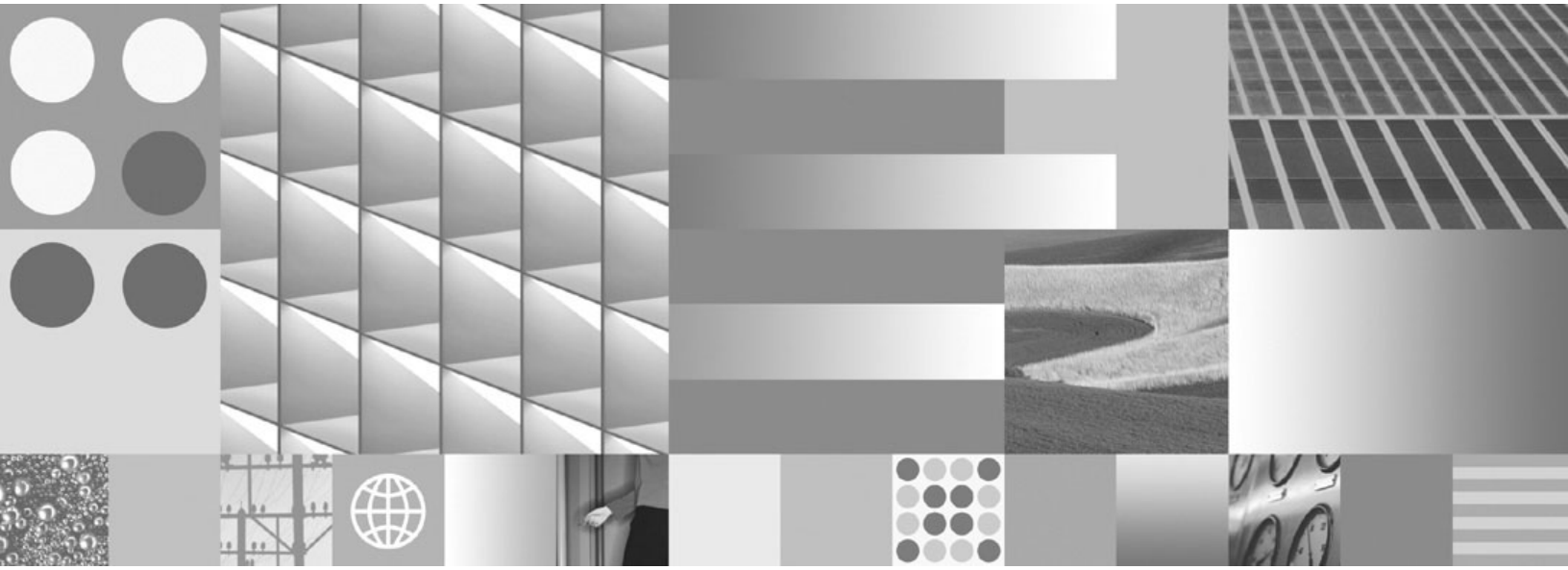
DB2 Version 9.5
for Linux, UNIX, and Windows

バージョン 9 リリース 5



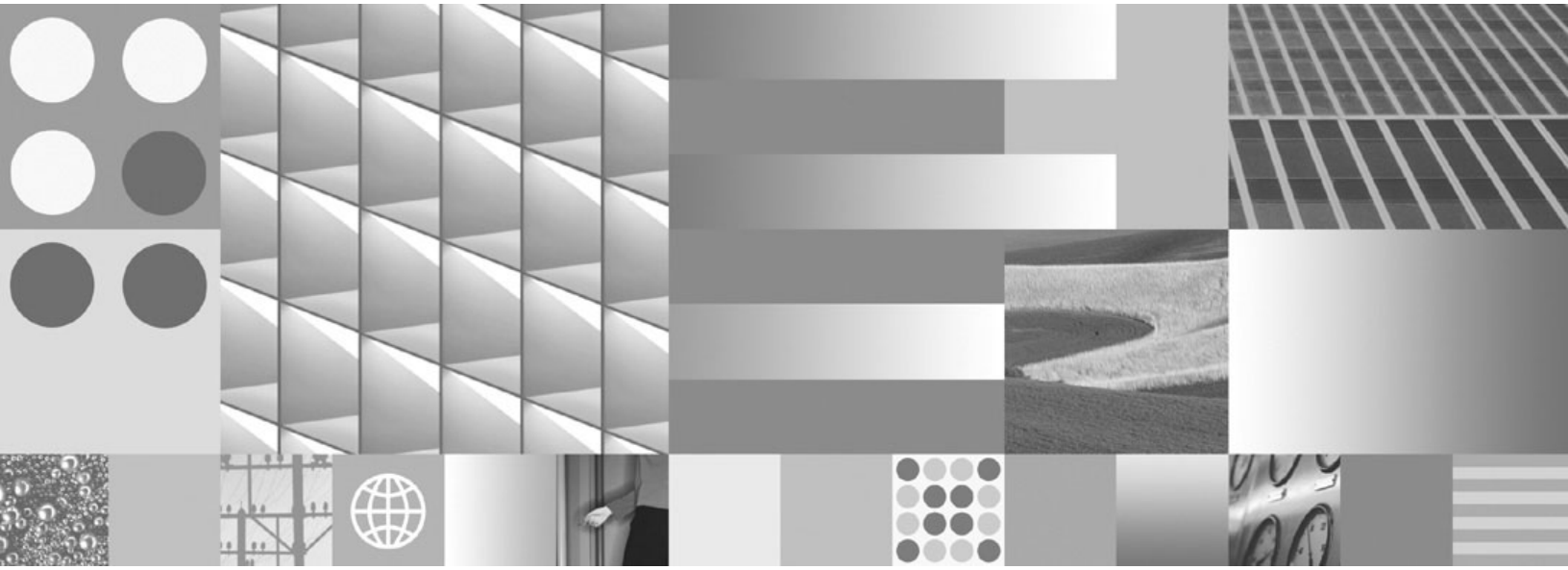
管理ルーチンおよびビュー

最終更新: 2010 年 12 月

SC88-4435-03
(英文原典：SC23-5843-03)

DB2 Version 9.5
for Linux, UNIX, and Windows

バージョン 9 リリース 5



管理ルーチンおよびビュー

最終更新: 2010 年 12 月

SC88-4435-03
(英文原典：SC23-5843-03)

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、 937 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： SC23-5843-03
DB2 Version 9.5
for Linux, UNIX, and Windows
Version 9 Release 5
Administrative Routines and Views
Updated December, 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 2010.12

© Copyright IBM Corporation 2006, 2010.

目次

第 1 部 管理 SQL ルーチンおよびビュー 1

第 1 章 管理ビューの許可 3

第 2 章 管理ビューと表関数との比較. . . 5

第 3 章 サポートされる管理 SQL ルーチンおよび管理ビュー 7

アクティビティー・モニター・ルーチン 21
AM_BASE_RPT_RECOMS - アクティビティー・レポートに関する推奨事項 21
AM_BASE_RPTS - アクティビティー・モニター・レポート 22
AM_DROP_TASK - モニター・タスクの削除 . . 24
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 . . . 24
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 25
AM_GET_LOCK_RPT - アプリケーション・ロックに関する詳細の検索 26
AM_GET_RPT - アクティビティー・モニター・データの検索 34
AM_SAVE_TASK - モニター・タスクの作成または変更 35
ADMIN_CMD プロシージャーと関連したルーチン 37
ADMIN_CMD - 管理コマンドの実行 37
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 223
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 225
ADMINTABCOMPRESSINFO ビューおよび ADMIN_GET_TAB_COMPRESS_INFO 226
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 231
ADMINTABINFO 管理ビューおよび ADMIN_GET_TAB_INFO_V95 表関数 - 表のサイズおよび状態に関する情報の検索 232
管理タスク・スケジューラー・ルーチンおよびビュー 239
ADMIN_TASK_ADD プロシージャー - 新規タスクのスケジュール 239
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 245
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 246
ADMIN_TASK_STATUS 管理ビュー - タスク状況情報の取得 248
ADMIN_TASK_UPDATE プロシージャー - 既存のタスクの更新. 251
監査ルーチンおよびプロシージャー 254
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 254
AUDIT_DELIM_EXTRACT - 区切り文字付きファイルへの抽出の実行 255
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 256
自動保守ルーチン 257
AUTOMAINT_GET_POLICY プロシージャー - 自動保守ポリシーの取得. 257
AUTOMAINT_GET_POLICYFILE プロシージャー - 自動保守ポリシーの取得 258
AUTOMAINT_SET_POLICY プロシージャー - 自動保守ポリシーの構成. 259
AUTOMAINT_SET_POLICYFILE プロシージャー - 自動保守ポリシーの構成 261
構成ルーチンおよびビュー 262
DB_PARTITIONS 262
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 263
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 265
REG_VARIABLES 管理ビュー - 使用中のDB2 レジストリー設定の検索. 267
環境ビュー 269
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. 269
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 270
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. 272
ENV_SYS_INFO 管理ビュー - システムに関する情報の検索 273
ENV_SYS_RESOURCES 管理ビュー - システム情報を戻す 274
ヘルス・スナップショット・ルーチン 277
HEALTH_CONT_HI 277
HEALTH_CONT_HI_HIS 279
HEALTH_CONT_INFO 281
HEALTH_DB_HI 283
HEALTH_DB_HI_HIS 286
HEALTH_DB_HIC 290
HEALTH_DB_HIC_HIS 292
HEALTH_DB_INFO 295
HEALTH_DBM_HI. 296
HEALTH_DBM_HI_HIS 297
HEALTH_DBM_INFO. 300

© Copyright IBM Corp. 2006, 2010

HEALTH_GET_ALERT_ACTION_CFG 表関数 - ヘルス・アラート・アクション構成設定値の検索 301
HEALTH_GET_ALERT_CFG 表関数 - ヘルス・アラート構成設定値の検索 304
HEALTH_GET_IND_DEFINITION 表関数 - ヘルス・インディケーター定義の検索. 308
HEALTH_HI_REC 310
HEALTH_TBS_HI 312
HEALTH_TBS_HI_HIS 314
HEALTH_TBS_INFO 318
MQSeries ルーチン 320
MQPUBLISH 320
MQREAD. 322
MQREADALL 323
MQREADALLCLOB 325
MQREADCLOB. 327
MQRECEIVE 328
MQRECEIVEALL 329
MQRECEIVEALLCLOB 332
MQRECEIVECLOB 334
MQSEND. 335
MQSUBSCRIBE. 337
MQUNSUBSCRIBE 338
セキュリティー・ルーチンおよびビュー 340
AUTH_LIST_AUTHORITIES_FOR_AUTHID . . 340
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 343
AUTH_LIST_ROLES_FOR_AUTHID 関数 - ロールのリストを戻す 345
AUTHORIZATIONIDS 管理ビュー - 許可 ID およびタイプの検索 347
OBJECTOWNERS 管理ビュー - オブジェクト所有権情報の検索. 348
PRIVILEGES 管理ビュー - 特権情報の検索 . . 349
スナップショット・ルーチンおよびビュー. . . . 350
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 350
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. . . 351
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. 356
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 358
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 360
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 361
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. 363
LOCKWAITS 管理ビュー - 現行のロック待機情報の検索 366
LOG_UTILIZATION 管理ビュー - ログ使用率に関する情報の検索 370
LONG_RUNNING_SQL 管理ビュー 371
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. 374
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. 375
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 378
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 382
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 . . 390
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 399
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. 404
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. 407
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 411
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 . 423
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 427
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. 431
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 434
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. . . 437
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 442
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 445

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 . . . 447
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 . . . 451
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 457
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 464
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 471
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. 473
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 477
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 . . 480
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 483
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 488
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. 494
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. . . 499
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 503
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 507
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 510
SNAP_WRITE_FILE プロシージャー. 513
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. 515
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 518
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 522
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 . . 530
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 539
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. 544
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. 547
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 551
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 . 563
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 567
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. 571
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 574
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. . . 577
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 582
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 585
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 . . . 587
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 . . . 591

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 597
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 604
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 611
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. 613
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 617
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 . . 620
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 623
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 628
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. 634
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. . . 639
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 643
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 647
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 650
SNAP_WRITE_FILE プロシージャー. 653
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 . . . 655
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 . . . 657
SQL プロシージャー・ルーチン 658
GET_ROUTINE_OPTS 658
GET_ROUTINE_SAR 659
PUT_ROUTINE_SAR 660
REBIND_ROUTINE_PACKAGE 661
SET_ROUTINE_OPTS. 662
段階的な再配分ルーチン. 663
ANALYZE_LOG_SPACE プロシージャー - ログ・スペース分析情報の検索 663
GENERATE_DISTFILE プロシージャー - データ配分ファイルの生成 665
GET_SWRD_SETTINGS プロシージャー - 再配分情報の検索 666
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. 669
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 671
ストレージ管理ツール・ルーチン. 673
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. 673
CREATE_STORAGEMGMT_TABLES プロシージャー - ストレージ管理表の作成 674
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 . . 675
テキスト検索ルーチン 676
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 . . . 676
SYSTS_ALTER プロシージャー - 索引の更新特性の変更 677
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 681
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 . . 683
SYSTS_CREATE プロシージャー - 列のテキスト検索索引の作成 686
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 . . . 693
SYSTS_DROP プロシージャー - テキスト検索索引のドロップ 695
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 . . . 698
SYSTS_UPDATE プロシージャー - テキスト検索索引の更新 699
ワークロード管理ルーチン 702
WLM_CANCEL_ACTIVITY - アクティビティーのキャンセル 702
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 703
WLM_COLLECT_STATS - ワークロード管理統計の収集およびリセット. 705
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 707
WLM_GET_QUEUE_STATS 表関数 - しきい値キュー統計を戻す 714
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 . . 718
WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES - ワークロード・オカレンスのリスト. 724
WLM_GET_SERVICE_SUBCLASS_STATS - サービス・サブクラスの統計を戻す 728

WLM_GET_SERVICE_SUPERCLASS_STATS - サービス・スーパークラスの統計を戻す . . . 734
WLM_GET_WORK_ACTION_SET_STATS - 作業アクション・セット統計を戻す 736
WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES - アクティビティーのリストを戻す 737
WLM_GET_WORKLOAD_STATS - ワークロード統計を戻す 741
WLM_SET_CLIENT_INFO プロシージャー - クライアント情報設定 744
その他のルーチンおよびビュー 746
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. . . 746
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 . . 751
ALTOBJ 753
APPLICATION_ID 756
COMPILATION_ENV 表関数 - コンパイル環境のエレメントの検索 756
CONTACTGROUPS 管理ビュー - 連絡先グループのリストの検索 759
CONTACTS 管理ビュー - 連絡先のリストの検索 760
DB_HISTORY 管理ビュー - 履歴ファイル情報の検索. 761
DBPATHS 管理ビュー - データベース・パスの検索 766
EXPLAIN_FORMAT_STATS 770
EXPLAIN_GET_MSGS 774
GET_DBSIZE_INFO 777
NOTIFICATIONLIST 管理ビュー - ヘルス通知の連絡先リストの検索 779
PD_GET_DIAG_HIST - 指定された機能からレコードを戻す 780
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 787
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 794
REORGCHK_TB_STATS プロシージャー - 再編成の評価用の表統計の検索 796
SQLERRM スカラー関数 - エラー・メッセージ情報の検索 797
SYSINSTALLOBJECTS 800
MON_GET_FCM - FCM メトリックの取得 . . 801
MON_GET_FCM_CONNECTION_LIST - すべての FCM 接続の詳細の取得 802

第 4 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 805

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 809
GET_DB_CONFIG 818
GET_DBM_CONFIG 819
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 820
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 . . 828
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 836
SNAP_GET_CONTAINER 840
SNAP_GET_DB 841
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 849
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. . . . 852
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 863
SNAP_GET_DYN_SQL 867
SNAP_GET_STO_PATHS. 869
SNAP_GET_TAB 870
SNAP_GET_TBSP 871
SNAP_GET_TBSP_PART. 874
SNAPSHOT_AGENT 877
SNAPSHOT_APPL 878
SNAPSHOT_APPL_INFO. 883
SNAPSHOT_BP. 885
SNAPSHOT_CONTAINER 888
SNAPSHOT_DATABASE. 889
SNAPSHOT_DBM 895
SNAPSHOT_DYN_SQL 897
SNAPSHOT_FCM 899
SNAPSHOT_FCMNODE 900
SNAPSHOT_FILEW 901
SNAPSHOT_LOCK. 902
SNAPSHOT_LOCKWAIT. 903
SNAPSHOT_QUIESCERS 905
SNAPSHOT_RANGES. 906
SNAPSHOT_STATEMENT 907
SNAPSHOT_SUBSECT 909
SNAPSHOT_SWITCHES 911
SNAPSHOT_TABLE 912
SNAPSHOT_TBREORG 913
SNAPSHOT_TBS 915
SNAPSHOT_TBS_CFG 917
SQLCACHE_SNAPSHOT. 919
SYSINSTALLROUTINES. 921

第 2 部 付録 923

付録 A. DB2 技術情報の概説 925

DB2 テクニカル・ライブラリー (ハードコピーまたは PDF 形式) 926
DB2 の印刷資料の注文方法. 929
コマンド行プロセッサーから SQL 状態ヘルプを表示する. 930
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 930
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. 930

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 931
DB2 チュートリアル 933
DB2 トラブルシューティング情報 934
ご利用条件 934
付録 B. 特記事項 937
索引 941

第 1 部 管理 SQL ルーチンおよびビュー

管理ルーチンおよびビューは、DB2® を SQL により管理するための、基本の使いやすいプログラマチック・インターフェースを備えています。これは、さまざまな DB2 管理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、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 の検索などです。

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。

© Copyright IBM Corp. 2006, 2010

第 1 章 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。

© Copyright IBM Corp. 2006, 2010

第 2 章 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、管理ビューを導入しています。

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' (_V95 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可能性があるので、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

第 3 章 サポートされる管理 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。

- アクティビティー・モニター用管理 SQL ルーチン: 表 1
- 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: 8 ページの表 2
- 管理タスク・スケジューラー・ルーチンおよびビュー: 21 ページの表 18
- 監査ルーチンおよびプロシージャー 8 ページの表 3
- 自動保守管理 SQL ルーチンおよびビュー: 8 ページの表 4
- 構成用管理 SQL ルーチンおよび管理ビュー: 9 ページの表 5
- 共通 SQL API ストアード・プロシージャー: 9 ページの表 6
- 環境用管理ビュー: 10 ページの表 7
- ヘルス・スナップショット用管理 SQL ルーチン: 10 ページの表 8
- MQSeries® 管理 SQL ルーチン: 12 ページの表 9
- セキュリティー用管理 SQL ルーチンおよび管理ビュー: 13 ページの表 10
- スナップショット用管理 SQL ルーチンおよび管理ビュー: 13 ページの表 11
- SQL プロシージャー用管理 SQL ルーチン: 16 ページの表 12
- 段階的再配分用管理 SQL ルーチン: 17 ページの表 13
- ストレージ管理ツール用管理 SQL ルーチン: 17 ページの表 14
- テキスト検索管理 SQL ルーチン: 17 ページの表 15
- ワークロード管理ルーチン: 18 ページの表 16
- その他の管理 SQL ルーチンおよび管理ビュー: 19 ページの表 17

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	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	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	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	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。
AM_GET_LOCK_CHNS	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。

© 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_RPT	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	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	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	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_GET_DBP_MEM_USAGE	SYSPROC	この関数は、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_MSGS	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。
ADMINTABCOMPRESSINFO および ADMIN_GET_TAB_COMPRESS_INFO	SYSPROC	この関数は、表、マテリアライズ照会表 (MQT)、および階層表のみの圧縮情報を戻します。
ADMIN_REMOVE_MSGS	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。

表 3. 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	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	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	SYSPROC	このプロシージャーは、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 4. 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	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 4. 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_GET_POLICYFILE	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_SET_POLICY	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	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 5. 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	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	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	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	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 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、指定したアクティビティーをキャンセルします。ユニークなアクティビティー ID を何も指定しない場合、接続されたアプリケーションのアクティビティーをすべてキャンセルし、このアプリケーションを強制的にシステムから切断します。
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。そのデータには、すべてのデータベース・パーティションからの 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、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・サーバーについての情報を取得します。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 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。

表 6. 共通 SQL API ストアード・プロシージャー (続き)

ルーチン名またはビュー名	スキーマ	説明
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される構成パラメーターを更新します。

表 7. 環境管理ビュー

ビュー名	スキーマ	説明
ENV_FEATURE_INFO	SYSPROC	この表関数は、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	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	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	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	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 8. 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	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	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	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	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	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	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。

表 8. 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_DB_HIC_HIS	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	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。
HEALTH_DBM_HI	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	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	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	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	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	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	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	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。

表 8. 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_TBS_HI_HIS	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	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 9. MQSeries 管理 SQL ルーチン

ルーチン名	スキーマ	説明
MQPUBLISH	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	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	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	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	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	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	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	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	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	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	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	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 10. 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	SYSPROC	この関数は、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	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グループのリストを戻します。
AUTH_LIST_ROLES_FOR_AUTHID	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	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	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	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 11. 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
APPLICATIONS	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。
APPL_PERFORMANCE	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。
BP_HITRATIO	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	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	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	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	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	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 11. 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	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	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	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	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 および SNAP_GET_AGENT	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 および SNAP_GET_AGENT_MEMORY_POOL	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 および SNAP_GET_APPL_V95	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 および SNAP_GET_APPL_INFO_V95	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 および SNAP_GET_BP_V95	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 および SNAP_GET_BP_PART	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 および SNAP_GET_CONTAINER_V91	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 および SNAP_GET_DB_V95	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 11. 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 および SNAP_GET_DB_MEMORY_POOL	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 および SNAP_GET_DBM_V95	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 および SNAP_GET_DBM_MEMORY_POOL	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 および SNAP_GET_DETAILLOG_V91	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 および SNAP_GET_DYN_SQL_V95	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 および SNAP_GET_FCM	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 および SNAP_GET_FCM_PART	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 および SNAP_GET_HADR	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 および SNAP_GET_LOCK	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 および SNAP_GET_LOCKWAIT	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。
	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 および SNAP_GET_STORAGE_PATHS	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 11. 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 および SNAP_GET_SUBSECTION	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。
	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 および SNAP_GET_TAB_V91	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 および SNAP_GET_TAB_REORG	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 および SNAP_GET_TBSP_V91	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 および SNAP_GET_TBSP_PART_V91	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 および SNAP_GET_TBSP_QUIESCER	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 および SNAP_GET_TBSP_RANGE	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 および SNAP_GET_UTIL	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 および SNAP_GET_UTIL_PROGRESS	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	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	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 12. 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
GET_ROUTINE_OPTS	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。

表 12. 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_SAR	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	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	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	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 13. 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	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	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	SYSPROC	このプロシージャーは、再配分情報を戻します。
SET_SWRD_SETTINGS	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	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 14. 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	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	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	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 15. 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	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。

表 15. テキスト検索管理 SQL ルーチン (続き)

ルーチン名	スキーマ	説明
SYSTS_ALTER	SYSPROC	このプロシージャーは、索引の更新特性を変更します。
	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	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	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	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	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	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	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 16. ワークロード管理ツール管理 SQL ルーチン

ルーチン名	スキーマ	説明
WLM_CANCEL_ACTIVITY	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	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	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_ACTIVITY_DETAILS	SYSPROC	この関数は、APPLICATION_HANDLE、UOW_ID、および ACTIVITY_ID で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。
WLM_GET_QUEUE_STATS	SYSPROC	この関数は、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 16. ワークロード管理ツール管理 SQL ルーチン (続き)

ルーチン名	スキーマ	説明
WLM_GET_SERVICE_CLASS_AGENTS	SYSPROC	この関数は、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	SYSPROC	この関数は、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	SYSPROC	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。
WLM_GET_SERVICE_SUPERCLASS_STATS	SYSPROC	この関数は、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	SYSPROC	この関数は、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	SYSPROC	この関数は、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	SYSPROC	この関数は、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	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 17. 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	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	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。
ADMINTABINFO および ADMIN_GET_TAB_INFO_V95	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、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。
ALTOBJ	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	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	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	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。

表 17. 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
CONTACTS	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	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	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	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	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	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。
PD_GET_DIAG_HIST	SYSPROC	この関数は、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。
NOTIFICATIONLIST	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。
PDLOGMSGS_LAST24HOURS	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	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	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。
SQLERRM	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。

表 17. 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
SYSINSTALLOBJECTS	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。

表 18. 管理タスク・スケジューラー・ルーチンおよびビュー

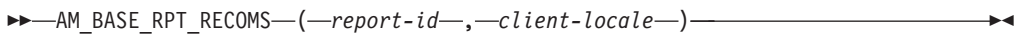
ルーチン名またはビュー名	スキーマ	説明
ADMIN_TASK_ADD	SYSPROC	このプロシージャーは、管理タスクをスケジュールに入れます。
ADMIN_TASK_LIST	SYSTOOLS	この管理ビューは、スケジューラーで定義されている各タスクに関する情報を検索します。
ADMIN_TASK_REMOVE	SYSPROC	このプロシージャーは、スケジュールされたタスクまたはタスク状況レコードを除去します。
ADMIN_TASK_STATUS	SYSTOOLS	この管理ビューは、各タスクの状況に関する情報を検索します。
ADMIN_TASK_UPDATE	SYSPROC	このプロシージ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

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
```

戻される情報

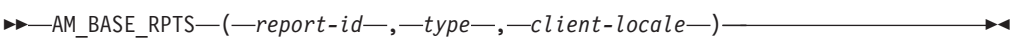
表 19. AM_BASE_RPT_RECOMS 表関数によって戻される情報

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。有効な値は次のとおりです。

'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 20. 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。

表 20. AM_BASE_RPTS 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
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

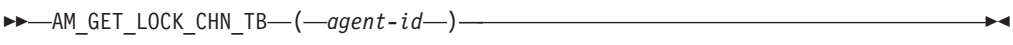
ID 5 のモニター・タスクをドロップします。

```
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。

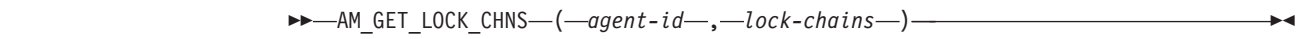
表 21. AM_GET_LOCK_CHN_TB プロシージャーによって戻される情報

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。

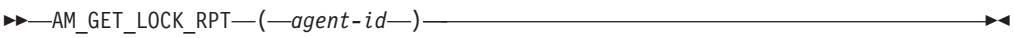
例

```
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 22. 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 22. *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 22. *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 22. 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 23. 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 23. 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 23. 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 • NX • S • SIX • U • W • 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 24. 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 24. 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

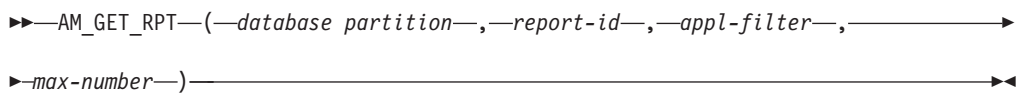
表 24. AM_GET_LOCK_RPT プロシージャーから戻されるロック待機情報 (続き)

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 • NX • S • SIX • U • W • 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 • NX • S • SIX • U • W • 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)
```

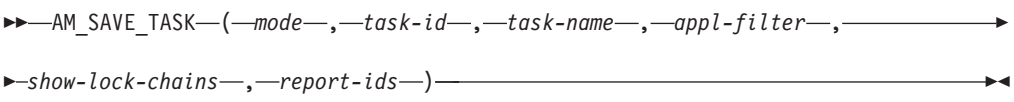
使用上の注意

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')
```

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
- 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・メッセージは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 25. 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)

このコマンドは、システムでローカルに定義されるか、またはグローバル・リストで定義される、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。

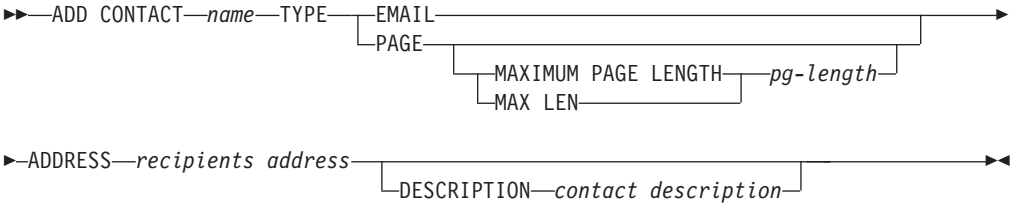
許可

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、DB2 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。

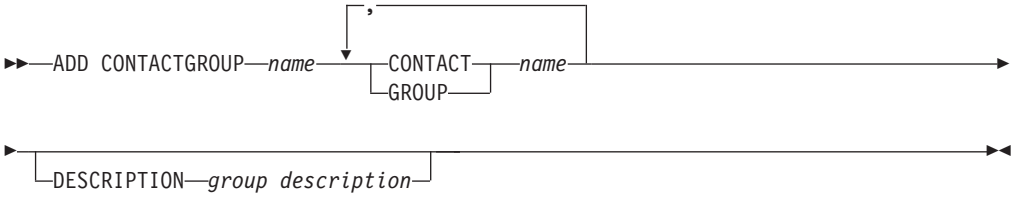
許可

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、これらの推奨値を適用するオプションと共に表示します。

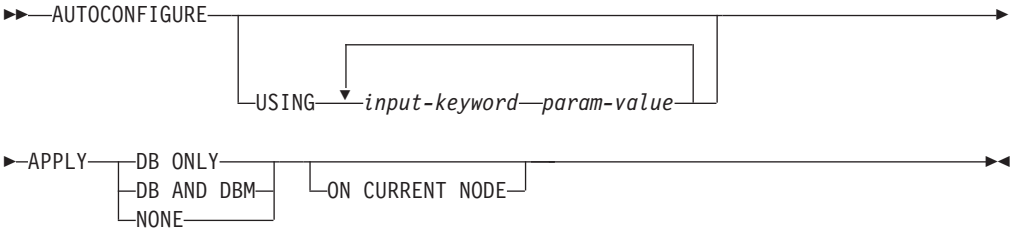
許可

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

USING *input-keyword param-value*

表 26. 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	1-100	25	データベースに割り当てられるインスタンス・メモリーのパーセンテージ。CREATE DATABASE コマンドが構成アドバイザーを呼び出す場合、mem_percent の値が指定されていなければ、パーセンテージはインスタンスとシステムでのメモリー使用量に基づいて計算され、インスタンス・メモリーの 25% が最大となります。
workload_type	simple、mixed、complex	mixed	単純 (simple) ワークロードは入出力集約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 集約の傾向があり大部分が照会です。
num_stmts	1-1,000,000	10	作業単位ごとのステートメント数
tpm	1-200,000	60	1 分ごとのトランザクション

表 26. 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
admin_priority	performance、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	データベースがデータで移植されるかどうか
num_local_apps	0-5,000	0	接続されたローカル・アプリケーションの数
num_remote_apps	0-5,000	10	接続されたリモート・アプリケーションの数
分離	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・プールがサイズ変更可能かどうかを示します。

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、データベース・パーティション機能 (DPF) 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、データベースに対する自動構成を呼び出します。

```
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

- セルフチューニング・メモリー・マネージャーが有効になっておらず、インスタンスに複数のアクティブ・データベースがある場合は、データベース・メモリー配分を反映した mem_percent 値を指定してください。例えば、インスタンスにアクティブ・データベースが 2 つあり、これらがインスタンス・メモリーの 80% を使用し、リソースを平等に共有する場合は、mem_percent 値として 40% (80% を 2 データベースで割る) を指定します。
- 同一コンピューター上に複数インスタンスがあり、セルフチューニング・メモリー・マネージャーが有効になっていない場合は、各インスタンスの **instance_memory** に固定値を設定するか、データベース・メモリー配分を反映した mem_percent 値を指定する必要があります。例えば、アクティブ・データベース全体がコンピューター・メモリーの 80% を使用し、4 つのインスタンスにそれぞれ 1 つのデータベースがある場合は、mem_percent 値として 20% (80% を 4 データベースで割る) を指定します。
- 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。しかし、SHEAPTHRES がゼロでないインスタンスにおいて、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・メモリー調整 (SORTHEAP) は自動的には有効になりません。ソート・メモリー調整 (SORTHEAP) を有効にするには、UPDATE DATABASE MANAGER CONFIGURATION コマンドを使用することによって 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、それ以前から存在しているデータベースのソート・メモリー使用量に影響することがあります。
- 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。
- 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 27. 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 27. 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について詳しくは、『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・データベース環境の場合、データベース・パーティションを指定しなければ、このコマンドは、それが実行されたデータベース・パーティションに対してだけ影響を与えます。

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を指定した場合は、カタログ・ノードに対してのみこのコマンドを呼び出せます。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オプションを指定した場合は、`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・サーバーが影響を受けます。そうでない場合は、コマンドに指定したデータベース・パーティション・サーバーが影響を受けます。

許可

以下のいずれか。

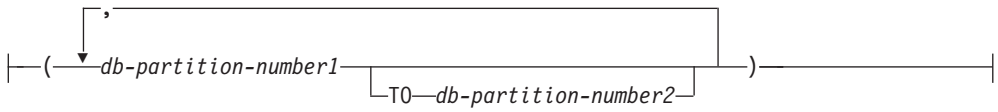
- SYSADM
- SYSCTRL
- SYSMAINT

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でデータベースをバックアップします。この節は、カタログ・パーティションの場合にのみ指定します。

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* (この番号を含む) まで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・リストに組み込まれます。

ALL DBPARTITIONNUMS
`db2nodes.cfg` ファイルに指定されている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
データベース・パーティション・リストに指定されているパーティションを除き、`db2nodes.cfg` ファイルに指定されている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* (この番号を含む) まで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・リストに組み込まれます。

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、正常に実行された全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・バックアップを作成す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 データ・サーバーに組み込まれているのは、以下のストレージ・ハードウェアに対応した DB2 ACS API ドライバーです。

- IBM TotalStorage SAN Volume Controller
- IBM Enterprise Storage Server Model 800
- IBM System Storage DS6000
- IBM System Storage DS8000
- IBM System Storage N シリーズ
- NetApp V シリーズ

- NetApp FAS

他のストレージ・ハードウェアと、そのストレージ・ハードウェアに対応した DB2 ACS API ドライバーがある場合は、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。このストリングは、入力されたとおりに (ただし、二重引用符なしで) 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。その代わりに、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。オンライン・バックアップで、**INCLUDE LOGS** オプションが指定されている場合、初期セッションがクローズした後で、このパラメーター用に追加のセッションが作成されます。

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・データベースの場合、ターゲット・ディレクトリーまたはターゲット装置は、すべてのデータベース・パーティションに存在している必要があります。また、オプションとして、共有パスを指定することも可能です。ディレクトリー名または装置名を、データベース・パーティション式を使用して指定することもできます。データベース・パーティション式の詳細については、『*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。指定したバッファーの数が、正常にバックアップを作成するために十分でない場合、このパラメーターにはバックアップを完了するために必要な最小値が自動的に選択されます。複数の場所にバックアップしている場合は、指定するバッファーの数を増やして、パフォーマンスを向上させることができます。パフォーマンスを向上させるために **COMPRESS** パラメーターを指定する場合、 **PARALLELISM** パラメーターに指定する各表スペースに対してさらに 1 つバッファーを追加できます。

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。

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/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 '@' なら、データの残りの部分は、サーバー上に存在するファイルの名前を指定するものとして DB2 によって解釈されます。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。次のバックアップ・シナリオでは、ログはデフォルトで除外されます。

- 単一パーティション・データベースのオフライン・バックアップ

- 複数パーティション・データベースのオンラインまたはオフライン・バックアップ (単一システム・ビュー・バックアップを使用しない場合)

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・バックアップは例外です。次のバックアップ・シナリオでは、ログはデフォルトで含められます。

- 単一パーティション・データベースのオンライン・バックアップ
- 複数パーティション・データベースの、オンラインまたはオフラインの単一システム・ビュー (SSV) バックアップ
- オンラインまたはオフラインのスナップショット・バックアップ

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時に可変長ブロック・サイズを使用することは、現在サポートされていません。このオプションを使用しなければならない場合は、可変長ブロック・サイズで作成されたバックアップ・イメージを使用して正常なリカバリーができる、十分にテストされたプロシージャーを用意しておいてください。

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。最適パフォーマンスを得るには、バッファー・サイズを、使用する装置の最大ブロック・サイズ限度と等しくしてください。

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。

データベースを定期的にバックアップすると、非常に大きなデータベース・バックアップ・イメージと、多くのデータベース・ログおよびロード・コピー・イメージが累積されて、これらすべてを合わせると、大量のディスク・スペースを占有する場合があります。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。

OPTIONS パラメーターを使用して、プロキシー・ノードをサポートする TSM 環境におけるバックアップ操作を有効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』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。バックアップ操作では 1 つの結果セットが戻され、その結果セットに、バックアップの対象になったデータベース・パーティションごとに 1 つの行が書き込まれます。

表 28. 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	指定のデータベース・パーティションのバックアップ処理の結果得られた最後の SQLCODE。
SQLERRMC	VARCHAR(70)	指定のデータベース・パーティションのバックアップ処理の結果得られた最後の SQLERRMC。
SQLERRML	SMALLINT	指定のデータベース・パーティションのバックアップ処理の結果得られた最後の SQLERRML。

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、または従来の単一パーティション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、単一行の結果セットが生成されます。DBPARTITIONNUM には、バックアップ

対象のデータベース・パーティションの ID 番号が入ります。

SQLCODE、SQLERRMC、SQLERRML はそれぞれ、指定のデータベース・パーティションのバックアップから戻された SQLCA の等価の名前の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)

このコマンドは以下の事柄を行います。

- SELECT、CALL、または XQuery ステートメントに関する出力情報の表示
- 表またはビューの列の表示
- 表またはビューの索引の表示
- 表またはビューのデータ・パーティションの表示

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、以下の表に表示されている権限のいずれかが必要です。

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 ステートメント中で参照された表またはビューごとに、以下の特権または権限のいずれか。 • SELECT 特権 • ACCESSCTRL 権限 • DATAACCESS 権限 • DBADM 権限 • SECADM 権限 • SQLADM 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。

情報を表示するオブジェクト	必要な特権または権限
表またはビューの列	SYSCAT.COLUMNS システム・カタログ表に対する以下の特権または権限のいずれか。 • 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、宣言された一時表に対するすべての特権が付与されているため、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された一時表に関する情報を表示できます。
表またはビューの索引	SYSCAT.INDEXES システム・カタログ表に対する以下の特権または権限のいずれか。 • 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、宣言された一時表に対するすべての特権が付与されているため、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された一時表に関する情報を表示できます。

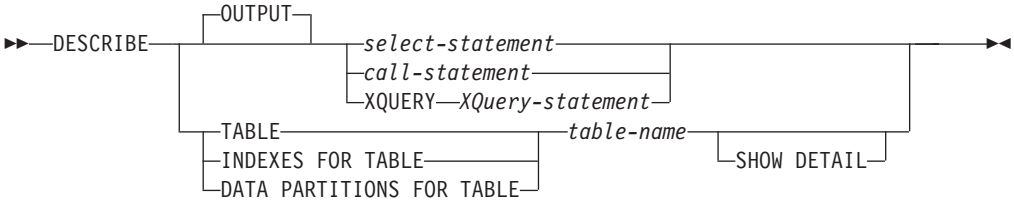
情報を表示するオブジェクト	必要な特権または権限
表またはビューのデータ・パーティション	SYSCAT.DATAPARTITIONS システム・カタログ表に対する以下の特権または権限のいずれか。 • SELECT 特権 • ACCESSCTRL 権限 • DATAACCESS 権限 • DBADM 権限 • SECADM 権限 • SQLADM 権限 PUBLIC には、宣言された一時表に対するすべての特権が付与されているため、このコマンドを使用して、接続内に存在するすべての宣言された一時表に関する情報を表示できます。

- 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* 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。

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
- 列カウント

DESCRIBE INDEXES FOR TABLE コマンドの場合、18 バイトを超えている索引名は切り捨てられます。出力には、以下のような追加情報が組み込まれます。

- 列名

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)

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。

```
CALL SYSPROC.ADMIN_CMD('describe indexes for table t1')
```

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        NUMBER_OF_COLUMNS COLNAMES
---------..- ------------------ ------------------ ----------------- -------------
SYSIBM       SQL050117181625680 PRIMARY_INDEX                      1 +PK
TXU          T1_INDEX1          DUPLICATES_ALLOWED                 1 +C1

  2 record(s) selected.

