - バッファー・プールへのデータの書き込み
POOL_INDEX_L_READS	BIGINT	pool_index_l_reads - バッファー・プール索引の論理読み取り
POOL_INDEX_P_READS	BIGINT	pool_index_p_reads - バッファー・プール索引の物理読み取り
POOL_INDEX_WRITES	BIGINT	pool_index_writes - バッファー・プール索引の書き込み
POOL_TEMP_DATA_L_READS	BIGINT	pool_temp_data_l_reads - バッファー・プール一時データの論理読み取り
POOL_TEMP_DATA_P_READS	BIGINT	pool_temp_data_p_reads - バッファー・プール一時データの物理読み取り
POOL_TEMP_INDEX_L_READS	BIGINT	pool_temp_index_l_reads - バッファー・プール一時索引の論理読み取り
POOL_TEMP_INDEX_P_READS	BIGINT	pool_temp_index_p_reads - バッファー・プール一時索引の物理読み取り
POOL_TEMP_XDA_L_READS	BIGINT	pool_temp_xda_l_reads - バッファー・プール一時 XDA データの論理読み取り
POOL_TEMP_XDA_P_READS	BIGINT	pool_temp_xda_p_reads - バッファー・プール一時 XDA データの物理読み取り : モニター・エレメント
POOL_XDA_L_READS	BIGINT	pool_xda_l_reads - バッファー・プール XDA データの論理読み取り
POOL_XDA_P_READS	BIGINT	pool_xda_p_reads - バッファー・プール XDA データの物理読み取り
POOL_XDA_WRITES	BIGINT	pool_xda_writes - バッファー・プール XDA データの書き込み
POOL_READ_TIME	BIGINT	pool_read_time - バッファー・プール物理読み取り時間の合計
POOL_WRITE_TIME	BIGINT	pool_write_time - バッファー・プール物理書き込み時間の合計
DIRECT_READS	BIGINT	direct_reads - データベースからの直接読み取り
DIRECT_WRITES	BIGINT	direct_writes - データベースへの直接書き込み
DIRECT_READ_REQS	BIGINT	direct_read_reqs - 直接読み取り要求
DIRECT_WRITE_REQS	BIGINT	direct_write_reqs - 直接書き込み要求

表 291. SNAP_GET_APPL 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
DIRECT_READ_TIME	BIGINT	direct_read_time - 直接読み取り時間
DIRECT_WRITE_TIME	BIGINT	direct_write_time - 直接書き込み時間
UNREAD_PREFETCH_PAGES	BIGINT	unread_prefetch_pages - 読み取り不能プリフェッチ・ページ
LOCKS_HELD	BIGINT	locks_held - ロック保持数
LOCK_WAITS	BIGINT	lock_waits - ロック待機数
LOCK_WAIT_TIME	BIGINT	lock_wait_time - ロック待機中の時間
LOCK_ESCALS	BIGINT	lock_escals - ロック・エスカレーション数
X_LOCK_ESCALS	BIGINT	x_lock_escals - 排他ロック・エスカレーション数
DEADLOCKS	BIGINT	deadlocks - デッドロック検出数
TOTAL_SORTS	BIGINT	total_sorts - ソート合計
TOTAL_SORT_TIME	BIGINT	total_sort_time - ソート時間合計
SORT_OVERFLOWS	BIGINT	sort_overflows - ソート・オーバーフロー
COMMIT_SQL_STMTS	BIGINT	commit_sql_stmts - 試行されたコミット・ステートメント
ROLLBACK_SQL_STMTS	BIGINT	rollback_sql_stmts - 試行されたロールバック・ステートメント
DYNAMIC_SQL_STMTS	BIGINT	dynamic_sql_stmts - 試行された動的 SQL ステートメント
STATIC_SQL_STMTS	BIGINT	static_sql_stmts - 試行された静的 SQL ステートメント
FAILED_SQL_STMTS	BIGINT	failed_sql_stmts - 失敗したステートメント操作
SELECT_SQL_STMTS	BIGINT	select_sql_stmts - 実行された選択 SQL ステートメント
DDL_SQL_STMTS	BIGINT	ddl_sql_stmts - データ定義言語 (DDL) SQL ステートメント
UID_SQL_STMTS	BIGINT	uid_sql_stmts - 実行された更新/挿入/削除 SQL ステートメント
INT_AUTO_REBINDS	BIGINT	int_auto_rebinds - 内部自動再バインド
INT_ROWS_DELETED	BIGINT	int_rows_deleted - 削除された内部行数
INT_ROWS_UPDATED	BIGINT	int_rows_updated - 更新された内部行数
INT_COMMITS	BIGINT	int_commits - 内部コミット数

表 291. SNAP_GET_APPL 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
INT_ROLLBACKS	BIGINT	int_rollbacks - 内部ロールバック数
INT_DEADLOCK_ROLLBACKS	BIGINT	int_deadlock_rollbacks - デッドロックによる内部ロールバック数
ROWS_DELETED	BIGINT	rows_deleted - 削除行数
ROWS_INSERTED	BIGINT	rows_inserted - 挿入行数
ROWS_UPDATED	BIGINT	rows_updated - 更新行数
ROWS_SELECTED	BIGINT	rows_selected - 選択行数
BINDS_PRECOMPILES	BIGINT	binds_precompiles - 試行されたバインド/プリコンパイル
OPEN_REM_CURS	BIGINT	open_rem_curs - 開かれているリモート・カーソル
OPEN_REM_CURS_BLK	BIGINT	open_rem_curs_blk - 開かれているリモート・ブロック・カーソル
REJ_CURS_BLK	BIGINT	rej_curs_blk - リジェクトされたブロック・カーソル要求
ACC_CURS_BLK	BIGINT	acc_curs_blk - 受け入れられたブロック・カーソル要求
SQL_REQS_SINCE_COMMIT	BIGINT	sql_reqs_since_commit - 最終コミット後の SQL 要求数
LOCK_TIMEOUTS	BIGINT	lock_timeouts - ロック・タイムアウト数
INT_ROWS_INSERTED	BIGINT	int_rows_inserted - 挿入された内部行数
OPEN_LOC_CURS	BIGINT	open_loc_curs - 開かれているローカル・カーソル
OPEN_LOC_CURS_BLK	BIGINT	open_loc_curs_blk - 開かれているローカル・ブロック・カーソル
PKG_CACHE_LOOKUPS	BIGINT	pkg_cache_lookups - パッケージ・キャッシュ参照
PKG_CACHE_INSERTS	BIGINT	pkg_cache_inserts - パッケージ・キャッシュ挿入
CAT_CACHE_LOOKUPS	BIGINT	cat_cache_lookups - カタログ・キャッシュ参照数
CAT_CACHE_INSERTS	BIGINT	cat_cache_inserts - カタログ・キャッシュ挿入数
CAT_CACHE_OVERFLOWS	BIGINT	cat_cache_overflows - カタログ・キャッシュ・オーバーフロー数
NUM_AGENTS	BIGINT	num_agents - ステートメントで作動しているエージェントの数
AGENTS_STOLEN	BIGINT	agents_stolen - スチールされたエージェント

表 291. SNAP_GET_APPL 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
ASSOCIATED_AGENTS_TOP	BIGINT	associated_agents_top - 関連エージェント最大数
APPL_PRIORITY	BIGINT	appl_priority - アプリケーション・エージェント優先順位
APPL_PRIORITY_TYPE	VARCHAR(16)	appl_priority_type - アプリケーション優先順位タイプ。このインターフェースは、sqlmon.h 内の定義に基づいてテキスト ID を戻します。それは、次のうちの 1 つです。 • DYNAMIC_PRIORITY • FIXED_PRIORITY
PREFETCH_WAIT_TIME	BIGINT	prefetch_wait_time - プリフェッチ待ち時間
APPL_SECTION_LOOKUPS	BIGINT	appl_section_lookups - セクションの参照回数
APPL_SECTION_INSERTS	BIGINT	appl_section_inserts - セクション挿入数
LOCKS_WAITING	BIGINT	locks_waiting - ロックで待機中の現行エージェント
TOTAL_HASH_JOINS	BIGINT	total_hash_joins - ハッシュ結合の合計
TOTAL_HASH_LOOPS	BIGINT	total_hash_loops - ハッシュ・ループの合計
HASH_JOIN_OVERFLOWS	BIGINT	hash_join_overflows - ハッシュ結合のオーバーフロー
HASH_JOIN_SMALL_OVERFLOWS	BIGINT	hash_join_small_overflows - ハッシュ結合の短精度オーバーフロー
APPL_IDLE_TIME	BIGINT	appl_idle_time - アプリケーション・アイドル時間
UOW_LOCK_WAIT_TIME	BIGINT	uow_lock_wait_time - ロック待機中の作業単位の合計時間

表 291. SNAP_GET_APPL 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
UOW_COMP_STATUS	VARCHAR(14)	uow_comp_status - 作業単位完了状況。このインターフェースは、sqlmon.h 内の定義に基づいてテキスト ID を戻します。それは、次のうちの 1 つです。 • APPL_END • UOWABEND • UOWCOMMIT • UOWDEADLOCK • UOWLOCKTIMEOUT • UOWROLLBACK • UOWUNKNOWN
AGENT_USR_CPU_TIME_S	BIGINT	agent_usr_cpu_time - エージェントが使用したユーザー CPU 時間 (秒単位)*
AGENT_USR_CPU_TIME_MS	BIGINT	agent_usr_cpu_time - エージェントが使用したユーザー CPU 時間 (小数部、マイクロ秒単位)*
AGENT_SYS_CPU_TIME_S	BIGINT	agent_sys_cpu_time - エージェントが使用したシステム CPU 時間 (秒単位)*
AGENT_SYS_CPU_TIME_MS	BIGINT	agent_sys_cpu_time - エージェントが使用したシステム CPU 時間 (小数部、マイクロ秒単位)*
APPL_CON_TIME	TIMESTAMP	appl_con_time - 接続要求開始タイム・スタンプ
CONN_COMPLETE_TIME	TIMESTAMP	conn_complete_time - 接続要求完了タイム・スタンプ
LAST_RESET	TIMESTAMP	last_reset - 最後のリセット・タイム・スタンプ
UOW_START_TIME	TIMESTAMP	uow_start_time - 作業単位開始タイム・スタンプ
UOW_STOP_TIME	TIMESTAMP	uow_stop_time - 作業単位停止タイム・スタンプ
PREV_UOW_STOP_TIME	TIMESTAMP	prev_uow_stop_time - 直前の作業単位完了タイム・スタンプ
UOW_ELAPSED_TIME_S	BIGINT	uow_elapsed_time - 最新の作業単位の経過時間 (秒単位)*
UOW_ELAPSED_TIME_MS	BIGINT	uow_elapsed_time - 最新の作業単位の経過時間 (小数部、マイクロ秒単位)*
ELAPSED_EXEC_TIME_S	BIGINT	elapsed_exec_time - ステートメント実行経過時間 (秒単位)*

表 291. SNAP_GET_APPL 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
ELAPSED_EXEC_TIME_MS	BIGINT	elapsed_exec_time - ステートメント実行経過時間 (小数部、マイクロ秒単位)*
INBOUND_COMM_ADDRESS	VARCHAR(32)	inbound_comm_address - インバウンド通信アドレス
LOCK_TIMEOUT_VAL	BIGINT	lock_timeout_val - ロック・タイムアウト (秒)
PRIV_WORKSPACE_NUM_OVERFLOWS	BIGINT	priv_workspace_num_overflows - 専用ワークスペースのオーバーフロー回数
PRIV_WORKSPACE_SECTION_INSERTS	BIGINT	priv_workspace_section_inserts - 専用ワークスペース・セクション挿入
PRIV_WORKSPACE_SECTION_LOOKUPS	BIGINT	priv_workspace_section_lookups - 専用ワークスペース・セクションの参照
PRIV_WORKSPACE_SIZE_TOP	BIGINT	priv_workspace_size_top - 専用ワークスペースの最大サイズ
SHR_WORKSPACE_NUM_OVERFLOWS	BIGINT	shr_workspace_num_overflows - 共有ワークスペースのオーバーフロー回数
SHR_WORKSPACE_SECTION_INSERTS	BIGINT	shr_workspace_section_inserts - 共有ワークスペース・セクション挿入数
SHR_WORKSPACE_SECTION_LOOKUPS	BIGINT	shr_workspace_section_lookups - 共有ワークスペース・セクションの参照回数
SHR_WORKSPACE_SIZE_TOP	BIGINT	shr_workspace_size_top - 最大共有ワークスペース・サイズ
DBPARTITIONNUM	SMALLINT	行のデータが検索されたデータベース・パーティション。
CAT_CACHE_SIZE_TOP	BIGINT	cat_cache_size_top - カタログ・キャッシュ最高水準点
* この列の元になるモニター・エレメントの合計消費時間を計算するには、このモニター・エレメントに関する _S で終わっている列で報告されている整数秒と、このモニター・エレメントに関する _MS で終わっている列に報告されている小数秒を、次の式を使用して加算しなければなりません: (*monitor-element-name*_S × 1,000,000 + *monitor-element-name*_MS) ÷ 1,000,000. 例えば、(ELAPSED_EXEC_TIME_S × 1,000,000 + ELAPSED_EXEC_TIME_MS) ÷ 1,000,000。		

SNAP_GET_APPL_INFO 表関数 - appl_info 論理データ・グループのスナップショット情報の検索

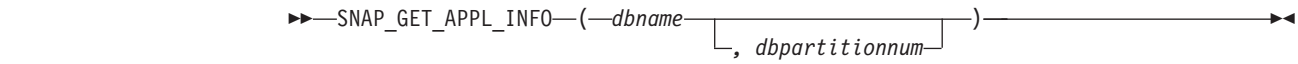
注: この表関数は使用すべきではなく、 644 ページの『SNAPAPPL_INFO 管理ビューおよび SNAP_GET_APPL_INFO_V95 表関数 - appl_info 論理データ・グループのスナップショット情報の検索』 に置き換えられました。

SNAP_GET_APPL_INFO 表関数は、アプリケーション・スナップショットから、特に appl_info 論理データ・グループのアプリケーション情報を戻します。

SNAP_GET_APPL_INFO 表関数を SNAP_GET_AGENT、SNAP_GET_AGENT_MEMORY_POOL、 SNAP_GET_APPL、SNAP_GET_APPL_INFO、 SNAP_GET_STMT、および SNAP_GET_SUBSECTION 表関数とともに使用すると、GET SNAPSHOT FOR ALL APPLICATIONS CLP コマンドに相当する情報が提供されます。ただし、すべてのデータベース・パーティションからデータを取得します。

戻される可能性のある情報の完全なリストは、 1197 ページの表 292を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、空ストリングを指定します。現在接続されているデータベースと同じインスタンス内のすべてのデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプを持つファイルが存在しない場合、SNAP_GET_APPL_INFO 表関数は、現在接続中のデータベースおよびデータベース・パーティション番号のスナップショットを取得します。

許可

- SYSMON 権限
- SNAP_GET_APPL_INFO 表関数に対する EXECUTE 特権。

例

接続中のデータベース・パーティション上のすべてのアプリケーションの状況を取得します。

```
SELECT SUBSTR(DB_NAME,1,8) AS DB_NAME, AGENT_ID,
   SUBSTR(APPL_NAME,1,10) AS APPL_NAME, APPL_STATUS
   FROM TABLE(SNAP_GET_APPL_INFO(CAST(NULL AS VARCHAR(128)),-1)) AS T
```

以下はこの照会の出力例です。

```
DB_NAME  AGENT_ID             APPL_NAME  APPL_STATUS
-------- -------------------- ---------- ----------------------
TOOLSDB                    14 db2bp.exe  CONNECTED
SAMPLE                     15 db2bp.exe  UOWEXEC
SAMPLE                      8 javaw.exe  CONNECTED
SAMPLE                      7 db2bp.exe  UOWWAIT

  4 record(s) selected.
```

以下は、表関数の結果からの SELECT の実行時に入手できる内容について示しています。

```
SELECT SUBSTR(DB_NAME,1,8) AS DB_NAME, AUTHORITY_LVL
   FROM TABLE(SNAP_GET_APPL_INFO_V95(CAST(NULL AS VARCHAR(128)),-1)) AS T
```

以下はこの照会の出力例です。

```
DB_NAME  AUTHORITY_LVL
-------- -----------------------------------------------------....
TESTDB   SYSADM(GROUP) + DBADM(USER) + CREATETAB(USER, GROUP) +
            BINDADD(USER, GROUP) + CONNECT(USER, GROUP) +
            CREATE_NOT_FENC(USER) + IMPLICIT_SCHEMA(USER, GROUP) +
            LOAD(USER) + CREATE_EXT_RT(USER) + QUIESCE_CONN(USER)
TESTDB   SYSADM(GROUP) + DBADM(USER) + CREATETAB(USER, GROUP) +
            BINDADD(USER, GROUP) + CONNECT(USER, GROUP) +
            CREATE_NOT_FENC(USER) + IMPLICIT_SCHEMA(USER, GROUP) +
            LOAD(USER) + CREATE_EXT_RT(USER) + QUIESCE_CONN(USER)
TESTDB   SYSADM(GROUP) + DBADM(USER) + CREATETAB(USER, GROUP) +
            BINDADD(USER, GROUP) + CONNECT(USER, GROUP) +
            CREATE_NOT_FENC(USER) + IMPLICIT_SCHEMA(USER, GROUP) +
            LOAD(USER) + CREATE_EXT_RT(USER) + QUIESCE_CONN(USER)

3 record(s) selected.
```

戻される情報

表 292. SNAP_GET_APPL_INFO 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
AGENT_ID	BIGINT	agent_id - アプリケーション・ハンドル (エージェント ID)
APPL_STATUS	VARCHAR(22)	appl_status - アプリケーション状況。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • BACKUP • COMMIT_ACT • COMP • CONNECTED • CONNECTPEND • CREATE_DB • DECOUPLED • DISCONNECTPEND • INTR • IOERROR_WAIT • LOAD • LOCKWAIT • QUIESCE_TABLESPACE • RECOMP • REMOTE_RQST • RESTART • RESTORE • ROLLBACK_ACT • ROLLBACK_TO_SAVEPOINT • TEND • THABRT • THCOMT • TPREP • UNLOAD • UOWEXEC • UOWWAIT • WAITFOR_REMOTE
CODEPAGE_ID	BIGINT	codepage_id - アプリケーションで使用するコード・ページ ID
NUM_ASSOC_AGENTS	BIGINT	num_assoc_agents - 関連したエージェント数

表 292. SNAP_GET_APPL_INFO 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
COORD_NODE_NUM	SMALLINT	coord_node - コーディネーター・ノード

表 292. SNAP_GET_APPL_INFO 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
AUTHORITY_LVL	VARCHAR(512)	authority_lvl - ユーザー許可レベル。 このインターフェースは、sql.h とそのソースで定義されているデータベース権限に基づくテキスト ID を戻します。これは authority(source, ...) + authority(source, ...) + ... という形式です。権限 (authority) のソース (source) は複数でも可能であり、USER、GROUP、または USER と GROUP のいずれかになります。 権限 (authority) に指定可能な値は以下のとおりです。 • BINDADD • CONNECT • CREATE_EXT_RT • CREATE_NOT_FENC • CREATETAB • DBADM • IMPLICIT_SCHEMA • LOAD • LIBADM • QUIESCE_CONN • SECADM • SYSADM • SYSCTRL • SYSMAINT • SYSMON • SYSQUIESCE ソース (source) に指定可能な値は以下のとおりです。 • USER - ユーザーに付与された権限、またはそのユーザーに付与されているロールに付与された権限。 • GROUP - ユーザーが属するグループに付与された権限、またはユーザーが属するグループに付与されるロールに付与された権限。

表 292. SNAP_GET_APPL_INFO 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
CLIENT_PID	BIGINT	client_pid - クライアント・プロセス ID
COORD_AGENT_PID	BIGINT	coord_agent_pid - コーディネーター・エージェント
STATUS_CHANGE_TIME	TIMESTAMP	status_change_time - アプリケーション状況変更時刻

表 292. SNAP_GET_APPL_INFO 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
CLIENT_PLATFORM	VARCHAR(12)	client_platform - クライアント・オペレーティング・プラットフォーム。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。 • AIX • AIX64 • AS400_DRDA • DOS • DYNIX • HP • HP64 • HPIA • HPIA64 • LINUX • LINUX390 • LINUXIA64 • LINUXPPC • LINUXPPC64 • LINUXX8664 • LINUXZ64 • MAC • MVS_DRDA • NT • NT64 • OS2 • OS390 • SCO • SGI • SNI • SUN • SUN64 • 不明 (UNKNOWN) • UNKNOWN_DRDA • VM_DRDA • VSE_DRDA • WINDOWS • WINDOWS95

表 292. SNAP_GET_APPL_INFO 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
CLIENT_PROTOCOL	VARCHAR(10)	client_protocol - クライアント通信プロトコル。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。 • CPIC • LOCAL • NETBIOS • NPIPE • TCPIP (DB2 UDB の場合) • TCPIP4 • TCPIP6
TERRITORY_CODE	SMALLINT	territory_code - データベース・テリトリー・コード
APPL_NAME	VARCHAR(256)	appl_name - アプリケーション名
APPL_ID	VARCHAR(128)	appl_id - アプリケーション ID
SEQUENCE_NO	VARCHAR(4)	sequence_no - シーケンス番号
PRIMARY_AUTH_ID	VARCHAR(128)	auth_id - 許可 ID
SESSION_AUTH_ID	VARCHAR(128)	session_auth_id - セッション許可 ID
CLIENT_NNAME	VARCHAR(128)	client_nname モニター・エレメントは使用すべきではありません。返される値は無効な値です。
CLIENT_PRDID	VARCHAR(128)	client_prdid - クライアント製品/バージョン ID
INPUT_DB_ALIAS	VARCHAR(128)	input_db_alias - 入力データベース別名
CLIENT_DB_ALIAS	VARCHAR(128)	client_db_alias - アプリケーションで使用するデータベース別名
DB_NAME	VARCHAR(128)	db_name - データベース名
DB_PATH	VARCHAR(1024)	db_path - データベース・パス
EXECUTION_ID	VARCHAR(128)	execution_id - ユーザー・ログイン ID
CORR_TOKEN	VARCHAR(128)	corr_token - DRDA 相関トークン
TPMON_CLIENT_USERID	VARCHAR(256)	tpmon_client_userid - TP モニター・クライアント・ユーザー ID
TPMON_CLIENT_WKSTN	VARCHAR(256)	tpmon_client_wkstn - TP モニター・クライアント・ワークステーション名
TPMON_CLIENT_APP	VARCHAR(256)	tpmon_client_app - TP モニター・クライアント・アプリケーション名

表 292. SNAP_GET_APPL_INFO 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
TPMON_ACC_STR	VARCHAR(200)	tpmon_acc_str - TP モニター・クライアント・アカウンティング・ストリング
DBPARTITIONNUM	SMALLINT	行のデータが検索されたデータベース・パーティション。

SNAP_GET_BP 表関数 - bufferpool 論理グループのスナップショット情報の検索

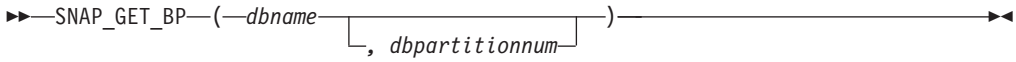
注: この表関数は使用すべきではなく、 661 ページの『SNAPBP 管理ビューおよび SNAP_GET_BP_V95 表関数 - bufferpool 論理グループのスナップショット情報の検索』 に置き換えられました。

SNAP_GET_BP 表関数は、バッファー・プール・スナップショットから、特に bufferpool 論理データ・グループのバッファー・プール情報を戻します。

SNAP_GET_BP 表関数を SNAP_GET_BP_PART 表関数とともに使用すると、GET SNAPSHOT FOR ALL BUFFERPOOLS CLP コマンドに相当するデータが提供されます。

戻される可能性のある情報の完全なリストは、 1204 ページの表 293を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、空ストリングを指定します。現在接続されているデータベースと同じインスタンス内のすべてのデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。

dbname が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプを持つファイルが存在しない場合、SNAP_GET_BP 表関数は、現在接続中のデータベースおよびデータベース・パーティション番号のスナップショットを取得します。

許可

- SYSMON 権限
- SNAP_GET_BP 表関数に対する EXECUTE 特権。

例

現在接続されているデータベース・パーティションのすべてのアクティブ・データベースのすべてのバッファー・プールについて、物理および論理読み取りの合計を取得します。

```
SELECT SUBSTR(T.DB_NAME,1,10) AS DB_NAME,
   SUBSTR(T.BP_NAME,1,20) AS BP_NAME,
   (T.POOL_DATA_L_READS+T.POOL_INDEX_L_READS) AS TOTAL_LOGICAL_READS,
   (T.POOL_DATA_P_READS+T.POOL_INDEX_P_READS) AS TOTAL_PHYSICAL_READS,
   T.DBPARTITIONNUM
   FROM TABLE(SNAP_GET_BP(CAST(NULL AS VARCHAR(128)), -1)) AS T
```

以下はこの照会の出力例です。

```
DB_NAME    BP_NAME          TOTAL_LOGICAL_READS  ...
---------- ------------...- -------------------- ...
SAMPLE     IBMDEFAULTBP                       0 ...
TOOLSDB    IBMDEFAULTBP                       0 ...
TOOLSDB    BP32K0000                          0 ...

  3 record(s) selected.
```

この照会からの出力 (続き)。

```
... TOTAL_PHYSICAL_READS DBPARTITIONNUM
... -------------------- --------------
...                    0              0
...                    0              0
...                    0              0
```

戻される情報

表 293. SNAP_GET_BP 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。

表 293. SNAP_GET_BP 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
BP_NAME	VARCHAR(128)	bp_name - バッファー・プール名
DB_NAME	VARCHAR(128)	db_name - データベース名
DB_PATH	VARCHAR(1024)	db_path - データベース・パス
INPUT_DB_ALIAS	VARCHAR(128)	input_db_alias - 入力データベース別名
POOL_DATA_L_READS	BIGINT	pool_data_l_reads - バッファー・プール・データの論理読み取り
POOL_DATA_P_READS	BIGINT	pool_data_p_reads - バッファー・プール・データの物理読み取り
POOL_DATA_WRITES	BIGINT	pool_data_writes - バッファー・プールへのデータの書き込み
POOL_INDEX_L_READS	BIGINT	pool_index_l_reads - バッファー・プール索引の論理読み取り
POOL_INDEX_P_READS	BIGINT	pool_index_p_reads - バッファー・プール索引の物理読み取り
POOL_INDEX_WRITES	BIGINT	pool_index_writes - バッファー・プール索引の書き込み
POOL_XDA_L_READS	BIGINT	pool_xda_l_reads - バッファー・プール XDA データの論理読み取り
POOL_XDA_P_READS	BIGINT	pool_xda_p_reads - バッファー・プール XDA データの物理読み取り
POOL_XDA_WRITES	BIGINT	pool_xda_writes - バッファー・プール XDA データの書き込み
POOL_READ_TIME	BIGINT	pool_read_time - バッファー・プール物理読み取り時間の合計
POOL_WRITE_TIME	BIGINT	pool_write_time - バッファー・プール物理書き込み時間の合計
POOL_ASYNC_DATA_READS	BIGINT	pool_async_data_reads - バッファー・プール非同期データ読み取り
POOL_ASYNC_DATA_WRITES	BIGINT	pool_async_data_writes - バッファー・プール非同期データ書き込み
POOL_ASYNC_INDEX_READS	BIGINT	pool_async_index_reads - バッファー・プール非同期索引読み取り
POOL_ASYNC_INDEX_WRITES	BIGINT	pool_async_index_writes - バッファー・プール非同期索引書き込み
POOL_ASYNC_XDA_READS	BIGINT	pool_async_xda_reads - バッファー・プール非同期 XDA データ読み取り
POOL_ASYNC_XDA_WRITES	BIGINT	pool_async_xda_writes - バッファー・プール非同期 XDA データ書き込み
POOL_ASYNC_READ_TIME	BIGINT	pool_async_read_time - バッファー・プール非同期読み取り時間

表 293. SNAP_GET_BP 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
POOL_ASYNC_WRITE_TIME	BIGINT	pool_async_write_time - バッファー・プール非同期書き込み時間
POOL_ASYNC_DATA_READ_REQS	BIGINT	pool_async_data_read_reqs - バッファー・プール非同期読み取り要求
POOL_ASYNC_INDEX_READ_REQS	BIGINT	pool_async_index_read_reqs - バッファー・プール非同期索引読み取り要求
POOL_ASYNC_XDA_READ_REQS	BIGINT	pool_async_xda_read_reqs - バッファー・プール非同期 XDA 読み取り要求
DIRECT_READS	BIGINT	direct_reads - データベースからの直接読み取り
DIRECT_WRITES	BIGINT	direct_writes - データベースへの直接書き込み
DIRECT_READ_REQS	BIGINT	direct_read_reqs - 直接読み取り要求
DIRECT_WRITE_REQS	BIGINT	direct_write_reqs - 直接書き込み要求
DIRECT_READ_TIME	BIGINT	direct_read_time - 直接読み取り時間
DIRECT_WRITE_TIME	BIGINT	direct_write_time - 直接書き込み時間
UNREAD_PREFETCH_PAGES	BIGINT	unread_prefetch_pages - 読み取り不能プリフェッチ・ページ
FILES_CLOSED	BIGINT	files_closed - 閉じられたデータベース・ファイル
POOL_TEMP_DATA_L_READS	BIGINT	pool_temp_data_l_reads - バッファー・プール一時データの論理読み取り
POOL_TEMP_DATA_P_READS	BIGINT	pool_temp_data_p_reads - バッファー・プール一時データの物理読み取り
POOL_TEMP_INDEX_L_READS	BIGINT	pool_temp_index_l_reads - バッファー・プール一時索引の論理読み取り
POOL_TEMP_INDEX_P_READS	BIGINT	pool_temp_index_p_reads - バッファー・プール一時索引の物理読み取り
POOL_TEMP_XDA_L_READS	BIGINT	pool_temp_xda_l_reads - バッファー・プール一時 XDA データの論理読み取り
POOL_TEMP_XDA_P_READS	BIGINT	pool_temp_xda_p_reads - バッファー・プール一時 XDA データの物理読み取り : モニター・エレメント

表 293. SNAP_GET_BP 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
POOL_NO_VICTIM_BUFFER	BIGINT	pool_no_victim_buffer - バッファー・プールの非ビクティム・バッファー数
PAGES_FROM_BLOCK_IOS	BIGINT	pages_from_block_ios - ブロック入出力によって読み取られたページ数の合計
PAGES_FROM_VECTORED_IOS	BIGINT	pages_from_vectored_ios - ベクトル化入出力によって読み取られたページ数の合計
PHYSICAL_PAGE_MAPS	BIGINT	physical_page_maps モニター・エレメントは廃止されています。廃止されたモニター・エレメントには NULL 値が戻されます。
VECTORED_IOS	BIGINT	vectored_ios - ベクトル化入出力要求数
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

SNAP_GET_CONTAINER

注: この表関数は使用すべきではなく、 670 ページの『SNAPCONTAINER 管理ビューおよび SNAP_GET_CONTAINER_V91 表関数 - tablespace_container 論理データ・グループ・スナップショット情報の検索』 に置き換えられました。

```
►►—SNAP_GET_CONTAINER—(—dbname—,—dbpartitionnum—)———————————————►◄
```

スキーマは SYSPROC です。

SNAP_GET_CONTAINER 表関数は、tablespace_container 論理データ・グループからのスナップショット情報を戻します。

dbname
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(255) の入力引数。"Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションには -1 を指定します。 NULL 値を指定すると、-1 が暗黙的に設定されます。

どちらのパラメーターも NULL に設定された場合は、対応するスナップショット API 要求タイプの SNAPSHOT_FILEW ストアード・プロシージャーによって以前にファイルが作成されていない場合にのみ、スナップショットがとられます。

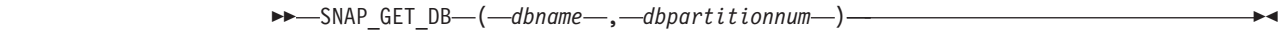
この関数より、次に示されている表が戻されます。

表 294. SNAP_GET_CONTAINER 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
TBSP_NAME	VARCHAR(128)	tablespace_name - 表スペース名
TBSP_ID	BIGINT	tablespace_id - 表スペース ID
CONTAINER_NAME	VARCHAR(256)	container_name - コンテナー名
CONTAINER_ID	BIGINT	container_id - コンテナー ID
CONTAINER_TYPE	SMALLINT	container_type - コンテナー・タイプ
TOTAL_PAGES	BIGINT	container_total_pages - コンテナー内の合計ページ数
USABLE_PAGES	BIGINT	container_usable_pages - コンテナー内の使用可能なページ数
ACCESSIBLE	SMALLINT	container_accessible - コンテナーのアクセス可能性
STRIPE_SET	BIGINT	container_stripe_set - ストライプ・セット
DBPARTITIONNUM	SMALLINT	node_number - ノード番号

SNAP_GET_DB

注: この表関数は使用すべきではなく、 1219 ページの『SNAP_GET_DB_V91 table function - dbase 論理グループからのスナップショット情報の検索』 に置き換えられました。



スキーマは SYSPROC です。

SNAP_GET_DB 表関数は、データベースからのスナップショット情報を戻します。

dbname
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(255) の入力引数。"Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE

DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

どちらのパラメーターも NULL に設定された場合は、対応するスナップショット API 要求タイプの SNAPSHOT_FILEW ストアード・プロシージャーによって以前にファイルが作成されていない場合にのみ、スナップショットがとられます。

この関数より、次に示されている表が戻されます。

表 295. SNAP_GET_DB 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
DB_NAME	VARCHAR(128)	**db_name** - データベース名
DB_PATH	VARCHAR(1024)	**db_path** - データベース・パス
INPUT_DB_ALIAS	VARCHAR(128)	**input_db_alias** - 入力データベース別名
DB_STATUS	BIGINT	**db_status** - データベース状況
CATALOG_PARTITION	SMALLINT	**catalog_node** - カタログ・ノード番号
CATALOG_PARTITION_NAME	VARCHAR(128)	**catalog_node_name** - カタログ・ノード・ネットワーク名
SERVER_PLATFORM	INTEGER	**server_platform** - サーバーのオペレーティング・システム
DB_LOCATION	INTEGER	**db_location** - データベース・ロケーション
DB_CONN_TIME	TIMESTAMP	**db_conn_time** - データベース・アクティブ化タイム・スタンプ
LAST_RESET	TIMESTAMP	**last_reset** - 最後のリセット・タイム・スタンプ
LAST_BACKUP	TIMESTAMP	**last_backup** - 最終バックアップ・タイム・スタンプ
CONNECTIONS_TOP	BIGINT	**connections_top** - 同時接続の最大数
TOTAL_CONS	BIGINT	**total_cons** - データベース・アクティブ化以降の接続
TOTAL_SEC_CONS	BIGINT	**total_sec_cons** - 2 次接続
APPLS_CUR_CONS	BIGINT	**appls_cur_cons** - 現在接続されているアプリケーション

表 295. SNAP_GET_DB 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
APPLS_IN_DB2	BIGINT	**appls_in_db2** - データベースで現在実行中のアプリケーション
NUM_ASSOC_AGENTS	BIGINT	**num_assoc_agents** - 関連したエージェント数
AGENTS_TOP	BIGINT	**agents_top** - 作成されたエージェントの数
COORD_AGENTS_TOP	BIGINT	**coord_agents_top** - コーディネーター・エージェント最大数
LOCKS_HELD	BIGINT	**locks_held** - ロック保持数
LOCK_WAITS	BIGINT	**lock_waits** - ロック待機数
LOCK_WAIT_TIME	BIGINT	**lock_wait_time** - ロック待機中の時間
LOCK_LIST_IN_USE	BIGINT	**lock_list_in_use** - 使用中のロック・リスト・メモリーの合計
DEADLOCKS	BIGINT	**deadlocks** - デッドロック検出数
LOCK_ESCALS	BIGINT	**lock_escals** - ロック・エスカレーション数
X_LOCK_ESCALS	BIGINT	**x_lock_escals** - 排他ロック・エスカレーション数
LOCKS_WAITING	BIGINT	**locks_waiting** - ロックで待機中の現行エージェント
LOCK_TIMEOUTS	BIGINT	**lock_timeouts** - ロック・タイムアウト数
NUM_INDOUBT_TRANS	BIGINT	**num_indoubt_trans** - 未確定トランザクション数
SORT_HEAP_ALLOCATED	BIGINT	**sort_heap_allocated** - 割り振られたソート・ヒープの合計
SORT_SHRHEAP_ALLOCATED	BIGINT	**sort_shrheap_allocated** - 現在割り振られているソート共有ヒープ
SORT_SHRHEAP_TOP	BIGINT	**sort_shrheap_top** - ソート共有ヒープの最高水準点
TOTAL_SORTS	BIGINT	**total_sorts** - ソート合計
TOTAL_SORT_TIME	BIGINT	**total_sort_time** - ソート時間合計
SORT_OVERFLOWS	BIGINT	**sort_overflows** - ソート・オーバーフロー
ACTIVE_SORTS	BIGINT	**active_sorts** - アクティブ・ソート
POOL_DATA_L_READS	BIGINT	**pool_data_l_reads** - バッファー・プール・データの論理読み取り
POOL_DATA_P_READS	BIGINT	**pool_data_p_reads** - バッファー・プール・データの物理読み取り

表 295. SNAP_GET_DB 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
POOL_TEMP_DATA_L_READS	BIGINT	**pool_temp_data_l_reads** - バッファー・プール一時データの論理読み取り
POOL_TEMP_DATA_P_READS	BIGINT	**pool_temp_data_p_reads** - バッファー・プール一時データの物理読み取り
POOL_ASYNC_DATA_READS	BIGINT	**pool_async_data_reads** - バッファー・プール非同期データ読み取り
POOL_DATA_WRITES	BIGINT	**pool_data_writes** - バッファー・プールへのデータの書き込み
POOL_ASYNC_DATA_WRITES	BIGINT	**pool_async_data_writes** - バッファー・プール非同期データ書き込み
POOL_INDEX_L_READS	BIGINT	**pool_index_l_reads** - バッファー・プール索引の論理読み取り
POOL_INDEX_P_READS	BIGINT	**pool_index_p_reads** - バッファー・プール索引の物理読み取り
POOL_TEMP_INDEX_L_READS	BIGINT	**pool_temp_index_l_reads** - バッファー・プール一時索引の論理読み取り
POOL_TEMP_INDEX_P_READS	BIGINT	**pool_temp_index_p_reads** - バッファー・プール一時索引の物理読み取り
POOL_INDEX_WRITES	BIGINT	**pool_index_writes** - バッファー・プール索引の書き込み
POOL_ASYNC_INDEX_READS	BIGINT	**pool_async_index_reads** - バッファー・プール非同期索引読み取り
POOL_ASYNC_INDEX_WRITES	BIGINT	**pool_async_index_writes** - バッファー・プール非同期索引書き込み
POOL_READ_TIME	BIGINT	**pool_read_time** - バッファー・プール物理読み取り時間の合計
POOL_WRITE_TIME	BIGINT	**pool_write_time** - バッファー・プール物理書き込み時間の合計
POOL_ASYNC_READ_TIME	BIGINT	**pool_async_read_time** - バッファー・プール非同期読み取り時間
POOL_ASYNC_WRITE_TIME	BIGINT	**pool_async_write_time** - バッファー・プール非同期書き込み時間
POOL_ASYNC_DATA_ READ_REQS	BIGINT	**pool_async_data_read_reqs** - バッファー・プール非同期読み取り要求
POOL_ASYNC_INDEX_ READ_REQS	BIGINT	**pool_async_index_read_reqs** - バッファー・プール非同期索引読み取り要求

表 295. SNAP_GET_DB 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
POOL_NO_VICTIM_BUFFER	BIGINT	**pool_no_victim_buffer** - バッファー・プールの非ビクティム・バッファー数
POOL_LSN_GAP_CLNS	BIGINT	**pool_lsn_gap_clns** - 起動されたバッファー・プール・ログ・スペース・クリーナー
POOL_DRTY_PG_STEAL_CLNS	BIGINT	**pool_drty_pg_steal_clns** - 起動されたバッファー・プール・ビクティム・ページ・クリーナー
POOL_DRTY_PG_THRSH_CLNS	BIGINT	**pool_drty_pg_thrsh_clns** - 起動されたバッファー・プールしきい値クリーナー
PREFETCH_WAIT_TIME	BIGINT	**prefetch_wait_time** - プリフェッチ待ち時間
UNREAD_PREFETCH_PAGES	BIGINT	**unread_prefetch_pages** - 読み取り不能プリフェッチ・ページ
DIRECT_READS	BIGINT	**direct_reads** - データベースからの直接読み取り
DIRECT_WRITES	BIGINT	**direct_writes** - データベースへの直接書き込み
DIRECT_READ_REQS	BIGINT	**direct_read_reqs** - 直接読み取り要求
DIRECT_WRITE_REQS	BIGINT	**direct_write_reqs** - 直接書き込み要求
DIRECT_READ_TIME	BIGINT	**direct_read_time** - 直接読み取り時間
DIRECT_WRITE_TIME	BIGINT	**direct_write_time** - 直接書き込み時間
FILES_CLOSED	BIGINT	**files_closed** - 閉じられたデータベース・ファイル
POOL_DATA_TO_ESTORE	BIGINT	**pool_data_to_estore** ESTORE モニター・エレメントは廃止されています。廃止されたモニター・エレメントには NULL 値が戻されます。
POOL_INDEX_TO_ESTORE	BIGINT	**pool_index_to_estore** ESTORE モニター・エレメントは廃止されています。廃止されたモニター・エレメントには NULL 値が戻されます。
POOL_INDEX_FROM_ESTORE	BIGINT	**pool_index_from_estore** ESTORE モニター・エレメントは廃止されています。廃止されたモニター・エレメントには NULL 値が戻されます。