  Return Status = 0
```

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 コマンドが一時表の作成を試みて失敗した場合、SYSTOOLSTMPSPACE の作成が試みられてから、一時表の作成が、今度は SYSTOOLSTMPSPACE において、もう一度試みられます。 SYSTOOLSTMPSPACE 表スペースを作成するには、SYSCTRL または SYSADM 権限が必要です。

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 29: DESCRIBE *select-statement*、DESCRIBE *call-statement*、DESCRIBE XQUERY *XQuery-statement* の各コマンド
- 63 ページの表 30: DESCRIBE TABLE コマンドの結果セット 1
- 64 ページの表 31: DESCRIBE TABLE コマンドの結果セット 2
- 65 ページの表 32: DESCRIBE INDEXES FOR TABLE コマンド
- 65 ページの表 33: DESCRIBE DATA PARTITIONS FOR TABLE コマンドの結果セット 1
- 65 ページの表 34: DESCRIBE DATA PARTITIONS FOR TABLE コマンドの結果セット 2

表 29. 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。

表 29. 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 (続き)

列名	データ・タイプ	LOB のみ[1]	説明
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]:「はい」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 30. 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。

表 30. DESCRIBE TABLE コマンドによって戻される結果セット 1 (続き)

列名	データ・タイプ	詳細[2]	説明
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 • 'Y' - 列が NULL 可能の場合 • 'N' - 列が NULL 可能でない場合
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]:「はい」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 31. 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 32. 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[2]	説明
INDSCHEMA	VARCHAR (128)	いいえ	索引スキーマ名。
INDNAME	VARCHAR (128)	いいえ	索引名。
UNIQUE_RULE	VARCHAR (30)	いいえ	以下のいずれか • DUPLICATES_ALLOWED • PRIMARY_INDEX • UNIQUE_ENTRIES_ONLY
COLCOUNT	SMALLINT	いいえ	キー内の列数と組み込み列 (存在する場合) の数の合計。
COLNAMES	VARCHAR (2048)	はい	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: [2]:「はい」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 33. 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]:「はい」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 34. 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。

表 34. 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
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。
ACCESSMODE	VARCHAR(20)	データ・パーティションのアクセシビリティー。以下のいずれか。 • FULL_ACCESS • NO_ACCESS • NO_DATA_MOVEMENT • READ_ONLY
STATUS	VARCHAR(64)	データ・パーティションの状況。以下のいずれか。 • 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保守されるタプルを SYSDATAPARTITIONS 内に持った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・フォーマットのどれかにデータをエクスポートします。ユーザーは、SQL SELECT ステートメントを提供するか、型付き表の階層情報を提供して、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。

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

許可

以下のいずれか。

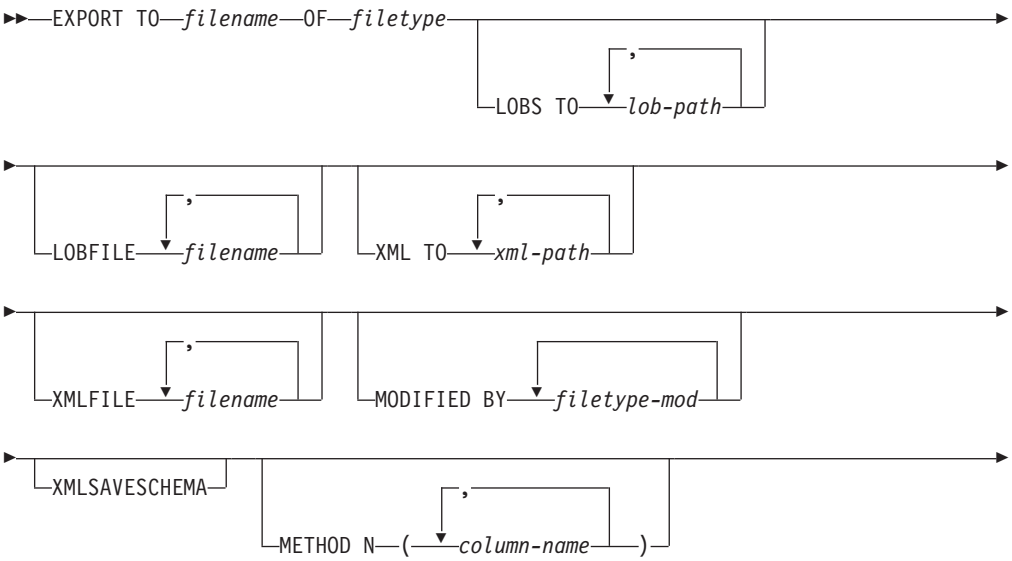
- *sysadm*
- *dbadm*

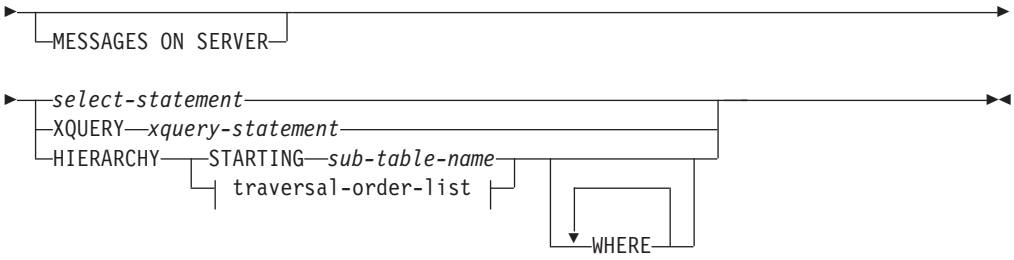
または、関係する各表またはビューに対する CONTROL または SELECT 特権

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。最初の名前の名前スペース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 ファイル名で使用する 3 桁のシーケンス番号は、999 の後は 4 桁になり、9999 の後は 5 桁になる、といった具合に大きくなっていきます。

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・ファイルを保管することを指定します。 戻される結果セットには、2 つの列が含まれます。1 つは MSG_RETRIEVAL で、この操作中に発生したすべての警告およびエラー・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、メッセージをクリーンアップするのに必要な 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。結果セット内の MSG_RETRIEVAL および MSG_REMOVAL 列には、NULL 値が入ります。

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。 75 ページの『エクスポート・ユーティリティーのファイル・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、これらの値のインポートまたはエクスポートの結果、データに間違いが生じる場合があります。
- 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。

すで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。情報の追加は行いません。

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。最初の名前の名前スペース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配分されます。

XMLSAVESCHEMA

すべての XML 列について XML スキーマ情報を保管することを指定します。挿入時に XML スキーマに照らして妥当性検査されたエクスポート後の各 XML 文書に関しては、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。エクスポート後の文書が XML スキーマに照らして妥当性検査されなかった場合や、スキーマ・オブジェクトがデータベース内にもう存在しない場合、対応する XDS に SCH 属性は組み入れられません。

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。
- 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・ファイルに置かれるメッセージには、メッセージ検索サービスから戻される情報が入っています。各メッセージは改行してから始まります。
- 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。行区切り文字の入った文字データを区切り文字付き ASCII (DEL) ファイルにエクスポートし、テキスト転送プログラムにより処理を行うと、行区切り文字の入ったフィールドは長さが伸縮します。
- 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 は、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ために使用できます。 PC/IXF エクスポートだけがサポートされています。
- IXF 形式にエクスポートするときに、ID が IXF 形式でサポートされている最大サイズを超えていると、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、後から CREATE モードでインポート操作を実行するときに使用できません。SQL27984W が戻されます。
- Windows でディスケットにエクスポートするときに、1 枚のディスケットの容量を超えるデータが表に含まれている場合は、もう 1 枚のディスケットを用意することを求めるプロンプトがシステムから出され、マルチパート PC/IXF ファイル (マルチボリューム PC/IXF ファイル、論理分割 PC/IXF ファイルともいう) がそれぞれのディスケットに生成され、格納されます。最後のファイルを除く各ファイルには、DB2 CONTINUATION RECORD (または簡略形で "AC" レコード) が書き込まれます。つまり、各ファイルが論理的に分割されていることと、次のファイルの検出場所を示すレコードです。それらのファイルは、AIX システムに転

送して、インポート・ユーティリティーとロード・ユーティリティーで読み取ることができます。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とき、全選択と*選択*ステートメント は指定できません。
- IXF 以外のファイル・フォーマットの場合は、走査順序リストを指定するようお勧めします。このリストは、階層を走査する方法やエクスポートする副表を DB2 に指示します。 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。または、OUTER 関数によって指定される順序である、デフォルトの順序を使用することができます。
- 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・ユーティリティーは、階層や副階層のロードをサポートしていません。
- 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 クリデンシャルのために、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 クリデンシャルのために、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 のようなデータ移動ユーティリティーを実行する場合、照会コンパイラーが、元になる照会は基本表よりも MQT に対して実行した方が効率的であると判断する場合があります。この場合、照会は REFRESH DEFERRED MQT に対して実行され、ユーティリティーの結果が基礎表のデータを正確に反映しない場合があります。
- エクスポート・パッケージは DATETIME ISO フォーマットを使ってバインドされるので、ストリング表現へのキャスト時に、すべての日付/時刻/タイムスタンプが ISO フォーマットに変換されます。 CLP パッケージは、DATETIME LOC フォーマット (ロケール固有のフォーマット) を使ってバインドされるので、CLP DATETIME フォーマットが ISO と異なる場合は、CLP とエクスポートの間に動作上の不整合が見られることがあります。 例えば、以下の SELECT ステートメントは、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、次のような、同じ select 節を使用した export コマンドは、そのような結果を戻しません。

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

次に、以下のように、LOCALE 日付フォーマットを ISO フォーマットに置き換えると、期待される結果が得られます。

```
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 35. 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・ユーティリティーのファイル・タイプ修飾子

表 36. 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: すべてのファイル形式

修飾子	説明
lobsinfile	*lob-path* では、LOB データが含まれているファイルのパスを指定します。 各パスには、データ・ファイルの LOB ロケーション指定子 (LLS) によって参照されている LOB が少なくとも 1 つ入っているファイルが 1 つ以上含まれています。LLS は、LOB ファイル・パスに格納されているファイルの LOB の位置を示したストリング表記です。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、データ・ファイルにストリング `db2exp.001.123.456/` が格納されている場合は、ファイル `db2exp.001` のオフセット 123 に LOB が配置されていて、その長さは 456 バイトということになります。 EXPORT 使用時に lobsinfile 修飾子を指定すると、LOB データは、LOBS TO 節に指定されている位置に配置されます。そうでなければ、LOB データは、データ・ファイル・ディレクトリーに送信されます。LOBS TO 節では、LOB ファイルを格納するディレクトリーのパスを 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 オプションまたは LOBFILE オプションを指定すると、LOBSINFILE の動作が暗黙的にアクティブになります。 NULL LOB を指定する場合は、サイズとして -1 を入力します。サイズとして 0 を指定すると、長さ 0 の LOB として処理されます。長さ -1 の NULL LOB の場合は、オフセットとファイル名が無視されます。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。
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か、その修飾子が指定されていない場合はアプリケーション・コード・ページになります。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 37. 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""`
	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。その場合は、以下のようにします。
	`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 (+) の代わりにブランク・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 進値の接頭部として正符号を使用します。
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。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]
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。正しくない使い方をすると、データが失われたり破損したりする可能性があります。 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 37. 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)
	UUUUUU - マイクロ秒 (000000 から 999999 の範囲の 6 桁の数。
	他のマイクロ秒エレメントとは相互に排他的)
	UUUUU - マイクロ秒 (00000 から 99999 の範囲の 5 桁の数。
	000000 から 999990 の範囲にマップされる。
	他のマイクロ秒エレメントとは相互に排他的)
	UUUU - マイクロ秒 (0000 から 9999 の範囲の 4 桁の数。
	000000 から 999900 の範囲にマップされる。
	他のマイクロ秒エレメントとは相互に排他的)
	UUU - マイクロ秒 (000 から 999 の範囲の 3 桁の数。
	000000 から 999000 の範囲にマップされる。
	他のマイクロ秒エレメントとは相互に排他的)
	UU - マイクロ秒 (00 から 99 の範囲の 2 桁の数。
	000000 から 990000 の範囲にマップされる。
	他のマイクロ秒エレメントとは相互に排他的)
	U - マイクロ秒 (0 から 9 の範囲の 1 桁の数。
	000000 から 900000 の範囲にマップされる。
	他のマイクロ秒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)
	
	タイム・スタンプ形式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。
	
	ユーザー定義のタイム・スタンプ形式が含まれているデータを 'schedule' という表からエクスポートする例を以下に示します。
	
	`db2 export to delfile2 of del`
	`modified by timestampformat="yyyy.mm.dd hh:mm tt"`
	`select * from schedule`

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: IXF ファイル形式

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。エクスポート操作の実行中に、文字データは、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。

表 39. 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

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 との互換性がある WSF ファイルを作成します。[5] これがデフォルトです。
2	Lotus Symphony リリース 1.0 との互換性がある 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 との互換性がある 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. MODIFIED BY オプションでサポートされていないファイル・タイプを使用しようとしても、エクスポート・ユーティリティーからは警告が生成されません。その場合は、エクスポート操作が失敗し、エラー・コードが戻されます。
2. 区切り文字のオーバーライドとして使用できる文字に適用される制約事項については、『*データ移動のための区切り文字の制約事項*』を参照してください。
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。
   - 日付データを *YYYYMMDD* 形式で書き込みます。
   - char(date) データを *YYYY-MM-DD* 形式で書き込みます。
   - 時間データを *HH.MM.SS* 形式で書き込みます。
   - タイム・スタンプ・データを *YYYY-MM-DD-HH. MM.SS.uuuuuu* 形式で書き込みます。

   エクスポート操作の SELECT ステートメントで指定する日時列に含まれているデータも、このような形式になります。
4. タイム・スタンプ形式の場合は、月の記述子と分の記述子の間であいまいさが残らないように注意する必要があります。どちらも、M という文字を使用するからです。月のフィールドは、他の日付フィールドと隣接している必要があります。分のフィールドは、他の時刻フィールドと隣接している必要があります。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)
   ```

   あいまいな場合は、ユーティリティーによってエラー・メッセージが生成され、操作は失敗します。

   あいまいでないタイム・スタンプ形式の例を以下に示します。

   ```
   "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. これらのファイルの出力先として、特定の製品を指定することもできます。*filetype-mod* パラメーター・ストリングに、Lotus 1-2-3 の場合は L、Symphony の場合は S をそれぞれ指定します。指定できるのは、1 つの値または製品指定子だけです。

6. XML 列では、WSF ファイル形式はサポートされていません。
7. XMLFILE 節や XML TO 節を指定しなくても、すべての XDM インスタンスは、メイン・データ・ファイルとは別の XML ファイルに書き込まれます。デフォルトでは、エクスポート・データ・ファイルのパスに XML ファイルも書き込まれます。XML ファイルのデフォルトのベース名は、エクスポート・データ・ファイルの名前に拡張子 ".xml" を追加した形になります。
8. XMLNODECLARATION ファイル・タイプ修飾子を指定しない限り、すべての XDM インスタンスが書き込まれるときには、エンコード属性を指定した XML 宣言が先頭に組み込まれます。
9. ファイル・タイプ修飾子 XMLCHAR または XMLGRAPHIC を指定しない限り、デフォルトでは、すべての XDM インスタンスが Unicode で書き込まれます。
10. XML データと LOB データのデフォルトのパスは、メイン・データ・ファイルのパスです。XML ファイルのデフォルトのベース名は、メイン・データ・ファイルです。LOB ファイルのデフォルトのベース名は、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・タイプ修飾子を指定する必要があります。
11. エクスポート・ユーティリティーは、それぞれの LOB ファイルまたは XML ファイルに数値 ID を追加します。この ID は、0 を埋め込んだ 3 桁のシーケンス値 (つまり、

    ```
    .001
    ```

) から始まります。999 番目の LOB ファイルまたは XML ファイルの後は、ID にゼロが埋め込まれなくなります (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
    ```

    になります)。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
```

という形式、生成される XML ファイルの名前は

```
myfile.del.001.xml
```

という形式になります。

12. エクスポート・ユーティリティーでは、整形式でない文書の XDM インスタンスでも、XQuery を指定することによってエクスポートすることが可能です。ただし、そのエクスポート文書を XML 列に直接インポートしたりロードしたりすることはできません。XML 列には完全な文書しか組み込めないからです。

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

サーバー上で保守を行えるようにするため、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システムから強制終了します。

重要: 割り込みできない操作 (例えば、RESTORE DATABASE) を強制終了する場合、データベースが利用できるようになるには、その操作の再実行が正常終了しなければなりません。

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

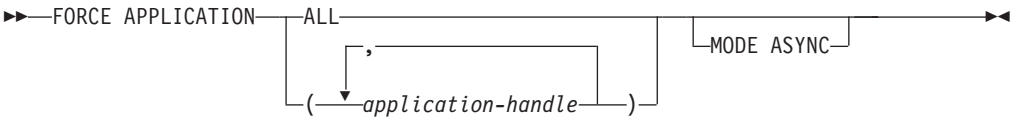
以下のいずれか。

- *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。

データベースの保全性を確保するため、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

SYSADM または DBADM 権限

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 40. 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。

101 ページの『インポート・ユーティリティーのファイル・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** コマンドを実行する場合、以下のどれかが必要です。
  - *sysadm*
  - *dbadm*
  - 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** する場合、以下のいずれかが必要です。
  - *sysadm*
  - *dbadm*
  - 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** する場合、以下のいずれかが必要です。
  - *sysadm*
  - *dbadm*
  - 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

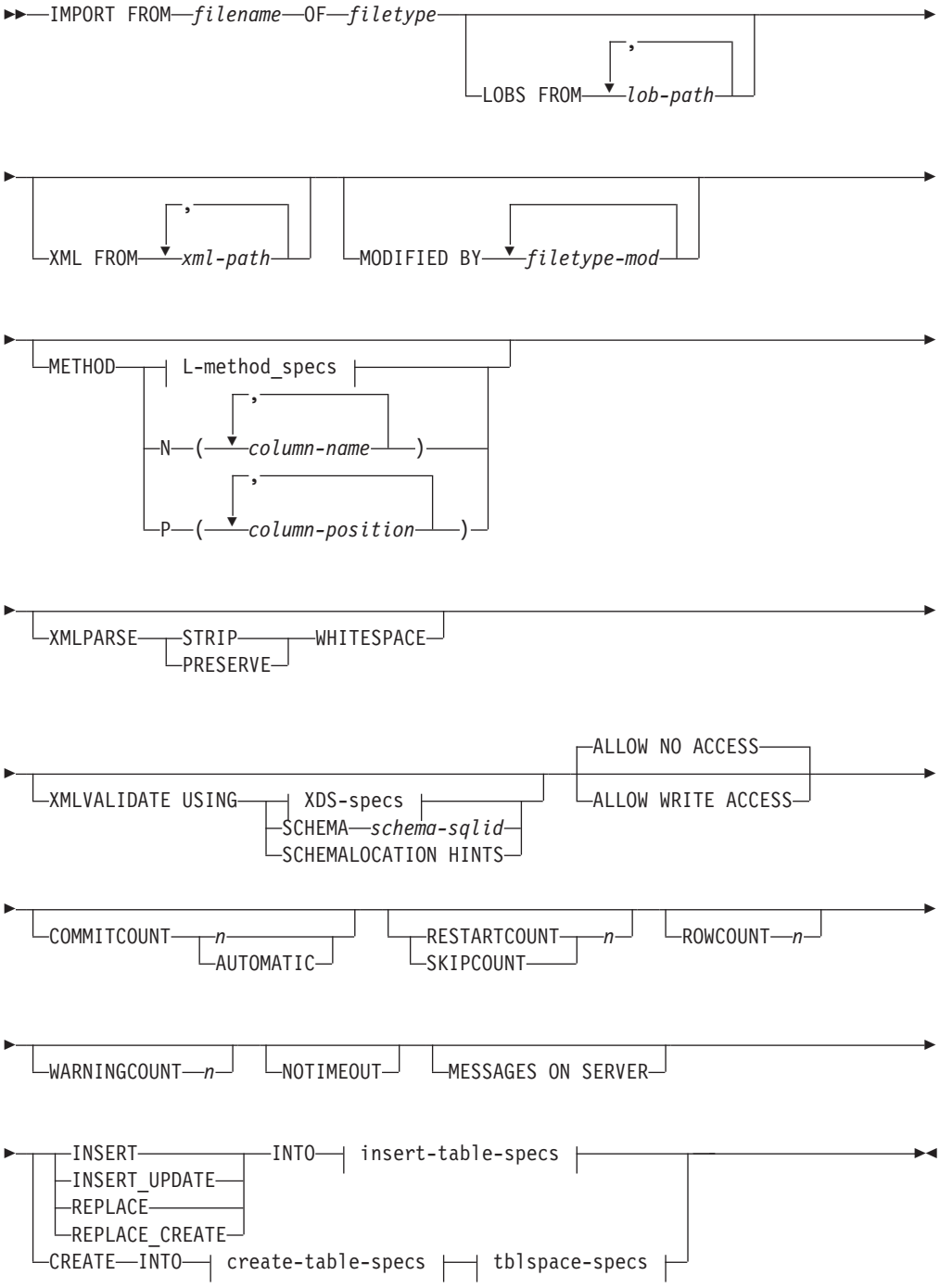
- CREATE または **REPLACE_CREATE** オプションを使用して新規の表に **IMPORT** する場合、以下のいずれかが必要です。
  - *sysadm*
  - *dbadm*
  - データベースに対する CREATETAB 権限、および表スペースに対する USE 特権に加えて、以下のいずれか。
    - データベースに対する IMPLICIT_SCHEMA 権限 (表の暗黙的または明示的スキーマ名が存在しない場合)
    - スキーマに対する CREATEIN 特権 (表のスキーマ名が既存のスキーマを指している場合)
- CREATE または **REPLACE_CREATE** 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。
  - *sysadm*
  - *dbadm*
  - 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。
    - 表のスキーマ名が存在しない場合は、データベースに対する IMPLICIT_SCHEMA 権限
    - 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、以下のいずれかが必要です。
  - *sysadm*
  - *dbadm*
  - 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 クリデンシャルが必要です。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 クリデンシャルが必要です。
  - 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

  挿入する行上のラベル、ユーザーの LBAC 信用証明情報、セキュリティー・ポリシーの定義、および LBAC 規則によって、行上のラベルが決まります。
- **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、指定したデータ・ソースにパススルー・モードでアクセスして使用できる特権をセッション許可 ID が持っていなければなりません。

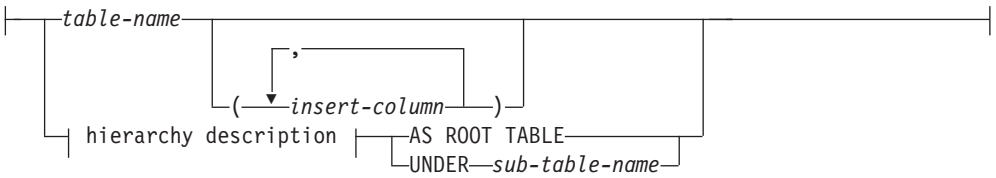
必要な接続

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。

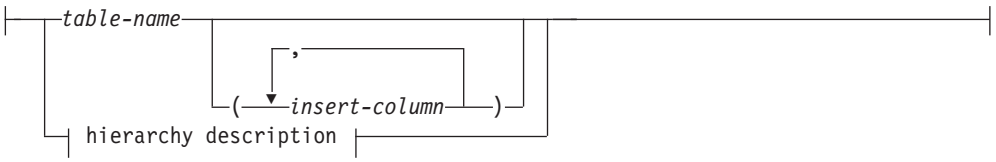
コマンド構文



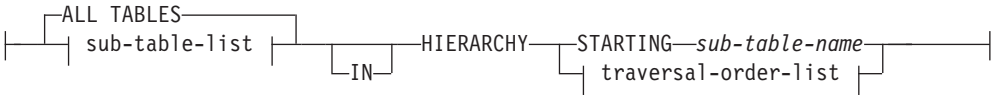
create-table-specs:



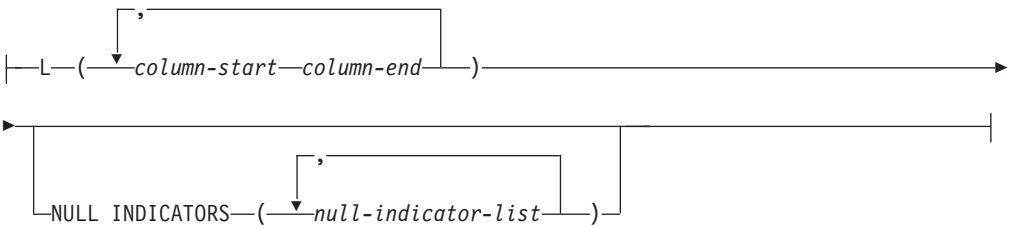
insert-table-specs:



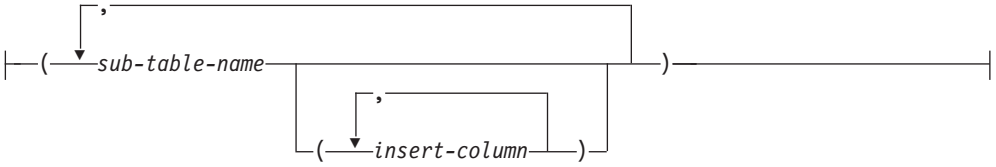
hierarchy description:



L-method-specs:

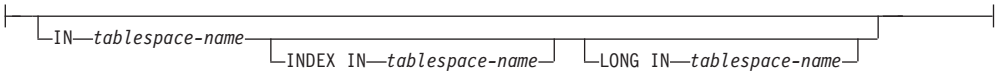


sub-table-list:

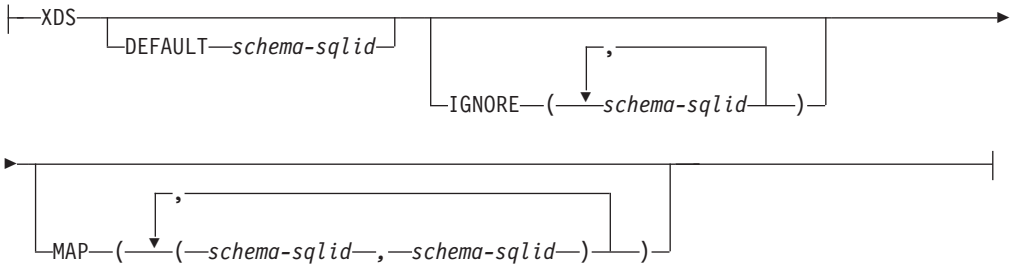


traversal-order-list:

tblspace-specs:



XDS-specs:



コマンド・パラメーター

ALL TABLES

暗黙のキーワード (階層のみ)。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。

インポート操作で、アクティブ・ログ・スペースを使いきらないようにできるかどうかは、DB2 レジストリー変数 **DB2_FORCE_APP_ON_MAX_LOG** によって影響を受けます。

- **DB2_FORCE_APP_ON_MAX_LOG** が FALSE に設定され、**COMMITCOUNT** AUTOMATIC コマンド・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自動的に、アクティブ・ログ・スペースを使いきらないようにすることができます。
- **DB2_FORCE_APP_ON_MAX_LOG** が FALSE に設定され、**COMMITCOUNT** *n* コマンド・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、レコードの挿入または更新中に SQL0964C (トランザクション・ログがいっぱい) が発生した場合に、ログ満杯状態を解決しようと試みます。インポート・ユーティリティーは、無条件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。それでも問題が解決しない場合 (ログの満杯状態がデータベース上の他のアクティビティーが原因で起きている場合など)、IMPORT コマンドは予期したとおりに失敗します。ただし、コミットされた行の数は、**COMMITCOUNT** *n* の値の倍数になっていない可能性があります。インポート操作の再試行時に、既にコミットされている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 (デフォルト) に設定されている場合、レコードの挿入または更新中に SQL0964C が発生すると、インポート操作は失敗します。これは、**COMMITCOUNT** AUTOMATIC または **COMMITCOUNT** *n* のどちらを指定する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、現行作業単位はロールバックされます。インポート操作の再試行時に、既にコミットされている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** パラメーターは非推奨になっており、将来のリリースで除去される可能性があります。詳細については、『推奨されなくなった IMPORT コマンド・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。 制約事項のリストについては、『『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

インポートする階層データを指定します。

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、インポートし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、宣言一時表、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。修飾子付き表名は、*schema.tablename* の形式です。 *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。 ペアの最初のスキーマは、XDS の中の SCH 属性によって示されるスキーマを表します。ペアの 2 番目のスキーマは、スキーマの妥当性検査を実行するために使用す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・ファイルを保管することを指定します。 戻される結果セットには、2 つの列が含まれます。1 つは MSG_RETRIEVAL で、この操作中に発生したすべての警告およびエラー・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、メッセージをクリーンアップするのに必要な 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。結果セット内の MSG_RETRIEVAL および MSG_REMOVAL 列には、NULL 値が入ります。

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。 101 ページの『インポート・ユーティリティーのファイル・タイプ修飾子』を参照してください。

NOTIMEOUT

インポート・ユーティリティーは、ロック待ちの間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 専用のバイナリー・フォーマットです。

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 UDB バージョン 8 に適用されます。

REPLACE_CREATE

注: REPLACE_CREATE パラメーターは非推奨になっており、将来のリリースで除去される可能性があります。詳細については、『『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。

表がすでにある場合には、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。制約事項のリストについては、『*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* 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トラバーサル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を誤って指定した場合、インポート・ユーティリティーは、インポートを試みた行ごとに警告を生成して、それがインポートの失敗の原因になります。 *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ってターゲット表に行を追加します。ユーティリティーは入力ファイル中のデータの行ごとに 1 つの 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すると、以下のどちらかのアクションが起きます。

- 後続の INSERT ステートメントを正常に実行できると思われる場合、メッセージ・ファイルに警告メッセージが書き込まれ、処理は続行します。
- 後続の INSERT ステートメントが失敗しそうで、データベースの損傷の可能性がある場合、メッセージ・ファイルにエラー・メッセージが書き込まれ、処理は停止します。

ユーティリティーは、**REPLACE** または **REPLACE_CREATE** 操作中に、古い行が削除された後、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、古いデータがもう必要ないことを必ず確認してください。

ログが **CREATE**、**REPLACE**、または **REPLACE_CREATE** 操作中にいっぱいになった場合、ユーティリティーは挿入されたレコード上で自動 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 コマンドによる更新は常に IMPORT タスクの終わりにコミットされます。ロック・リストとアクティブ・ログ・スペースのサイズを減らすために、IMPORT コマンドの実行中に自動コミットを行うようにすることもできます。IMPORT 処理中にアクティブ・ログがいっぱいになった場合には、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** オプションで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 を実行するときはいつでも、 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。 1 つはコミットされるレコードの数を示し、もう 1 つは正常に終了した COMMIT の後に書き込まれます。失敗の後にインポート操作を再始動する場合、最後に正常に終了した COMMIT から決定されたとおり、スキップするレコードの数を指定してください。

インポート・ユーティリティーは、小さい非互換性問題のある入力データ (例えば、文字データのインポート時に埋め込みまたは切り捨てが発生するものや、数値データのインポート時に異なる数値データ・タイプが使用されるもの) は受け入れますが、大きな非互換性に関する問題のあるデータは受け入れません。

それ自体以外への依存があるオブジェクト表や、基本表に何らかの依存 (それ自体も含めて) があるオブジェクト・ビューを、 **REPLACE** または **REPLACE_CREATE** することはできません。このような表またはビューを置換するには、次のようにします。

1. 表が親であ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、参照保全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には、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がサイズが大きくなる可能性に備えて、十分なディスク・スペースをサーバー上に確保する必要があります。

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い (60 KB より長い) 場合、クライアント上でユーザーに戻されるメッセージ・ファイルがインポート操作中に欠落することがあります。メッセージ情報の最初の 30 KB と最後の 30 KB は、常に保存されます。

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。

ASC、**DEL**、または **WSF** のファイル形式のデータをインポートするためには、それ以前にデータベース表または階層がすでに存在していなければなりません。ただし、表がまだ存在していない場合でも、IMPORT **CREATE** または IMPORT **REPLACE_CREATE** を使えば、PC/IXF ファイルからデータをインポートする際に表が作成されます。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。

データ (階層データを含む) を別のデータベースに移動するには、PC/IXF インポートを使う必要があります。行区切り文字の入った文字データを区切り文字付き ASCII (DEL) ファイルにエクスポートし、テキスト転送プログラムにより処理を行うと、行区切り文字の入ったフィールドは長さが伸縮します。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。

ここでは、ASC ファイルおよび DEL ファイル内のデータは、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・ページのデータであると想定します。別々のコード・ページにデータをインポートする場合、PC/IXF ファイル (異なるコード・ページへのインポートが考慮されたファイル) の使用をお勧めします。PC/IXF ファイルとインポート・ユーティリティーが同一のコード・ページの場合、処理は通常のアプリケーションと同じようになります。両者が異なるコード・ページであっても、**FORCEIN** オプションが指定されている場合には、インポート・ユーティリティーにおいて、PC/IXF ファイル内のデータが、インポートを実行するアプリケーションと同一のコード・ページであると見なされます。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。このようになるのは、CHAR、VARCHAR、または CLOB 型の列の場合だけです。 **DEL** または **ASC** ファイルのインポートでは、この制限は当てはまりません。 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 は、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ために使用できます。PC/IXF インポート (**INSERT** オプション) だけがサポートされています。**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。

型付き表に対して **CREATE** オプションを使うと、PC/IXF ファイルの中で定義されているすべての副表が作成されます。副表定義は変更されません。型付き表に対して **CREATE** 以外のオプションを使うと、トラバーサル順序リストによって、トラバース順序を指定できます。その場合、トラバーサル順序リストはエクスポート操作で使用されたものと一致していなければなりません。 PC/IXF ファイル・フォーマットの場合は、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横断順序を使用するだけです。

インポート・ユーティリティーを使って、前に PC/IXF ファイルにエクスポートされた表をリカバリーできます。表は、エクスポート時の状態に戻ります。

データは、システム表、宣言一時表、またはサマリー表にはインポートできません。

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。

マルチパート PC/IXF ファイルの個々のパートを Windows システムから AIX システムにコピーするインポート操作もサポートされています。IMPORT コマンドには、最初のファイルの名前だけを指定する必要があります。例えば、`IMPORT FROM data.ixf OF IXF INSERT INTO TABLE1` のように記述します。 `data.002` などのファイルも、`data.ixf` と同じディレクトリーに入れておく必要があります。

Windows オペレーティング・システム:

- 論理分割 PC/IXF ファイルのインポートはサポートされていません。
- 正しくない DEL フォーマット・ファイルの 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。この問題が起きた場合は、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、区切り文字に以前の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するニックネームがすで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。

一部のデータ・ソース (ODBC など) では、ニックネームへのインポートが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 41. 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まれたレコード数。
ROWS_SKIPPED	BIGINT	挿入または更新の開始前にスキップされたレコード数。
ROWS_INSERTED	BIGINT	ターゲット表に挿入された行数。
ROWS_UPDATED	BIGINT	ターゲット表内の行のうち、インポートされたレコードに含まれる情報によって更新されたもの (表の中に主キー値が既に存在していたレコード) 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て、データベースにコミットされたレコード数。
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`
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・ユーティリティーのファイル・タイプ修飾子

表 42. 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 までの (両端を含む) 数値です。非アトミックのコンパウンド SQL を使用してデータを挿入し、毎回 *x* 個のステートメントを試行します。 この修飾子を指定した場合に、トランザクション・ログに十分な大きさがないと、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** で指定されている行数か、**COMMITCOUNT** が指定されていなければデータ・ファイルに含まれている行数に対応できるだけの大きさでなければなりません。したがって、トランザクション・ログのオーバーフローを回避するために、**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** モードや階層表とは互換性がありません。さらに、`usedefaults`、`identitymissing`、`identityignore`、`generatedmissing`、`generatedignore` の各修飾子とも互換性がありません。
generatedignore	この修飾子を指定すると、インポート・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。その結果、生成済み列のすべての値がユーティリティーによって生成されます。この修飾子を `generatedmissing` 修飾子と一緒に使用することはできません。
generatedmissing	この修飾子を指定すると、ユーティリティーは、入力データ・ファイルに生成済み列のデータが入っていない (NULL もない) という想定で動作し、各行の値を生成します。この修飾子を `generatedignore` 修飾子と一緒に使用することはできません。
identityignore	この修飾子を指定すると、インポート・ユーティリティーは、データ・ファイルに入っている、ID 列のデータを無視するようになります。その結果、すべての IDENTITY 値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 の ID 列の場合も GENERATED BY DEFAULT の ID 列の場合も同じです。したがって、GENERATED ALWAYS 列の場合は、行がリジェクトされません。この修飾子を `identitymissing` 修飾子と一緒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、入力データ・ファイルに ID 列のデータが入っていない (NULL もない) という想定で動作し、各行の値を生成します。この動作は、GENERATED ALWAYS の ID 列の場合も GENERATED BY DEFAULT の ID 列の場合も同じです。この修飾子を `identityignore` 修飾子と一緒に使用することはできません。

表 42. 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* では、LOB データが含まれているファイルのパスを指定します。 各パスには、データ・ファイルの LOB ロケーション指定子 (LLS) によって参照されている LOB が少なくとも 1 つ入っているファイルが 1 つ以上含まれています。LLS は、LOB ファイル・パスに格納されているファイルの LOB の位置を示したストリング表記です。LLS の形式は、*filename.ext.nnn.mmm/* になります (*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、データ・ファイルにストリング db2exp.001.123.456/ が格納されている場合は、ファイル db2exp.001 のオフセット 123 に LOB が配置されていて、その長さは 456 バイトということになります。 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。**IMPORT** ユーティリティーは、データをインポートするときに、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。 NULL LOB を指定する場合は、サイズとして -1 を入力します。サイズとして 0 を指定すると、長さ 0 の LOB として処理されます。長さ -1 の NULL LOB の場合は、オフセットとファイル名が無視されます。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。通常の表からエクスポートしたデータについて、この修飾子を使用してインポート操作を起動し、データを 1 つの副表に変換する、というのが典型的な使用法です。
nodefaults	ターゲット表の列のソース列を明示的に指定しない場合に、その表列が NULL 可能でなければ、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しない状態で、ターゲット表のいずれかの列のソース列を明示的に指定しない場合は、以下のいずれか動作が発生します。 • 列のデフォルト値を指定できる場合は、そのデフォルト値がロードされます。 • 列が NULL 可能で、その列のデフォルト値を指定できない場合は、NULL がロードされます。 • 列が NULL 可能ではなく、デフォルト値も指定できない場合は、エラーが戻され、ユーティリティーが処理を停止します。
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	この修飾子を指定すると、インポート・ユーティリティーは、データ・ファイルに入っている、ROW CHANGE TIMESTAMP 列のデータを無視するようになります。その結果、すべての ROW CHANGE TIMESTAMP 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 の列の場合も GENERATED BY DEFAULT の列の場合も同じです。したがって、GENERATED ALWAYS 列の場合は、行がリジェクトされません。この修飾子を rowchangetimestampmissing 修飾子と一緒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 の列の場合も GENERATED BY DEFAULT の列の場合も同じです。この修飾子を rowchangetimestampignore 修飾子と一緒に使用することはできません。

表 42. 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・ファイルに含まれているセキュリティー・ラベルが、デフォルトのエンコード数値形式ではなく、ストリング・フォーマットのセキュリティー・ラベル値であることを指定します。IMPORT 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。ストリングが正しい形式になっていないと、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・ラベルに対応していなければ、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。そのようなことをすると、インポートは失敗し、エラー (SQLCODE SQL3525N) が戻されます。
seclabelname	入力ソース・ファイルに含まれているセキュリティー・ラベルが、デフォルトのエンコード数値形式ではなく、名前で示されていることを指定します。IMPORT 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。そのようなことをすると、インポートは失敗し、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。そのような動作を避けるには、`striptblanks` ファイル・タイプ修飾子を使用して、スペースを除去するようにします。
usedefaults	ターゲット表の列のソース列が指定されていても、1 つ以上の行インスタンスでその列に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、明示的な欠落とは見なされず、NULL 列値の代わりにデフォルトが入ることもありません。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 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: ASCII ファイル形式 (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。インポート操作の実行時に、文字データは、このコード・ページからアプリケーション・コード・ページに変換されます。 以下の規則が適用されます。 • ピュア DBCS (GRAPHIC)、混合 DBCS、EUC では、区切りが x00 から x3F の範囲 (両端を含む) に限定されています。 • 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 (両端を含む) 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。 **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・ページをアプリケーション・コード・ページからデータベース・コード・ページに変換するときに、データの拡張が発生すると、データが切り捨てられ、データが失われる可能性があります。
dateformat="*x*"	*x* は、ソース・ファイルの日付の形式です。[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) 指定されていないそれぞれのエレメントには、デフォルト値の 1 が割り当てられます。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 DECIMAL(8,2) 列に 12345.00 としてではなく、123.45 としてロードされます。
timeformat="*x*"	*x* は、ソース・ファイルの時刻の形式です。[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) 指定されていないそれぞれのエレメントには、デフォルト値の 0 が割り当てられます。時刻形式の例を以下に示します。 "HH:MM:SS" "HH.MM TT" "SSSSS"

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: ASCII ファイル形式 (ASC/DEL) (続き)

修飾子	説明

timestampformat="*x*"	*x* は、ソース・ファイルのタイム・スタンプの形式です。[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 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)
	UUUUUU - マイクロ秒 (000000 から 999999 の範囲の 6 桁の数。
	他のマイクロ秒エレメントとは相互に排他的)
	UUUUU - マイクロ秒 (00000 から 99999 の範囲の 5 桁の数。
	000000 から 999990 の範囲にマップされる。
	他のマイクロ秒エレメントとは相互に排他的)
	UUUU - マイクロ秒 (0000 から 9999 の範囲の 4 桁の数。
	000000 から 999900 の範囲にマップされる。
	他のマイクロ秒エレメントとは相互に排他的)
	UUU - マイクロ秒 (000 から 999 の範囲の 3 桁の数。
	000000 から 999000 の範囲にマップされる。
	他のマイクロ秒エレメントとは相互に排他的)
	UU - マイクロ秒 (00 から 99 の範囲の 2 桁の数。
	000000 から 990000 の範囲にマップされる。
	他のマイクロ秒エレメントとは相互に排他的)
	U - マイクロ秒 (0 から 9 の範囲の 1 桁の数。
	000000 から 900000 の範囲にマップされる。
	他のマイクロ秒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)
	
	指定されていない YYYY、M、MM、D、DD、DDD のいずれかのエレメントには、デフォルト値の 1 が割り当てられます。指定されていない MMM エレメントには、デフォルト値の 'Jan' が割り当てられます。指定されていない他のすべてのエレメントには、デフォルト値の 0 が割り当てられます。タイム・スタンプ形式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'、'dec' です。これらの値では、大/小文字は区別されません。
	
	ユーザー定義の日付と時刻の形式が含まれているデータを schedule という表にインポートする例を以下に示します。
	
	`db2 import from delfile2 of del`
	`modified by timestampformat="yyyy.mm.dd hh:mm tt"`
	`insert into schedule`

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: ASCII ファイル形式 (ASC/DEL) (続き)

修飾子	説明
usegraphiccodepage	usegraphiccodepage を指定すると、グラフィックまたは 2 バイト文字のラージ・オブジェクト (DBCLOB) データ・フィールドにインポートするデータは、グラフィック・コード・ページのデータであるという想定で、処理が行われます。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。グラフィック・コード・ページは、文字コード・ページに関連付けられています。IMPORT は、codepage 修飾子が指定されていればその修飾子によって、codepage 修飾子が指定されていなければアプリケーションのコード・ページによって、文字コード・ページを判別します。 ドロップ済み表のリカバリーで生成される区切り付きデータ・ファイルとこの修飾子を併用するのは、リカバリーする表にグラフィック・データが入っている場合に限られます。 **制約事項** EXPORT ユーティリティーで作成される DEL ファイルでは、usegraphiccodepage 修飾子を指定しないでください。そのファイルには、1 つのコード・ページでエンコードされたデータだけが入っているからです。usegraphiccodepage 修飾子は、ファイルに含まれている 2 バイト文字ラージ・オブジェクト (DBCLOB) でも無視されます。
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。 それぞれの文書で、宣言タグが存在していて、エンコード属性が含まれている場合は、そのエンコードが文字コード・ページと一致している必要があります。そうでないと、その文書が含まれている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か、その修飾子が指定されていない場合はアプリケーション・コード・ページになります。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 文書が指定のグラフィック・コード・ページでエンコードされていることを指定します。 指定のグラフィック・コード・ページでエンコードされているものの、エンコード宣言が含まれていない XML 文書を処理するときに、このオプションは便利です。 それぞれの文書で、宣言タグが存在していて、エンコード属性が含まれている場合は、そのエンコードがグラフィック・コード・ページと一致している必要があります。そうでないと、その文書が含まれている行はリジェクトされます。グラフィック・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・コンポーネントか、その修飾子が指定されていない場合はアプリケーション・コード・ページのグラフィック・コンポーネントになります。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:** IMPORT コマンドで xmlgraphic 修飾子を指定する場合は、インポート対象の XML 文書のエンコードが UTF-16 コード・ページになっている必要があります。そうでない場合は、XML 文書が構文解析エラーでリジェクトされるか、表にインポートされてもデータ破損が生じる可能性があります。

表 44. 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: ASC (区切りなし ASCII) ファイル形式

修飾子	説明
nochecklengths	nochecklengths を指定すると、ターゲット表の列のサイズを超える列定義がソース・データに含まれている場合でも、各行のインポートが試行されるようになります。コード・ページ変換によってソース・データが縮小される場合は、そのような行を正常にインポートできます。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 データに縮小されれば、必要なスペースが半分になります。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] この修飾子は、EBCDIC データ・ファイルでは大文字と小文字の区別があります。ただし、文字が英字の場合は例外です。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 44. 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: ASC (区切りなし ASCII) ファイル形式 (続き)

修飾子	説明
reclen=*x*	*x* は、最大値 32,767 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	可変長フィールドにデータをロードするときに、末尾ブランク・スペースを切り捨てます。このオプションを指定しなければ、ブランク・スペースは維持されます。
	
	以下の例では striptblanks を指定しているので、インポート・ユーティリティーは、末尾ブランク・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`
	
	このオプションを 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、廃止オプションの t の代わりに用意されています。その廃止オプションは、旧バージョンとの互換性のためだけにサポートされています。

striptnulls	可変長フィールドにデータをロードするときに、末尾 NULL (0x00 文字) を切り捨てます。このオプションを指定しなければ、NULL は維持されます。 このオプションを 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、廃止オプションの padwithzero の代わりに用意されています。その廃止オプションは、旧バージョンとの互換性のためだけにサポートされています。

表 45. 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: 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。[34] 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""`
	
	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。以下の例では chardel'' を指定しているので、インポート・ユーティリティーは、検出する単一引用符 (') を文字ストリング区切りとして解釈します。
	
	`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。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。[34]
	
	以下の例では coldel; を指定しているので、インポート・ユーティリティーは、検出するセミコロン (;) を列区切りとして解釈します。
	
	`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 (+) の代わりにブランク・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 進値の接頭部として正符号を使用します。

表 45. 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: DEL (区切り付き ASCII) ファイル形式 (続き)

修飾子	説明

decpt*x*	*x* は、小数点文字としてピリオドの代わりに使用する単一文字です。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。[34]
	
	以下の例では decpt; を指定しているので、インポート・ユーティリティーは、検出するセミコロン (;) 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:
	`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。1 番目と 3 番目の <row delimiter> は、実際のレコード区切りとして解釈されますが、2 番目の <row delimiter> は、第 2 行の最初のデータ列の一部として解釈されるからです。この修飾子を指定しなければ、それぞれの <row delimiter> が区切り文字として解釈され、このデータ・ファイルは 3 行になります。

keepblanks	タイプ CHAR、VARCHAR、LONG VARCHAR、CLOB の各フィールドで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、文字区切りの内側にない前後のすべてのブランクが除去され、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、列区切りの間で検出するすべてのバイトを列のデータの一部と見なします。文字区切りも、列データの一部として解析されます。DB2 で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。正しくない使い方をすると、データが失われたり破損したりする可能性があります。 このオプションは、chardelx、 delprioritychar、および nodoubledel のいずれとも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 46. 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: IXF ファイル形式

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 を指定すると、チェックなしで各行のインポートが試行されます。
indexixf	ユーティリティーは、既存の表に現在定義されているすべての索引をドロップし、PC/IXF ファイルの索引定義から新しい索引を作成します。このオプションを使用できるのは、表の内容を置き換える場合に限られます。ビューで使用することはできません。*insert-column* を指定した場合も、使用できません。
indexschema=*schema*	索引作成時に、索引名として指定の *schema* を使用します。 *schema* を指定しない場合に、キーワード indexschema が指定されていれば、接続ユーザー ID が使用されます。そのキーワードが指定されていなければ、IXF ファイルのスキーマが使用されます。

表 46. 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: IXF ファイル形式 (続き)

修飾子	説明
nochecklengths	nochecklengths を指定すると、ターゲット表の列のサイズを超える列定義がソース・データに含まれている場合でも、各行のインポートが試行されるようになります。コード・ページ変換によってソース・データが縮小される場合は、そのような行を正常にインポートできます。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 データに縮小されれば、必要なスペースが半分になります。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	インポート操作時に情報が欠落していたり、限定されていたりする場合でも、SQL3311N を戻してから、表を作成することを指定します。

表 47. codepage と usegraphiccodepage を使用する場合の IMPORT の動作

codepage=N	usegraphiccodepage	IMPORT の動作
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・ページのデータであるという想定になります。
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・ページの場合に、グラフィック・データをデータベースにインポートすると、グラフィック・データが破損します。
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・ページのデータであるという想定になります。グラフィック・データは、アプリケーション・グラフィック・データのコード・ページのデータであるという想定になります。 アプリケーション・コード・ページが 1 バイトの場合は、すべてのデータがアプリケーション・コード・ページのデータであるという想定になります。 **警告:** アプリケーション・コード・ページが 1 バイトの場合に、グラフィック・データをデータベースにインポートすると、データベースにグラフィック列が含まれていても、グラフィック・データは破損します。
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 バイトのコード・ページの場合は、すべてのデータがコード・ページ N のデータであるという想定になります。 **警告:** N が 1 バイト・コード・ページの場合に、グラフィック・データをデータベースにインポートすると、グラフィック・データが破損します。

注:

1. **MODIFIED BY** オプションでサポートされていないファイル・タイプを使用しようとしても、インポート・ユーティリティーからは警告が生成されません。その場合は、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、 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。エレメントの開始位置と終了位置があいまいでない場合は、フィールド区切り文字はオプションになります。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・スタンプ形式の場合は、月の記述子と分の記述子の間であいまいさが残らないように注意する必要があります。どちらも、M という文字を使用するからです。月のフィールドは、他の日付フィールドと隣接している必要があります。分のフィールドは、他の時刻フィールドと隣接している必要があります。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)
   ```

   あいまいな場合は、ユーティリティーによってエラー・メッセージが生成され、操作は失敗します。

   あいまいでないタイム・スタンプ形式の例を以下に示します。

   ```
   "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。
3. 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 # 文字を指定する場合は、以下のいずれかを使用します。

   ```
   ... modified by coldel# ...
   ... modified by coldel0x23 ...
   ... modified by coldelX23 ...
   ```
4. 区切り文字のオーバーライドとして使用できる文字に適用される制約事項については、『*データ移動のための区切り文字の制約事項*』を参照してください。
5. ニックネームにインポートする場合、以下のファイル・タイプ修飾子は使用できません。
   - 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。
7. XML 列では、**CREATE** モードはサポートされていません。
8. すべての XML データをメイン・データ・ファイルとは別の XML ファイルに配置する必要があります。メイン・データ・ファイルの各 XML 列に XML Data Specifier (XDS) (または NULL 値) が入っている必要があります。
9. ファイル・タイプ修飾子 XMLCHAR または XMLGRAPHIC が指定されている場合を除き、XML 文書は、Unicode 形式であるか、エンコード属性の宣言タグが含まれているという前提で処理が行われます。
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** オプションを指定した場合、対応するスキーマによる妥当性検査に成功した文書には、挿入時にスキーマ情報の注釈が付けられます。対応するスキーマによる妥当性検査に失敗した文書が含まれている行はリジェクトされます。妥当性検査を正常に実行するには、インポートを起動するユーザーの特権に、少なくとも以下のいずれかの権限が含まれている必要があります。
    - SYSADM または 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. 暗黙的な非表示設定になっている Row Change Timestamp 列が含まれている表にインポートする場合は、その列の暗黙的な非表示のプロパティーが適用されません。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、インポート・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。

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。

- *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時に可変長ブロック・サイズを使用することは、現在サポートされていません。このオプションを使用しなければならない場合は、可変長ブロック・サイズで作成されたバックアップ・イメージを使用して正常なリカバリーができる、十分にテストされたプロシージャーを用意しておいてください。

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。最適パフォーマンスを得るには、バッファー・サイズを、使用する装置の最大ブロック・サイズ限度と等しくしてください。

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。サーバー上に保管されているデータでは、ファイル、テープ、または名前付きパイプの形式が使用可能です。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。

141 ページの『ロード・ユーティリティーのファイル・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。

- SYSADM
- DBADM
- データベースに対する LOAD 権限と以下の特権
  - 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・ポリシーの一部を成すセキュリティー・ラベルであること。
  - 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 に付与されているセキュリティー・ラベルであること。

  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。セッション許可 ID が、LBAC 資格情報により、データ内のロード対象の行を保護するセキュ

リティー・ラベルへの書き込みを許可されていない場合、その行はセキュリティー・ラベルによって保護されます。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。

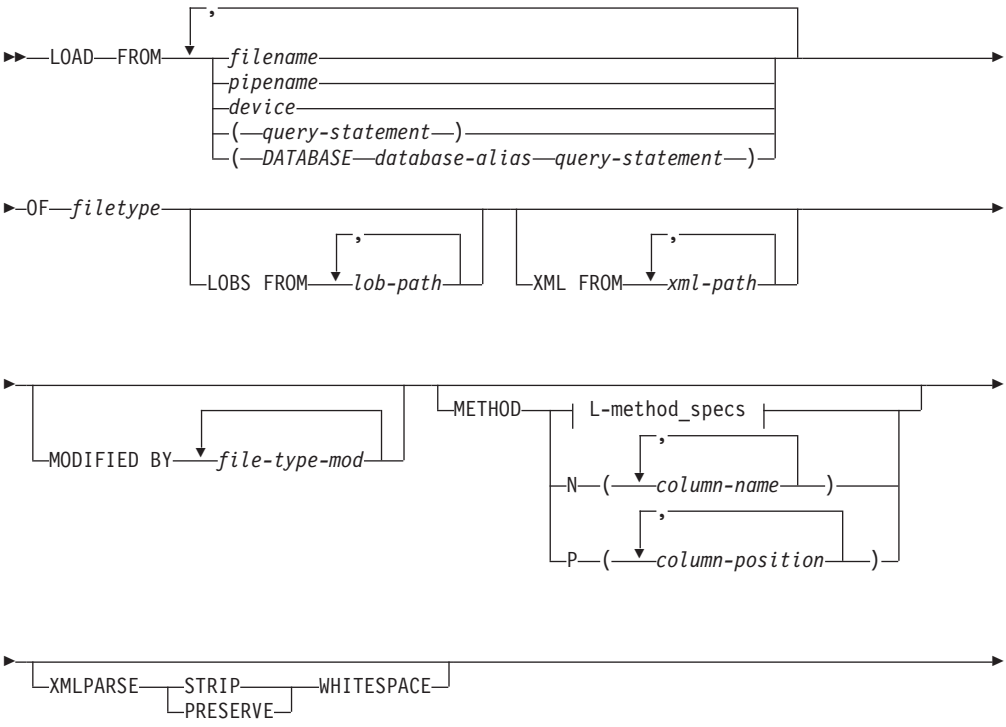
すべてのロード・プロセス (および一般にすべての DB2 サーバー・プロセス) はインスタンス所有者によって所有されており、それらのプロセスすべてにおいて、必要なファイルにアクセスするためにそのインスタンス所有者の ID を使用するため、インスタンス所有者には入力データ・ファイルに対する読み取りアクセス権が必要です。このコマンドをだれが呼び出すかには関係なく、それらの入力データ・ファイルをインスタンス所有者から読むことができなければなりません。

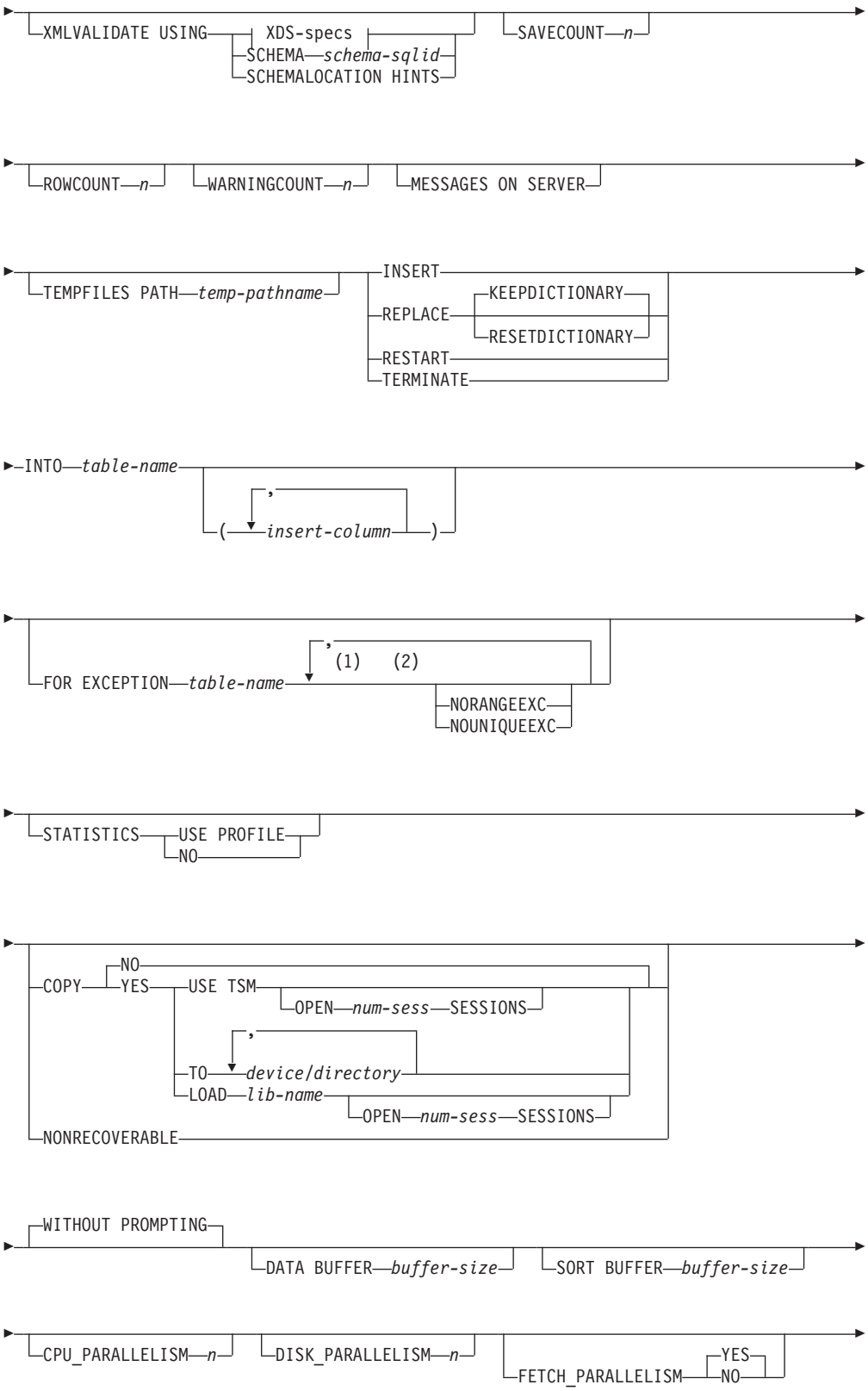
必要な接続

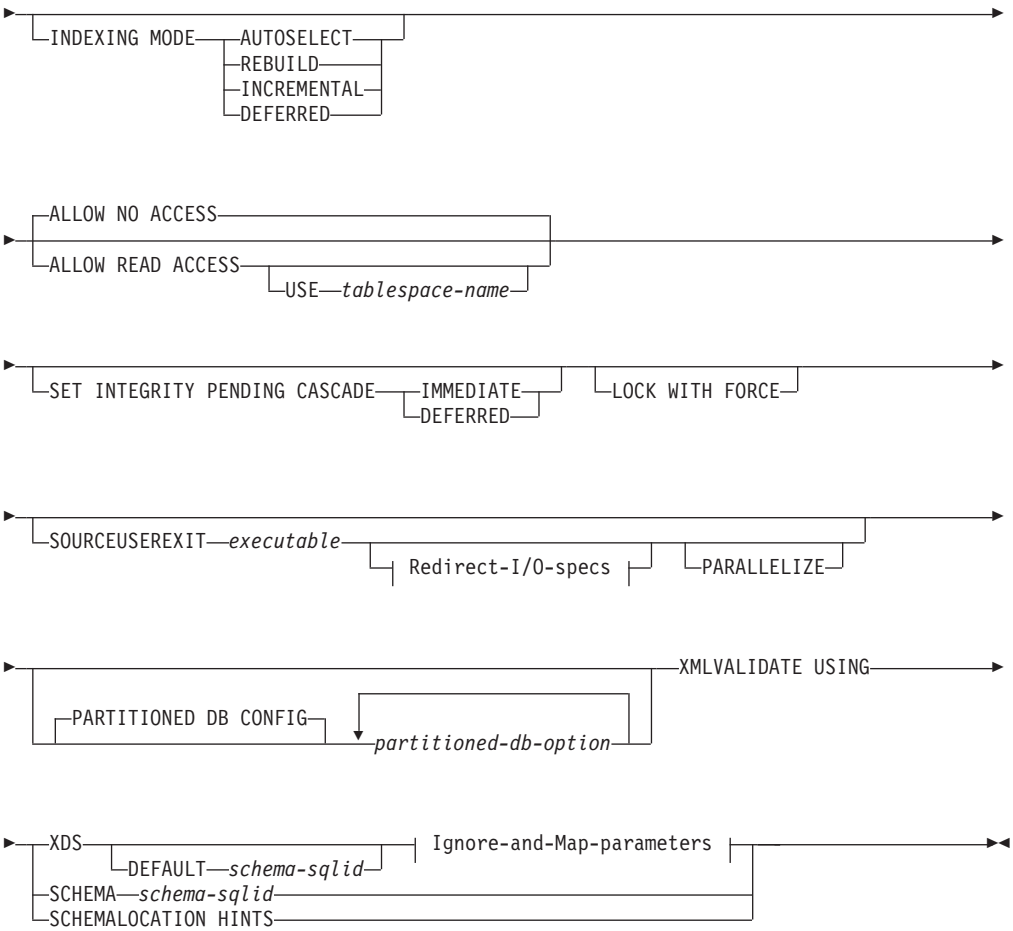
データベース。

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。

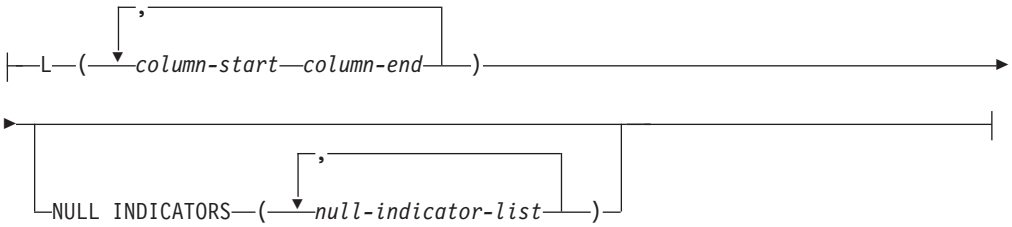
コマンド構文



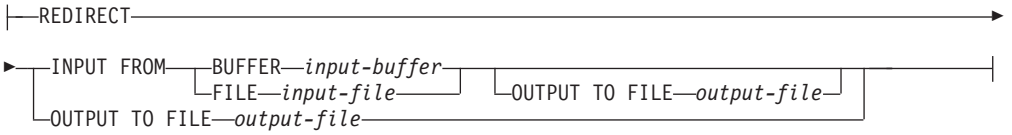

XMLVALIDATE USING
XDS-specs
SCHEMA schema-sqlid
SCHEMALOCATION HINTS
SAVECOUNT n
ROWCOUNT n
WARNINGCOUNT n
MESSAGES ON SERVER
TEMPFILES PATH temp-pathname
INSERT
REPLACE
KEEPDICTIONARY
RESETDICTIONARY
RESTART
TERMINATE
INTO table-name
(, insert-column)
FOR EXCEPTION table-name , (1) (2)
NORANGEEXC
NOUNIQUEEXC
STATISTICS
USE PROFILE
NO
COPY
NO
YES
USE TSM
OPEN num-sess SESSIONS
TO , device/directory
LOAD lib-name
OPEN num-sess SESSIONS
NONRECOVERABLE
WITHOUT PROMPTING
DATA BUFFER buffer-size
SORT BUFFER buffer-size
CPU_PARALLELISM n
DISK_PARALLELISM n
FETCH_PARALLELISM
YES
NO



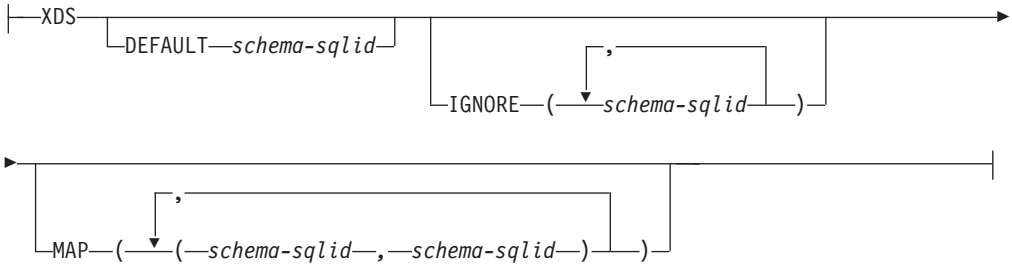
L-method-specs:



Redirect-I/O-specs:



XDS-specs:



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。以下に例を示します。

```
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 一度の 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。)

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 専用のバイナリー・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)。

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。 141 ページの『ロード・ユーティリティーのファイル・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、このオプションは許可されません。

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を誤って指定した場合、ロード・ユーティリティーは、ロードを試みた行ごとに警告を生成して、それがロードの失敗の原因になります。 *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・ファイルを保管することを指定します。戻される結果セットには、2 つの列が含まれます。1 つは MSG_RETRIEVAL で、この操作中に発生したすべての警告およびエラー・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、メッセージをクリーンアップするのに必要な 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。結果セット内の MSG_RETRIEVAL および MSG_REMOVAL 列には、NULL 値が入ります。

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 になっていると、新しく置換するデータは、ロードの呼び出し前に存在していたディクショナリーに基づく圧縮の対象になります。表にディクショナリーが存在していなかった場合は、表の COMPRESS 属性が YES になっている限り、置換によって表に挿入されるデータによって新しいディクショナリーが作成されます。この場合、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の量は、ADC のポリシーによって左右されます。そのデータは、圧縮されていない状態で表に取り込まれます。表にディクショナリーが挿入されると、その後にロードされる残りのデータは、そのディクショナリーによる圧縮の対象になります。これはデフォルトのパラメーターです。要約を以下の表 1 に示します。

次の例では、古いディクショナリーが表の中に存在する場合に、そのディクショナリーを維持します。

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

表 48. LOAD REPLACE KEEPDICTIONARY

圧縮	ディクショナリーが存在するかどうか	結果
Y	Y	ディクショナリーを保存します。すべての入力行が既存のディクショナリーによる圧縮の対象になります。
Y	N	十分なユーザー・データが存在する場合にのみ、新しいディクショナリーを表に挿入します。残りの行は、ディクショナリーの作成後に圧縮の対象になります。
N	Y	ディクショナリーを保存します。すべての入力行が圧縮されません。
N	N	影響はありません。すべての行が圧縮されません。

RESETDICTIONARY

表の COMPRESS 属性が YES の場合にこのディレクティブを指定

すると、LOAD REPLACE 処理の実行時に、表のデータ・オブジェクトに対応した新しいディクショナリーが作成されます。
COMPRESS 属性が NO で、表の中にディクショナリーが既に存在している場合は、そのディクショナリーが除去されるだけで、新しいディクショナリーが表に挿入されることはありません。コンプレッション・ディクショナリーは、1 つのユーザー・レコードだけでも作成できます。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指定した場合、ディクショナリーを作成するために必要なデータの量は、ADC のポリシーに左右されません。要約を以下の表 2 に示します。

次の例では、現在のディクショナリーをリセットして、新しい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 49. LOAD REPLACE RESETDICTIONARY

圧縮	ディクショナリーが存在するかどうか	結果
Y	Y	新しいディクショナリーを作成します。* ロードする残りの行は、ディクショナリーの作成後に圧縮の対象になります。
Y	N	新しいディクショナリーを作成します。残りの行は、ディクショナリーの作成後に圧縮の対象になります。
N	Y	ディクショナリーを除去します。すべての入力行が圧縮されません。
N	N	影響はありません。すべての行が圧縮されません。

* 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、表パーティションにロードするレコードがない場合は、新しいディクショナリーを作成できません。**RESETDICTIONARY** 操作では、既存のディクショナリーが維持されなくなります。

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を以下の表 3 に示します。

ロード TERMINATE オプションでは、表スペースのバックアップ・ペンディング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、システム表または宣言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。修飾子付き表名は、*schema.tablename* の形式です。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。以下に例を示します。

```
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。

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。

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
フォワード・リカバリーが使用可能 (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バックアップ・ペンディング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、表スペースをバックアップ・ペンディング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。バックアップ・ペンディング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 (全バックアップまたは表スペースのバックアップ) からのみ、リストアすることができます。

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バックアップ・ペンディング状態になりません。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るとロード操作は失敗し、表はロード・ペンディング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 ページ数を設定します (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 から、システムのメモリーをさらに使用できる場合は、LOAD コマンドの DATA BUFFER オプションの値が一時的に **util_heap_sz** を超えることができるようになりました。この場合、ユーティリティー・ヒープは *database_memory* 限界に達するまで必要に応じて動的に増加していきます。このメモリーは、ロード操作が完了すると解放されます。

値を指定しないと、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、表の状態 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。このオプションと一緒に使用できるのは、SYSTEM TEMPORARY 表スペースだけです。これを指定しないと、索引オブジェクトと同じ表スペースにシャドー索引が作成されます。索引オブジェクトと同じ表スペースにシャドー・コピーが作成される場合、シャドー索引オブジェクトは瞬間的に古い索引オブジェクトの上にコピーされます。シャドー・コピーが索引オブジェクトと別の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確保できます。

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 PENDING 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 PENDING 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

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 PENDING 状態になったことを示す警告 (SQLSTATE 01586) が発行されます。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
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
非推奨。

USING *directory*
非推奨。

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、表に挿入する文書を記述するために、XDS フィールドを使用してデータ・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' />"
```

第 1 行では、XML 文書が file1.xml というファイル名で指定されています。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。第 2 行では、XML 文書が file2.xml というファイル名で指定されています。その文書の開始点のバイト・オフセットは 23、長さは 45 バイトです。

例 2

ユーザーは、XML 列の構文解析や妥当性検査のオプションを指定しないでロード・コマンドを実行し、データのロードに成功します。

```
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。ユーザーには 2 つの表 T1 と T2 があり、それぞれは C1 という 1 つの XML 列だけで構成されています。T1 から T2 への LOAD を実行するために、ユーザーはまずカーソルを宣言します。

```
DECLARE X1 CURSOR FOR SELECT C1 FROM T1;
```

次に、ユーザーは、カーソル・タイプを使用して LOAD を実行します。

```
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。特定の順序にしたい場合には、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に対しては実行できません。
- 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。

- マルチパート PC/IXF ファイルの個々のパートを Windows システムから AIX システムにコピーするロード操作もサポートされています。すべてのファイルの名前を LOAD コマンドに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 のように記述します。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 を再開する場合の動作は、既存の動作と同じで、BUILD フェーズでは、索引の REBUILD モードの使用が強制されます。
- COPY YES オプションが使用されている場合、ロード・ユーティリティーはイメージ・ファイルの非常に大きなコピーを生成する場合があります。この動作は、ロードされる LOB/LF 列値ごとに LOAD コマンドがデータのバッファー全体をコピー・イメージ (約 128 KB) に書き出すときに行われることになっています。
- LOAD ユーティリティーは、fenced プロシージャーを参照する列を持つ表へのロードをサポートしていません。そのような表に LOAD コマンドを実行すると、エラー・メッセージ SQL1376N が返されます。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

TERMINATE ディレクティブの下で **LOAD** 処理を実行する場合のコンプレッション・ディクショナリー管理の動作を以下の表にまとめます。

表 50. LOAD TERMINATE のディクショナリー管理

表の COMPRESS 属性	LOAD の前にディクショナリーが存在するかどうか	TERMINATE: LOAD REPLACE KEEPDICTIONARY または LOAD INSERT	TERMINATE: LOAD REPLACE RESETDICTIONARY
YES	YES	既存のディクショナリーを維持します。	何も維持しません。
YES	NO	何も維持しません。	何も維持しません。
NO	YES	既存のディクショナリーを維持します。	何も維持しません。
NO	NO	何も実行しません。	何も実行しません。

LOAD RESTART は、到達した最後の整合点まで表を切り捨てます。最後の LOAD 整合点が取られた時点で表にコンプレッション・ディクショナリーが存在していた場合は、LOAD RESTART 処理によって、コンプレッション・ディクショナリーが表に配置されます。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。考えられる条件を以下の表 4 にまとめます。

表 51. LOAD RESTART のディクショナリー管理

表の COMPRESS 属性	LOAD の整合点の前にディクショナリーが存在するかどうか	RESTART: LOAD REPLACE KEEPDICTIONARY または LOAD INSERT	RESTART: LOAD REPLACE RESETDICTIONARY
YES	YES	既存のディクショナリーを維持します。	既存のディクショナリーを維持します。
YES	NO	ADC に基づいてディクショナリーを作成します。	ディクショナリーを作成します。
NO	YES	既存のディクショナリーを維持します。	既存のディクショナリーを除去します。
NO	NO	何も実行しません。	何も実行しません。

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。

- 表 52: ロード操作の結果セット。
- 140 ページの表 53: 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 52. 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数。この情報は、単一パーティション・データベースの場合に戻されます。複数パーティション・データベースの場合、この情報は、ターゲット表がパーティション化されていない場合にのみ戻されます。
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。

表 52. 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
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	すべてのパーティション・エージェントによってパーティション化される行の総数。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。複数パーティション・データベースの場合でも、ターゲット表がパーティション化されていない場合は、パーティショニング・エージェントが存在しないため、この情報は戻されません。
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。以下に例を示します。 `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。
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 53. 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。
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・ユーティリティーのファイル・タイプ修飾子

表 54. ロード・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
anyorder	この修飾子は、**cpu_parallelism** パラメーターと一緒に使用します。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。この修飾子を **generatedmissing** 修飾子または **generatedoverride** 修飾子と一緒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。この修飾子を **generatedignore** 修飾子または **generatedoverride** 修飾子と一緒に使用することはできません。

generatedoverride	この修飾子を指定すると、ロード・ユーティリティーは、表の中のすべての生成済み列でユーザー指定データを受け入れるようになります (この種の列の通常の規則とは反対の動作です)。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。この修飾子を使用すると、NULL 不可の生成済み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)。この修飾子を使用すると、表は Set Integrity Pending 状態になります。ユーザー指定値を検証しないで表の Set Integrity Pending 状態を解除する場合は、ロード操作の後に以下のコマンドを実行します。
	`SET INTEGRITY FOR table-name GENERATED COLUMN IMMEDIATE UNCHECKED`
	ユーザー指定値の検証を強制実行して表の Set Integrity Pending 状態を解除する場合は、ロード操作の後に以下のコマンドを実行します。
	`SET INTEGRITY FOR table-name IMMEDIATE CHECKED.`
	この修飾子を指定した場合、パーティション・キー、ディメンション・キー、分散キーのいずれかに生成済み列があれば、LOAD コマンドの実行時にその修飾子が自動的に **generatedignore** に変換され、ロードの処理が進められます。つまり、生成済み列のすべての値が再生成される結果になります。
	この修飾子を **generatedmissing** 修飾子または **generatedignore** 修飾子と一緒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。その結果、すべての IDENTITY 値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 の ID 列の場合も GENERATED BY DEFAULT の ID 列の場合も同じです。したがって、GENERATED ALWAYS 列の場合は、行がリジェクトされません。この修飾子を **identitymissing** 修飾子または **identityoverride** 修飾子と一緒に使用することはできません。

表 54. ロード・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
identitymissing	この修飾子を指定すると、ユーティリティーは、入力データ・ファイルに ID 列のデータが入っていない (NULL もない) という想定で動作し、各行の値を生成します。この動作は、GENERATED ALWAYS の ID 列の場合も GENERATED BY DEFAULT の ID 列の場合も同じです。この修飾子を **identityignore** 修飾子または **identityoverride** 修飾子と一緒に使用することはできません。
identityoverride	この修飾子を使用するのは、GENERATED ALWAYS として定義されている ID 列がロード対象の表に存在している場合に限られます。この修飾子を指定すると、ユーティリティーは、そのような列で明示的な非 NULL データを受け入れるようになります (この種の ID 列の通常の規則とは反対の動作です)。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)。この修飾子を **identitymissing** 修飾子または **identityignore** 修飾子と一緒に使用することはできません。このオプションを使用すると、ロード・ユーティリティーは、表の ID 列に入っている値の固有性を保持したり検証したりする操作を試行しなくなります。
indexfreespace=*x*	*x* は、0 から 99 までの (両端を含む) 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 ステートメントで指定されている PCTFREE 値よりも優先されます。 **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。

表 54. ロード・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* では、LOB データが含まれているファイル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・ファイルの LOB ロケーション指定子 (LLS) によって参照されている LOB が少なくとも 1 つ入っているファイルが 1 つ以上含まれています。LLS は、LOB ファイル・パスに格納されているファイルの LOB の位置を示したストリング表記です。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、データ・ファイルにストリング db2exp.001.123.456/ が格納されている場合は、ファイル db2exp.001 のオフセット 123 に LOB が配置されていて、その長さは 456 バイトということになります。 NULL LOB を指定する場合は、サイズとして -1 を入力します。サイズとして 0 を指定すると、長さ 0 の LOB として処理されます。長さ -1 の NULL LOB の場合は、オフセットとファイル名が無視されます。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 (該当するのは、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へのロード操作だけです)。 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) を使用する場合は、ファイルにヘッダーが存在するとは考えられません。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。
pagefreespace=*x*	*x* は、0 から 100 までの (両端を含む) 整数です。この値は、各データ・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。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。

表 54. ロード・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
rowchangetimestampignore	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、ROW CHANGE TIMESTAMP 列のデータを無視するようになります。その結果、すべての ROW CHANGE TIMESTAMP 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 の列の場合も GENERATED BY DEFAULT の列の場合も同じです。したがって、GENERATED ALWAYS 列の場合は、行がリジェクトされません。この修飾子を **rowchangetimestampmissing** 修飾子または **rowchangetimestampoverride** 修飾子と一緒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 の列の場合も GENERATED BY DEFAULT の列の場合も同じです。この修飾子を **rowchangetimestampignore** 修飾子または **rowchangetimestampoverride** 修飾子と一緒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。この修飾子を指定すると、ユーティリティーは、そのような列で明示的な非 NULL データを受け入れるようになります (この種の ROW CHANGE TIMESTAMP 列の通常の規則とは反対の動作です)。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、ROW CHANGE TIMESTAMP 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)。この修飾子を **rowchangetimestampmissing** 修飾子または **rowchangetimestampignore** 修飾子と一緒に使用することはできません。このオプションを使用すると、ロード・ユーティリティーは、表の ROW CHANGE TIMESTAMP 列に入っている値の固有性を保持したり検証したりする操作を試行しなくなります。

表 54. ロード・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・ファイルに含まれているセキュリティー・ラベルが、デフォルトのエンコード数値形式ではなく、ストリング・フォーマットのセキュリティー・ラベル値であることを指定します。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。ストリングが正しい形式になっていないと、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・ラベルに対応していなければ、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・ファイルに含まれているセキュリティー・ラベルが、デフォルトのエンコード数値形式ではなく、名前で示されていることを指定します。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。そのような動作を避けるには、**striptblanks** ファイル・タイプ修飾子を使用して、スペースを除去するようにします。

表 54. ロード・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
totalfreespace=*x*	*x* は、0 以上の整数です。この値は、表の合計ページ数に対する、フリー・スペースとして表の末尾に追加するページ数のパーセンテージとして解釈されます。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。表のデータ・ページの総数は、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ても、1 つ以上の行インスタンスでその列に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、明示的な欠落とは見なされず、NULL 列値の代わりにデフォルトが入ることもありません。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 55. ロード・ユーティリティーの有効なファイル・タイプ修飾子: ASCII ファイル形式 (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。ロード操作の実行時に、文字データ (および文字で指定されている数値データ) は、このコード・ページからデータベース・コード・ページに変換されます。 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 (両端を含む) 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 55. ロード・ユーティリティーの有効なファイル・タイプ修飾子: ASCII ファイル形式 (ASC/DEL) (続き)

修飾子	説明

dateformat=*"x"*	*x* は、ソース・ファイルの日付の形式です。[1] 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)`
	指定されていないそれぞれのエレメントには、デフォルト値の 1 が割り当てられます。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・パーティションに応じた) 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`
	このファイルは、インスタンス所有者によって作成され、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. 指定したファイルが既に存在している場合は、そのファイルは再作成されず切り捨てられます。

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、以下の場合に限られます。 1. **dumpfile** ファイル・タイプ修飾子と併用する場合 2. ユーザーがロードのターゲット表に対する SELECT 特権を持っている場合 3. UNIX オペレーティング・システムに配置されている DB2 サーバー・データベース・パーティションで実行する場合 指定したファイルがすでに存在している場合は、そのファイルの許可が変更なしでそのまま使用されます。

表 55. ロード・ユーティリティーの有効なファイル・タイプ修飾子: ASCII ファイル形式 (ASC/DEL) (続き)

修飾子	説明
fastparse	使用時には注意が必要です。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 DECIMAL(8,2) 列に `12345.00` としてではなく、`123.45` としてロードされます。 この修飾子を **packeddecimal** 修飾子と一緒に使用することはできません。

timeformat="*x***"**	*x* は、ソース・ファイルの時刻の形式です。[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)`
	
	指定されていないそれぞれのエレメントには、デフォルト値の 0 が割り当てられます。時刻形式の例を以下に示します。
	
	`"HH:MM:SS"`
	`"HH.MM TT"`
	`"SSSSS"`

表 55. ロード・ユーティリティーの有効なファイル・タイプ修飾子: ASCII ファイル形式 (ASC/DEL) (続き)

修飾子	説明
timestampformat=″*x*″	*x* は、ソース・ファイルのタイム・スタンプの形式です。[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) UUUUUU - マイクロ秒 (000000 から 999999 の範囲の 6 桁の数。 他のマイクロ秒エレメントとは相互に排他的) UUUUU - マイクロ秒 (00000 から 99999 の範囲の 5 桁の数。 000000 から 999990 の範囲にマップされる。 他のマイクロ秒エレメントとは相互に排他的) UUUU - マイクロ秒 (0000 から 9999 の範囲の 4 桁の数。 000000 から 999900 の範囲にマップされる。 他のマイクロ秒エレメントとは相互に排他的) UUU - マイクロ秒 (000 から 999 の範囲の 3 桁の数。 000000 から 999000 の範囲にマップされる。 他のマイクロ秒エレメントとは相互に排他的) UU - マイクロ秒 (00 から 99 の範囲の 2 桁の数。 000000 から 990000 の範囲にマップされる。 他のマイクロ秒エレメントとは相互に排他的) U - マイクロ秒 (0 から 9 の範囲の 1 桁の数。 000000 から 900000 の範囲にマップされる。 他のマイクロ秒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)

表 55. ロード・ユーティリティーの有効なファイル・タイプ修飾子: ASCII ファイル形式 (ASC/DEL) (続き)

修飾子	説明

timestampformat=*"x"* (続き)	指定されていない YYYY、M、MM、D、DD、DDD のいずれかのエレメントには、デフォルト値の 1 が割り当てられます。指定されていない MMM エレメントには、デフォルト値の 'Jan' が割り当てられます。指定されていない他のすべてのエレメントには、デフォルト値の 0 が割り当てられます。タイム・スタンプ形式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'、'dec' です。これらの値では、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 の間の区切り文字に基づいて形式を選択します。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。ブランク・スペースになっていれば、ロード・ユーティリティーは、HH と MM と SS の間を区切るためにコロン ':' を使用します。
	どちらの形式でも、マイクロ秒のフィールド (UUUUUU) を組み込むと、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`

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ればその修飾子によって、**codepage** 修飾子が指定されていなければデータベースのコード・ページによって、文字コード・ページを判別します。 ドロップ済み表のリカバリーで生成される区切り付きデータ・ファイルとこの修飾子を併用するのは、リカバリーする表にグラフィック・データが入っている場合に限られます。 **制約事項** EXPORT ユーティリティーで作成される DEL ファイルでは、**usegraphiccodepage** 修飾子を指定しないでください。そのファイルには、1 つのコード・ページでエンコードされたデータだけが入っているからです。**usegraphiccodepage** 修飾子は、ファイルに含まれている 2 バイト文字ラージ・オブジェクト (DBCLOB) でも無視されます。

表 55. ロード・ユーティリティーの有効なファイル・タイプ修飾子: ASCII ファイル形式 (ASC/DEL) (続き)

修飾子	説明
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。 それぞれの文書で、宣言タグが存在していて、エンコード属性が含まれている場合は、そのエンコードが文字コード・ページと一致している必要があります。そうでないと、その文書が含まれている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か、その修飾子が指定されていない場合はアプリケーション・コード・ページになります。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 文書が指定のグラフィック・コード・ページでエンコードされていることを指定します。 指定のグラフィック・コード・ページでエンコードされているものの、エンコード宣言が含まれていない XML 文書を処理するときに、このオプションは便利です。 それぞれの文書で、宣言タグが存在していて、エンコード属性が含まれている場合は、そのエンコードがグラフィック・コード・ページと一致している必要があります。そうでないと、その文書が含まれている行はリジェクトされます。グラフィック・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・コンポーネントか、その修飾子が指定されていない場合はアプリケーション・コード・ページのグラフィック・コンポーネントになります。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 56. ロード・ユーティリティーの有効なファイル・タイプ修飾子: ASC ファイル形式 (区切りなし ASCII)

修飾子	説明
binarynumerics	数値データ (DECIMAL ではない) は、文字表現ではなく 2 進形式でなければなりません。このようにして、コストのかかる変換を回避します。 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。 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、ターゲット列の定義と一致している必要があります。 • FLOAT は、IEEE 浮動小数点形式でなければなりません。 • ロード・ソース・ファイルのバイナリー・データは、ロード操作を実行するプラットフォームにかかわりなく、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。

表 56. ロード・ユーティリティーの有効なファイル・タイプ修飾子: ASC ファイル形式 (区切りなし ASCII) (続き)

修飾子	説明
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。

packeddecimal	パック 10 進数データを直接ロードします。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。
	
	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・プラットフォームにかかわりなく、ロード・ソース・ファイルのバイナリー・データのバイト・オーダーは、ビッグ・エンディアンであると見なされます。したがって、Windows オペレーティング・システムでこの修飾子を使用する場合は、バイト・オーダーを逆にしてはなりません。
	
	この修飾子を **implieddecimal** 修飾子と一緒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	可変長フィールドにデータをロードするときに、末尾ブランク・スペースを切り捨てます。このオプションを指定しなければ、ブランク・スペースは維持されます。 このオプションを **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、廃止オプションの **t** の代わりに用意されています。その廃止オプションは、旧バージョンとの互換性のためだけにサポートされています。
striptnulls	可変長フィールドにデータをロードするときに、末尾 NULL (0x00 文字) を切り捨てます。このオプションを指定しなければ、NULL は維持されます。 このオプションを **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、廃止オプションの **padwithzero** の代わりに用意されています。その廃止オプションは、旧バージョンとの互換性のためだけにサポートされています。

表 56. ロード・ユーティリティーの有効なファイル・タイプ修飾子: ASC ファイル形式 (区切りなし ASCII) (続き)

修飾子	説明

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 57. ロード・ユーティリティーの有効なファイル・タイプ修飾子: DEL ファイル形式 (区切り付き 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。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。[23] 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""`
	
	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。その場合は、以下のようにします。
	`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 (+) の代わりにブランク・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 進値の接頭部として正符号を使用します。
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。1 番目と 3 番目の <row delimiter> は、実際のレコード区切りとして解釈されますが、2 番目の <row delimiter> は、第 2 行の最初のデータ列の一部として解釈されるからです。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 57. ロード・ユーティリティーの有効なファイル・タイプ修飾子: DEL ファイル形式 (区切り付き ASCII) (続き)

修飾子	説明

keepblanks	タイプ CHAR、VARCHAR、LONG VARCHAR、CLOB の各フィールドで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、文字区切りの内側にない前後のすべてのブランクが除去され、表のすべてのブランク・フィールドに NULL が挿入されます。
	
	データ・ファイルにある前後のスペースをすべて保持して、TABLE1 という表にデータをロードする例を以下に示します。
	
	`db2 load from delfile3 of del`
	`  modified by keepblanks`
	`  insert into table1`

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。文字区切りも、列データの一部として解析されます。DB2 で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。正しくない使い方をすると、データが失われたり破損したりする可能性があります。 このオプションは、**chardelx**、 **delprioritychar**、および **nodoubledel** のいずれとも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 58. ロード・ユーティリティーの有効なファイル・タイプ修飾子: IXF ファイル形式

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。エレメントの開始位置と終了位置があいまいでない場合は、フィールド区切り文字はオプションになります。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・スタンプ形式の場合は、月の記述子と分の記述子の間であいまいさが残らないように注意する必要があります。どちらも、M という文字を使用するからです。月のフィールドは、他の日付フィールドと隣接している必要があります。分のフィールドは、他の時刻フィールドと隣接している必要があります。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)
```

あいまいな場合は、ユーティリティーによってエラー・メッセージが生成され、操作は失敗します。

あいまいでないタイム・スタンプ形式の例を以下に示します。

```
"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 # 文字を指定する場合は、以下のいずれかを使用します。

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

3. 区切り文字のオーバーライドとして使用できる文字に適用される制約事項については、『*データ移動のための区切り文字の制約事項*』を参照してください。
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. 暗黙的な非表示設定になっている Row Change Timestamp 列が含まれている表にインポートする場合は、その列の暗黙的な非表示のプロパティーが適用されません。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、インポート・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 59. codepage と usegraphiccodepage を使用する場合の LOAD の動作

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

表 59. codepage と usegraphiccodepage を使用する場合の LOAD の動作 (続き)

codepage=N	usegraphiccodepage	LOAD の動作
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。
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。グラフィック・データは、**CLIENT** オプションが指定されている場合でも、データベース・グラフィック・データのコード・ページのデータであるという想定になります。 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存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。

パーティション環境では、PRUNE HISTORY コマンドは実行されたデータベース・パーティションに対してのみ実行されます。複数のパーティションの履歴の整理は、個々のデータベース・パーティションそれぞれから PRUNE HISTORY コマンドを実行するか、db2_all 接頭部を使用してすべてのデータベース・パーティションで PRUNE HISTORY コマンドを実行するかのいずれかで行うことができます。

許可

以下のいずれか。

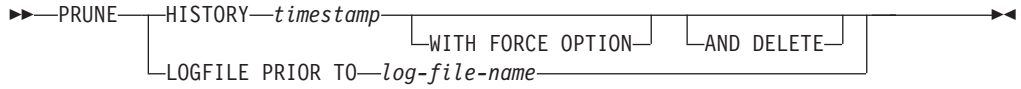
- *sysadm*
- *sysctrl*

- *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 に設定した場合に、AND DELETE パラメーターを指定して PRUNE HISTORY を呼び出すと、履歴ファイル項目が除去されていれば、バックアップ・イメージとロード・コピー・イメージも物理的に削除されます。

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こともあります。 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。

DB2HISTORY_STATUS_DO_NOT_DELETE 状況の項目は除去されません。WITH FORCE OPTION を使用すると、DB2HISTORY_STATUS_DO_NOT_DELETE のマークが設定されているオブジェクトも除去 (または削除) されます。リカバリー履歴ファイルの状況を DB2HISTORY_STATUS_DO_NOT_DELETE に設定するには、UPDATE HISTORY コマンドを使用するか、UPDATE_HISTORY を指定して ADMIN_CMD を使用するか、db2HistoryUpdate API を使用します。DB2HISTORY_STATUS_DO_NOT_DELETE 状況を使用すれば、重要なリカバリー履歴ファイル項目を除去してしまったり、関連するリカバリー・オブジェクトを削除してしまったりする事態を回避できます。

PRUNE HISTORY コマンドを使用すれば、スナップショット・バックアップ・データベース履歴ファイル項目を除去できますが、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。

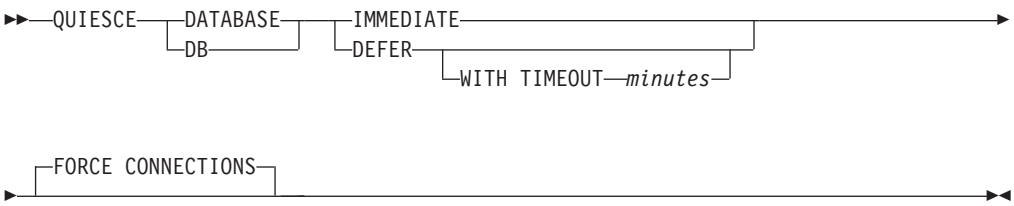
データベース・レベルの静止の場合:

- *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、すべてのユーザーがデータベースから強制的に切断されます。 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/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)

表の表スペースを静止させます。共用、更新意図、排他の 3 つの有効な静止モードがあります。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

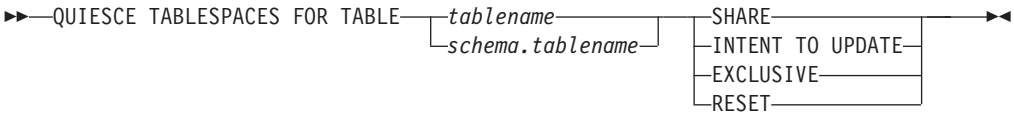
以下のいずれか。

- *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)

データベース・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。データの配分先は、均一 (デフォルト) にするか、またはシステム固有の必要を満たすためにユーザー指定にすることができます。

REDISTRIBUTE DATABASE PARTITION GROUP コマンドにより、データベース・パーティション・グループ内にあるすべてのパーティション間でデータが再配分されます。これにより、データベース・パーティション・グループ内にあるすべてのオブジェクトが影響を受けます。1 つのオブジェクトのみに制限できません。

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

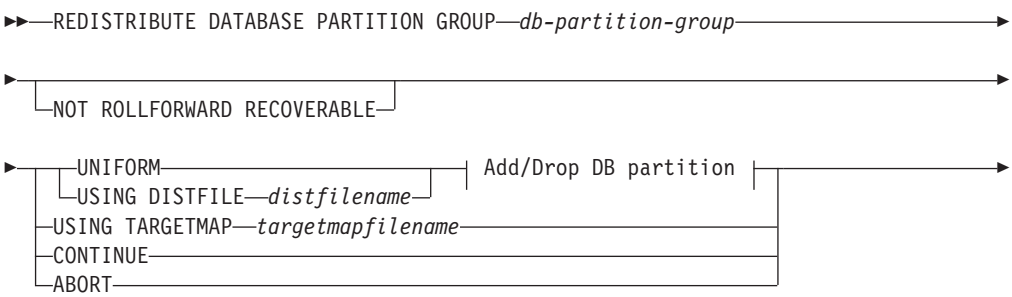
以下のいずれかの権限。

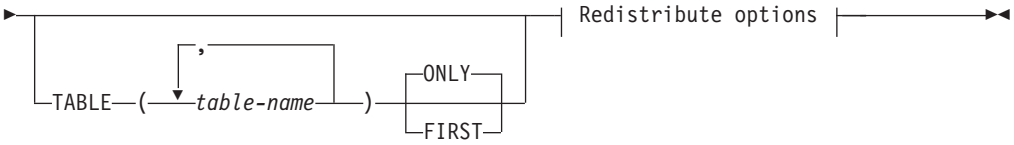
- SYSADM
- SYSCTRL (再配分されるデータベース・パーティション・グループ内のすべての表に対する DELETE、INSERT、および UPDATE 特権も必要)
- DBADM