表 295. SNAP_GET_DB 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
POOL_DATA_FROM_ESTORE	BIGINT	**pool_data_from_estore** ESTORE モニター・エレメントは廃止されています。廃止されたモニター・エレメントには NULL 値が戻されます。
ELAPSED_EXEC_TIME_S	BIGINT	**elapsed_exec_time** - ステートメント実行経過時間 (秒単位)*
ELAPSED_EXEC_TIME_MS	BIGINT	**elapsed_exec_time** - ステートメント実行経過時間 (小数部、マイクロ秒単位)*
COMMIT_SQL_STMTS	BIGINT	**commit_sql_stmts** - 試行されたコミット・ステートメント
ROLLBACK_SQL_STMTS	BIGINT	**rollback_sql_stmts** - 試行されたロールバック・ステートメント
DYNAMIC_SQL_STMTS	BIGINT	**dynamic_sql_stmts** - 試行された動的 SQL ステートメント
STATIC_SQL_STMTS	BIGINT	**static_sql_stmts** - 試行された静的 SQL ステートメント
FAILED_SQL_STMTS	BIGINT	**failed_sql_stmts** - 失敗したステートメント操作
SELECT_SQL_STMTS	BIGINT	**select_sql_stmts** - 実行された選択 SQL ステートメント
UID_SQL_STMTS	BIGINT	**uid_sql_stmts** - 実行された更新/挿入/削除 SQL ステートメント
DDL_SQL_STMTS	BIGINT	**ddl_sql_stmts** - データ定義言語 (DDL) SQL ステートメント
INT_AUTO_REBINDS	BIGINT	**int_auto_rebinds** - 内部自動再バインド
INT_ROWS_DELETED	BIGINT	**int_rows_deleted** - 削除された内部行数
INT_ROWS_INSERTED	BIGINT	**int_rows_inserted** - 挿入された内部行数
INT_ROWS_UPDATED	BIGINT	**int_rows_updated** - 更新された内部行数
INT_COMMITS	BIGINT	**int_commits** - 内部コミット数
INT_ROLLBACKS	BIGINT	**int_rollbacks** - 内部ロールバック数
INT_DEADLOCK_ROLLBACKS	BIGINT	**int_deadlock_rollbacks** - デッドロックによる内部ロールバック数
ROWS_DELETED	BIGINT	**rows_deleted** - 削除行数
ROWS_INSERTED	BIGINT	**rows_inserted** - 挿入行数
ROWS_UPDATED	BIGINT	**rows_updated** - 更新行数
ROWS_SELECTED	BIGINT	**rows_selected** - 選択行数
ROWS_READ	BIGINT	**rows_read** - 読み取り行数

表 295. SNAP_GET_DB 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
BINDS_PRECOMPILES	BIGINT	**binds_precompiles** - 試行されたバインド/プリコンパイル
TOTAL_LOG_AVAILABLE	BIGINT	**total_log_available** - 使用可能なログ合計
TOTAL_LOG_USED	BIGINT	**total_log_used** - 使用されているログ・スペースの合計
SEC_LOG_USED_TOP	BIGINT	**sec_log_used_top** - 使用された最大 2 次ログ・スペース
TOT_LOG_USED_TOP	BIGINT	**tot_log_used_top** - 使用された最大合計ログ・スペース
SEC_LOGS_ALLOCATED	BIGINT	**sec_logs_allocated** - 現在割り振られている 2 次ログ
LOG_READS	BIGINT	**log_reads** - 読み取られたログ・ページの数
LOG_READ_TIME_S	BIGINT	**log_read_time** - ログ読み取り時間 (秒単位)†
LOG_READ_TIME_NS	BIGINT	**log_read_time** - ログ読み取り時間 (小数部、ナノ秒単位)†
LOG_WRITES	BIGINT	**log_writes** - 書き込まれたログ・ページの数
LOG_WRITE_TIME_S	BIGINT	**log_write_time** - ログ書き込み時間 (秒単位)†
LOG_WRITE_TIME_NS	BIGINT	**log_write_time** - ログ書き込み時間 (小数部、ナノ秒単位)†
NUM_LOG_WRITE_IO	BIGINT	**num_log_write_io** - ログ書き込み数
NUM_LOG_READ_IO	BIGINT	**num_log_read_io** - ログ読み取り数
NUM_LOG_PART_PAGE_IO	BIGINT	**num_log_part_page_io** - 部分ログ・ページ書き込み数
NUM_LOG_BUFFER_FULL	BIGINT	**num_log_buffer_full** - フル・ログ・バッファーの回数
NUM_LOG_DATA_FOUND_IN_BUFFER	BIGINT	**num_log_data_found_in_buffer** - ログ・データがバッファーにある回数
APPL_ID_OLDEST_XACT	BIGINT	**appl_id_oldest_xact** - 最も古いトランザクションを使用するアプリケーション
LOG_TO_REDO_FOR_RECOVERY	BIGINT	**log_to_redo_for_recovery** - リカバリーの場合に再実行されるログの量
LOG_HELD_BY_DIRTY_PAGES	BIGINT	**log_held_by_dirty_pages** - ダーティー・ページ別に計算されるログ・スペースの量

表 295. SNAP_GET_DB 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
PKG_CACHE_LOOKUPS	BIGINT	**pkg_cache_lookups** - パッケージ・キャッシュ参照
PKG_CACHE_INSERTS	BIGINT	**pkg_cache_inserts** - パッケージ・キャッシュ挿入
PKG_CACHE_NUM_ OVERFLOWS	BIGINT	**pkg_cache_num_overflows** - パッケージ・キャッシュ・オーバーフロー数
PKG_CACHE_SIZE_TOP	BIGINT	**pkg_cache_size_top** - パッケージ・キャッシュ最高水準点
APPL_SECTION_LOOKUPS	BIGINT	**appl_section_lookups** - セクションの参照回数
APPL_SECTION_INSERTS	BIGINT	**appl_section_inserts** - セクション挿入数
CAT_CACHE_LOOKUPS	BIGINT	**cat_cache_lookups** - カタログ・キャッシュ参照数
CAT_CACHE_INSERTS	BIGINT	**cat_cache_inserts** - カタログ・キャッシュ挿入数
CAT_CACHE_OVERFLOWS	BIGINT	**cat_cache_overflows** - カタログ・キャッシュ・オーバーフロー数
CAT_CACHE_SIZE_TOP	BIGINT	**cat_cache_size_top** - カタログ・キャッシュ最高水準点
PRIV_WORKSPACE_SIZE_TOP	BIGINT	**priv_workspace_size_top** - 専用ワークスペースの最大サイズ
PRIV_WORKSPACE_NUM_ OVERFLOWS	BIGINT	**priv_workspace_num_overflows** - 専用ワークスペースのオーバーフロー回数
PRIV_WORKSPACE_SECTION_ INSERTS	BIGINT	**priv_workspace_section_inserts** - 専用ワークスペース・セクション挿入
PRIV_WORKSPACE_SECTION_ LOOKUPS	BIGINT	**priv_workspace_section_lookups** - 専用ワークスペース・セクションの参照
SHR_WORKSPACE_SIZE_TOP	BIGINT	**shr_workspace_size_top** - 最大共有ワークスペース・サイズ
SHR_WORKSPACE_NUM_ OVERFLOWS	BIGINT	**shr_workspace_num_overflows** - 共有ワークスペースのオーバーフロー回数
SHR_WORKSPACE_SECTION_ INSERTS	BIGINT	**shr_workspace_section_inserts** - 共有ワークスペース・セクション挿入数
SHR_WORKSPACE_SECTION_ LOOKUPS	BIGINT	**shr_workspace_section_lookups** - 共有ワークスペース・セクションの参照回数

表 295. SNAP_GET_DB 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
TOTAL_HASH_JOINS	BIGINT	**total_hash_joins** - ハッシュ結合の合計
TOTAL_HASH_LOOPS	BIGINT	**total_hash_loops** - ハッシュ・ループの合計
HASH_JOIN_OVERFLOWS	BIGINT	**hash_join_overflows** - ハッシュ結合のオーバーフロー
HASH_JOIN_SMALL_ OVERFLOWS	BIGINT	**hash_join_small_overflows** - ハッシュ結合の短精度オーバーフロー
NUM_DB_STORAGE_PATHS	BIGINT	**num_db_storage_paths** - 自動ストレージ・パスの数
DBPARTITIONNUM	SMALLINT	**node_number** - ノード番号

* この列の元になるモニター・エレメントの合計消費時間を計算するには、このモニター・エレメントに関する _S で終わっている列で報告されている整数秒と、このモニター・エレメントに関する _MS で終わっている列に報告されている小数秒を、次の式を使用して加算しなければなりません: (*monitor-element-name*_S × 1,000,000 + *monitor-element-name*_MS) ÷ 1,000,000. 例えば、(ELAPSED_EXEC_TIME_S × 1,000,000 + ELAPSED_EXEC_TIME_MS) ÷ 1,000,000。

†このモニター・エレメントの合計経過時間を計算するには、このモニター・エレメントに関する _S で終わっている列で報告されている整数秒と、このモニター・エレメントに関する _NS で終わっている列に報告されている小数秒を、次の式を使用して加算しなければなりません: (*monitor-element-name*_S × 1,000,000,000 + *monitor-element-name*_NS) ÷ 1,000,000,000。例えば、(LOG_READ_TIME_S × 1,000,000,000 + LOG_READ_TIME_NS) ÷ 1,000,000,000。

SNAP_GET_DBM 表関数 - dbm 論理グループ・スナップショット情報の検索

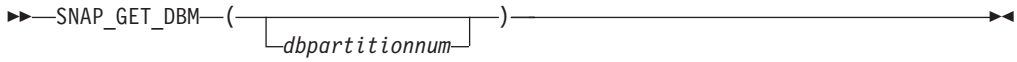
注: この表関数は使用すべきではなく、 693 ページの『SNAPDBM 管理ビューおよび SNAP_GET_DBM_V95 表関数 - dbm 論理グループ・スナップショット情報の検索』 に置き換えられました。

SNAP_GET_DBM 表関数は、DB2 データベース・マネージャー (dbm) 論理グループのスナップショット・モニター情報を戻します。

SNAP_GET_DBM_MEMORY_POOL、SNAP_GET_FCM、SNAP_GET_FCM_PART、および SNAP_GET_SWITCHES 表関数と共に使用すると、SNAP_GET_DBM 表関数は、GET SNAPSHOT FOR DBM コマンドと同等のデータを提供します。

戻される可能性のある情報の完全なリストは、 1217 ページの表 296を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。この入力オプションが使用されない場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbpartitionnum が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプが含まれるファイルが存在しない場合、SNAP_GET_DBM 表関数はメモリーからスナップショットを呼び出します。

許可

- SYSMON 権限
- SNAP_GET_DBM 表関数に対する EXECUTE 特権。

例

データベース・パーティション番号 2 の開始時刻と現行状況を検索します。

```
SELECT DB2START_TIME, DB2_STATUS FROM TABLE(SNAP_GET_DBM(2)) AS T
```

以下はこの照会の出力例です。

```
DB2START_TIME              DB2_STATUS
-------------------------- ------------
2006-01-06-14.59.59.062798 ACTIVE
```

戻される情報

表 296. SNAP_GET_DBM 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
SORT_HEAP_ALLOCATED	BIGINT	sort_heap_allocated - 割り振られたソート・ヒープの合計
POST_THRESHOLD_SORTS	BIGINT	post_threshold_sorts - ポストしきい値ソート

表 296. SNAP_GET_DBM 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
PIPED_SORTS_REQUESTED	BIGINT	piped_sorts_requested - 要求されたパイプ・ソート数
PIPED_SORTS_ACCEPTED	BIGINT	piped_sorts_accepted - 受け入れられたパイプ・ソート
REM_CONS_IN	BIGINT	rem_cons_in - データベース・マネージャーへのリモート接続
REM_CONS_IN_EXEC	BIGINT	rem_cons_in_exec - データベース・マネージャーで実行中のリモート接続 : モニター・エレメント
LOCAL_CONS	BIGINT	local_cons - ローカル接続
LOCAL_CONS_IN_EXEC	BIGINT	local_cons_in_exec - データベース・マネージャーで実行中のローカル接続 : モニター・エレメント
CON_LOCAL_DBASES	BIGINT	con_local_dbases - 現行接続を使用したローカル・データベース
AGENTS_REGISTERED	BIGINT	agents_registered - 登録済みエージェント
AGENTS_WAITING_ON_TOKEN	BIGINT	agents_waiting_on_token - トークン待ちエージェント
DB2_STATUS	VARCHAR(12)	db2_status - DB2 インスタンス状況。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • ACTIVE • QUIESCE_PEND • QUIESCED
AGENTS_REGISTERED_TOP	BIGINT	agents_registered_top - エージェント最大登録数
AGENTS_WAITING_TOP	BIGINT	agents_waiting_top - エージェント最大待機数
COMM_PRIVATE_MEM	BIGINT	comm_private_mem - コミット済み専用メモリー
IDLE_AGENTS	BIGINT	idle_agents - アイドル・エージェント数
AGENTS_FROM_POOL	BIGINT	agents_from_pool - プールから割り当てられたエージェント
AGENTS_CREATED_EMPTY_POOL	BIGINT	agents_created_empty_pool - エージェント・プールが空のために作成されたエージェント
COORD_AGENTS_TOP	BIGINT	coord_agents_top - コーディネーター・エージェント最大数

表 296. SNAP_GET_DBM 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
MAX_AGENT_OVERFLOWS	BIGINT	max_agent_overflows - 最大エージェント・オーバーフロー回数
AGENTS_STOLEN	BIGINT	agents_stolen - スチールされたエージェント
GW_TOTAL_CONS	BIGINT	gw_total_cons - DB2 Connect の接続試行合計回数
GW_CUR_CONS	BIGINT	gw_cur_cons - DB2 Connect の現在の接続数
GW_CONS_WAIT_HOST	BIGINT	gw_cons_wait_host - ホストの応答を待機している接続の数
GW_CONS_WAIT_CLIENT	BIGINT	gw_cons_wait_client - クライアントの要求送信を待機している接続の数
POST_THRESHOLD_ HASH_JOINS	BIGINT	post_threshold_hash_joins - ハッシュ結合のしきい値
NUM_GW_CONN_SWITCHES	BIGINT	num_gw_conn_switches - 接続切り替え回数
DB2START_TIME	TIMESTAMP	db2start_time - データベース・マネージャー開始タイム・スタンプ
LAST_RESET	TIMESTAMP	last_reset - 最後のリセット・タイム・スタンプ
NUM_NODES_IN_ DB2_INSTANCE	INTEGER	num_nodes_in_db2_instance - データベース・パーティション内のノード数
PRODUCT_NAME	VARCHAR(32)	product_name - 製品名
SERVICE_LEVEL	VARCHAR(18)	service_level - サービス・レベル
SORT_HEAP_TOP	BIGINT	sort_heap_top - ソート専用ヒープの最高水準点
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

SNAP_GET_DB_V91 table function - dbase 論理グループからのスナップショット情報の検索

注: この表関数は使用すべきではなく、 828 ページの『SNAPDB 管理ビューおよび SNAP_GET_DB_V95 表関数 - dbase 論理グループからのスナップショット情報の検索』 に置き換えられました。

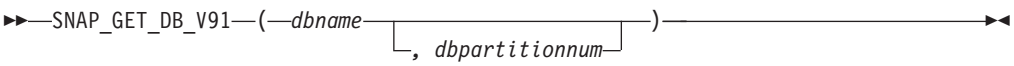
SNAP_GET_DB_V91 表関数は、データベース (dbase) 論理グループからのスナップショット情報を戻します。

SNAP_GET_DB_V91 表関数を SNAP_GET_DB_MEMORY_POOL、SNAP_GET_DETAILLOG_V91、 SNAP_GET_HADR、および

SNAP_GET_STORAGE_PATHS 表関数と併せて使用することにより、GET SNAPSHOT FOR ALL DATABASES CLP コマンドと同等の情報を戻します。

戻される情報の完全なリストは、 1221 ページの表 297 を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、空ストリングを指定します。現在接続されているデータベースと同じインスタンス内のすべてのデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプを持つファイルが存在しない場合には、現在接続されているデータベースのスナップショットとデータベース・パーティション番号を SNAP_GET_DB_V91 表関数が取得します。

許可

以下のいずれかの権限または特権が必要です。

- SNAP_GET_DB_V91 表関数に対する EXECUTE 特権。
- DATAACCESS 権限
- SYSMON 権限
- SYSMAINT 権限

- SYSCTRL 権限
- SYSADM 権限

例

例 1: 現在接続されているデータベースのすべてのデータベース・パーティションに渡る集約ビューとして、状況、プラットフォーム、ロケーション、および接続時間を取り出します。

```
SELECT SUBSTR(DB_NAME, 1, 20) AS DB_NAME, DB_STATUS, SERVER_PLATFORM,
   DB_LOCATION, DB_CONN_TIME FROM TABLE(SNAP_GET_DB_V91('', -2)) AS T
```

以下はこの照会の出力例です。

```
DB_NAME     DB_STATUS    SERVER_PLATFORM ...
-------...- ------------ --------------- ...
SAMPLE      ACTIVE       AIX64           ...

  1 record(s) selected.
```

この照会からの出力 (続き)。

```
... DB_LOCATION  DB_CONN_TIME
... ------------ --------------------------
... LOCAL        2005-07-24-22.09.22.013196
```

例 2: 現在接続されているデータベースを含む同じインスタンス内にあるすべてのアクティブ・データベースのすべてのデータベース・パーティションに渡る集約ビューとして、状況、プラットフォーム、ロケーション、および接続時間を取り出します。

```
SELECT SUBSTR(DB_NAME, 1, 20) AS DB_NAME, DB_STATUS, SERVER_PLATFORM,
   DB_LOCATION, DB_CONN_TIME
   FROM TABLE(SNAP_GET_DB_V91(CAST (NULL AS VARCHAR(128)), -2)) AS T
```

以下はこの照会の出力例です。

```
DB_NAME      DB_STATUS    SERVER_PLATFORM ...
--------...- ------------ --------------- ...
TOOLSDB      ACTIVE       AIX64           ...
SAMPLE       ACTIVE       AIX64           ...
```

この照会からの出力 (続き)。

```
... DB_LOCATION  DB_CONN_TIME
... ------------ --------------------------
... LOCAL        2005-07-24-22.26.54.396335
... LOCAL        2005-07-24-22.09.22.013196
```

SNAP_GET_DB_V91 表関数のメタデータ

表 297. SNAP_GET_DB_V91 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
DB_NAME	VARCHAR(128)	db_name - データベース名
DB_PATH	VARCHAR(1024)	db_path - データベース・パス
INPUT_DB_ALIAS	VARCHAR(128)	input_db_alias - 入力データベース別名

表 297. SNAP_GET_DB_V91 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
DB_STATUS	VARCHAR(12)	db_status - データベース状況。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • ACTIVE • QUIESCE_PEND • QUIESCED • ROLLFWD
CATALOG_PARTITION	SMALLINT	catalog_node - カタログ・ノード番号
CATALOG_PARTITION_NAME	VARCHAR(128)	catalog_node_name - カタログ・ノード・ネットワーク名

表 297. SNAP_GET_DB_V91 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
SERVER_PLATFORM	VARCHAR(12)	server_platform - サーバーのオペレーティング・システム。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • AIX • AIX64 • AS400_DRDA • DOS • DYNIX • HP • HP64 • HPIA • HPIA64 • LINUX • LINUX390 • LINUXIA64 • LINUXPPC • LINUXPPC64 • LINUXX8664 • LINUXZ64 • MAC • MVS_DRDA • NT • NT64 • OS2 • OS390 • SCO • SGI • SNI • SUN • SUN64 • 不明 (UNKNOWN) • UNKNOWN_DRDA • VM_DRDA • VSE_DRDA • WINDOWS • WINDOWS95

表 297. SNAP_GET_DB_V91 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
DB_LOCATION	VARCHAR(12)	db_location - データベース・ロケーション。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • LOCAL • REMOTE
DB_CONN_TIME	TIMESTAMP	db_conn_time - データベース活動化タイム・スタンプ
LAST_RESET	TIMESTAMP	last_reset - 最後のリセット・タイム・スタンプ
LAST_BACKUP	TIMESTAMP	last_backup - 最終バックアップ・タイム・スタンプ
CONNECTIONS_TOP	BIGINT	connections_top - 同時接続の最大数
TOTAL_CONS	BIGINT	total_cons - データベース活動化以降の接続
TOTAL_SEC_CONS	BIGINT	total_sec_cons - 2 次接続
APPLS_CUR_CONS	BIGINT	appls_cur_cons - 現在接続されているアプリケーション
APPLS_IN_DB2	BIGINT	appls_in_db2 - データベースで現在実行中のアプリケーション
NUM_ASSOC_AGENTS	BIGINT	num_assoc_agents - 関連したエージェント数
AGENTS_TOP	BIGINT	agents_top - 作成されたエージェントの数
COORD_AGENTS_TOP	BIGINT	coord_agents_top - コーディネーター・エージェント最大数
LOCKS_HELD	BIGINT	locks_held - ロック保持数
LOCK_WAITS	BIGINT	lock_waits - ロック待機数
LOCK_WAIT_TIME	BIGINT	lock_wait_time - ロック待機中の時間
LOCK_LIST_IN_USE	BIGINT	lock_list_in_use - 使用中のロック・リスト・メモリーの合計
DEADLOCKS	BIGINT	deadlocks - デッドロック検出数
LOCK_ESCALS	BIGINT	lock_escals - ロック・エスカレーション数
X_LOCK_ESCALS	BIGINT	x_lock_escals - 排他ロック・エスカレーション数
LOCKS_WAITING	BIGINT	locks_waiting - ロックで待機中の現行エージェント
LOCK_TIMEOUTS	BIGINT	lock_timeouts - ロック・タイムアウト数

表 297. SNAP_GET_DB_V91 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
NUM_INDOUBT_TRANS	BIGINT	num_indoubt_trans - 未確定トランザクション数
SORT_HEAP_ALLOCATED	BIGINT	sort_heap_allocated - 割り振られたソート・ヒープの合計
SORT_SHRHEAP_ALLOCATED	BIGINT	sort_shrheap_allocated - 現在割り振られているソート共有ヒープ
SORT_SHRHEAP_TOP	BIGINT	sort_shrheap_top - ソート共有ヒープの最高水準点
POST_SHRTHRESHOLD_SORTS	BIGINT	post_shrthreshold_sorts - ポスト共有しきい値ソート
TOTAL_SORTS	BIGINT	total_sorts - ソート合計
TOTAL_SORT_TIME	BIGINT	total_sort_time - ソート時間合計
SORT_OVERFLOWS	BIGINT	sort_overflows - ソート・オーバーフロー
ACTIVE_SORTS	BIGINT	active_sorts - アクティブ・ソート
POOL_DATA_L_READS	BIGINT	pool_data_l_reads - バッファー・プール・データの論理読み取り
POOL_DATA_P_READS	BIGINT	pool_data_p_reads - バッファー・プール・データの物理読み取り
POOL_TEMP_DATA_L_READS	BIGINT	pool_temp_data_l_reads - バッファー・プール一時データの論理読み取り
POOL_TEMP_DATA_P_READS	BIGINT	pool_temp_data_p_reads - バッファー・プール一時データの物理読み取り
POOL_ASYNC_DATA_READS	BIGINT	pool_async_data_reads - バッファー・プール非同期データ読み取り
POOL_DATA_WRITES	BIGINT	pool_data_writes - バッファー・プールへのデータの書き込み
POOL_ASYNC_DATA_WRITES	BIGINT	pool_async_data_writes - バッファー・プール非同期データ書き込み
POOL_INDEX_L_READS	BIGINT	pool_index_l_reads - バッファー・プール索引の論理読み取り
POOL_INDEX_P_READS	BIGINT	pool_index_p_reads - バッファー・プール索引の物理読み取り
POOL_TEMP_INDEX_L_READS	BIGINT	pool_temp_index_l_reads - バッファー・プール一時索引の論理読み取り
POOL_TEMP_INDEX_P_READS	BIGINT	pool_temp_index_p_reads - バッファー・プール一時索引の物理読み取り
POOL_ASYNC_INDEX_READS	BIGINT	pool_async_index_reads - バッファー・プール非同期索引読み取り
POOL_INDEX_WRITES	BIGINT	pool_index_writes - バッファー・プール索引の書き込み

表 297. SNAP_GET_DB_V91 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
POOL_ASYNC_INDEX_WRITES	BIGINT	pool_async_index_writes - バッファー・プール非同期索引書き込み
POOL_XDA_P_READS	BIGINT	pool_xda_p_reads - バッファー・プール XDA データの物理読み取り
POOL_XDA_L_READS	BIGINT	pool_xda_l_reads - バッファー・プール XDA データの論理読み取り
POOL_XDA_WRITES	BIGINT	pool_xda_writes - バッファー・プール XDA データの書き込み
POOL_ASYNC_XDA_READS	BIGINT	pool_async_xda_reads - バッファー・プール非同期 XDA データ読み取り
POOL_ASYNC_XDA_WRITES	BIGINT	pool_async_xda_writes - バッファー・プール非同期 XDA データ書き込み
POOL_TEMP_XDA_P_READS	BIGINT	pool_temp_xda_p_reads - バッファー・プール一時 XDA データの物理読み取り : モニター・エレメント
POOL_TEMP_XDA_L_READS	BIGINT	pool_temp_xda_l_reads - バッファー・プール一時 XDA データの論理読み取り
POOL_READ_TIME	BIGINT	pool_read_time - バッファー・プール物理読み取り時間の合計
POOL_WRITE_TIME	BIGINT	pool_write_time - バッファー・プール物理書き込み時間の合計
POOL_ASYNC_READ_TIME	BIGINT	pool_async_read_time - バッファー・プール非同期読み取り時間
POOL_ASYNC_WRITE_TIME	BIGINT	pool_async_write_time - バッファー・プール非同期書き込み時間
POOL_ASYNC_DATA_READ_REQS	BIGINT	pool_async_data_read_reqs - バッファー・プール非同期読み取り要求
POOL_ASYNC_INDEX_READ_REQS	BIGINT	pool_async_index_read_reqs - バッファー・プール非同期索引読み取り要求
POOL_ASYNC_XDA_READ_REQS	BIGINT	pool_async_xda_read_reqs - バッファー・プール非同期 XDA 読み取り要求
POOL_NO_VICTIM_BUFFER	BIGINT	pool_no_victim_buffer - バッファー・プールの非ビクティム・バッファー数
POOL_LSN_GAP_CLNS	BIGINT	pool_lsn_gap_clns - 起動されたバッファー・プール・ログ・スペース・クリーナー

表 297. SNAP_GET_DB_V91 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
POOL_DRTY_PG_STEAL_CLNS	BIGINT	pool_drty_pg_steal_clns - 起動されたバッファー・プール・ビクティム・ページ・クリーナー
POOL_DRTY_PG_THRSH_CLNS	BIGINT	pool_drty_pg_thrsh_clns - 起動されたバッファー・プールしきい値クリーナー
PREFETCH_WAIT_TIME	BIGINT	prefetch_wait_time - プリフェッチ待ち時間
UNREAD_PREFETCH_PAGES	BIGINT	unread_prefetch_pages - 読み取り不能プリフェッチ・ページ
DIRECT_READS	BIGINT	direct_reads - データベースからの直接読み取り
DIRECT_WRITES	BIGINT	direct_writes - データベースへの直接書き込み
DIRECT_READ_REQS	BIGINT	direct_read_reqs - 直接読み取り要求
DIRECT_WRITE_REQS	BIGINT	direct_write_reqs - 直接書き込み要求
DIRECT_READ_TIME	BIGINT	direct_read_time - 直接読み取り時間
DIRECT_WRITE_TIME	BIGINT	direct_write_time - 直接書き込み時間
FILES_CLOSED	BIGINT	files_closed - 閉じられたデータベース・ファイル
ELAPSED_EXEC_TIME_S	BIGINT	elapsed_exec_time - ステートメント実行経過時間 (秒単位)*
ELAPSED_EXEC_TIME_MS	BIGINT	elapsed_exec_time - ステートメント実行経過時間 (小数部、マイクロ秒単位)*
COMMIT_SQL_STMTS	BIGINT	commit_sql_stmts - 試行されたコミット・ステートメント
ROLLBACK_SQL_STMTS	BIGINT	rollback_sql_stmts - 試行されたロールバック・ステートメント
DYNAMIC_SQL_STMTS	BIGINT	dynamic_sql_stmts - 試行された動的 SQL ステートメント
STATIC_SQL_STMTS	BIGINT	static_sql_stmts - 試行された静的 SQL ステートメント
FAILED_SQL_STMTS	BIGINT	failed_sql_stmts - 失敗したステートメント操作
SELECT_SQL_STMTS	BIGINT	select_sql_stmts - 実行された選択 SQL ステートメント
UID_SQL_STMTS	BIGINT	uid_sql_stmts - 実行された更新/挿入/削除 SQL ステートメント

表 297. SNAP_GET_DB_V91 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
DDL_SQL_STMTS	BIGINT	ddl_sql_stmts - データ定義言語 (DDL) SQL ステートメント
INT_AUTO_REBINDS	BIGINT	int_auto_rebinds - 内部自動再バインド
INT_ROWS_DELETED	BIGINT	int_rows_deleted - 削除された内部行数
INT_ROWS_INSERTED	BIGINT	int_rows_inserted - 挿入された内部行数
INT_ROWS_UPDATED	BIGINT	int_rows_updated - 更新された内部行数
INT_COMMITS	BIGINT	int_commits - 内部コミット数
INT_ROLLBACKS	BIGINT	int_rollbacks - 内部ロールバック数
INT_DEADLOCK_ROLLBACKS	BIGINT	int_deadlock_rollbacks - デッドロックによる内部ロールバック数
ROWS_DELETED	BIGINT	rows_deleted - 削除行数
ROWS_INSERTED	BIGINT	rows_inserted - 挿入行数
ROWS_UPDATED	BIGINT	rows_updated - 更新行数
ROWS_SELECTED	BIGINT	rows_selected - 選択行数
ROWS_READ	BIGINT	rows_read - 読み取り行数
BINDS_PRECOMPILES	BIGINT	binds_precompiles - 試行されたバインド/プリコンパイル
TOTAL_LOG_AVAILABLE	BIGINT	total_log_available - 使用可能なログ合計
TOTAL_LOG_USED	BIGINT	total_log_used - 使用されているログ・スペースの合計
SEC_LOG_USED_TOP	BIGINT	sec_log_used_top - 使用された最大 2 次ログ・スペース
TOT_LOG_USED_TOP	BIGINT	tot_log_used_top - 使用された最大合計ログ・スペース
SEC_LOGS_ALLOCATED	BIGINT	sec_logs_allocated - 現在割り振られている 2 次ログ
LOG_READS	BIGINT	log_reads - 読み取られたログ・ページの数
LOG_READ_TIME_S	BIGINT	log_read_time - ログ読み取り時間 (秒単位)†
LOG_READ_TIME_NS	BIGINT	log_read_time - ログ読み取り時間 (小数部、ナノ秒単位)†
LOG_WRITES	BIGINT	log_writes - 書き込まれたログ・ページの数
LOG_WRITE_TIME_S	BIGINT	log_write_time - ログ書き込み時間 (秒単位)†

表 297. SNAP_GET_DB_V91 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
LOG_WRITE_TIME_NS	BIGINT	log_write_time - ログ書き込み時間 (小数部、ナノ秒単位)†
NUM_LOG_WRITE_IO	BIGINT	num_log_write_io - ログ書き込み数
NUM_LOG_READ_IO	BIGINT	num_log_read_io - ログ読み取り数
NUM_LOG_PART_PAGE_IO	BIGINT	num_log_part_page_io - 部分ログ・ページ書き込み数
NUM_LOG_BUFFER_FULL	BIGINT	num_log_buffer_full - フル・ログ・バッファーの回数
NUM_LOG_DATA_FOUND_IN_BUFFER	BIGINT	num_log_data_found_in_buffer - ログ・データがバッファーにある回数
APPL_ID_OLDEST_XACT	BIGINT	appl_id_oldest_xact - 最も古いトランザクションを使用するアプリケーション
LOG_TO_REDO_FOR_RECOVERY	BIGINT	log_to_redo_for_recovery - リカバリーの場合に再実行されるログの量
LOG_HELD_BY_DIRTY_PAGES	BIGINT	log_held_by_dirty_pages - ダーティー・ページ別に計算されるログ・スペースの量
PKG_CACHE_LOOKUPS	BIGINT	pkg_cache_lookups - パッケージ・キャッシュ参照
PKG_CACHE_INSERTS	BIGINT	pkg_cache_inserts - パッケージ・キャッシュ挿入
PKG_CACHE_NUM_OVERFLOWS	BIGINT	pkg_cache_num_overflows - パッケージ・キャッシュ・オーバーフロー数
PKG_CACHE_SIZE_TOP	BIGINT	pkg_cache_size_top - パッケージ・キャッシュ最高水準点
APPL_SECTION_LOOKUPS	BIGINT	appl_section_lookups - セクションの参照回数
APPL_SECTION_INSERTS	BIGINT	appl_section_inserts - セクション挿入数
CAT_CACHE_LOOKUPS	BIGINT	cat_cache_lookups - カタログ・キャッシュ参照数
CAT_CACHE_INSERTS	BIGINT	cat_cache_inserts - カタログ・キャッシュ挿入数
CAT_CACHE_OVERFLOWS	BIGINT	cat_cache_overflows - カタログ・キャッシュ・オーバーフロー数
CAT_CACHE_SIZE_TOP	BIGINT	cat_cache_size_top - カタログ・キャッシュ最高水準点
PRIV_WORKSPACE_SIZE_TOP	BIGINT	priv_workspace_size_top - 専用ワークスペースの最大サイズ

表 297. SNAP_GET_DB_V91 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
PRIV_WORKSPACE_NUM_OVERFLOWS	BIGINT	priv_workspace_num_overflows - 専用ワークスペースのオーバーフロー回数
PRIV_WORKSPACE_SECTION_INSERTS	BIGINT	priv_workspace_section_inserts - 専用ワークスペース・セクション挿入
PRIV_WORKSPACE_SECTION_LOOKUPS	BIGINT	priv_workspace_section_lookups - 専用ワークスペース・セクションの参照
SHR_WORKSPACE_SIZE_TOP	BIGINT	shr_workspace_size_top - 最大共有ワークスペース・サイズ
SHR_WORKSPACE_NUM_OVERFLOWS	BIGINT	shr_workspace_num_overflows - 共有ワークスペースのオーバーフロー回数
SHR_WORKSPACE_SECTION_INSERTS	BIGINT	shr_workspace_section_inserts - 共有ワークスペース・セクション挿入数
SHR_WORKSPACE_SECTION_LOOKUPS	BIGINT	shr_workspace_section_lookups - 共有ワークスペース・セクションの参照回数
TOTAL_HASH_JOINS	BIGINT	total_hash_joins - ハッシュ結合の合計
TOTAL_HASH_LOOPS	BIGINT	total_hash_loops - ハッシュ・ループの合計
HASH_JOIN_OVERFLOWS	BIGINT	hash_join_overflows - ハッシュ結合のオーバーフロー
HASH_JOIN_SMALL_OVERFLOWS	BIGINT	hash_join_small_overflows - ハッシュ結合の短精度オーバーフロー
POST_SHRTHRESHOLD_HASH_JOINS	BIGINT	post_shrthreshold_hash_joins - ポストしきい値ハッシュ結合
ACTIVE_HASH_JOINS	BIGINT	active_hash_joins - アクティブ・ハッシュ結合
NUM_DB_STORAGE_PATHS	BIGINT	num_db_storage_paths - 自動ストレージ・パスの数
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。
SMALLEST_LOG_AVAIL_NODE	INTEGER	smallest_log_avail_node - 使用可能なログ・スペースが最小のノード

表 297. SNAP_GET_DB_V91 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
* この列の元になるモニター・エレメントの合計消費時間を計算するには、このモニター・エレメントに関する _S で終わっている列で報告されている整数秒と、このモニター・エレメントに関する _MS で終わっている列に報告されている小数秒を、次の式を使用して加算しなければなりません: (*monitor-element-name*_S × 1,000,000 + *monitor-element-name*_MS) ÷ 1,000,000. 例えば、(ELAPSED_EXEC_TIME_S × 1,000,000 + ELAPSED_EXEC_TIME_MS) ÷ 1,000,000。 †このモニター・エレメントの合計経過時間を計算するには、このモニター・エレメントに関する _S で終わっている列で報告されている整数秒と、このモニター・エレメントに関する _NS で終わっている列に報告されている小数秒を、次の式を使用して加算しなければなりません: (*monitor-element-name*_S × 1,000,000,000 + *monitor-element-name*_NS) ÷ 1,000,000,000。例えば、(LOG_READ_TIME_S × 1,000,000,000 + LOG_READ_TIME_NS) ÷ 1,000,000,000。		

SNAPDB 管理ビューおよび SNAP_GET_DB_V95 表関数 - dbase 論理グループからのスナップショット情報の検索

注: SNAP_GET_DB_V95 表関数は非推奨になり、 SNAP_GET_DB_V97 表関数 (dbase 論理グループからのスナップショット情報の取得) に置き換わりました。

828 ページの『SNAPDB 管理ビュー』と 830 ページの『SNAP_GET_DB_V95 表関数』は、データベース (dbase) 論理グループからのスナップショット情報を戻します。

SNAPDB 管理ビュー

この管理ビューを使用すると、現在接続されているデータベースに関するスナップショット情報を dbase 論理グループから検索できます。

SNAPDB 管理ビューを SNAPDB_MEMORY_POOL、 SNAPDETAILLOG、SNAPHADR、および SNAPSTORAGE_PATHS 管理ビューと併せて使用することにより、GET SNAPSHOT FOR DATABASE on database-alias CLP コマンドと同等の情報を戻します。

スキーマは SYSIBMADM です。

戻される情報の完全なリストは、 832 ページの表 206 を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPDB 管理ビューに対する SELECT 特権
- SNAPDB 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_DB_V95 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベースのすべてのデータベース・パーティションに関する状況、プラットフォーム、ロケーション、および接続時間を取り出します。

```
SELECT SUBSTR(DB_NAME, 1, 20) AS DB_NAME, DB_STATUS, SERVER_PLATFORM,
   DB_LOCATION, DB_CONN_TIME, DBPARTITIONNUM
   FROM SYSIBMADM.SNAPDB ORDER BY DBPARTITIONNUM
```

以下はこの照会の出力例です。

```
DB_NAME     DB_STATUS    SERVER_PLATFORM DB_LOCATION  ...
-------...- ------------ --------------- ------------ ...
TEST        ACTIVE       AIX64           LOCAL        ...
TEST        ACTIVE       AIX64           LOCAL        ...
TEST        ACTIVE       AIX64           LOCAL        ...

3 record(s) selected.
```

この照会からの出力 (続き)。

```
... DB_CONN_TIME               DBPARTITIONNUM
... -------------------------- --------------
... 2006-01-08-16.48.30.665477              0
... 2006-01-08-16.48.34.005328              1
... 2006-01-08-16.48.34.007937              2
```

このルーチンは、コマンド行で以下を呼び出すことにより使用できます。

```
SELECT TOTAL_OLAP_FUNCS, OLAP_FUNC_OVERFLOWS, ACTIVE_OLAP_FUNCS
   FROM SYSIBMADM.SNAPDB

TOTAL_OLAP_FUNCS     OLAP_FUNC_OVERFLOWS  ACTIVE_OLAP_FUNCS
-------------------- -------------------- -----------------
                   7                    2                 1

  1 record(s) selected.
```

ワークロードの実行後に、ユーザーは次の照会を使用できます。

```
SELECT STATS_CACHE_SIZE, STATS_FABRICATIONS, SYNC_RUNSTATS,
   ASYNC_RUNSTATS, STATS_FABRICATE_TIME, SYNC_RUNSTATS_TIME
   FROM SYSIBMADM.SNAPDB

STATS_CACHE_SIZE STATS_FABRICATIONS SYNC_RUNSTATS ASYNC_RUNSTATS ...
---------------- ------------------ ------------- -------------- ...
             128                  2             1              0 ...


... STATS_FABRICATE_TIME SYNC_RUNSTATS_TIME
```

```
... -------------------- ------------------
...                   10                100

1 record(s) selected.
```

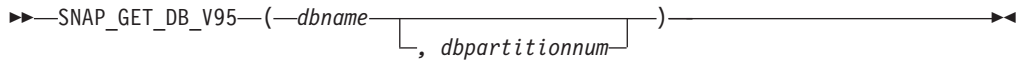
SNAP_GET_DB_V95 表関数

SNAP_GET_DB_V95 表関数は、SNAPDB 管理ビューと同じ情報を戻します。

SNAP_GET_DB_V95 表関数を SNAP_GET_DB_MEMORY_POOL、SNAP_GET_DETAILLOG_V91、 SNAP_GET_HADR、および SNAP_GET_STORAGE_PATHS 表関数と併せて使用することにより、GET SNAPSHOT FOR ALL DATABASES CLP コマンドと同等の情報を戻します。

戻される情報の完全なリストは、 832 ページの表 206 を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、空ストリングを指定します。現在接続されているデータベースと同じインスタンス内のすべてのデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプを持つファイルが存在しない場合には、現在接続されているデータベースのスナップショットとデータベース・パーティション番号を

SNAP_GET_DB_V95 表関数が取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_DB_V95 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

例 1: 現在接続されているデータベースのすべてのデータベース・パーティションに渡る集約ビューとして、状況、プラットフォーム、ロケーション、および接続時間を取り出します。

```
SELECT SUBSTR(DB_NAME, 1, 20) AS DB_NAME, DB_STATUS, SERVER_PLATFORM,
   DB_LOCATION, DB_CONN_TIME FROM TABLE(SNAP_GET_DB_V95('', -2)) AS T
```

以下はこの照会の出力例です。

```
DB_NAME     DB_STATUS    SERVER_PLATFORM ...
-------...- ------------ --------------- ...
SAMPLE      ACTIVE       AIX64           ...

  1 record(s) selected.
```

この照会からの出力 (続き)。

```
... DB_LOCATION  DB_CONN_TIME
... ------------ --------------------------
... LOCAL        2005-07-24-22.09.22.013196
```

例 2: 現在接続されているデータベースを含む同じインスタンス内にあるすべてのアクティブ・データベースのすべてのデータベース・パーティションに渡る集約ビューとして、状況、プラットフォーム、ロケーション、および接続時間を取り出します。

```
SELECT SUBSTR(DB_NAME, 1, 20) AS DB_NAME, DB_STATUS, SERVER_PLATFORM,
   DB_LOCATION, DB_CONN_TIME
   FROM TABLE(SNAP_GET_DB_V95(CAST (NULL AS VARCHAR(128)), -2)) AS T
```

以下はこの照会の出力例です。

```
DB_NAME      DB_STATUS    SERVER_PLATFORM ...
--------...- ------------ --------------- ...
TOOLSDB      ACTIVE       AIX64           ...
SAMPLE       ACTIVE       AIX64           ...
```

この照会からの出力 (続き)。

```
... DB_LOCATION  DB_CONN_TIME
... ------------ --------------------------
... LOCAL        2005-07-24-22.26.54.396335
... LOCAL        2005-07-24-22.09.22.013196
```

例 3: このルーチンは、データベースへの接続時にコマンド行で以下を呼び出すことにより使用できます。

```
SELECT TOTAL_OLAP_FUNCS, OLAP_FUNC_OVERFLOWS, ACTIVE_OLAP_FUNCS
   FROM TABLE (SNAP_GET_DB_V95('', 0)) AS T
```

出力は次のようになります。

```
TOTAL_OLAP_FUNCS  OLAP_FUNC_OVERFLOWS  ACTIVE_OLAP_FUNCS
----------------  --------------------  --------------------
               7                     2                     1

1 record(s) selected.
```