必要な接続

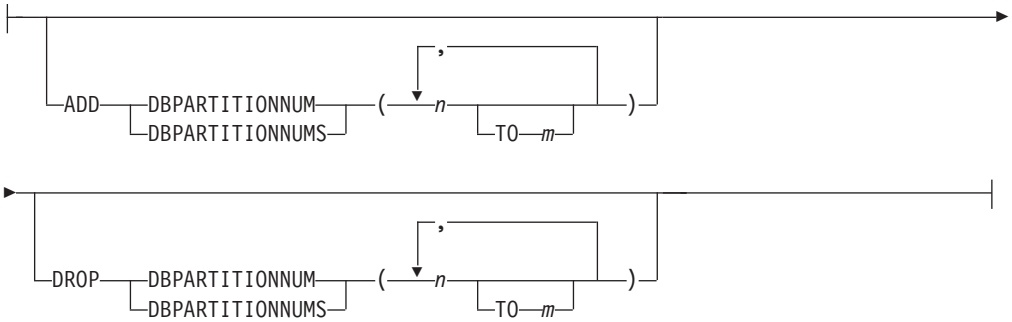
カタログ・パーティションとの接続。

コマンド構文

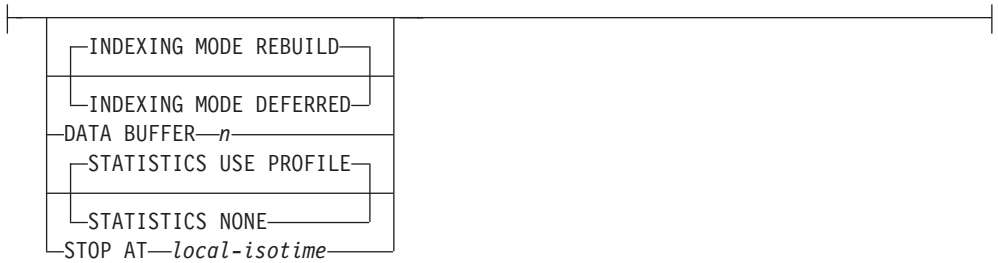




Add/Drop DB partition:



Redistribute options:



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、現在再配布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

このオプションは、DB2 9.5 フィックスパック 1 がインストールされている場合にのみ使用できます。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。

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・パーティションにわたって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、4096 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、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データをできるだけ均一に再配分するために使用します。再配分の後、データベース・パーティション・グループ中の各データベース・パーティションの重みは、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、データベース・パーティションにマップしたすべてのハッシュ・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、4096 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、4294967295 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の既存のパーティション内の対応する表スペース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*
再配分処理の表の順序を指定します。

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、再配分時の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

索引は、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れば、索引ページはディスク上で一緒にクラスター化されます。

DEFERRED

再配分では、索引の保守作業を実行しません。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 ページ数を設定します。指定された値がサポートされている最小値よりも低い場合には、最低の値が使用され、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 から、システムのメモリーをさらに使用できる場合は、REDISTRIBUTE DATABASE PARTITION GROUP コマンドの DATA BUFFER オプションの値が一時的に **util_heap_sz** を超えることができるようになりました。

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・グループを特定します。この資料では、再配分する必要のあるノード・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: 再配分されるデータベース・パーティション・グループにある表を使用する表書き込みイベント・モニターが自動的に活動化されている場合は、それらの表書き込みイベント・モニターを無効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・グループにそれらが定義されている場合は、(日付用に生成されたデータを維持する必要がない場合に) それらを再配分前にドロップし、再配分の完了後に再定義することを考慮できます。
   d. 表のアクセス・モードおよびロード状態: 再配分対象であるノード・グループのすべての表が、フル・アクセス・モードで、ロード・ペンディングや進行状態のロードを含まないことを確認します。

      ```
      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。その結果として、新しいノードでデータベース構成を更新した後に、REDISTRIBUTE コマンドを発行することが重要です。こうして、ウェアハウス全体にわたって構成を均衡に保つようにします。

```
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. 再配分処理の開始前に、データベース (すなわち、再配分されるノード・グループ内の表スペース) をバックアップします。こうして、最新のリカバリー・ポイントを確保しておきます
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 オプションの使用結果

REDISTRIBUTE DATABASE PARTITION GROUP コマンドが発行され、NOT ROLLFORWARD RECOVERABLE オプションが指定されると、移動される各行のログ・レコードの書き込みを最小化する、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。一方、この最小限のロギング・モデルの結果として、REDISTRIBUTE DATABASE PARTITION GROUP コマンドはロールフォワード・リカバリー可能ではない ということをユーザーが忘れないようにするのは重要です。つまり、再配分操作の途中でデータベースのロールフォワードが必要になるような操作を実行すると、再配分操作の対象になったすべての表は、UNAVAILABLE 状態のままになってしまいます。そのような表については、ドロップ操作しか実行できません。つまり、そのような表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に (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てですが)、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。その更新内容が失われてしまうのは、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、ロールフォワードのない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、イベント・ログ・ファイルが生成されます。そのファイルには、再配分操作に関する一般情報と、各表の処理の開始時刻と終了時刻などの情報が組み込まれます。このイベント・ログ・ファイルは、サーバーの以下の場所に書き込まれます。
  - 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。

  再配分イベント・ログの詳細については、『再配分イベント・ログ・ファイル』を参照してください。
- 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 で定義されている表が含まれているデータベース・パーティション・グループでは、再配分を実行できません。
- ユーザー TEMPORARY 表スペースと既存の宣言済み一時表とがデータベース・パーティション・グループに存在する場合、再配分は許可されません。
- **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。

互換性

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。別の方法として、範囲パーティション表上で個々の索引を再編成するというオプションもあります。

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。

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*
- 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

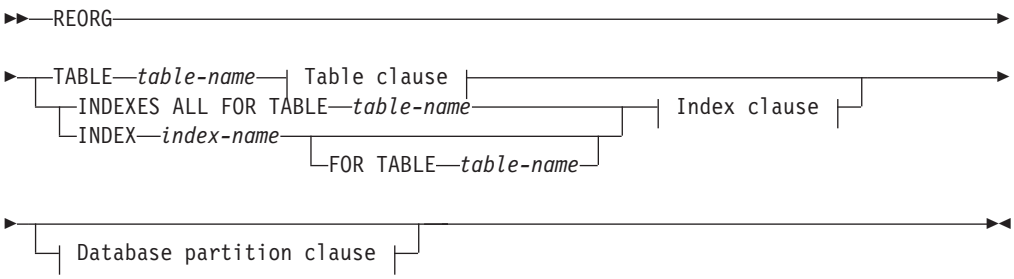
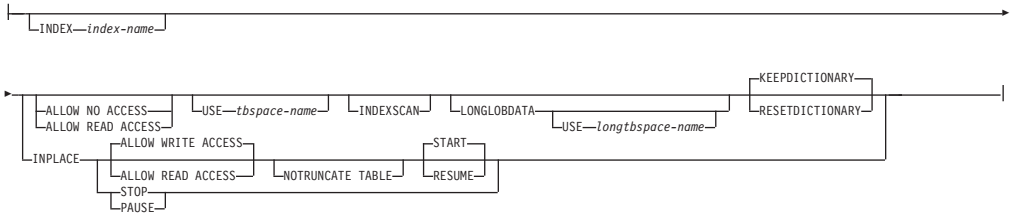
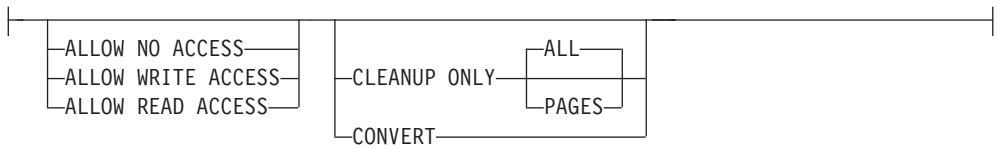


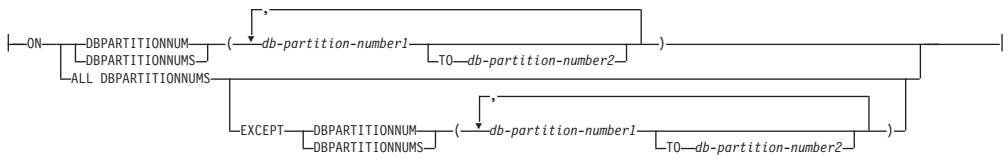
Table clause:



Index clause:



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*
パーティション表に対して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の場合のみ サポートされています。このパラメーターは、ブロック索引の場合はサポートされません。

FOR TABLE *table-name*

パーティション表について再編成する個別索引の表名のロケーションを指定します。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

索引の再編成の間は、他のユーザーはその表にアクセスできないことを指定します。

ALLOW READ ACCESS

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、読み取り専用のアクセスを行うことができることを指定します。

CLEANUP ONLY オプションを指定しない限り、パーティション表の REORG INDEXES では、このアクセス・レベルはサポートされません。

ALLOW WRITE ACCESS

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込んだ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るのでない限り、このアクセス・レベルは、マルチディメンション・クラスター (MDC) 表、パーティション表、拡張索引、または XML データ・タイプの列を含む表ではサポートされていません。

アクセス・モードの指定がない場合は、次のようにして、いずれか 1 つが自動的に選択されます。

表 60. コマンド、表タイプ、および、索引節に対して指定された追加のパラメーターに基づいて選択されるデフォルトの表アクセス

コマンド	表タイプ	索引節用に指定する追加のパラメーター	デフォルトのアクセス・モード
REORG INDEXES	非パーティション表	任意の値	ALLOW READ ACCESS
REORG INDEXES	パーティション表	指定なし	ALLOW NO ACCESS
REORG INDEXES	パーティション表	CLEANUP ONLY を指定	ALLOW READ ACCESS
REORG INDEX	パーティション表	任意の値	ALLOW READ ACCESS

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
作業中の表の索引がタイプ 1 かタイプ 2 かが分からない場合で、タイプ 2 の索引を使用したいときは、CONVERT オプションを使用できます。索引がタイプ 1 であれば、このオプションはそれをタイプ 2 に変換します。索引がすでにタイプ 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 8 よりも前の DB2 によって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 8 によって作成された索引は、すべてタイプ 2 の索引です。ただし、すでにタイプ 1 の索引を持つ表に作成した索引は例外です。この場合、新しい索引もタイプ 1 索引になります。

INSPECT コマンドを使用して索引のタイプを判別する手順は時間がかかることがあります。 CONVERT を使用すると、元のタイプを判別しなくても、新規の索引が必ずタイプ 2 となるようにすることができます。

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 *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。

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。

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。パーティション表を再編成する場合、これがデフォルトです。パーティション表の再編成はオフラインで実行されます。

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。これは、非パーティション表のデフォルトです。

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、タイプ 2 の索引を備えている一方で、拡張索引をもたず、しかも表内の XML 列に対して索引を定義されていない非パーティション表および非 MDC 表においてのみです。インプレース表再編成は、サイズが 3 ページ以上の表でのみ実行できます。

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インプレース表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。

8KB、16KB、または 32KB 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、表に含まれるすべてのデータ・パーティションの再編成において、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。そのためのスペース所要量は、表全体ではなく、表に含まれる最大のデータ・パーティションに等しくなります。

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 を読み取って行を取り出す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 オブジェクトのサイズが削減されます。

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* を指定する場合は、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、表のコンプレッション・ディクショナリーが存在しない場合に、このシナリオでディクショナリーが作成され、表が圧縮されるのは、表が一定のサイズ (1 から 2 MB 程度) で、その表に十分なデータが含まれている場合に限られます。一方、REORG RESETDICTIONARY を明示的に指定した場合は、表に少なくとも 1 つの行が含まれている限り、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。 LONG オブジェクト、LOB、索引、または XML オブジェクトを圧縮することはできません。

表 61. 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保存され、行は圧縮されます。
Y	N	ディクショナリーが構築され、行は圧縮されます。
N	Y	ディクショナリーは保存され、すべての行は圧縮解除されます。
N	N	何も効果はなく、すべての行は圧縮解除されます。

表のどのような再初期設定または切り捨て (置換操作の場合など) でも、表の圧縮属性が NO の場合にディクショナリーが存在すると、そのディクショナリーは廃棄されます。逆に、ディクショナリーが存在し、表の圧縮属性が YES の場合、切り捨てによってそのディクショナリーは保管され、廃棄されません。リカバリーを目的とするとともに、将来のデータ・キャプチャー変更 (つまり、レプリケーション) に備えて、ディクショナリー全体が記録されます。

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。再編成中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を使用して圧縮されます。それ以前のディクショナリーはすべて、このディクショナリーによって置き換えられます。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。 LONG オブジェクト、LOB、索引、または XML オブジェクトを圧縮することはできません。

表 62. 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	新規のディクショナリーが構築され(*)、行は圧縮されます。
Y	N	新規のディクショナリーが構築され、行は圧縮されます。
N	Y	ディクショナリーは除去され、すべての行は圧縮解除されます。
N	N	何も効果はなく、すべての行は圧縮解除されます。

* - 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。

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を実行す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パーティションを除き、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を実行す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で操作を実行します。

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、宣言一時表に対しては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。
- 表が再編成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。
- パーティション表の場合、
  - REORG は表レベルでサポートされています。個々のデータ・パーティションの再編成は、そのデータ・パーティションをデタッチし、その結果の非パーティション表を再編成した後、そのデータ・パーティションを再びアタッチすることによって実現できます。
  - 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。
  - 表の再編成中にエラーが発生した場合、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。索引は次回表にアクセスしたときに再作成されます。
  - ALLOW NONE アクセス・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、表の一部の非パーティション索引が無効のままになる場合があります。表の RID 索引については、エラー発生時点で再編成中である索引のみが無効のままになります。 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。無効のマークが付いたすべての索引は、次回表にアクセスしたときに再作成されます。
  - 表再編成操作が失敗した場合、再編成状態のデータ・パーティションとそうでないデータ・パーティションが混在している可能性があります。 REORG TABLE コマンドを再発行すると、データ・パーティションの再編成状態には関係なくすべてのデータ・パーティションが再編成されます。

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、表上の 1 つまたはすべての索引を再編成することができます。 索引の再編成によって、フラグメント化がすべて除去され、リーフ・ページへの物理クラスタリングのリストアが行われます。 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。しかし、インプレース表再編成 (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、再編成が失敗する可能性があります (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 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。

表の値圧縮を活動化または非活動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ない 場合、および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順序付けられます。

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。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で可能なかぎり多くのスペースを満たします。

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。

XML データの索引を再作成するには、REORG INDEXES/TABLE コマンドを使用できます。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)

ヘルス・インディケーターの設定を、特定のオブジェクトに関して、そのオブジェクト・タイプの現行のデフォルトにリセットするか、またはオブジェクト・タイプについて現行デフォルトのヘルス・インディケーターの設定を、インストール時のデフォルトにリセットします。

許可

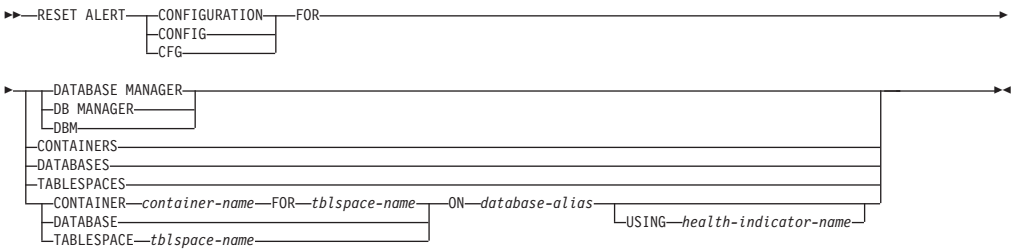
以下のいずれか。

- *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* 節を使って指定される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* 節を使って指定されるデータベースで、*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

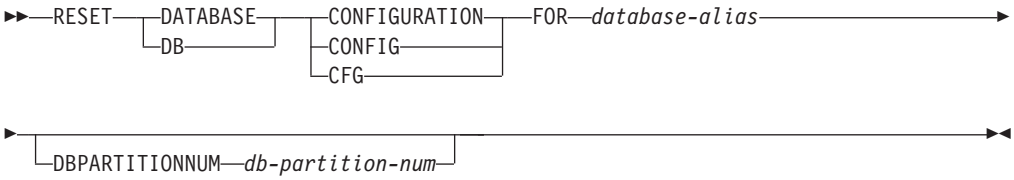
以下のいずれか。

- 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を特定のデータベース・パーティションだけに適用する場合は、このパラメーターを使用できます。このパラメーターを指定しなければ、すべてのデータベース・パーティションでリセットが有効になります。

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、記述されているデフォルト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。

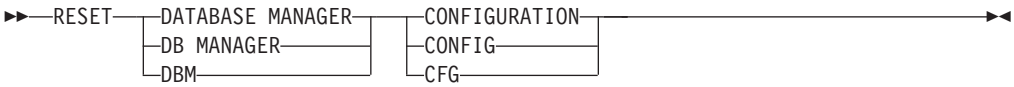
許可

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管します。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、リセットできません。これは構成ファイルを手動で編集して、適切なコマンドを使用しない場合に生じることがあります。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。

- *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・パスを決定する際に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。カタログ・データベース・パーティションのカタログを更新するのに使用します。

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
- 表に対する CONTROL 特権
- LOAD 権限

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グローバル一時表のいずれにおいても明示特権は必要ありません。

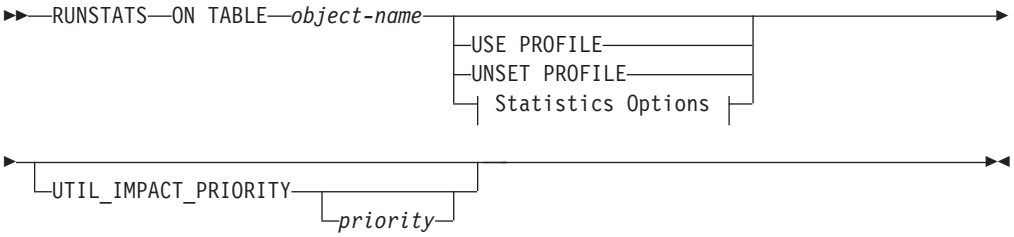
統計ビューの場合、以下のいずれか 1 つです。

- SYSADM
- SYSCTRL
- SYSMAINT
- DBADM
- 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:

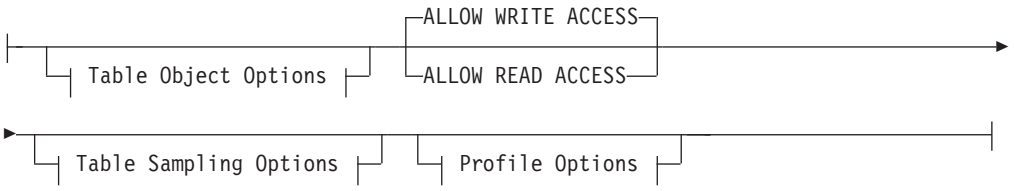


Table Object Options:

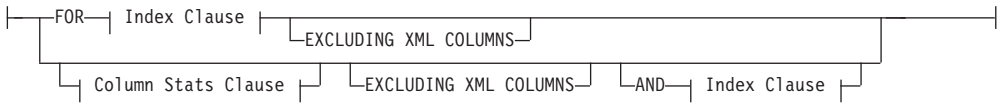
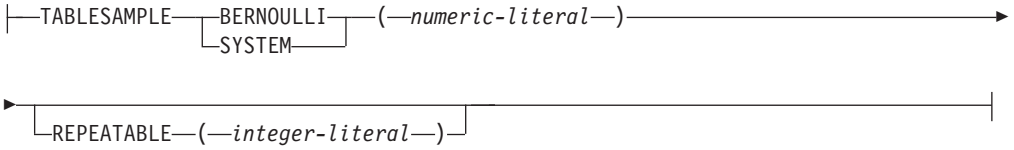
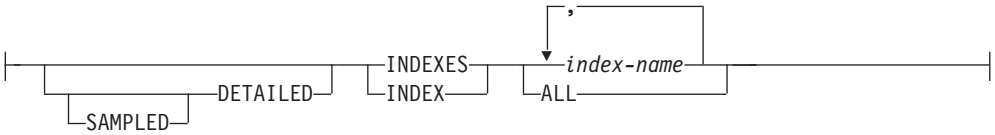


Table Sampling Options:

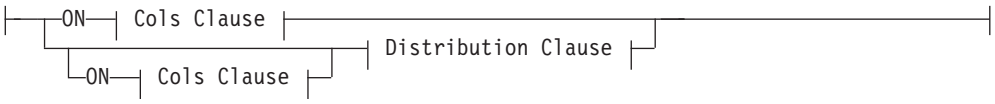


Profile Options:

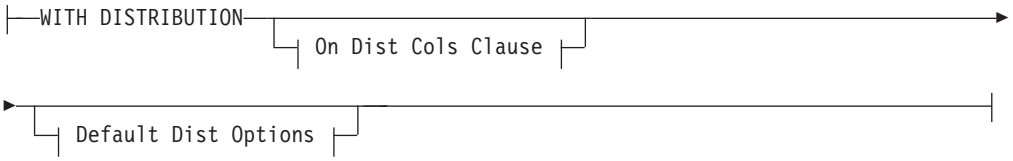
Index Clause:



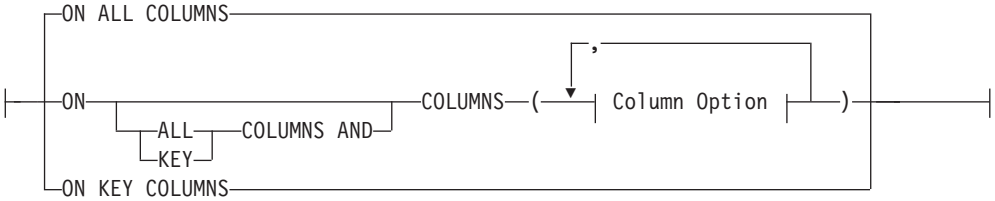
Column Stats Clause:



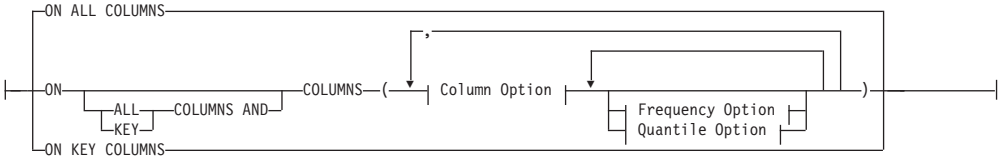
Distribution Clause:



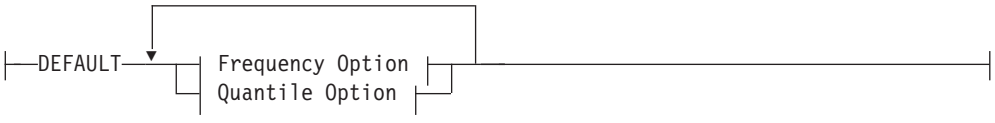
On Cols Clause:



On Dist Cols Clause:

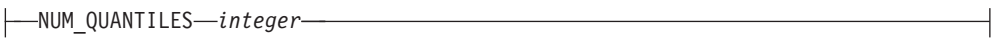


Default Dist Option:

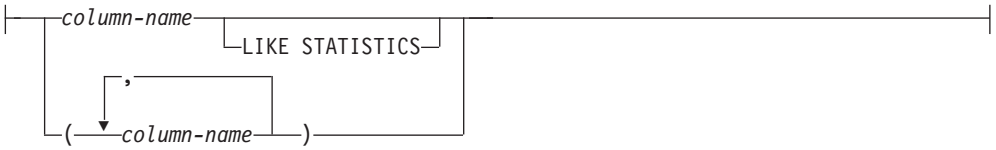


Frequency Option:

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で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、索引内のすべての項目で、拡張された索引統計を計算するかどうかが調べられます。ビューの場合、このオプションは使用できません。

ON ALL COLUMNS

一部の列で統計の収集を行えますが、収集を行えない列もあります。LONG VARCHAR 列や CLOB 列では統計を収集できません。収集が可能なすべての列で統計を収集することが希望の場合は、**ON ALL COLUMNS** 節を使用できます。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、すべての列では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、ユーザーは列オプション (Column Option) のリストを指定できます。列オプションは、統計を収集する単一列のリストおよび 1 つ以上の列グループ、またはそのいずれかとすることができます。列グループを指定するには、括弧で囲んだ列オプションのリストの中に、括弧で囲んだコンマ区切りの列名のリストを入れます。列グループを指定した場合は、そのグループの特色となるいくつかの値が収集されます。

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。

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、収集する変位値の最大数になります。

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、指定されたリストにある列でのみ (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頻度と変位値の統計の最大数が決まります。 **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。

多数の挿入、更新、または削除が並行して行われる表では、**ALLOW WRITE ACCESS** オプションはお勧めしません。 RUNSTATS コマンドは、まず表統計を実行してから、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。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トの行だけが、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。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トのデータ・ページだけが、検索されて統計収集に使用されます。統計ビューの場合、システム・サンプリングは、特定のクラスのビューに制限されます。単一の基本表かニックネームのいずれかにアクセスするビュー、あるいは参照保全関係を通じて結合された複数の基本表にアクセスするビューがあります。どちらの場合も、ビュー定義の中にローカル述部が含まれていてはなりません。システム・サンプリングのサポートが不可能なビューに対して、そのようなサンプリングが指定された場合、SQL20288N エラーが発生します。

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 未満の値は、DB2 データベース・システムではエラーとして処理されます (SQL1197N)。

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 コマンド・オプションを実行しなくても、RUNSTATS が、特定の統計プロファイルを生成してシステム・カタログ表に保管できるようにします。

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、SYSDATAPARTITIONS 表の中の状況フィールドをチェックして、値が I (索引クリーンアップ) かそれとも D (依存 MQT からデタッチされている) かを調べることができます。

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た場合。
   - 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. 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。
8. コマンド構文の On Dist Cols 節では、Column 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。
9. 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
10. **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。
11. 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。
12. 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、収集する頻度値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。

13. RUNSTATS は SYSTABLES に対して IX 表ロックを取得し、また、RUNSTATS 開始時には、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。それでも、U ロックのかかった行を含めて 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。

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

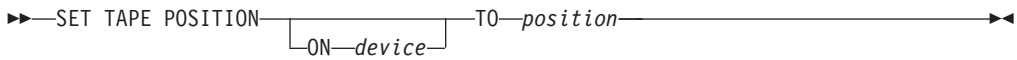
以下のいずれか。

- *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 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)

保守またはその他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、シャットダウンしたりデータベースを再開したりせずにユーザー・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。

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。

許可

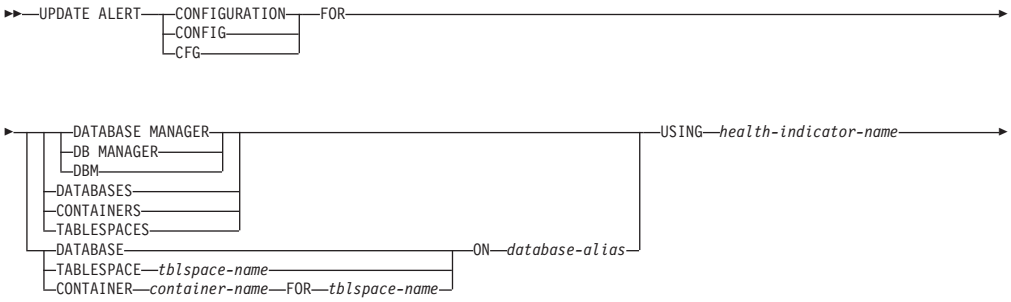
以下のいずれか。

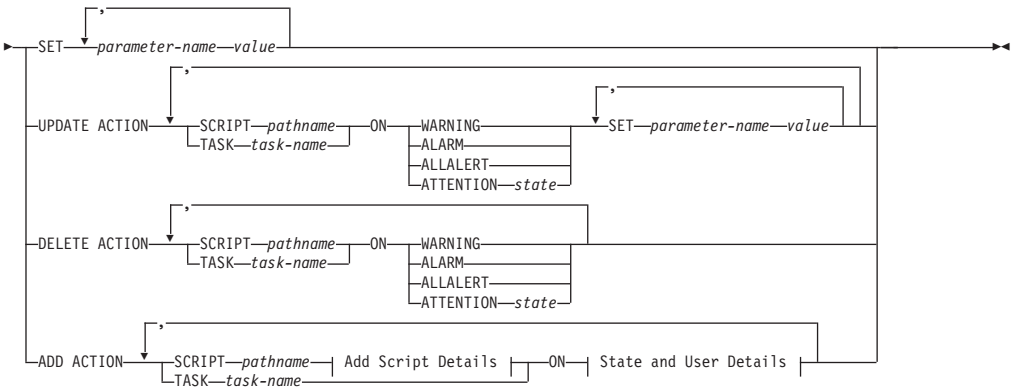
- *sysadm*
- *sysmaint*
- *sysctrl*

必要な接続

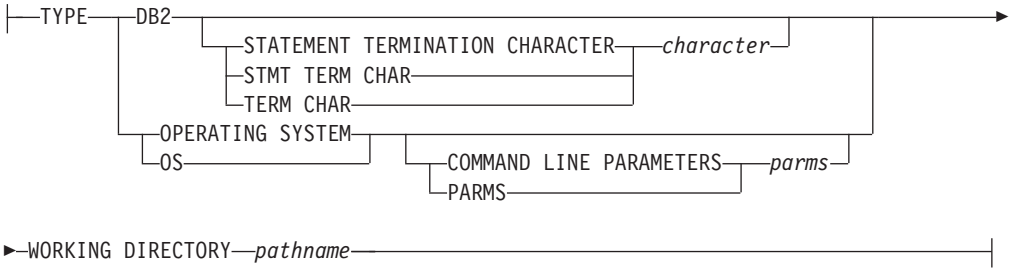
データベース

コマンド構文

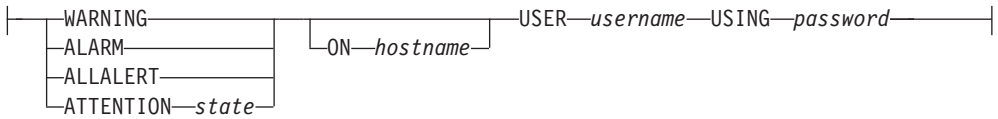




Add Script Details:



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* 節を使って指定される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* 節を使って指定されるデータベースで、*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。

個々の DB2 バージョンで使用可能なヘルス・インディケーター・ユニットのリストは、以下の照会を実行して収集することができます。

```
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.log、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、db2pd 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***]**
指定した条件で、*name* という名前の新規タスクが追加され、実行されるように指定します。

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、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*
ADD ACTION SCRIPT CLP コマンド・オプションの例として、データベース・ヘルス・インディケーターのいくつかの状態に対応した有効な数値を以下にまとめます。

- 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・プロシージャーから UPDATE ALERT CFG コマンドを呼び出す場合は、*state* として数値またはテキスト ID を使用することも可能です。その他のヘルス・インディケーターの例として、表スペース操作状況ヘルス・インディケーター (ts.ts_op_status) を取り上げると、その有効な数値とテキスト ID は、以下のとおりです。

- 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ヘルス・インディケーター (ts.ts_op_status) の場合は、OR 結合を使用して 1 つの数値で複数の状態を指定することも可能です。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。あるいは、3 つの別々の UPDATE ALERT CFG コマンド実行で、QUIESCED_SHARE、QUIESCED_UPDATE、QUIESCED_EXCLUSIVE をそれぞれ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 オプションの場合は、SQL ステートメント・テキストがキャプチャーされるさまざまな場所で、指定した *username* と *password* が公開されてしまう可能性があります。

- ネットワーク (ワイヤー上で username/password が暗号化されていない状態で渡される)
- db2diag.log
- 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 (特に、パッケージ・キャッシュとロック・タイムアウトのメカニズム)
- 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。

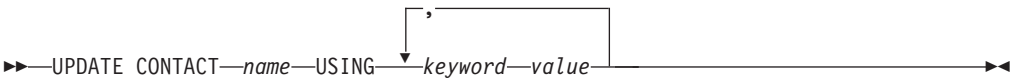
許可

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。

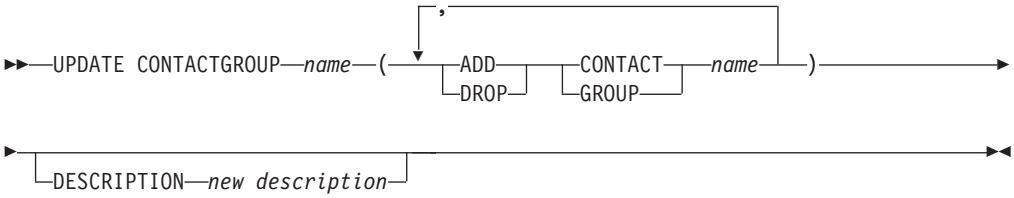
許可

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

DBPARTITIONNUM を指定して、1 つのデータベース・パーティションだけを更新する場合を除き、このコマンドを実行すると、デフォルトですべてのデータベース・パーティションが更新されます。

許可

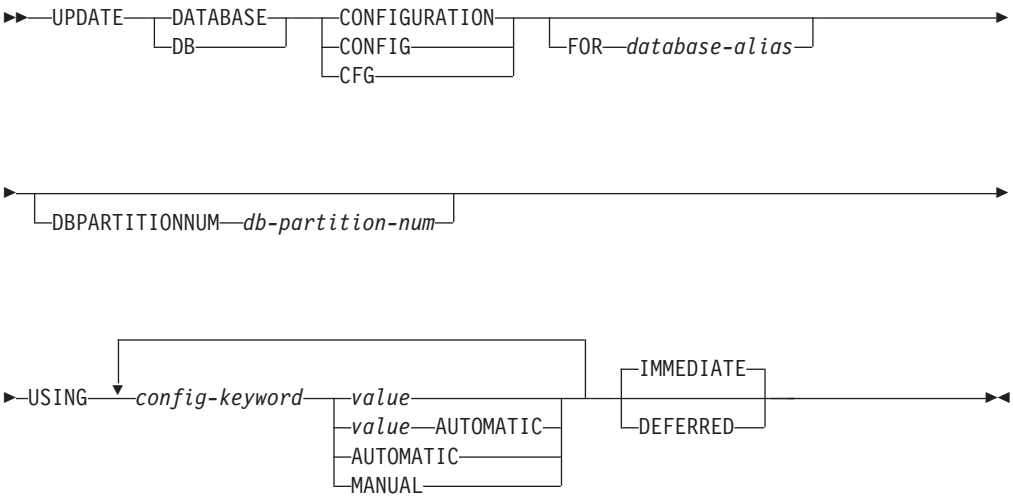
以下のいずれか。

- *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すで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・インスタンスのもとに置かれている別のデータベース用の構成ファイルを更新することができます。例えば、`db11` にのみ接続している場合に、`update db config for alias db22 using .... immediate` を発行すると、次のようになります。

- アクティブな接続が `db22` 上にない場合でも、更新する必要があるのは構成ファイルだけなので、更新は正常に完了します。新規の接続 (これによってデータベースが活動化されます) では、メモリー内で新規の変更が見られます。
- 他のアプリケーションからのアクティブな接続が `db22` 上にある場合、更新はディスク上では正常に完了しますが、メモリー内では正常に完了しません。データベースの再始動が必要なことを知らせる警告が出ます。

DBPARTITIONNUM *db-partition-num*
データベース構成の更新を特定のデータベース・パーティションだけに適用する場合は、このパラメーターを使用できます。このパラメーターを指定しなければ、すべてのデータベース・パーティションで更新が有効になります。

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。

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。

DBPARTITIONNUM を指定しない限り、このコマンドは、すべてのデータベース・パーティションに影響することになります。

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・パーティションだけでデータベース構成パラメーターを更新する場合は、以下のような方法があります。

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します。
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します。
4. データベース・パーティションから切断します。

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、1 つの特定パーティション (30) だけでロギング・モードを更新する場合は、以下のようなコマンドを使用します。

```
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。

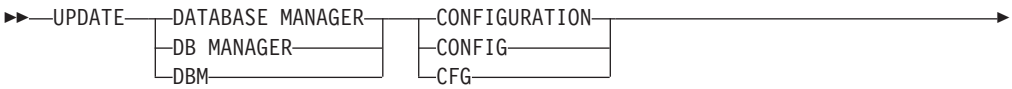
許可

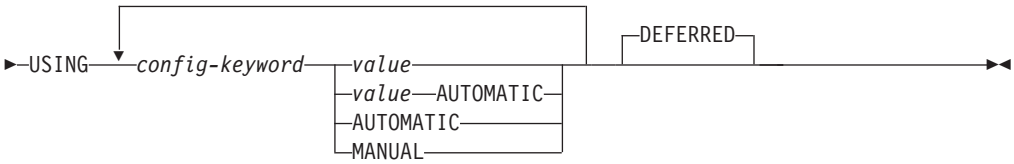
sysadm

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、インスタンスの再始動時に有効になります。これがデフォルトです。

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)

インスタンスが発行するヘルス・アラートについての通知に関して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・パーティションの履歴ファイル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

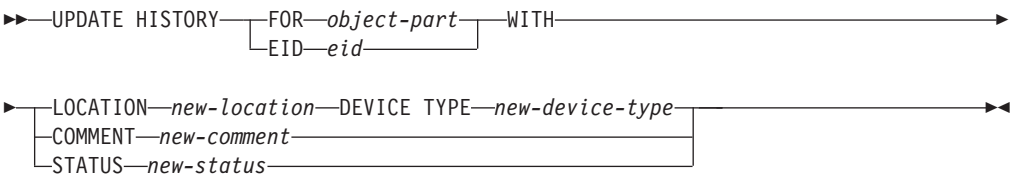
以下のいずれか。

- *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。

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。バックアップ項目のみがその状況を更新できます。有効な値は次のとおりです。

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。バックアップ・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いバックアップ・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 削除不能。DB2HISTORY_STATUS_DO_NOT_DELETE というマークが設定されているリカバリー履歴ファイル項目は、PRUNE HISTORY コマンドの呼び出しでも、PRUNE HISTORY を指定した ADMIN_CMD プロシージャーの実行でも、db2Prune API の呼び出しでも、リカバリー履歴ファイルの自動除去でも、除去されません。DB2HISTORY_STATUS_DO_NOT_DELETE 状況を使用すれば、重要なリカバリー・ファイル項目を除去したり、それらの項目に関連したリカバリー・オブジェクトを削除したりすることがないように保護できます。DB2HISTORY_STATUS_DO_NOT_DELETE のマークを設定できるのは、ログ・ファイル、バックアップ・イメージ、ロード・コピー・イメージに限られます。

例

1997 年 4 月 13 日午前 10 時 00 分に取った全データベース・バックアップの履歴ファイルを更新するには、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

データベース履歴ファイル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、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。

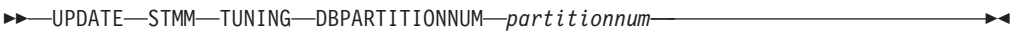
許可

SYSADM または DBADM 権限

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。

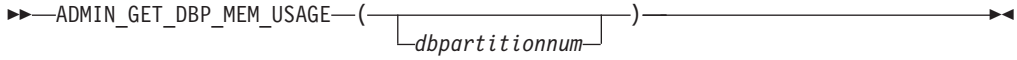
このコマンドは、その変更内容を ADMIN_CMD プロシージャーでコミットします。

ADMIN_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。データベース・パーティション・フィーチャー (DPF) を使用しているとき、*dbpartitionnum* に -1 または NULL 値を指定する場合、データは現在接続中のパーティションから戻されます。

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 63. 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。
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.
```

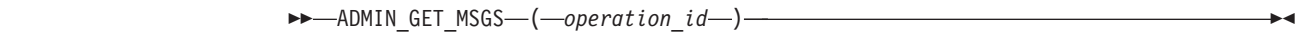
例 3: メモリー使用量統計をすべてのパーティションから取得します。

```
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.
```

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 64. 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。使用できる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。

ADMINTABCOMPRESSINFO ビューおよび ADMIN_GET_TAB_COMPRESS_INFO

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

ADMINTABCOMPRESSINFO 管理ビューに対する SELECT または CONTROL 特権、および ADMIN_GET_TAB_COMPRESS_INFO 表関数に対する EXECUTE 特権。

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。

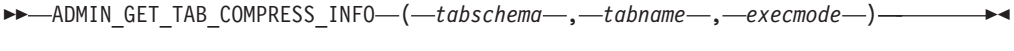
```
TABSCHEMA       TABNAME         DBPARTITIONNUM DATA_PARTITION_ID DICT_BUILDER                DICT_BUILD_TIMESTAMP
--------------- --------------- -------------- ----------------- -------------------------- --------------------------
SYSIBM          SYSTABLES                    0                 0 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、AADMINTABCOMPRESSINFO 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。

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。これがデフォルト値です。
- '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: DB2 の以前のバージョンからマイグレーションされた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 (例えば、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 65. 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
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。

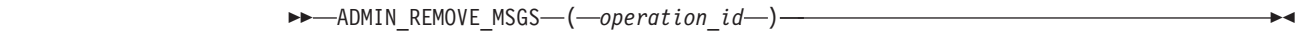
表 65. ADMINTABCOMPRESSINFO 管理ビューおよび ADMIN_GET_TAB_COMPRESS_INFO 表関数のメタデータ (続き)

列名	データ・タイプ	説明
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	圧縮により保管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	圧縮により保管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。

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。

ADMINTABINFO 管理ビューおよび ADMIN_GET_TAB_INFO_V95 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_V95 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_V95 表関数のメタデータ表を参照してください。

許可

ADMINTABINFO 管理ビューに対する SELECT または CONTROL 特権、および ADMIN_GET_TAB_INFO_V95 表関数に対する EXECUTE 特権。

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。

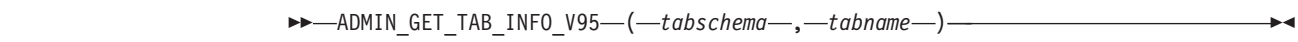
```
SELECT TABSCHEMA, TABNAME FROM SYSIBMADM.ADMINTABINFO
   WHERE INDEX_TYPE = 1
```

ADMIN_GET_TAB_INFO_V95 表関数

ADMIN_GET_TAB_INFO_V95 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_V95 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_V95 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_V95('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 (例えば索引や 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_V95( '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_V95( 'DBUSER1', 'EMPLOYEE' )) AS T
```

例 3: 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。

```
db2 => 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.
```

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_V95 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 (例えば、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_V95 表関数のメタデータ

表 66. ADMINTABINFO 管理ビューおよび ADMIN_GET_TAB_INFO_V95 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 (例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 66. ADMINTABINFO 管理ビューおよび ADMIN_GET_TAB_INFO_V95 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 (例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 (例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 (例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。

表 66. *ADMINTABINFO 管理ビューおよび ADMIN_GET_TAB_INFO_V95 表関数のメタデータ (続き)*

列名	データ・タイプ	説明
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 (例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)	表のインプレース表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

表 66. ADMINTABINFO 管理ビューおよび ADMIN_GET_TAB_INFO_V95 表関数のメタデータ (続き)

列名	データ・タイプ	説明
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 (例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」。
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。
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 66. ADMINTABINFO 管理ビューおよび ADMIN_GET_TAB_INFO_V95 表関数のメタデータ (続き)

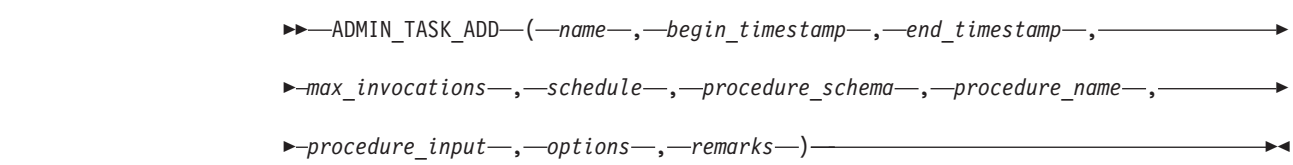
列名	データ・タイプ	説明
STATSTYPE	CHAR(1)	• 「F」= 表または索引のスキャンを行わずにシステムが作り上げた統計。これらの統計はメモリーに保管され、システム・カタログに保管されているものとは異なります。これは一時状態であり、最終的に完全な統計が DB2 によって収集され、システム・カタログに保管されます。 • 「A」= システムが非同期に収集した統計。統計は DB2 によりバックグラウンド・プロセスで自動的に収集され、システム・カタログに保管されました。 • 「S」= システムが同期に収集した統計。統計は DB2 によって SQL ステートメントのコンパイル中に自動的に収集されました。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

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。

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 67. 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。

表 67. UNIX cron 形式のフィールド名および値 (続き)

フィールド名	許容値
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 68. 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 69. 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

表 69. 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
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。
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。

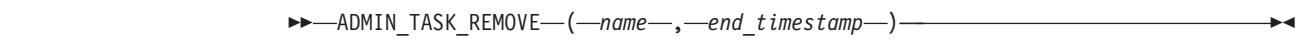
注:

- [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 70. 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。

表 70. 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
STATUS	VARCHAR(10)	タスクの状況。有効な値は次のとおりです。 • 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。

表 70. 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
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。
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。

表 70. ADMIN_TASK_STATUS 管理ビューによって戻される情報 (続き)

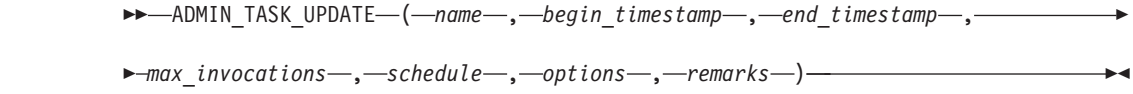
列名	データ・タイプ	説明
SQLERRMC	VARCHAR(70) FOR BIT DATA	SQLCA の SQLERRMC フィールドに現れるとおり、X'FF' で区切られた 1 つ以上のトークンが入ります。これらのトークンは、エラー条件の説明の中の変数を置き換えます。 STATUS が COMPLETED である場合、この値は、プロシージャーの呼び出しによって戻される SQLERRMC を示します。 STATUS が NOTRUN である場合、この値は、タスクを実行できない原因となった SQLERRMC のエラーを示します。 状況が RUNNING または UNKNOWN である場合、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。

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 特権とデータベースに対する SECADM 権限。

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
```

戻される情報

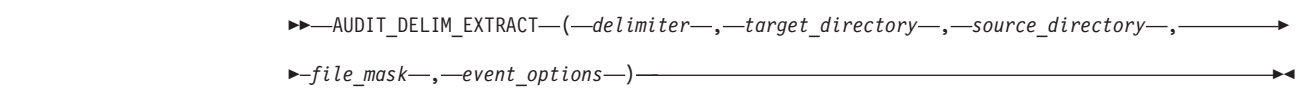
表 71. AUDIT_ARCHIVE 表関数によって戻される情報

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 および AUDIT_LIST_LOGS プロシージャーに対する EXECUTE 特権とデータベースに対する SECADM 権限。

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 特権とデータベースに対する SECADM 権限。

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 72. AUDIT_LIST_LOGS について戻される情報

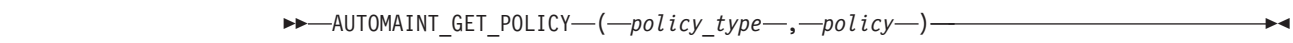
列名	データ・タイプ	説明
PATH	VARCHAR(1024)	アーカイブ対象ファイルのパス位置。
FILE	VARCHAR(1024)	アーカイブ対象ファイルのファイル名。
SIZE	BIGINT	アーカイブ対象ファイルのファイル・サイズ。

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。
- 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 登録変数および 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。

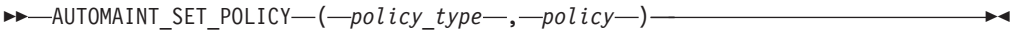
```
call sysproc.automaint_get_policyfile( 'AUTO_BACKUP', 'AutoBackup.xml' )
```

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。

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

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。

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、上記の例で示されるように、XML の内容を BLOB() スカラー関数の内部に渡すことができます。

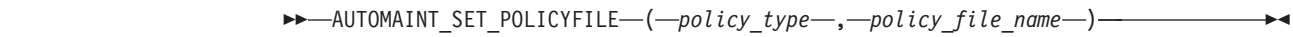
注: AUTOMAINT_SET_POLICY プロシージャーについては、単一のコンテキスト内での更新および読み取りは 2 回サポートされません。このプロシージャーが単一の「BEGIN ATOMIC」ブロック内で 2 回呼び出されると、データ競合が発生します。エラー (SQLCODE -1438, SQLSTATE 5U0ZZ) が、競合を引き起こしたステートメントに返されます。

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 登録変数および 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。

注: AUTOMAINT_SET_POLICYFILE プロシージャーについては、単一のコンテキスト内での更新および読み取りは 2 回サポートされません。このプロシージャーが単一の「BEGIN ATOMIC」ブロック内で 2 回呼び出されると、データ競合が発生します。エラー (SQLCODE -1438, SQLSTATE 5U0ZZ) が、競合を引き起こしたステートメントに返されます。

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 73. 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。
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

DBCFG 管理ビューに対する SELECT または CONTROL 特権、および DB_GET_CFG 表関数に対する EXECUTE 特権。

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 74. 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。有効な値は次のとおりです。 • 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。(データベースが活動化されていた場合には、非活動にしてから再活動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。

表 74. 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
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。有効な値は次のとおりです。 • 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。
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

DBMCFG 管理ビューに対する SELECT または CONTROL 特権、および DBM_GET_CFG 表関数に対する EXECUTE 特権。

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
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
...
```

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        ...
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
... 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 75. 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。有効な値は次のとおりです。 • 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。有効な値は次のとおりです。 • 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

REG_VARIABLES 管理ビューに対する SELECT または CONTROL 特権、および REG_LIST_VARIABLES 表関数に対する EXECUTE 特権。

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 76. 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。
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。

表 76. 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
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

環境ビュー

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

ENV_INST_INFO 管理ビューに対する SELECT または CONTROL 特権、および ENV_GET_INST_INFO 表関数に対する EXECUTE 特権。

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 77. 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。

表 77. ENV_INST_INFO 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
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

ENV_PROD_INFO 管理ビューに対する SELECT または CONTROL 特権、および ENV_GET_PROD_INFO_V95 表関数に対する EXECUTE 特権。

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 78. 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。
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_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

ENV_FEATURE_INFO 管理ビューに対する SELECT または CONTROL 特権、および ENV_GET_FEATURE_INFO 表関数に対する EXECUTE 特権。

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
```

ENV_FEATURE_INFO 管理ビューのメタデータ

表 79. 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。

表 79. *ENV_FEATURE_INFO 管理ビューのメタデータ (続き)*

列名	データ・タイプ	説明
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_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

ENV_SYS_INFO 管理ビューに対する SELECT または CONTROL 特権、および ENV_GET_SYS_INFO 表関数に対する EXECUTE 特権。

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 80. 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

ENV_SYS_RESOURCES 管理ビューに対する SELECT または CONTROL 特権、および ENV_GET_SYS_RESOURCES 表関数に対する EXECUTE 特権。

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 81. 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、表 82 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 82. 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 82. 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 82. 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	SMALLINT	マシンの全 CPU 使用量のパーセント。	すべて	パーセント

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。

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 83. 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。

表 83. HEALTH_CONT_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
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。

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 84. 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。

構文

```
►►—HEALTH_CONT_INFO—(—dbname—,—dbpartitionnum—)———————————————►◄
```

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 85. 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

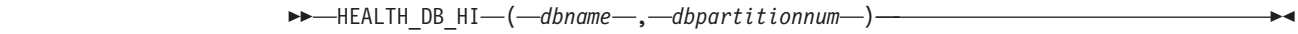
表 85. HEALTH_CONT_INFO 表関数によって戻される情報 (続き)

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
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。

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 86. 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。
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。

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 87. 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。

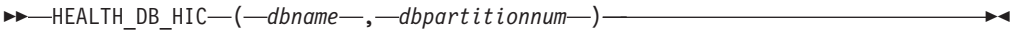
表 87. HEALTH_DB_HI_HIS 表関数によって戻される情報 (続き)

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_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。
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。

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 88. 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 88. 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。

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

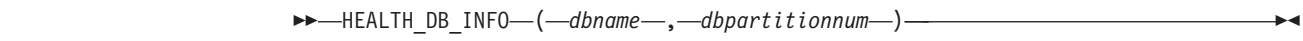
表 89. HEALTH_DB_HIC_HIS 表関数によって戻される情報

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。
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。

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 90. 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
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。

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)                   -
```

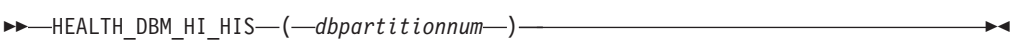
表 91. HEALTH_DBM_HI 表関数によって戻される情報

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。

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

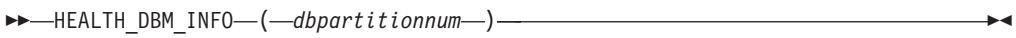
表 92. HEALTH_DBM_HI_HIS 表関数によって戻される情報

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。
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。

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 93. 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。

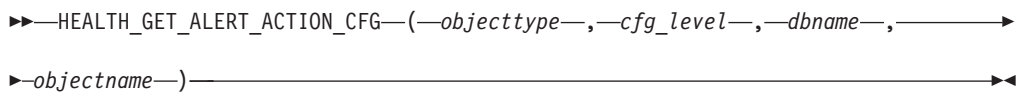
表 93. HEALTH_DBM_INFO 表関数によって戻される情報 (続き)

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
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 表関数 - ヘルス・アラート・アクション構成設定値の検索

HEALTH_GET_ALERT_ACTION_CFG 表関数は、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。

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 94. 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。
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'。

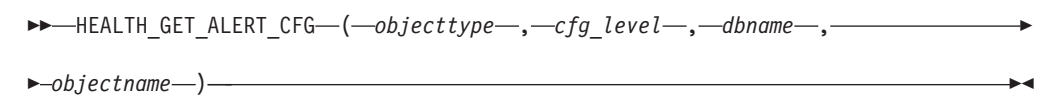
表 94. HEALTH_GET_ALERT_ACTION_CFG 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明
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 表関数 - ヘルス・アラート構成設定値の検索

HEALTH_GET_ALERT_CFG 表関数は、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。

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         O         SAMPLE                                 ...
DB         O         SAMPLE                                 ...
DB         O         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 95. 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。
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。

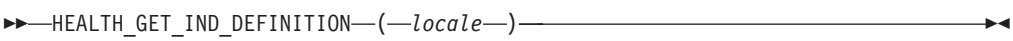
表 95. HEALTH_GET_ALERT_CFG 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明
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 表関数 - ヘルス・インディケーター定義の検索

HEALTH_GET_IND_DEFINITION 表関数は、ヘルス・インディケーター定義を戻します。

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

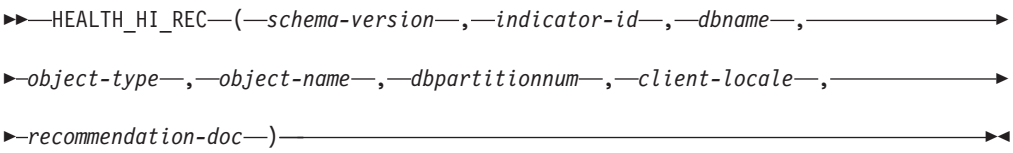
表 96. HEALTH_GET_IND_DEFINITION 表関数によって戻される情報

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。
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。

表 96. 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
TYPE	VARCHAR(16)	ヘルス・インディケーター・タイプ。使用できる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



スキーマは SYSPROC です。

HEALTH_HI_REC プロシージャーは、特定の DB2 オブジェクトで、アラート状態のヘルス・インディケーターに関する一連の推奨事項を検索します。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。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)。

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。上記の場合以外は、NULL を指定してください。

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。

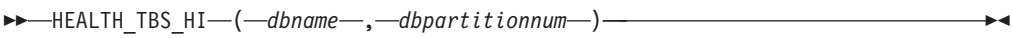
recommendation-doc
DB2 Health Recommendation スキーマ定義 (sqllib ディレクトリーの misc サブディレクトリーにある、XML スキーマ DB2RecommendationSchema.xsd を参照) に従ってフォーマットされた推奨事項文書 (XML) を含む、タイプ BLOB(2M) の出力引数。メッセージがターゲット・インスタンスでの呼び出し

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。

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               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 97. 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。
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。

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 98. 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。
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。

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 99. 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

表 99. HEALTH_TBS_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
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。

MQSeries ルーチン

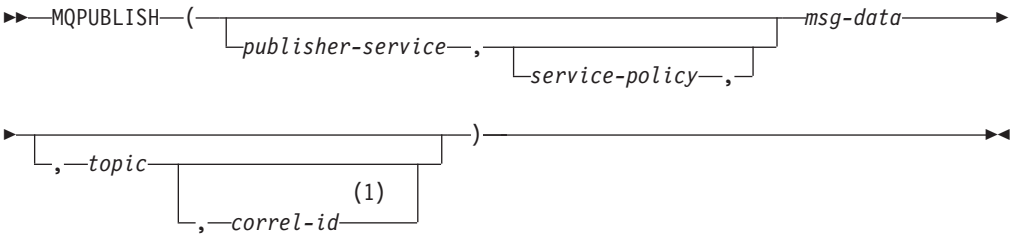
MQPUBLISH

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。

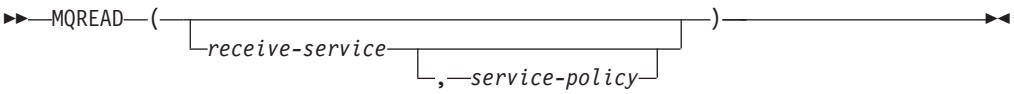
```
VALUES MQPUBLISH('Testing 901','TESTS')
```

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')
```

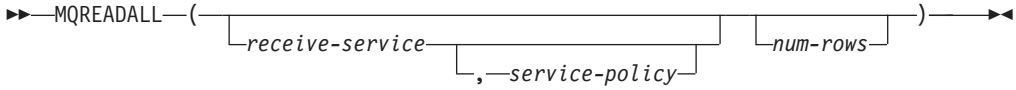
例 3: この例では、サービス "MYSERVICE" に指定されたキューの先頭にあるメッセージを、ポリシー "MYPOLICY" を使用して読み取ります。

```
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 100. 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)。

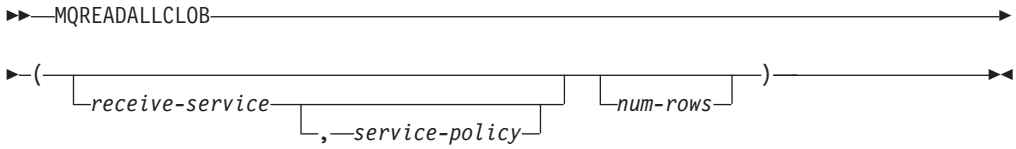
表 100. MQREADALL 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明
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。

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 101. 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。

表 101. MQREADALLCLOB 表関数によって戻される情報 (続き)

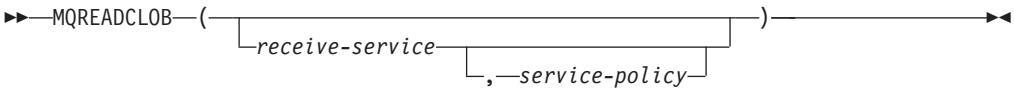
列名	データ・タイプ	説明
TOPIC	VARCHAR(40)	メッセージがパブリッシュされたときのトピックを含みます (使用可能な場合)。
QNAME	VARCHAR(48)	メッセージが受信されたキューの名前を含みます。
MSGID	CHAR(24)	このメッセージの割り当てられた固有の MQSeries ID を含みます。
MSGFORMAT	VARCHAR(8)	MQSeries によって定義された、メッセージの形式を含みます。通常、ストリングの形式は MQSTR です。

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。

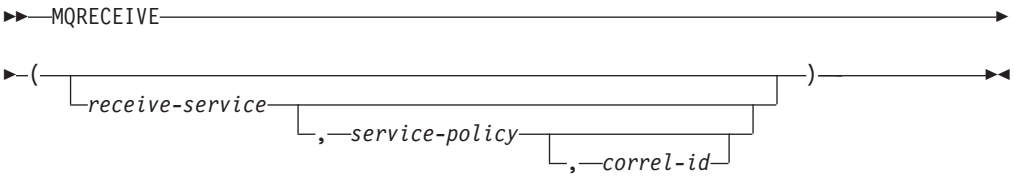
```
VALUES MQREADCLOB('MYSERVICE','MYPOLICY')
```

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')
```

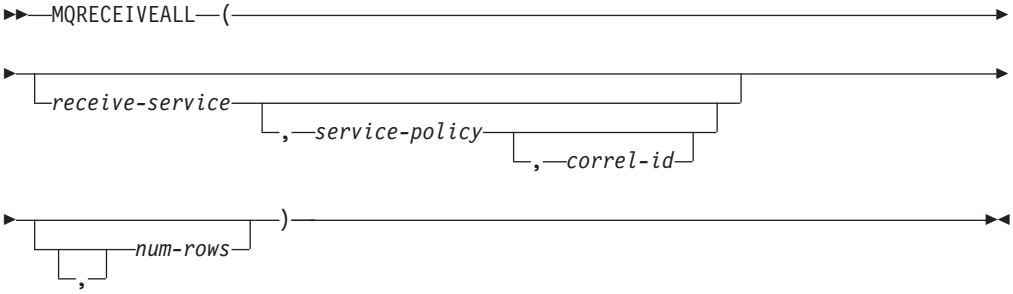
例 4: この例では、相関 ID '1234' に一致する最初のメッセージを、サービス "MYSERVICE" に指定されたキューの先頭から検索して受信します。その際、ポリシー "MYPOLICY" を使用します。

```
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 102. 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。

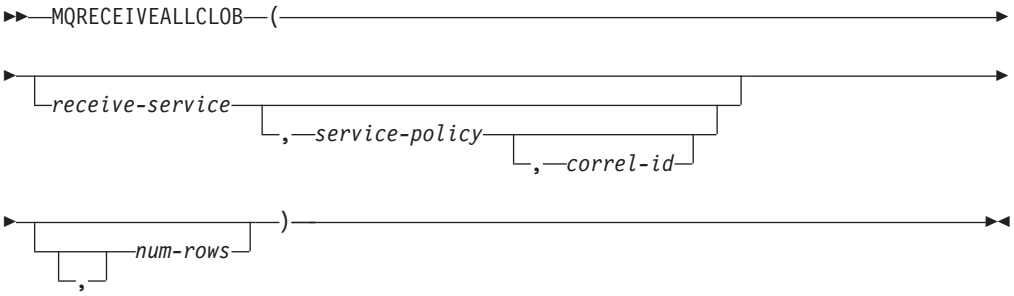
表 102. MQRECEIVEALL 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明
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。

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 103. 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 103. MQRECEIVEALLCLOB 表関数によって戻される情報 (続き)

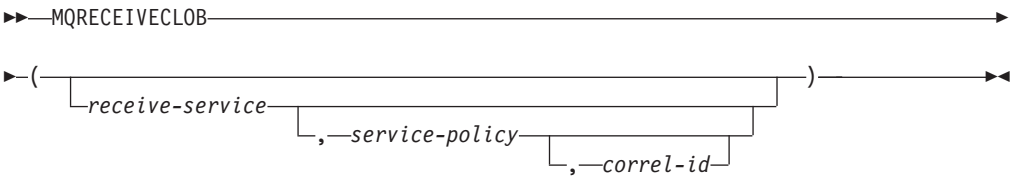
列名	データ・タイプ	説明
CORRELID	VARCHAR(24)	メッセージの識別に使用できる相関 ID を含みます。この ID を使用して、キューから特定メッセージを選択できます。要求と応答のシナリオでは、相関 ID を使用すると、応答を特定の要求と関連付けられるようになります。
TOPIC	VARCHAR(40)	メッセージがパブリッシュされたときのトピックを含みます (使用可能な場合)。
QNAME	VARCHAR(48)	メッセージが受信されたキューの名前を含みます。
MSGID	CHAR(24)	このメッセージの割り当てられた固有の MQSeries ID を含みます。
MSGFORMAT	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。

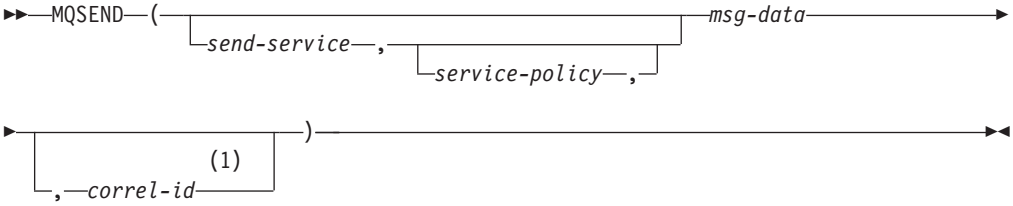
```
VALUES MQRECEIVECLOB('MYSERVICE',MYPOLICY','1234')
```

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。

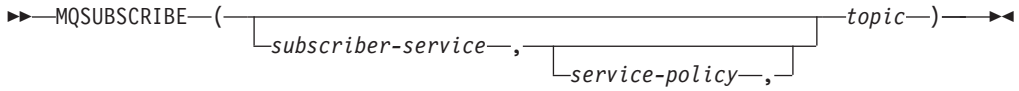
```
VALUES MQSEND('MYSERVICE','Testing 901')
```

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。

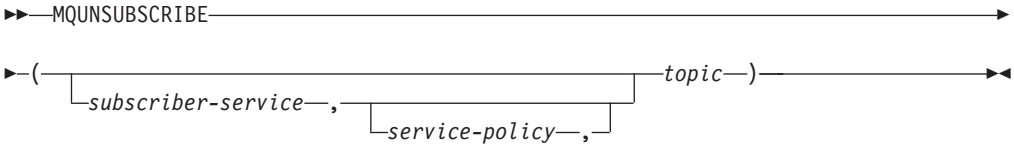
```
VALUES MQSUBSCRIBE('PORTFOLIO-UPDATES','BASIC-POLICY','Stocks')
```

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。

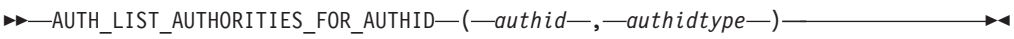
```
VALUES MQUNSUBSCRIBE('PORTFOLIO-UPDATES','BASIC-POLICY','Stocks')
```

セキュリティー・ルーチンおよびビュー

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 104. 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

表 104. 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
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
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
------------------------ ------ ------- -------- --------- ---------- ----------- ------
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           *
DBADM                    Y      Y       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           *
SYSADM                   *      Y       *        *         *          *           *
SYSCTRL                  *      Y       *        *         *          *           *
SYSMAINT                 *      Y       *        *         *          *           *
SYSMON                   *      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
------------------------ ------ ------- -------- --------- ---------- ----------- ------
BINDADD                  *      N       *        *         N          *           *
CONNECT                  *      N       *        *         N          *           *
CREATE_EXTERNAL_ROUTINE  *      N       *        *         N          *           *
CREATE_NOT_FENCED_ROUTINE *     N       *        *         Y          *           *
CREATETAB                *      N       *        *         N          *           *
DBADM                    *      N       *        *         N          *           *
IMPLICIT_SCHEMA          *      N       *        *         N          *           *
LOAD                     *      N       *        *         N          *           *
QUIESCE_CONNECT          *      N       *        *         N          *           *
SECADM                   *      N       *        *         N          *           *
SYSADM                   *      Y       *        *         *          *           *
SYSCTRL                  *      Y       *        *         *          *           *
SYSMAINT                 *      Y       *        *         *          *           *
SYSMON                   *      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
------------------------ ------ ------- -------- --------- ---------- ----------- ------
BINDADD                  *      *       Y        *         *          N           *
```

```
CONNECT                   *      *       Y       *       *          N          *
CREATE_EXTERNAL_ROUTINE   *      *       N       *       *          N          *
CREATE_NOT_FENCED_ROUTINE *      *       N       *       *          N          *
CREATETAB                 *      *       Y       *       *          N          *
DBADM                     *      *       N       *       *          N          *
IMPLICIT_SCHEMA           *      *       Y       *       *          N          *
LOAD                      *      *       N       *       *          N          *
QUIESCE_CONNECT           *      *       N       *       *          N          *
SECADM                    *      *       N       *       *          N          *
SYSADM                    *      *       *       *       *          *          *
SYSCTRL                   *      *       *       *       *          *          *
SYSMAINT                  *      *       *       *       *          *          *
SYSMON                    *      *       *       *       *          *          *
```

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
------------------------- ------ ------- -------- --------- ---------- ----------- ------
BINDADD                   *      *       *       *         *          *           N
CONNECT                   *      *       *       *         *          *           N
CREATE_EXTERNAL_ROUTINE   *      *       *       *         *          *           N
CREATE_NOT_FENCED_ROUTINE *      *       *       *         *          *           Y
CREATETAB                 *      *       *       *         *          *           N
DBADM                     *      *       *       *         *          *           N
IMPLICIT_SCHEMA           *      *       *       *         *          *           N
LOAD                      *      *       *       *         *          *           Y
QUIESCE_CONNECT           *      *       *       *         *          *           N
SECADM                    *      *       *       *         *          *           N
SYSADM                    *      *       *       *         *          *           *
SYSCTRL                   *      *       *       *         *          *           *
SYSMAINT                  *      *       *       *         *          *           *
SYSMON                    *      *       *       *         *          *           *
```

使用上の注意

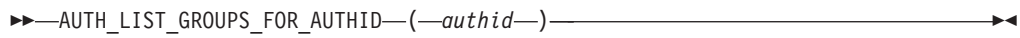
AUTH_LIST_AUTHORITIES_FOR_AUTHID 表関数の出力は *authidtype* に応じて異なります。例えば、USER の *authidtype* の場合、*authid* が以下の手段で保持しているすべての権限を戻します。

- *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。例えば、*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。例えば、*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

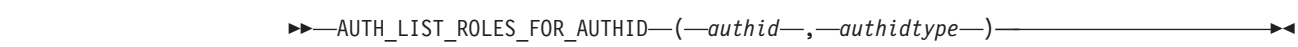
表 105. AUTH_LIST_GROUPS_FOR_AUTHID 表関数によって戻される情報

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 106. 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。

表 106. AUTH_LIST_ROLES_FOR_AUTHID の結果セット (続き)

列名	データ・タイプ	説明
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

AUTHORIZATIONIDS 管理ビューに対する SELECT または CONTROL 特権。

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 107. 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

OBJECTOWNERS 管理ビューに対する SELECT または CONTROL 特権。

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 OJECTSCHEMA='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 108. 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

PRIVILEGES 管理ビューに対する SELECT または CONTROL 特権。

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 109. 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。
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。

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

- APPL_PERFORMANCE、SNAPAPPL_INFO、および SNAPAPPL 管理ビューに対する SELECT または CONTROL 特権。
- SYSMON、SYSCTRL、SYSMAINT、または SYSADM 権限も、スナップショット・モニター・データへのアクセスで必要です。

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)。

```
... AGENT_ID      PERCENT_ROWS_SELECTED DBPARTITIONNUM
... --------...-- -------------------- --------------
...           67                    -              1
...           68                    -              0
...           67                57.14              0
...
```

戻される情報

表 110. 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。
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

- APPLICATIONS および SNAPAPPL_INFO 管理ビューに対する SELECT または CONTROL 特権。
- SYSMON、SYSCTRL、SYSMAINT、または SYSADM 権限。この権限はスナップショット・モニター・データへのアクセスで必要です。

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 111. 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

表 111. 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
AUTHID	VARCHAR(128)	auth_id - 許可 ID
APPL_ID	VARCHAR(128)	appl_id - アプリケーション 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
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

表 111. 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_PID	BIGINT	client_pid - クライアント・プロセス ID

表 111. 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 • UNKNOWN • UNKNOWN_DRDA • VM_DRDA • VSE_DRDA • WINDOWS • WINDOWS95

表 111. 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

- BP_HITRATIO および SNAPBP 管理ビューに対する SELECT または CONTROL 特権。
- SYSMON、SYSCTRL、SYSMAINT、または SYSADM 権限も、スナップショット・モニター・データへのアクセスで必要です。

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、論理読み取りに対する物理読み取りの率として算出されます。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、その分操作にコストがかかることになります。

戻される情報

表 112. 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。

表 112. BP_HITRATIO 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
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

- BP_READ_IO および SNAPBP 管理ビューに対する SELECT または CONTROL 特権。
- SYSMON、SYSCTRL、SYSMAINT、または SYSADM 権限も、スナップショット・モニター・データへのアクセスで必要です。

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 113. 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。
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。

表 113. 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
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

- BP_WRITE_IO および SNAPBP 管理ビューに対する SELECT または CONTROL 特権。
- SYSMON、SYSCTRL、SYSMAINT、または SYSADM 権限も、スナップショット・モニター・データへのアクセスで必要です。

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 114. 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。
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。ビューは LIST TABLESPACE CONTAINERS CLP コマンドの SQL インターフェースです。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

- CONTAINER_UTILIZATION および SNAPCONTAINER 管理ビューに対する SELECT または CONTROL 特権。
- SYSMON、SYSCTRL、SYSMAINT、または SYSADM 権限 (スナップショット・モニター・データへのアクセスで必要です)。

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 115. 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

表 115. 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
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
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

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

- LOCKS_HELD、SNAPLOCK、および SNAPAPPL_INFO 管理ビューに対する SELECT または CONTROL 特権。

- SYSMON、SYSCTRL、SYSMAINT、または SYSADM 権限も、スナップショット・モニター・データへのアクセスで必要です。

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 116. 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

表 116. 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
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
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 116. 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 • NX • S • SIX • U • W • 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

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

- LOCKWAITS、SNAPAPPL_INFO、および SNAPLOCKWAIT 管理ビューに対する SELECT または CONTROL 特権。
- SYSMON、SYSCTRL、SYSMAINT、または SYSADM 権限も、スナップショット・モニター・データへのアクセスで必要です。

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 117. 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
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

表 117. 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_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 117. 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 • NX • S • SIX • U • W • 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 • NX • S • SIX • U • W • 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

表 117. 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
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

- LOG_UTILIZATION および SNAPDB 管理ビューに対する SELECT または CONTROL 特権。
- SYSMON、SYSCTRL、SYSMAINT、または SYSADM 権限も、スナップショット・モニター・データへのアクセスで必要です。

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 118. 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

表 118. LOG_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
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

- LONG_RUNNING_SQL、SNAPSTMT、SNAPAPPL_INFO、および SNAPAPPL 管理ビューに対する SELECT または CONTROL 特権。
- SYSMON、SYSCTRL、SYSMAINT、または SYSADM 権限も、スナップショット・モニター・データへのアクセスで必要です。

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 119. 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 119. 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

- QUERY_PREP_COST および SNAPDYN_SQL 管理ビューに対する SELECT または CONTROL 特権。
- SYSMON、SYSCTRL、SYSMAINT、または SYSADM 権限も、スナップショット・モニター・データへのアクセスで必要です。

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 120. 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。

表 120. QUERY_PREP_COST 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
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、378 ページの表 121 を参照してください。

許可

- SYSMON 権限
- SNAPAGENT 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_AGENT 表関数に対する EXECUTE 特権。

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
...
...
```

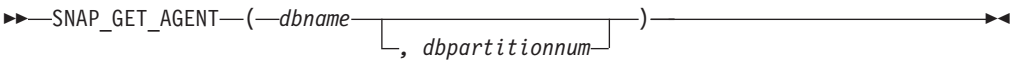
SNAP_GET_AGENT 表関数

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、 378 ページの表 121 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

- SYSMON 権限
- SNAP_GET_AGENT 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 121. 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、381 ページの表 122 を参照してください。

許可

- SYSMON 権限

- SNAPAGENT_MEMORY_POOL 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_AGENT_MEMORY_POOL 表関数に対する EXECUTE 特権。

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.
```

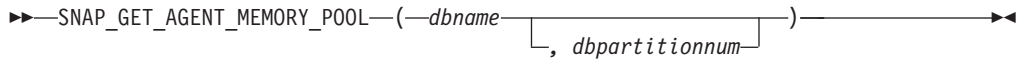
SNAP_GET_AGENT_MEMORY_POOL 表関数

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、 381 ページの表 122 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

- SYSMON 権限
- SNAP_GET_AGENT_MEMORY_POOL 表関数に対する EXECUTE 特権。

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 122. 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
POOL_CONFIG_SIZE	BIGINT	pool_config_size - メモリー・プールの構成済みサイズ

表 122. 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
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、 384 ページの表 123 を参照してください。

許可

- SYSMON 権限
- SNAPAPPL_INFO 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_APPL_INFO_V95 表関数に対する EXECUTE 特権。

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.
```

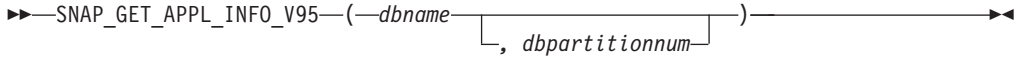
SNAP_GET_APPL_INFO_V95 表関数

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、 384 ページの表 123 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

- SYSMON 権限
- SNAP_GET_APPL_INFO_V95 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_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 123. 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。

表 123. 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
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
COORD_NODE_NUM	SMALLINT	coord_node - コーディネーター・ノード

表 123. 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 : モニター・エレメント。 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 • BINDADD • CONNECT • CREATE_EXT_RT • CREATE_NOT_FENC • CREATETAB • DBADM • IMPLICIT_SCHEMA • LOAD • LIBADM • QUIESCE_CONN • SECADM • SYSADM • SYSCTRL • SYSMAINT • SYSMON • SYSQUIESCE "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 123. 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

表 123. 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_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 123. 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

表 123. 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
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.
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 391 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、 393 ページの表 124 を参照してください。

許可

- SYSMON 権限
- SNAPAPPL 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_APPL_V95 表関数に対する EXECUTE 特権。

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.
```

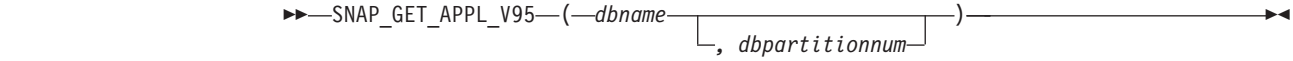
SNAP_GET_APPL_V95 表関数

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、SN AP_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、 393 ページの表 124 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

- SYSMON 権限
- SNAP_GET_APPL_V95 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_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 124. 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
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

表 124. 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_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
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

表 124. 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
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
PKG_CACHE_LOOKUPS	BIGINT	pkg_cache_lookups - パッケージ・キャッシュ参照

表 124. 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
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
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 124. 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
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 (小数部分、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 (小数部分、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)*

表 124. 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
UOW_ELAPSED_TIME_MS	BIGINT	uow_elapsed_time - 最新の作業単位の経過時間 (小数部分、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 (小数部分、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
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 124. 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

* この列の基になっているモニター・エレメントの経過合計時間を計算するには、このモニター・エレメントの _S で終わっている列に報告されている秒の整数部分と、このモニター・エレメントの _MS で終わっている列に報告されている小数部分を合計しなければなりません。これは、次の計算式を使って行います。 (*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、401 ページの表 125 を参照してください。

許可

- SYSMON 権限
- SNAPBP 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_BP_V95 表関数に対する EXECUTE 特権。

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
```

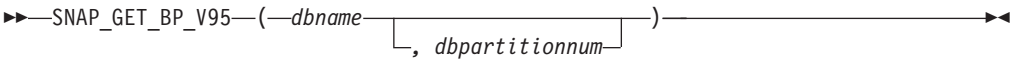
SNAP_GET_BP_V95 表関数

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、 401 ページの表 125 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

- SYSMON 権限
- SNAP_GET_BP_V95 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_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 125. 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

表 125. 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_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
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 125. 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_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
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

表 125. 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
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、407 ページの表 126 を参照してください。

許可

- SYSMON 権限
- SNAPBP_PART 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_BP_PART 表関数に対する EXECUTE 特権。

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
...
```

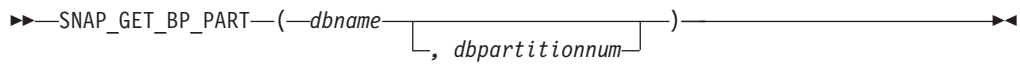
SNAP_GET_BP_PART 表関数

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、407 ページの表 126 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

- SYSMON 権限
- SNAP_GET_BP_PART 表関数に対する EXECUTE 特権。

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 126. 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、410 ページの表 127 を参照してください。

許可

- SYSMON 権限
- SNAPCONTAINER 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_CONTAINER_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
```

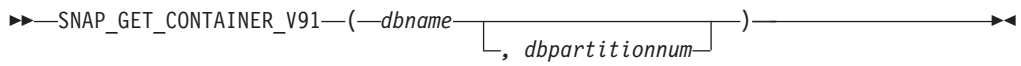
SNAP_GET_CONTAINER_V91 表関数

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、 410 ページの表 127 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

- SYSMON 権限
- SNAP_GET_CONTAINER_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・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 127. 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

表 127. 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
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	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

『SNAPDB 管理ビュー』と 412 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、 415 ページの表 128 を参照してください。

許可

- SYSMON 権限
- SNAPDB 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_DB_V95 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.
```

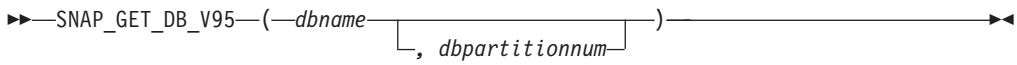
SNAP_GET_DB_V95 表関数

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、 415 ページの表 128 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

- SYSMON 権限
- SNAP_GET_DB_V95 表関数に対する EXECUTE 特権。

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 128. 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
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 128. 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 128. 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・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
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

表 128. 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
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
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

表 128. 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_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
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

表 128. 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
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 (小数部分、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 128. 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 (小数部分、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 (小数部分、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 128. 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 128. 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。

* この列の基になっているモニター・エレメントの経過合計時間を計算するには、このモニター・エレメントの _S で終わっている列に報告されている秒の整数部分と、このモニター・エレメントの _MS で終わっている列に報告されている小数部分を合計しなければなりません。これは、次の計算式を使って行います。 (*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、426 ページの表 129 を参照してください。

許可

- SYSMON 権限
- SNAPDB_MEMORY_POOL 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_DB_MEMORY_POOL 表関数に対する EXECUTE 特権。

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.
```

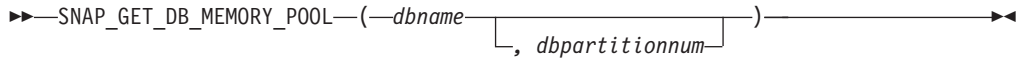
SNAP_GET_DB_MEMORY_POOL 表関数

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、426 ページの表 129 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

- SYSMON 権限
- SNAP_GET_DB_MEMORY_POOL 表関数に対する EXECUTE 特権。

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 129. 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 129. 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、429 ページの表 130 を参照してください。

許可

- SYSMON 権限
- SNAPDBM 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_DBM_V95 表関数に対する EXECUTE 特権。

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
```

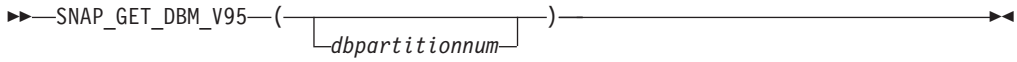
SNAP_GET_DBM_V95 表関数

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、429 ページの表 130 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

- SYSMON 権限
- SNAP_GET_DBM_V95 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_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 130. 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

表 130. 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
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
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

表 130. 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
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。
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、433 ページの表 131 を参照してください。

許可

- SYSMON 権限
- SNAPDBM_MEMORY_POOL 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_DBM_MEMORY_POOL 表関数に対する EXECUTE 特権。

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
...
```

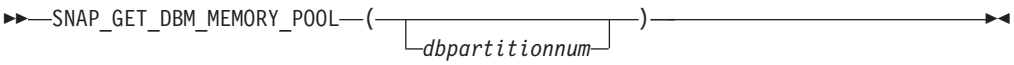
SNAP_GET_DBM_MEMORY_POOL 表関数

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、433 ページの表 131 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

- SYSMON 権限
- SNAP_GET_DBM_MEMORY_POOL 表関数に対する EXECUTE 特権。

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 131. 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。

表 131. 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 (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。

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、437 ページの表 132 を参照してください。

許可

- SYSMON 権限
- SNAPDETAILLOG 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_DETAILLOG_V91 表関数に対する EXECUTE 特権。

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
```

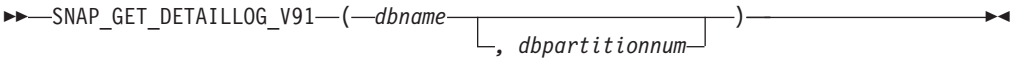
SNAP_GET_DETAILLOG_V91 表関数

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、437 ページの表 132 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

- SYSMON 権限
- SNAP_GET_DETAILLOG_V91 表関数に対する EXECUTE 特権。

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 132. 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 438 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、 441 ページの表 133 を参照してください。

許可

- SYSMON 権限
- SNAPDYN_SQL 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_DYN_SQL_V95 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

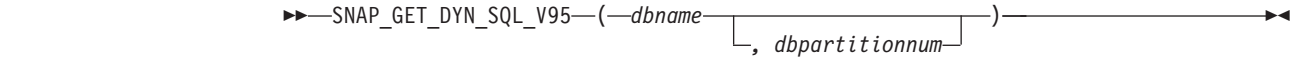
SNAP_GET_DYN_SQL_V91 表関数は SNAPDYN_SQL 管理ビューと同じ情報を戻します。ただし、SNAP_GET_DYN_SQL_V95 表関数の場合は、特定のデータベー

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、 441 ページの表 133 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

- SYSMON 権限
- SNAP_GET_DYN_SQL_V95 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_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 133. 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
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

表 133. 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_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 (小数部分、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 (小数部分、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 (小数部分、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)。

* この列の基になっているモニター・エレメントの経過合計時間を計算するには、このモニター・エレメントの _S で終わっている列に報告されている秒の整数部分と、このモニター・エレメントの _MS で終わっている列に報告されている小数部分を合計しなければなりません。これは、次の計算式を使って行います。 (*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、 444 ページの表 134 を参照してください。

許可

- SYSMON 権限
- SNAPFCM 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_FCM 表関数に対する EXECUTE 特権。

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
```

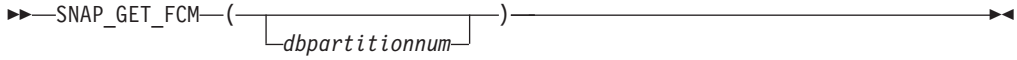
SNAP_GET_FCM 表関数

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、 444 ページの表 134 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

- SYSMON 権限
- SNAP_GET_FCM 表関数に対する EXECUTE 特権。

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 134. 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、447 ページの表 135 を参照してください。

許可

- SYSMON 権限
- SNAPFCM_PART 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_FCM_PART 表関数に対する EXECUTE 特権。

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.
```

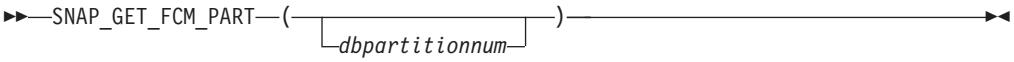
SNAP_GET_FCM_PART 表関数

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、447 ページの表 135 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

- SYSMON 権限
- SNAP_GET_FCM_PART 表関数に対する EXECUTE 特権。

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 135. 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、 449 ページの表 136 を参照してください。

許可

- SYSMON 権限
- SNAPHADR 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_HADR 表関数に対する EXECUTE 特権。

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.
```

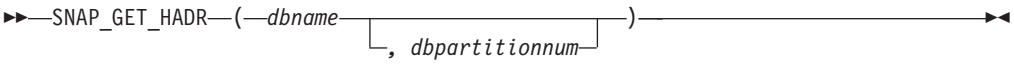
SNAP_GET_HADR 表関数

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、 449 ページの表 136 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

- SYSMON 権限
- SNAP_GET_HADR 表関数に対する EXECUTE 特権。

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 136. 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

表 136. 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_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
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

表 136. 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_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

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、453 ページの表 137 を参照してください。

許可

- SYSMON 権限

- SNAPLOCK 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_LOCK 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.
```

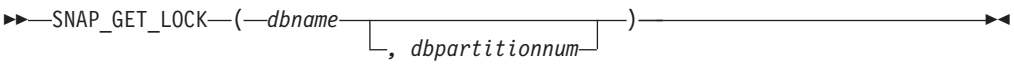
SNAP_GET_LOCK 表関数

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、 453 ページの表 137 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

- SYSMON 権限
- SNAP_GET_LOCK 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 137. 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 137. 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 137. 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 • NX • S • SIX • U • W • 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 137. 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 での定義を基にしてテキスト 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 • NX • S • SIX • U • W • 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 137. 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

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、460 ページの表 138 を参照してください。

許可

- SYSMON 権限
- SNAPLOCKWAIT 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_LOCKWAIT 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
```

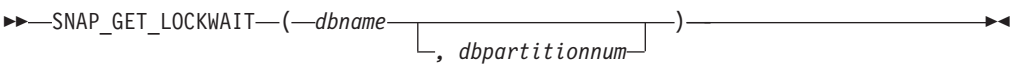
SNAP_GET_LOCKWAIT 表関数

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、460 ページの表 138 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

- SYSMON 権限
- SNAP_GET_LOCKWAIT 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 138. 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
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 • NX • S • SIX • U • W • X • Z

表 138. 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 138. 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 • NX • S • SIX • U • W • 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

表 138. 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_ATTRIBUTES	VARCHAR(128)	lock_attributes - ロック属性。このインターフェースは、sqlmon.h での定義を基にしてテキスト 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_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 • NX • S • SIX • U • W • 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。

表 138. 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
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、466 ページの表 139 を参照してください。

許可

- SYSMON 権限
- SNAPSTMT 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_STMT 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.
```

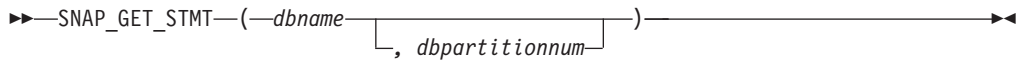
SNAP_GET_STMT 表関数

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、 466 ページの表 139 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

- SYSMON 権限
- SNAP_GET_STMT 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 139. 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

表 139. 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
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	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

表 139. 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_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
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

表 139. 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
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 (小数部分、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 (小数部分、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 (小数部分、マイクロ秒単位)*
BLOCKING_CURSOR	SMALLINT	blocking_cursor - ブロック・カーソル
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

表 139. 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
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
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。

* この列の基になっているモニター・エレメントの経過合計時間を計算するには、このモニター・エレメントの _S で終わっている列に報告されている秒の整数部分と、このモニター・エレメントの _MS で終わっている列に報告されている小数部分を合計しなければなりません。これは、次の計算式を使って行います。 (*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