例 4: ワークロードの実行後に、ユーザーは次の照会を表関数とともに使用できます。

```
 SELECT STATS_CACHE_SIZE, STATS_FABRICATIONS, SYNC_RUNSTATS,
   ASYNC_RUNSTATS, STATS_FABRICATE_TIME, SYNC_RUNSTATS_TIME
   FROM TABLE (SNAP_GET_DB_V95('mytestdb', -1)) AS SNAPDB

STATS_CACHE_SIZE STATS_FABRICATIONS SYNC_RUNSTATS ASYNC_RUNSTATS ...
---------------- ------------------ ------------- -------------- ...
             200                  1             2              0 ...

Continued

...STATS_FABRICATE_TIME  SYNC_RUNSTATS_TIME
...--------------------  ------------------
...                    2                  32

  1 record(s) selected.
```

SNAPDB 管理ビューおよび SNAP_GET_DB_V95 表関数のメタデータ

表 298. SNAPDB 管理ビューおよび SNAP_GET_DB_V95 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
DB_NAME	VARCHAR(128)	db_name - データベース名
DB_PATH	VARCHAR(1024)	db_path - データベース・パス
INPUT_DB_ALIAS	VARCHAR(128)	input_db_alias - 入力データベース別名
DB_STATUS	VARCHAR(12)	db_status - データベース状況。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • ACTIVE • QUIESCE_PEND • QUIESCED • ROLLFWD • ACTIVE_STANDBY • STANDBY

表 298. SNAPDB 管理ビューおよび SNAP_GET_DB_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
CATALOG_PARTITION	SMALLINT	catalog_node - カタログ・ノード番号
CATALOG_PARTITION_NAME	VARCHAR(128)	catalog_node_name - カタログ・ノード・ネットワーク名
SERVER_PLATFORM	VARCHAR(12)	server_platform - サーバーのオペレーティング・システム。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • AIX • AIX64 • AS400_DRDA • DOS • DYNIX • HP • HP64 • HPIA • HPIA64 • LINUX • LINUX390 • LINUXIA64 • LINUXPPC • LINUXPPC64 • LINUXX8664 • LINUXZ64 • MAC • MVS_DRDA • NT • NT64 • OS2 • OS390 • SCO • SGI • SNI • SUN • SUN64 • UNKNOWN • UNKNOWN_DRDA • VM_DRDA • VSE_DRDA • WINDOWS

表 298. SNAPDB 管理ビューおよび SNAP_GET_DB_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
DB_LOCATION	VARCHAR(12)	db_location - データベース・ロケーション。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • LOCAL • REMOTE
DB_CONN_TIME	TIMESTAMP	db_conn_time - データベース活動化タイム・スタンプ
LAST_RESET	TIMESTAMP	last_reset - 最後のリセット・タイム・スタンプ
LAST_BACKUP	TIMESTAMP	last_backup - 最終バックアップ・タイム・スタンプ
CONNECTIONS_TOP	BIGINT	connections_top - 同時接続の最大数
TOTAL_CONS	BIGINT	total_cons - データベース活動化以降の接続
TOTAL_SEC_CONS	BIGINT	total_sec_cons - 2 次接続
APPLS_CUR_CONS	BIGINT	appls_cur_cons - 現在接続されているアプリケーション
APPLS_IN_DB2	BIGINT	appls_in_db2 - データベースで現在実行中のアプリケーション
NUM_ASSOC_AGENTS	BIGINT	num_assoc_agents - 関連したエージェント数
AGENTS_TOP	BIGINT	agents_top - 作成されたエージェントの数
COORD_AGENTS_TOP	BIGINT	coord_agents_top - コーディネーター・エージェント最大数
LOCKS_HELD	BIGINT	locks_held - ロック保持数
LOCK_WAITS	BIGINT	lock_waits - ロック待機数
LOCK_WAIT_TIME	BIGINT	lock_wait_time - ロック待機中の時間
LOCK_LIST_IN_USE	BIGINT	lock_list_in_use - 使用中のロック・リスト・メモリーの合計
DEADLOCKS	BIGINT	deadlocks - デッドロック検出数
LOCK_ESCALS	BIGINT	lock_escals - ロック・エスカレーション数
X_LOCK_ESCALS	BIGINT	x_lock_escals - 排他ロック・エスカレーション数
LOCKS_WAITING	BIGINT	locks_waiting - ロックで待機中の現行エージェント
LOCK_TIMEOUTS	BIGINT	lock_timeouts - ロック・タイムアウト数
NUM_INDOUBT_TRANS	BIGINT	num_indoubt_trans - 未確定トランザクション数
SORT_HEAP_ALLOCATED	BIGINT	sort_heap_allocated - 割り振られたソート・ヒープの合計
SORT_SHRHEAP_ALLOCATED	BIGINT	sort_shrheap_allocated - 現在割り振られているソート共有ヒープ
SORT_SHRHEAP_TOP	BIGINT	sort_shrheap_top - ソート共有ヒープの最高水準点

表 298. SNAPDB 管理ビューおよび SNAP_GET_DB_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
POST_SHRTHRESHOLD_SORTS	BIGINT	post_shrthreshold_sorts - ポスト共有しきい値ソート
TOTAL_SORTS	BIGINT	total_sorts - ソート合計
TOTAL_SORT_TIME	BIGINT	total_sort_time - ソート時間合計
SORT_OVERFLOWS	BIGINT	sort_overflows - ソート・オーバーフロー
ACTIVE_SORTS	BIGINT	active_sorts - アクティブ・ソート
POOL_DATA_L_READS	BIGINT	pool_data_l_reads - バッファー・プール・データの論理読み取り
POOL_DATA_P_READS	BIGINT	pool_data_p_reads - バッファー・プール・データの物理読み取り
POOL_TEMP_DATA_L_READS	BIGINT	pool_temp_data_l_reads - バッファー・プール一時データの論理読み取り
POOL_TEMP_DATA_P_READS	BIGINT	pool_temp_data_p_reads - バッファー・プール一時データの物理読み取り
POOL_ASYNC_DATA_READS	BIGINT	pool_async_data_reads - バッファー・プール非同期データ読み取り
POOL_DATA_WRITES	BIGINT	pool_data_writes - バッファー・プールへのデータの書き込み
POOL_ASYNC_DATA_WRITES	BIGINT	pool_async_data_writes - バッファー・プール非同期データ書き込み
POOL_INDEX_L_READS	BIGINT	pool_index_l_reads - バッファー・プール索引の論理読み取り
POOL_INDEX_P_READS	BIGINT	pool_index_p_reads - バッファー・プール索引の物理読み取り
POOL_TEMP_INDEX_L_READS	BIGINT	pool_temp_index_l_reads - バッファー・プール一時索引の論理読み取り
POOL_TEMP_INDEX_P_READS	BIGINT	pool_temp_index_p_reads - バッファー・プール一時索引の物理読み取り
POOL_ASYNC_INDEX_READS	BIGINT	pool_async_index_reads - バッファー・プール非同期索引読み取り
POOL_INDEX_WRITES	BIGINT	pool_index_writes - バッファー・プール索引の書き込み
POOL_ASYNC_INDEX_WRITES	BIGINT	pool_async_index_writes - バッファー・プール非同期索引書き込み
POOL_XDA_P_READS	BIGINT	pool_xda_p_reads - バッファー・プール XDA データの物理読み取り
POOL_XDA_L_READS	BIGINT	pool_xda_l_reads - バッファー・プール XDA データの論理読み取り
POOL_XDA_WRITES	BIGINT	pool_xda_writes - バッファー・プール XDA データの書き込み
POOL_ASYNC_XDA_READS	BIGINT	pool_async_xda_reads - バッファー・プール非同期 XDA データ読み取り

表 298. SNAPDB 管理ビューおよび SNAP_GET_DB_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
POOL_ASYNC_XDA_WRITES	BIGINT	pool_async_xda_writes - バッファー・プール非同期 XDA データ書き込み
POOL_TEMP_XDA_P_READS	BIGINT	pool_temp_xda_p_reads - バッファー・プール一時 XDA データの物理読み取り : モニター・エレメント
POOL_TEMP_XDA_L_READS	BIGINT	pool_temp_xda_l_reads - バッファー・プール一時 XDA データの論理読み取り
POOL_READ_TIME	BIGINT	pool_read_time - バッファー・プール物理読み取り時間の合計
POOL_WRITE_TIME	BIGINT	pool_write_time - バッファー・プール物理書き込み時間の合計
POOL_ASYNC_READ_TIME	BIGINT	pool_async_read_time - バッファー・プール非同期読み取り時間
POOL_ASYNC_WRITE_TIME	BIGINT	pool_async_write_time - バッファー・プール非同期書き込み時間
POOL_ASYNC_DATA_ READ_REQS	BIGINT	pool_async_data_read_reqs - バッファー・プール非同期読み取り要求
POOL_ASYNC_INDEX_ READ_REQS	BIGINT	pool_async_index_read_reqs - バッファー・プール非同期索引読み取り要求
POOL_ASYNC_XDA_ READ_REQS	BIGINT	pool_async_xda_read_reqs - バッファー・プール非同期 XDA 読み取り要求
POOL_NO_VICTIM_BUFFER	BIGINT	pool_no_victim_buffer - バッファー・プールの非ビクティム・バッファー数
POOL_LSN_GAP_CLNS	BIGINT	pool_lsn_gap_clns - 起動されたバッファー・プール・ログ・スペース・クリーナー
POOL_DRTY_PG_STEAL_CLNS	BIGINT	pool_drty_pg_steal_clns - 起動されたバッファー・プール・ビクティム・ページ・クリーナー
POOL_DRTY_PG_THRSH_CLNS	BIGINT	pool_drty_pg_thrsh_clns - 起動されたバッファー・プールしきい値クリーナー
PREFETCH_WAIT_TIME	BIGINT	prefetch_wait_time - プリフェッチ待ち時間
UNREAD_PREFETCH_PAGES	BIGINT	unread_prefetch_pages - 読み取り不能プリフェッチ・ページ
DIRECT_READS	BIGINT	direct_reads - データベースからの直接読み取り
DIRECT_WRITES	BIGINT	direct_writes - データベースへの直接書き込み
DIRECT_READ_REQS	BIGINT	direct_read_reqs - 直接読み取り要求
DIRECT_WRITE_REQS	BIGINT	direct_write_reqs - 直接書き込み要求
DIRECT_READ_TIME	BIGINT	direct_read_time - 直接読み取り時間
DIRECT_WRITE_TIME	BIGINT	direct_write_time - 直接書き込み時間

表 298. SNAPDB 管理ビューおよび SNAP_GET_DB_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
FILES_CLOSED	BIGINT	files_closed - 閉じられたデータベース・ファイル
ELAPSED_EXEC_TIME_S	BIGINT	elapsed_exec_time - ステートメント実行経過時間 (秒単位)*
ELAPSED_EXEC_TIME_MS	BIGINT	elapsed_exec_time - ステートメント実行経過時間 (小数部、マイクロ秒単位)*
COMMIT_SQL_STMTS	BIGINT	commit_sql_stmts - 試行されたコミット・ステートメント
ROLLBACK_SQL_STMTS	BIGINT	rollback_sql_stmts - 試行されたロールバック・ステートメント
DYNAMIC_SQL_STMTS	BIGINT	dynamic_sql_stmts - 試行された動的 SQL ステートメント
STATIC_SQL_STMTS	BIGINT	static_sql_stmts - 試行された静的 SQL ステートメント
FAILED_SQL_STMTS	BIGINT	failed_sql_stmts - 失敗したステートメント操作
SELECT_SQL_STMTS	BIGINT	select_sql_stmts - 実行された選択 SQL ステートメント
UID_SQL_STMTS	BIGINT	uid_sql_stmts - 実行された更新/挿入/削除 SQL ステートメント
DDL_SQL_STMTS	BIGINT	ddl_sql_stmts - データ定義言語 (DDL) SQL ステートメント
INT_AUTO_REBINDS	BIGINT	int_auto_rebinds - 内部自動再バインド
INT_ROWS_DELETED	BIGINT	int_rows_deleted - 削除された内部行数
INT_ROWS_INSERTED	BIGINT	int_rows_inserted - 挿入された内部行数
INT_ROWS_UPDATED	BIGINT	int_rows_updated - 更新された内部行数
INT_COMMITS	BIGINT	int_commits - 内部コミット数
INT_ROLLBACKS	BIGINT	int_rollbacks - 内部ロールバック数
INT_DEADLOCK_ROLLBACKS	BIGINT	int_deadlock_rollbacks - デッドロックによる内部ロールバック数
ROWS_DELETED	BIGINT	rows_deleted - 削除行数
ROWS_INSERTED	BIGINT	rows_inserted - 挿入行数
ROWS_UPDATED	BIGINT	rows_updated - 更新行数
ROWS_SELECTED	BIGINT	rows_selected - 選択行数
ROWS_READ	BIGINT	rows_read - 読み取り行数
BINDS_PRECOMPILES	BIGINT	binds_precompiles - 試行されたバインド/プリコンパイル
TOTAL_LOG_AVAILABLE	BIGINT	total_log_available - 使用可能なログ合計
TOTAL_LOG_USED	BIGINT	total_log_used - 使用されているログ・スペースの合計
SEC_LOG_USED_TOP	BIGINT	sec_log_used_top - 使用された最大 2 次ログ・スペース

表 298. SNAPDB 管理ビューおよび SNAP_GET_DB_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
TOT_LOG_USED_TOP	BIGINT	tot_log_used_top - 使用された最大合計ログ・スペース
SEC_LOGS_ALLOCATED	BIGINT	sec_logs_allocated - 現在割り振られている 2 次ログ
LOG_READS	BIGINT	log_reads - 読み取られたログ・ページの数
LOG_READ_TIME_S	BIGINT	log_read_time - ログ読み取り時間 (秒単位)†
LOG_READ_TIME_NS	BIGINT	log_read_time - ログ読み取り時間 (小数部、ナノ秒単位)†
LOG_WRITES	BIGINT	log_writes - 書き込まれたログ・ページの数
LOG_WRITE_TIME_S	BIGINT	log_write_time - ログ書き込み時間 (秒単位)†
LOG_WRITE_TIME_NS	BIGINT	log_write_time - ログ書き込み時間 (小数部、ナノ秒単位)†
NUM_LOG_WRITE_IO	BIGINT	num_log_write_io - ログ書き込み数
NUM_LOG_READ_IO	BIGINT	num_log_read_io - ログ読み取り数
NUM_LOG_PART_PAGE_IO	BIGINT	num_log_part_page_io - 部分ログ・ページ書き込み数
NUM_LOG_BUFFER_FULL	BIGINT	num_log_buffer_full - フル・ログ・バッファーの回数
NUM_LOG_DATA_FOUND_IN_BUFFER	BIGINT	num_log_data_found_in_buffer - ログ・データがバッファーにある回数
APPL_ID_OLDEST_XACT	BIGINT	appl_id_oldest_xact - 最も古いトランザクションを使用するアプリケーション
LOG_TO_REDO_FOR_RECOVERY	BIGINT	log_to_redo_for_recovery - リカバリーの場合に再実行されるログの量
LOG_HELD_BY_DIRTY_PAGES	BIGINT	log_held_by_dirty_pages - ダーティー・ページ別に計算されるログ・スペースの量
PKG_CACHE_LOOKUPS	BIGINT	pkg_cache_lookups - パッケージ・キャッシュ参照
PKG_CACHE_INSERTS	BIGINT	pkg_cache_inserts - パッケージ・キャッシュ挿入
PKG_CACHE_NUM_OVERFLOWS	BIGINT	pkg_cache_num_overflows - パッケージ・キャッシュ・オーバーフロー数
PKG_CACHE_SIZE_TOP	BIGINT	pkg_cache_size_top - パッケージ・キャッシュ最高水準点
APPL_SECTION_LOOKUPS	BIGINT	appl_section_lookups - セクションの参照回数
APPL_SECTION_INSERTS	BIGINT	appl_section_inserts - セクション挿入数
CAT_CACHE_LOOKUPS	BIGINT	cat_cache_lookups - カタログ・キャッシュ参照数
CAT_CACHE_INSERTS	BIGINT	cat_cache_inserts - カタログ・キャッシュ挿入数

表 298. SNAPDB 管理ビューおよび SNAP_GET_DB_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
CAT_CACHE_OVERFLOWS	BIGINT	cat_cache_overflows - カタログ・キャッシュ・オーバーフロー数
CAT_CACHE_SIZE_TOP	BIGINT	cat_cache_size_top - カタログ・キャッシュ最高水準点
PRIV_WORKSPACE_SIZE_TOP	BIGINT	priv_workspace_size_top - 専用ワークスペースの最大サイズ
PRIV_WORKSPACE_NUM_OVERFLOWS	BIGINT	priv_workspace_num_overflows - 専用ワークスペースのオーバーフロー回数
PRIV_WORKSPACE_SECTION_INSERTS	BIGINT	priv_workspace_section_inserts - 専用ワークスペース・セクション挿入
PRIV_WORKSPACE_SECTION_LOOKUPS	BIGINT	priv_workspace_section_lookups - 専用ワークスペース・セクションの参照
SHR_WORKSPACE_SIZE_TOP	BIGINT	shr_workspace_size_top - 最大共有ワークスペース・サイズ
SHR_WORKSPACE_NUM_OVERFLOWS	BIGINT	shr_workspace_num_overflows - 共有ワークスペースのオーバーフロー回数
SHR_WORKSPACE_SECTION_INSERTS	BIGINT	shr_workspace_section_inserts - 共有ワークスペース・セクション挿入数
SHR_WORKSPACE_SECTION_LOOKUPS	BIGINT	shr_workspace_section_lookups - 共有ワークスペース・セクションの参照回数
TOTAL_HASH_JOINS	BIGINT	total_hash_joins - ハッシュ結合の合計
TOTAL_HASH_LOOPS	BIGINT	total_hash_loops - ハッシュ・ループの合計
HASH_JOIN_OVERFLOWS	BIGINT	hash_join_overflows - ハッシュ結合のオーバーフロー
HASH_JOIN_SMALL_OVERFLOWS	BIGINT	hash_join_small_overflows - ハッシュ結合の短精度オーバーフロー
POST_SHRTHRESHOLD_HASH_JOINS	BIGINT	post_shrthreshold_hash_joins - ポストしきい値ハッシュ結合
ACTIVE_HASH_JOINS	BIGINT	active_hash_joins - アクティブ・ハッシュ結合
NUM_DB_STORAGE_PATHS	BIGINT	num_db_storage_paths - 自動ストレージ・パスの数
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。
SMALLEST_LOG_AVAIL_NODE	INTEGER	smallest_log_avail_node - 使用可能なログ・スペースが最小のノード
TOTAL_OLAP_FUNCS	BIGINT	実行される OLAP 関数の合計数。

表 298. SNAPDB 管理ビューおよび SNAP_GET_DB_V95 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
OLAP_FUNC_OVERFLOWS	BIGINT	OLAP 関数データが使用可能なソート・ヒープ・スペースを超えた回数。
ACTIVE_OLAP_FUNCS	BIGINT	現在実行中で、ソート・ヒープ・メモリーを消費している OLAP 関数の合計数。
STATS_CACHE_SIZE	BIGINT	統計キャッシュのサイズ (バイト)。
STATS_FABRICATIONS	BIGINT	表または索引のスキャンを実行しないでシステムが統計を作成するための statistics-collect アクティビティーの合計数。
SYNC_RUNSTATS	BIGINT	照会コンパイル中の同期 statistics-collect アクティビティーの合計数。
ASYNC_RUNSTATS	BIGINT	この列の出力は、成功した非同期 statistics-collect アクティビティーの合計数に変更されます。
STATS_FABRICATE_TIME	BIGINT	照会コンパイル中に表または索引のスキャンを実行しないでシステムが統計を作成するのに費やされる合計時間 (ミリ秒)。
SYNC_RUNSTATS_TIME	BIGINT	同期 statistics-collect アクティビティーに費やされる合計時間 (ミリ秒)。
NUM_THRESHOLD_VIOLATIONS	BIGINT	データベースで発生したしきい値違反の数。
* この列の元になるモニター・エレメントの合計消費時間を計算するには、このモニター・エレメントに関する _S で終わっている列で報告されている整数秒と、このモニター・エレメントに関する _MS で終わっている列に報告されている小数秒を、次の式を使用して加算しなければなりません: (*monitor-element-name*_S × 1,000,000 + *monitor-element-name*_MS) ÷ 1,000,000. 例えば、(ELAPSED_EXEC_TIME_S × 1,000,000 + ELAPSED_EXEC_TIME_MS) ÷ 1,000,000。 †このモニター・エレメントの合計経過時間を計算するには、このモニター・エレメントに関する _S で終わっている列で報告されている整数秒と、このモニター・エレメントに関する _NS で終わっている列に報告されている小数秒を、次の式を使用して加算しなければなりません: (*monitor-element-name*_S × 1,000,000,000 + *monitor-element-name*_NS) ÷ 1,000,000,000。例えば、(LOG_READ_TIME_S × 1,000,000,000 + LOG_READ_TIME_NS) ÷ 1,000,000,000。		

SNAP_GET_DYN_SQL_V91 表関数 - dynsql 論理グループのスナップショット情報の検索

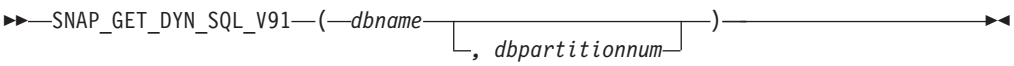
注: この表関数は使用すべきではなく、 704 ページの『SNAPDYN_SQL 管理ビューおよび SNAP_GET_DYN_SQL_V95 表関数 - dynsql 論理グループのスナップショット情報の検索』 に置き換えられました。

SNAP_GET_DYN_SQL_V91 表関数は、dynsql 論理データ・グループからのスナップショット情報を戻します。

この表関数は、GET SNAPSHOT FOR DYNAMIC SQL ON database-alias CLP コマンドと同等の情報を戻します。

戻される可能性のある情報の完全なリストは、 1245 ページの表 299を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL または空ストリングを指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプを持つファイルが存在しない場合には、現在接続されているデータベースのスナップショットとデータベース・パーティション番号を SNAP_GET_DYN_SQL_V91 表関数が取得します。

許可

- SYSMON 権限
- SNAP_GET_DYN_SQL_V91 表関数に対する EXECUTE 特権。

例

現在接続されているデータベースの現在接続されているデータベース・パーティションで実行される動的 SQL のリスト (読み取られる行の番号順に並んでいるもの) を検索します。

```
SELECT PREP_TIME_WORST, NUM_COMPILATIONS, SUBSTR(STMT_TEXT, 1, 60)
   AS STMT_TEXT FROM TABLE(SNAP_GET_DYN_SQL_V91('',-1)) as T
    ORDER BY ROWS_READ
```

以下はこの照会の出力例です。

```
PREP_TIME_WORST      ...
-------------------- ...
                   0 ...
                   3 ...
                     ...
                   4 ...
                     ...
                   4 ...
                     ...
                   4 ...
                     ...
                   3 ...
                     ...
                   4 ...
                     ...
```

この照会からの出力 (続き)。

```
... NUM_COMPILATIONS     STMT_TEXT
... -------------------- ----------------------------------------...-
...                    0 SET CURRENT LOCALE LC_CTYPE = 'en_US'
...                    1 select rows_read, rows_written,
...                          substr(stmt_text, 1, 40) as
...                    1 select * from table
...                          (snap_get_dyn_sqlv9('',-1)) as t
...                    1 select * from table
...                          (snap_getdetaillog9('',-1)) as t
...                    1 select * from table
...                          (snap_get_hadr('',-1)) as t
...                    1 select prep_time_worst, num_compilations,
...                          substr(stmt_text,
...                    1 select prep_time_worst, num_compilations,
...                          substr(stmt_text,
```

戻される情報

表 299. SNAP_GET_DYN_SQL_V91 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
NUM_EXECUTIONS	BIGINT	num_executions - ステートメント実行回数
NUM_COMPILATIONS	BIGINT	num_compilations - ステートメント・コンパイル数
PREP_TIME_WORST	BIGINT	prep_time_worst - ステートメント最長準備時間
PREP_TIME_BEST	BIGINT	prep_time_best - ステートメント最短準備時間
INT_ROWS_DELETED	BIGINT	int_rows_deleted - 削除された内部行数
INT_ROWS_INSERTED	BIGINT	int_rows_inserted - 挿入された内部行数
INT_ROWS_UPDATED	BIGINT	int_rows_updated - 更新された内部行数
ROWS_READ	BIGINT	rows_read - 読み取り行数

表 299. SNAP_GET_DYN_SQL_V91 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
ROWS_WRITTEN	BIGINT	rows_written - 書き込み行数
STMT_SORTS	BIGINT	stmt_sorts - ステートメント・ソート回数
SORT_OVERFLOWS	BIGINT	sort_overflows - ソート・オーバーフロー
TOTAL_SORT_TIME	BIGINT	total_sort_time - ソート時間合計
POOL_DATA_L_READS	BIGINT	pool_data_l_reads - バッファー・プール・データの論理読み取り
POOL_DATA_P_READS	BIGINT	pool_data_p_reads - バッファー・プール・データの物理読み取り
POOL_TEMP_DATA_L_READS	BIGINT	pool_temp_data_l_reads - バッファー・プール一時データの論理読み取り
POOL_TEMP_DATA_P_READS	BIGINT	pool_temp_data_p_reads - バッファー・プール一時データの物理読み取り
POOL_INDEX_L_READS	BIGINT	pool_index_l_reads - バッファー・プール索引の論理読み取り
POOL_INDEX_P_READS	BIGINT	pool_index_p_reads - バッファー・プール索引の物理読み取り
POOL_TEMP_INDEX_L_READS	BIGINT	pool_temp_index_l_reads - バッファー・プール一時索引の論理読み取り
POOL_TEMP_INDEX_P_READS	BIGINT	pool_temp_index_p_reads - バッファー・プール一時索引の物理読み取り
POOL_XDA_L_READS	BIGINT	pool_xda_l_reads - バッファー・プール XDA データの論理読み取り
POOL_XDA_P_READS	BIGINT	pool_xda_p_reads - バッファー・プール XDA データの物理読み取り
POOL_TEMP_XDA_L_READS	BIGINT	pool_temp_xda_l_reads - バッファー・プール一時 XDA データの論理読み取り
POOL_TEMP_XDA_P_READS	BIGINT	pool_temp_xda_p_reads - バッファー・プール一時 XDA データの物理読み取り : モニター・エレメント
TOTAL_EXEC_TIME	BIGINT	total_exec_time - ステートメント実行の経過時間 (秒単位)*
TOTAL_EXEC_TIME_MS	BIGINT	total_exec_time - ステートメント実行の経過時間 (小数部、マイクロ秒単位)*
TOTAL_USR_CPU_TIME	BIGINT	total_usr_cpu_time - ステートメントのユーザー CPU の合計 (秒単位)*

表 299. SNAP_GET_DYN_SQL_V91 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
TOTAL_USR_CPU_TIME_MS	BIGINT	total_usr_cpu_time - ステートメントのユーザー CPU の合計 (小数部、マイクロ秒単位)*
TOTAL_SYS_CPU_TIME	BIGINT	total_sys_cpu_time - ステートメントのシステム CPU の合計 (秒単位)*
TOTAL_SYS_CPU_TIME_MS	BIGINT	total_sys_cpu_time - ステートメントのシステム CPU の合計 (小数部、マイクロ秒単位)*
STMT_TEXT	CLOB(2 M)	stmt_text - SQL 動的ステートメント・テキスト
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。
* この列の元になるモニター・エレメントの合計消費時間を計算するには、このモニター・エレメントに関する _S で終わっている列で報告されている整数秒と、このモニター・エレメントに関する _MS で終わっている列に報告されている小数秒を、次の式を使用して加算しなければなりません: (*monitor-element-name*_S × 1,000,000 + *monitor-element-name*_MS) ÷ 1,000,000. 例えば、(ELAPSED_EXEC_TIME_S × 1,000,000 + ELAPSED_EXEC_TIME_MS) ÷ 1,000,000。		

SNAP_GET_DYN_SQL

注: この表関数は使用すべきではなく、 1243 ページの
『SNAP_GET_DYN_SQL_V91 表関数 - dynsql 論理グループのスナップショット情報の検索』 に置き換えられました。

スキーマは SYSPROC です。

SNAP_GET_DYN_SQL 表関数は、dynsql 論理データ・グループからのスナップショット情報を戻します。

dbname
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(255) の入力引数。"Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションに -1、またはすべてのアクティブなデータベース・パーティションに -2 を指定します。アクティブなデータベ

ース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

どちらのパラメーターも NULL に設定された場合は、対応するスナップショット API 要求タイプの SNAPSHOT_FILEW ストアード・プロシージャーによって以前にファイルが作成されていない場合にのみ、スナップショットがとられます。

この関数より、次に示されている表が戻されます。

表 300. SNAP_GET_DYN_SQL 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
NUM_EXECUTIONS	BIGINT	num_executions - ステートメント実行回数
NUM_COMPILATIONS	BIGINT	num_compilations - ステートメント・コンパイル数
PREP_TIME_WORST	BIGINT	prep_time_worst - ステートメント最長準備時間
PREP_TIME_BEST	BIGINT	prep_time_best - ステートメント最短準備時間
INT_ROWS_DELETED	BIGINT	int_rows_deleted - 削除された内部行数
INT_ROWS_INSERTED	BIGINT	int_rows_inserted - 挿入された内部行数
INT_ROWS_UPDATED	BIGINT	int_rows_updated - 更新された内部行数
ROWS_READ	BIGINT	rows_read - 読み取り行数
ROWS_WRITTEN	BIGINT	rows_written - 書き込み行数
STMT_SORTS	BIGINT	stmt_sorts - ステートメント・ソート回数
SORT_OVERFLOWS	BIGINT	sort_overflows - ソート・オーバーフロー
TOTAL_SORT_TIME	BIGINT	total_sort_time - ソート時間合計
POOL_DATA_L_READS	BIGINT	pool_data_l_reads - バッファー・プール・データの論理読み取り
POOL_DATA_P_READS	BIGINT	pool_data_p_reads - バッファー・プール・データの物理読み取り
POOL_TEMP_DATA_L_READS	BIGINT	pool_temp_data_l_reads - バッファー・プール一時データの論理読み取り
POOL_TEMP_DATA_P_READS	BIGINT	pool_temp_data_p_reads - バッファー・プール一時データの物理読み取り
POOL_INDEX_L_READS	BIGINT	pool_index_l_reads - バッファー・プール索引の論理読み取り

表 300. SNAP_GET_DYN_SQL 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
POOL_INDEX_P_READS	BIGINT	pool_index_p_reads - バッファー・プール索引の物理読み取り
POOL_TEMP_INDEX_L_READS	BIGINT	pool_temp_index_l_reads - バッファー・プール一時索引の論理読み取り
POOL_TEMP_INDEX_P_READS	BIGINT	pool_temp_index_p_reads - バッファー・プール一時索引の物理読み取り
TOTAL_EXEC_TIME	BIGINT	total_exec_time - ステートメント実行の経過時間 (秒単位)*
TOTAL_EXEC_TIME_MS	BIGINT	total_exec_time - ステートメント実行の経過時間 (小数部、マイクロ秒単位)*
TOTAL_USR_TIME	BIGINT	total_usr_cpu_time - ステートメントのユーザー CPU の合計 (秒単位)*
TOTAL_USR_TIME_MS	BIGINT	total_usr_cpu_time - ステートメントのユーザー CPU の合計 (小数部、マイクロ秒単位)*
TOTAL_SYS_TIME	BIGINT	total_sys_cpu_time - ステートメントのシステム CPU の合計 (秒単位)*
TOTAL_SYS_TIME_MS	BIGINT	total_sys_cpu_time - ステートメントのシステム CPU の合計 (小数部、マイクロ秒単位)*
STMT_TEXT	CLOB	stmt_text - SQL 動的ステートメント・テキスト

* この列の元になるモニター・エレメントの合計消費時間を計算するには、このモニター・エレメントに関する _S で終わっている列で報告されている整数秒と、このモニター・エレメントに関する _MS で終わっている列に報告されている小数秒を、次の式を使用して加算しなければなりません: (*monitor-element-name*_S × 1,000,000 + *monitor-element-name*_MS) ÷ 1,000,000. 例えば、(ELAPSED_EXEC_TIME_S × 1,000,000 + ELAPSED_EXEC_TIME_MS) ÷ 1,000,000。

SNAPLOCK 管理ビューおよび SNAP_GET_LOCK 表関数 - lock 論理データ・グループのスナップショット情報の検索

注: この管理ビューおよび表関数は非推奨になり、以下のものに置き換えられました。すなわち、 460 ページの『MON_GET_APPL_LOCKWAIT - アプリケーションが待機しているロックについての情報の収集』、 490 ページの『MON_GET_LOCKS - 現在接続されているデータベース内のすべてのロックのリスト』、および 425 ページの『MON_FORMAT_LOCK_NAME - 内部ロック名のフォーマット設定と詳細の出力』。

SNAPLOCK 管理ビューおよび SNAP_GET_LOCK 表関数は、特に lock 論理データ・グループの、ロック・スナップショット情報を戻します。

SNAPLOCK 管理ビュー

この管理ビューでは、現在接続されているデータベースの lock 論理データ・グループ・スナップショット情報を検索できます。

SNAPLOCKWAIT 管理ビューと共に使用すると、SNAPLOCK 管理ビューは、GET SNAPSHOT FOR LOCKS ON database-alias CLP コマンドと同等の情報を提供します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、722 ページの表 185を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPLOCK 管理ビューに対する SELECT 特権
- SNAPLOCK 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_LOCK 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベースのデータベース・パーティション 0 のロック情報を検索します。

```
SELECT AGENT_ID, LOCK_OBJECT_TYPE, LOCK_MODE, LOCK_STATUS
   FROM SYSIBMADM.SNAPLOCK WHERE DBPARTITIONNUM = 0
```

以下はこの照会の出力例です。

```
AGENT_ID             LOCK_OBJECT_TYPE LOCK_MODE LOCK_STATUS
-------------------- ---------------- --------- -----------
                   7 TABLE            IX        GRNT

  1 record(s) selected.
```

SNAP_GET_LOCK 表関数

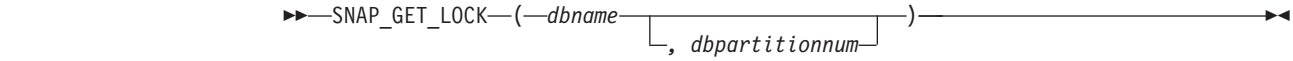
SNAP_GET_LOCK 表関数は SNAPLOCK 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集

約、またはすべてのデータベース・パーティションのいずれかの特定のデータベースを対象とした情報を検索することができます。

SNAP_GET_LOCKWAIT 表関数と共に使用すると、SNAP_GET_LOCK 表関数は、GET SNAPSHOT FOR LOCKS ON database-alias CLP コマンドと同等の情報を提供します。

戻される可能性のある情報の完全なリストは、 722 ページの表 185を参照してください。

構文



スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL 値または空ストリングを指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプが含まれるファイルが存在しない場合、SNAP_GET_LOCK 表関数は、現在接続されているデータベースのスナップショットとデータベース・パーティション番号を取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_LOCK 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベースの現行データベース・パーティションのロック情報を検索します。

```
SELECT AGENT_ID, LOCK_OBJECT_TYPE, LOCK_MODE, LOCK_STATUS
   FROM TABLE(SNAP_GET_LOCK('',-1)) as T
```

以下はこの照会の出力例です。

```
AGENT_ID       LOCK_OBJECT_TYPE   LOCK_MODE  LOCK_STATUS
--------...--- ------------------ ---------- -----------
           680 INTERNALV_LOCK     S          GRNT
           680 INTERNALP_LOCK     S          GRNT

2 record(s) selected.
```

戻される情報

表 301. SNAPLOCK 管理ビューおよび SNAP_GET_LOCK 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
AGENT_ID	BIGINT	agent_id - アプリケーション・ハンドル (エージェント ID)
TAB_FILE_ID	BIGINT	table_file_id - 表ファイル ID

表 301. SNAPLOCK 管理ビューおよび SNAP_GET_LOCK 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
LOCK_OBJECT_TYPE	VARCHAR(18)	lock_object_type - 待機中のロック対象タイプ。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • AUTORESIZE_LOCK • AUTOSTORAGE_LOCK • BLOCK_LOCK • EOT_LOCK • INPLACE_REORG_LOCK • INTERNAL_LOCK • INTERNALB_LOCK • INTERNALC_LOCK • INTERNALJ_LOCK • INTERNALL_LOCK • INTERNALO_LOCK • INTERNALQ_LOCK • INTERNALP_LOCK • INTERNALS_LOCK • INTERNALT_LOCK • INTERNALV_LOCK • KEYVALUE_LOCK • ROW_LOCK • SYSBOOT_LOCK • TABLE_LOCK • TABLE_PART_LOCK • TABLESPACE_LOCK • XML_PATH_LOCK

表 301. SNAPLOCK 管理ビューおよび SNAP_GET_LOCK 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
LOCK_MODE	VARCHAR(10)	lock_mode - ロック・モード。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • IN • IS • IX • NON (ロックなしの場合) • NS • NW • S • SIX • U • X • Z
LOCK_STATUS	VARCHAR(10)	lock_status - ロック状況。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • CONV • GRNT
LOCK_ESCALATION	SMALLINT	lock_escalation - ロック・エスカレーション
TABNAME	VARCHAR(128)	table_name - 表名
TABSCHEMA	VARCHAR(128)	table_schema - 表スキーマ名
TBSP_NAME	VARCHAR(128)	tablespace_name - 表スペース名

表 301. SNAPLOCK 管理ビューおよび SNAP_GET_LOCK 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
LOCK_ATTRIBUTES	VARCHAR(128)	lock_attributes - ロック属性。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。ロックがない場合、テキスト ID は NONE となり、それ以外の場合、以下のいずれかの組み合わせを '+' 記号で区切ったものとなります。 • ALLOW_NEW • DELETE_IN_BLOCK • ESCALATED • INSERT • NEW_REQUEST • RR • RR_IN_BLOCK • UPDATE_DELETE • WAIT_FOR_AVAIL
LOCK_COUNT	BIGINT	lock_count - ロック・カウント
LOCK_CURRENT_MODE	VARCHAR(10)	lock_current_mode - 移行前の元のロック・モード。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • IN • IS • IX • NON (ロックなしの場合) • NS • NW • S • SIX • U • X • Z
LOCK_HOLD_COUNT	BIGINT	lock_hold_count - ロック保留カウント
LOCK_NAME	VARCHAR(32)	lock_name - ロック名
LOCK_RELEASE_FLAGS	BIGINT	lock_release_flags - ロック保留解除フラグ

表 301. SNAPLOCK 管理ビューおよび SNAP_GET_LOCK 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
DATA_PARTITION_ID	INTEGER	data_partition_id - データ・パーティション ID。非パーティション表では、このエレメントは NULL です。
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

SNAPLOCKWAIT 管理ビューおよび SNAP_GET_LOCKWAIT 表関数 - lockwait 論理データ・グループのスナップショット情報の検索

注: この管理ビューおよび表関数は非推奨になり、以下のものに置き換えられました。すなわち、 556 ページの『MON_LOCKWAITS 管理ビュー - ロックの取得を待機しているアプリケーションに関するメトリックの取得』、および 460 ページの『MON_GET_APPL_LOCKWAIT - アプリケーションが待機しているロックについての情報の収集』、 490 ページの『MON_GET_LOCKS - 現在接続されているデータベース内のすべてのロックのリスト』、 425 ページの『MON_FORMAT_LOCK_NAME - 内部ロック名のフォーマット設定と詳細の出力』。

SNAPLOCKWAIT 管理ビューおよび SNAP_GET_LOCKWAIT 表関数は、特に lockwait 論理データ・グループの、ロック待機スナップショット情報を戻します。

SNAPLOCKWAIT 管理ビュー

この管理ビューでは、現在接続されているデータベースの lockwait 論理データ・グループ・スナップショット情報を検索できます。

SNAPLOCK 管理ビューと共に使用すると、SNAPLOCKWAIT 管理ビューは、GET SNAPSHOT FOR LOCKS ON database-alias CLP コマンドと同等の情報を提供します。

スキーマは SYSIBMADM です。

戻される可能性のある情報の完全なリストは、 729 ページの表 186を参照してください。

許可

以下のいずれかの権限が必要です。

- SNAPLOCKWAIT 管理ビューに対する SELECT 特権
- SNAPLOCKWAIT 管理ビューに対する CONTROL 特権
- DATAACCESS 権限

さらに、以下のいずれかの特権または権限も必要です。

- SNAP_GET_LOCKWAIT 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベースのデータベース・パーティション 0 のロック待機情報を検索します。

```
SELECT AGENT_ID, LOCK_MODE, LOCK_OBJECT_TYPE, AGENT_ID_HOLDING_LK,
   LOCK_MODE_REQUESTED FROM SYSIBMADM.SNAPLOCKWAIT
   WHERE DBPARTITIONNUM = 0
```

以下はこの照会の出力例です。

```
AGENT_ID     LOCK_MODE LOCK_OBJECT_TYPE ...
---------...- --------- ---------------- ...
           7 IX        TABLE            ...

  1 record(s) selected.
```

この照会からの出力 (続き)。

```
... AGENT_ID_HOLDING_LK  LOCK_MODE_REQUESTED
... -------------------- --------------------
...                   12 IS
```

SNAP_GET_LOCKWAIT 表関数

SNAP_GET_LOCKWAIT 表関数は SNAPLOCKWAIT 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションのいずれかの特定のデータベースを対象とした情報を検索することができます。

SNAP_GET_LOCK 表関数と共に使用すると、SNAP_GET_LOCKWAIT 表関数は、GET SNAPSHOT FOR LOCKS ON database-alias CLP コマンドと同等の情報を提供します。

戻される可能性のある情報の完全なリストは、729 ページの表 186を参照してください。

構文

スキーマは SYSPROC です。

表関数パラメーター

dbname
現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。 "Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL 値または空ストリングを指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER のオプション入力引数。現行のデータベース・パーティションには -1、すべてのアクティブなデータベース・パーティションの集約には -2 を指定します。*dbname* が NULL に設定されておらず、*dbpartitionnum* が NULL に設定されている場合、*dbpartitionnum* には暗黙的に -1 が設定されます。この入力オプションが使用されない場合、つまり、*dbname* のみが指定されている場合、データはすべてのアクティブなデータベース・パーティションから戻されます。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

dbname および *dbpartitionnum* の両方が NULL に設定された場合、SNAP_WRITE_FILE プロシージャーにより作成されるファイルからのデータの読み取りを試行します。このファイルはいつでも作成される可能性があるため、データは現行のものであるとは限らないことに注意してください。対応するスナップショット API 要求タイプが含まれるファイルが存在しない場合、SNAP_GET_LOCKWAIT 表関数は、現在接続されているデータベースのスナップショットとデータベース・パーティション番号を取得します。

許可

以下のいずれかの権限が必要です。

- SNAP_GET_LOCKWAIT 表関数に対する EXECUTE 特権
- DATAACCESS 権限

さらに、スナップショット・モニター・データにアクセスするには、以下のいずれかの権限も必要です。

- SYSMON
- SYSCTRL
- SYSMAINT
- SYSADM

例

現在接続されているデータベースの現行データベース・パーティションのロック待機情報を検索します。

```
SELECT AGENT_ID, LOCK_MODE, LOCK_OBJECT_TYPE, AGENT_ID_HOLDING_LK,
   LOCK_MODE_REQUESTED FROM TABLE(SNAP_GET_LOCKWAIT('',-1)) AS T
```

以下はこの照会の出力例です。

```
AGENT_ID      LOCK_MODE  LOCK_OBJECT_TYPE   ...
---------...-- ---------- ------------------ ...
           12 X          ROW_LOCK           ...

1 record(s) selected.
```

この照会からの出力 (続き)。

```
... AGENT_ID_HOLDING_LK  LOCK_MODE_REQUESTED
... -------------------- ------------------
...                    7 X
```

使用上の注意

ロック待機情報を表示するには、まずデータベース・マネージャー構成でデフォルトの LOCK モニター・スイッチをオンにする必要があります。変更を即時に有効にするには、CLP を使用してインスタンスに明示的にアタッチし、次いで以下の CLP コマンドを発行します。

```
UPDATE DATABASE MANAGER CONFIGURATION CLP USING DFT_MON_LOCK ON
```

デフォルトの設定値も、ADMIN_CMD ストアード・プロシージャーによりオンにすることができます。以下に例を示します。

```
CALL SYSPROC.ADMIN_CMD('update dbm cfg using DFT_MON_LOCK ON')
```

ADMIN_CMD ストアード・プロシージャーを使用する場合、または事前にインスタンスにアタッチせずに CLP コマンドを使用する場合、インスタンスをリサイクルしなければ変更は有効になりません。

戻される情報

表 302. SNAPLOCKWAIT 管理ビューおよび SNAP_GET_LOCKWAIT 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
AGENT_ID	BIGINT	agent_id - アプリケーション・ハンドル (エージェント ID)
SUBSECTION_NUMBER	BIGINT	ss_number - サブセクション番号

表 302. SNAPLOCKWAIT 管理ビューおよび SNAP_GET_LOCKWAIT 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
LOCK_MODE	VARCHAR(10)	lock_mode - ロック・モード。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • IN • IS • IX • NON (ロックなしの場合) • NS • NW • S • SIX • U • X • Z

表 302. SNAPLOCKWAIT 管理ビューおよび SNAP_GET_LOCKWAIT 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
LOCK_OBJECT_TYPE	VARCHAR(18)	lock_object_type - 待機中のロック対象タイプ。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • AUTORESIZE_LOCK • AUTOSTORAGE_LOCK • BLOCK_LOCK • EOT_LOCK • INPLACE_REORG_LOCK • INTERNAL_LOCK • INTERNALB_LOCK • INTERNALC_LOCK • INTERNALJ_LOCK • INTERNALL_LOCK • INTERNALO_LOCK • INTERNALQ_LOCK • INTERNALP_LOCK • INTERNALS_LOCK • INTERNALT_LOCK • INTERNALV_LOCK • KEYVALUE_LOCK • ROW_LOCK • SYSBOOT_LOCK • TABLE_LOCK • TABLE_PART_LOCK • TABLESPACE_LOCK • XML_PATH_LOCK
AGENT_ID_HOLDING_LK	BIGINT	agent_id_holding_lock - ロックを保持しているエージェント ID
LOCK_WAIT_START_TIME	TIMESTAMP	lock_wait_start_time - ロック待機開始タイム・スタンプ

表 302. SNAPLOCKWAIT 管理ビューおよび SNAP_GET_LOCKWAIT 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
LOCK_MODE_REQUESTED	VARCHAR(10)	lock_mode_requested - 要求されているロック・モード。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • IN • IS • IX • NON (ロックなしの場合) • NS • NW • S • SIX • U • X • Z
LOCK_ESCALATION	SMALLINT	lock_escalation - ロック・エスカレーション
TABNAME	VARCHAR(128)	table_name - 表名
TABSCHEMA	VARCHAR(128)	table_schema - 表スキーマ名
TBSP_NAME	VARCHAR(128)	tablespace_name - 表スペース名
APPL_ID_HOLDING_LK	VARCHAR(128)	appl_id_holding_lk - ロックを保持しているアプリケーション ID
LOCK_ATTRIBUTES	VARCHAR(128)	lock_attributes - ロック属性。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。ロックがない場合、テキスト ID は NONE となり、それ以外の場合、以下のいずれかの組み合わせを '+' 記号で区切ったものとなります。 • ALLOW_NEW • DELETE_IN_BLOCK • ESCALATED • INSERT • NEW_REQUEST • RR • RR_IN_BLOCK • UPDATE_DELETE • WAIT_FOR_AVAIL

表 302. SNAPLOCKWAIT 管理ビューおよび SNAP_GET_LOCKWAIT 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
LOCK_CURRENT_MODE	VARCHAR(10)	lock_current_mode - 移行前の元のロック・モード。このインターフェースは、sqlmon.h での定義に基づくテキスト ID を戻します。これは次のいずれかです。 • IN • IS • IX • NON (ロックなしの場合) • NS • NW • S • SIX • U • X • Z
LOCK_NAME	VARCHAR(32)	lock_name - ロック名
LOCK_RELEASE_FLAGS	BIGINT	lock_release_flags - ロック保留解除フラグ。
DATA_PARTITION_ID	INTEGER	data_partition_id - データ・パーティション ID。非パーティション表では、このエレメントは NULL です。
DBPARTITIONNUM	SMALLINT	この行のデータが検索されたデータベース・パーティション。

SNAP_GET_STO_PATHS

注: この表関数は使用すべきではなく、 892 ページの『SNAPSTORAGE_PATHS 管理ビューおよび SNAP_GET_STORAGE_PATHS 表関数 - 自動ストレージ・パスの情報の検索』 に置き換えられました。

スキーマは SYSPROC です。

SNAP_GET_STO_PATHS 表関数は、storage_paths 論理データ・グループからのスナップショット情報を戻します。

dbname
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(255) の入力引数。

"Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションに -1、またはすべてのアクティブなデータベース・パーティションに -2 を指定します。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

どちらのパラメーターも NULL に設定された場合は、対応するスナップショット API 要求タイプの SNAPSHOT_FILEW ストアード・プロシージャーによって以前にファイルが作成されていない場合にのみ、スナップショットがとられます。

この関数より、次に示されている表が戻されます。

表 303. SNAP_GET_STO_PATHS 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
DB_NAME	VARCHAR(128)	db_name - データベース名
DB_STORAGE_PATH	VARCHAR(256)	db_storage_path - 自動ストレージ・パス

SNAP_GET_TAB

注: この表関数は使用すべきではなく、 751 ページの『SNAPTAB 管理ビューおよび SNAP_GET_TAB_V91 表関数 - table 論理データ・グループのスナップショット情報の検索』 に置き換えられました。

```
►►—SNAP_GET_TAB—(—dbname—,—dbpartitionnum—)————————————————————►◄
```

スキーマは SYSPROC です。

SNAP_GET_TAB 表関数は、table 論理データ・グループからのスナップショット情報を戻します。

dbname
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(255) の入力引数。"Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションに -1、またはすべてのアクティブなデータベース・パーティションに -2 を指定します。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

どちらのパラメーターも NULL に設定された場合は、対応するスナップショット API 要求タイプの SNAPSHOT_FILEW ストアード・プロシージャーによって以前にファイルが作成されていない場合にのみ、スナップショットがとられます。

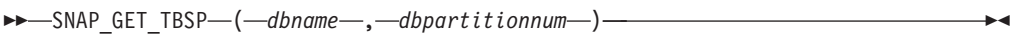
この関数より、次に示されている表が戻されます。

表 304. SNAP_GET_TAB 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
TABSCHEMA	VARCHAR(128)	table_schema - 表スキーマ名
TABNAME	VARCHAR(128)	table_name - 表名
TAB_FILE_ID	BIGINT	table_file_id - 表ファイル ID
TAB_TYPE	BIGINT	table_type - 表タイプ
DATA_OBJECT_PAGES	BIGINT	data_object_pages - データ・オブジェクト・ページ数
INDEX_OBJECT_PAGES	BIGINT	index_object_pages - 索引オブジェクト・ページ数
LOB_OBJECT_PAGES	BIGINT	lob_object_pages - LOB オブジェクト・ページ数
LONG_OBJECT_PAGES	BIGINT	long_object_pages - 長オブジェクト・ページ数
ROWS_READ	BIGINT	rows_read - 読み取り行数
ROWS_WRITTEN	BIGINT	rows_written - 書き込み行数
OVERFLOW_ACCESSES	BIGINT	overflow_accesses - オーバーフロー・レコードへのアクセス
PAGE_REORGS	BIGINT	page_reorgs - ページ再編成
DBPARTITIONNUM	SMALLINT	node_number - ノード番号

SNAP_GET_TBSP

注: この表関数は使用すべきではなく、 760 ページの『SNAPTBSP 管理ビューおよび SNAP_GET_TBSP_V91 表関数 - 表スペース論理データ・グループのスナップショット情報の検索』 に置き換えられました。



スキーマは SYSPROC です。

SNAP_GET_TBSP は、table 論理データ・グループからのスナップショット情報を戻します。

dbname
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(255) の入力引数。"Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションには -1 を指定します。 NULL 値を指定すると、-1 が暗黙的に設定されます。

どちらのパラメーターも NULL に設定された場合は、対応するスナップショット API 要求タイプの SNAPSHOT_FILEW ストアード・プロシージャーによって以前にファイルが作成されていない場合にのみ、スナップショットがとられます。

この関数より、次に示されている表が戻されます。

表 305. SNAP_GET_TBSP 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
TBSP_NAME	VARCHAR(128)	tablespace_name - 表スペース名
TBSP_ID	BIGINT	tablespace_id - 表スペース ID
TBSP_TYPE	SMALLINT	tablespace_type - 表スペース・タイプ
TBSP_CONTENT_TYPE	SMALLINT	tablespace_content_type - 表スペースのコンテンツ・タイプ
TBSP_PAGE_SIZE	BIGINT	tablespace_page_size - 表スペースのページ・サイズ
TBSP_EXTENT_SIZE	BIGINT	tablespace_extent_size - 表スペースのエクステント・サイズ
TBSP_PREFETCH_SIZE	BIGINT	tablespace_prefetch_size - 表スペースのプリフェッチ・サイズ

表 305. SNAP_GET_TBSP 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
TBSP_CUR_POOL_ID	BIGINT	tablespace_cur_pool_id - 現在使用中のバッファー・プール
TBSP_NEXT_POOL_ID	BIGINT	tablespace_next_pool_id - 次の始動時に使用されるバッファー・プール
FS_CACHING[1]	SMALLINT	fs_caching - ファイル・システム・キャッシング
POOL_DATA_L_READS	BIGINT	pool_data_l_reads - バッファー・プール・データの論理読み取り
POOL_DATA_P_READS	BIGINT	pool_data_p_reads - バッファー・プール・データの物理読み取り
POOL_TEMP_DATA_L_READS	BIGINT	pool_temp_data_l_reads - バッファー・プール一時データの論理読み取り
POOL_TEMP_DATA_P_READS	BIGINT	pool_temp_data_p_reads - バッファー・プール一時データの物理読み取り
POOL_ASYNC_DATA_READS	BIGINT	pool_async_data_reads - バッファー・プール非同期データ読み取り
POOL_DATA_WRITES	BIGINT	pool_data_writes - バッファー・プールへのデータの書き込み
POOL_ASYNC_DATA_WRITES	BIGINT	pool_async_data_writes - バッファー・プール非同期データ書き込み
POOL_INDEX_L_READS	BIGINT	pool_index_l_reads - バッファー・プール索引の論理読み取り
POOL_INDEX_P_READS	BIGINT	pool_index_p_reads - バッファー・プール索引の物理読み取り
POOL_TEMP_INDEX_L_READS	BIGINT	pool_temp_index_l_reads - バッファー・プール一時索引の論理読み取り
POOL_TEMP_INDEX_P_READS	BIGINT	pool_temp_index_p_reads - バッファー・プール一時索引の物理読み取り
POOL_ASYNC_INDEX_READS	BIGINT	pool_async_index_reads - バッファー・プール非同期索引読み取り
POOL_INDEX_WRITES	BIGINT	pool_index_writes - バッファー・プール索引の書き込み
POOL_ASYNC_INDEX_WRITES	BIGINT	pool_async_index_writes - バッファー・プール非同期索引書き込み
POOL_READ_TIME	BIGINT	pool_read_time - バッファー・プール物理読み取り時間の合計
POOL_WRITE_TIME	BIGINT	pool_write_time - バッファー・プール物理書き込み時間の合計
POOL_ASYNC_READ_TIME	BIGINT	pool_async_read_time - バッファー・プール非同期読み取り時間

表 305. SNAP_GET_TBSP 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
POOL_ASYNC_WRITE_TIME	BIGINT	pool_async_write_time - バッファー・プール非同期書き込み時間
POOL_ASYNC_DATA_READ_REQS	BIGINT	pool_async_data_read_reqs - バッファー・プール非同期読み取り要求
POOL_ASYNC_INDEX_READ_REQS	BIGINT	pool_async_index_read_reqs - バッファー・プール非同期索引読み取り要求
POOL_NO_VICTIM_BUFFER	BIGINT	pool_no_victim_buffer - バッファー・プールの非ビクティム・バッファー数
DIRECT_READS	BIGINT	direct_reads - データベースからの直接読み取り
DIRECT_WRITES	BIGINT	direct_writes - データベースへの直接書き込み
DIRECT_READ_REQS	BIGINT	direct_read_reqs - 直接読み取り要求
DIRECT_WRITE_REQS	BIGINT	direct_write_reqs - 直接書き込み要求
DIRECT_READ_TIME	BIGINT	direct_read_time - 直接読み取り時間
DIRECT_WRITE_TIME	BIGINT	direct_write_time - 直接書き込み時間
FILES_CLOSED	BIGINT	files_closed - 閉じられたデータベース・ファイル
UNREAD_PREFETCH_PAGES	BIGINT	unread_prefetch_pages - 読み取り不能プリフェッチ・ページ
POOL_DATA_TO_ESTORE	BIGINT	pool_data_to_estore ESTORE モニター・エレメントは廃止されています。廃止されたモニター・エレメントには NULL 値が戻されます。
POOL_INDEX_TO_ESTORE	BIGINT	pool_index_to_estore ESTORE モニター・エレメントは廃止されています。廃止されたモニター・エレメントには NULL 値が戻されます。
POOL_INDEX_FROM_ESTORE	BIGINT	pool_index_from_estore ESTORE モニター・エレメントは廃止されています。廃止されたモニター・エレメントには NULL 値が戻されます。
POOL_DATA_FROM_ESTORE	BIGINT	pool_data_from_estore ESTORE モニター・エレメントは廃止されています。廃止されたモニター・エレメントには NULL 値が戻されます。
TBSP_REBALANCER_MODE	BIGINT	tablespace_rebalancer_mode - リバランサー・モード

表 305. SNAP_GET_TBSP 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
TBSP_USING_AUTO_STORAGE	SMALLINT	tablespace_using_auto_storage - 自動ストレージが使用可能な表スペース
TBSP_AUTO_RESIZE_ENABLED	SMALLINT	tablespace_auto_resize_enabled - 表スペースの自動サイズ変更可能
[1] FS_CACHING が 0 であれば、ファイル・システム・キャッシュは使用可能であり、FS_CACHING が 1 であれば、ファイル・システム・キャッシュは使用不可です。		

SNAP_GET_TBSP_PART

注: この表関数は使用すべきではなく、 919 ページの『SNAPTBSP_PART 管理ビューおよび SNAP_GET_TBSP_PART_V91 表関数 - tablespace_nodeinfo 論理データ・グループのスナップショット情報の検索』 に置き換えられました。

スキーマは SYSPROC です。

SNAP_GET_TBSP_PART 表関数は、tablespace_nodeinfo 論理データ・グループからのスナップショット情報を戻します。

dbname
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(255) の入力引数。"Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションには -1 を指定します。 NULL 値を指定すると、-1 が暗黙的に設定されます。

どちらのパラメーターも NULL に設定された場合は、対応するスナップショット API 要求タイプの SNAPSHOT_FILEW ストアード・プロシージャーによって以前にファイルが作成されていない場合にのみ、スナップショットがとられます。

この関数より、次に示されている表が戻されます。

表 306. SNAP_GET_TBSP_PART 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
TBSP_NAME	VARCHAR (128)	tablespace_name - 表スペース名

表 306. SNAP_GET_TBSP_PART 表関数によって戻される情報 (続き)

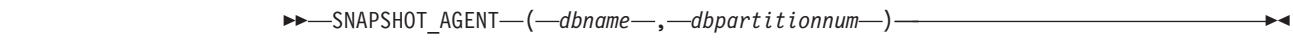
列名	データ・タイプ	説明または対応するモニター・エレメント
TBSP_ID	BIGINT	tablespace_id - 表スペース ID
TBSP_STATE	BIGINT	tablespace_state - 表スペースの状態
TBSP_PREFETCH_SIZE	BIGINT	tablespace_prefetch_size - 表スペースのプリフェッチ・サイズ
TBSP_NUM_QUIESCERS	BIGINT	tablespace_num_quiescers - 静止プログラム数
TBSP_STATE_CHANGE_OBJECT_ID	BIGINT	tablespace_state_change_object_id - 状態変更オブジェクト ID
TBSP_STATE_CHANGE_TBSP_ID	BIGINT	tablespace_state_change_ts_id - 状態変更表スペース ID
TBSP_MIN_RECOVERY_TIME	TIMESTAMP	tablespace_min_recovery_time - ロールフォワードの最小リカバリー時間
TBSP_TOTAL_PAGES	BIGINT	tablespace_total_pages - 表スペース内の合計ページ数
TBSP_USABLE_PAGES	BIGINT	tablespace_usable_pages - 表スペース内の使用可能ページ数
TBSP_USED_PAGES	BIGINT	tablespace_used_pages - 表スペース内の使用されているページ数
TBSP_FREE_PAGES	BIGINT	tablespace_free_pages - 表スペース内のフリー・ページ数
TBSP_PENDING_FREE_PAGES	BIGINT	tablespace_pending_free_pages - 表スペース内のペンディング・フリー・ページ数
TBSP_PAGE_TOP	BIGINT	tablespace_page_top - 表スペース最高水準点
REBALANCER_MODE	BIGINT	tablespace_rebalancer_mode - リバランサー・モード
REBALANCER_EXTENTS_REMAINING	BIGINT	tablespace_rebalancer_extents_remaining - リバランサーで処理されるエクステントの合計数
REBALANCER_EXTENTS_PROCESSED	BIGINT	tablespace_rebalancer_extents_processed - リバランサーで処理されたエクステントの数
REBALANCER_PRIORITY	BIGINT	tablespace_rebalancer_priority - 現行のリバランサー優先順位
REBALANCER_START_TIME	TIMESTAMP	tablespace_rebalancer_start_time - リバランサー開始時刻
REBALANCER_RESTART_TIME	TIMESTAMP	tablespace_rebalancer_restart_time - リバランサー再始動時刻
REBALANCER_LAST_EXTENT_MOVED	BIGINT	tablespace_rebalancer_last_extent_moved - リバランサーによって最後に移動されたエクステント

表 306. SNAP_GET_TBSP_PART 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
TBSP_NUM_RANGES	BIGINT	tablespace_num_ranges - 表スペース・マップ内の範囲数
TBSP_NUM_CONTAINERS	BIGINT	tablespace_num_containers - 表スペース内のコンテナー数
TBSP_INITIAL_SIZE	BIGINT	tablespace_initial_size - 表スペースの初期サイズ
TBSP_CURRENT_SIZE	BIGINT	tablespace_current_size - 表スペースの現行サイズ
TBSP_MAX_SIZE	BIGINT	tablespace_max_size - 表スペースの最大サイズ
TBSP_INCREASE_SIZE	BIGINT	tablespace_increase_size - バイト単位のサイズの増加
TBSP_INCREASE_SIZE_PERCENT	SMALLINT	tablespace_increase_size_percent - パーセント単位のサイズの増加
TBSP_LAST_RESIZE_TIME	TIMESTAMP	tablespace_last_resize_time - 最後にサイズ変更が正常に行われた時刻
TBSP_LAST_RESIZE_FAILED	SMALLINT	tablespace_last_resize_failed - 失敗した最後のサイズ変更
DBPARTITIONNUM	SMALLINT	node_number - ノード番号

SNAPSHOT_AGENT

注: この表関数は使用すべきではなく、 636 ページの『SNAPAGENT 管理ビューおよび SNAP_GET_AGENT 表関数 - agent 論理データ・グループのアプリケーション・スナップショット情報の検索』 に置き換えられました。



スキーマは SYSPROC です。

SNAPSHOT_AGENT 関数は、アプリケーション・スナップショットからのエージェントに関する情報を戻します。

dbname
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(255) の入力引数。"Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。データベース・インスタンス下のすべてのデータベースからスナップショットを取る場合は、 NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションに -1、またはすべてのアクティブなデータベース・パーティションに -2 を指定します。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

どちらのパラメーターも NULL に設定された場合は、対応するスナップショット API 要求タイプの SNAPSHOT_FILEW ストアード・プロシージャーによって以前にファイルが作成されていない場合にのみ、スナップショットがとられます。

この関数より、次に示されている表が戻されます。

表 307. SNAPSHOT_AGENT 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
AGENT_ID	BIGINT	agent_id - アプリケーション・ハンドル (エージェント ID)
AGENT_PID	BIGINT	agent_pid - エンジン・ディスパッチ可能単位 (EDU)

SNAPSHOT_APPL

アプリケーション・スナップショットからの一般情報を戻します。

注: この表関数は使用すべきではなく、 1187 ページの『SNAP_GET_APPL 表関数 - appl 論理データ・グループのスナップショット情報の検索』 に置き換えられました。

```
►►—SNAPSHOT_APPL—(—dbname—,—dbpartitionnum—)———————————————————►◄
```

スキーマは SYSPROC です。

dbname
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(255) の入力引数。"Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。データベース・インスタンス下のすべてのデータベースからスナップショットを取る場合は、 NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションに -1、またはすべてのアクティ

ブなデータベース・パーティションに -2 を指定します。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

どちらのパラメーターも NULL に設定された場合は、対応するスナップショット API 要求タイプの SNAPSHOT_FILEW ストアード・プロシージャーによって以前にファイルが作成されていない場合にのみ、スナップショットがとられます。

この関数より、次に示されている表が戻されます。

表 308. SNAPSHOT_APPL 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
AGENT_ID	BIGINT	**agent_id** - アプリケーション・ハンドル (エージェント ID)
UOW_LOG_SPACE_USED	BIGINT	**uow_log_space_used** - 作業単位ログ・スペース
ROWS_READ	BIGINT	**rows_read** - 読み取り行数
ROWS_WRITTEN	BIGINT	**rows_written** - 書き込み行数
POOL_DATA_L_READS	BIGINT	**pool_data_l_reads** - バッファー・プール・データの論理読み取り
POOL_DATA_P_READS	BIGINT	**pool_data_p_reads** - バッファー・プール・データの物理読み取り
POOL_DATA_WRITES	BIGINT	**pool_data_writes** - バッファー・プールへのデータの書き込み
POOL_INDEX_L_READS	BIGINT	**pool_index_l_reads** - バッファー・プール索引の論理読み取り
POOL_INDEX_P_READS	BIGINT	**pool_index_p_reads** - バッファー・プール索引の物理読み取り
POOL_INDEX_WRITES	BIGINT	**pool_index_writes** - バッファー・プール索引の書き込み
POOL_READ_TIME	BIGINT	**pool_read_time** - バッファー・プール物理読み取り時間の合計
POOL_WRITE_TIME	BIGINT	**pool_write_time** - バッファー・プール物理書き込み時間の合計
DIRECT_READS	BIGINT	**direct_reads** - データベースからの直接読み取り
DIRECT_WRITES	BIGINT	**direct_writes** - データベースへの直接書き込み
DIRECT_READ_REQS	BIGINT	**direct_read_reqs** - 直接読み取り要求
DIRECT_WRITE_REQS	BIGINT	**direct_write_reqs** - 直接書き込み要求
DIRECT_READ_TIME	BIGINT	**direct_read_time** - 直接読み取り時間

表 308. SNAPSHOT_APPL 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
DIRECT_WRITE_TIME	BIGINT	**direct_write_time** - 直接書き込み時間
POOL_DATA_TO_ESTORE	BIGINT	**pool_data_to_estore** ESTORE モニター・エレメントは廃止されています。廃止されたモニター・エレメントには NULL 値が戻されます。
POOL_INDEX_TO_ESTORE	BIGINT	**pool_index_to_estore** ESTORE モニター・エレメントは廃止されています。廃止されたモニター・エレメントには NULL 値が戻されます。
POOL_INDEX_FROM_ESTORE	BIGINT	**pool_index_from_estore** ESTORE モニター・エレメントは廃止されています。廃止されたモニター・エレメントには NULL 値が戻されます。
POOL_DATA_FROM_ESTORE	BIGINT	**pool_data_from_estore** ESTORE モニター・エレメントは廃止されています。廃止されたモニター・エレメントには NULL 値が戻されます。
UNREAD_PREFETCH_PAGES	BIGINT	**unread_prefetch_pages** - 読み取り不能プリフェッチ・ページ
LOCKS_HELD	BIGINT	**locks_held** - ロック保持数
LOCK_WAITS	BIGINT	**lock_waits** - ロック待機数
LOCK_WAIT_TIME	BIGINT	**lock_wait_time** - ロック待機中の時間
LOCK_ESCALS	BIGINT	**lock_escals** - ロック・エスカレーション数
X_LOCK_ESCALS	BIGINT	**x_lock_escals** - 排他ロック・エスカレーション数
DEADLOCKS	BIGINT	**deadlocks** - デッドロック検出数
TOTAL_SORTS	BIGINT	**total_sorts** - ソート合計
TOTAL_SORT_TIME	BIGINT	**total_sort_time** - ソート時間合計
SORT_OVERFLOWS	BIGINT	**sort_overflows** - ソート・オーバーフロー
COMMIT_SQL_STMTS	BIGINT	**commit_sql_stmts** - 試行されたコミット・ステートメント
ROLLBACK_SQL_STMTS	BIGINT	**rollback_sql_stmts** - 試行されたロールバック・ステートメント
DYNAMIC_SQL_STMTS	BIGINT	**dynamic_sql_stmts** - 試行された動的 SQL ステートメント
STATIC_SQL_STMTS	BIGINT	**static_sql_stmts** - 試行された静的 SQL ステートメント

表 308. SNAPSHOT_APPL 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
FAILED_SQL_STMTS	BIGINT	**failed_sql_stmts** - 失敗したステートメント操作
SELECT_SQL_STMTS	BIGINT	**select_sql_stmts** - 実行された選択 SQL ステートメント
DDL_SQL_STMTS	BIGINT	**ddl_sql_stmts** - データ定義言語 (DDL) SQL ステートメント
UID_SQL_STMTS	BIGINT	**uid_sql_stmts** - 実行された更新/挿入/削除 SQL ステートメント
INT_AUTO_REBINDS	BIGINT	**int_auto_rebinds** - 内部自動再バインド
INT_ROWS_DELETED	BIGINT	**int_rows_deleted** - 削除された内部行数
INT_ROWS_UPDATED	BIGINT	**int_rows_updated** - 更新された内部行数
INT_COMMITS	BIGINT	**int_commits** - 内部コミット数
INT_ROLLBACKS	BIGINT	**int_rollbacks** - 内部ロールバック数
INT_DEADLOCK_ROLLBACKS	BIGINT	**int_deadlock_rollbacks** - デッドロックによる内部ロールバック数
ROWS_DELETED	BIGINT	**rows_deleted** - 削除行数
ROWS_INSERTED	BIGINT	**rows_inserted** - 挿入行数
ROWS_UPDATED	BIGINT	**rows_updated** - 更新行数
ROWS_SELECTED	BIGINT	**rows_selected** - 選択行数
BINDS_PRECOMPILES	BIGINT	**binds_precompiles** - 試行されたバインド/プリコンパイル
OPEN_REM_CURS	BIGINT	**open_rem_curs** - 開かれているリモート・カーソル
OPEN_REM_CURS_BLK	BIGINT	**open_rem_curs_blk** - 開かれているリモート・ブロック・カーソル
REJ_CURS_BLK	BIGINT	**rej_curs_blk** - リジェクトされたブロック・カーソル要求
ACC_CURS_BLK	BIGINT	**acc_curs_blk** - 受け入れられたブロック・カーソル要求
SQL_REQS_SINCE_COMMIT	BIGINT	**sql_reqs_since_commit** - 最終コミット後の SQL 要求数
LOCK_TIMEOUTS	BIGINT	**lock_timeouts** - ロック・タイムアウト数
INT_ROWS_INSERTED	BIGINT	**int_rows_inserted** - 挿入された内部行数
OPEN_LOC_CURS	BIGINT	**open_loc_curs** - 開かれているローカル・カーソル
OPEN_LOC_CURS_BLK	BIGINT	**open_loc_curs_blk** - 開かれているローカル・ブロック・カーソル

表 308. SNAPSHOT_APPL 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
PKG_CACHE_LOOKUPS	BIGINT	**pkg_cache_lookups** - パッケージ・キャッシュ参照
PKG_CACHE_INSERTS	BIGINT	**pkg_cache_inserts** - パッケージ・キャッシュ挿入
CAT_CACHE_LOOKUPS	BIGINT	**cat_cache_lookups** - カタログ・キャッシュ参照数
CAT_CACHE_INSERTS	BIGINT	**cat_cache_inserts** - カタログ・キャッシュ挿入数
CAT_CACHE_OVERFLOWS	BIGINT	**cat_cache_overflows** - カタログ・キャッシュ・オーバーフロー数
CAT_CACHE_HEAP_FULL	BIGINT	**cat_cache_overflows** - カタログ・キャッシュ・オーバーフロー数
NUM_AGENTS	BIGINT	**num_agents** - ステートメントで作動しているエージェントの数
AGENTS_STOLEN	BIGINT	**agents_stolen** - スチールされたエージェント
ASSOCIATED_AGENTS_TOP	BIGINT	**associated_agents_top** - 関連エージェント最大数
APPL_PRIORITY	BIGINT	**appl_priority** - アプリケーション・エージェント優先順位
APPL_PRIORITY_TYPE	BIGINT	**appl_priority_type** - アプリケーション優先順位タイプ
PREFETCH_WAIT_TIME	BIGINT	**prefetch_wait_time** - プリフェッチ待ち時間
APPL_SECTION_LOOKUPS	BIGINT	**appl_section_lookups** - セクションの参照回数
APPL_SECTION_INSERTS	BIGINT	**appl_section_inserts** - セクション挿入数
LOCKS_WAITING	BIGINT	**locks_waiting** - ロックで待機中の現行エージェント
TOTAL_HASH_JOINS	BIGINT	**total_hash_joins** - ハッシュ結合の合計
TOTAL_HASH_LOOPS	BIGINT	**total_hash_loops** - ハッシュ・ループの合計
HASH_JOIN_OVERFLOWS	BIGINT	**hash_join_overflows** - ハッシュ結合のオーバーフロー
HASH_JOIN_SMALL_ OVERFLOWS	BIGINT	**hash_join_small_overflows** - ハッシュ結合の短精度オーバーフロー
APPL_IDLE_TIME	BIGINT	**appl_idle_time** - アプリケーション・アイドル時間

表 308. SNAPSHOT_APPL 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
UOW_LOCK_WAIT_TIME	BIGINT	**uow_lock_wait_time** - ロック待機中の作業単位の合計時間
UOW_COMP_STATUS	BIGINT	**uow_comp_status** - 作業単位完了状況
AGENT_USR_CPU_TIME_S	BIGINT	**agent_usr_cpu_time** - エージェントが使用したユーザー CPU 時間 (秒単位)*
AGENT_USR_CPU_TIME_MS	BIGINT	**agent_usr_cpu_time** - エージェントが使用したユーザー CPU 時間 (小数部、マイクロ秒単位)*
AGENT_SYS_CPU_TIME_S	BIGINT	**agent_sys_cpu_time** - エージェントが使用したシステム CPU 時間 (秒単位)*
AGENT_SYS_CPU_TIME_MS	BIGINT	**agent_sys_cpu_time** - エージェントが使用したシステム CPU 時間 (小数部、マイクロ秒単位)*
APPL_CON_TIME	TIMESTAMP	**appl_con_time** - 接続要求開始タイム・スタンプ
CONN_COMPLETE_TIME	TIMESTAMP	**conn_complete_time** - 接続要求完了タイム・スタンプ
LAST_RESET	TIMESTAMP	**last_reset** - 最後のリセット・タイム・スタンプ
UOW_START_TIME	TIMESTAMP	**uow_start_time** - 作業単位開始タイム・スタンプ
UOW_STOP_TIME	TIMESTAMP	**uow_stop_time** - 作業単位停止タイム・スタンプ
PREV_UOW_STOP_TIME	TIMESTAMP	**prev_uow_stop_time** - 直前の作業単位完了タイム・スタンプ
UOW_ELAPSED_TIME_S	BIGINT	**uow_elapsed_time** - 最新の作業単位の経過時間 (秒単位)*
UOW_ELAPSED_TIME_MS	BIGINT	**uow_elapsed_time** - 最新の作業単位の経過時間 (小数部、マイクロ秒単位)*
ELAPSED_EXEC_TIME_S	BIGINT	**elapsed_exec_time** - ステートメント実行経過時間 (秒単位)*
ELAPSED_EXEC_TIME_MS	BIGINT	**elapsed_exec_time** - ステートメント実行経過時間 (小数部、マイクロ秒単位)*
INBOUND_COMM_ADDRESS	VARCHAR(32)	**inbound_comm_address** - インバウンド通信アドレス

表 308. SNAPSHOT_APPL 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
* この列の元になるモニター・エレメントの合計消費時間を計算するには、このモニター・エレメントに関する _S で終わっている列で報告されている整数秒と、このモニター・エレメントに関する _MS で終わっている列に報告されている小数秒を、次の式を使用して加算しなければなりません: (*monitor-element-name*_S × 1,000,000 + *monitor-element-name*_MS) ÷ 1,000,000. 例えば、(ELAPSED_EXEC_TIME_S × 1,000,000 + ELAPSED_EXEC_TIME_MS) ÷ 1,000,000。		

SNAPSHOT_APPL_INFO

アプリケーション・スナップショットからの一般情報を戻します。

注: この表関数は使用すべきではなく、 1195 ページの『SNAP_GET_APPL_INFO 表関数 - appl_info 論理データ・グループのスナップショット情報の検索』 に置き換えられました。

```
►►—SNAPSHOT_APPL_INFO—(—dbname—,—dbpartitionnum—)————————————————►◄
```

スキーマは SYSPROC です。

dbname
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(255) の入力引数。"Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。データベース・インスタンス下のすべてのデータベースからスナップショットを取る場合は、 NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションに -1、またはすべてのアクティブなデータベース・パーティションに -2 を指定します。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

どちらのパラメーターも NULL に設定された場合は、対応するスナップショット API 要求タイプの SNAPSHOT_FILEW ストアード・プロシージャーによって以前にファイルが作成されていない場合にのみ、スナップショットがとられます。

この関数より、次に示されている表が戻されます。

表 309. SNAPSHOT_APPL_INFO 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
AGENT_ID	BIGINT	**agent_id** - アプリケーション・ハンドル (エージェント ID)
APPL_STATUS	BIGINT	**appl_status** - アプリケーション状況
CODEPAGE_ID	BIGINT	**codepage_id** - アプリケーションで使用するコード・ページ ID
NUM_ASSOC_AGENTS	BIGINT	**num_assoc_agents** - 関連したエージェント数
COORD_PARTITION_NUM	BIGINT	**coord_node** - コーディネーター・ノード
AUTHORITY_LVL	BIGINT	**authority_lvl** - ユーザー許可レベル
CLIENT_PID	BIGINT	**client_pid** - クライアント・プロセス ID
COORD_AGENT_PID	BIGINT	**coord_agent_pid** - コーディネーター・エージェント
STATUS_CHANGE_TIME	TIMESTAMP	**status_change_time** - アプリケーション状況変更時刻
CLIENT_PLATFORM	SMALLINT	**client_platform** - クライアント・オペレーティング・プラットフォーム
CLIENT_PROTOCOL	SMALLINT	**client_protocol** - クライアント通信プロトコル
COUNTRY_CODE	SMALLINT	**territory_code** - データベース・テリトリー・コード
APPL_NAME	VARCHAR(256)	**appl_name** - アプリケーション名
APPL_ID	VARCHAR(128)	**appl_id** - アプリケーション ID
SEQUENCE_NO	VARCHAR(4)	**sequence_no** - シーケンス番号
AUTH_ID	VARCHAR(128)	**auth_id** - 許可 ID
CLIENT_NNAME	VARCHAR(128)	**client_nname** モニター・エレメントは使用すべきではありません。返される値は無効な値です。
CLIENT_PRDID	VARCHAR(128)	**client_prdid** - クライアント製品/バージョン ID

表 309. SNAPSHOT_APPL_INFO 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
INPUT_DB_ALIAS	VARCHAR(128)	**input_db_alias** - 入力データベース別名
CLIENT_DB_ALIAS	VARCHAR(128)	**client_db_alias** - アプリケーションで使用するデータベース別名
DB_NAME	VARCHAR(128)	**db_name** - データベース名
DB_PATH	VARCHAR(1024)	**db_path** - データベース・パス
EXECUTION_ID	VARCHAR(128)	**execution_id** - ユーザー・ログイン ID
CORR_TOKEN	VARCHAR(128)	**corr_token** - DRDA 相関トークン
TPMON_CLIENT_USERID	VARCHAR(256)	**tpmon_client_userid** - TP モニター・クライアント・ユーザー ID
TPMON_CLIENT_WKSTN	VARCHAR(256)	**tpmon_client_wkstn** - TP モニター・クライアント・ワークステーション名
TPMON_CLIENT_APP	VARCHAR(256)	**tpmon_client_app** - TP モニター・クライアント・アプリケーション名
TPMON_ACC_STR	VARCHAR(200)	**tpmon_acc_str** - TP モニター・クライアント・アカウンティング・ストリング

SNAPSHOT_BP

バッファー・プール・スナップショットからの情報を戻します。

注: この表関数は使用すべきではなく、 1203 ページの『SNAP_GET_BP 表関数 - bufferpool 論理グループのスナップショット情報の検索』 に置き換えられました。

```
►►—SNAPSHOT_BP—(—dbname—,—dbpartitionnum—)———————————————————►◄
```

スキーマは SYSPROC です。

dbname
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(255) の入力引数。"Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。データベース・インスタンス下のすべてのデータベースからスナップショットを取る場合は、 NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションに -1、またはすべてのアクティブなデータベース・パーティションに -2 を指定します。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

どちらのパラメーターも NULL に設定された場合は、対応するスナップショット API 要求タイプの SNAPSHOT_FILEW ストアード・プロシージャーによって以前にファイルが作成されていない場合にのみ、スナップショットがとられます。

この関数より、次に示されている表が戻されます。

表 310. SNAPSHOT_BP 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
POOL_DATA_L_READS	BIGINT	**pool_data_l_reads** - バッファー・プール・データの論理読み取り
POOL_DATA_P_READS	BIGINT	**pool_data_p_reads** - バッファー・プール・データの物理読み取り
POOL_DATA_WRITES	BIGINT	**pool_data_writes** - バッファー・プールへのデータの書き込み
POOL_INDEX_L_READS	BIGINT	**pool_index_l_reads** - バッファー・プール索引の論理読み取り
POOL_INDEX_P_READS	BIGINT	**pool_index_p_reads** - バッファー・プール索引の物理読み取り
POOL_INDEX_WRITES	BIGINT	**pool_index_writes** - バッファー・プール索引の書き込み
POOL_READ_TIME	BIGINT	**pool_read_time** - バッファー・プール物理読み取り時間の合計
POOL_WRITE_TIME	BIGINT	**pool_write_time** - バッファー・プール物理書き込み時間の合計
POOL_ASYNC_DATA_READS	BIGINT	**pool_async_data_reads** - バッファー・プール非同期データ読み取り
POOL_ASYNC_DATA_WRITES	BIGINT	**pool_async_data_writes** - バッファー・プール非同期データ書き込み
POOL_ASYNC_INDEX_WRITES	BIGINT	**pool_async_index_writes** - バッファー・プール非同期索引書き込み
POOL_ASYNC_READ_TIME	BIGINT	**pool_async_read_time** - バッファー・プール非同期読み取り時間
POOL_ASYNC_WRITE_TIME	BIGINT	**pool_async_write_time** - バッファー・プール非同期書き込み時間
POOL_ASYNC_DATA_READ_REQS	BIGINT	**pool_async_data_read_reqs** - バッファー・プール非同期読み取り要求

表 310. SNAPSHOT_BP 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
DIRECT_READS	BIGINT	**direct_reads** - データベースからの直接読み取り
DIRECT_WRITES	BIGINT	**direct_writes** - データベースへの直接書き込み
DIRECT_READ_REQS	BIGINT	**direct_read_reqs** - 直接読み取り要求
DIRECT_WRITE_REQS	BIGINT	**direct_write_reqs** - 直接書き込み要求
DIRECT_READ_TIME	BIGINT	**direct_read_time** - 直接読み取り時間
DIRECT_WRITE_TIME	BIGINT	**direct_write_time** - 直接書き込み時間
POOL_ASYNC_INDEX_READS	BIGINT	**pool_async_index_reads** - バッファー・プール非同期索引読み取り
POOL_DATA_TO_ESTORE	BIGINT	**pool_data_to_estore** ESTORE モニター・エレメントは廃止されています。廃止されたモニター・エレメントには NULL 値が戻されます。
POOL_INDEX_TO_ESTORE	BIGINT	**pool_index_to_estore** ESTORE モニター・エレメントは廃止されています。廃止されたモニター・エレメントには NULL 値が戻されます。
POOL_INDEX_FROM_ESTORE	BIGINT	**pool_index_from_estore** ESTORE モニター・エレメントは廃止されています。廃止されたモニター・エレメントには NULL 値が戻されます。
POOL_DATA_FROM_ESTORE	BIGINT	**pool_data_from_estore** ESTORE モニター・エレメントは廃止されています。廃止されたモニター・エレメントには NULL 値が戻されます。
UNREAD_PREFETCH_PAGES	BIGINT	**unread_prefetch_pages** - 読み取り不能プリフェッチ・ページ
FILES_CLOSED	BIGINT	**files_closed** - 閉じられたデータベース・ファイル
BP_NAME	VARCHAR(128)	**bp_name** - バッファー・プール名
DB_NAME	VARCHAR(128)	**db_name** - データベース名
DB_PATH	VARCHAR(1024)	**db_path** - データベース・パス
INPUT_DB_ALIAS	VARCHAR(128)	**input_db_alias** - 入力データベース別名