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、473 ページの表 140 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 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 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.
```

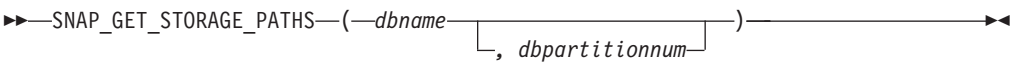
SNAP_GET_STORAGE_PATHS 表関数

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、473 ページの表 140 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

- SYSMON 権限
- SNAP_GET_STORAGE_PATHS 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(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 140. 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、476 ページの表 141 を参照してください。

許可

- SYSMON 権限
- SNAPSUBSECTION 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_SUBSECTION 表関数に対する EXECUTE 特権。

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
```

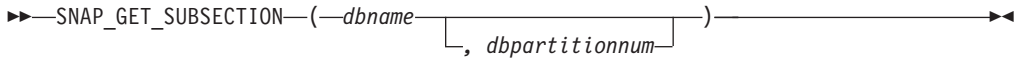
SNAP_GET_SUBSECTION 表関数

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、476 ページの表 141 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

- SYSMON 権限
- SNAP_GET_SUBSECTION 表関数に対する EXECUTE 特権。

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 141. 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 (小数部分、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 (小数部分、マイクロ秒単位)*
SS_NUMBER	INTEGER	ss_number - サブセクション番号

表 141. 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_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。

* この列の基になっているモニター・エレメントの経過合計時間を計算するには、このモニター・エレメントの _S で終わっている列に報告されている秒の整数部分と、このモニター・エレメントの _MS で終わっている列に報告されている小数部分を合計しなければなりません。これは、次の計算式を使って行います。 (*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、479 ページの表 142 を参照してください。

許可

- SYSMON 権限
- SNAPSWITCHES 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_SWITCHES 表関数に対する EXECUTE 特権。

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
```

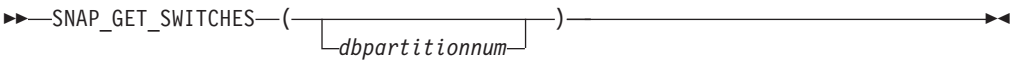
SNAP_GET_SWITCHES 表関数

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、 479 ページの表 142 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

- SYSMON 権限
- SNAP_GET_SWITCHES 表関数に対する EXECUTE 特権。

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 142. 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)。

表 142. 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
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、482 ページの表 143 を参照してください。

許可

- SYSMON 権限
- SNAPTAB 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_TAB_V91 表関数に対する EXECUTE 特権。

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.
```

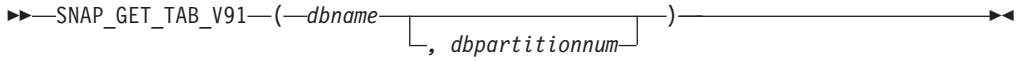
SNAP_GET_TAB_V91 表関数

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、482 ページの表 143 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

- SYSMON 権限
- SNAP_GET_TAB_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 143. 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

表 143. 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
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。

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、486 ページの表 144 を参照してください。

許可

- SYSMON 権限
- SNAPTAB_REORG 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_TAB_REORG 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
```

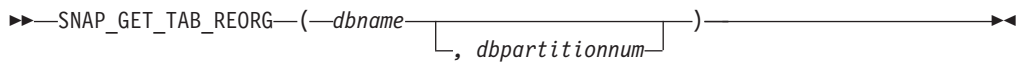
SNAP_GET_TAB_REORG 表関数

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、 486 ページの表 144 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

- SYSMON 権限
- SNAP_GET_TAB_REORG 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
...
```

戻される情報

表 144. 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 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 144. 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
		
		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`

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

表 144. 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_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、490 ページの表 145 を参照してください。

許可

- SYSMON 権限
- SNAPTBSP 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_TBSP_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、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.
```

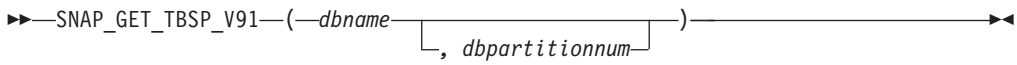
SNAP_GET_TBSP_V91 表関数

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、490 ページの表 145 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

- SYSMON 権限
- SNAP_GET_TBSP_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、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 145. 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

表 145. 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_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 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。 • 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
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

表 145. 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_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_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

表 145. 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_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
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 - 自動ストレージの使用
TBSP_AUTO_RESIZE_ENABLED	SMALLINT	tablespace_auto_resize_enabled - 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

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、496 ページの表 146 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_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

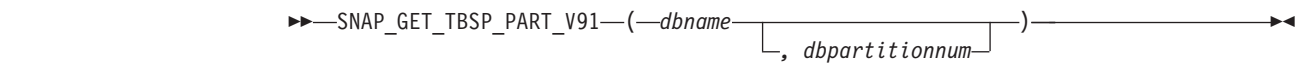
SNAP_GET_TBSP_PART_V91 表関数は SNAPTBSP_PART 管理ビューと同じ情報を戻します。ただし、SNAP_GET_TBSP_PART_V91 表関数の場合は、特定のデータベ

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、496 ページの表 146 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

- SYSMON 権限
- SNAP_GET_TBSP_PART_V91 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_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 146. 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。
TBSP_NAME	VARCHAR (128)	tablespace_name - 表スペース名
TBSP_ID	BIGINT	tablespace_id - 表スペース ID

表 146. 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	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 146. 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_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
TBSP_NUM_CONTAINERS	BIGINT	tablespace_num_containers - 表スペース内のコンテナー数

表 146. 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_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、502 ページの表 147 を参照してください。

許可

- SYSMON 権限

- SNAPTBSP_QUIESCER 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_TBSP_QUIESCER 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化された表の名前を特定する

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.
```

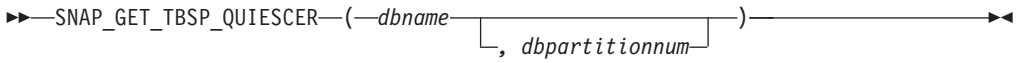
SNAP_GET_TBSP_QUIESCER 表関数

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、 502 ページの表 147 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

- SYSMON 権限
- SNAP_GET_TBSP_QUIESCER 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 147. 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

表 147. 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 (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
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、506 ページの表 148 を参照してください。

許可

- SYSMON 権限
- SNAPTBSP_RANGE 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_TBSP_RANGE 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

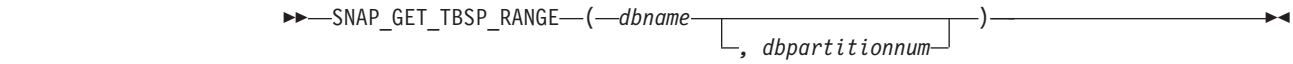
SNAP_GET_TBSP_RANGE 表関数は SNAPTBSP_RANGE 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションのいずれかの特定のデータベースを対象とした情報を検索することができます。

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、 506 ページの表 148 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

- SYSMON 権限
- SNAP_GET_TBSP_RANGE 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 148. 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

表 148. 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
RANGE_CONTAINER_ID	BIGINT	range_container_id - 範囲コンテナー
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、509 ページの表 149 を参照してください。

許可

- SYSMON 権限
- SNAPUTIL 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_UTIL 表関数に対する EXECUTE 特権。

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
```

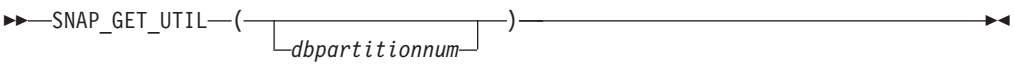
SNAP_GET_UTIL 表関数

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、 509 ページの表 149 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

- SYSMON 権限
- SNAP_GET_UTIL 表関数に対する EXECUTE 特権。

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 149. 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。
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。

表 149. 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_DESCRIPTION	VARCHAR(2048)	utility_description - ユーティリティー記述。NULL 可能。
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
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、 512 ページの表 150 を参照してください。

許可

- SYSMON 権限
- SNAPUTIL_PROGRESS 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_UTIL_PROGRESS 表関数に対する EXECUTE 特権。

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 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.
```

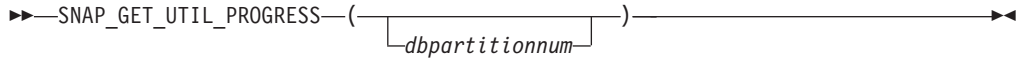
SNAP_GET_UTIL_PROGRESS 表関数

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、 512 ページの表 150 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

- SYSMON 権限
- SNAP_GET_UTIL_PROGRESS 表関数に対する EXECUTE 特権。

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 150. 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 150. 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、 378 ページの表 121 を参照してください。

許可

- SYSMON 権限
- SNAPAGENT 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_AGENT 表関数に対する EXECUTE 特権。

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
...
...
```

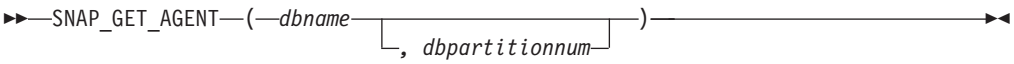
SNAP_GET_AGENT 表関数

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、378 ページの表 121 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

- SYSMON 権限
- SNAP_GET_AGENT 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 151. 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)

表 151. 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 (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_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、 381 ページの表 122 を参照してください。

許可

- SYSMON 権限
- SNAPAGENT_MEMORY_POOL 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_AGENT_MEMORY_POOL 表関数に対する EXECUTE 特権。

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.
```

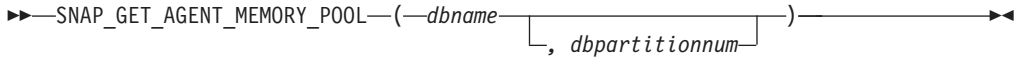
SNAP_GET_AGENT_MEMORY_POOL 表関数

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、 381 ページの表 122 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

- SYSMON 権限
- SNAP_GET_AGENT_MEMORY_POOL 表関数に対する EXECUTE 特権。

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 152. 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)

表 152. 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
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
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、384 ページの表 123 を参照してください。

許可

- SYSMON 権限
- SNAPAPPL_INFO 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_APPL_INFO_V95 表関数に対する EXECUTE 特権。

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

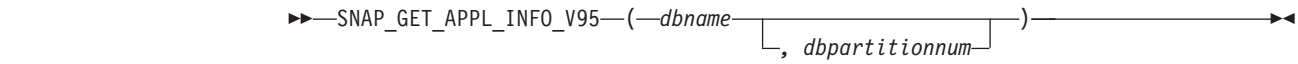
SNAP_GET_APPL_INFO_V95 表関数は、SNAPAPPL_INFO 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションに関する情報を取得することができます。

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、384 ページの表 123 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

- SYSMON 権限
- SNAP_GET_APPL_INFO_V95 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_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 153. 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 153. 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 153. 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 : モニター・エレメント。 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 • BINDADD • CONNECT • CREATE_EXT_RT • CREATE_NOT_FENC • CREATETAB • DBADM • IMPLICIT_SCHEMA • LOAD • LIBADM • QUIESCE_CONN • SECADM • SYSADM • SYSCTRL • SYSMAINT • SYSMON • SYSQUIESCE "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 153. 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

表 153. 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_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 153. 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

表 153. 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
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.
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

390 ページの『SNAPAPPL 管理ビュー』および 391 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、 393 ページの表 124 を参照してください。

許可

- SYSMON 権限
- SNAPAPPL 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_APPL_V95 表関数に対する EXECUTE 特権。

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.
```

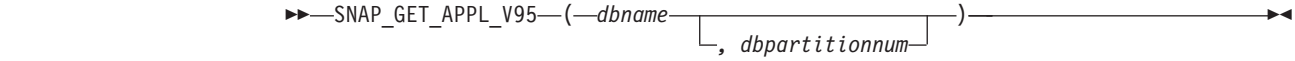
SNAP_GET_APPL_V95 表関数

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、SN AP_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、 393 ページの表 124 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

- SYSMON 権限
- SNAP_GET_APPL_V95 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_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 154. 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
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

表 154. 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_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
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

表 154. 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
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
PKG_CACHE_LOOKUPS	BIGINT	pkg_cache_lookups - パッケージ・キャッシュ参照

表 154. 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
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
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 154. 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
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 (小数部分、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 (小数部分、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)*

表 154. 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
UOW_ELAPSED_TIME_MS	BIGINT	uow_elapsed_time - 最新の作業単位の経過時間 (小数部分、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 (小数部分、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
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 154. 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

* この列の基になっているモニター・エレメントの経過合計時間を計算するには、このモニター・エレメントの _S で終わっている列に報告されている秒の整数部分と、このモニター・エレメントの _MS で終わっている列に報告されている小数部分を合計しなければなりません。これは、次の計算式を使って行います。 (*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、401 ページの表 125 を参照してください。

許可

- SYSMON 権限
- SNAPBP 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_BP_V95 表関数に対する EXECUTE 特権。

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
```

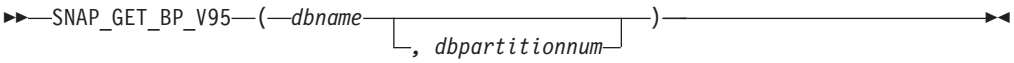
SNAP_GET_BP_V95 表関数

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、401 ページの表 125 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

- SYSMON 権限
- SNAP_GET_BP_V95 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_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 155. 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

表 155. 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_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
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 155. 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_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
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

表 155. 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
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、407 ページの表 126 を参照してください。

許可

- SYSMON 権限
- SNAPBP_PART 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_BP_PART 表関数に対する EXECUTE 特権。

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
...
```

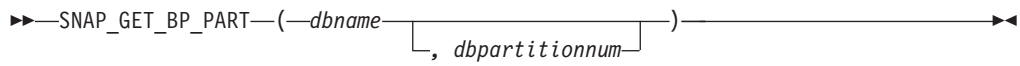
SNAP_GET_BP_PART 表関数

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、407 ページの表 126 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

- SYSMON 権限
- SNAP_GET_BP_PART 表関数に対する EXECUTE 特権。

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 156. 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、410 ページの表 127 を参照してください。

許可

- SYSMON 権限
- SNAPCONTAINER 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_CONTAINER_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
```

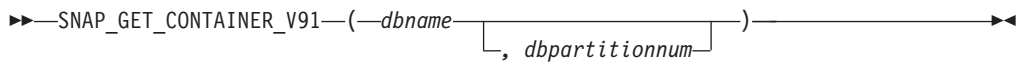
SNAP_GET_CONTAINER_V91 表関数

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、 410 ページの表 127 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

- SYSMON 権限
- SNAP_GET_CONTAINER_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・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 157. 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

表 157. 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
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	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

411 ページの『SNAPDB 管理ビュー』と 412 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、415 ページの表 128 を参照してください。

許可

- SYSMON 権限
- SNAPDB 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_DB_V95 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.
```

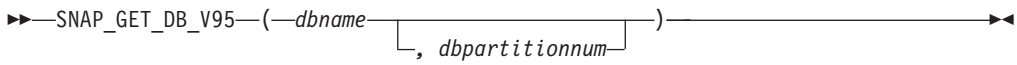
SNAP_GET_DB_V95 表関数

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、 415 ページの表 128 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

- SYSMON 権限
- SNAP_GET_DB_V95 表関数に対する EXECUTE 特権。

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 158. 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
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 158. 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 158. 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・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
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

表 158. 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
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
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

表 158. 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_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
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

表 158. 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
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 (小数部分、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 158. 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 (小数部分、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 (小数部分、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 158. 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 158. 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。

* この列の基になっているモニター・エレメントの経過合計時間を計算するには、このモニター・エレメントの _S で終わっている列に報告されている秒の整数部分と、このモニター・エレメントの _MS で終わっている列に報告されている小数部分を合計しなければなりません。これは、次の計算式を使って行います。 (*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、426 ページの表 129 を参照してください。

許可

- SYSMON 権限
- SNAPDB_MEMORY_POOL 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_DB_MEMORY_POOL 表関数に対する EXECUTE 特権。

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.
```

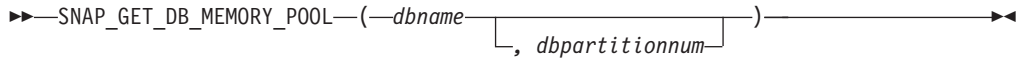
SNAP_GET_DB_MEMORY_POOL 表関数

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、426 ページの表 129 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

- SYSMON 権限
- SNAP_GET_DB_MEMORY_POOL 表関数に対する EXECUTE 特権。

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 159. 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 159. 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、429 ページの表 130 を参照してください。

許可

- SYSMON 権限
- SNAPDBM 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_DBM_V95 表関数に対する EXECUTE 特権。

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
```

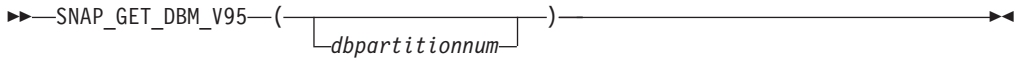
SNAP_GET_DBM_V95 表関数

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、429 ページの表 130 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

- SYSMON 権限
- SNAP_GET_DBM_V95 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_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 160. 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

表 160. 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
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
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

表 160. 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
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。
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、433 ページの表 131 を参照してください。

許可

- SYSMON 権限
- SNAPDBM_MEMORY_POOL 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_DBM_MEMORY_POOL 表関数に対する EXECUTE 特権。

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
...
```

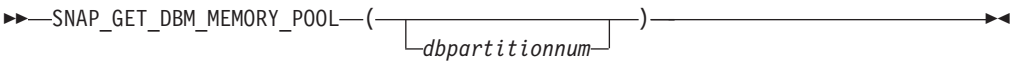
SNAP_GET_DBM_MEMORY_POOL 表関数

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、433 ページの表 131 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

- SYSMON 権限
- SNAP_GET_DBM_MEMORY_POOL 表関数に対する EXECUTE 特権。

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 161. 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。

表 161. 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 (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。

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、437 ページの表 132 を参照してください。

許可

- SYSMON 権限
- SNAPDETAILLOG 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_DETAILLOG_V91 表関数に対する EXECUTE 特権。

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
```

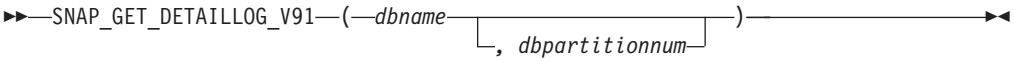
SNAP_GET_DETAILLOG_V91 表関数

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、437 ページの表 132 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

- SYSMON 権限
- SNAP_GET_DETAILLOG_V91 表関数に対する EXECUTE 特権。

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 162. 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

437 ページの『SNAPDYN_SQL 管理ビュー』と 438 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、 441 ページの表 133 を参照してください。

許可

- SYSMON 権限
- SNAPDYN_SQL 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_DYN_SQL_V95 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
```

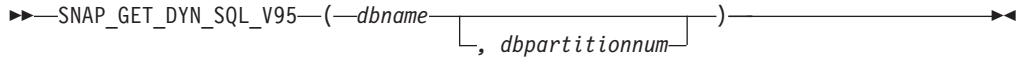
SNAP_GET_DYN_SQL_V95 表関数

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、 441 ページの表 133 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

- SYSMON 権限
- SNAP_GET_DYN_SQL_V95 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_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 163. 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
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

表 163. 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_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 (小数部分、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 (小数部分、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 (小数部分、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)。

* この列の基になっているモニター・エレメントの経過合計時間を計算するには、このモニター・エレメントの _S で終わっている列に報告されている秒の整数部分と、このモニター・エレメントの _MS で終わっている列に報告されている小数部分を合計しなければなりません。これは、次の計算式を使って行います。 (*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、444 ページの表 134 を参照してください。

許可

- SYSMON 権限
- SNAPFCM 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_FCM 表関数に対する EXECUTE 特権。

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
```

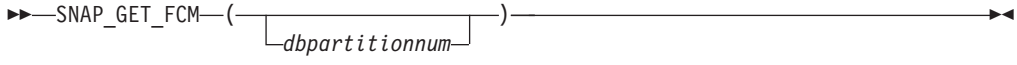
SNAP_GET_FCM 表関数

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、444 ページの表 134 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

- SYSMON 権限
- SNAP_GET_FCM 表関数に対する EXECUTE 特権。

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 164. 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、447 ページの表 135 を参照してください。

許可

- SYSMON 権限
- SNAPFCM_PART 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_FCM_PART 表関数に対する EXECUTE 特権。

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.
```

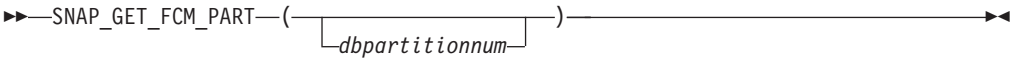
SNAP_GET_FCM_PART 表関数

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、447 ページの表 135 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

- SYSMON 権限
- SNAP_GET_FCM_PART 表関数に対する EXECUTE 特権。

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 165. 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、 449 ページの表 136 を参照してください。

許可

- SYSMON 権限
- SNAPHADR 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_HADR 表関数に対する EXECUTE 特権。

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.
```

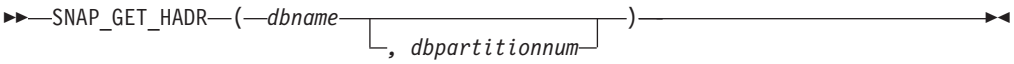
SNAP_GET_HADR 表関数

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、 449 ページの表 136 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

- SYSMON 権限
- SNAP_GET_HADR 表関数に対する EXECUTE 特権。

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 166. 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

表 166. 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_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
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

表 166. 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_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

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、453 ページの表 137 を参照してください。

許可

- SYSMON 権限

- SNAPLOCK 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_LOCK 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.
```

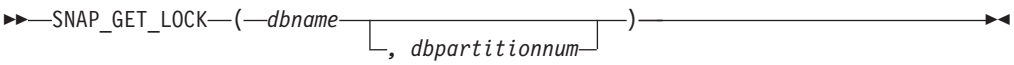
SNAP_GET_LOCK 表関数

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、 453 ページの表 137 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

- SYSMON 権限
- SNAP_GET_LOCK 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 167. 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 167. 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 167. 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 • NX • S • SIX • U • W • 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 167. 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 での定義を基にしてテキスト 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 • NX • S • SIX • U • W • 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 167. 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

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、460 ページの表 138 を参照してください。

許可

- SYSMON 権限
- SNAPLOCKWAIT 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_LOCKWAIT 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
```

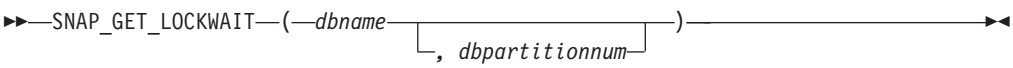
SNAP_GET_LOCKWAIT 表関数

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、460 ページの表 138 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

- SYSMON 権限
- SNAP_GET_LOCKWAIT 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 168. 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
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 • NX • S • SIX • U • W • X • Z

表 168. 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 168. 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 • NX • S • SIX • U • W • 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

表 168. 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_ATTRIBUTES	VARCHAR(128)	lock_attributes - ロック属性。このインターフェースは、sqlmon.h での定義を基にしてテキスト 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_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 • NX • S • SIX • U • W • 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。

表 168. 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
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、466 ページの表 139 を参照してください。

許可

- SYSMON 権限
- SNAPSTMT 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_STMT 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.
```

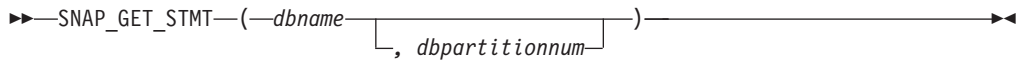
SNAP_GET_STMT 表関数

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、 466 ページの表 139 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

- SYSMON 権限
- SNAP_GET_STMT 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 169. 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

表 169. 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
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	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

表 169. 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_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
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

表 169. 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
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 (小数部分、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 (小数部分、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 (小数部分、マイクロ秒単位)*
BLOCKING_CURSOR	SMALLINT	blocking_cursor - ブロック・カーソル
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

表 169. 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
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
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。

* この列の基になっているモニター・エレメントの経過合計時間を計算するには、このモニター・エレメントの _S で終わっている列に報告されている秒の整数部分と、このモニター・エレメントの _MS で終わっている列に報告されている小数部分を合計しなければなりません。これは、次の計算式を使って行います。 (*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

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、473 ページの表 140 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 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 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.
```

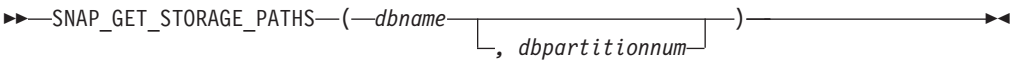
SNAP_GET_STORAGE_PATHS 表関数

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、473 ページの表 140 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

- SYSMON 権限
- SNAP_GET_STORAGE_PATHS 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(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 170. 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、476 ページの表 141 を参照してください。

許可

- SYSMON 権限
- SNAPSUBSECTION 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_SUBSECTION 表関数に対する EXECUTE 特権。

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
```

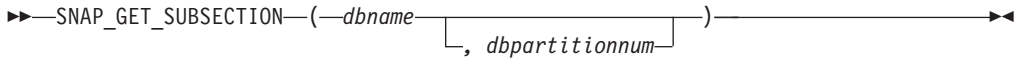
SNAP_GET_SUBSECTION 表関数

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、476 ページの表 141 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

- SYSMON 権限
- SNAP_GET_SUBSECTION 表関数に対する EXECUTE 特権。

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 171. 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 (小数部分、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 (小数部分、マイクロ秒単位)*
SS_NUMBER	INTEGER	ss_number - サブセクション番号

表 171. 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_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。

* この列の基になっているモニター・エレメントの経過合計時間を計算するには、このモニター・エレメントの _S で終わっている列に報告されている秒の整数部分と、このモニター・エレメントの _MS で終わっている列に報告されている小数部分を合計しなければなりません。これは、次の計算式を使って行います。 (*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、 479 ページの表 142 を参照してください。

許可

- SYSMON 権限
- SNAPSWITCHES 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_SWITCHES 表関数に対する EXECUTE 特権。

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
```

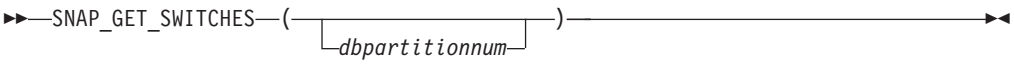
SNAP_GET_SWITCHES 表関数

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、 479 ページの表 142 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

- SYSMON 権限
- SNAP_GET_SWITCHES 表関数に対する EXECUTE 特権。

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 172. 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)。

表 172. 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
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、482 ページの表 143 を参照してください。

許可

- SYSMON 権限
- SNAPTAB 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_TAB_V91 表関数に対する EXECUTE 特権。

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.
```

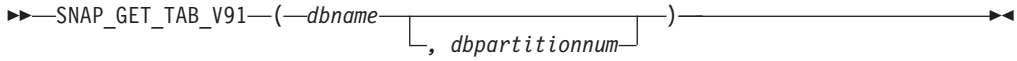
SNAP_GET_TAB_V91 表関数

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、482 ページの表 143 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

- SYSMON 権限
- SNAP_GET_TAB_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 173. 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

表 173. 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
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。

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、486 ページの表 144 を参照してください。

許可

- SYSMON 権限
- SNAPTAB_REORG 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_TAB_REORG 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
```

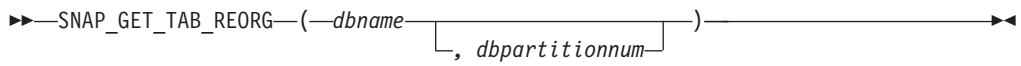
SNAP_GET_TAB_REORG 表関数

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、 486 ページの表 144 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

- SYSMON 権限
- SNAP_GET_TAB_REORG 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
...
```

戻される情報

表 174. 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 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 174. 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
		
		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`

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

表 174. 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_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、 490 ページの表 145 を参照してください。

許可

- SYSMON 権限
- SNAPTBSP 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_TBSP_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、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.
```

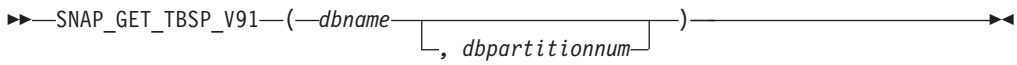
SNAP_GET_TBSP_V91 表関数

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、 490 ページの表 145 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

- SYSMON 権限
- SNAP_GET_TBSP_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、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 175. 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

表 175. 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_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 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。 • 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
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

表 175. 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_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_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

表 175. 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_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
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 - 自動ストレージの使用
TBSP_AUTO_RESIZE_ENABLED	SMALLINT	tablespace_auto_resize_enabled - 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

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、496 ページの表 146 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_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

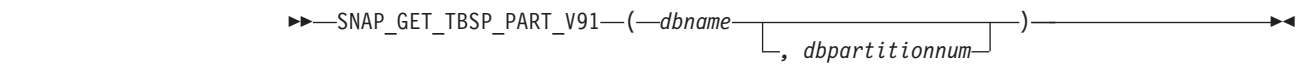
SNAP_GET_TBSP_PART_V91 表関数は SNAPTBSP_PART 管理ビューと同じ情報を戻します。ただし、SNAP_GET_TBSP_PART_V91 表関数の場合は、特定のデータベ

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、496 ページの表 146 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

- SYSMON 権限
- SNAP_GET_TBSP_PART_V91 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_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 176. 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。
TBSP_NAME	VARCHAR (128)	tablespace_name - 表スペース名
TBSP_ID	BIGINT	tablespace_id - 表スペース ID

表 176. 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	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 176. 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_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
TBSP_NUM_CONTAINERS	BIGINT	tablespace_num_containers - 表スペース内のコンテナー数

表 176. 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_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、502 ページの表 147 を参照してください。

許可

- SYSMON 権限

- SNAPTBSP_QUIESCER 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_TBSP_QUIESCER 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化された表の名前を特定する

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.
```

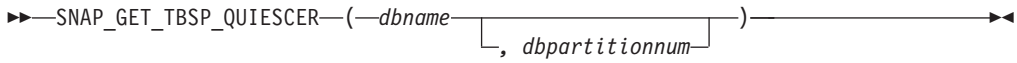
SNAP_GET_TBSP_QUIESCER 表関数

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、 502 ページの表 147 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

- SYSMON 権限
- SNAP_GET_TBSP_QUIESCER 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 177. 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

表 177. 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 (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
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、506 ページの表 148 を参照してください。

許可

- SYSMON 権限
- SNAPTBSP_RANGE 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_TBSP_RANGE 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

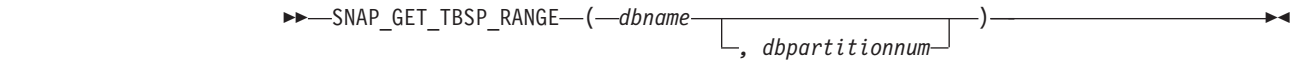
SNAP_GET_TBSP_RANGE 表関数は SNAPTBSP_RANGE 管理ビューと同じ情報を戻しますが、特定のデータベース・パーティション、すべてのデータベース・パーティションの集約、またはすべてのデータベース・パーティションのいずれかの特定のデータベースを対象とした情報を検索することができます。

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、 506 ページの表 148 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

- SYSMON 権限
- SNAP_GET_TBSP_RANGE 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 178. 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

表 178. 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
RANGE_CONTAINER_ID	BIGINT	range_container_id - 範囲コンテナー
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、509 ページの表 149 を参照してください。

許可

- SYSMON 権限
- SNAPUTIL 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_UTIL 表関数に対する EXECUTE 特権。

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
```

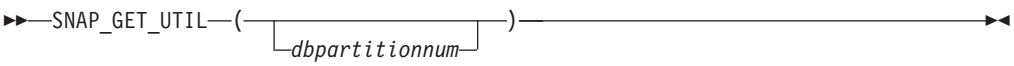
SNAP_GET_UTIL 表関数

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、 509 ページの表 149 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

- SYSMON 権限
- SNAP_GET_UTIL 表関数に対する EXECUTE 特権。

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 179. 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。
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。

表 179. 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_DESCRIPTION	VARCHAR(2048)	utility_description - ユーティリティー記述。NULL 可能。
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
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、 512 ページの表 150 を参照してください。

許可

- SYSMON 権限
- SNAPUTIL_PROGRESS 管理ビューに対する SELECT または CONTROL 特権、および SNAP_GET_UTIL_PROGRESS 表関数に対する EXECUTE 特権。

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 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.
```

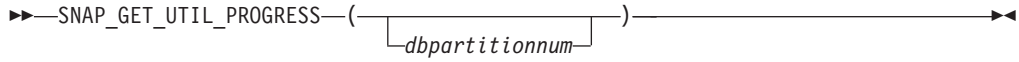
SNAP_GET_UTIL_PROGRESS 表関数

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、 512 ページの表 150 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

- SYSMON 権限
- SNAP_GET_UTIL_PROGRESS 表関数に対する EXECUTE 特権。

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 180. 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 180. 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。ビューは LIST TABLESPACES CLP コマンドの SQL インターフェースです。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

- TBSP_UTILIZATION、SNAPTBSP、SNAPTBSP_PART 管理ビューおよび SYSCAT.TABLESPACES カタログ・ビューに対する SELECT または CONTROL 特権。
- SYSMON、SYSCTRL、SYSMAINT、または SYSADM 権限も、スナップショット・モニター・データへのアクセスで必要です。

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 181. 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。

表 181. 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_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
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

表 181. 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_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 - 自動ストレージの使用
TBSP_AUTO_RESIZE_ENABLED	SMALLINT	tablespace_auto_resize_enabled - 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

- TOP_DYNAMIC_SQL および SNAPDYN_SQL 管理ビューに対する SELECT または CONTROL 特権。
- SYSMON、SYSCTRL、SYSMAINT、または SYSADM 権限も、スナップショット・モニター・データへのアクセスで必要です。

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 182. 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
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。

SQL プロシージャー・ルーチン

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。

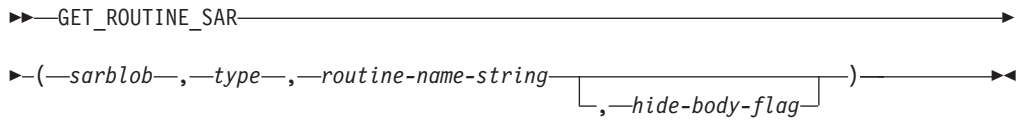
この関数の結果は、長さ属性が 1024 である、可変長文字ストリング (VARCHAR) 値です。

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。

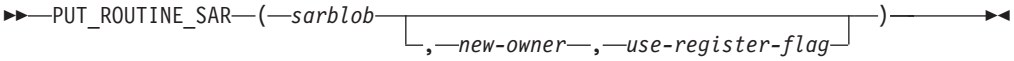
使用上の注意

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)。

HIDE BODY オプションを指定して得た SAR を使用して、PUT ROUTINE コマンドによって作成したストアード・プロシージャーに対し、新規アーカイブ・ファイルを生成するために GET ROUTINE コマンドを使用した場合、ストアード・プロシージャーの新規作成は失敗します。 HIDE BODY オプションを指定すると、ストアード・プロシージャーから SQL 本体が除去されるため、アーカイブ・ファイルは失敗します。

PUT_ROUTINE_SAR



スキーマは SYSFUN です。

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

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。

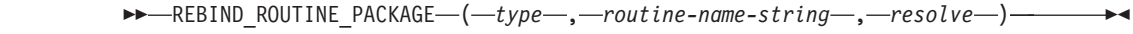
sarblob 引数の内容は SQL アーカイブ・ファイルを構成する個々のファイルへ抽出されます。共有ライブラリーとバインド・ファイルは、一時ディレクトリーのファイルに書き込まれます。環境は、コンパイルおよびリンクが不要であること、共有ライブラリーとバインド・ファイルのロケーションが使用可能であることを、ルーチン定義ステートメント処理に認識させるように設定されます。 DDL ファイルの内容が、ルーチン定義ステートメントを動的に実行するために使用されます。

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 (あるいは対応するプロシージャー) を正常に実行できなかった場合、問題の原因についての情報を提供する診断テキストともに、常にエラーが返されます (SQLSTATE 38000)。例えば、GET ROUTINE に与えられているプロシージャー名が SQL プロシージャーを識別していない場合、問題の原因を示す診断テキスト "-204, 42704" (ここで "-204" と "42704" はそれぞれ SQLCODE と SQLSTATE) が返されます。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



スキーマは SYSPROC です。

REBIND_ROUTINE_PACKAGE プロシージャーは、SQL プロシージャ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。

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に、デフォルトは CURRENT SCHEMA 特殊レジスターの値です。 *routine-name-string* に二重引用符 (") を組み込むことはできません。

resolve
どのバインディング・セマンティクスを使用するかを指定する、タイプ VARCHAR(12) の入力引数。 'ANY' 値は、SQL パスのすべての可能な一致が、オブジェクト解決に SQL パスを使用するどのオブジェクトの参照の解決にも考慮されることを示します。 'CONSERVATIVE' 値は、最後の明示的なバインドのタイム・スタンプより前に定義されたオブジェクトだけが、解決に考慮されることを示します。

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。

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。

例:

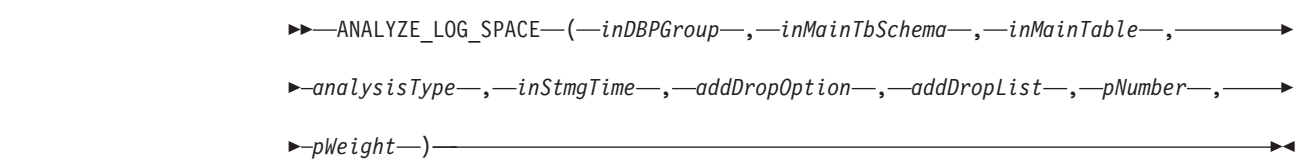
```
CALL SYSPROC.SET_ROUTINE_OPTS(CAST (NULL AS VARCHAR(1)))
```

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。

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。

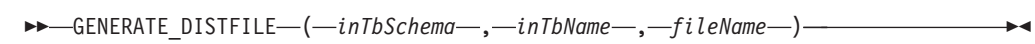
表 183. ANALYZE_LOG_SPACE プロシージャーによって戻される情報

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管します。

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管され、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')"
```

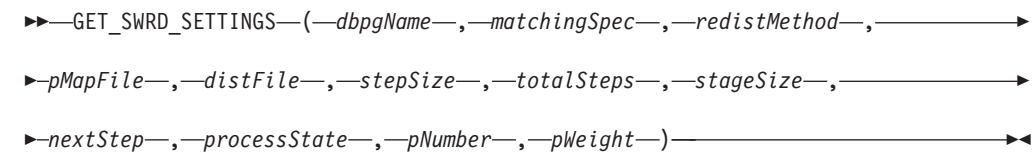
使用上の注意

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 184 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、他の出力パラメーターは NULL にすることができます。

表 184. 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, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)
```

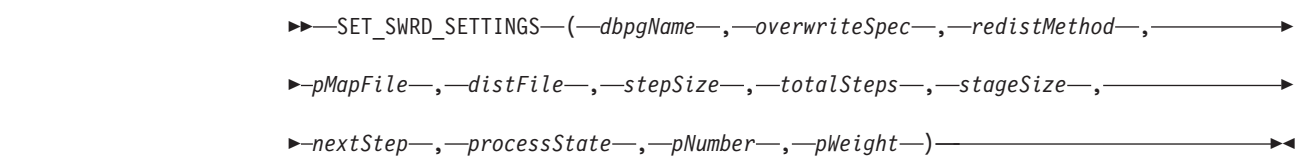
使用上の注意

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 185 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 185. 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')
```

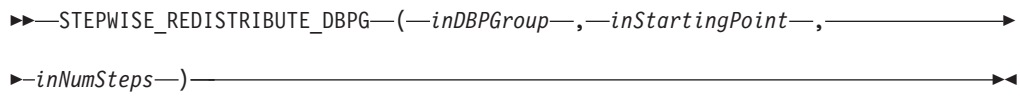
使用上の注意

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。たとえば、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)
```

ステップ単位の再配分プロシージャーの詳細な使用例については、「*パーティションおよびクラスタリングのガイド*」の『STEPWISE_REDISTRIBUTE_DBPG プロシージャー』を参照してください。

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。

ストレージ管理ツール・ルーチン

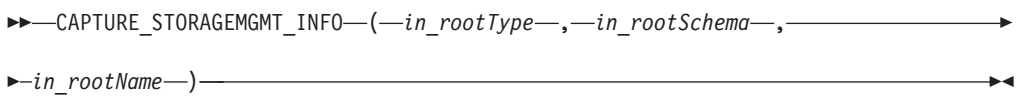
CAPTURE_STORAGEMGMT_INFO プロシージャー - 特定ルート・オブジェクトのストレージ関連情報の検索

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 186. 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
- 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。

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、「SYSTOOLS」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、表スペースに対する USE 特権、および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)
- SYSADM または DBADM 権限

使用上の注意

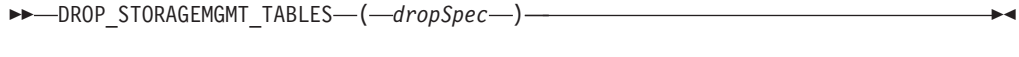
以下の表が SYSTOOLS スキーマで作成されます。

- 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。

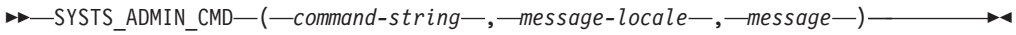
- SYSADM または DBADM 権限
- これらの表のスキーマに対する DROPIN 特権
- これらの表に対する CONTROL 特権