SNAPSHOT_CONTAINER

表スペース・スナップショットからのコンテナー構成情報を戻します。

注: この表関数は使用すべきではなく、 670 ページの『SNAPCONTAINER 管理ビューおよび SNAP_GET_CONTAINER_V91 表関数 - tablespace_container 論理データ・グループ・スナップショット情報の検索』 に置き換えられました。

```
►►—SNAPSHOT_CONTAINER—(—dbname—,—dbpartitionnum—)—————————————————►◄
```

スキーマは SYSPROC です。

dbname
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(255) の入力引数。"Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションに -1、またはすべてのアクティブなデータベース・パーティションに -2 を指定します。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

どちらのパラメーターも NULL に設定された場合は、対応するスナップショット API 要求タイプの SNAPSHOT_FILEW ストアード・プロシージャーによって以前にファイルが作成されていない場合にのみ、スナップショットがとられます。

この関数より、次に示されている表が戻されます。

表 311. SNAPSHOT_CONTAINER 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
TABLESPACE_ID	BIGINT	**tablespace_id** - 表スペース ID
TABLESPACE_NAME	VARCHAR(128)	**tablespace_name** - 表スペース名
CONTAINER_ID	BIGINT	**container_id** - コンテナー ID
CONTAINER_NAME	VARCHAR(256)	**container_name** - コンテナー名
CONTAINER_TYPE	SMALLINT	**container_type** - コンテナー・タイプ

表 311. SNAPSHOT_CONTAINER 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
TOTAL_PAGES	BIGINT	**container_total_pages** - コンテナー内の合計ページ数
USABLE_PAGES	BIGINT	**container_usable_pages** - コンテナー内の使用可能なページ数
ACCESSIBLE	BIGINT	**container_accessible** - コンテナーのアクセス可能性
STRIPE_SET	BIGINT	**container_stripe_set** - ストライプ・セット

SNAPSHOT_DATABASE

データベース・スナップショットからの情報を戻します。

注: この表関数は使用すべきではなく、 1219 ページの『SNAP_GET_DB_V91 table function - dbase 論理グループからのスナップショット情報の検索』 に置き換えられました。

スキーマは SYSPROC です。

dbname
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(255) の入力引数。"Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。データベース・インスタンス下のすべてのデータベースからスナップショットを取る場合は、 NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションに -1、またはすべてのアクティブなデータベース・パーティションに -2 を指定します。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

どちらのパラメーターも NULL に設定された場合は、対応するスナップショット API 要求タイプの SNAPSHOT_FILEW ストアード・プロシージャーによって以前にファイルが作成されていない場合にのみ、スナップショットがとられます。

この関数より、次に示されている表が戻されます。

表 312. SNAPSHOT_DATABASE 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
SEC_LOG_USED_TOP	BIGINT	**sec_log_used_top** - 使用された最大 2 次ログ・スペース
TOT_LOG_USED_TOP	BIGINT	**tot_log_used_top** - 使用された最大合計ログ・スペース
TOTAL_LOG_USED	BIGINT	**total_log_used** - 使用されているログ・スペースの合計
TOTAL_LOG_AVAILABLE	BIGINT	**total_log_available** - 使用可能なログ合計
ROWS_READ	BIGINT	**rows_read** - 読み取り行数
POOL_DATA_L_READS	BIGINT	**pool_data_l_reads** - バッファー・プール・データの論理読み取り
POOL_DATA_P_READS	BIGINT	**pool_data_p_reads** - バッファー・プール・データの物理読み取り
POOL_DATA_WRITES	BIGINT	**pool_data_writes** - バッファー・プールへのデータの書き込み
POOL_INDEX_L_READS	BIGINT	**pool_index_l_reads** - バッファー・プール索引の論理読み取り
POOL_INDEX_P_READS	BIGINT	**pool_index_p_reads** - バッファー・プール索引の物理読み取り
POOL_INDEX_WRITES	BIGINT	**pool_index_writes** - バッファー・プール索引の書き込み
POOL_READ_TIME	BIGINT	**pool_read_time** - バッファー・プール物理読み取り時間の合計
POOL_WRITE_TIME	BIGINT	**pool_write_time** - バッファー・プール物理書き込み時間の合計
POOL_ASYNC_INDEX_READS	BIGINT	**pool_async_index_reads** - バッファー・プール非同期索引読み取り
POOL_DATA_TO_ESTORE	BIGINT	**pool_data_to_estore** ESTORE モニター・エレメントは廃止されています。廃止されたモニター・エレメントには NULL 値が戻されます。
POOL_INDEX_TO_ESTORE	BIGINT	**pool_index_to_estore** ESTORE モニター・エレメントは廃止されています。廃止されたモニター・エレメントには NULL 値が戻されます。
POOL_INDEX_FROM_ESTORE	BIGINT	**pool_index_from_estore** ESTORE モニター・エレメントは廃止されています。廃止されたモニター・エレメントには NULL 値が戻されます。

表 312. SNAPSHOT_DATABASE 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
POOL_DATA_FROM_ESTORE	BIGINT	**pool_data_from_estore** ESTORE モニター・エレメントは廃止されています。廃止されたモニター・エレメントには NULL 値が戻されます。
POOL_ASYNC_DATA_READS	BIGINT	**pool_async_data_reads** - バッファー・プール非同期データ読み取り
POOL_ASYNC_DATA_WRITES	BIGINT	**pool_async_data_writes** - バッファー・プール非同期データ書き込み
POOL_ASYNC_INDEX_WRITES	BIGINT	**pool_async_index_writes** - バッファー・プール非同期索引書き込み
POOL_ASYNC_READ_TIME	BIGINT	**pool_async_read_time** - バッファー・プール非同期読み取り時間
POOL_ASYNC_WRITE_TIME	BIGINT	**pool_async_write_time** - バッファー・プール非同期書き込み時間
POOL_ASYNC_DATA_ READ_REQS	BIGINT	**pool_async_data_read_reqs** - バッファー・プール非同期読み取り要求
DIRECT_READS	BIGINT	**direct_reads** - データベースからの直接読み取り
DIRECT_WRITES	BIGINT	**direct_writes** - データベースへの直接書き込み
DIRECT_READ_REQS	BIGINT	**direct_read_reqs** - 直接読み取り要求
DIRECT_WRITE_REQS	BIGINT	**direct_write_reqs** - 直接書き込み要求
DIRECT_READ_TIME	BIGINT	**direct_read_time** - 直接読み取り時間
DIRECT_WRITE_TIME	BIGINT	**direct_write_time** - 直接書き込み時間
UNREAD_PREFETCH_PAGES	BIGINT	**unread_prefetch_pages** - 読み取り不能プリフェッチ・ページ
FILES_CLOSED	BIGINT	**files_closed** - 閉じられたデータベース・ファイル
POOL_LSN_GAP_CLNS	BIGINT	**pool_lsn_gap_clns** - 起動されたバッファー・プール・ログ・スペース・クリーナー
POOL_DRTY_PG_STEAL_CLNS	BIGINT	**pool_drty_pg_steal_clns** - 起動されたバッファー・プール・ビクティム・ページ・クリーナー
POOL_DRTY_PG_THRSH_CLNS	BIGINT	**pool_drty_pg_thrsh_clns** - 起動されたバッファー・プールしきい値クリーナー
LOCKS_HELD	BIGINT	**locks_held** - ロック保持数
LOCK_WAITS	BIGINT	**lock_waits** - ロック待機数

表 312. SNAPSHOT_DATABASE 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
LOCK_WAIT_TIME	BIGINT	**lock_wait_time** - ロック待機中の時間
LOCK_LIST_IN_USE	BIGINT	**lock_list_in_use** - 使用中のロック・リスト・メモリーの合計
DEADLOCKS	BIGINT	**deadlocks** - デッドロック検出数
LOCK_ESCALS	BIGINT	**lock_escals** - ロック・エスカレーション数
X_LOCK_ESCALS	BIGINT	**x_lock_escals** - 排他ロック・エスカレーション数
LOCKS_WAITING	BIGINT	**locks_waiting** - ロックで待機中の現行エージェント
SORT_HEAP_ALLOCATED	BIGINT	**sort_heap_allocated** - 割り振られたソート・ヒープの合計
TOTAL_SORTS	BIGINT	**total_sorts** - ソート合計
TOTAL_SORT_TIME	BIGINT	**total_sort_time** - ソート時間合計
SORT_OVERFLOWS	BIGINT	**sort_overflows** - ソート・オーバーフロー
ACTIVE_SORTS	BIGINT	**active_sorts** - アクティブ・ソート
COMMIT_SQL_STMTS	BIGINT	**commit_sql_stmts** - 試行されたコミット・ステートメント
ROLLBACK_SQL_STMTS	BIGINT	**rollback_sql_stmts** - 試行されたロールバック・ステートメント
DYNAMIC_SQL_STMTS	BIGINT	**dynamic_sql_stmts** - 試行された動的 SQL ステートメント
STATIC_SQL_STMTS	BIGINT	**static_sql_stmts** - 試行された静的 SQL ステートメント
FAILED_SQL_STMTS	BIGINT	**failed_sql_stmts** - 失敗したステートメント操作
SELECT_SQL_STMTS	BIGINT	**select_sql_stmts** - 実行された選択 SQL ステートメント
DDL_SQL_STMTS	BIGINT	ddl_sql_stmts - データ定義言語 (DDL) SQL ステートメント
UID_SQL_STMTS	BIGINT	**uid_sql_stmts** - 実行された更新/挿入/削除 SQL ステートメント
INT_AUTO_REBINDS	BIGINT	**int_auto_rebinds** - 内部自動再バインド
INT_ROWS_DELETED	BIGINT	**int_rows_deleted** - 削除された内部行数
INT_ROWS_UPDATED	BIGINT	**int_rows_updated** - 更新された内部行数
INT_COMMITS	BIGINT	**int_commits** - 内部コミット数
INT_ROLLBACKS	BIGINT	**int_rollbacks** - 内部ロールバック数

表 312. SNAPSHOT_DATABASE 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
INT_DEADLOCK_ROLLBACKS	BIGINT	**int_deadlock_rollbacks** - デッドロックによる内部ロールバック数
ROWS_DELETED	BIGINT	**rows_deleted** - 削除行数
ROWS_INSERTED	BIGINT	**rows_inserted** - 挿入行数
ROWS_UPDATED	BIGINT	**rows_updated** - 更新行数
ROWS_SELECTED	BIGINT	**rows_selected** - 選択行数
BINDS_PRECOMPILES	BIGINT	**binds_precompiles** - 試行されたバインド/プリコンパイル
TOTAL_CONS	BIGINT	**total_cons** - データベース・アクティブ化以降の接続
APPLS_CUR_CONS	BIGINT	**appls_cur_cons** - 現在接続されているアプリケーション
APPLS_IN_DB2	BIGINT	**appls_in_db2** - データベースで現在実行中のアプリケーション
SEC_LOGS_ALLOCATED	BIGINT	**sec_logs_allocated** - 現在割り振られている 2 次ログ
DB_STATUS	BIGINT	**db_status** - データベース状況
LOCK_TIMEOUTS	BIGINT	**lock_timeouts** - ロック・タイムアウト数
CONNECTIONS_TOP	BIGINT	**connections_top** - 同時接続の最大数
DB_HEAP_TOP	BIGINT	**db_heap_top** - 割り振られた最大データベース・ヒープ
INT_ROWS_INSERTED	BIGINT	**int_rows_inserted** - 挿入された内部行数
LOG_READS	BIGINT	**log_reads** - 読み取られたログ・ページの数
LOG_WRITES	BIGINT	**log_writes** - 書き込まれたログ・ページの数
PKG_CACHE_LOOKUPS	BIGINT	**pkg_cache_lookups** - パッケージ・キャッシュ参照
PKG_CACHE_INSERTS	BIGINT	**pkg_cache_inserts** - パッケージ・キャッシュ挿入
CAT_CACHE_LOOKUPS	BIGINT	**cat_cache_lookups** - カタログ・キャッシュ参照数
CAT_CACHE_INSERTS	BIGINT	**cat_cache_inserts** - カタログ・キャッシュ挿入数
CAT_CACHE_OVERFLOWS	BIGINT	**cat_cache_overflows** - カタログ・キャッシュ・オーバーフロー数
CAT_CACHE_HEAP_FULL	BIGINT	**cat_cache_overflows** - カタログ・キャッシュ・オーバーフロー数

表 312. SNAPSHOT_DATABASE 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
CATALOG_PARTITION	SMALLINT	**catalog_node** - カタログ・ノード番号
TOTAL_SEC_CONS	BIGINT	**total_sec_cons** - 2 次接続
NUM_ASSOC_AGENTS	BIGINT	**num_assoc_agents** - 関連したエージェント数
AGENTS_TOP	BIGINT	**agents_top** - 作成されたエージェントの数
COORD_AGENTS_TOP	BIGINT	**coord_agents_top** - コーディネーター・エージェント最大数
PREFETCH_WAIT_TIME	BIGINT	**prefetch_wait_time** - プリフェッチ待ち時間
APPL_SECTION_LOOKUPS	BIGINT	**appl_section_lookups** - セクションの参照回数
APPL_SECTION_INSERTS	BIGINT	**appl_section_inserts** - セクション挿入数
TOTAL_HASH_JOINS	BIGINT	**total_hash_joins** - ハッシュ結合の合計
TOTAL_HASH_LOOPS	BIGINT	total_hash_loops - ハッシュ・ループの合計
HASH_JOIN_OVERFLOWS	BIGINT	**hash_join_overflows** - ハッシュ結合のオーバーフロー
HASH_JOIN_SMALL_ OVERFLOWS	BIGINT	**hash_join_small_overflows** - ハッシュ結合の短精度オーバーフロー
PKG_CACHE_NUM_ OVERFLOWS	BIGINT	**pkg_cache_num_overflows** - パッケージ・キャッシュ・オーバーフロー数
PKG_CACHE_SIZE_TOP	BIGINT	**pkg_cache_size_top** - パッケージ・キャッシュ最高水準点
DB_CONN_TIME	TIMESTAMP	**db_conn_time** - データベース・アクティブ化タイム・スタンプ
SQLM_ELM_LAST_RESET	TIMESTAMP	**last_reset** - 最後のリセット・タイム・スタンプ
SQLM_ELM_LAST_BACKUP	TIMESTAMP	**last_backup** - 最終バックアップ・タイム・スタンプ
APPL_CON_TIME	TIMESTAMP	**appl_con_time** - 接続要求開始タイム・スタンプ
DB_LOCATION	INTEGER	**db_location** - データベース・ロケーション
SERVER_PLATFORM	INTEGER	**server_platform** - サーバーのオペレーティング・システム

表 312. SNAPSHOT_DATABASE 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
APPL_ID_OLDEST_XACT	BIGINT	**appl_id_oldest_xact** - 最も古いトランザクションを使用するアプリケーション
CATALOG_PARTITION_NAME	VARCHAR(128)	**catalog_node_name** - カタログ・ノード・ネットワーク名
INPUT_DB_ALIAS	VARCHAR(128)	**input_db_alias** - 入力データベース別名
DB_NAME	VARCHAR(128)	**db_name** - データベース名
DB_PATH	VARCHAR(1024)	**db_path** - データベース・パス

SNAPSHOT_DBM

DB2 データベース・マネージャーのスナップショットからの情報を戻します。

注: この表関数は使用すべきではなく、 1216 ページの『SNAP_GET_DBM 表関数 - dbm 論理グループ・スナップショット情報の検索』 に置き換えられました。

```
►►—SNAPSHOT_DBM—(—dbpartitionnum—)————————————————————————►◄
```

スキーマは SYSPROC です。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションに -1、またはすべてのアクティブなデータベース・パーティションに -2 を指定します。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

NULL 値が指定された場合は、対応するスナップショット API 要求タイプの SNAPSHOT_FILEW ストアード・プロシージャーによって以前にファイルが作成されていない場合にのみ、スナップショットがとられます。

この関数より、次に示されている表が戻されます。

表 313. SNAPSHOT_DBM 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
SORT_HEAP_ALLOCATED	BIGINT	**sort_heap_allocated** - 割り振られたソート・ヒープの合計
POST_THRESHOLD_SORTS	BIGINT	**post_threshold_sorts** - ポストしきい値ソート

表 313. SNAPSHOT_DBM 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
PIPED_SORTS_REQUESTED	BIGINT	**piped_sorts_requested** - 要求されたパイプ・ソート数
PIPED_SORTS_ACCEPTED	BIGINT	**piped_sorts_accepted** - 受け入れられたパイプ・ソート
REM_CONS_IN	BIGINT	**rem_cons_in** - データベース・マネージャーへのリモート接続
REM_CONS_IN_EXEC	BIGINT	**rem_cons_in_exec** - データベース・マネージャーで実行中のリモート接続 : モニター・エレメント
LOCAL_CONS	BIGINT	**local_cons** - ローカル接続
LOCAL_CONS_IN_EXEC	BIGINT	**local_cons_in_exec** - データベース・マネージャーで実行中のローカル接続 : モニター・エレメント
CON_LOCAL_DBASES	BIGINT	**con_local_dbases** - 現行接続を使用したローカル・データベース
AGENTS_REGISTERED	BIGINT	**agents_registered** - 登録済みエージェント
AGENTS_WAITING_ON_TOKEN	BIGINT	**agents_waiting_on_token** - トークン待ちエージェント
DB2_STATUS	BIGINT	**db_status** - データベース状況
AGENTS_REGISTERED_TOP	BIGINT	**agents_registered_top** - エージェント最大登録数
AGENTS_WAITING_TOP	BIGINT	**agents_waiting_top** - エージェント最大待機数
COMM_PRIVATE_MEM	BIGINT	**comm_private_mem** - コミット済み専用メモリー
IDLE_AGENTS	BIGINT	**idle_agents** - アイドル・エージェント数
AGENTS_FROM_POOL	BIGINT	**agents_from_pool** - プールから割り当てられたエージェント
AGENTS_CREATED_ EMPTY_POOL	BIGINT	**agents_created_empty_pool** - エージェント・プールが空のために作成されたエージェント
COORD_AGENTS_TOP	BIGINT	**coord_agents_top** - コーディネーター・エージェント最大数
MAX_AGENT_OVERFLOWS	BIGINT	**max_agent_overflows** - 最大エージェント・オーバーフロー回数
AGENTS_STOLEN	BIGINT	**agents_stolen** - スチールされたエージェント
GW_TOTAL_CONS	BIGINT	**gw_total_cons** - DB2 Connect の接続試行合計回数
GW_CUR_CONS	BIGINT	**gw_cur_cons** - DB2 Connect の現在の接続数

表 313. SNAPSHOT_DBM 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
GW_CONS_WAIT_HOST	BIGINT	**gw_cons_wait_host** - ホストの応答を待機している接続の数
GW_CONS_WAIT_CLIENT	BIGINT	**gw_cons_wait_client** - クライアントの要求送信を待機している接続の数
POST_THRESHOLD_ HASH_JOINS	BIGINT	**post_threshold_hash_joins** - ハッシュ結合のしきい値
INACTIVE_GW_AGENTS	BIGINT	**idle_agents** - アイドル・エージェント数
NUM_GW_CONN_SWITCHES	BIGINT	**num_gw_conn_switches** - 接続切り替え回数
DB2START_TIME	TIMESTAMP	**db2start_time** - データベース・マネージャー開始タイム・スタンプ
LAST_RESET	TIMESTAMP	**last_reset** - 最後のリセット・タイム・スタンプ

SNAPSHOT_DYN_SQL

動的 SQL スナップショットからの情報を戻します。この関数は SQLCACHE_SNAPSHOT 関数に代わるものですが、互換性の理由から、SQLCACHE_SNAPSHOT も依然として使用可能です。

注: この表関数は使用すべきではなく、 1243 ページの『SNAP_GET_DYN_SQL_V91 表関数 - dynsql 論理グループのスナップショット情報の検索』 に置き換えられました。

```
►►—SNAPSHOT_DYN_SQL—(—dbname—,—dbpartitionnum—)———————————————————►◄
```

スキーマは SYSPROC です。

dbname
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(255) の入力引数。"Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションに -1、またはすべてのアクティブなデータベース・パーティションに -2 を指定します。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

どちらのパラメーターも NULL に設定された場合は、対応するスナップショット API 要求タイプの SNAPSHOT_FILEW ストアード・プロシージャーによって以前にファイルが作成されていない場合にのみ、スナップショットがとられます。

この関数より、次に示されている表が戻されます。

表 314. SNAPSHOT_DYN_SQL 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
ROWS_READ	BIGINT	**rows_read** - 読み取り行数
ROWS_WRITTEN	BIGINT	**rows_written** - 書き込み行数
NUM_EXECUTIONS	BIGINT	**num_executions** - ステートメント実行回数
NUM_COMPILATIONS	BIGINT	**num_compilations** - ステートメント・コンパイル数
PREP_TIME_WORST	BIGINT	**prep_time_worst** - ステートメント最長準備時間
PREP_TIME_BEST	BIGINT	**prep_time_best** - ステートメント最短準備時間
INT_ROWS_DELETED	BIGINT	**int_rows_deleted** - 削除された内部行数
INT_ROWS_INSERTED	BIGINT	**int_rows_inserted** - 挿入された内部行数
INT_ROWS_UPDATED	BIGINT	**int_rows_updated** - 更新された内部行数
STMT_SORTS	BIGINT	**stmt_sorts** - ステートメント・ソート回数
TOTAL_EXEC_TIME	BIGINT	**total_exec_time** - ステートメント実行の経過時間
TOTAL_SYS_CPU_TIME	BIGINT	**total_sys_cpu_time** - ステートメントのシステム CPU の合計
TOTAL_USR_CPU_TIME	BIGINT	**total_usr_cpu_time** - ステートメントのユーザー CPU の合計
STMT_TEXT	CLOB(16M)[1]	**stmt_text** - SQL 動的ステートメント・テキスト

[1] STMT_TEXT は CLOB(16M) として定義されていますが、これは単に将来の拡張に備えるために過ぎません。ステートメント・テキストの実際の出力は 64K で切り捨てられます。

SNAPSHOT_FCM

注: この表関数は使用すべきではなく、 710 ページの『SNAPFCM 管理ビューおよび SNAP_GET_FCM 表関数 - fcm 論理データ・グループ・スナップショット情報の検索』 に置き換えられました。

スキーマは SYSPROC です。

SNAPSHOT_FCM 関数は、高速コミュニケーション・マネージャー (FCM) に関するデータベース・マネージャー・レベル情報を戻します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションに -1、またはすべてのアクティブなデータベース・パーティションに -2 を指定します。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

この関数より、次に示されている表が戻されます。

表 315. SNAPSHOT_FCM 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
BUFF_FREE	BIGINT	**buff_free** - 現在空いている FCM バッファー
BUFF_FREE_BOTTOM	BIGINT	**buff_free_bottom** - 空き FCM バッファーの最小数
MA_FREE	BIGINT	**ma_free** モニター・エレメントは廃止されています。廃止されたモニター・エレメントには NULL 値が戻されます。
MA_FREE_BOTTOM	BIGINT	**ma_free_bottom** モニター・エレメントは廃止されています。廃止されたモニター・エレメントには NULL 値が戻されます。
CE_FREE	BIGINT	**ce_free monitor** モニター・エレメントは廃止されています。廃止されたモニター・エレメントには NULL 値が戻されます。

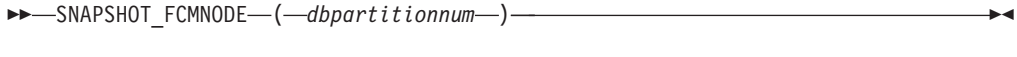
表 315. SNAPSHOT_FCM 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
CE_FREE_BOTTOM	BIGINT	**ce_free_bottom** モニター・エレメントは廃止されています。廃止されたモニター・エレメントには NULL 値が戻されます。
RB_FREE	BIGINT	**rb_free monitor** モニター・エレメントは廃止されています。廃止されたモニター・エレメントには NULL 値が戻されます。
RB_FREE_BOTTOM	BIGINT	**rb_free_bottom** モニター・エレメントは廃止されています。廃止されたモニター・エレメントには NULL 値が戻されます。
PARTITION_NUMBER	SMALLINT	**node_number** - ノード番号

SNAPSHOT_FCMNODE

データベース・マネージャーの高速コミュニケーション・マネージャーのスナップショットから情報を戻します。

注: この表関数は使用すべきではなく、 712 ページの『SNAPFCM_PART 管理ビューおよび SNAP_GET_FCM_PART 表関数 - fcm_node 論理データ・グループ・スナップショット情報の検索』 に置き換えられました。



スキーマは SYSPROC です。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションに -1、またはすべてのアクティブなデータベース・パーティションに -2 を指定します。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

NULL 値が指定された場合は、対応するスナップショット API 要求タイプの SNAPSHOT_FILEW ストアード・プロシージャーによって以前にファイルが作成されていない場合にのみ、スナップショットがとられます。

この関数より、次に示されている表が戻されます。

表 316. SNAPSHOT_FCMNODE 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
CONNECTION_STATUS	BIGINT	**connection_status** - 接続状況
TOTAL_BUFFERS_SENT	BIGINT	**total_buffers_sent** - 送信された FCM バッファーの合計
TOTAL_BUFFERS_RCVD	BIGINT	**total_buffers_rcvd** - 受信された FCM バッファーの合計
PARTITION_NUMBER	SMALLINT	**node_number** - ノード番号

SNAPSHOT_FILEW

注: このプロシージャーは使用すべきではなく、 788 ページの『SNAP_WRITE_FILE プロシージャー』によって置き換えられました。

```
►►—SNAPSHOT_FILEW—(—requestType—,—dbname—,—dbpartitionnum—)———————————►◄
```

スキーマは SYSPROC です。

SNAPSHOT_FILEW プロシージャーはシステム・スナップショット・データを、インスタンス・ディレクトリーの tmp サブディレクトリーにあるファイルに書き込みます。 SNAPSHOT_FILEW プロシージャーを実行するには、ユーザーに SYSADM、SYSCTRL、または SYSMAINT 権限が必要です。保存されたスナップショットは、スナップショット関数への入力として NULL 値を渡すことにより、SYSADM、SYSCTRL、または SYSMAINT 権限のないユーザーでも読み取れます。

requestType
有効なスナップショット要求タイプ (sqlmon.h で定義されている) を指定する、タイプ SMALLINT の入力引数。

dbname
このプロシージャーを呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(128) の入力引数。現在接続されているデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ SMALLINT の入力引数。現行のデータベース・パーティションに -1、またはすべてのアクティブなデータベース・パーティションに -2 を指定します。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

例: 1 の要求タイプ (SQLMA_DB2 に相当) を指定し、現在接続されているデータベースおよび現行のデータベース・パーティションをデフォルトにすることで、データベース・マネージャー情報のスナップショットをとります。

```
CALL SNAPSHOT_FILEW (1, CAST (NULL AS VARCHAR(128)), CAST (NULL AS SMALLINT))
```

この場合、スナップショット・データは、UNIX オペレーティング・システムでは、インスタンス・ディレクトリーの /tmp/SQLMA_DB2.dat に、Windows オペレーティング・システムでは、インスタンス・ディレクトリーの ¥tmp¥SQLMA_DB2.dat に書き込まれます。

SNAPSHOT_LOCK

ロック・スナップショットからの情報を戻します。

注: この表関数は使用すべきではなく、 720 ページの『SNAPLOCK 管理ビューおよび SNAP_GET_LOCK 表関数 - lock 論理データ・グループのスナップショット情報の検索』 に置き換えられました。

```
►►—SNAPSHOT_LOCK—(—dbname—,—dbpartitionnum—)————————————————————►◄
```

スキーマは SYSPROC です。

dbname
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(255) の入力引数。"Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションに -1、またはすべてのアクティブなデータベース・パーティションに -2 を指定します。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

どちらのパラメーターも NULL に設定された場合は、対応するスナップショット API 要求タイプの SNAPSHOT_FILEW ストアード・プロシージャーによって以前にファイルが作成されていない場合にのみ、スナップショットがとられます。

この関数より、次に示されている表が戻されます。

表 317. SNAPSHOT_LOCK 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。

表 317. SNAPSHOT_LOCK 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
AGENT_ID	BIGINT	**agent_id** - アプリケーション・ハンドル (エージェント ID)
TABLE_FILE_ID	BIGINT	**table_file_id** - 表ファイル ID
LOCK_OBJECT_TYPE	BIGINT	**lock_object_type** - 待機中のロック対象タイプ
LOCK_MODE	BIGINT	**lock_mode** - ロック・モード
LOCK_STATUS	BIGINT	**lock_status** - ロック状況
LOCK_OBJECT_NAME	BIGINT	**lock_object_name** - ロック対象名
PARTITION_NUMBER	SMALLINT	**node_number** - ノード番号
LOCK_ESCALATION	SMALLINT	**lock_escalation** - ロック・エスカレーション
TABLE_NAME	VARCHAR(128)	**table_name** - 表名
TABLE_SCHEMA	VARCHAR(128)	**table_schema** - 表スキーマ名
TABLESPACE_NAME	VARCHAR(128)	**tablespace_name** - 表スペース名

SNAPSHOT_LOCKWAIT

アプリケーション・スナップショットからロック待機情報を戻します。

注: この表関数は使用すべきではなく、 726 ページの『SNAPLOCKWAIT 管理ビューおよび SNAP_GET_LOCKWAIT 表関数 - lockwait 論理データ・グループのスナップショット情報の検索』 に置き換えられました。

```
►►—SNAPSHOT_LOCKWAIT—(—dbname—,—dbpartitionnum—)———————————————►◄
```

スキーマは SYSPROC です。

dbname
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(255) の入力引数。"Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。データベース・インスタンス下のすべてのデータベースからスナップショットを取る場合は、 NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションに -1、またはすべてのアクティブなデータベース・パーティションに -2 を指定します。アクティブなデータベ

ース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

どちらのパラメーターも NULL に設定された場合は、対応するスナップショット API 要求タイプの SNAPSHOT_FILEW ストアード・プロシージャーによって以前にファイルが作成されていない場合にのみ、スナップショットがとられます。

この関数より、次に示されている表が戻されます。

表 318. SNAPSHOT_LOCKWAIT 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
AGENT_ID	BIGINT	**agent_id** - アプリケーション・ハンドル (エージェント ID)
SUBSECTION_NUMBER	BIGINT	**ss_number** - サブセクション番号
LOCK_MODE	BIGINT	**lock_mode** - ロック・モード
LOCK_OBJECT_TYPE	BIGINT	**lock_object_type** - 待機中のロック対象タイプ
AGENT_ID_HOLDING_LK	BIGINT	**agent_id_holding_lock** - ロックを保持しているエージェント ID
LOCK_WAIT_START_TIME	TIMESTAMP	**lock_wait_start_time** - ロック待機開始タイム・スタンプ
LOCK_MODE_REQUESTED	BIGINT	**lock_mode_requested** - 要求されているロック・モード
PARTITION_NUMBER	SMALLINT	**node_number** - ノード番号
LOCK_ESCALATION	SMALLINT	**lock_escalation** - ロック・エスカレーション
TABLE_NAME	VARCHAR(128)	**table_name** - 表名
TABLE_SCHEMA	VARCHAR(128)	**table_schema** - 表スキーマ名
TABLESPACE_NAME	VARCHAR(128)	**tablespace_name** - 表スペース名
APPL_ID_HOLDING_LK	VARCHAR(128)	**appl_id_holding_lk** - ロックを保持しているアプリケーション ID

SNAPSHOT_QUIESCERS

注: この表関数は使用すべきではなく、 772 ページの『SNAPTBSP_QUIESCER 管理ビューおよび SNAP_GET_TBSP_QUIESCER 表関数 - quiescer 表スペース・スナップショット情報の検索』 に置き換えられました。

```
►►—SNAPSHOT_QUIESCERS—(—dbname—,—dbpartitionnum—)—————————————————►◄
```

スキーマは SYSPROC です。

SNAPSHOT_QUIESCERS 関数は、静止プログラムに関する情報を表スペース・スナップショットから戻します。

dbname
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(255) の入力引数。"Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションに -1、またはすべてのアクティブなデータベース・パーティションに -2 を指定します。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

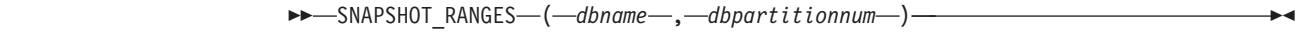
この関数より、次に示されている表が戻されます。

表 319. SNAPSHOT_QUIESCERS 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
TABLESPACE_NAME	VARCHAR(128)	**tablespace_name** - 表スペース名
QUIESCER_TBS_ID	BIGINT	**quiescer_ts_id** - 静止プログラム表スペース ID
QUIESCER_OBJ_ID	BIGINT	**quiescer_obj_id** - 静止プログラム・オブジェクト ID
QUIESCER_AUTH_ID	BIGINT	**quiescer_auth_id** - 静止プログラム・ユーザー許可 ID
QUIESCER_AGENT_ID	BIGINT	**quiescer_agent_id** - 静止プログラム・エージェント ID
QUIESCER_STATE	BIGINT	**quiescer_state** - 静止プログラムの状態

SNAPSHOT_RANGES

注: この表関数は使用すべきではなく、 777 ページの『SNAPTBSP_RANGE 管理ビューおよび SNAP_GET_TBSP_RANGE 表関数 - 範囲スナップショット情報の検索』 に置き換えられました。



スキーマは SYSPROC です。

SNAPSHOT_RANGES 関数は、範囲スナップショットから情報を戻します。

dbname
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(255) の入力引数。"Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションに -1、またはすべてのアクティブなデータベース・パーティションに -2 を指定します。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

この関数より、次に示されている表が戻されます。

表 320. SNAPSHOT_RANGES 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
TABLESPACE_ID	BIGINT	**tablespace_id** - 表スペース ID
TABLESPACE_NAME	VARCHAR(128)	**tablespace_name** - 表スペース名
RANGE_NUMBER	BIGINT	**range_number** - 範囲番号
RANGE_STRIPE_SET_NUMBER	BIGINT	**range_stripe_set_number** - ストライプ・セット番号
RANGE_OFFSET	BIGINT	**range_offset** - 範囲オフセット
RANGE_MAX_PAGE	BIGINT	**range_max_page_number** - 範囲内の最大ページ
RANGE_MAX_EXTENT	BIGINT	**range_max_extent** - 範囲内の最大エクステント
RANGE_START_STRIPE	BIGINT	**range_start_stripe** - 開始ストライプ
RANGE_END_STRIPE	BIGINT	**range_end_stripe** - 終了ストライプ
RANGE_ADJUSTMENT	BIGINT	**range_adjustment** - 範囲調整

表 320. SNAPSHOT_RANGES 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
RANGE_NUM_CONTAINER	BIGINT	**range_num_containers** - 範囲内コンテナーの数
RANGE_CONTAINER_ID	BIGINT	**range_container_id** - 範囲コンテナー

SNAPSHOT_STATEMENT

アプリケーション・スナップショットからステートメントに関する情報を戻します。

注: この表関数は使用すべきではなく、 733 ページの『SNAPSTMT 管理ビューおよび SNAP_GET_STMT 表関数 - ステートメント・スナップショット情報の検索』に置き換えられました。

スキーマは SYSPROC です。

dbname
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(255) の入力引数。"Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。データベース・インスタンス下のすべてのデータベースからスナップショットを取る場合は、 NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションに -1、またはすべてのアクティブなデータベース・パーティションに -2 を指定します。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

どちらのパラメーターも NULL に設定された場合は、対応するスナップショット API 要求タイプの SNAPSHOT_FILEW ストアード・プロシージャーによって以前にファイルが作成されていない場合にのみ、スナップショットがとられます。

この関数より、次に示されている表が戻されます。

表 321. SNAPSHOT_STATEMENT 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。

表 321. SNAPSHOT_STATEMENT 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
AGENT_ID	BIGINT	**agent_id** - アプリケーション・ハンドル (エージェント ID)
ROWS_READ	BIGINT	**rows_read** - 読み取り行数
ROWS_WRITTEN	BIGINT	**rows_written** - 書き込み行数
NUM_AGENTS	BIGINT	**num_agents** - ステートメントで作動しているエージェントの数
AGENTS_TOP	BIGINT	**agents_top** - 作成されたエージェントの数
STMT_TYPE	BIGINT	**stmt_type** - ステートメント・タイプ
STMT_OPERATION	BIGINT	**stmt_operation/operation** - ステートメント操作
SECTION_NUMBER	BIGINT	**section_number** - セクション番号
QUERY_COST_ESTIMATE	BIGINT	**query_cost_estimate** - 照会コストの見積もり
QUERY_CARD_ESTIMATE	BIGINT	**query_card_estimate** - 照会行数の見積もり
DEGREE_PARALLELISM	BIGINT	**degree_parallelism** - 並列処理の度合い
STMT_SORTS	BIGINT	**stmt_sorts** - ステートメント・ソート回数
TOTAL_SORT_TIME	BIGINT	**total_sort_time** - ソート時間合計
SORT_OVERFLOWS	BIGINT	**sort_overflows** - ソート・オーバーフロー
INT_ROWS_DELETED	BIGINT	**int_rows_deleted** - 削除された内部行数
INT_ROWS_UPDATED	BIGINT	**int_rows_updated** - 更新された内部行数
INT_ROWS_INSERTED	BIGINT	**int_rows_inserted** - 挿入された内部行数
FETCH_COUNT	BIGINT	**fetch_count** - 成功した取り出しの数
STMT_START	TIMESTAMP	**stmt_start** - ステートメント操作開始タイム・スタンプ
STMT_STOP	TIMESTAMP	**stmt_stop** - ステートメント操作停止タイム・スタンプ
STMT_USR_CPU_TIME_S	BIGINT	**stmt_usr_cpu_time** - ステートメントに使用されたユーザー CPU 時間 (秒単位)*
STMT_USR_CPU_TIME_MS	BIGINT	**stmt_usr_cpu_time** - ステートメントに使用されたユーザー CPU 時間 (小数部、マイクロ秒単位)*

表 321. SNAPSHOT_STATEMENT 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
STMT_SYS_CPU_TIME_S	BIGINT	**stmt_sys_cpu_time** - ステートメントが使用したシステム CPU 時間 (秒単位)*
STMT_SYS_CPU_TIME_MS	BIGINT	**stmt_sys_cpu_time** - ステートメントが使用したシステム CPU 時間 (小数部、マイクロ秒単位)*
STMT_ELAPSED_TIME_S	BIGINT	**stmt_elapsed_time** - 最新のステートメント経過時間 (秒単位)*
STMT_ELAPSED_TIME_MS	BIGINT	**stmt_elapsed_time** - 最新のステートメント経過時間 (小数部、マイクロ秒単位)*
BLOCKING_CURSOR	SMALLINT	**blocking_cursor** - ブロック・カーソル
STMT_PARTITION_NUMBER	SMALLINT	**stmt_node_number** - ステートメント・ノード
CURSOR_NAME	VARCHAR(128)	**cursor_name** - カーソル名
CREATOR	VARCHAR(128)	**creator** - アプリケーション作成者
PACKAGE_NAME	VARCHAR(128)	**package_name** - パッケージ名
STMT_TEXT	CLOB(16M)[1]	**stmt_text** - SQL 動的ステートメント・テキスト

[1] STMT_TEXT は CLOB(16M) として定義されていますが、これは単に将来の拡張に備えるために過ぎません。ステートメント・テキストの実際の出力は 64K で切り捨てられます。

* この列の元になるモニター・エレメントの合計消費時間を計算するには、このモニター・エレメントに関する _S で終わっている列で報告されている整数秒と、このモニター・エレメントに関する _MS で終わっている列に報告されている小数秒を、次の式を使用して加算しなければなりません: (*monitor-element-name*_S × 1,000,000 + *monitor-element-name*_MS) ÷ 1,000,000. 例えば、(ELAPSED_EXEC_TIME_S × 1,000,000 + ELAPSED_EXEC_TIME_MS) ÷ 1,000,000。

SNAPSHOT_SUBSECT

アプリケーション・スナップショットからアクセス・プランのサブセクションに関する情報を戻します。

注: この表関数は使用すべきではなく、 743 ページの『SNAPSUBSECTION 管理ビューおよび SNAP_GET_SUBSECTION 表関数 - subsection 論理モニター・グループ・スナップショット情報の検索』 に置き換えられました。

スキーマは SYSPROC です。

dbname
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス

内の有効なデータベース名を指定する、タイプ VARCHAR(255) の入力引数。"Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。データベース・インスタンス下のすべてのデータベースからスナップショットを取る場合は、 NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションに -1、またはすべてのアクティブなデータベース・パーティションに -2 を指定します。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

どちらのパラメーターも NULL に設定された場合は、対応するスナップショット API 要求タイプの SNAPSHOT_FILEW ストアード・プロシージャーによって以前にファイルが作成されていない場合にのみ、スナップショットがとられます。

この関数より、次に示されている表が戻されます。

表 322. SNAPSHOT_SUBSECT 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
STMT_TEXT	CLOB(16M)[1]	**stmt_text** - SQL 動的ステートメント・テキスト
SS_EXEC_TIME	BIGINT	**ss_exec_time** - サブセクション実行経過時間
TQ_TOT_SEND_SPILLS	BIGINT	**tq_tot_send_spills** - オーバーフローした表キュー・バッファーの合計数
TQ_CUR_SEND_SPILLS	BIGINT	**tq_cur_send_spills** - オーバーフローした表キュー・バッファーの現在数
TQ_MAX_SEND_SPILLS	BIGINT	**tq_max_send_spills** - 表キュー・バッファー・オーバーフローの最大数
TQ_ROWS_READ	BIGINT	**tq_rows_read** - 表キューから読み取られた行数
TQ_ROWS_WRITTEN	BIGINT	**tq_rows_written** - 表キューに書き込まれた行数
ROWS_READ	BIGINT	**rows_read** - 読み取り行数
ROWS_WRITTEN	BIGINT	**rows_written** - 書き込み行数
SS_USR_CPU_TIME	BIGINT	**ss_usr_cpu_time** - サブセクションに使用されたユーザー CPU 時間
SS_SYS_CPU_TIME	BIGINT	**ss_sys_cpu_time** - サブセクションに使用されたシステム CPU 時間
SS_NUMBER	INTEGER	**ss_number** - サブセクション番号
SS_STATUS	INTEGER	**ss_status** - サブセクションの状況

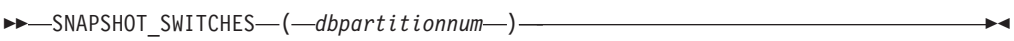
表 322. SNAPSHOT_SUBSECT 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
SS_PARTITION_NUMBER	SMALLINT	**ss_node_number** - サブセクション・ノード番号
TQ_PARTITION_WAITED_FOR	SMALLINT	**tq_node_waited_for** - 表キュー上のノード待機
TQ_WAIT_FOR_ANY	INTEGER	**tq_wait_for_any** - 表キュー上のノード送信待機
TQ_ID_WAITING_ON	INTEGER	**tq_id_waiting_on** - ノード上の表キュー待機
[1] STMT_TEXT は CLOB(16M) として定義されていますが、これは単に将来の拡張に備えるために過ぎません。ステートメント・テキストの実際の出力は 64K で切り捨てられます。		

SNAPSHOT_SWITCHES

データベース・スナップショットのスイッチ状態に関する情報を戻します。

注: この表関数は使用すべきではなく、 747 ページの『SNAPSWITCHES 管理ビューおよび SNAP_GET_SWITCHES 表関数 - データベース・スナップショットのスイッチ状態情報の検索』 に置き換えられました。



スキーマは SYSPROC です。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションに -1、またはすべてのアクティブなデータベース・パーティションに -2 を指定します。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

この関数より、次に示されている表が戻されます。

表 323. SNAPSHOT_SWITCHES 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
UOW_SW_STATE	SMALLINT	作業単位モニター記録スイッチの状態 (0 または 1)。
UOW_SW_TIME	TIMESTAMP	作業単位モニター記録スイッチがオンの場合、このスイッチがオンになった日時。

表 323. SNAPSHOT_SWITCHES 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
STATEMENT_SW_STATE	SMALLINT	SQL ステートメント・モニター記録スイッチの状態 (0 または 1)。
STATEMENT_SW_TIME	TIMESTAMP	SQL ステートメント・モニター記録スイッチがオンの場合、このスイッチがオンになった日時。
TABLE_SW_STATE	SMALLINT	表アクティビティー・モニター記録スイッチの状態 (0 または 1)。
TABLE_SW_TIME	TIMESTAMP	表アクティビティー・モニター記録スイッチがオンの場合、このスイッチがオンになった日時。
BUFFPOOL_SW_STATE	SMALLINT	バッファー・プール・アクティビティー・モニター記録スイッチの状態 (0 または 1)。
BUFFPOOL_SW_TIME	TIMESTAMP	バッファー・プール・アクティビティー・モニター記録スイッチがオンの場合、このスイッチがオンになった日時。
LOCK_SW_STATE	SMALLINT	ロック・モニター記録スイッチの状態 (0 または 1)。
LOCK_SW_TIME	TIMESTAMP	ロック・モニター記録スイッチがオンの場合、このスイッチがオンになった日時。
SORT_SW_STATE	SMALLINT	ソート・モニター記録スイッチの状態 (0 または 1)。
SORT_SW_TIME	TIMESTAMP	ソート・モニター記録スイッチがオンの場合、このスイッチがオンになった日時。
PARTITION_NUMBER	SMALLINT	**node_number** - ノード番号

SNAPSHOT_TABLE

表スナップショットからアクティビティー情報を戻します。

注: この表関数は使用すべきではなく、 751 ページの『SNAPTAB 管理ビューおよび SNAP_GET_TAB_V91 表関数 - table 論理データ・グループのスナップショット情報の検索』 に置き換えられました。

```
►►—SNAPSHOT_TABLE—(—dbname—,—dbpartitionnum—)———————————————————►◄
```

スキーマは SYSPROC です。

dbname
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(255) の入力引数。"Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションに -1、またはすべてのアクティブなデータベース・パーティションに -2 を指定します。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

どちらのパラメーターも NULL に設定された場合は、対応するスナップショット API 要求タイプの SNAPSHOT_FILEW ストアード・プロシージャーによって以前にファイルが作成されていない場合にのみ、スナップショットがとられます。

この関数より、次に示されている表が戻されます。

表 324. SNAPSHOT_TABLE 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
ROWS_WRITTEN	BIGINT	**rows_written** - 書き込み行数
ROWS_READ	BIGINT	**rows_read** - 読み取り行数
OVERFLOW_ACCESSES	BIGINT	**overflow_accesses** - オーバーフロー・レコードへのアクセス
TABLE_FILE_ID	BIGINT	**table_file_id** - 表ファイル ID
TABLE_TYPE	BIGINT	**table_type** - 表タイプ
PAGE_REORGS	BIGINT	**page_reorgs** - ページ再編成
TABLE_NAME	VARCHAR(128)	**table_name** - 表名
TABLE_SCHEMA	VARCHAR(128)	**table_schema** - 表スキーマ名

SNAPSHOT_TBREORG

注: この表関数は使用すべきではなく、 754 ページの『SNAPTAB_REORG 管理ビューおよび SNAP_GET_TAB_REORG 表関数 - 表再編成スナップショット情報の検索』 に置き換えられました。

```
►►—SNAPSHOT_TBREORG—(—dbname—,—dbpartitionnum—)————————————————►◄
```

スキーマは SYSPROC です。

SNAPSHOT_TBREORG 関数は、表の再編成に関する情報を結果セットの形式で戻します。再編成された表がない場合は、0 行が戻されます。リアルタイム・スナップショット情報を入手するには、 SYSADM、SYSCTRL、または SYSMAINT 権限がなければなりません。

dbname
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(255) の入力引数。"Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションに -1、またはすべてのアクティブなデータベース・パーティションに -2 を指定します。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

どちらのパラメーターも NULL に設定された場合は、対応するスナップショット API 要求タイプの SNAPSHOT_FILEW ストアード・プロシージャーによって以前にファイルが作成されていない場合にのみ、スナップショットがとられます。

この関数より、次に示されている表が戻されます。

表 325. SNAPSHOT_TBREORG 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
TABLE_NAME	VARCHAR(128)	**table_name** - 表名
TABLE_SCHEMA	VARCHAR(128)	**table_schema** - 表スキーマ名
PAGE_REORGS	BIGINT	**page_reorgs** - ページ再編成
REORG_PHASE	BIGINT	**reorg_phase** - 表再編成フェーズ
REORG_MAX_PHASE	INTEGER	**reorg_max_phase** - 表再編成の最大フェーズ数
REORG_CURRENT_COUNTER	BIGINT	**reorg_current_counter** - 表再編成の進行状況
REORG_MAX_COUNTER	BIGINT	**reorg_max_counter** - 表再編成の合計量
REORG_TYPE	INTEGER	**reorg_type** - 表再編成の属性
REORG_STATUS	BIGINT	**reorg_status** - 表再編成の状況
REORG_COMPLETION	INTEGER	**reorg_completion** - 表再編成完了フラグ
REORG_START	TIMESTAMP	**reorg_start** - 表再編成開始時刻
REORG_END	TIMESTAMP	**reorg_end** - 表再編成終了時刻

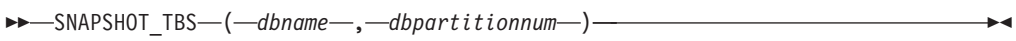
表 325. SNAPSHOT_TBREORG 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
REORG_PHASE_START	TIMESTAMP	**reorg_phase_start** - 表再編成フェーズ開始時刻
REORG_INDEX_ID	BIGINT	**reorg_index_id** - 表の再編成に使用される索引
REORG_TBSPC_ID	BIGINT	**reorg_tbspc_id** - 表が再編成される表スペース
PARTITION_NUMBER	SMALLINT	**node_number** - ノード番号

SNAPSHOT_TBS

表スペース・スナップショットからアクティビティー情報を戻します。

注: この表関数は使用すべきではなく、 760 ページの『SNAPTBSP 管理ビューおよび SNAP_GET_TBSP_V91 表関数 - 表スペース論理データ・グループのスナップショット情報の検索』 に置き換えられました。



スキーマは SYSPROC です。

dbname
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(255) の入力引数。"Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションに -1、またはすべてのアクティブなデータベース・パーティションに -2 を指定します。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

どちらのパラメーターも NULL に設定された場合は、対応するスナップショット API 要求タイプの SNAPSHOT_FILEW ストアード・プロシージャーによって以前にファイルが作成されていない場合にのみ、スナップショットがとられます。

この関数より、次に示されている表が戻されます。

表 326. SNAPSHOT_TBS 表関数によって戻される情報

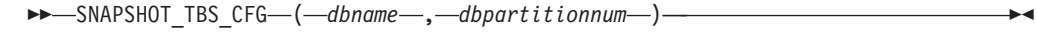
列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
POOL_DATA_L_READS	BIGINT	**pool_data_l_reads** - バッファー・プール・データの論理読み取り
POOL_DATA_P_READS	BIGINT	**pool_data_p_reads** - バッファー・プール・データの物理読み取り
POOL_ASYNC_DATA_READS	BIGINT	**pool_async_data_reads** - バッファー・プール非同期データ読み取り
POOL_DATA_WRITES	BIGINT	**pool_data_writes** - バッファー・プールへのデータの書き込み
POOL_ASYNC_DATA_WRITES	BIGINT	**pool_async_data_writes** - バッファー・プール非同期データ書き込み
POOL_INDEX_L_READS	BIGINT	**pool_index_l_reads** - バッファー・プール索引の論理読み取り
POOL_INDEX_P_READS	BIGINT	**pool_index_p_reads** - バッファー・プール索引の物理読み取り
POOL_INDEX_WRITES	BIGINT	**pool_index_writes** - バッファー・プール索引の書き込み
POOL_ASYNC_INDEX_WRITES	BIGINT	**pool_async_index_writes** - バッファー・プール非同期索引書き込み
POOL_READ_TIME	BIGINT	**pool_read_time** - バッファー・プール物理読み取り時間の合計
POOL_WRITE_TIME	BIGINT	**pool_write_time** - バッファー・プール物理書き込み時間の合計
POOL_ASYNC_READ_TIME	BIGINT	**pool_async_read_time** - バッファー・プール非同期読み取り時間
POOL_ASYNC_WRITE_TIME	BIGINT	**pool_async_write_time** - バッファー・プール非同期書き込み時間
POOL_ASYNC_DATA_ READ_REQS	BIGINT	**pool_async_data_read_reqs** - バッファー・プール非同期読み取り要求
DIRECT_READS	BIGINT	**direct_reads** - データベースからの直接読み取り
DIRECT_WRITES	BIGINT	**direct_writes** - データベースへの直接書き込み
DIRECT_READ_REQS	BIGINT	**direct_read_reqs** - 直接読み取り要求
DIRECT_WRITE_REQS	BIGINT	**direct_write_reqs** - 直接書き込み要求
DIRECT_READ_TIME	BIGINT	**direct_read_time** - 直接読み取り時間
DIRECT_WRITE_TIME	BIGINT	**direct_write_time** - 直接書き込み時間

表 326. SNAPSHOT_TBS 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
UNREAD_PREFETCH_PAGES	BIGINT	**unread_prefetch_pages** - 読み取り不能プリフェッチ・ページ
POOL_ASYNC_INDEX_READS	BIGINT	**pool_async_index_reads** - バッファー・プール非同期索引読み取り
POOL_DATA_TO_ESTORE	BIGINT	**pool_data_to_estore** ESTORE モニター・エレメントは廃止されています。廃止されたモニター・エレメントには NULL 値が戻されます。
POOL_INDEX_TO_ESTORE	BIGINT	**pool_index_to_estore** ESTORE モニター・エレメントは廃止されています。廃止されたモニター・エレメントには NULL 値が戻されます。
POOL_INDEX_FROM_ESTORE	BIGINT	**pool_index_from_estore** ESTORE モニター・エレメントは廃止されています。廃止されたモニター・エレメントには NULL 値が戻されます。
POOL_DATA_FROM_ESTORE	BIGINT	**pool_data_from_estore** ESTORE モニター・エレメントは廃止されています。廃止されたモニター・エレメントには NULL 値が戻されます。
FILES_CLOSED	BIGINT	**files_closed** - 閉じられたデータベース・ファイル
TABLESPACE_NAME	VARCHAR(128)	**tablespace_name** - 表スペース名

SNAPSHOT_TBS_CFG

注: この表関数は使用すべきではなく、 919 ページの『SNAPTBSP_PART 管理ビューおよび SNAP_GET_TBSP_PART_V91 表関数 - tablespace_nodeinfo 論理データ・グループのスナップショット情報の検索』 に置き換えられました。



スキーマは SYSPROC です。

SNAPSHOT_TBS_CFG 関数は、表スペース・スナップショットから構成情報を戻します。

dbname
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なデータベース名を指定する、タイプ VARCHAR(255) の入力引数。"Indirect" または "Home" のディレクトリー項目タイプを持つデータベース名を指定します。このディレクトリー項目タイプは、LIST DATABASE

DIRECTORY コマンドで確認できます。現在接続されているデータベースからのスナップショットを取得するには、NULL 値を指定します。

dbpartitionnum
有効なデータベース・パーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションに -1、またはすべてのアクティブなデータベース・パーティションに -2 を指定します。アクティブなデータベース・パーティションとは、アプリケーションからデータベースに接続して、これを使用することが可能なパーティションのことです。

NULL 値を指定すると、-1 が暗黙的に設定されます。

どちらのパラメーターも NULL に設定された場合は、対応するスナップショット API 要求タイプの SNAPSHOT_FILEW ストアード・プロシージャーによって以前にファイルが作成されていない場合にのみ、スナップショットがとられます。

この関数より、次に示されている表が戻されます。

表 327. SNAPSHOT_TBS_CFG 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
TABLESPACE_ID	BIGINT	**tablespace_id** - 表スペース ID
TABLESPACE_NAME	VARCHAR (128)	**tablespace_name** - 表スペース名
TABLESPACE_TYPE	SMALLINT	**tablespace_type** - 表スペース・タイプ
TABLESPACE_STATE	BIGINT	**tablespace_state** - 表スペースの状態
NUM_QUIESCERS	BIGINT	**tablespace_num_quiescers** - 静止プログラム数
STATE_CHANGE_OBJ_ID	BIGINT	**tablespace_state_change_object_id** - 状態変更オブジェクト ID
STATE_CHANGE_TBS_ID	BIGINT	**tablespace_state_change_ts_id** - 状態変更表スペース ID
MIN_RECOVERY_TIME	TIMESTAMP	**tablespace_min_recovery_time** - ロールフォワードの最小リカバリー時間
TBS_CONTENTS_TYPE	SMALLINT	**tablespace_content_type** - 表スペースのコンテンツ・タイプ
BUFFERPOOL_ID	BIGINT	**tablespace_cur_pool_id** - 現在使用中のバッファー・プール
NEXT_BUFFERPOOL_ID	BIGINT	**tablespace_next_pool_id** - 次の始動時に使用されるバッファー・プール
PAGE_SIZE	BIGINT	**tablespace_page_size** - 表スペースのページ・サイズ
EXTENT_SIZE	BIGINT	**tablespace_extent_size** - 表スペースのエクステント・サイズ
PREFETCH_SIZE	BIGINT	**tablespace_prefetch_size** - 表スペースのプリフェッチ・サイズ
TOTAL_PAGES	BIGINT	**tablespace_total_pages** - 表スペース内の合計ページ数

表 327. SNAPSHOT_TBS_CFG 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
USABLE_PAGES	BIGINT	**tablespace_usable_pages** - 表スペース内の使用可能ページ数
USED_PAGES	BIGINT	**tablespace_used_pages** - 表スペース内の使用されているページ数
FREE_PAGES	BIGINT	**tablespace_free_pages** - 表スペース内のフリー・ページ数
PENDING_FREE_PAGES	BIGINT	**tablespace_pending_free_pages** - 表スペース内のペンディング・フリー・ページ数
HIGH_WATER_MARK	BIGINT	**pool_watermark** - メモリー・プール水準点
REBALANCER_MODE	BIGINT	**tablespace_rebalancer_mode** - リバランサー・モード
REBALANCER_EXTENTS_REMAINING	BIGINT	**tablespace_rebalancer_extents_remaining** - リバランサーで処理されるエクステントの合計数
REBALANCER_EXTENTS_PROCESSED	BIGINT	**tablespace_rebalancer_extents_processed** - リバランサーで処理されたエクステントの数
REBALANCER_PRIORITY	BIGINT	**tablespace_rebalancer_priority** - 現行のリバランサー優先順位
REBALANCER_START_TIME	TIMESTAMP	**tablespace_rebalancer_start_time** - リバランサー開始時刻
REBALANCER_RESTART_TIME	TIMESTAMP	**tablespace_rebalancer_restart_time** - リバランサー再始動時刻
LAST_EXTENT_MOVED	BIGINT	**tablespace_rebalancer_last_extent_moved** - リバランサーによって最後に移動されたエクステント
NUM_RANGES	BIGINT	**tablespace_num_ranges** - 表スペース・マップ内の範囲数
NUM_CONTAINERS	BIGINT	**tablespace_num_containers** - 表スペース内のコンテナー数

SQLCACHE_SNAPSHOT

注: この表関数は使用すべきではなく、 1243 ページの『SNAP_GET_DYN_SQL_V91 表関数 - dynsql 論理グループのスナップショット情報の検索』 に置き換えられました。

スキーマは SYSFUN です。

SQLCACHE_SNAPSHOT 関数は、 DB2 動的 SQL ステートメント・キャッシュのスナップショットの結果を戻します。

この関数は引数を取りません。この関数は、次に示されている表を戻します。

表 328. SQLCACHE_SNAPSHOT 表関数によって戻される情報

列名	データ・タイプ	説明または対応するモニター・エレメント
NUM_EXECUTIONS	INTEGER	num_executions - ステートメント実行回数
NUM_COMPILATIONS	INTEGER	num_compilations - ステートメント・コンパイル数
PREP_TIME_WORST	INTEGER	prep_time_worst - ステートメント最長準備時間
PREP_TIME_BEST	INTEGER	prep_time_best - ステートメント最短準備時間
INT_ROWS_DELETED	INTEGER	int_rows_deleted - 削除された内部行数
INT_ROWS_INSERTED	INTEGER	int_rows_inserted - 挿入された内部行数
ROWS_READ	INTEGER	rows_read - 読み取り行数
INT_ROWS_UPDATED	INTEGER	int_rows_updated - 更新された内部行数
ROWS_WRITE	INTEGER	rows_written - 書き込み行数
STMT_SORTS	INTEGER	stmt_sorts - ステートメント・ソート回数
TOTAL_EXEC_TIME_S	INTEGER	total_exec_time - ステートメント実行の経過時間 (秒単位)*
TOTAL_EXEC_TIME_MS	INTEGER	total_exec_time - ステートメント実行の経過時間 (小数部、マイクロ秒単位)*
TOT_U_CPU_TIME_S	INTEGER	total_usr_cpu_time - ステートメントのユーザー CPU の合計 (秒単位)*
TOT_U_CPU_TIME_MS	INTEGER	total_usr_cpu_time - ステートメントのユーザー CPU の合計 (小数部、マイクロ秒単位)*
TOT_S_CPU_TIME_S	INTEGER	total_sys_cpu_time - ステートメントのシステム CPU の合計 (秒単位)*
TOT_S_CPU_TIME_MS	INTEGER	total_sys_cpu_time - ステートメントのシステム CPU の合計 (小数部、マイクロ秒単位)*
DB_NAME	VARCHAR(128)	db_name - データベース名

表 328. SQLCACHE_SNAPSHOT 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明または対応するモニター・エレメント
STMT_TEXT	CLOB(16M)[1]	stmt_text - SQL 動的ステートメント・テキスト

[1] STMT_TEXT は CLOB(16M) として定義されていますが、これは単に将来の拡張に備えるために過ぎません。ステートメント・テキストの実際の出力は 64K で切り捨てられます。

* この列の元になるモニター・エレメントの合計消費時間を計算するには、このモニター・エレメントに関する _S で終わっている列で報告されている整数秒と、このモニター・エレメントに関する _MS で終わっている列に報告されている小数秒を、次の式を使用して加算しなければなりません: (*monitor-element-name*_S × 1,000,000 + *monitor-element-name*_MS) ÷ 1,000,000. 例えば、(ELAPSED_EXEC_TIME_S × 1,000,000 + ELAPSED_EXEC_TIME_MS) ÷ 1,000,000。

SYSINSTALLROUTINES

注: このプロシージャーは使用すべきではありません。このプロシージャーは、DB2 UDB for Linux, UNIX, and Windows バージョン 8 において新規のプロシージャーおよび関数の作成のために使用されていました。

```
►►—SYSINSTALLROUTINES—(—)————————————————————————————►◄
```

スキーマは SYSPROC です。

WLM_GET_ACTIVITY_DETAILS - 特定のアクティビティーに関する詳細情報を戻す

注: この表関数は使用すべきではなく、 に置き換えられました。 MON_GET_ACTIVITY_DETAILS 表関数.

この関数は、1 つ以上のサービス・サブクラスの基本統計を戻します。

この関数は、そのアプリケーション・ハンドル、作業単位 ID、およびアクティビティー ID によって識別される特定のアクティビティーに関する詳細情報を戻します。この情報には、アクティビティーが違反したしきい値に関する詳細が含まれます。

構文

```
►►—WLM_GET_ACTIVITY_DETAILS—(—application_handle—,—uow_id—,————————————►
►—activity_id—,—dbpartitionnum—)————————————————————————————————►◄
```

スキーマは SYSPROC です。

表関数パラメーター

application_handle
有効なアプリケーション・ハンドルを指定する、タイプ BIGINT の入力引数。引数が NULL の場合、行はこの関数から戻されません。引数が NULL の場合、SQL171N エラーが戻されます。

uow_id
アプリケーション内で固有の有効な作業単位 ID を指定する、タイプ INTEGER の入力引数。引数が NULL の場合、行はこの関数から戻されません。引数が NULL の場合、SQL171N エラーが戻されます。

activity_id
作業単位内で固有の有効なアクティビティー ID を指定する、タイプ INTEGER の入力引数。引数が NULL の場合、行はこの関数から戻されません。引数が NULL の場合、SQL171N エラーが戻されます。

dbpartitionnum
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なパーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションには -1、すべてのデータベース・パーティションには -2 を指定します。 NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

WLM_GET_ACTIVITY_DETAILS 関数に対する EXECUTE 特権。

例

個々のアクティビティーに関する詳細情報は、WLM_GET_ACTIVITY_DETAILS 表関数を使用して取得できます。この表関数は、パーティションごとにアクティビティー情報を、名前と値のペアで戻します。この例は、アプリケーション・ハンドル 1、作業単位 ID 1、アクティビティー ID 5 で識別されるアクティビティーのパーティションごとに、名前と値のペアの 11 個のメンバー・サブセットのみを示すことに限定しています。名前と値のペアの完全なリストについては、1318 ページの表 330および 1322 ページの表 331を参照してください。

```
SELECT SUBSTR(CHAR(DBPARTITIONNUM),1,4) AS PART,
       SUBSTR(NAME, 1, 20) AS NAME,
       SUBSTR(VALUE, 1, 30) AS VALUE
FROM TABLE(WLM_GET_ACTIVITY_DETAILS(1, 1, 5, -2)) AS ACTDETAIL
WHERE NAME IN ('APPLICATION_HANDLE',
               'COORD_PARTITION_NUM',
               'LOCAL_START_TIME',
               'UOW_ID',
               'ACTIVITY_ID',
               'PARENT_UOW_ID',
               'PARENT_ACTIVITY_ID',
               'ACTIVITY_TYPE',
               'NESTING_LEVEL',
               'INVOCATION_ID',
               'ROUTINE_ID')
ORDER BY PART
```

以下はこの照会の出力例です。

```
PART NAME                 VALUE
---- -------------------- ------------------------------
0    APPLICATION_HANDLE   1
0    COORD_PARTITION_NUM  0
0    LOCAL_START_TIME     2005-11-25-18.52.49.343000
0    UOW_ID               1
0    ACTIVITY_ID          5
0    PARENT_UOW_ID        1
0    PARENT_ACTIVITY_ID   3
0    ACTIVITY_TYPE        READ_DML
0    NESTING_LEVEL        0
0    INVOCATION_ID        1
0    ROUTINE_ID           0
1    APPLICATION_HANDLE   1
1    COORD_PARTITION_NUM  0
1    LOCAL_START_TIME     2005-11-25-18.52.49.598000
1    UOW_ID               1
1    ACTIVITY_ID          5
1    PARENT_UOW_ID
1    PARENT_ACTIVITY_ID
1    ACTIVITY_TYPE        READ_DML
1    NESTING_LEVEL        0
1    INVOCATION_ID        1
1    ROUTINE_ID           0
```

使用上の注意

ACTIVITY_STATE が QUEUED である場合、コーディネーター・アクティビティーがカタログ・パーティションに対する RPC を行ってしきい値チケットを取得したが、まだ応答を受け取っていないことを意味します。この状態が表示されることは、アクティビティーが WLM によってキューに入れられていることを示すか、または短期間にわたって、アクティビティーがそのチケットを取得する処理中であることを示すことがあります。アクティビティーが本当にキューに入れられているかどうかについてもっと正確な実態を把握するために、どのエージェントが (WLM_GET_SERVICE_CLASS_AGENTS 表関数を使用して) アクティビティーで作業しているかを判別し、カタログ・パーティションにあるこのエージェントの event_object の値が WLM_QUEUE であるかどうかを検出することができます。

戻される情報

表 329. WLM_GET_ACTIVITY_DETAILS について戻される情報

列名	データ・タイプ	説明
DBPARTITIONNUM	SMALLINT	このレコードが収集されたパーティション番号。
NAME	VARCHAR(256)	エレメント名。考えられる値については、表 330および 1322 ページの表 331を参照してください。
VALUE	VARCHAR(1024)	エレメントの値。考えられる値については、表 330および 1322 ページの表 331を参照してください。

表 330. 戻されるエレメント

エレメント名	説明
ACTIVITY_ID	アプリケーション内の固有のアクティビティー ID。

表 330. 戻されるエレメント (続き)

エレメント名	説明
ACTIVITY_STATE	使用できる値は以下のとおりです。 • CANCEL_PENDING • EXECUTING • IDLE • INITIALIZING • QP_CANCEL_PENDING • QP_QUEUED • QUEUED • TERMINATING • UNKNOWN
ACTIVITY_TYPE	使用できる値は以下のとおりです。 • CALL • DDL • LOAD • OTHER • READ_DML • WRITE_DML
APPLICATION_HANDLE	システム全体におけるアプリケーションのユニーク ID。単一パーティション・データベースの場合、この ID は 16 ビットのカウンターで構成されます。複数パーティション・データベースの場合、この ID はコーディネーター・パーティション番号と 16 ビットのカウンターを連結したもので構成されます。さらに、この ID はアプリケーションが 2 次接続を行うすべてのパーティションにおいて同じです。
COORD_PARTITION_NUM	アクティビティーのコーディネーター・パーティション。
DATABASE_WORK_ACTION_SET_ID	このアクティビティーがデータベースに適用されている作業アクション・セットにマップされている場合、この列にはその作業アクション・セットの ID が入っています。アクティビティーがデータベースに適用されている作業アクション・セットにマップされていない場合、この列には 0 が入っています。
DATABASE_WORK_CLASS_ID	このアクティビティーがデータベースに適用されている作業アクション・セットにマップされている場合、この列にはこのアクティビティーの作業クラスの ID が入っています。アクティビティーがデータベースに適用されている作業アクション・セットにマップされていない場合、この列には 0 が入っています。
EFFECTIVE_ISOLATION	このアクティビティーに有効な分離レベル。
EFFECTIVE_LOCK_TIMEOUT	このアクティビティーに有効なロック・タイムアウト値。
EFFECTIVE_QUERY_DEGREE	このアクティビティーに有効な照会度の値。
ENTRY_TIME	このアクティビティーがシステムに到達した時刻。

表 330. 戻されるエレメント (続き)

エレメント名	説明
INVOCATION_ID	1 つのルーチン呼び出しを、作業単位内の同じネスト・レベルにある他の呼び出しと区別するための ID。これは作業単位内の特定のネスト・レベルにおいて固有です。
LAST_REFERENCE_TIME	要求がこのアクティビティーで発生するたびに、このフィールドは更新されます。
LOCAL_START_TIME	アクティビティーがパーティションで作業を開始した時刻。これはローカル時刻です。アクティビティーがシステムに入ったが、キューに入れられており、実行を開始していない場合は、このフィールドを空ストリングにすることができます。
NESTING_LEVEL	これはこのアクティビティーのネスト・レベルを表します。ネスト・レベルは、このアクティビティーが一番上の親アクティビティー内でネストされる深さです。
PACKAGE_NAME	アクティビティーが SQL ステートメントの場合、これはそのパッケージの名前を表します。
PACKAGE_SCHEMA	アクティビティーが SQL ステートメントの場合、これはそのパッケージのスキーマ名を表します。
PACKAGE_VERSION_ID	アクティビティーが SQL ステートメントの場合、これはそのパッケージのバージョンを表します。
PARENT_ACTIVITY_ID	親のアクティビティー ID が ACTIVITY_ID である、作業単位内の固有のアクティビティー。アクティビティーに親アクティビティーがない場合、空ストリングを戻します。
PARENT_UOW_ID	アプリケーション内の固有の作業単位 ID。このアクティビティーの親アクティビティーが開始された元の作業単位を表します。アクティビティーに親アクティビティーがない場合、またはそれがリモート・パーティションにある場合は、空ストリングを戻します。
QP_QUERY_ID	アクティビティーが照会の場合に、Query Patroller によってこのアクティビティーに割り当てられる照会 ID。照会 ID が 0 であると、Query Patroller が照会 ID をこのアクティビティーに割り当てなかったことを示します。
QUERY_COST_ESTIMATE	SQL コンパイラーによって決定された、timeron 時の照会の見積コスト。
ROUTINE_ID	ルーチン固有 ID。アクティビティーがルーチンの一部でない場合にはゼロを返します。
ROWS_FETCHED	これは表から読み取られた行数です。これは、このレコードが記録されているデータベース・パーティションのこれらの値のみ報告します。パーティション・データベース環境では、これらの値は、アクティビティー全体の正しい合計を反映しない場合があります。ステートメント・モニター・スイッチがオンにされていないと、このエレメントは収集されず、代わりに -1 が書き込まれます。

表 330. 戻されるエレメント (続き)

エレメント名	説明
ROWS_MODIFIED	これは挿入、更新、または削除された行数です。これは、このレコードが記録されているデータベース・パーティションのこれらの値のみ報告します。パーティション・データベース環境では、これらの値は、アクティビティー全体の正しい合計を反映しない場合があります。ステートメント・モニター・スイッチがオンにされていないと、このエレメントは収集されず、代わりに -1 が書き込まれます。
SECTION_NUMBER	アクティビティーが SQL ステートメントの場合、これはそのセクション番号を表します。
SERVICE_CLASS_ID	このアクティビティーが属するサービス・クラスのユニーク ID。
SERVICE_CLASS_WORK_ACTION_SET_ID	このアクティビティーがサービス・クラスに適用されている作業アクション・セットにマップされている場合、この列にはその作業アクション・セットの ID が入っています。アクティビティーがサービス・クラスに適用されている作業アクション・セットにマップされていない場合、この列には 0 が入っています。
SERVICE_CLASS_WORK_CLASS_ID	このアクティビティーがサービス・クラスに適用されている作業アクション・セットにマップされている場合、この列にはこのアクティビティーの作業クラスの ID が入っています。アクティビティーがサービス・クラスに適用されている作業アクション・セットにマップされていない場合、この列には 0 が入っています。
STMT_PKG_CACHE_ID	ステートメント・パッケージ・キャッシュ ID。
STMT_TEXT	アクティビティーが動的 SQL であるか、またはステートメント・テキストが使用可能になっている静的 SQL である場合、このフィールドにはそのステートメント・テキストの最初の 1024 文字が入っています。そうでない場合、これは空ストリングです。
SYSTEM_CPU_TIME	データベース・マネージャー・エージェント・プロセス、作業単位、またはステートメントによって使用されるシステム CPU 時間の合計 (秒およびマイクロ秒)。ステートメント・モニター・スイッチまたはタイム・スタンプ・スイッチがオンにされていないと、このエレメントは収集されず、代わりに -1 が書き込まれます。
UOW_ID	アプリケーション内の固有の作業単位 ID。このアクティビティーが開始した元の作業単位を表します。
USER_CPU_TIME	データベース・マネージャー・エージェント・プロセス、作業単位、またはステートメントによって使用されるユーザー CPU 時間の合計 (秒およびマイクロ秒)。ステートメント・モニター・スイッチまたはタイム・スタンプ・スイッチがオンにされていないと、このエレメントは収集されず、代わりに -1 が書き込まれます。
UTILITY_ID	アクティビティーがユーティリティーの場合、これはそのユーティリティー ID です。それ以外の場合、このフィールドは 0 です。

重要: WLM_GET_ACTIVITY_DETAILS 表関数は、現在アクティビティーに適用されているしきい値のみを示します。

以下のエレメントは、対応するしきい値がアクティビティーに適用される場合にのみ戻されます。

表 331. 適用される場合に戻されるエレメント

エレメント名	説明
ACTIVITYTOTALTIME_THRESHOLD_ID	アクティビティーに適用された ACTIVITYTOTALTIME しきい値の ID。
ACTIVITYTOTALTIME_THRESHOLD_VALUE	ACTIVITYTOTALTIME しきい値期間をアクティビティー開始時刻に加算して算出されるタイム・スタンプ。このタイム・スタンプに達した時にアクティビティーがまだ実行されている場合、しきい値に違反します。
ACTIVITYTOTALTIME_THRESHOLD_VIOLATED	「Yes」は、アクティビティーが ACTIVITYTOTALTIME しきい値に違反したことを示します。「No」は、そのアクティビティーがまだしきい値に違反していないことを示します。
CONCURRENTDBCOORDACTIVITIES_DB _THRESHOLD_ID	アクティビティーに適用された CONCURRENTDBCOORDACTIVITIES_DB しきい値の ID。
CONCURRENTDBCOORDACTIVITIES_DB _THRESHOLD_QUEUED	「Yes」は、アクティビティーが CONCURRENTDBCOORDACTIVITIES_DB しきい値によってキューに入れられたことを示します。「No」は、アクティビティーがキューに入れられなかったことを示します。
CONCURRENTDBCOORDACTIVITIES_DB _THRESHOLD_VALUE	アクティビティーに適用された CONCURRENTDBCOORDACTIVITIES_DB しきい値の上限。
CONCURRENTDBCOORDACTIVITIES_DB _THRESHOLD_VIOLATED	「Yes」は、アクティビティーが CONCURRENTDBCOORDACTIVITIES_DB しきい値に違反したことを示します。「No」は、そのアクティビティーがまだしきい値に違反していないことを示します。
CONCURRENTDBCOORDACTIVITIES_SUBCLASS _THRESHOLD_ID	アクティビティーに適用された CONCURRENTDBCOORDACTIVITIES_SUBCLASS しきい値の ID。
CONCURRENTDBCOORDACTIVITIES_SUBCLASS _THRESHOLD_QUEUED	「Yes」は、アクティビティーが CONCURRENTDBCOORDACTIVITIES_SUBCLASS しきい値によってキューに入れられたことを示します。「No」は、アクティビティーがキューに入れられなかったことを示します。
CONCURRENTDBCOORDACTIVITIES_SUBCLASS _THRESHOLD_VALUE	アクティビティーに適用された CONCURRENTDBCOORDACTIVITIES_SUBCLASS しきい値の上限。
CONCURRENTDBCOORDACTIVITIES_SUBCLASS _THRESHOLD_VIOLATED	「Yes」は、アクティビティーが CONCURRENTDBCOORDACTIVITIES_SUBCLASS しきい値に違反したことを示します。「No」は、そのアクティビティーがまだしきい値に違反していないことを示します。
CONCURRENTDBCOORDACTIVITIES_SUPERCLASS _THRESHOLD_ID	アクティビティーに適用された CONCURRENTDBCOORDACTIVITIES_SUPERCLASS しきい値の ID。

表 331. 適用される場合に戻されるエレメント (続き)

エレメント名	説明
CONCURRENTDBCOORDACTIVITIES_SUPERCLASS _THRESHOLD_QUEUED	「Yes」は、アクティビティーが CONCURRENTDBCOORDACTIVITIES_SUPERCLASS しきい値によってキューに入れられた ことを示します。 「No」は、アクティビティーがキューに 入れられなかったことを示します。
CONCURRENTDBCOORDACTIVITIES_SUPERCLASS _THRESHOLD_VALUE	アクティビティーに適用された CONCURRENTDBCOORDACTIVITIES_SUPERCLASS しきい値の上限。
CONCURRENTDBCOORDACTIVITIES_SUPERCLASS _THRESHOLD_VIOLATED	「Yes」は、アクティビティーが CONCURRENTDBCOORDACTIVITIES_SUPERCLASS しきい値に違反したことを示します。 「No」は、そのアクティビティーがまだ しきい値に違反していないことを示します。
CONCURRENTDBCOORDACTIVITIES_WORK_ACTION_SET _THRESHOLD_ID	アクティビティーに適用された CONCURRENTDBCOORDACTIVITIES_WORK _ACTION_SET しきい値の ID。
CONCURRENTDBCOORDACTIVITIES_WORK_ACTION_SET _THRESHOLD_QUEUED	「Yes」は、アクティビティーが CONCURRENTDBCOORDACTIVITIES_WORK _ACTION_SET しきい値によって キューに入れられたことを示します。 「No」は、アクティビティーがキューに 入れられなかったことを示します。
CONCURRENTDBCOORDACTIVITIES_WORK_ACTION_SET _THRESHOLD_VALUE	アクティビティーに適用された CONCURRENTDBCOORDACTIVITIES_WORK _ACTION_SET しきい値の上限。
CONCURRENTDBCOORDACTIVITIES_WORK_ACTION_SET _THRESHOLD_VIOLATED	「Yes」は、アクティビティーが CONCURRENTDBCOORDACTIVITIES_WORK _ACTION_SET しきい値に違反した ことを示します。 「No」は、そのアクティビティーが まだしきい値に違反していない ことを示します。
CONCURRENTWORKLOADACTIVITIES _THRESHOLD_ID	アクティビティーに適用された CONCURRENTWORKLOADACTIVITIES しきい値の ID。
CONCURRENTWORKLOADACTIVITIES _THRESHOLD_VALUE	アクティビティーに適用された CONCURRENTWORKLOADACTIVITIES しきい値の上限。
CONCURRENTWORKLOADACTIVITIES _THRESHOLD_VIOLATED	「Yes」は、アクティビティーが CONCURRENTWORKLOADACTIVITIES しきい値に違反したことを示します。「No」は、そのアクティビティーがまだしきい値に違反していないことを示します。
ESTIMATEDSQLCOST_THRESHOLD_ID	アクティビティーに適用された ESTIMATEDSQLCOST しきい値の ID。
ESTIMATEDSQLCOST_THRESHOLD_VALUE	アクティビティーに適用された ESTIMATEDSQLCOST しきい値の上限。
ESTIMATEDSQLCOST_THRESHOLD_VIOLATED	「1」は、アクティビティーが ESTIMATEDSQLCOST しきい値に違反したことを示します。「0」は、そのアクティビティーがまだしきい値に違反していないことを示します。
SQLROWSRETURNED_THRESHOLD_ID	アクティビティーに適用された SQLROWSRETURNED しきい値の ID。
SQLROWSRETURNED_THRESHOLD_VALUE	アクティビティーに適用された SQLROWSRETURNED しきい値の上限。

表 331. 適用される場合に戻されるエレメント (続き)

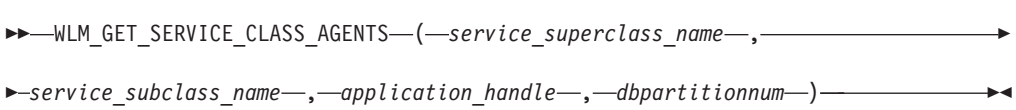
エレメント名	説明
SQLROWSRETURNED_THRESHOLD_VIOLATED	「Yes」は、アクティビティーが SQLROWSRETURNED しきい値に違反したことを示します。「No」は、そのアクティビティーがまだしきい値に違反していないことを示します。
SQLTEMPSPACE_THRESHOLD_ID	アクティビティーに適用された SQLTEMPSPACE しきい値の ID。
SQLTEMPSPACE_THRESHOLD_VALUE	アクティビティーに適用された SQLTEMPSPACE しきい値の上限。
SQLTEMPSPACE_THRESHOLD_VIOLATED	「Yes」は、アクティビティーが SQLTEMPSPACE しきい値に違反したことを示します。「No」は、そのアクティビティーがまだしきい値に違反していないことを示します。

WLM_GET_SERVICE_CLASS_AGENTS - サービス・クラスで実行中のエージェントのリスト

WLM_GET_SERVICE_CLASS_AGENTS 関数は、指定されたサービス・クラスで実行しているか、または指定されたアプリケーションの代わりに実行している、指定されたパーティション上のエージェント、fenced モード・プロセス (db2fmp プロセス)、およびシステム・エンティティーのリストを戻します。システム・エンティティーは、非エージェント・スレッドおよびプロセス (ページ・クリーナーおよびプリフェッチャーなど) です。

注: この表関数は使用すべきではなく、 1009 ページの『WLM_GET_SERVICE_CLASS_AGENTS_V97 表関数 - サービス・クラスで実行中のエージェントのリスト』 に置き換えられました。

構文



スキーマは SYSPROC です。

表関数パラメーター

service_superclass_name
現在接続されているデータベースのサービス・スーパークラスの名前を指定する、タイプ VARCHAR(128) の入力引数。引数が NULL または空ストリングである場合、他のパラメーターの値と一致する、データベース内のすべてのスーパークラスについてデータが取得されます。

service_subclass_name
スーパークラス内の特定のサブクラスを参照する、タイプ VARCHAR(128) の入力引数。引数が NULL または空ストリングである場合、他のパラメーターの値と一致する、データベース内のすべてのサブクラスについてデータが取得されます。

application_handle
エージェント情報が戻されるアプリケーション・ハンドルを指定する、タイプ BIGINT の入力引数。引数が NULL である場合、他のパラメーターの値と一致する、データベース内のすべてのアプリケーションについてデータが取得されます。 0 を指定すると、システム・エンティティーだけが返されます。

dbpartitionnum
現在接続されているデータベースと同じインスタンス内の有効なパーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションには -1、すべてのデータベース・パーティションには -2 を指定します。 NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

WLM_GET_SERVICE_CLASS_AGENTS 関数に対する EXECUTE 特権。

例

以下の照会は、すべてのデータベース・パーティションについてアプリケーション・ハンドル 1 に関連付けられたエージェントのリストを戻します。アプリケーション・ハンドルは、LIST APPLICATIONS コマンドまたは WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES 表関数を使用して判別される可能性もありました。

```
SELECT SUBSTR(CHAR(APPLICATION_HANDLE),1,7) AS APPHANDLE,
     SUBSTR(CHAR(DBPARTITIONNUM),1,4) AS PART,
     SUBSTR(CHAR(AGENT_TID),1,9) AS AGENT_TID,
     SUBSTR(AGENT_TYPE,1,11) AS AGENTTYPE,
     SUBSTR(AGENT_STATE,1,10) AS AGENTSTATE,
     SUBSTR(REQUEST_TYPE,1,12) AS REQTYPE,
     SUBSTR(CHAR(UOW_ID),1,6) AS UOW_ID,
     SUBSTR(CHAR(ACTIVITY_ID),1,6) AS ACT_ID
FROM TABLE(WLM_GET_SERVICE_CLASS_AGENTS(CAST(NULL AS VARCHAR(128)),
     CAST(NULL AS VARCHAR(128)), 1, -2)) AS SCDETAILS
ORDER BY APPHANDLE, PART, AGENT_TID
```