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
- 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
  ```

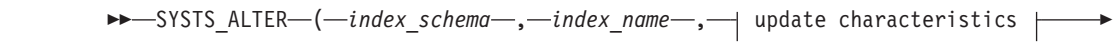
- SQLCODE がプロシージャーにより戻される場合、メッセージは切り捨てられる可能性があります。詳細なメッセージ情報は db2diag.log ファイル内にあります。

SYSTS_ALTER プロシージャー - 索引の更新特性の変更

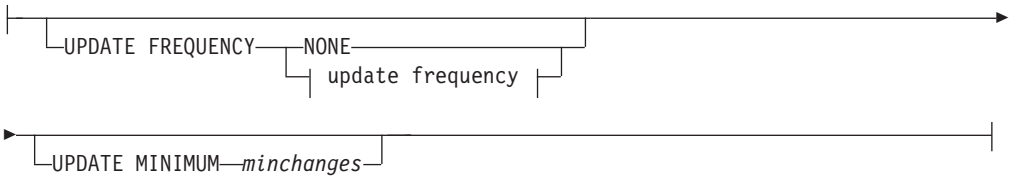
このプロシージャーは、索引の更新特性を変更します。

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。

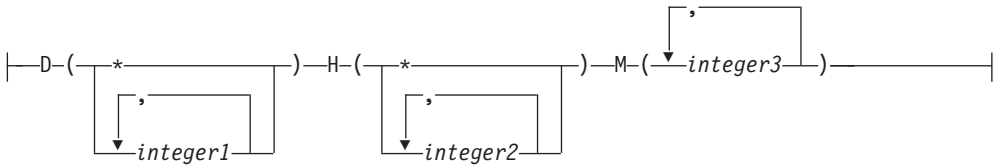
構文



update characteristics:



update frequency:



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がある表内のテキスト列に役立てることができます。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。自動更新は、START FOR TEXT コマンドが実行されており、DB2 Text Search インスタンス・サービスが実行中の場合にのみ実行できます。

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
テキスト索引に適用される自動更新はありません。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。

***** 週のすべての曜日。

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
その時間ちょうど (0)、またはその時間以後の 5 分刻みの増分で指定 (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*
索引がインクリメント更新される前に生じる必要がある、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。

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

SYSTS_ALTER プロシージャーに対する EXECUTE 特権。

データベース接続の許可名には、テキスト検索索引が作成された表に対する CONTROL 特権がなければなりません。

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、AUTOCOMMIT をオンにすることです。

  注: テキスト検索管理操作を実行する前にトランザクションをコミットしておかないと、同じデータベース・オブジェクトに変更が加えられる場合に、操作が直前のステップのコミットを待機することになって、デッドロックが発生する可能性があります。例えば、AUTOCOMMIT がオフになっている場合、表を作成した後にテキスト索引を作成し、この 2 つのトランザクションの間に明示的にコミットを行わないなら、これが発生する可能性があります。
- 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。

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

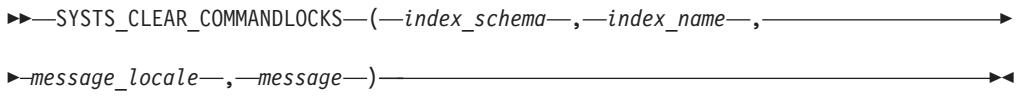
このプロシージャーは、データベース内の特定のテキスト検索索引またはすべてのテキスト検索索引のすべてのコマンド・ロックを解除します。

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。これにより異なるコマンド間の望ましくない競合が回避されます。

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。これを実行することで、テキスト検索索引は新規検索要求にアクセス可能となります。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

SYSTS_CLEAR_COMMANDLOCKS プロシージャーに対する EXECUTE 特権。

index_name が引数として指定されない場合、データベース接続の許可名には、DBADM または SYSADM 権限がなければなりません。

index_name が引数として指定されている場合、データベース接続の許可名には、(DB2 カタログ・ビューに従って) テキスト検索索引が作成された表に対する CONTROL 特権がなければなりません。

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。したがって、テキスト検索管理

プロシージャー内の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、プロシージャーを実行する前に、すべてのトランザクション変更をコミットすることをお勧めします。これを行う 1 つの方法は、AUTOCOMMIT をオンにすることです。

注: テキスト検索管理操作を実行する前にトランザクションをコミットしておかないと、同じデータベース・オブジェクトに変更が加えられる場合に、操作が直前のステップのコミットを待機することになって、デッドロックが発生する可能性があります。例えば、AUTOCOMMIT がオフになっている場合、表を作成した後にテキスト索引を作成し、この 2 つのトランザクションの間に明示的にコミットを行わないなら、これが発生する可能性があります。

- 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。

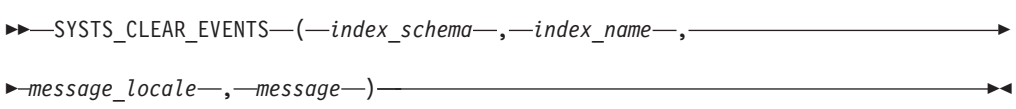
SYSTS_CLEAR_EVENTS プロシージャー - 索引のイベント表からの索引付けイベントの削除

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すべての索引更新操作で、情報項目、および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には、イベントをクリアできます (むしろスペースを消費しすぎないように、そうする必要があります)。

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

SYSTS_CLEAR_EVENTS プロシージャーに対する EXECUTE 特権。

データベース接続の許可名には、テキスト検索索引が作成された表に対する CONTROL 特権がなければなりません。

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。したがって、テキスト検索管理プロシージャー内の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、プロシージャーを実行する前に、すべてのトランザクション変更をコミットすることをお勧めします。これを行う 1 つの方法は、AUTOCOMMIT をオンにすることです。

  注: テキスト検索管理操作を実行する前にトランザクションをコミットしておかないと、同じデータベース・オブジェクトに変更が加えられる場合に、操作が直前のステップのコミットを待機することになって、デッドロックが発生する可能性があります。例えば、AUTOCOMMIT がオフになっている場合、表を作成した後にテキスト索引を作成し、この 2 つのトランザクションの間に明示的にコミットを行わないなら、これが発生する可能性があります。
- 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。

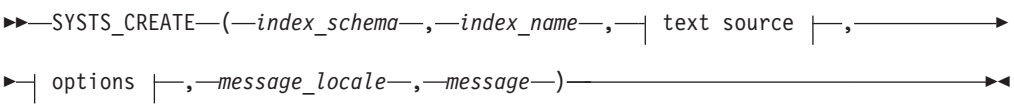
SYSTS_CREATE プロシージャー - 列のテキスト検索索引の作成

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。

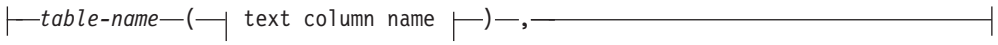
テキスト検索索引を作成すると、照会でテキスト検索関数を使用して列を検索できます。テキスト検索 UPDATE INDEX コマンドまたは SYSTS_UPDATE プロシージャーがユーザーにより明示的に実行されるまで、または索引に定義された更新頻度に従ってテキスト検索インスタンス・レベル・サービスにより暗黙に実行されるまで、索引にデータが入ることはありません。

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。

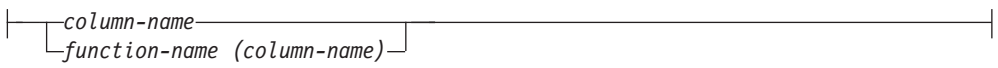
構文



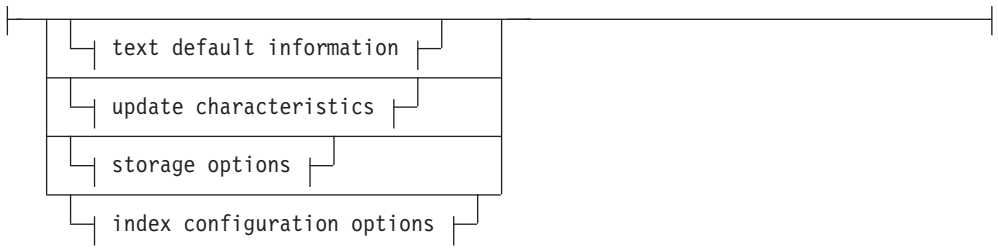
text source:



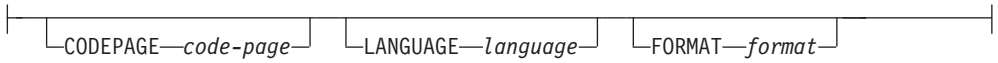
text column name:



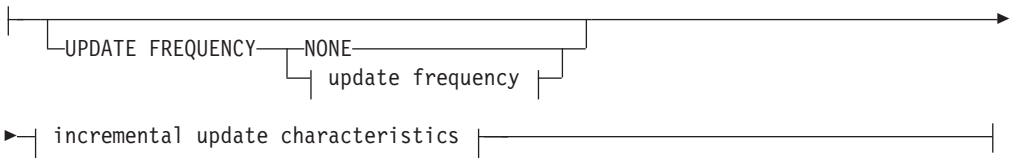
オプション:



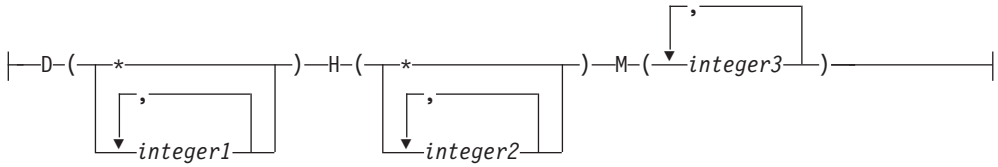
text default information:



update characteristics:

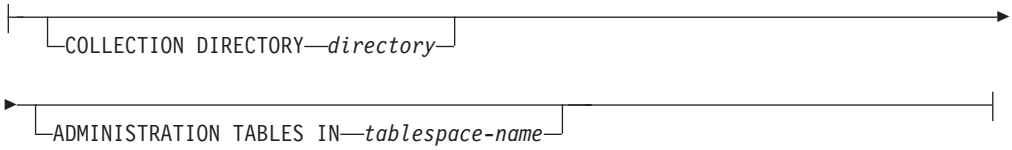


update frequency:

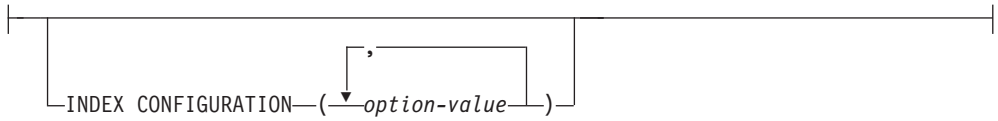


incremental update characteristics:

storage options:



index configuration 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。

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。テキスト検索索引は、以下の表には作成できません。

- 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
索引付けされる列の列名。

column-name
列は、CHAR、VARCHAR、LONG VARCHAR、CLOB、DBCLOB、BLOB、GRAPHIC、VARGRAPHIC、LONG VARGRAPHIC、または XML のいずれかのデータ・タイプでなければなりません。

注: データ・タイプ LONG VARCHAR と LONG VARGRAPHIC は、引き続きサポートされていますが、非推奨になっており、将来のリリースで除去される可能性があります。
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*
索引付け時に文書の言語固有の処理のために DB2 テキスト検索が使用す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、索引の更新がこれ以上行われないことを示します。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。自動更新は、START FOR TEXT コマンドが実行されており、DB2 Text Search インスタンス・サービスが実行中の場合にのみ実行できます。

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。

***** 週のすべての曜日。

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。これらの値は単一引用符で囲む必要があります。

注: 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。以下の値がサポートされています。

表 187. オプション値の仕様

オプション	**指定可能な値 (デフォルト)**	**意味**
COMMENT	512 バイトより短いストリング値。	ストリング・コメント値を、DB2 テキスト検索カタログ・ビュー TSINDEXES の REMARKS 列に追加します。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。

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

SYSTS_CREATE プロシージャーに対する EXECUTE 特権。

データベース接続の許可名には、テキスト検索索引が作成される表に対する CONTROL 特権がなければなりません。

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、AUTOCOMMIT をオンにすることです。

  注: テキスト検索管理操作を実行する前にトランザクションをコミットしておかないと、同じデータベース・オブジェクトに変更が加えられる場合に、操作が直

前のステップのコミットを待機することになって、デッドロックが発生する可能性があります。例えば、AUTOCOMMIT がオフになっている場合、表を作成した後にテキスト索引を作成し、この 2 つのトランザクションの間に明示的にコミットを行わないなら、これが発生する可能性があります。

- 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 テキスト検索サーバー・データが更新されます。以下の例で示すように、名前の集合 *instance_database-name_index-identifier_number* 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。
  - 新しく作成されたテキスト検索索引には、自動的にデータが追加されることはありません。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 により許可される主キー列の数より 2 列少ない数に制限されます。
- 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。

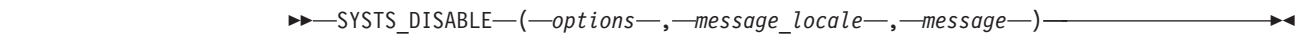
SYSTS_DISABLE プロシージャー - テキスト検索の現行データベースを使用不可にする

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

SYSTS_DISABLE プロシージャーに対する EXECUTE 特権。

データベース接続の許可名には、DBADM または SYSADM 権限がなければなりません。

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。

  注: テキスト検索管理操作を実行する前にトランザクションをコミットしておかないと、同じデータベース・オブジェクトに変更が加えられる場合に、操作が直前のステップのコミットを待機することになって、デッドロックが発生する可能性があります。例えば、AUTOCOMMIT がオフになっている場合、表を作成した後にテキスト索引を作成し、この 2 つのトランザクションの間に明示的にコミットを行わないなら、これが発生する可能性があります。
- 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。加えて、テキスト・サービスは更新され、残りの更新スケジュール情報は削除されます。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。
- 一部の索引が FORCE オプションを使用して削除できない場合、集合名が `db2diag.log` 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。オーファン集合の原因となるいくつかのケースを以下に示します。
  – 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。
  – その他のエラー状態。 CLEANUP FOR TEXT コマンドは一部のシナリオで使用できます。

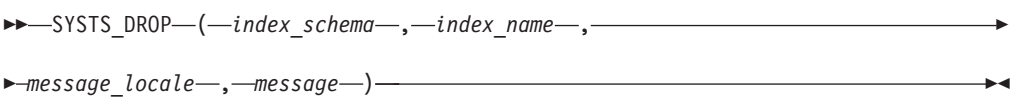
SYSTS_DROP プロシージャー - テキスト検索索引のドロップ

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

SYSTS_DROP プロシージャーに対する EXECUTE 特権。

データベース接続の許可名には、以下のいずれか 1 つがなければなりません。

- テキスト検索索引が作成された表に対する CONTROL 特権。
- DBADM 権限または SYSADM 権限。この許可は、FORCE オプションを指定した DISABLE DATABASE FOR TEXT コマンドの一部として、テキスト検索をドロップするために使用します。

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。したがって、テキスト検索管理プロシージャー内の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、プロシージャーを実行する前に、すべてのトランザクション変更をコミットすることをお勧めします。これを行う 1 つの方法は、AUTOCOMMIT をオンにすることです。

  注: テキスト検索管理操作を実行する前にトランザクションをコミットしておかないと、同じデータベース・オブジェクトに変更が加えられる場合に、操作が直前のステップのコミットを待機することになって、デッドロックが発生する可能性があります。例えば、AUTOCOMMIT がオフになっている場合、表を作成した後にテキスト索引を作成し、この 2 つのトランザクションの間に明示的にコミットを行わないなら、これが発生する可能性があります。
- 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

SYSTS_ENABLE プロシージャーに対する EXECUTE 特権。

データベース接続の許可名には、SYSADM 権限が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。したがって、テキスト検索管理プロシージャー内の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、プロシージャーを実行する前に、すべてのトランザクション変更をコミットすることをお勧めします。これを行う 1 つの方法は、AUTOCOMMIT をオンにすることです。

  注: テキスト検索管理操作を実行する前にトランザクションをコミットしておかないと、同じデータベース・オブジェクトに変更が加えられる場合に、操作が直前のステップのコミットを待機することになって、デッドロックが発生する可能性があります。例えば、AUTOCOMMIT がオフになっている場合、表を作成した後にテキスト索引を作成し、この 2 つのトランザクションの間に明示的にコミットを行わないなら、これが発生する可能性があります。
- 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。

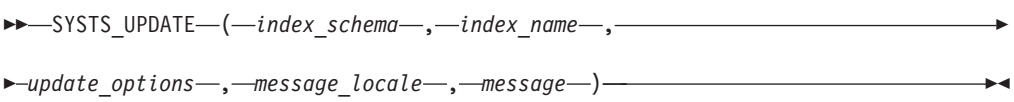
SYSTS_UPDATE プロシージャー - テキスト検索索引の更新

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。

更新が実行中でも、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。使用できる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。
- 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

SYSTS_UPDATE プロシージャーに対する EXECUTE 特権。

データベース接続の許可名には、テキスト検索索引が作成された表に対する CONTROL 特権がなければなりません。

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。索引の集合名は、SYSIBMTS.TSCOLLECTIONNAMES ビュー (列 COLLECTIONNAME) から取得できます。
- 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。

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)
```

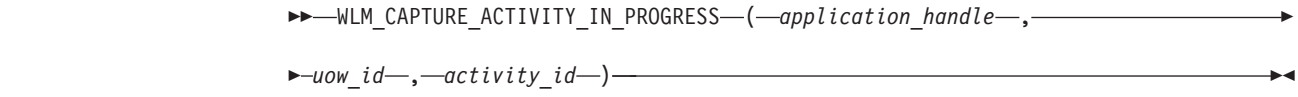
使用上の注意

- 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。
- 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

このプロシージャーにより、指定したアクティビティーに関する情報が収集され、アクティブなアクティビティー・イベント・モニターに書き込まれます。このプロシージャーが子アクティビティーを持つアクティビティーに適用される場合、これは最低のレベルに至るまでの間、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。このプロシージャーが呼び出されるとすぐに、この情報が収集され、送信されます。アクティビティーが実行を完了するまで待機しません。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

application_handle
そのアクティビティーがキャプチャー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。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。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_CAPTURE_ACTIVITY_IN_PROGRESS プロシージャーに対する EXECUTE 特権。

例

特定のプロシージャー MYSCHEMA.MYSLOWSTP が通常より遅く実行しているとします。ユーザーは苦情を漏らし、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アクティビティーをキャプチャーする能力を持っています。

管理者は、DB2ACTIVITIES という名前の、既存の DB2 アクティビティーのイベント・モニターが活動化されていると想定し、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 を取得できます。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すると、DB2ACTIVITIES という名前のアクティビティー・イベント・モニターについて、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検出できます。

```
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
```

表関数を使用する照会の例は次のとおりです。

```
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を収集するためにこのプロシージャーを使用している場合には、入力データ値は収集されません。

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

このプロシージャーでは、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が収集され、統計イベント・モニターに書き込まれます。また、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がリセットされます。アクティブな統計イベント・モニターがない場合、統計のみリセットされます。

構文

スキーマは SYSPROC です。

プロシージャー・パラメーター

wait
　このプロシージャーが統計収集とリセットの開始直後に戻るかどうかを指定す

る、タイプ CHAR のオプション入力引数。 'Y' を指定すると、すべての統計を統計イベント・モニター表に書き込んでフラッシュするまで、プロシージャーは戻りません。それ以外の場合、プロシージャーは、統計収集とリセットを開始した直後に戻ります。

statistics_timestamp
統計収集の開始のタイム・スタンプ値を返す、タイプ TIMESTAMP のオプション出力引数。

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、統計の収集とリセットを開始する。

```
CALL WLM_COLLECT_STATS()
```

以下はこの照会の出力例です。

```
Return Status = 0
```

例 2: WLM_COLLECT_STATS を呼び出して、統計を収集およびリセットするが、データが統計イベント・モニター表に書き込まれるまでは戻らないようにする。

```
CALL WLM_COLLECT_STATS('Y', ::collect_timestamp)
```

以下はこの照会の出力例です。

```
Return Status = 0
```

例 3: 別の呼び出しの進行中に、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、統計を手動で収集するために使用されます。これは、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 (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。プロシージャーが別の収集およびリセット要求の進行中と同時に呼び出される場合 (例えば、プロシージャーが別のプロシージャー呼び出しの実行中と同時に呼び出されたり、自動収集の発生時と同時に呼び出される)、警告 SQL1632W が SQLSTATE 01H53 とともに戻され、要求は無視されます。

非同期モードの場合、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プロセスが完了する前に戻ることがあります。つまり、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すことがあります。統計の収集およびリセットが発生する速度に応じて、WLM_COLLECT_STATS プロシージャーへの呼び出し

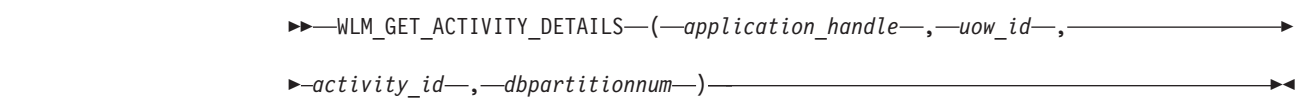
(これはアクティビティーそのもので、アクティビティー統計でカウントされる) は、前の収集間隔または開始された新規の収集間隔のいずれかでカウントされます。

同期モードの場合、WLM_COLLECT_STATS プロシージャーは、統計収集が完了し、すべての統計がアクティブな統計イベント・モニター用の表に書き込まれるまで戻りません。統計収集の開始のタイム・スタンプは、*statistics_timestamp* 出力パラメーターを介して返されます。

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

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。

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、709 ページの表 189 および 712 ページの表 190 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 188. 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 189 および 712 ページの表 190 を参照してください。
VALUE	VARCHAR(1024)	エレメント名。考えられる値については、表 189 および 712 ページの表 190 を参照してください。

表 189. 戻されるエレメント

エレメント名	説明
APPLICATION_HANDLE	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。
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。
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。
ACTIVITY_ID	アプリケーション内の固有のアクティビティー ID。
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し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。
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。

表 189. 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
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。
INVOCATION_ID	これは、このアクティビティーのある特定の呼び出しを同じネスト・レベルの他の呼び出しと区別します。アクティビティーがネストされていない場合にはゼロを返します。
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。
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。
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。
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。

表 189. 戻されるエレメント (続き)

エレメント名	説明
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。
ENTRY_TIME	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	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。
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。
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。
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。
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。
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。
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。

表 189. 戻されるエレメント (続き)

エレメント名	説明
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。 DPF システムでは、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。
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。 DPF システムでは、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。
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。
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。
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。

以下は、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 190. 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
CONCURRENTWORKLOADACTIVITIES_THRESHOLD_ID	しきい値の ID。
CONCURRENTWORKLOADACTIVITIES_THRESHOLD_VALUE	その値を超えると、しきい値を起動する値。
CONCURRENTWORKLOADACTIVITIES_THRESHOLD_VIOLATED	「はい」は、このアクティビティーがしきい値に違反したことを示します。「いいえ」は、このアクティビティーがしきい値に違反していないことを示します。
CONCURRENTDBCOORDACTIVITIES_DB_THRESHOLD_ID	しきい値の ID。
CONCURRENTDBCOORDACTIVITIES_DB_THRESHOLD_VALUE	その値を超えると、しきい値を起動する値。

表 190. 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_DB_THRESHOLD_QUEUED	アクティビティーがこのしきい値によってキューに入れられたかどうか。
CONCURRENTDBCOORDACTIVITIES_DB_THRESHOLD_VIOLATED	「はい」は、しきい値が違反されていることを示します。「いいえ」は、しきい値が違反されていないことを示します。
CONCURRENTDBCOORDACTIVITIES_WORK_ACTION_SET _THRESHOLD_ID	しきい値の ID。
CONCURRENTDBCOORDACTIVITIES_WORK_ACTION_SET _THRESHOLD_VALUE	その値を超えると、しきい値を起動する値。
CONCURRENTDBCOORDACTIVITIES_WORK_ACTION_SET _THRESHOLD_QUEUED	「はい」は、アクティビティーがこのしきい値によってキューに入れられたことを示します。「いいえ」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_VIOLATED	「はい」は、しきい値が違反されていることを示します。「いいえ」は、しきい値が違反されていないことを示します。
CONCURRENTDBCOORDACTIVITIES_SUPERCLASS _THRESHOLD_ID	しきい値の ID。
CONCURRENTDBCOORDACTIVITIES_SUPERCLASS _THRESHOLD_VALUE	その値を超えると、しきい値を起動する値。
CONCURRENTDBCOORDACTIVITIES_SUPERCLASS _THRESHOLD_QUEUED	「はい」は、アクティビティーがこのしきい値によってキューに入れられたことを示します。「いいえ」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_VIOLATED	「はい」は、しきい値が違反されていることを示します。「いいえ」は、しきい値が違反されていないことを示します。
CONCURRENTDBCOORDACTIVITIES_SUBCLASS _THRESHOLD_ID	しきい値の ID。
CONCURRENTDBCOORDACTIVITIES_SUBCLASS _THRESHOLD_VALUE	その値を超えると、しきい値を起動する値。
CONCURRENTDBCOORDACTIVITIES_SUBCLASS _THRESHOLD_QUEUED	アクティビティーがこのしきい値によってキューに入れられたかどうか。
CONCURRENTDBCOORDACTIVITIES_SUBCLASS _THRESHOLD_VIOLATED	「はい」は、しきい値が違反されていることを示します。「いいえ」は、しきい値が違反されていないことを示します。
ESTIMATEDSQLCOST_THRESHOLD_ID	しきい値の ID。
ESTIMATEDSQLCOST_THRESHOLD_VALUE	その値を超えると、しきい値を起動する値。
ESTIMATEDSQLCOST_THRESHOLD_VIOLATED	「1」は、しきい値が違反されていることを示します。「0」は、しきい値が違反されていないことを示します。
SQLTEMPSPACE_THRESHOLD_ID	しきい値の ID。

表 190. 適用される場合に戻されるエレメント (続き)

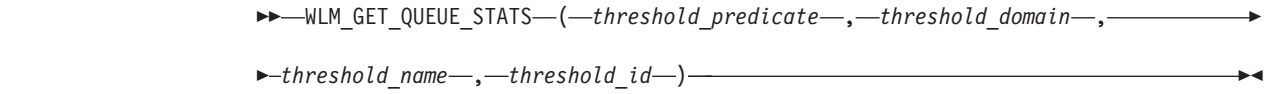
エレメント名	説明
SQLTEMPSPACE_THRESHOLD_VALUE	その値を超えると、しきい値を起動する値。
SQLTEMPSPACE_THRESHOLD_VIOLATED	「はい」は、しきい値が違反されていることを示します。「いいえ」は、しきい値が違反されていないことを示します。
SQLROWSRETURNED_THRESHOLD_ID	しきい値の ID。
SQLROWSRETURNED_THRESHOLD_VALUE	その値を超えると、しきい値を起動する値。
SQLROWSRETURNED_THRESHOLD_VIOLATED	「はい」は、しきい値が違反されていることを示します。「いいえ」は、しきい値が違反されていないことを示します。
ACTIVITYTOTALTIME_THRESHOLD_ID	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	「はい」は、しきい値が違反されていることを示します。「いいえ」は、しきい値が違反されていないことを示します。

WLM_GET_QUEUE_STATS 表関数 - しきい値キュー統計を戻す

この関数は、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。統計はすべてのアクティブ・パーティション上のキューについて戻されます。

構文



スキーマは SYSPROC です。

表関数パラメーター

threshold_predicate
有効なしきい値述部を指定する、タイプ VARCHAR(27) の入力引数。使用できる値は次のとおりです。

- 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

- NULL または空ストリング: 考えられるすべてのしきい値述部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
有効なしきい値ドメインを指定する、タイプ VARCHAR(18) の入力引数。使用できる値は次のとおりです。

- 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
- NULL または空ストリング: 考えられるすべてのしきい値ドメイン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
有効な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
有効なしきい値 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

システム上のすべてのキューのすべての基本統計をすべてのパーティションの間で表示するには、次のようにします。

```
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
```

以下はこの照会の出力例です。

```
THRESHNAME THRESHOLD_PREDICATE         THRESHOLD_DOMAIN   ...
---------- --------------------------- ------------------ ...
LIMIT1     CONCDBC                     DB                 ...
LIMIT2     SCCONN                      SP                 ...
LIMIT3     DBCONN                      DB                 ...
```

この照会からの出力 (続き)。

```
... 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

(パーティション上の) キューの間または (1 つ以上のキューの) パーティションの間の集約は実行されませんが、このタイプの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。

戻される情報

表 191. 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。使用できる値は次のとおりです。 • 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。使用できる値は次のとおりです。 • 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。

表 191. 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
DBPARTITIONNUM	SMALLINT	このレコードが収集されたパーティション番号。
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は NULL。
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は NULL。
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・セットでない場合は NULL。
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・セットでない場合は NULL。
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は NULL。
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。 • キュー統計が報告されているしきい値が変更され、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。
QUEUE_SIZE_TOP	INTEGER	最後のリセット以降に達した、キュー内の接続またはアクティビティーの最高数。

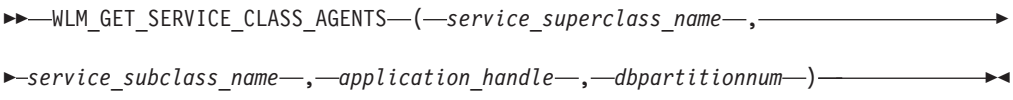
表 191. WLM_GET_QUEUE_STATS について戻される情報 (続き)

列名	データ・タイプ	説明
QUEUE_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。
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。これはミリ秒単位で測定されます。
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 - サービス・クラスで実行中のエージェントのリスト

この関数は、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 (db2fmps)、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
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、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
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、他のパラメーターが一致する、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
エージェント情報を戻さなければならないアプリケーション・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、他のパラメーターが一致する、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。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
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_CLASS_AGENTS 関数に対する EXECUTE 特権。

例

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
```

以下はこの照会の出力例です。

```
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
```

ここでは、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伝えています。

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 などの競合レコードを、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 192. 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	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)	この行のエンティティーのタイプがエージェントの場合、このフィールドには "db2agent" が表示されます。この行のエンティティーのタイプが fenced モード・プロセスの場合、このフィールドには "db2fmp (pid)" が表示されます。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、システム・エンティティーの名前が表示されます。
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。あるいは、アプリケーション・ハンドルをコーディネーター・データベース・パーティション番号の代わりに使用することもできます。
UOW_ID	INTEGER	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。
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。このアクティビティーに親がない場合、NULL を戻します。
PARENT_ACTIVITY_ID	INTEGER	親のアクティビティー ID が activity_id である、作業単位内の固有のアクティビティー。このアクティビティーに親がない場合、NULL を戻します。
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、このフィールドは NULL です。

表 192. *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
AGENT_TYPE	VARCHAR(32)	コーディネーターまたはサブエージェント。コーディネーターの場合、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。エージェント・タイプは、以下によって表されます。 • COORDINATOR • OTHER • PDBSUBAGENT • SMPSUBAGENT
SMP_COORDINATOR	INTEGER	エージェントが smp コーディネーターかどうか。「はい」の場合は 1、「いいえ」の場合は 0。
AGENT_SUBTYPE	VARCHAR(32)	使用できる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どうか。使用できる値は次のとおりです。 • 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。使用できる値は次のとおりです。 • ACQUIRE • PROCESS • WAIT この列で使用できる値についての詳細は、 724 ページの表 193 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。使用できる値は次のとおりです。 • COMPRESSION_DICTIONARY_BUILD • IMPLICIT_REBIND • INDEX_RECREATE • LOCK • LOCK_ESCALATION • QP_QUEUE • REMOTE_REQUEST • REQUEST • ROUTINE • WLM_QUEUE この列で使用できる値についての詳細は、 724 ページの表 193 を参照してください。

表 192. 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
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。使用できる値は次のとおりです。 • EXECUTING • IDLE この列で使用できる値についての詳細は、 724 ページの表 193 を参照してください。
REQUEST_ID	VARCHAR(64)	application_handle と組み合わせる場合にのみ固有です。これは、長時間かかる 1 つの要求がある場合と複数の要求がある場合とを区別するために使用できます。例えば、複数のフェッチを 1 つの長いフェッチと区別します。
REQUEST_TYPE	VARCHAR(32)	要求のタイプ。使用できる値は次のとおりです。 • 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> – OPEN – PREPARE – REBIND – REDISTRIBUTE – REORG – ROLLBACK – RUNSTATS • サブエージェント (DSS および SMP) の場合: – サブセクション番号がゼロ以外の場合に、"SUBSECTION:<subsection number>" の形式でサブセクション番号を表示します。それ以外の場合は、NULL を戻します。

表 192. *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)	• サブエージェント (RPC) の場合: – ABP – CATALOG – INTERNAL – REORG – RUNSTATS – WLM • サブエージェント (OTHER) の場合: – ABP – APP_RBSVPT – APP_RELSVPT – BACKUP – CLOSE – EXTERNAL_RBSVPT – EVMON – FORCE – FORCE_ALL – INTERNAL <number> – INTERRUPT – NOOP: 要求がない場合 – QP – REDISTRIBUTE – STMT_RBSVPT – STOP_USING – UPDATE_DBM_CFG – WLM 要求タイプが内部タイプのいずれかである場合、値は 'INTERNAL' の後に内部定数の実際の値が続く形で表示されます。
NESTING_LEVEL	INTEGER	これは、ID が activity_id である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。
INVOCATION_ID	INTEGER	これは、あるアクティビティーのある特定の呼び出しを同じネスト・レベルの他の呼び出しと区別します。
ROUTINE_ID	INTEGER	ルーチン固有 ID。ルーチンの一部でない場合は NULL。

注: EVENT_STATE、EVENT_TYPE、および EVENT_OBJECT 列の値の組み合わせとして可能なものを次の表にリストします。

表 193. EVENT_STATE、EVENT_TYPE、および EVENT_OBJECT 列の値の可能な組み合わせ

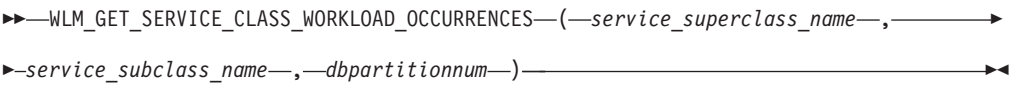
イベント記述	EVENT_STATE 値	EVENT_TYPE 値	EVENT_OBJECT 値
ロック獲得	IDLE	ACQUIRE	LOCK
ロックのエスカレート	EXECUTING	PROCESS	LOCK_ESCALATION
要求の処理	EXECUTING	PROCESS	REQUEST
新規要求の待機	IDLE	WAIT	REQUEST
要求のリモート・パーティションでの処理の待機	IDLE	WAIT	REMOTE_REQUEST
Query Patroller キューでの待機	IDLE	WAIT	QP_QUEUE
WLM しきい値キューでの待機	IDLE	WAIT	WLM_QUEUE
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。

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 -1、またはすべてのデータベース・パーティションに -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 などの競合レコードを、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 194. 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	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。
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。あるいは、アプリケーション・ハンドルをコーディネーター・データベース・パーティション番号の代わりに使用することもできます。

表 194. 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。

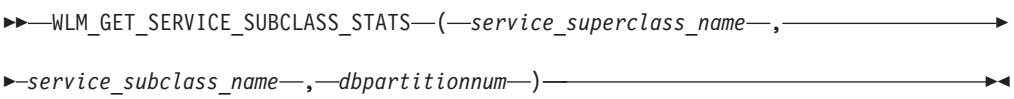
表 194. WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES について戻される情報 (続き)

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

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は実行前に DB2 サービス・クラスに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定期的にモニターすることができます (引数に NULL 値を渡すと、最後の引数である dbpartitionnum を除いて、その引数によって結果を制限しないことを表すことに注意してください。dbpartitionnum の場合、-2 はすべてのデータベース・パーティションからのデータが戻されることを意味します。) 次のステートメントは、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: 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
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
------------------ ------------------ ---- ---------
SYSDEFAULTUSERCLASS SYSDEFAULTSUBCLASS 0          10
SYSDEFAULTUSERCLASS SYSDEFAULTSUBCLASS 1           0
SYSDEFAULTUSERCLASS SYSDEFAULTSUBCLASS 2           0
SYSDEFAULTUSERCLASS SYSDEFAULTSUBCLASS 3           0
```

この表関数の出力では、平均実行時間およびアクティビティーの数を調べることにより、特定のデータベースの各パーティション上の「ロード」についての優れた高水準の見解を管理者に示します。これらの表関数によって戻される高水準ゲージの大きなバリエーションは、システム上でのロードの変更を示すことがあります。

使用上の注意

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にのみ、一部の統計が戻されます。

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・クラスの間または (1 つ以上のサービス・クラスの) パーティションの間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。

このパラメーターの影響として、ANDing されます。つまり、スーパークラス名 SUPA とサブクラス名 SUBB などの競合レコードを、SUBB が SUPA 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 195. 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。
DBPARTITIONNUM	SMALLINT	このレコードが収集されたパーティション番号。

表 195. 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
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。
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。
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。
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。

表 195. 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_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。
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。
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。
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。
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。

表 195. 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_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。
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。
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。
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。

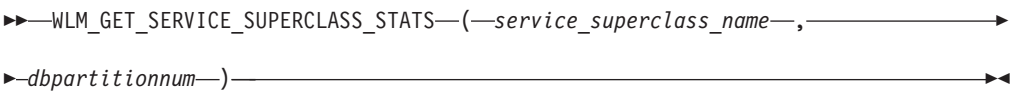
表 195. WLM_GET_SERVICE_SUBCLASS_STATS について戻される情報 (続き)

列名	データ・タイプ	説明
REQUEST_EXEC_TIME__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。

WLM_GET_SERVICE_SUPERCLASS_STATS - サービス・スーパークラスの統計を戻す

この関数は、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
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。

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_SUPERCLASS_STATS 関数に対する EXECUTE 特権。

例

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するには、次のようにします。

```
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
```

以下はこの照会の出力例です。

```
SERVICE_SUPERCLASS_NAME    DBPARTITIONNUM ...
-------------------------- -------------- ...
SYSDEFAULTSYSTEMCLASS                   0 ...
SYSDEFAULTMAINTENANCECLASS              0 ...
SYSDEFAULTUSERCLASS                     0 ...
```

この照会からの出力 (続き)。

```
... 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。 (パーティション上の) サービス・スーパークラスの間または (1 つ以上のサービス・スーパークラスの) パーティションの間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。

戻される情報

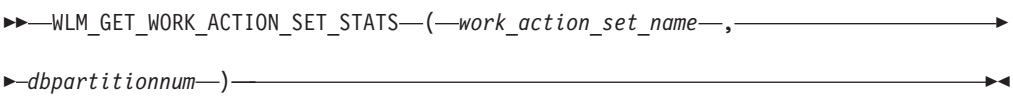
表 196. WLM_GET_SERVICE_SUPERCLASS_STATS について戻される情報

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。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・スーパークラスが変更され、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。
CONCURRENT_CONNECTION_TOP	INTEGER	最後のリセット以降にこのクラスで達した並行コーディネーター接続の最高数。

WLM_GET_WORK_ACTION_SET_STATS - 作業アクション・セット統計を戻す

この関数は、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
統計を戻す特定の作業アクション・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
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_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 では、ReadClass で現在実行中の (またはキューに入れられた) 8 つのアクティビティー、WriteClass で現在実行中の (またはキューに入れられた) 4 つのアクティビティー、LoadClass で現在実行中の (またはキューに入れられた) 2 つのアクティビティー、およびどの作業クラスにも割り当てられていない現在実行中の (またはキューに入れられた) 3 つのアクティビティーがあります。 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、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とともに人工的な“*” クラスでカウントされます。

```
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
```

以下はこの照会の出力例です。

```
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 197. 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)	アクション・セットの名前。この表に表示されるためには、設定された作業アクションが使用可能になっている必要があります。
DBPARTITIONNUM	SMALLINT	このレコードが収集されたパーティション番号。
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作業アクション・セットが変更され、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。
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。作業クラスをこの表に表示するには、この作業クラスに関連付けられた使用可能な作業アクションがなければなりません。 “*” は、ユーザーが 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 - アクティビティーのリストを戻す

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。

構文

```
►►—WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES—(—application_handle—,————————►
```

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、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アプリケーション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 -1、またはすべてのデータベース・パーティションに -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・ハンドルが識別されたら、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。例えば、管理者が list applications コマンドを使用して判別したアプリケーション・ハンドルが 1 であることがわかったアプリケーションのアクティビティーをリストするとします。

```
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
```

以下はこの照会の出力例です。

```
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 198. 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	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。
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 NULL にすることができます。
UOW_ID	INTEGER	アプリケーション内の固有の作業単位 ID。アクティビティーが開始した元の作業単位を表します。
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	アプリケーション内の固有の作業単位 ID。アクティビティーの親アクティビティーが開始し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、NULL を戻します。
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、またはそれがリモート・パーティションにある場合は、NULL を戻します。

表 198. 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)	使用できる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ときに、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、そのアクティビティーをサブミットしたユーザーは SQL4725N エラーを受け取ります。 • 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。 • QP_CANCEL_PENDING - CANCEL_PENDING 状態と同じですが、アクティビティーは 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ました。 • 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。この状態が表示されることは、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。 • 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。
ACTIVITY_TYPE	VARCHAR(32)	使用できる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 ステートメントの様々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『作業クラスの作業タイプおよび SQL ステートメント』を参照してください。

表 198. 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
NESTING_LEVEL	INTEGER	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。
INVOCATION_ID	INTEGER	これは、このアクティビティーのある特定の呼び出しを同じネスト・レベルの他の呼び出しと区別します。
ROUTINE_ID	INTEGER	ルーチン固有 ID。
UTILITY_ID	INTEGER	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 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	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、この列に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 が入っています。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、この列には NULL が入っています。
DATABASE_WORK_CLASS_ID	INTEGER	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、この列には作業クラスの ID が入っています。アクティビティーがデータベース有効範囲の作業クラスに分類されていない場合、この列には NULL が入っています。
SERVICE_CLASS_WORK_ACTION_SET_ID	INTEGER	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、この列に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 が入っています。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、この列には NULL が入っています。
SERVICE_CLASS_WORK_CLASS_ID	INTEGER	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、この列に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 が入っています。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、この列には NULL が入っています。

WLM_GET_WORKLOAD_STATS - ワークロード統計を戻す

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

```
►►—WLM_GET_WORKLOAD_STATS—(—workload_name—,—dbpartitionnum—)———————————►◄
```

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 199. 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 199. 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 199. 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。

WLM_SET_CLIENT_INFO プロシージャー - クライアント情報設定

このプロシージャーは、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、アプリケーションは、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、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、DB2 サーバーにおいて、この接続用に保管された、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値に変更されます。

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エンド・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。

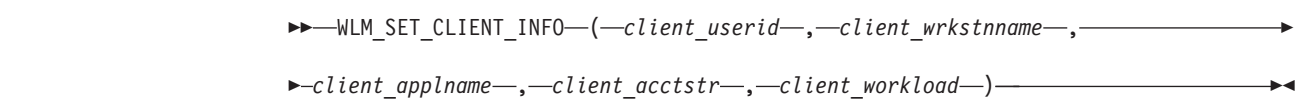
sqleseti API とは異なり、このプロシージャーは、クライアントでクライアント情報を設定するのではなく、対応するクライアント属性を DB2 サーバーで設定します。そのため、このプロシージャーを使用して DB2 サーバーに設定されたクライアント情報を照会するために sqleqry API を使用することはできません。

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、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。これらの切り捨てられた値は、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り返されます。

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 が指定される場合、値は変わりません。空ストリング (") が指定された場合、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 が指定される場合、値は変わりません。空ストリング (") が指定された場合、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 が指定される場合、値は変わりません。空ストリング (") が指定された場合、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
クライアントの会計情報ストリングを指定する、タイプ VARCHAR(255) の入力引数。 NULL が指定される場合、値は変わりません。空ストリング (") が指定された場合、クライアントの会計情報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 が指定される場合、値は変わりません。有効な値は次のとおりです。

- 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、DBADM または SYSADM 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: サーバーにより自動的に実行されるワークロード評価により選択されるワークロードにデータベース接続が割り当てられることを指定します。

  注: クライアント・ワークロードは大文字小文字を区別します。

注: *client_userid*、*client_wrkstnname*、*client_applname*、*client_acctstr* の場合、デフォルト値は空ストリングです。

許可

WLM_SET_CLIENT_INFO プロシージャーに対する EXECUTE 特権。

例

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、会計情報ストリング、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')
```

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)
```

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。

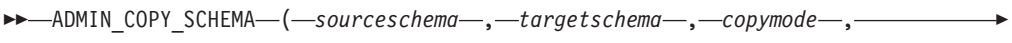
```
CALL SYSPROC.WLM_SET_CLIENT_INFO('', NULL, NULL, NULL, NULL)
```

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。

注: *copymode* が「COPY」または「COPYNO」の場合、完全修飾ファイル名 (例えば「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(2M) 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 200. 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。

表 200. ADMIN_COPY_SCHEMA errortab の形式 (続き)

列名	データ・タイプ	説明
DIAGTEXT	CLOB(2 K)	失敗した操作のエラー・メッセージ・テキスト。

許可

スキーマのコピーが成功するためには、CREATE_SCHEMA 特権のほかに、DB2 オブジェクト固有の特権も必要です。

例: 表をコピーするためには CREATE_TABLE 特権が必要であり、ADMIN_COPY_SCHEMA コマンドのもとで索引をコピーするためには 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
  - 配列型
  - 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。

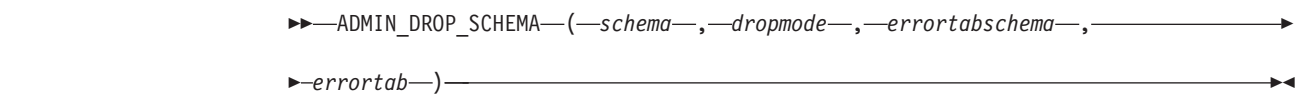
トランザクションの考慮事項

- 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 201. 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。
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
  - 配列型
  - 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。

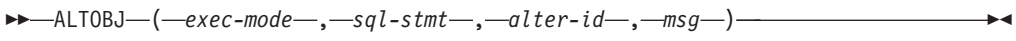
ALTOBJ

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、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。

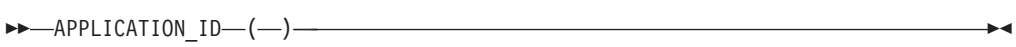
SYSTOOLSPACE をルーチンの操作表に使用してメタデータを保管します。つまり、データベース・オブジェクトとその操作の記述に使用されるデータです。

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。

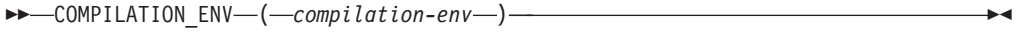
例

```
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
デッドロック・イベント・モニターによって提供されるコンパイル環境が入った、タイプ BLOB(2 M) の入力引数。

関数は、NAME VARCHAR(256) および VALUE VARCHAR(1024) という 2 つの列 (表 202 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、表 203 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'
```

戻される情報

表 202. 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、表 203 を参照してください。
VALUE	VARCHAR(1024)	エレメントの値。

表 203. 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。

表 203. 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
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。
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

CONTACTGROUPS 管理ビューに対する SELECT または CONTROL 特権、および ADMIN_GET_CONTACTGROUPS 表関数に対する EXECUTE 特権。

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 204. 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

CONTACTS 管理ビューに対する SELECT または CONTROL 特権、および ADMIN_GET_CONTACTS 表関数に対する EXECUTE 特権。

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 205. 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。例えば、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。

PRUNE HISTORY コマンドをデータベース・パーティションに使用して、DB_HISTORY ビューで返される情報の量を減らせます。LIST HISTORY コマンドを使用しても、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せます。

スキーマは SYSIBMADM です。

許可

DB_HISTORY 管理ビューに対する SELECT または CONTROL 特権、および ADMIN_LIST_HIST 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、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 206. 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。

表 206. 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
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。
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、 764 ページの表 207 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、 764 ページの表 207 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。

表 206. 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
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。
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。

表 206. 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
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英数字 5 文字が続きます。
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。
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 207. 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

表 207. OPERATION および OPERATIONTYPE 値 (続き)

操作値	操作値の説明	操作タイプ
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
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

表 207. OPERATION および OPERATIONTYPE 値 (続き)

操作値	操作値の説明	操作タイプ
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

DBPATHS 管理ビューに対する SELECT または CONTROL 特権、および ADMIN_LIST_DB_PATHS 表関数に対する EXECUTE 特権。

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機構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

ADMIN_LIST_DB_PATHS 表関数は、次のようにして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\
```

分割ミラー操作を行うストレージ・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は、次の 2 つの照会を使用して、ロー・デバイス上のすべての場所を表すリストをまず取得します。

```
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 208. DBPATHS 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。
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 209 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 209. 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。

表 209. TYPE 列の値 (続き)

タイプ値	説明
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。というのは、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。

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。

スカラー関数パラメーター

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
```

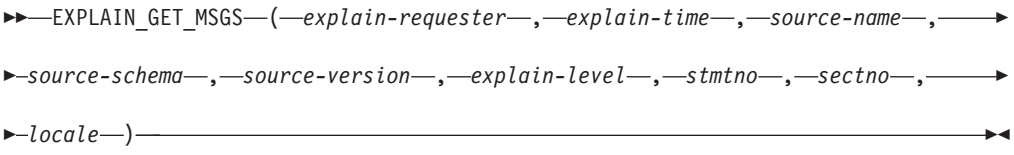
```
Base Table Statistics:
----------------------
Name  :  T1
Schema:  DB2USER2
  Number of Columns:                          3
  Number of Pages with Rows:                  1
  Number of Pages:                            1
  Number of Rows:                             5
  Table Overflow Record Count:                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_GET_MSGS



スキーマは Explain 表スキーマと同じです。

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。

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。

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。

この関数より、次に示されている表が戻されます。

表 210. 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
```