出力例を以下に示します。

```
APPHANDLE PART AGENT_TID AGENTTYPE   AGENTSTATE REQTYPE        UOW_ID ACT_ID
--------- ---- --------- ----------- ---------- -------------- ------ ------
1         0    3         COORDINATOR ACTIVE     FETCH          1      5
1         0    4         SUBAGENT    ACTIVE     SUBSECTION:1   1      5
1         1    2         SUBAGENT    ACTIVE     SUBSECTION:2   1      5
```

出力は、UOW ID 1 およびアクティビティー ID 5 のアクティビティーの代わりに作動しているパーティション 0 上のコーディネーター・エージェントとサブエージェント、およびパーティション 1 上のサブエージェントを示しています。コーディネーター・エージェントは要求がフェッチ要求であることを報告しています。

使用上の注意

これらのパラメーターの作用については、論理積 (AND) が取られます。つまり、矛盾する入力パラメーターを指定する (例えば、サービス・スーパークラス SUP_A とサブクラス SUB_B を、SUB_B が SUP_A のサブクラスにならないように指定する) 場合、行は戻されません。

戻される情報

表 332. WLM_GET_SERVICE_CLASS_AGENTS によって戻される情報

列名	データ・タイプ	説明
SERVICE_SUPERCLASS_NAME	VARCHAR(128)	このレコードが収集されたサービス・スーパークラスの名前。
SERVICE_SUBCLASS_NAME	VARCHAR(128)	このレコードが収集されたサービス・サブクラスの名前。
APPLICATION_HANDLE	BIGINT	システム全体におけるアプリケーションのユニーク ID。単一パーティション・データベースの場合、この ID は 16 ビットのカウンターで構成されます。複数パーティション・データベースの場合、この ID はコーディネーター・パーティション番号と 16 ビットのカウンターを連結したもので構成されます。さらに、この ID はアプリケーションが 2 次接続を行うすべてのパーティションにおいて同じです。
DBPARTITIONNUM	SMALLINT	このレコードが収集されたパーティション番号。
ENTITY	VARCHAR(32)	以下の値のいずれか。 • db2agent。この値は、エンティティーのタイプがエージェントであることを示します。 • db2fmp (*pid*)。これは、エンティティーが fenced モード・プロセスであることを示します。ここで *pid* は、fenced モード・プロセスのプロセス ID です。 • システム・エンティティーの名前。
WORKLOAD_NAME	VARCHAR(128)	このレコードが収集されたワークロードの名前。
WORKLOAD_OCCURRENCE_ID	INTEGER	ワークロード・オカレンスの ID。ワークロード・オカレンスがコーディネーター・データベース・パーティション番号およびワークロード名と結合されていなければ、この ID はワークロード・オカレンスを一意的に識別しません。
UOW_ID	INTEGER	このアクティビティーが開始された作業単位のユニーク ID。
ACTIVITY_ID	INTEGER	作業単位内の固有のアクティビティー ID。
PARENT_UOW_ID	INTEGER	アクティビティーの親アクティビティーが開始された作業単位のユニーク ID。アクティビティーに親アクティビティーがない場合、列の値は NULL です。
PARENT_ACTIVITY_ID	INTEGER	親のアクティビティー ID が activity_id である、作業単位内のアクティビティーのユニーク ID。アクティビティーに親アクティビティーがない場合、列の値は NULL です。
AGENT_TID	BIGINT	エージェントまたはシステム・エンティティーのスレッド ID。この ID が使用できない場合、列の値は NULL です。
AGENT_TYPE	VARCHAR(32)	エージェント・タイプ。可能な値は、以下のとおりです。 • COORDINATOR • OTHER • PDBSUBAGENT • SMPSUBAGENT エージェント・タイプが COORDINATOR の場合、エージェント ID はコンセントレーター環境で変わることがあります。

表 332. *WLM_GET_SERVICE_CLASS_AGENTS* によって戻される情報 (続き)

列名	データ・タイプ	説明
SMP_COORDINATOR	INTEGER	エージェントが SMP コーディネーターかどうかを示す標識。エージェントが SMP コーディネーターである場合、値は 1 です。そうでない場合、値は 0 です。
AGENT_SUBTYPE	VARCHAR(32)	エージェント・サブタイプ。可能な値は、以下のとおりです。 • DSS • OTHER • RPC • SMP
AGENT_STATE	VARCHAR(32)	エージェント状態。可能な値は、以下のとおりです。 • ACTIVE • ASSOCIATED
EVENT_TYPE	VARCHAR(32)	このエージェントによって最後に処理されたイベントのタイプ。可能な値は、以下のとおりです。 • ACQUIRE • PROCESS • WAIT この列に使用できる値についての詳細は、 1330 ページの表 333 を参照してください。
EVENT_OBJECT	VARCHAR(32)	このエージェントによって最後に処理されたイベントのオブジェクト。可能な値は、以下のとおりです。 • COMPRESSION_DICTIONARY_BUILD • IMPLICIT_REBIND • INDEX_RECREATE • LOCK • LOCK_ESCALATION • QP_QUEUE • REMOTE_REQUEST • REQUEST • ROUTINE • WLM_QUEUE この列に使用できる値についての詳細は、 1330 ページの表 333 を参照してください。
EVENT_STATE	VARCHAR(32)	このエージェントによって最後に処理されたイベントの状態。可能な値は、以下のとおりです。 • EXECUTING • IDLE この列に使用できる値についての詳細は、 1330 ページの表 333 を参照してください。

表 332. *WLM_GET_SERVICE_CLASS_AGENTS* によって戻される情報 (続き)

列名	データ・タイプ	説明
REQUEST_ID	VARCHAR(64)	要求 ID。この値は、*application_handle* の値と組み合わせて指定される場合のみ固有です。これによって、長い時間を要する 1 つの要求と複数の要求とを区別することができます。例えば、1 つの長いフェッチと複数のフェッチを区別するなどです。
REQUEST_TYPE	VARCHAR(32)	要求のタイプ。可能な値は、以下のとおりです。 • コーディネーター・エージェントの場合: – CLOSE – COMMIT – COMPILE – DESCRIBE – EXCSQLSET – EXECIMMD – EXECUTE – FETCH – INTERNAL *number* (*number* は内部定数の値) – OPEN – PREPARE – REBIND – REDISTRIBUTE – REORG – ROLLBACK – RUNSTATS • AGENT_SUBTYPE が DSS または SMP であるサブエージェントの場合 – サブセクション番号がゼロ以外の場合は SUBSECTION:*subsection number* 形式のサブセクション番号、ゼロの場合は NULL

表 332. *WLM_GET_SERVICE_CLASS_AGENTS* によって戻される情報 (続き)

列名	データ・タイプ	説明
REQUEST_TYPE (続く)	VARCHAR(32)	• AGENT_SUBTYPE が RPC であるサブエージェントの場合 – ABP – CATALOG – INTERNAL – REORG – RUNSTATS – WLM • AGENT_SUBTYPE が OTHER のサブエージェントの場合 – ABP – APP_RBSVPT – APP_RELSVPT – BACKUP – CLOSE – EXTERNAL_RBSVPT – EVMON – FORCE – FORCE_ALL – INTERNAL *number* (*number* は内部定数の値) – INTERRUPT – NOOP (要求がない場合) – QP – REDISTRIBUTE – STMT_RBSVPT – STOP_USING – UPDATE_DBM_CFG – WLM
NESTING_LEVEL	INTEGER	ID が activity_id であるアクティビティーのネスト・レベル。ネスト・レベルは、このアクティビティーが一番上の親アクティビティー内でネストされる深さです。
INVOCATION_ID	INTEGER	1 つのルーチン呼び出しを、作業単位内の同じネスト・レベルにある他の呼び出しと区別するための ID。これは作業単位内の特定のネスト・レベルにおいて固有です。
ROUTINE_ID	INTEGER	ルーチンのユニーク ID。列がルーチンの一部でない場合、列の値は NULL です。

注: EVENT_STATE、EVENT_TYPE、および EVENT_OBJECT 列値の可能な組み合わせを、以下の表にリストします。

表 333. EVENT_STATE、EVENT_TYPE、および EVENT_OBJECT 列値の可能な組み合わせ

イベント記述	EVENT_STATE 値	EVENT_TYPE 値	EVENT_OBJECT 値
ロックの獲得	IDLE	ACQUIRE	LOCK
ロックのエスカレート	EXECUTING	PROCESS	LOCK_ESCALATION
要求の処理	EXECUTING	PROCESS	REQUEST
新規要求の待機	IDLE	WAIT	REQUEST
リモート・パーティションで処理される要求の待機	IDLE	WAIT	REMOTE_REQUEST
Query Patroller キューの待機	IDLE	WAIT	QP_QUEUE
WLM threshold キューの待機	IDLE	WAIT	WLM_QUEUE
ルーチンの処理	EXECUTING	PROCESS	ROUTINE
索引の再作成	EXECUTING	PROCESS	INDEX_RECREATE
コンプレッション・ディクショナリーの作成	EXECUTING	PROCESS	COMP_DICT_BLD
暗黙的な再バインド	EXECUTING	PROCESS	IMPLICIT_REBIND

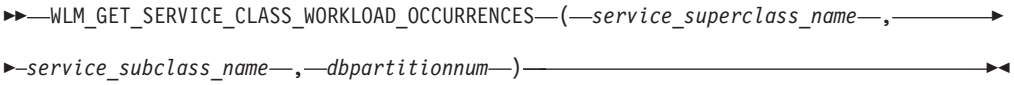
WLM_GET_SERVICE_CLASS_WORKLOAD _OCCURRENCES - ワークロード・オカレンスのリスト

特定のパーティション上の指定されたサービス・クラスで実行しているすべてのワークロード・オカレンスのリストを戻します。

注: この表関数は使用すべきではなく、 1017 ページの『WLM_GET_SERVICE_CLASS_WORKLOAD _OCCURRENCES_V97 - ワークロード・オカレンスのリスト』 に置き換えられました。

ワークロード・オカレンスとは、属性がワークロードの定義と一致しており、そのためにワークロードに関連付けられた、またはワークロードに割り当てられた特定のデータベース接続です。

構文



スキーマは SYSPROC です。

表関数パラメーター

service_superclass_name
現在接続されているデータベースで有効なサービス・スーパークラス名を指定す

る、タイプ VARCHAR(128) の入力引数。引数が NULL または空ストリングである場合、他のパラメーターが一致する、データベース内のすべてのスーパークラスについてデータが取得されます。

service_subclass_name
ワークロード・オカレンスのターゲット・サービス・サブクラス。このワークロード・オカレンスによってサブミットされる作業は、別のサブクラスにマップまたは再マップされるアクティビティーを除いて、すべてターゲット・サービス・スーパークラスの下のこのサービス・サブクラスで実行されます。

dbpartitionnum
現在接続されているデータベースと同じインスタンス内の有効なパーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションには -1、すべてのデータベース・パーティションには -2 を指定します。NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES 関数に対する EXECUTE 特権。

例

管理者がどのワークロード・オカレンスがシステム上で全体として実行しているかを調べたい場合、*service_superclass_name* および *service_subclass_name* NULL 値または空ストリングを指定し、 *dbpartitionnum* に -2 を指定した WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES 関数を呼び出すことができます。

```
SELECT SUBSTR(SERVICE_SUPERCLASS_NAME,1,19) AS SUPERCLASS_NAME,
       SUBSTR(SERVICE_SUBCLASS_NAME,1,18) AS SUBCLASS_NAME,
       SUBSTR(CHAR(DBPARTITIONNUM),1,4) AS PART,
       SUBSTR(CHAR(COORD_PARTITION_NUM),1,4) AS COORDPART,
       SUBSTR(CHAR(APPLICATION_HANDLE),1,7) AS APPHNDL,
       SUBSTR(WORKLOAD_NAME,1,22) AS WORKLOAD_NAME,
       SUBSTR(CHAR(WORKLOAD_OCCURRENCE_ID),1,6) AS WLO_ID
FROM TABLE(WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES
      (CAST(NULL AS VARCHAR(128)), CAST(NULL AS VARCHAR(128)), -2))
       AS SCINFO
ORDER BY SUPERCLASS_NAME, SUBCLASS_NAME, PART, APPHNDL,
       WORKLOAD_NAME, WLO_ID
```

システムに 4 つのデータベース・パーティションがあり、現時点で 2 つのワークロードを実行していると想定すると、上記の照会は以下のような結果を生成します。

```
SUPERCLASS_NAME     SUBCLASS_NAME      PART COORDPART ...
------------------- ------------------ ---- --------- ...
SYSDEFAULTMAINTENAN SYSDEFAULTSUBCLASS 0    0         ...
SYSDEFAULTSYSTEMCLA SYSDEFAULTSUBCLASS 0    0         ...
SYSDEFAULTUSERCLASS SYSDEFAULTSUBCLASS 0    0         ...
SYSDEFAULTUSERCLASS SYSDEFAULTSUBCLASS 0    0         ...
SYSDEFAULTUSERCLASS SYSDEFAULTSUBCLASS 1    0         ...
SYSDEFAULTUSERCLASS SYSDEFAULTSUBCLASS 1    0         ...
SYSDEFAULTUSERCLASS SYSDEFAULTSUBCLASS 2    0         ...
SYSDEFAULTUSERCLASS SYSDEFAULTSUBCLASS 2    0         ...
SYSDEFAULTUSERCLASS SYSDEFAULTSUBCLASS 3    0         ...
SYSDEFAULTUSERCLASS SYSDEFAULTSUBCLASS 3    0         ...
```

この照会からの出力 (続き)。

```
... APPHNDL WORKLOAD_NAME          WLO_ID
... ------- ---------------------- ------
... -       -                      -
... -       -                      -
... 1       SYSDEFAULTUSERWORKLOAD 1
... 2       SYSDEFAULTUSERWORKLOAD 2
... 1       SYSDEFAULTUSERWORKLOAD 1
... 2       SYSDEFAULTUSERWORKLOAD 2
... 1       SYSDEFAULTUSERWORKLOAD 1
... 2       SYSDEFAULTUSERWORKLOAD 2
... 1       SYSDEFAULTUSERWORKLOAD 1
... 2       SYSDEFAULTUSERWORKLOAD 2
```

使用上の注意

このパラメーターの影響として、ANDing されます。つまり、サービス・スーパークラス SUP_A とサブクラス SUB_B などの競合入力パラメーターを、SUB_B が SUP_A のサブクラスにならないように指定する場合、行は戻されません。

注: ワークロード・オカレンスについて報告される統計 (例えば、coord_act_completed_total) が、対応するワークロード統計と結合されると、ワークロード・オカレンスについて報告される統計が各作業単位の初めにリセットされます。

戻される情報

表 334. WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES について戻される情報

列名	データ・タイプ	説明
SERVICE_SUPERCLASS_NAME	VARCHAR(128)	このレコードが収集されたサービス・スーパークラスの名前。
SERVICE_SUBCLASS_NAME	VARCHAR(128)	このレコードが収集されたサービス・サブクラスの名前。
DBPARTITIONNUM	SMALLINT	このレコードが収集されたパーティション番号。
COORD_PARTITION_NUM	SMALLINT	指定されたワークロード・オカレンスのコーディネーター・パーティションのパーティション番号。
APPLICATION_HANDLE	BIGINT	システム全体におけるアプリケーションのユニーク ID。単一パーティション・データベースの場合、この ID は 16 ビットのカウンターで構成されます。複数パーティション・データベースの場合、この ID はコーディネーター・パーティション番号と 16 ビットのカウンターを連結したもので構成されます。さらに、この ID はアプリケーションが 2 次接続を行うすべてのパーティションにおいて同じです。
WORKLOAD_NAME	VARCHAR(128)	このレコードが収集されたワークロードの名前。
WORKLOAD_OCCURRENCE_ID	INTEGER	ワークロード・オカレンスの ID。ワークロード・オカレンスがコーディネーター・データベース・パーティション番号およびワークロード名と結合されていなければ、これはワークロード・オカレンスを一意的に識別しません。

表 334. WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES について戻される情報 (続き)

列名	データ・タイプ	説明
WORKLOAD_OCCURRENCE_STATE	VARCHAR(32)	使用できる値は以下のとおりです。 • DECOUPLED - ワークロード・オカレンスには割り当てられたコーディネーター・エージェントがありません (コンセントレーターの場合)。 • DISCONNECTPEND - ワークロード・オカレンスはデータベースから切断中です。 • FORCED - ワークロード・オカレンスが強制されました。 • INTERRUPTED - ワークロード・オカレンスが中断されました。 • QUEUED - ワークロード・オカレンス・コーディネーター・エージェントが、Query Patroller またはワークロード管理キューイングしきい値によってキューに入れられています。パーティション・データベース環境では、この状態は、コーディネーター・エージェントがカタログ・パーティションに対する RPC を行ってしきい値チケットを取得したものの、まだ応答を受け取っていないことを示す場合があります。 • TRANSIENT - ワークロード・オカレンスがまだサービス・スーパークラスにマップされていません。 • UOWEXEC - ワークロード・オカレンスは要求を処理中です。 • UOWWAIT - ワークロード・オカレンスはクライアントからの要求を待機中です。
UOW_ID	INTEGER	アプリケーション内の固有の作業単位 ID。
SYSTEM_AUTH_ID	VARCHAR(128)	ワークロード・オカレンスがシステムに注入されるときに使用したシステム許可 ID。
SESSION_AUTH_ID	VARCHAR(128)	ワークロード・オカレンスがシステムに注入されるときに使用したセッション許可 ID。
APPLICATION_NAME	VARCHAR(128)	このワークロード・オカレンスを作成したアプリケーションの名前。
CLIENT_WRKSTNNAME	VARCHAR(255)	このワークロード・オカレンスの CLIENT_WRKSTNNAME 特殊レジスターの現行値。
CLIENT_ACCTNG	VARCHAR(255)	このワークロード・オカレンスの CLIENT_ACCTNG 特殊レジスターの現行値。
CLIENT_USER	VARCHAR(255)	このワークロード・オカレンスの CLIENT_USERID 特殊レジスターの現行値。
CLIENT_APPLNAME	VARCHAR(255)	このワークロード・オカレンスの CLIENT_APPLNAME 特殊レジスターの現行値。

表 334. WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES について戻される情報 (続き)

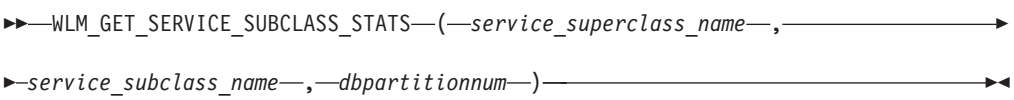
列名	データ・タイプ	説明
COORD_ACT_COMPLETED_TOTAL	INTEGER	このワークロード・オカレンスの現在の作業単位でこれまでに完了した、任意のネスト・レベルのコーディネーター・アクティビティーの数。この統計は、このワークロード・オカレンスのアクティビティーが完了し、各作業単位の初めにリセットされるたびに更新されます。
COORD_ACT_ABORTED_TOTAL	INTEGER	このワークロード・オカレンスの現在の作業単位でこれまでにアボートしたコーディネーター・アクティビティーの数。この統計は、このワークロード・オカレンスのアクティビティーがアボートされ、各作業単位の初めにリセットされるたびに更新されます。
COORD_ACT_REJECTED_TOTAL	INTEGER	このワークロード・オカレンスの現在の作業単位でこれまでにリジェクトしたコーディネーター・アクティビティーの数。アクティビティーが実行抑制作業アクションまたは予測しきい値のいずれかによって実行を妨げられている場合、そのアクティビティーはリジェクトとしてカウントされます。この統計は、このワークロード・オカレンスのアクティビティーがリジェクトされ、各作業単位の初めにリセットされるたびに更新されます。
CONCURRENT_ACT_TOP	INTEGER	現在の作業単位でこのワークロード・オカレンスについて到達している、実行状態 (アイドルおよび待機中を含む) またはキューに入れられた状態の任意のネスト・レベルの並行アクティビティーの最高数。この統計は、各作業単位の初めにリセットされます。

WLM_GET_SERVICE_SUBCLASS_STATS - サービス・サブクラスの統計を戻す

注: この表関数は使用すべきではなく、 1022 ページの『WLM_GET_SERVICE_SUBCLASS_STATS_V97 表関数 - サービス・サブクラスの統計を戻す』 に置き換えられました。

この関数は、1 つ以上のサービス・サブクラスの基本統計を戻します。

構文



スキーマは SYSPROC です。

表関数パラメーター

service_superclass_name
この関数を呼び出すときに現在接続されているデータベースと同じデータベース内の有効なサービス・スーパークラス名を指定する、タイプ VARCHAR(128) の入力引数。引数が NULL または空ストリングである場合、データベース内のすべてのスーパークラスについてデータが取得されます。

service_subclass_name
この関数を呼び出すときに現在接続されているデータベースと同じデータベース内の有効なサービス・サブクラス名を指定する、タイプ VARCHAR(128) の入力引数。引数が NULL または空ストリングである場合、データベース内のすべてのサブクラスについてデータが取得されます。

dbpartitionnum
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なパーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションには -1、すべてのデータベース・パーティションには -2 を指定します。 NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

WLM_GET_SERVICE_SUBCLASS_STATS 関数に対する EXECUTE 特権。

例

例 1: すべてのアクティビティーを実行前に DB2 サービス・クラスにマップする必要があることから、サービス・クラス統計表関数を使用し、すべてのパーティション上のすべてのサービス・クラスを照会して、システムの全体的な状態を定期的にモニターすることができます。引数として NULL 値を渡すと、最後の引数である *dbpartitionnum* を除いて、その引数が結果を制限しないことを表します。*dbpartitionnum* では、値 -2 がすべてのデータベース・パーティション上のデータを戻すことを表します。この例は、アクティビティー存続期間の平均および標準偏差などのサービス・クラス統計を秒単位で戻します。

```
SELECT SUBSTR(SERVICE_SUPERCLASS_NAME,1,19) AS SUPERCLASS_NAME,
       SUBSTR(SERVICE_SUBCLASS_NAME,1,18) AS SUBCLASS_NAME,
       SUBSTR(CHAR(DBPARTITIONNUM),1,4) AS PART,
       CAST(COORD_ACT_LIFETIME_AVG / 1000 AS DECIMAL(9,3))
         AS AVGLIFETIME,
       CAST(COORD_ACT_LIFETIME_STDDEV / 1000 AS DECIMAL(9,3))
         AS STDDEVLIFETIME,
       SUBSTR(CAST(LAST_RESET AS VARCHAR(30)),1,16) AS LAST_RESET
  FROM TABLE(WLM_GET_SERVICE_SUBCLASS_STATS(CAST(NULL AS VARCHAR(128)),
       CAST(NULL AS VARCHAR(128)), -2)) AS SCSTATS
  ORDER BY SUPERCLASS_NAME, SUBCLASS_NAME, PART
```

以下はこの照会の出力例です。

```
SUPERCLASS_NAME     SUBCLASS_NAME      PART ...
------------------- ------------------ ---- ...
SYSDEFAULTUSERCLASS SYSDEFAULTSUBCLASS 0    ...
SYSDEFAULTUSERCLASS SYSDEFAULTSUBCLASS 1    ...
SYSDEFAULTUSERCLASS SYSDEFAULTSUBCLASS 2    ...
SYSDEFAULTUSERCLASS SYSDEFAULTSUBCLASS 3    ...
```

この照会からの出力 (続き)。

```
... AVGLIFETIME STDDEVLIFETIME LAST_RESET
... ----------- -------------- ----------------
...     691.242         34.322 2006-07-24-11.44
...     644.740         22.124 2006-07-24-11.44
...     612.431         43.347 2006-07-24-11.44
...     593.451         28.329 2006-07-24-11.44
```

例 2: この同じ表関数が、各パーティションのサービス・クラスで実行されているコーディネーター・アクティビティーの平均並行性の最高値も示します。

```
SELECT SUBSTR(SERVICE_SUPERCLASS_NAME,1,19) AS SUPERCLASS_NAME,
       SUBSTR(SERVICE_SUBCLASS_NAME,1,18) AS SUBCLASS_NAME,
       SUBSTR(CHAR(DBPARTITIONNUM),1,4) AS PART,
       CONCURRENT_ACT_TOP AS ACTTOP,
       CONCURRENT_WLO_TOP AS CONNTOP
FROM TABLE(WLM_GET_SERVICE_SUBCLASS_STATS(CAST(NULL AS VARCHAR(128)),
       CAST(NULL AS VARCHAR(128)), -2)) AS SCSTATS
ORDER BY SUPERCLASS_NAME, SUBCLASS_NAME, PART
```

以下はこの照会の出力例です。

```
SUPERCLASS_NAME     SUBCLASS_NAME      PART ACTTOP    CONNTOP
------------------- ------------------ ---- --------- ---------
SYSDEFAULTUSERCLASS SYSDEFAULTSUBCLASS 0           10         7
SYSDEFAULTUSERCLASS SYSDEFAULTSUBCLASS 1            0         0
SYSDEFAULTUSERCLASS SYSDEFAULTSUBCLASS 2            0         0
SYSDEFAULTUSERCLASS SYSDEFAULTSUBCLASS 3            0         0
```

この表関数の出力から、特定のデータベースの各パーティションの「負荷」の概要を、的確に知ることができます。この出力は、平均実行時間とアクティビティー数を検査することによって得られます。これらの表関数によって戻される高水準値が大きく変わった場合は、システムの負荷の変化を示していることがあります。

使用上の注意

一部の統計は、対応するサービス・サブクラスの COLLECT AGGREGATE ACTIVITY DATA および COLLECT AGGREGATE REQUEST DATA 設定が「NONE」以外の値に設定されている場合のみ、戻されます。

WLM_GET_SERVICE_SUBCLASS_STATS 表関数は、サービス・サブクラスごとおよびパーティションごとに 1 行のデータを戻します。 (パーティション上の) サービス・クラス間または (1 つ以上のサービス・クラスの) パーティション間の集約はありません。しかし、集約は上の例で示された SQL 照会を使用して実行できます。

これらのパラメーターは、結果的に「AND」節として論理的に結合されます。つまり、例えばスーパークラス SUPA と SUPA のサブクラスでないサブクラス SUBB などの競合する入力パラメーターを指定した場合、行は戻されません。

戻される情報

表 335. WLM_GET_SERVICE_SUBCLASS_STATS について戻される情報

列名	データ・タイプ	説明
SERVICE_SUPERCLASS_NAME	VARCHAR(128)	このレコードが収集されたサービス・スーパークラスの名前。
SERVICE_SUBCLASS_NAME	VARCHAR(128)	このレコードが収集されたサービス・サブクラスの名前。

表 335. WLM_GET_SERVICE_SUBCLASS_STATS について戻される情報 (続き)

列名	データ・タイプ	説明
DBPARTITIONNUM	SMALLINT	このレコードが収集されたパーティション番号。
LAST_RESET	TIMESTAMP	統計が最後にリセットされた時刻。以下の 4 つのイベントが発生する可能性があります。これらは、統計のリセットを起動し、このタイム・スタンプを更新します。 • WLM_COLLECT_STATS プロシージャーが呼び出される。 • WLM_COLLECT_INT 構成パラメーターによって制御された定期的なコレクションおよびリセット・プロセスにより、コレクションおよびリセットが発生する。 • データベースが再活動化される。 • 統計が報告されているサービス・サブクラスが変更され、その変更がコミットされた。 LAST_RESET タイム・スタンプがローカル時刻である。
COORD_ACT_COMPLETED_TOTAL	BIGINT	ユーザーが最後のリセット以降にサブミットし、正常に完了したコーディネーター・アクティビティーの合計数。この数は、アクティビティーが完了するたびに更新されます。 異なるサービス・サブクラスにアクティビティーを再マップする場合、そのアクティビティーのカウント対象となるのは、それが完了したサブクラスの合計だけです。
COORD_ACT_ABORTED_TOTAL	BIGINT	ユーザーが最後のリセット以降にサブミットし、エラーを出して完了したコーディネーター・アクティビティーの合計数。この数は、アクティビティーが異常終了するたびに更新されます。 異なるサービス・サブクラスにアクティビティーを再マップする場合、そのアクティビティーのカウント対象となるのは、それが打ち切ったサブクラスの合計だけです。
COORD_ACT_REJECTED_TOTAL	BIGINT	ユーザーが最後のリセット以降にサブミットし、実行が許可される代わりに実行前にリジェクトされたコーディネーター・アクティビティーの合計数。アクティビティーが実行抑制作業アクションまたは予測しきい値のいずれかによって実行を妨げられている場合、そのアクティビティーはリジェクトとしてカウントされます。この数は、アクティビティーがリジェクトされるたびに更新されます。

表 335. WLM_GET_SERVICE_SUBCLASS_STATS について戻される情報 (続き)

列名	データ・タイプ	説明
CONCURRENT_ACT_TOP	INTEGER	このサービス・サブクラスについて到達している、実行状態 (アイドルおよび待機中を含む) またはキューに入れられた状態の任意のネスト・レベルの並行アクティビティーの最高数。
COORD_ACT_LIFETIME_TOP	BIGINT	すべてのネスト・レベルでカウントされる、コーディネーター・アクティビティー存続期間の最高水準点。サービス・クラスの COLLECT AGGREGATE ACTIVITY DATA が NONE の場合は、NULL です。単位はミリ秒です。 サービス・クラスに再マップされるサブクラスが含まれている場合にこの統計を効率的に使用するには、サービス・サブクラスの COORD_ACT_LIFETIME_TOP 最高水準点を、同じ再マップしきい値によって影響を受ける他のサブクラスのこの最高水準点と集約する必要があります。こうした値を集約する必要があるのは、アクティビティーを完了するにはその前に別のサービス・サブクラスに再マップする必要がありますが、再マップされる前に他のサービス・サブクラスでアクティビティーが費やす時間はアクティビティーが完了したサービス・クラスでのみカウント対象となるためです。
COORD_ACT_LIFETIME_AVG	DOUBLE	最後のリセット以降、このサービス・サブクラスに関連付けられたネスト・レベル 0 のコーディネーター・アクティビティーの存続期間の算術平均。内部的に追跡された平均がオーバーフローした場合、値 -2 が戻されます。サービス・クラスの COLLECT AGGREGATE ACTIVITY DATA が NONE の場合は、NULL です。単位はミリ秒です。 サービス・サブクラスの COORD_ACT_LIFETIME_AVG は、完了前にサブクラスを通過したものの別のサブクラスに再マップされるアクティビティーの影響は受けません。

表 335. WLM_GET_SERVICE_SUBCLASS_STATS について戻される情報 (続き)

列名	データ・タイプ	説明
COORD_ACT_LIFETIME_STDDEV	DOUBLE	最後のリセット以降、このサービス・サブクラスに関連付けられたネスト・レベル 0 のコーディネーター・アクティビティーの存続期間の標準偏差。サービス・クラスの COLLECT AGGREGATE ACTIVITY DATA が NONE の場合は、NULL です。単位はミリ秒です。この標準偏差はコーディネーター・アクティビティーの存続期間ヒストグラムから計算され、ヒストグラムのサイズがデータに合わせて正しく設定されていない場合は不正確になることがあります。値が最後のヒストグラム bin に入る場合、値 -1 が戻されます。 サービス・サブクラスの COORD_ACT_LIFETIME_STDDEV は、完了前にサービス・サブクラスを通過したものの別のサブクラスに再マップされるアクティビティーの影響は受けません。
COORD_ACT_EXEC_TIME_AVG	DOUBLE	最後のリセット以降、このサービス・サブクラスに関連付けられたネスト・レベル 0 のコーディネーター・アクティビティーの実行時間の算術平均。内部的に追跡された平均がオーバーフローした場合、値 -2 が戻されます。サービス・クラスの COLLECT AGGREGATE ACTIVITY DATA が NONE の場合は、NULL です。単位はミリ秒です。 サービス・サブクラスの実行時間平均は、完了前にサブクラスを通過したものの別のサブクラスに再マップされるアクティビティーの影響は受けません。
COORD_ACT_EXEC_TIME_STDDEV	DOUBLE	最後のリセット以降、このサービス・サブクラスに関連付けられたネスト・レベル 0 のコーディネーター・アクティビティーの実行時間の標準偏差。単位はミリ秒です。この標準偏差はコーディネーター・アクティビティーの実行時間ヒストグラムから計算され、ヒストグラムのサイズがデータに合わせて正しく設定されていない場合は不正確になることがあります。値が最後のヒストグラム bin に入る場合、値 -1 が戻されます。 サービス・サブクラスの実行時間標準偏差は、完了前にサブクラスを通過したものの別のサブクラスに再マップされるアクティビティーの影響は受けません。

表 335. WLM_GET_SERVICE_SUBCLASS_STATS について戻される情報 (続き)

列名	データ・タイプ	説明
COORD_ACT_QUEUE_TIME_AVG	DOUBLE	最後のリセット以降、このサービス・サブクラスに関連付けられたネスト・レベル 0 のコーディネーター・アクティビティーのキュー時間の算術平均。内部的に追跡された平均がオーバーフローした場合、値 -2 が戻されます。サービス・クラスの COLLECT AGGREGATE ACTIVITY DATA が NONE の場合は、NULL です。単位はミリ秒です。 キュー時間平均のカウント対象は、アクティビティーがキューに入れられたサービス・サブクラスだけです。
COORD_ACT_QUEUE_TIME_STDDEV	DOUBLE	最後のリセット以降、このサービス・サブクラスに関連付けられたネスト・レベル 0 のコーディネーター・アクティビティーのキュー時間の標準偏差。サービス・クラスの COLLECT AGGREGATE ACTIVITY DATA が NONE の場合は、NULL です。単位はミリ秒です。この標準偏差はコーディネーター・アクティビティーのキュー時間ヒストグラムから計算され、ヒストグラムのサイズがデータに合わせて正しく設定されていない場合は不正確になることがあります。値が最後のヒストグラム bin に入る場合、値 -1 が戻されます。 キュー時間標準偏差のカウント対象は、アクティビティーがキューに入れられたサービス・サブクラスだけです。
NUM_REQUESTS_ACTIVE	BIGINT	この表関数の実行時にサービス・サブクラスで実行している要求の数。
NUM_REQUESTS_TOTAL	BIGINT	最後のリセット以降、このサービス・サブクラスで実行を終了する要求の数。これは、アクティビティー内の要求のメンバーシップに関係なく、任意の要求に適用されます。このサービス・サブクラスの COLLECT AGGREGATE REQUEST DATA が NONE に設定される場合、この列の値は NULL です。 サービス・サブクラスの NUM_REQUESTS_TOTAL は、サービス・サブクラスを通過したものの、その中で完了しない要求の影響は受けません。

表 335. WLM_GET_SERVICE_SUBCLASS_STATS について戻される情報 (続き)

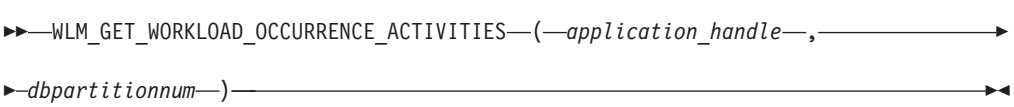
列名	データ・タイプ	説明
REQUEST_EXEC_TIME_AVG	DOUBLE	最後のリセット以降、このサービス・サブクラスに関連付けられた要求の実行時間の算術平均。単位はミリ秒です。内部的に追跡された平均がオーバーフローした場合、値 -2 が戻されます。このサービス・クラスの COLLECT AGGREGATE REQUEST DATA が NONE に設定される場合、この列の値は NULL です。 サービス・サブクラスの実行時間平均は、サブクラスを通過したものの、その中で完了しない要求の影響は受けません。
REQUEST_EXEC_TIME_STDDEV	DOUBLE	最後のリセット以降、このサービス・サブクラスに関連付けられた要求の実行時間の標準偏差。単位はミリ秒です。このサービス・クラスの COLLECT AGGREGATE REQUEST DATA が NONE に設定される場合、この列の値は NULL です。この標準偏差は要求実行時間ヒストグラムから計算され、ヒストグラムのサイズがデータに合わせて正しく設定されていない場合は不正確になることがあります。値が最後のヒストグラム bin に入る場合、値 -1 が戻されます。 サービス・サブクラスの実行時間標準偏差は、サブクラスを通過したものの、その中で完了しない要求の影響は受けません。
REQUEST_EXEC_TIME_TOTAL	BIGINT	最後のリセット以降、このサービス・サブクラスに関連付けられた要求の実行時間の合計。単位はミリ秒です。このサービス・クラスの COLLECT AGGREGATE REQUEST DATA が NONE に設定される場合、この列の値は NULL です。この合計は要求実行時間ヒストグラムから計算され、ヒストグラムのサイズがデータに合わせて正しく設定されていない場合は不正確になることがあります。値が最後のヒストグラム bin に入る場合、値 -1 が戻されます。 サービス・サブクラスの実行時間合計は、サブクラスを通過したものの、その中で完了しない要求の影響は受けません。

WLM_GET_WORKLOAD_OCCURRENCE _ACTIVITIES - アクティビティーのリストを戻す

WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES 関数は、指定されたパーティション上の指定されたアプリケーションからサブミットされ、まだ完了していないすべてのアクティビティーのリストを戻します。

注: この表関数は使用すべきではなく、 1034 ページの『WLM_GET_WORKLOAD_OCCURRENCE _ACTIVITIES_V97 - アクティビティーのリストを戻す』 に置き換えられました。

構文



スキーマは SYSPROC です。

表関数パラメーター

application_handle
アクティビティーのリストが戻されるアプリケーション・ハンドルを指定する、タイプ BIGINT の入力引数。引数が NULL である場合、データベース内のすべてのアプリケーションについてデータが取得されます。

dbpartitionnum
現在接続されているデータベースと同じインスタンス内の有効なパーティション番号を指定する、タイプ INTEGER の入力引数。現在のデータベース・パーティションに対しては -1、すべてのデータベース・パーティションに対しては -2 を指定します。 NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES 関数に対する EXECUTE 特権。

例

アプリケーション・ハンドルを識別した後、このアプリケーションで現在実行中のすべてのアクティビティーを検索できます。 LIST APPLICATIONS コマンドを使用して判別されるアプリケーション・ハンドルが 1 であるアプリケーションのアクティビティーをリストすることを管理者が望んでいるという場面を、例として考えてみましょう。管理者は以下の照会を実行します。

```
SELECT SUBSTR(CHAR(COORD_PARTITION_NUM),1,5) AS COORD,
       SUBSTR(CHAR(DBPARTITIONNUM),1,4) AS PART,
       SUBSTR(CHAR(UOW_ID),1,5) AS UOWID,
       SUBSTR(CHAR(ACTIVITY_ID),1,5) AS ACTID,
       SUBSTR(CHAR(PARENT_UOW_ID),1,8) AS PARUOWID,
       SUBSTR(CHAR(PARENT_ACTIVITY_ID),1,8) AS PARACTID,
              ACTIVITY_TYPE AS ACTTYPE,
       SUBSTR(CHAR(NESTING_LEVEL),1,7) AS NESTING
  FROM TABLE(WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES(1, -2)) AS WLOACTS
  ORDER BY PART, UOWID, ACTID
```

照会からの出力例は、次のようになります。

```
COORD PART UOWID ACTID PARUOWID PARACTID ACTTYPE  NESTING
----- ---- ----- ----- -------- -------- -------- -------
0     0    2     3     -        -        CALL     0
0     0    2     5     2        3        READ_DML 1
```

戻される情報

表 336. WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES によって戻される情報

列名	データ・タイプ	説明
APPLICATION_HANDLE	BIGINT	システム全体におけるアプリケーションのユニーク ID。単一パーティション・データベースの場合、この ID は 16 ビットのカウンターで構成されます。複数パーティション・データベースの場合、この ID はコーディネーター・パーティション番号と 16 ビットのカウンターを連結したもので構成されます。さらに、この ID はアプリケーションが 2 次接続を行うすべてのパーティションにおいて同じです。
DBPARTITIONNUM	SMALLINT	このレコードが収集されたパーティション番号。
COORD_PARTITION_NUM	SMALLINT	アクティビティーのコーディネーター・パーティション。
LOCAL_START_TIME	TIMESTAMP	アクティビティーがパーティションで作業を開始した現地時間。アクティビティーがシステムに入ったが、キューに入れられており、実行を開始していない場合は、この列の値は NULL です。
UOW_ID	INTEGER	アクティビティーが開始した作業単位のユニーク ID。
ACTIVITY_ID	INTEGER	作業単位内の固有のアクティビティー ID。
PARENT_UOW_ID	INTEGER	アクティビティーの親アクティビティーが開始した作業単位のユニーク ID。アクティビティーに親アクティビティーがない場合、またはアクティビティーがリモート・パーティションにある場合、この列の値は NULL です。
PARENT_ACTIVITY_ID	INTEGER	親のアクティビティー ID が ACTIVITY_ID 列の値である、作業単位内のアクティビティーのユニーク ID。アクティビティーに親アクティビティーがない場合、またはアクティビティーがリモート・パーティションにある場合、この列の値は NULL です。

表 336. WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES によって戻される情報 (続き)

列名	データ・タイプ	説明
ACTIVITY_STATE	VARCHAR(32)	アクティビティーの状態。可能な値は、以下のとおりです。 **CANCEL_PENDING** アクティビティーの要求をアクティブに実行しているエージェントがないため、アクティビティーは取り消されました。次に要求がアクティビティーの一部としてサブミットされるときに、そのアクティビティーは取り消され、SQL4725N エラーが生成されます。 **EXECUTING** エージェントはアクティビティーの要求をアクティブに実行しています。 **IDLE** アクティビティーの要求をアクティブに処理しているエージェントがありません。 **INITIALIZING** アクティビティーはサブミットされましたが、まだ実行を開始していません。初期化状態中に、予測しきい値がアクティビティーに適用され、アクティビティーが実行を許可されるかどうかを判別します。

表 336. WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES によって戻される情報 (続き)

列名	データ・タイプ	説明
ACTIVITY_STATE (続き)	VARCHAR(32)	アクティビティーの状態。可能な値は、以下のとおりです。 **QP_CANCEL_PENDING** アクティビティーが WLM_CANCEL_ACTIVITY プロシージャーではなく、Query Patroller によって取り消された点を除いて、この状態は、CANCEL_PENDING 状態と同じです。 **QP_QUEUED** アクティビティーは Query Patroller によってキューに入れられます。 **QUEUED** アクティビティーが、ワークロード管理キューイングしきい値によってキューに入れられています。パーティション・データベース環境では、この状態は、コーディネーター・エージェントがカタログ・パーティションに対する RPC を行ってしきい値チケットを取得したものの、まだ応答を受け取っていないことを示す場合があります。この状態は、アクティビティーがワークロード管理キューイングしきい値によってキューに入れられているか、それほど時間が経過していない場合はアクティビティーがそのチケットを取得する処理中であることを示すことがあります。アクティビティーがキューに入れられているかどうかについての詳細を知るために、どのエージェントがアクティビティーで作業しているかを判別し、カタログ・パーティションにあるエージェントの EVENT_OBJECT の値が WLM_QUEUE であるかどうかを調べることができます。 **TERMINATING** アクティビティーは実行を完了し、システムから除去されています。

表 336. WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES によって戻される情報 (続き)

列名	データ・タイプ	説明
ACTIVITY_TYPE	VARCHAR(32)	アクティビティー・タイプ。可能な値は、以下のとおりです。 • CALL • DDL • LOAD • OTHER • READ_DML • WRITE_DML 各アクティビティー・タイプに関連付けられた SQL ステートメントのさまざまなタイプの説明については、「ワークロード・マネージャー *ガイドおよびリファレンス* 」の『ワーク・クラスによる作業タイプの識別』を参照してください。
NESTING_LEVEL	INTEGER	このアクティビティーが一番上の親アクティビティー内でネストされる深さ。
INVOCATION_ID	INTEGER	1 つのルーチン呼び出しを、作業単位内の同じネスト・レベルにある他の呼び出しと区別するための ID。これは作業単位内の特定のネスト・レベルにおいて固有です。
ROUTINE_ID	INTEGER	ルーチンのユニーク ID。
UTILITY_ID	INTEGER	以下の値のいずれか。 • アクティビティーがユーティリティーの場合、値はユーティリティーの ID です。 • アクティビティーがユーティリティーでない場合、値は NULL です。
SERVICE_CLASS_ID	INTEGER	このアクティビティーが属するサービス・クラスのユニーク ID。
DATABASE_WORK_ACTION_SET_ID	INTEGER	以下の値のいずれか。 • このアクティビティーがデータベース有効範囲の作業クラスに分類されている場合、値はこの作業クラスがメンバーとなっている作業クラス・セットの ID です。 • このアクティビティーがデータベース有効範囲の作業クラスに分類されていない場合、値は NULL です。
DATABASE_WORK_CLASS_ID	INTEGER	以下の値のいずれか。 • このアクティビティーがデータベース有効範囲の作業クラスに分類されている場合、値は作業クラスの ID です。 • このアクティビティーがデータベース有効範囲の作業クラスに分類されていない場合、値は NULL です。

表 336. WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES によって戻される情報 (続き)

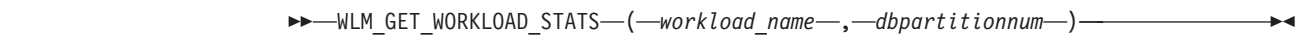
列名	データ・タイプ	説明
SERVICE_CLASS_WORK_ACTION_SET_ID	INTEGER	以下の値のいずれか。 • このアクティビティーがサービス・クラス有効範囲の作業クラスに分類されている場合、値は作業クラスが属する作業クラス・セットに関連付けられた作業アクション・セットの ID です。 • このアクティビティーがサービス・クラス有効範囲の作業クラスに分類されていない場合、値は NULL です。
SERVICE_CLASS_WORK_CLASS_ID	INTEGER	以下の値のいずれか。 • このアクティビティーがサービス・クラス有効範囲の作業クラスに分類されている場合、この値はこのアクティビティーに割り当てられた作業クラスの ID です。 • このアクティビティーがサービス・クラス有効範囲の作業クラスに分類されていない場合、値は NULL です。

WLM_GET_WORKLOAD_STATS - ワークロード統計を戻す

注: この表関数は使用すべきではなく、 1040 ページの『WLM_GET_WORKLOAD_STATS_V97 表関数 - ワークロード統計を戻す』 に置き換えられました。

この関数は、ワークロード名とデータベース・パーティション番号のすべての組み合わせについてのワークロード統計を戻します。

構文



スキーマは SYSPROC です。

表関数パラメーター

workload_name
統計が戻される特定のワークロードを指定する、タイプ VARCHAR(128) の入力引数。引数が NULL または空ストリングである場合、すべてのワークロードについて統計が戻されます。

dbpartitionnum
この関数を呼び出すときに現在接続されているデータベースと同じインスタンス内の有効なパーティション番号を指定する、タイプ INTEGER の入力引数。現行のデータベース・パーティションには -1、すべてのデータベース・パーティションには -2 を指定します。 NULL 値を指定すると、-1 が暗黙的に設定されます。

許可

WLM_GET_WORKLOAD_STATS 関数に対する EXECUTE 特権。

例

管理者がワークロードの統計を調べるとします。これは以下の照会を使用して行うことができます。

```
SELECT SUBSTR(WORKLOAD_NAME,1,22) AS WL_DEF_NAME,
     SUBSTR(CHAR(DBPARTITIONNUM),1,4) AS PART,
     CONCURRENT_WLO_TOP AS WLO_TOP,
     CONCURRENT_WLO_ACT_TOP AS WLO_ACT_TOP
FROM TABLE(WLM_GET_WORKLOAD_STATS(CAST(NULL AS VARCHAR(128)), -2))
  AS WLSTATS
ORDER BY WL_DEF_NAME, PART
```

以下はこの照会の出力例です。

```
WL_DEF_NAME            PART WLO_TOP         WLO_ACT_TOP
---------------------- ---- --------------- -------------------
MYUSERWORKLOAD         0                 2                   8
MYUSERWORKLOAD         1                 0                   0
SYSDEFAULTUSERWORKLOAD 0                 1                   1
SYSDEFAULTUSERWORKLOAD 1                 0                   0
```

ここで、パーティション 0 では、MYUSERWORKLOAD ワークロードの並行オカレンスの最高数が 2 であり、これらのワークロード・オカレンスのいずれかの並行アクティビティーの最高数が 8 であることがわかります。

使用上の注意

この関数は、ワークロード名とデータベース・パーティション番号のすべての組み合わせについて 1 行を戻します。ワークロードの間、パーティションの間、またはサービス・クラスの間の集約は実行されません。しかし、集約は SQL 照会を使用して実行できます。

戻される情報

表 337. WLM_GET_WORKLOAD_STATS によって戻される情報

列名	データ・タイプ	説明
WORKLOAD_NAME	VARCHAR(128)	このレコードが収集されたワークロードの名前。
DBPARTITIONNUM	SMALLINT	このレコードが収集されたパーティション番号。

表 337. WLM_GET_WORKLOAD_STATS によって戻される情報 (続き)

列名	データ・タイプ	説明
LAST_RESET	TIMESTAMP	統計が最後にリセットされた時刻。以下の 4 つのイベントが発生する可能性があります。これらは、統計のリセットを起動し、このタイム・スタンプを更新します。 • WLM_COLLECT_STATS プロシージャーが呼び出される。 • WLM_COLLECT_INT 構成パラメーターによって制御された定期的なコレクションおよびリセット・プロセスにより、コレクションおよびリセットが発生する。 • データベースが再活動化される。 • 統計が報告されているワークロードが変更され、その変更がコミットされた。 LAST_RESET タイム・スタンプがローカル時刻である。
CONCURRENT_WLO_TOP	INTEGER	最後のリセット以降、このパーティション上の指定されたワークロードの並行オカレンスの最高数。
CONCURRENT_WLO_ACT_TOP	INTEGER	最後のリセット以降、このワークロードのいずれかのオカレンスで到達した、実行状態 (アイドルおよび待機中を含む) またはキューに入れられた状態の並行アクティビティー (コーディネーターとネストを含む) の最高数。各ワークロード・オカレンスによってその作業単位の終わりに更新されます。
COORD_ACT_COMPLETED_TOTAL	BIGINT	最後のリセット以降に完了したこのワークロードのいずれかのオカレンスに割り当てられた、任意のネスト・レベルのコーディネーター・アクティビティーの合計数。各ワークロード・オカレンスによってその作業単位の終わりに更新されます。
COORD_ACT_ABORTED_TOTAL	BIGINT	最後のリセット以降で完了前にアボートされたこのワークロードのいずれかのオカレンスに割り当てられた、任意のネスト・レベルのコーディネーター・アクティビティーの合計数。各ワークロード・オカレンスによってその作業単位の終わりに更新されます。

表 337. WLM_GET_WORKLOAD_STATS によって戻される情報 (続き)

列名	データ・タイプ	説明
COORD_ACT_REJECTED_TOTAL	BIGINT	最後のリセット以降で実行前にリジェクトされたこのワークロードのいずれかのオカレンスに割り当てられた、任意のネスト・レベルのコーディネーター・アクティビティーの合計数。各ワークロード・オカレンスによってその作業単位の終わりに更新されます。アクティビティーが実行抑制作業アクションまたは予測しきい値のいずれかによって実行を妨げられている場合、そのアクティビティーはリジェクトとしてカウントされます。 WLM_GET_SERVICE_SUBCLASS_STATS 関数の同じ名前の列とは異なり、これはアクティビティーがサービス・クラスに割り当てられる前に発生するリジェクトもカウントします。アクティビティーが ConcurrentWorkloadOccurrences しきい値に違反すると、そうしたリジェクトの例が発生します。
WLO_COMPLETED_TOTAL	BIGINT	最後のリセット以降、完了するワークロード・オカレンスの数。

付録 A. DB2 技術情報の概説

DB2 技術情報は、以下のツールと方法を介して利用できます。

- DB2 インフォメーション・センター
  - トピック (タスク、概念、およびリファレンス・トピック)
  - DB2 ツールのヘルプ
  - サンプル・プログラム
  - チュートリアル
- DB2 資料
  - PDF ファイル (ダウンロード可能)
  - PDF ファイル (DB2 PDF DVD に含まれる)
  - 印刷資料
- コマンド行ヘルプ
  - コマンド・ヘルプ
  - メッセージ・ヘルプ

注: DB2 インフォメーション・センターのトピックは、PDF やハードコピー資料よりも頻繁に更新されます。最新の情報を入手するには、資料の更新が発行されたときにそれをインストールするか、ibm.com にある DB2 インフォメーション・センターを参照してください。

技術資料、ホワイト・ペーパー、IBM Redbooks® 資料などのその他の DB2 技術情報には、オンライン (ibm.com) でアクセスできます。 DB2 Information Management ソフトウェア・ライブラリー・サイト (http://www.ibm.com/software/data/sw-library/) にアクセスしてください。

資料についてのフィードバック

DB2 の資料についてのお客様からの貴重なご意見をお待ちしています。DB2 の資料を改善するための提案については、db2docs@ca.ibm.com まで E メールを送信してください。DB2 の資料チームは、お客様からのフィードバックすべてに目を通しますが、直接お客様に返答することはありません。お客様が関心をお持ちの内容について、可能な限り具体的な例を提供してください。特定のトピックまたはヘルプ・ファイルについてのフィードバックを提供する場合は、そのトピック・タイトルおよび URL を含めてください。

DB2 お客様サポートに連絡する場合には、この E メール・アドレスを使用しないでください。資料を参照しても、DB2 の技術的な問題が解決しない場合は、お近くの IBM サービス・センターにお問い合わせください。

© Copyright IBM Corp. 2006, 2010

DB2 テクニカル・ライブラリー (ハードコピーまたは PDF 形式)

以下の表は、IBM Publications Center (www.ibm.com/e-business/linkweb/publications/servlet/pbi.wss) から利用できる DB2 ライブラリーについて説明しています。英語および翻訳された DB2 バージョン 9.7 のマニュアル (PDF 形式) は、www.ibm.com/support/docview.wss?rs=71&uid=swg2700947 からダウンロードできます。

この表には印刷資料が入手可能かどうかが示されていますが、国または地域によっては入手できない場合があります。

資料番号は、資料が更新される度に大きくなります。資料を参照する際は、以下にリストされている最新版であることを確認してください。

注: *DB2 インフォメーション・センター*は、PDF やハードコピー資料よりも頻繁に更新されます。

表 338. DB2 の技術情報

資料名	資料番号	印刷資料が入手可能かどうか	最終更新
管理 API リファレンス	SC88-5883-02	入手可能	2010 年 9 月
管理ルーチンおよびビュー	SC88-5880-02	入手不可	2010 年 9 月
コール・レベル・インターフェース ガイドおよびリファレンス 第 1 巻	SC88-5885-02	入手可能	2010 年 9 月
コール・レベル・インターフェース ガイドおよびリファレンス 第 2 巻	SC88-5886-02	入手可能	2010 年 9 月
コマンド・リファレンス	SC88-5884-02	入手可能	2010 年 9 月
データ移動ユーティリティー ガイドおよびリファレンス	SC88-5903-00	入手可能	2009 年 8 月
データ・リカバリーと高可用性 ガイドおよびリファレンス	SC88-5904-02	入手可能	2010 年 9 月
データベース: 管理の概念および構成リファレンス	SC88-5870-02	入手可能	2010 年 9 月
データベースのモニタリング ガイドおよびリファレンス	SC88-5872-02	入手可能	2010 年 9 月
データベース・セキュリティー・ガイド	SC88-5905-01	入手可能	2009 年 11 月

表 338. DB2 の技術情報 (続き)

資料名	資料番号	印刷資料が入手可能かどうか	最終更新
DB2 Text Search ガイド	SC88-5902-02	入手可能	2010 年 9 月
ADO.NET および OLE DB アプリケーションの開発	SC88-5874-01	入手可能	2009 年 11 月
組み込み SQL アプリケーションの開発	SC88-5875-01	入手可能	2009 年 11 月
Java アプリケーションの開発	SC88-5878-02	入手可能	2010 年 9 月
Perl、PHP、Python および Ruby on Rails アプリケーションの開発	SC88-5879-01	入手不可	2010 年 9 月
SQL および外部ルーチンの開発	SC88-5876-01	入手可能	2009 年 11 月
データベース・アプリケーション開発の基礎	GI88-4201-01	入手可能	2009 年 11 月
DB2 インストールおよび管理 概説 (Linux および Windows 版)	GI88-4202-00	入手可能	2009 年 8 月
グローバリゼーション・ガイド	SC88-5906-00	入手可能	2009 年 8 月
DB2 サーバー機能 インストール	GC88-5888-02	入手可能	2010 年 9 月
IBM データ・サーバー・クライアント機能 インストール	GC88-5889-01	入手不可	2010 年 9 月
メッセージ・リファレンス 第 1 巻	SC88-5897-00	入手不可	2009 年 8 月
メッセージ・リファレンス 第 2 巻	SC88-5898-00	入手不可	2009 年 8 月
Net Search Extender 管理およびユーザーズ・ガイド	SC88-5901-02	入手不可	2010 年 9 月
パーティションおよびクラスタリングのガイド	SC88-5907-01	入手可能	2009 年 11 月
pureXML ガイド	SC88-5895-01	入手可能	2009 年 11 月
Query Patroller 管理およびユーザーズ・ガイド	SC88-5908-00	入手不可	2009 年 8 月

表 338. DB2 の技術情報 (続き)

資料名	資料番号	印刷資料が入手可能かどうか	最終更新
Spatial Extender および Geodetic Data Management Feature ユーザーズ・ガイドおよびリファレンス	SC88-5900-01	入手不可	2010 年 9 月
SQL プロシージャー言語: アプリケーションのイネーブルメントおよびサポート	SC88-5877-02	入手可能	2010 年 9 月
SQL リファレンス 第 1 巻	SC88-5881-02	入手可能	2010 年 9 月
SQL リファレンス 第 2 巻	SC88-5882-02	入手可能	2010 年 9 月
問題判別およびデータベース・パフォーマンスのチューニング	SC88-5871-02	入手可能	2010 年 9 月
DB2 バージョン 9.7 へのアップグレード	SC88-5887-02	入手可能	2010 年 9 月
Visual Explain チュートリアル	SC88-5899-00	入手不可	2009 年 8 月
DB2 バージョン 9.7 の新機能	SC88-5893-02	入手可能	2010 年 9 月
ワークロード・マネージャー ガイドおよびリファレンス	SC88-5894-02	入手可能	2010 年 9 月
XQuery リファレンス	SC88-5896-01	入手不可	2009 年 11 月

表 339. DB2 Connect 固有の技術情報

資料名	資料番号	印刷資料が入手可能かどうか	最終更新
DB2 Connect Personal Edition インストールおよび構成	SC88-5891-02	入手可能	2010 年 9 月
DB2 Connect サーバー機能 インストールおよび構成	SC88-5892-02	入手可能	2010 年 9 月
DB2 Connect ユーザーズ・ガイド	SC88-5890-02	入手可能	2010 年 9 月

表 340. Information Integration の技術情報

資料名	資料番号	印刷資料が入手可能かどうか	最終更新
Information Integration: フェデレーテッド・システム管理ガイド	SC88-4166-02	入手可能	2009 年 8 月
Information Integration: レプリケーションおよびイベント・パブリッシングのための ASNCLP プログラム・リファレンス	SC88-4167-04	入手可能	2009 年 8 月
Information Integration: フェデレーテッド・データ・ソース構成ガイド	SC88-4185-02	入手不可	2009 年 8 月
Information Integration: SQL レプリケーション ガイドとリファレンス	SC88-4168-02	入手可能	2009 年 8 月
Information Integration: レプリケーションとイベント・パブリッシング 概説	GC88-4187-02	入手可能	2009 年 8 月

DB2 の印刷資料の注文方法

DB2 の印刷資料が必要な場合、オンラインで購入することができますが、すべての国および地域で購入できるわけではありません。 DB2 の印刷資料については、IBM 営業担当員にお問い合わせください。 *DB2 PDF ドキュメンテーション* DVD の一部のソフトコピー・ブックは、印刷資料では入手できないことに留意してください。例えば、「*DB2 メッセージ・リファレンス*」はどちらの巻も印刷資料としては入手できません。

DB2 PDF ドキュメンテーション DVD で利用できる DB2 の印刷資料の大半は、IBM に有償で注文することができます。国または地域によっては、資料を IBM Publications Center からオンラインで注文することもできます。お客様の国または地域でオンライン注文が利用できない場合、DB2 の印刷資料については、IBM 営業担当員にお問い合わせください。 DB2 PDF ドキュメンテーション DVD に収録されている資料の中には、印刷資料として提供されていないものもあります。

注: 最新で完全な DB2 資料は、DB2 インフォメーション・センター (http://publib.boulder.ibm.com/infocenter/db2luw/v9r7) で参照することができます。

DB2 の印刷資料は以下の方法で注文することができます。

- 日本 IBM 発行のマニュアルはインターネット経由でご購入いただけます。詳しくは http://www.ibm.com/shop/publications/order をご覧ください。資料の注文情報にアクセスするには、お客様の国、地域、または言語を選択してください。その後、各ロケーションにおける注文についての指示に従ってください。

- DB2 の印刷資料を IBM 営業担当員に注文するには、以下のようにします。
  1. 以下の Web サイトのいずれかから、営業担当員の連絡先情報を見つけてください。
     - IBM Directory of world wide contacts (www.ibm.com/planetwide)
     - IBM Publications Web サイト (http://www.ibm.com/shop/publications/order)。国、地域、または言語を選択し、お客様の所在地に該当する Publications ホーム・ページにアクセスしてください。このページから、「このサイトについて」のリンクにアクセスしてください。
  2. 電話をご利用の場合は、DB2 資料の注文であることをご指定ください。
  3. 担当者に、注文する資料のタイトルと資料番号をお伝えください。タイトルと資料番号は、1352 ページの『DB2 テクニカル・ライブラリー (ハードコピーまたは PDF 形式)』でご確認いただけます。

コマンド行プロセッサーから SQL 状態ヘルプを表示する

DB2 製品は、SQL ステートメントの結果の原因になったと考えられる条件の SQLSTATE 値を戻します。 SQLSTATE ヘルプは、SQL 状態および SQL 状態クラス・コードの意味を説明します。

SQL 状態ヘルプを開始するには、コマンド行プロセッサーを開いて以下のように入力します。

```
? sqlstate or ? class code
```

ここで、*sqlstate* は有効な 5 桁の SQL 状態を、*class code* は SQL 状態の最初の 2 桁を表します。
例えば、`? 08003` を指定すると SQL 状態 08003 のヘルプが表示され、 `? 08` を指定するとクラス・コード 08 のヘルプが表示されます。

異なるバージョンの DB2 インフォメーション・センターへのアクセス

DB2 バージョン 9.8 のトピックを扱っている *DB2* インフォメーション・センターの URL は、http://publib.boulder.ibm.com/infocenter/db2luw/v9r8/ です。

DB2 バージョン 9.7 のトピックを扱っている *DB2* インフォメーション・センターの URL は、http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/ です。

DB2 バージョン 9.5 のトピックを扱っている *DB2* インフォメーション・センターの URL は、http://publib.boulder.ibm.com/infocenter/db2luw/v9r5 です。

DB2 バージョン 9.1 のトピックを扱っている *DB2* インフォメーション・センターの URL は、http://publib.boulder.ibm.com/infocenter/db2luw/v9/ です。

DB2 バージョン 8 のトピックについては、*DB2* インフォメーション・センター の URL http://publib.boulder.ibm.com/infocenter/db2luw/v8/ にアクセスしてください。

DB2 インフォメーション・センターでの希望する言語でのトピックの表示

DB2 インフォメーション・センターでは、ブラウザーの設定で指定した言語でのトピックの表示が試みられます。トピックがその指定言語に翻訳されていない場合は、 DB2 インフォメーション・センターでは英語でトピックが表示されます。

- Internet Explorer Web ブラウザーで、指定どおりの言語でトピックを表示するには、以下のようにします。
  1. Internet Explorer の「**ツール**」 -> 「**インターネット オプション**」 -> 「**言語...**」ボタンをクリックします。「言語の優先順位」ウィンドウがオープンします。
  2. 該当する言語が、言語リストの先頭の項目に指定されていることを確認します。
     - リストに新しい言語を追加するには、「**追加...**」ボタンをクリックします。

       注: 言語を追加しても、特定の言語でトピックを表示するのに必要なフォントがコンピューターに備えられているとはかぎりません。
     - リストの先頭に新しい言語を移動するには、その言語を選択してから、その言語が言語リストに先頭に行くまで「**上に移動**」ボタンをクリックします。
  3. ページを最新表示します。希望する言語で DB2 インフォメーション・センターが表示されます。
- Firefox または Mozilla Web ブラウザーの場合に、希望する言語でトピックを表示するには、以下のようにします。
  1. 「**ツール**」 -> 「**オプション**」 -> 「**詳細**」 ダイアログの「**言語**」セクションにあるボタンを選択します。「設定」ウィンドウに「言語」パネルが表示されます。
  2. 該当する言語が、言語リストの先頭の項目に指定されていることを確認します。
     - リストに新しい言語を追加するには、「**追加...**」ボタンをクリックしてから、「言語を追加」ウィンドウで言語を選択します。
     - リストの先頭に新しい言語を移動するには、その言語を選択してから、その言語が言語リストに先頭に行くまで「**上に移動**」ボタンをクリックします。
  3. ページを最新表示します。希望する言語で DB2 インフォメーション・センターが表示されます。

ブラウザーとオペレーティング・システムの組み合わせによっては、オペレーティング・システムの地域の設定も希望のロケールと言語に変更しなければなりません。

コンピューターまたはイントラネット・サーバーにインストールされた DB2 インフォメーション・センターの更新

ローカルにインストールされた DB2 インフォメーション・センターは、定期的に更新する必要があります。

DB2 バージョン 9.7 インフォメーション・センターが既にインストールされている必要があります。詳しくは、「*DB2 サーバー機能 インストール*」の『DB2 セットアップ・ウィザードによる DB2 インフォメーション・センターのインストール』のトピックを参照してください。インフォメーション・センターのインストールに適用されるすべての前提条件と制約事項は、インフォメーション・センターの更新にも適用されます。

既存の DB2 インフォメーション・センターは、自動で更新することも。手動で更新することもできます。

- 自動更新 - 既存のインフォメーション・センターのフィーチャーと言語を更新します。自動更新を使用すると、更新中にインフォメーション・センターが使用できなくなる時間が最小限で済むというメリットもあります。さらに、自動更新は、定期的に実行する他のバッチ・ジョブの一部として実行されるように設定することができます。
- 手動更新 - 更新処理中にフィーチャーまたは言語を追加する場合に使用する必要があります。例えば、ローカルのインフォメーション・センターが最初は英語とフランス語でインストールされており、その後ドイツ語もインストールすることにした場合、手動更新でドイツ語をインストールし、同時に、既存のインフォメーション・センターのフィーチャーおよび言語を更新できます。しかし、手動更新ではインフォメーション・センターを手動で停止、更新、再始動する必要があります。更新処理の間はずっと、インフォメーション・センターは使用できなくなります。

このトピックでは、自動更新のプロセスを詳しく説明しています。手動更新の手順については、『コンピューターまたはイントラネット・サーバーにインストールされた DB2 インフォメーション・センターの手動更新』のトピックを参照してください。

コンピューターまたはイントラネット・サーバーにインストールされている DB2 インフォメーション・センターを自動で更新するには、次のようにします。

1. Linux オペレーティング・システムの場合、次のようにします。
   a. インフォメーション・センターがインストールされているパスにナビゲートします。デフォルトでは、DB2 インフォメーション・センターは、`/opt/ibm/db2ic/V9.7` ディレクトリーにインストールされています。
   b. インストール・ディレクトリーから `doc/bin` ディレクトリーにナビゲートします。
   c. 次のように `ic-update` スクリプトを実行します。

      ```
      ic-update
      ```
2. Windows オペレーティング・システムの場合、次のようにします。
   a. コマンド・ウィンドウを開きます。
   b. インフォメーション・センターがインストールされているパスにナビゲートします。デフォルトでは、DB2 インフォメーション・センターは、`<Program Files>¥IBM¥DB2 Information Center¥Version 9.7` ディレクトリーにインストールされています (`<Program Files>` は「Program Files」ディレクトリーのロケーション)。
   c. インストール・ディレクトリーから `doc¥bin` ディレクトリーにナビゲートします。

d. 次のように ic-update.bat ファイルを実行します。

```
ic-update.bat
```

DB2 インフォメーション・センターが自動的に再始動します。更新が入手可能な場合、インフォメーション・センターに、更新された新しいトピックが表示されます。インフォメーション・センターの更新が入手可能でなかった場合、メッセージがログに追加されます。ログ・ファイルは、doc¥eclipse¥configuration ディレクトリーにあります。ログ・ファイル名はランダムに生成された名前です。例えば、1239053440785.log のようになります。

コンピューターまたはイントラネット・サーバーにインストールされた DB2 インフォメーション・センターの手動更新

DB2 インフォメーション・センターをローカルにインストールしている場合は、IBM から資料の更新を入手してインストールすることができます。

ローカルにインストールされた *DB2 インフォメーション・センター* を手動で更新するには、以下のことを行う必要があります。

1. コンピューター上の *DB2 インフォメーション・センター* を停止し、インフォメーション・センターをスタンドアロン・モードで再始動します。インフォメーション・センターをスタンドアロン・モードで実行すると、ネットワーク上の他のユーザーがそのインフォメーション・センターにアクセスできなくなります。これで、更新を適用できるようになります。DB2 インフォメーション・センターのワークステーション・バージョンは、常にスタンドアロン・モードで実行されます。
2. 「更新」機能を使用することにより、どんな更新が利用できるかを確認します。インストールしなければならない更新がある場合は、「更新」機能を使用してそれを入手およびインストールできます。

   注: ご使用の環境において、インターネットに接続されていないマシンに *DB2 インフォメーション・センター* の更新をインストールする必要がある場合、インターネットに接続されていて *DB2 インフォメーション・センター* がインストールされているマシンを使用して、更新サイトをローカル・ファイル・システムにミラーリングしてください。ネットワーク上の多数のユーザーが資料の更新をインストールする場合にも、更新サイトをローカルにミラーリングして、更新サイト用のプロキシーを作成することにより、個々のユーザーが更新を実行するのに要する時間を短縮できます。
   更新パッケージが入手可能な場合、「更新」機能を使用してパッケージを入手します。ただし、「更新」機能は、スタンドアロン・モードでのみ使用できます。
3. スタンドアロンのインフォメーション・センターを停止し、コンピューター上の *DB2 インフォメーション・センター* を再開します。

注: Windows 2008、Windows Vista (およびそれ以上) では、このセクションの後の部分でリストされているコマンドは管理者として実行する必要があります。完全な管理者特権でコマンド・プロンプトまたはグラフィカル・ツールを開くには、ショートカットを右クリックしてから、**「管理者として実行」**を選択します。

コンピューターまたはイントラネット・サーバーにインストール済みの *DB2* インフォメーション・センター を更新するには、以下のようにします。

1. *DB2* インフォメーション・センター を停止します。
   - Windows では、「**スタート**」 → 「**コントロール パネル**」 → 「**管理ツール**」 → 「**サービス**」をクリックします。次に、「**DB2 インフォメーション・センター**」サービスを右クリックして「**停止**」を選択します。
   - Linux では、以下のコマンドを入力します。

     ```
     /etc/init.d/db2icdv97 stop
     ```
2. インフォメーション・センターをスタンドアロン・モードで開始します。
   - Windows の場合:
     a. コマンド・ウィンドウを開きます。
     b. インフォメーション・センターがインストールされているパスにナビゲートします。デフォルトでは、*DB2* インフォメーション・センター は、*Program_Files*¥IBM¥DB2 Information Center¥Version 9.7 ディレクトリーにインストールされています (*Program_Files* は Program Filesディレクトリーのロケーション)。
     c. インストール・ディレクトリーから doc¥bin ディレクトリーにナビゲートします。
     d. 次のように help_start.bat ファイルを実行します。

        ```
        help_start.bat
        ```
   - Linux の場合:
     a. インフォメーション・センターがインストールされているパスにナビゲートします。デフォルトでは、*DB2* インフォメーション・センター は、/opt/ibm/db2ic/V9.7 ディレクトリーにインストールされています。
     b. インストール・ディレクトリーから doc/bin ディレクトリーにナビゲートします。
     c. 次のように help_start スクリプトを実行します。

        ```
        help_start
        ```

   システムのデフォルト Web ブラウザーが開き、スタンドアロンのインフォメーション・センターが表示されます。
3. 「**更新**」ボタン () をクリックします。(ブラウザーで JavaScript™ が有効になっている必要があります。) インフォメーション・センターの右側のパネルで、「**更新の検索 (Find Updates)**」をクリックします。 既存の文書に対する更新のリストが表示されます。
4. インストール・プロセスを開始するには、インストールする更新をチェックして選択し、「**更新のインストール**」をクリックします。
5. インストール・プロセスが完了したら、「**完了**」をクリックします。
6. 次のようにして、スタンドアロンのインフォメーション・センターを停止します。
   - Windows の場合は、インストール・ディレクトリーの doc¥bin ディレクトリーにナビゲートしてから、次のように help_end.bat ファイルを実行します。

     ```
     help_end.bat
     ```

注: help_end バッチ・ファイルには、help_start バッチ・ファイルを使用して開始したプロセスを安全に停止するのに必要なコマンドが含まれています。help_start.bat は、Ctrl-C や他の方法を使用して停止しないでください。

- Linux の場合は、インストール・ディレクトリーの doc/bin ディレクトリーにナビゲートしてから、次のように help_end スクリプトを実行します。

  ```
  help_end
  ```

  注: help_end スクリプトには、help_start スクリプトを使用して開始したプロセスを安全に停止するのに必要なコマンドが含まれています。他の方法を使用して、help_start スクリプトを停止しないでください。

7. *DB2 インフォメーション・センター* を再開します。
   - Windows では、「**スタート**」 → 「**コントロール パネル**」 → 「**管理ツール**」 → 「**サービス**」をクリックします。次に、「**DB2 インフォメーション・センター**」サービスを右クリックして「**開始**」を選択します。
   - Linux では、以下のコマンドを入力します。

     ```
     /etc/init.d/db2icdv97 start
     ```

更新された *DB2 インフォメーション・センター* に、更新された新しいトピックが表示されます。

DB2 チュートリアル

DB2 チュートリアルは、DB2 製品のさまざまな機能について学習するのを支援します。この演習をとおして段階的に学習することができます。

はじめに

インフォメーション・センター (http://publib.boulder.ibm.com/infocenter/db2help/) から、このチュートリアルの XHTML 版を表示できます。

演習の中で、サンプル・データまたはサンプル・コードを使用する場合があります。個々のタスクの前提条件については、チュートリアルを参照してください。

DB2 チュートリアル

チュートリアルを表示するには、タイトルをクリックします。

「*pureXML ガイド*」の『**pureXML**』
: XML データを保管し、ネイティブ XML データ・ストアに対して基本的な操作を実行できるように、DB2 データベースをセットアップします。

「*Visual Explain チュートリアル*」の『**Visual Explain**』
: Visual Explain を使用して、パフォーマンスを向上させるために SQL ステートメントを分析し、最適化し、調整します。

DB2 トラブルシューティング情報

DB2 データベース製品を使用する際に役立つ、トラブルシューティングおよび問題判別に関する広範囲な情報を利用できます。

DB2 ドキュメンテーション

トラブルシューティング情報は、「*問題判別およびデータベース・パフォーマンスのチューニング*」または *DB2 インフォメーション・センター* の『データベースの基本』セクションにあります。ここには、DB2 診断ツールおよびユーティリティーを使用して、問題を切り分けて識別する方法、最も頻繁に起こる幾つかの問題に対するソリューションについての情報、および DB2 データベース製品を使用する際に発生する可能性のある問題の解決方法についての他のアドバイスがあります。

DB2 Technical Support の Web サイト

現在問題が発生していて、考えられる原因とソリューションを検索したい場合は、DB2 Technical Support の Web サイトを参照してください。Technical Support サイトには、最新の DB2 資料、TechNotes、プログラム診断依頼書 (APAR またはバグ修正)、フィックスパック、およびその他のリソースへのリンクが用意されています。この知識ベースを活用して、問題に対する有効なソリューションを探し出すことができます。

DB2 Technical Support の Web サイト (http://www.ibm.com/software/data/db2/support/db2_9/) にアクセスしてください。

ご利用条件

これらの資料は、以下の条件に同意していただける場合に限りご使用いただけます。

個人使用: これらの資料は、すべての著作権表示その他の所有権表示をしていただくことを条件に、非商業的な個人による使用目的に限り複製することができます。ただし、IBM の明示的な承諾をえずに、これらの資料またはその一部について、二次的著作物を作成したり、配布 (頒布、送信を含む) または表示 (上映を含む) することはできません。

商業的使用: これらの資料は、すべての著作権表示その他の所有権表示をしていただくことを条件に、お客様の企業内に限り、複製、配布、および表示することができます。 ただし、IBM の明示的な承諾をえずにこれらの資料の二次的著作物を作成したり、お客様の企業外で資料またはその一部を複製、配布、または表示することはできません。

ここで明示的に許可されているもの以外に、資料や資料内に含まれる情報、データ、ソフトウェア、またはその他の知的所有権に対するいかなる許可、ライセンス、または権利を明示的にも黙示的にも付与するものではありません。

資料の使用が IBM の利益を損なうと判断された場合や、上記の条件が適切に守られていないと判断された場合、IBM はいつでも自らの判断により、ここで与えた許可を撤回できるものとさせていただきます。

お客様がこの情報をダウンロード、輸出、または再輸出する際には、米国のすべての輸出入関連法規を含む、すべての関連法規を遵守するものとします。

IBM は、これらの資料の内容についていかなる保証もしません。これらの資料は、特定物として現存するままの状態で提供され、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含むすべての明示もしくは黙示の保証責任なしで提供されます。

付録 B. 特記事項

本書は米国 IBM が提供する製品およびサービスについて作成したものです。IBM 以外の製品に関する情報は、本書の最初の発行時点で入手可能な情報に基づいており、変更される場合があります。

本書に記載の製品、サービス、または機能が日本においては提供されていない場合があります。日本で利用可能な製品、サービス、および機能については、日本 IBM の営業担当員にお尋ねください。本書で IBM 製品、プログラム、またはサービスに言及していても、その IBM 製品、プログラム、またはサービスのみが使用可能であることを意味するものではありません。これらに代えて、IBM の知的所有権を侵害することのない、機能的に同等の製品、プログラム、またはサービスを使用することができます。ただし、IBM 以外の製品とプログラムの操作またはサービスの評価および検証は、お客様の責任で行っていただきます。

IBM は、本書に記載されている内容に関して特許権 (特許出願中のものを含む) を保有している場合があります。本書の提供は、お客様にこれらの特許権について実施権を許諾することを意味するものではありません。実施権についてのお問い合わせは、書面にて下記宛先にお送りください。

〒242-8502
神奈川県大和市下鶴間1623番14号
日本アイ・ビー・エム株式会社
法務・知的財産
知的財産権ライセンス渉外

以下の保証は、国または地域の法律に沿わない場合は、適用されません。 IBM およびその直接または間接の子会社は、本書を特定物として現存するままの状態で提供し、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含むすべての明示もしくは黙示の保証責任を負わないものとします。国または地域によっては、法律の強行規定により、保証責任の制限が禁じられる場合、強行規定の制限を受けるものとします。

この情報には、技術的に不適切な記述や誤植を含む場合があります。本書は定期的に見直され、必要な変更は本書の次版に組み込まれます。 IBM は予告なしに、随時、この文書に記載されている製品またはプログラムに対して、改良または変更を行うことがあります。

本書において IBM 以外の Web サイトに言及している場合がありますが、便宜のため記載しただけであり、決してそれらの Web サイトを推奨するものではありません。それらの Web サイトにある資料は、この IBM 製品の資料の一部ではありません。それらの Web サイトは、お客様の責任でご使用ください。

IBM は、お客様が提供するいかなる情報も、お客様に対してなんら義務も負うことのない、自ら適切と信ずる方法で、使用もしくは配布することができるものとします。

© Copyright IBM Corp. 2006, 2010

本プログラムのライセンス保持者で、(i) 独自に作成したプログラムとその他のプログラム (本プログラムを含む) との間での情報交換、および (ii) 交換された情報の相互利用を可能にすることを目的として、本プログラムに関する情報を必要とする方は、下記に連絡してください。

IBM Canada Limited
U59/3600
3600 Steeles Avenue East
Markham, Ontario L3R 9Z7
CANADA

本プログラムに関する上記の情報は、適切な使用条件の下で使用することができますが、有償の場合もあります。

本書で説明されているライセンス・プログラムまたはその他のライセンス資料は、IBM 所定のプログラム契約の契約条項、IBM プログラムのご使用条件、またはそれと同等の条項に基づいて、IBM より提供されます。

この文書に含まれるいかなるパフォーマンス・データも、管理環境下で決定されたものです。そのため、他の操作環境で得られた結果は、異なる可能性があります。一部の測定が、開発レベルのシステムで行われた可能性がありますが、その測定値が、一般に利用可能なシステムのものと同じである保証はありません。さらに、一部の測定値が、推定値である可能性があります。実際の結果は、異なる可能性があります。お客様は、お客様の特定の環境に適したデータを確かめる必要があります。

IBM 以外の製品に関する情報は、その製品の供給者、出版物、もしくはその他の公に利用可能なソースから入手したものです。 IBM は、それらの製品のテストは行っておりません。したがって、他社製品に関する実行性、互換性、またはその他の要求については確証できません。 IBM 以外の製品の性能に関する質問は、それらの製品の供給者にお願いします。

IBM の将来の方向または意向に関する記述については、予告なしに変更または撤回される場合があり、単に目標を示しているものです。

本書には、日常の業務処理で用いられるデータや報告書の例が含まれています。より具体性を与えるために、それらの例には、個人、企業、ブランド、あるいは製品などの名前が含まれている場合があります。これらの名称はすべて架空のものであり、名称や住所が類似する企業が実在しているとしても、それは偶然にすぎません。

著作権使用許諾:

本書には、様々なオペレーティング・プラットフォームでのプログラミング手法を例示するサンプル・アプリケーション・プログラムがソース言語で掲載されています。お客様は、サンプル・プログラムが書かれているオペレーティング・プラットフォームのアプリケーション・プログラミング・インターフェースに準拠したアプリケーション・プログラムの開発、使用、販売、配布を目的として、いかなる形式においても、IBM に対価を支払うことなくこれを複製し、改変し、配布することができます。このサンプル・プログラムは、あらゆる条件下における完全なテストを経ていません。従って IBM は、これらのサンプル・プログラムについて信頼性、

利便性もしくは機能性があることをほのめかしたり、保証することはできません。サンプル・プログラムは、現存するままの状態で提供されるものであり、いかなる種類の保証も提供されません。 IBM は、これらのサンプル・プログラムの使用から生ずるいかなる損害に対しても責任を負いません。

それぞれの複製物、サンプル・プログラムのいかなる部分、またはすべての派生した創作物には、次のように、著作権表示を入れていただく必要があります。

© (*お客様の会社名*) (*西暦年*). このコードの一部は、IBM Corp. のサンプル・プログラムから取られています。 © Copyright IBM Corp. _*年を入れる*_. All rights reserved.

商標

IBM、IBM ロゴおよび ibm.com® は、世界の多くの国で登録された International Business Machines Corp. の商標です。他の製品名およびサービス名等は、それぞれ IBM または各社の商標である場合があります。現時点での IBM の商標リストについては、http://www.ibm.com/legal/copytrade.shtml をご覧ください。

以下は、それぞれ各社の商標または登録商標です。

- Linux は、Linus Torvalds の米国およびその他の国における商標です。
- Java およびすべての Java 関連の商標およびロゴは、Sun Microsystems, Inc. の米国およびその他の国における商標です。
- UNIX は The Open Group の米国およびその他の国における登録商標です。
- Intel®、Intel ロゴ、Intel Inside®、 Intel Inside ロゴ、Intel® Centrino®、Intel Centrino ロゴ、Celeron®、Intel® Xeon®、Intel SpeedStep®、Itanium®、Pentium® は、Intel Corporation または子会社の米国およびその他の国における商標または登録商標です。
- Microsoft®、 Windows、 Windows NT®、および Windows ロゴは、Microsoft Corporation の米国およびその他の国における商標です。

索引

日本語, 数字, 英字, 特殊文字の順に配列されています。なお, 濁音と半濁音は清音と同等に扱われています。

[ア行]

[カ行]

© Copyright IBM Corp. 2006, 2010

[サ行]

[タ行]

[ハ行]

［マ行］

［ラ行］

A

B

C

D

E

F

G

H

I

L

M

N

O

P

Q

R

S

T

U

W

X

Printed in Japan

SC88-5880-02

日本アイ・ビー・エム株式会社
〒103-8510 東京都中央区日本橋箱崎町19-21

Spine information:

IBM DB2 9.7 for Linux, UNIX, and Windows　バージョン 9 リリース 7

管理ルーチンおよびビュー