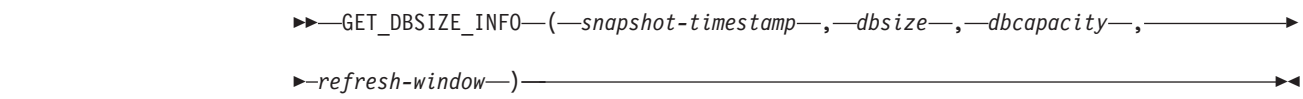
以下はこの照会の出力例です。

```
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".
```

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

NOTIFICATIONLIST 管理ビューに対する SELECT または CONTROL 特権、および HEALTH_GET_NOTIFICATION_LIST 表関数に対する EXECUTE 特権。

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

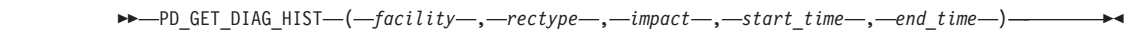
表 211. NOTIFICATIONLIST 管理ビューによって戻される情報

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。使用できる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。これは現在、db2diag.log、admin 通知ログ、および回転イベント・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。使用できる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。使用できる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 212. 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。使用できる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。これは現在、db2diag.log、admin 通知ログ、および回転イベント・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。
RECTYPE	VARCHAR(3)	レコードのタイプ。使用できる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。使用できる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。使用できる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。

表 212. 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
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のユニーク 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。
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」や「DB2 UDB」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 212. 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)。使用できる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 212. 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。使用できる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 212. 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。使用できる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 212. 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。使用できる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、 792 ページの表 213 を参照してください。

許可

PDLOGMSGS_LAST24HOURS 管理ビューに対する SELECT または CONTROL 特権、および PD_GET_LOG_MSGS 表関数に対する EXECUTE 特権。

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)。

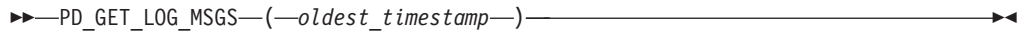
```
...  MSGSEVERITY MSG
...  ----------- -------------------------------------
...  C           ADM0504C  An unexpected internal
...              processing error has occurred.  ALL
...              DB2 PROCESSES ASSOCIATED WITH THIS
...              INSTANCE HAVE BEEN SUSPENDED.
...              Diagnostic 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、 792 ページの表 213 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, DBPARTITOINNUM, 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 213. 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。

表 213. 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
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)	メッセージを書き込んだ 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。
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)。

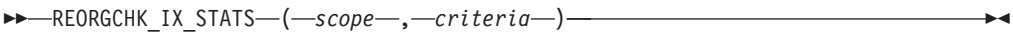
表 213. PDLOGMSGS_LAST24HOURS 管理ビューおよび PD_GET_LOG_MSGS 表関数によって戻される情報 (続き)

列名	データ・タイプ	説明
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 214. 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。
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。例えば、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、オーバーヘッドをすべて除外した索引項目の平均サイズであり、索引に関係するすべての列での列の平均の長さから計算されます。
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、オーバーヘッドをすべて除外した索引項目の平均サイズであり、索引に関係するすべての列 (すべての INCLUDE 列を除く) での列の平均の長さから計算されます。
LEAF_PAGE_OVERHEAD	BIGINT	内部使用に備えて索引リーフ・ページ上で予約されているスペース。
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。
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。

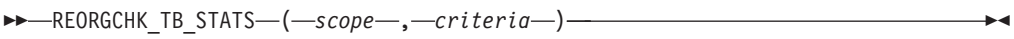
表 214. REORGCHK_IX_STATS プロシージャーによって戻される情報 (続き)

列名	データ・タイプ	説明
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 215. 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。
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。
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))
```

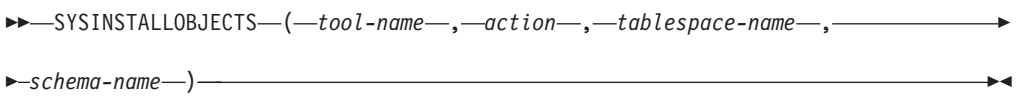
以下は戻される出力の例です。

```
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。

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、または値が空またはブランク・ストリングである場合に、ツール名が AM であると、デフォルトのユーザー・スペースが使用されます。それ以外の場合、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)))
```

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 216. MON_GET_FCM について戻される情報

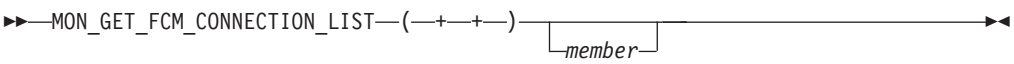
列名	データ・タイプ	説明または対応するモニター・エレメント
HOSTNAME	VARCHAR(128)	hostname - ホスト名
MEMBER	SMALLINT	member - データベース・メンバー
BUFF_MAX	BIGINT	buff_max - FCM バッファーの最大可能数
BUFF_TOTAL	BIGINT	buff_total - 現在割り振られている FCM バッファー数
BUFF_FREE	BIGINT	buff_free - 現在空いている FCM バッファー
BUFF_FREE_BOTTOM	BIGINT	buff_free_bottom - 空き FCM バッファーの最小数
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 217. 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。
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。

第 4 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 バージョン 9.5 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 9.1 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 バージョン 9.1 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' (_V95 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。

表 218. 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 9.1 の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**
809 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』	232 ページの『ADMINTABINFO 管理ビューおよび ADMIN_GET_TAB_INFO_V95 表関数 - 表のサイズおよび状態に関する情報の検索』
820 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』	390 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』
828 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』	382 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』
836 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』	399 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』
852 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』	411 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』
849 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』	427 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』
863 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』	437 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』

© Copyright IBM Corp. 2006, 2010

前のリリース、DB2 バージョン 9.1 でも、DB2 UDB for Linux, UNIX, and Windows バージョン 8 関数と取り替えられた新規関数がありました。

表 219. DB2 バージョン 9.1 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 UDB for Linux, UNIX, and Windows バージョン 8 の使用すべきでない関数	**DB2 バージョン 9.1 の新規の関数またはビュー**
818 ページの『GET_DB_CONFIG』	263 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』
819 ページの『GET_DBM_CONFIG』	265 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』
840 ページの『SNAP_GET_CONTAINER』	407 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』
841 ページの『SNAP_GET_DB』	852 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』
SNAP_GET_DETAILLOG (1)	434 ページの『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』
867 ページの『SNAP_GET_DYN_SQL』	863 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』
869 ページの『SNAP_GET_STO_PATHS』	471 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』
870 ページの『SNAP_GET_TAB』	480 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』
871 ページの『SNAP_GET_TBSP』	488 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』
874 ページの『SNAP_GET_TBSP_PART』	494 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』
877 ページの『SNAPSHOT_AGENT』	375 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』
878 ページの『SNAPSHOT_APPL』	820 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』
883 ページの『SNAPSHOT_APPL_INFO』	828 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』

表 219. DB2 バージョン 9.1 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 (続き)

DB2 UDB for Linux, UNIX, and Windows バージョン 8 の使用すべきでない関数	DB2 バージョン 9.1 の新規の関数またはビュー
885 ページの『SNAPSHOT_BP』	836 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』
888 ページの『SNAPSHOT_CONTAINER』	407 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』
889 ページの『SNAPSHOT_DATABASE』	852 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』
895 ページの『SNAPSHOT_DBM』	849 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』
897 ページの『SNAPSHOT_DYN_SQL』	863 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』
899 ページの『SNAPSHOT_FCM』	442 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』
900 ページの『SNAPSHOT_FCMNODE』	445 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』
901 ページの『SNAPSHOT_FILEW』	513 ページの『SNAP_WRITE_FILE プロシージャー』
902 ページの『SNAPSHOT_LOCK』	451 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』
903 ページの『SNAPSHOT_LOCKWAIT』	457 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』
905 ページの『SNAPSHOT_QUIESCERS』	499 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』
906 ページの『SNAPSHOT_RANGES』	503 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』
907 ページの『SNAPSHOT_STATEMENT』	464 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』
909 ページの『SNAPSHOT_SUBSECT』	473 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』
911 ページの『SNAPSHOT_SWITCHES』	477 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』

表 219. DB2 バージョン 9.1 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 (続き)

DB2 UDB for Linux, UNIX, and Windows バージョン 8 の使用すべきでない関数	**DB2 バージョン 9.1 の新規の関数またはビュー**
912 ページの『SNAPSHOT_TABLE』	480 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』
913 ページの『SNAPSHOT_TBREORG』	483 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』
915 ページの『SNAPSHOT_TBS』	488 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』
917 ページの『SNAPSHOT_TBS_CFG』	494 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』
SNAPSHOT_UTIL (1)	507 ページの『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』
SNAPSHOT_UTIL_PROG (1)	510 ページの『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』
919 ページの『SQLCACHE_SNAPSHOT』	863 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』。SQLCACHE_SNAPSHOT 表関数により戻される情報は、この新しいビューおよび表関数に組み込まれるようになりました。
SYSFUN.GROUPS (1)	このプロシージャーは使用すべきではありません。
SYSFUN.GROUPS_FOR_USER (1)	343 ページの『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』
SYSFUN.USER_GROUPS (1)	このプロシージャーは使用すべきではありません。
SYSFUN.USERS (1)	このプロシージャーは使用すべきではありません。
921 ページの『SYSINSTALLROUTINES』	このプロシージャーは使用すべきではありません。

注: (1) これらの関数は DB2 UDB for Linux, UNIX, and Windows バージョン 8 にはありましたが、資料では削除されました。

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、 232 ページの『ADMINTABINFO 管理ビューおよび ADMIN_GET_TAB_INFO_V95 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 (例えば索引や 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 (例えば、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 220. 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 220. 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 (例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 220. 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
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 (例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。

表 220. 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_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 (例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。

表 220. 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
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 (例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 220. 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 (例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。

表 220. 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
INPLACE_REORG_STATUS	VARCHAR(10)	表のインプレース表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。

表 220. 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
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 (例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。
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)。

表 220. 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_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。

GET_DB_CONFIG

注: このプロシージャーは使用すべきではなく、 263 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 221. 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。

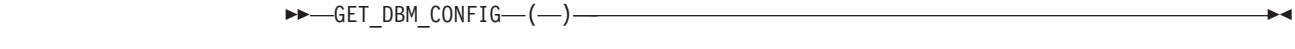
```
CONNECT TO SAMPLE

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、 265 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 222. 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.
```

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、 390 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。

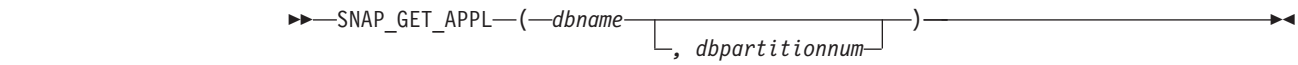
SNAP_GET_APPL 表関数は、アプリケーション・スナップショットから、特に appl 論理データ・グループのアプリケーション情報を戻します。

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、 822 ページの表 223 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 223. 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
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

表 223. 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_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
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

表 223. 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
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
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

表 223. 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
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
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

表 223. 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
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
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 (小数部分、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 (小数部分、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 223. 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
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 (小数部分、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 (小数部分、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

表 223. 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
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

* この列の基になっているモニター・エレメントの経過合計時間を計算するには、このモニター・エレメントの _S で終わっている列に報告されている秒の整数部分と、このモニター・エレメントの _MS で終わっている列に報告されている小数部分を合計しなければなりません。これは、次の計算式を使って行います。 (*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、 382 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、 830 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、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 224. 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)

表 224. 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
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 224. 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 224. 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 224. 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 224. 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 224. 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

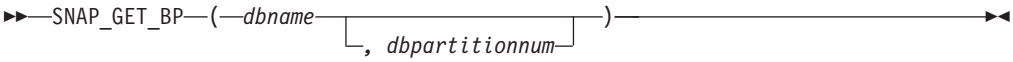
注: この表関数は使用すべきではなく、 399 ページの『SNAPBP 管理ビューおよび SNAP_GET_BP_V95 表関数 - bufferpool 論理グループのスナップショット情報の検索』に置き換えられました。

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、 837 ページの表 225 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 225. 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 225. 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 225. 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
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

表 225. 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
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、 407 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 226. 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、 852 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。

```
►►—SNAP_GET_DB—(—dbname—,—dbpartitionnum—)———————————————————►◄
```

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 227. 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
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

表 227. 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
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
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

表 227. 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_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
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

表 227. 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_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。
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 (小数部分、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

表 227. 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
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
TOT_LOG_USED_TOP	BIGINT	**tot_log_used_top** - 使用された最大合計ログ・スペース

表 227. 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
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 (小数部分、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 (小数部分、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

表 227. 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
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
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

表 227. 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

* この列の基になっているモニター・エレメントの経過合計時間を計算するには、このモニター・エレメントの _S で終わっている列に報告されている秒の整数部分と、このモニター・エレメントの _MS で終わっている列に報告されている小数部分を合計しなければなりません。これは、次の計算式を使って行います。 (*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

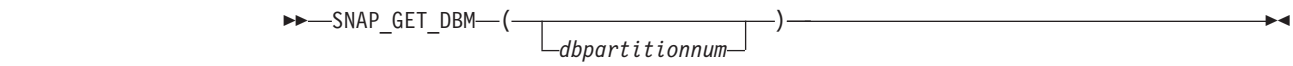
注: この表関数は使用すべきではなく、 427 ページの『SNAPDBM 管理ビューおよび SNAP_GET_DBM_V95 表関数 - dbm 論理グループ・スナップショット情報の検索』に置き換えられました。

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、 850 ページの表 228 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 228. 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
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

表 228. 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
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
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

表 228. 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
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、 411 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、 854 ページの表 229 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

- SYSMON 権限
- SNAP_GET_DB_V91 表関数に対する EXECUTE 特権。

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 229. 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
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 229. 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 • UNKNOWN • UNKNOWN_DRDA • VM_DRDA • VSE_DRDA • WINDOWS • WINDOWS95

表 229. 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・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
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 229. 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 229. 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 229. 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 (小数部分、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

表 229. 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
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 (小数部分、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 (小数部分、ナノ秒単位)†
NUM_LOG_WRITE_IO	BIGINT	num_log_write_io - ログ書き込み数

表 229. 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_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
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

表 229. 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_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 229. 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

* この列の基になっているモニター・エレメントの経過合計時間を計算するには、このモニター・エレメントの _S で終わっている列に報告されている秒の整数部分と、このモニター・エレメントの _MS で終わっている列に報告されている小数部分を合計しなければなりません。これは、次の計算式を使って行います。 (*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

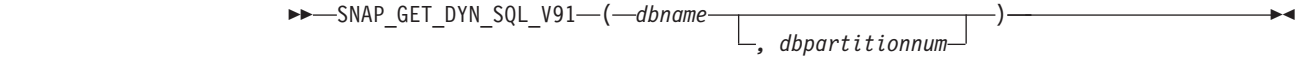
注: この表関数は使用すべきではなく、 437 ページの『SNAPDYN_SQL 管理ビューおよび SNAP_GET_DYN_SQL_V95 表関数 - dynsql 論理グループのスナップショット情報の検索』に置き換えられました。

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、 865 ページの表 230 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 230. 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
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 230. 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
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 (小数部分、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 (小数部分、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 (小数部分、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。

* この列の基になっているモニター・エレメントの経過合計時間を計算するには、このモニター・エレメントの _S で終わっている列に報告されている秒の整数部分と、このモニター・エレメントの _MS で終わっている列に報告されている小数部分を合計しなければなりません。これは、次の計算式を使って行います。 (*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、 863 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。

```
►►—SNAP_GET_DYN_SQL—(—dbname—,—dbpartitionnum—)————————————————►◄
```

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 231. 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

表 231. 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
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
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 (小数部分、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 (小数部分、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)*

表 231. 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
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 (小数部分、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

* この列の基になっているモニター・エレメントの経過合計時間を計算するには、このモニター・エレメントの _S で終わっている列に報告されている秒の整数部分と、このモニター・エレメントの _MS で終わっている列に報告されている小数部分を合計しなければなりません。これは、次の計算式を使って行います。 (*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_STO_PATHS

注: この表関数は使用すべきではなく、 471 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 232. 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、 480 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 233. 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、 488 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。

```
►►—SNAP_GET_TBSP—(—dbname—,—dbpartitionnum—)———————————————————►◄
```

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 234. 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
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

表 234. 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_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
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

表 234. 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
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
TBSP_USING_AUTO_STORAGE	SMALLINT	tablespace_using_auto_storage - 自動ストレージの使用
TBSP_AUTO_RESIZE_ENABLED	SMALLINT	tablespace_auto_resize_enabled - 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 0 以外であれば、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、 494 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。

```
►►—SNAP_GET_TBSP_PART—(—dbname—,—dbpartitionnum—)———————————————————►◄
```

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 235. 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
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

表 235. 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
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
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、 375 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。

```
►►—SNAPSHOT_AGENT—(—dbname—,—dbpartitionnum—)———————————————————►◄
```

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 236. 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、 820 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 237. 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

表 237. 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
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
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

表 237. 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
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
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

表 237. 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
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
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

表 237. 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
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
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 (小数部分、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 (小数部分、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

表 237. 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_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 (小数部分、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 (小数部分、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

* この列の基になっているモニター・エレメントの経過合計時間を計算するには、このモニター・エレメントの _S で終わっている列に報告されている秒の整数部分と、このモニター・エレメントの _MS で終わっている列に報告されている小数部分を合計しなければなりません。これは、次の計算式を使って行います。 (*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、 828 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 238. 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

表 238. 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
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
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、 836 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 239. 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

表 239. 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
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
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。

表 239. 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
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、 407 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 240. 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
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、 852 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。

```
►►—SNAPSHOT_DATABASE—(—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 241. 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

表 241. 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_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。
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

表 241. 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
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
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

表 241. 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
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
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

表 241. 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
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
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

表 241. 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
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
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、 849 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。

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 242. 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
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

表 242. 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
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
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、 863 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 243. 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

表 243. SNAPSHOT_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
[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、 442 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。

```
►►—SNAPSHOT_FCM—(—dbpartitionnum—)————————————————————————►◄
```

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 244. 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。

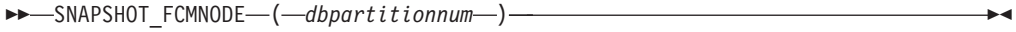
表 244. SNAPSHOT_FCM 表関数によって戻される情報 (続き)

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
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。
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、 445 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 245. 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、 513 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、 451 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。

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 246. 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。
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、 457 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 247. 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_ESCALLATION	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、 499 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 248. 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、 503 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。

```
►►—SNAPSHOT_RANGES—(—dbname—,—dbpartitionnum—)————————————————►◄
```

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 249. 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 249. 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、 464 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 250. 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 250. 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 (小数部分、マイクロ秒単位)*

表 250. 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 (小数部分、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 (小数部分、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

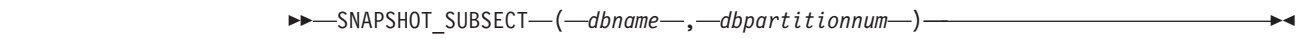
[1] STMT_TEXT は CLOB(16M) として定義されていますが、これは単に将来の拡張に備えるために過ぎません。ステートメント・テキストの実際の出力は 64K で切り捨てられます。

* この列の基になっているモニター・エレメントの経過合計時間を計算するには、このモニター・エレメントの _S で終わっている列に報告されている秒の整数部分と、このモニター・エレメントの _MS で終わっている列に報告されている小数部分を合計しなければなりません。これは、次の計算式を使って行います。 (*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、 473 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 251. 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 251. 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、 477 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 252. 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 252. 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、 480 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 253. 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、 483 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 254. 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 254. 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、 488 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。

```
►►—SNAPSHOT_TBS—(—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 255. SNAPSHOT_TBS 表関数によって戻される情報

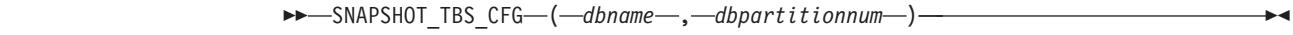
列名	データ・タイプ	説明または対応するモニター・エレメント
SNAPSHOT_TIMESTAMP	TIMESTAMP	スナップショットがとられた日時。
POOL_DATA_L_READS	BIGINT	**pool_data_l_reads** - バッファー・プール・データの論理読み取り
POOL_DATA_P_READS	BIGINT	**pool_data_p_reads** - バッファー・プール・データの物理読み取り
POOL_ASYNC_DATA_READS	BIGINT	**pool_async_data_reads** - バッファー・プール非同期データ読み取り
POOL_DATA_WRITES	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 255. 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、 494 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 256. 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
USABLE_PAGES	BIGINT	**tablespace_usable_pages** - 表スペース内の使用可能ページ数

表 256. 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
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、 863 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 257. 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 (小数部分、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 (小数部分、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 (小数部分、マイクロ秒単位)*
DB_NAME	VARCHAR(128)	db_name - データベース名
STMT_TEXT	CLOB(16M)[1]	stmt_text - SQL 動的ステートメント・テキスト

表 257. 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

[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。

* この列の基になっているモニター・エレメントの経過合計時間を計算するには、このモニター・エレメントの _S で終わっている列に報告されている秒の整数部分と、このモニター・エレメントの _MS で終わっている列に報告されている小数部分を合計しなければなりません。これは、次の計算式を使って行います。 (*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。

スキーマは SYSPROC です。

第 2 部 付録

© Copyright IBM Corp. 2006, 2010

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。

IBM Information Management 製品の使用をより容易にするために IBM にご協力いただける場合は、コンシューマビリティーに関する次のアンケートにご回答ください。http://www.ibm.com/software/data/info/consumability-survey/

© 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 9.5 のマニュアル (PDF 形式)は、それぞれ www.ibm.com/support/docview.wss?uid=swg27009727 および www.ibm.com/support/docview.wss?uid=swg27009728 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、以下の表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 258. 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	SC88-4431-03	入手可能	2010 年 12 月
管理ルーチンおよびビュー	SC88-4435-03	入手不可	2010 年 12 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	SC88-4433-03	入手可能	2010 年 12 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	SC88-4434-03	入手可能	2010 年 12 月
コマンド・リファレンス	SC88-4432-03	入手可能	2010 年 12 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	SC88-4421-03	入手可能	2010 年 12 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	SC88-4423-03	入手可能	2010 年 12 月
データ・サーバー、データベース、およびデータベース・オブジェクトのガイド	SC88-4259-03	入手可能	2010 年 12 月
データベース・セキュリティー・ガイド	SC88-4418-03	入手可能	2010 年 12 月
ADO.NET および OLE DB アプリケーションの開発	SC88-4425-02	入手可能	2009 年 4 月

表 258. 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
組み込み SQL アプリケーションの開発	SC88-4426-02	入手可能	2009 年 4 月
Java アプリケーションの開発	SC88-4427-03	入手可能	2010 年 12 月
Perl および PHP アプリケーションの開発	SC88-4428-02	入手不可	2009 年 4 月
SQL および外部ルーチンの開発	SC88-4429-03	入手可能	2010 年 12 月
データベース・アプリケーション開発の基礎	GC88-4430-03	入手可能	2010 年 12 月
DB2 インストールおよび管理 概説 (Linux および Windows 版)	GC88-4439-03	入手可能	2010 年 12 月
国際化対応ガイド	SC88-4420-02	入手可能	2009 年 4 月
メッセージ・リファレンス 第 1 巻	GI88-4109-00	入手不可	
メッセージ・リファレンス 第 2 巻	GI88-4110-00	入手不可	
マイグレーション・ガイド	GC88-4438-03	入手可能	2010 年 12 月
Net Search Extender 管理およびユーザーズ・ガイド	SC88-4630-02	入手可能	2009 年 4 月
パーティションおよびクラスタリングのガイド	SC88-4419-03	入手可能	2010 年 12 月
Query Patroller 管理およびユーザーズ・ガイド	SC88-4611-01	入手可能	2009 年 4 月
IBM データ・サーバー・クライアント機能 概説およびインストール	GC88-4441-03	入手不可	2010 年 12 月
DB2 サーバー機能 概説およびインストール	GC88-4440-03	入手可能	2010 年 12 月
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	SC88-4629-02	入手可能	2009 年 4 月
SQL リファレンス 第 1 巻	SC88-4436-03	入手可能	2010 年 12 月
SQL リファレンス 第 2 巻	SC88-4437-03	入手可能	2010 年 12 月

表 258. 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
システム・モニター ガイドおよびリファレンス	SC88-4422-03	入手可能	2010 年 12 月
Text Search ガイド	SC88-4710-02	入手可能	2010 年 12 月
問題判別ガイド	GI88-4108-03	入手不可	2010 年 12 月
データベース・パフォーマンスのチューニング	SC88-4417-03	入手可能	2010 年 12 月
Visual Explain チュートリアル	SC88-4449-00	入手不可	
新機能	SC88-4445-03	入手可能	2010 年 12 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	SC88-4446-03	入手可能	2010 年 12 月
pureXML ガイド	SC88-4447-03	入手可能	2010 年 12 月
XQuery リファレンス	SC88-4448-02	入手不可	2009 年 4 月

表 259. 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 概説およびインストール	GC88-4443-03	入手可能	2010 年 12 月
DB2 Connect サーバー機能 概説およびインストール	GC88-4444-03	入手可能	2010 年 12 月
DB2 Connect ユーザーズ・ガイド	SC88-4442-03	入手可能	2010 年 12 月

表 260. 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	SC88-4166-01	入手可能	
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	SC88-4167-02	入手可能	

表 260. *Information Integration* 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
Information Integration: フェデレーテッド・データ・ソース 構成ガイド	SC88-4185-01	入手不可	
Information Integration: SQL レプリケーション ガイドおよびリファレンス	SC88-4168-01	入手可能	
Information Integration: レプリケーションとイベント・パブリッシング 概説	GC88-4187-01	入手可能	

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/v9r5) 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 の「ご注文について」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、 926 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 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 状態ヘルプを呼び出すには、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. ブラウザー・キャッシュを消去してから、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. ブラウザー・キャッシュを消去してから、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らない場合があります。

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。非管理者および非 root の *DB2* インフォメーション・センター は常にスタンドアロン・モードで実行されます。を参照してください。

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。インストールする更新がある場合は、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は、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する必要があります。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 を再始動します。

注: Windows Vista の場合、下記の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・ツールを起動するには、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されている *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/db2icdv95 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。*DB2* インフォメーション・センター は、デフォルトで *Program_files*¥IBM¥DB2 Information Center¥Version 9.5 ディレクトリーにインストールされます。ここで、*Program_files* は Program Files ディレクトリーのロケーションを表します。
     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/V9.5 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 ブラウザーが起動し、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。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終了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終了しないでください。
   - 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終了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 スクリプトを終了しないでください。
7. *DB2 インフォメーション・センター* を再始動します。
   - 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/db2icdv95 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、「DB2 問題判別ガイド」、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。

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。

本書は、IBM 以外の Web サイトおよびリソースへのリンクまたは参照を含む場合があります。 IBM は、本書より参照もしくはアクセスできる、または本書からリンクされた IBM 以外の Web サイトもしくは第三者のリソースに対して一切の責任を負いません。 IBM 以外の Web サイトにリンクが張られていることにより IBM が当該 Web サイトを推奨するものではなく、またその内容、使用もしくはサイトの所有者について IBM が責任を負うことを意味するものではありません。 また、IBM は、お客様が IBM Web サイトから第三者の存在を知ることになった場合にも (もしくは、IBM Web サイトから第三者へのリンクを使用した場合にも)、お客様と第三者との間のいかなる取引に対しても一切責任を負いません。 従って、お客様は、IBM が上記の外部サイトまたはリソースの利用について責任を負うものではなく、また、外部サイトまたはリソースからアクセス可能なコンテンツ、サービス、

© Copyright IBM Corp. 2006, 2010

製品、またはその他の資料一切に対して IBM が責任を負うものではないことを承諾し、同意するものとします。 第三者により提供されるソフトウェアには、そのソフトウェアと共に提供される固有の使用条件が適用されます。

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。

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。

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]

インスタンス
  現在のインスタンスに関する情報の検索 269
インストール
  DB2 製品情報の取得 270
  DB2 製品ライセンス情報を戻す 272, 274
エラー・メッセージ
  検索
    SQLERRM スカラー関数 797
オブジェクト
  オブジェクト所有権の検索 348

[カ行]

関数
  サポートされている 7
  スカラー
    APPLICATION_ID 756
    GET_ROUTINE_OPTS 658
    MQPUBLISH 320
    MQREAD 322
    MQREADCLOB 327
    MQRECEIVE 328
    MQRECEIVECLOB 334
    MQSEND 335
    MQSUBSCRIBE 337
    MQUNSUBSCRIBE 338
    SQLERRM 797
  スカラー関数
    EXPLAIN_FORMAT_STATS 770
  ストアード・プロシージャー
    SYSTS_ALTER 677
    SYSTS_CLEAR_COMMANDLOCKS 681
    SYSTS_CLEAR_EVENTS 683
    SYSTS_DROP 695
    SYSTS_ENABLE 698
    SYSTS_UPDATE 699
  表
    MON_GET_FCM 801
    MON_GET_FCM_CONNECTION_LIST 802
  表関数 1
    管理ビューとの 5
    サポートされている 7
    使用すべきでない 805
    ADMIN_GET_MSGS 225
関数 (続き)
  表関数 (続き)
    ADMIN_GET_TAB_COMPRESS_INFO 226
    ADMIN_GET_TAB_INFO 809
    ADMIN_GET_TAB_INFO_V95 232
    AM_BASE_RPTS 22
    AM_BASE_RPT_RECOMS 21
    AUDIT_ARCHIVE 254
    AUTH_LIST_AUTHORITIES_FOR_AUTHID 340
    AUTH_LIST_GROUPS_FOR_AUTHID 343
    COMPILATION_ENV 756
    DB_PARTITIONS 262
    EXPLAIN_GET_MSGS 774
    GET_DBM_CONFIG 819
    GET_DB_CONFIG 818
    HEALTH_CONT_HI 277
    HEALTH_CONT_HI_HIS 279
    HEALTH_CONT_INFO 281
    HEALTH_DBM_HI 296
    HEALTH_DBM_HI_HIS 298
    HEALTH_DBM_INFO 300
    HEALTH_DB_HI 283
    HEALTH_DB_HIC 290
    HEALTH_DB_HIC_HIS 292
    HEALTH_DB_HI_HIS 286
    HEALTH_DB_INFO 295
    HEALTH_GET_ALERT_ACTION_CFG 301
    HEALTH_GET_ALERT_CFG 304
    HEALTH_GET_IND_DEFINITION 308
    HEALTH_TBS_HI 312
    HEALTH_TBS_HI_HIS 314
    HEALTH_TBS_INFO 318
    MQREADALL 323
    MQREADALLCLOB 325
    MQRECEIVEALL 329
    MQRECEIVEALLCLOB 332
    PD_GET_DIAG_HIST 780
    PD_GET_LOG_MSGS 787
    SNAPSHOT_AGENT (使用すべきでない) 877
    SNAPSHOT_APPL (使用すべきでない) 878
    SNAPSHOT_APPL_INFO (使用すべきでない) 883
    SNAPSHOT_BP (使用すべきでない) 886
    SNAPSHOT_CONTAINER (使用すべきでない) 888
    SNAPSHOT_DATABASE (使用すべきでない) 889
    SNAPSHOT_DBM (使用すべきでない) 895
    SNAPSHOT_DYN_SQL (使用すべきでない) 897
    SNAPSHOT_FCM (使用すべきでない) 899
    SNAPSHOT_FCMNODE (使用すべきでない) 900
    SNAPSHOT_LOCK (使用すべきでない) 902
    SNAPSHOT_LOCKWAIT (使用すべきでない) 903
    SNAPSHOT_QUIESCERS (使用すべきでない) 905

© Copyright IBM Corp. 2006, 2010

関数 *(続き)*
表関数 *(続き)*
SNAPSHOT_RANGES (使用すべきでない) 906
SNAPSHOT_STATEMENT (使用すべきでない) 907
SNAPSHOT_SUBSECT (使用すべきでない) 909
SNAPSHOT_SWITCHES (使用すべきでない) 911
SNAPSHOT_TABLE (使用すべきでない) 912
SNAPSHOT_TBREORG (使用すべきでない) 913
SNAPSHOT_TBS (使用すべきでない) 915
SNAPSHOT_TBS_CFG (使用すべきでない) 917
SNAP_GET_AGENT 375, 515
SNAP_GET_AGENT_MEMORY_POOL 378, 518
SNAP_GET_APPL 820
SNAP_GET_APPL_INFO 828
SNAP_GET_APPL_INFO_V95 382, 522
SNAP_GET_APPL_V95 390, 530
SNAP_GET_BP 836
SNAP_GET_BP_PART 404, 544
SNAP_GET_BP_V95 399, 539
SNAP_GET_CONTAINER (使用すべきでない) 840
SNAP_GET_CONTAINER_V91 407, 547
SNAP_GET_DB (使用すべきでない) 841
SNAP_GET_DBM 849
SNAP_GET_DBM_MEMORY_POOL 431, 571
SNAP_GET_DBM_V95 427, 567
SNAP_GET_DB_MEMORY_POOL 423, 563
SNAP_GET_DB_V91 852
SNAP_GET_DB_V95 411, 551
SNAP_GET_DETAIL_LOG_V91 434, 574
SNAP_GET_DYN_SQL (使用すべきでない) 867
SNAP_GET_DYN_SQL_V91 863
SNAP_GET_DYN_SQL_V95 437, 577
SNAP_GET_FCM 442, 582
SNAP_GET_FCM_PART 445, 585
SNAP_GET_HADR 447, 587
SNAP_GET_LOCK 451, 591
SNAP_GET_LOCKWAIT 457, 597
SNAP_GET_STMT 464, 604
SNAP_GET_STORAGE_PATHS 471, 611
SNAP_GET_STO_PATHS (使用すべきでない) 869
SNAP_GET_SUBSECTION 473, 613
SNAP_GET_SWITCHES 477, 617
SNAP_GET_TAB (使用すべきでない) 870
SNAP_GET_TAB_REORG 483, 623
SNAP_GET_TAB_V91 480, 620
SNAP_GET_TBSP (使用すべきでない) 871
SNAP_GET_TBSP_PART (使用すべきでない) 874
SNAP_GET_TBSP_PART_V91 494, 634
SNAP_GET_TBSP_QUIESCER 499, 639
SNAP_GET_TBSP_RANGE 503, 643
SNAP_GET_TBSP_V91 488, 628
SNAP_GET_UTIL 507, 647
SNAP_GET_UTIL_PROGRESS 510, 650
SQLCACHE_SNAPSHOT (使用すべきでない) 919
WLM_GET_ACTIVITY_DETAILS 707
WLM_GET_QUEUE_STATS 714
関数 *(続き)*
表関数 *(続き)*
WLM_GET_SERVICE_CLASS_AGENTS 718
WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES 724
WLM_GET_SERVICE_SUBCLASS_STATS 728
WLM_GET_SERVICE_SUPERCLASS_STATS 734
WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES 738
WLM_GET_WORKLOAD_STATS 741
WLM_GET_WORK_ACTION_SET_STATS 736
AUDIT_LIST_LOGS 256
管理 SQL ルーチン
サポートされている 7
管理タスク・スケジューラー
タスク・スケジュールの定義 243
管理ビュー
概要 1
許可 3
サポートされている 7
表関数との 5
ADMINTABINFO 232
ADMIN_TASK_LIST 245
ADMIN_TASK_STATUS 248
APPLICATIONS 351
APPL_PERFORMANCE 350
AUTHORIZATIONIDS 347
BP_HITRATIO 356
BP_READ_IO 358
BP_WRITE_IO 360
CONTACTGROUPS 759
CONTACTS 760
CONTAINER_UTILIZATION 361
DBCFG 263
DBMCFG 265
DBPATHS 766
DB_HISTORY 761
ENV_FEATURE_INFO 272
ENV_INST_INFO 269
ENV_PROD_INFO 270
ENV_SYS_INFO 273
ENV_SYS_RESOURCES 274
LOCKS_HELD 363
LOCKWAIT 366
LOG_UTILIZATION 370
LONG_RUNNING_SQL 371
NOTIFICATIONLIST 779
OBJECTOWNERS 348
PDLOGMSGS_LAST24HOURS 787
PRIVILEGES 349
QUERY_PREP_COST 374
REG_VARIABLES 267
SNAPAGENT 375, 515
SNAPAGENT_MEMORY_POOL 378, 518
SNAPAPPL 390, 530
SNAPAPPL_INFO 382, 522

管理ビュー *(続き)*
SNAPBP 399, 539
SNAPBP_PART 404, 544
SNAPCONTAINER 407, 547
SNAPDB 411, 551
SNAPDBM 427, 567
SNAPDBM_MEMORY_POOL 431, 571
SNAPDB_MEMORY_POOL 423, 563
SNAPDETAILLOG 434, 574
SNAPDYN_SQL 437, 577
SNAPFCM 442, 582
SNAPFCM_PART 445, 585
SNAPHADR 447, 587
SNAPLOCK 451, 591
SNAPLOCKWAIT 457, 597
SNAPSTMT 464, 604
SNAPSTORAGE_PATHS 471, 611
SNAPSUBSECTION 473, 613
SNAPSWITCHES 477, 617
SNAPTAB 480, 620
SNAPTAB_REORG 483, 623
SNAPTBSP 488, 628
SNAPTBSPPART 494, 634
SNAPTBSP_QUIESCER 499, 639
SNAPTBSP_RANGE 503, 643
SNAPUTIL 507, 647
SNAPUTIL_PROGRESS 510, 650
TBSP_UTILIZATION 655
TOP_DYNAMIC_SQL 657
管理ルーチン
AUTH_LIST_ROLES_FOR_AUTHID 345
AUTOMAINT_SET_POLICY プロシージャー 260
グループ
グループ・メンバーシップの取得 343
権限
管理ビューの 3
許可 ID の検索 347
グループ・メンバーシップの取得 343
更新
DB2 インフォメーション・センター 931
コピー
スキーマとオブジェクト 746
コマンド
プロシージャーから呼び出し 37, 676
ADD CONTACT 39
ADD CONTACTGROUP 41
AUTOCONFIGURE 42
BACKUP DATABASE 46
DESCRIBE 55
DROP CONTACT 66
DROP CONTACTGROUP 67
EXPORT 68
FORCE APPLICATION 80
GET STMM TUNING DBPARTITIONNUM 82
IMPORT 83
INITIALIZE TAPE 111
コマンド *(続き)*
LOAD 112
PRUNE HISTORY/LOGFILE 156
QUIESCE DATABASE 158
QUIESCE TABLESPACES FOR TABLE 160
REDISTRIBUTE DATABASE PARTITION GROUP 163
REORG INDEXES/TABLE 174
RESET ALERT CONFIGURATION 185
RESET DATABASE CONFIGURATION 187
RESET DATABASE MANAGER CONFIGURATION 189
REWIND TAPE 190
RUNSTATS 191
SET TAPE POSITION 203
UNQUIESCE DATABASE 204
UPDATE ALERT CONFIGURATION 205
UPDATE CONTACT 211
UPDATE CONTACTGROUP 212
UPDATE DATABASE CONFIGURATION 213
UPDATE DATABASE MANAGER CONFIGURATION 217
UPDATE HEALTH NOTIFICATION CONTACT LIST 219
UPDATE HISTORY 220
UPDATE STMM TUNING DBPARTITIONNUM 222
ご利用条件
資料の使用 934

［サ行］

サポートされている関数 7
システム情報
検索 273
使用すべきでない SNAPSHOT_AGENT 表関数 877
使用すべきでない SNAPSHOT_APPL 表関数 878
使用すべきでない SNAPSHOT_APPL_INFO 表関数 883
使用すべきでない SNAPSHOT_BP 表関数 886
使用すべきでない SNAPSHOT_CONTAINER 表関数 888
使用すべきでない SNAPSHOT_DATABASE 表関数 889
使用すべきでない SNAPSHOT_DBM 表関数 895
使用すべきでない SNAPSHOT_DYN_SQL 表関数 897
使用すべきでない SNAPSHOT_FCM 表関数 899
使用すべきでない SNAPSHOT_FCMNODE 表関数 900
使用すべきでない SNAPSHOT_FILEW プロシージャー 901
使用すべきでない SNAPSHOT_LOCK 表関数 902
使用すべきでない SNAPSHOT_LOCKWAIT 表関数 903
使用すべきでない SNAPSHOT_QUIESCERS 表関数 905
使用すべきでない SNAPSHOT_RANGES 表関数 906
使用すべきでない SNAPSHOT_STATEMENT 表関数 907
使用すべきでない SNAPSHOT_SUBSECT 表関数 909
使用すべきでない SNAPSHOT_SWITCHES 表関数 911
使用すべきでない SNAPSHOT_TABLE 表関数 912
使用すべきでない SNAPSHOT_TBREORG 表関数 913
使用すべきでない SNAPSHOT_TBS 表関数 915
使用すべきでない SNAPSHOT_TBS_CFG 表関数 917
使用すべきでない SNAP_GET_CONTAINER 表関数 840
使用すべきでない SNAP_GET_DB 表関数 841
使用すべきでない SNAP_GET_DYN_SQL 表関数 867
使用すべきでない SNAP_GET_STO_PATHS 表関数 869

使用すべきでない SNAP_GET_TAB 表関数 870
使用すべきでない SNAP_GET_TBSP 表関数 871
使用すべきでない SNAP_GET_TBSP_PART 表関数 874
使用すべきでない SQLCACHE_SNAPSHOT 表関数 919
使用すべきでない SYSINSTALLROUTINES プロシージャー 921
使用すべきでない機能
  表関数
    GET_DBM_CONFIG 819
    SNAPSHOT_AGENT 877
    SNAPSHOT_APPL 878
    SNAPSHOT_APPL_INFO 883
    SNAPSHOT_BP 886
    SNAPSHOT_CONTAINER 888
    SNAPSHOT_DATABASE 889
    SNAPSHOT_DBM 895
    SNAPSHOT_DYN_SQL 897
    SNAPSHOT_FCM 899
    SNAPSHOT_FCMNODE 900
    SNAPSHOT_LOCK 902
    SNAPSHOT_LOCKWAIT 903
    SNAPSHOT_QUIESCERS 905
    SNAPSHOT_RANGES 906
    SNAPSHOT_STATEMENT 907
    SNAPSHOT_SUBSECT 909
    SNAPSHOT_SWITCHES 911
    SNAPSHOT_TABLE 912
    SNAPSHOT_TBREORG 913
    SNAPSHOT_TBS 915
    SNAPSHOT_TBS_CFG 917
    SNAP_GET_APP 820
    SNAP_GET_BP 836
    SNAP_GET_CONTAINER 840
    SNAP_GET_DB 841
    SNAP_GET_DBM 849
    SNAP_GET_DB_V91 852
    SNAP_GET_DYN_SQL 867
    SNAP_GET_STO_PATHS 869
    SNAP_GET_TAB 870
    SNAP_GET_TBSP 871
    SNAP_GET_TBSP_PART 874
    SQLCACHE_SNAPSHOT 919
  プロシージャー
    GET_DB_CONFIG 818
    SNAPSHOT_FILEW 901
    SYSINSTALLROUTINES 921
  SQL 管理ルーチン 805
資料
  印刷 926
    注文 929
  概要 925
  使用に関するご利用条件 934
  PDF 926
スカラー関数
  SQLERRM 797
スキーマ
  スキーマとオブジェクトのコピー 746
  ドロップ、オブジェクト 751
ストアード・プロシージャー
  AUDIT_ARCHIVE 254
  AUDIT_DELIM_EXTRACT 255
ストレージ管理ツール
  ストアード・プロシージャー 673, 674, 675

[タ行]

チュートリアル
  トラブルシューティング 934
  問題判別 934
  Visual Explain 933
通知リスト
  連絡先リストの検索 779
通知ログ・メッセージ
  検索 787
データベース構成パラメーター
  検索 263
データベース・パス
  検索 766
データベース・マネージャー構成パラメーター
  検索、値 265
データを再配分する
  プロシージャー 663, 665, 666, 669, 671
特記事項 937
特権
  付与されたものに関する情報
    PRIVILEGES 管理ビュー 349
トラブルシューティング
  オンライン情報 934
  チュートリアル 934
ドロップ
  スキーマとそのオブジェクト 751

[ハ行]

ビュー
  管理ビュー
    ADMINTABCOMPRESSINFO 226
    ADMINTABINFO 232
    ADMIN_TASK_LIST 245
    ADMIN_TASK_STATUS 248
    APPLICATIONS 351
    APPL_PERFORMANCE 350
    AUTHORIZATIONIDS 347
    BP_HITRATIO 356
    BP_READ_IO 358
    BP_WRITE_IO 360
    CONTACTGROUPS 759
    CONTACTS 760
    CONTAINER_UTILIZATION 361
    DBCFG 263

ビュー *(続き)*
管理ビュー *(続き)*
DBMCFG 265
DBPATHS 766
DB_HISTORY 761
ENV_FEATURE_INFO 272
ENV_INST_INFO 269
ENV_PROD_INFO 270
ENV_SYS_INFO 273
ENV_SYS_RESOURCES 274
LOCKS_HELD 363
LOCKWAIT 366
LOG_UTILIZATION 370
LONG_RUNNING_SQL 371
NOTIFICATIONLIST 779
OBJECTOWNERS 348
PDLOGMSGS_LAST24HOURS 787
PRIVILEGES 349
QUERY_PREP_COST 374
REG_VARIABLES 267
SNAPAGENT 375, 515
SNAPAGENT_MEMORY_POOL 378, 518
SNAPAPPL 390, 530
SNAPAPPL_INFO 382, 522
SNAPBP 399, 539
SNAPBP_PART 404, 544
SNAPCONTAINER 407, 547
SNAPDB 411, 551
SNAPDBM 427, 567
SNAPDBM_MEMORY_POOL 431, 571
SNAPDB_MEMORY_POOL 423, 563
SNAPDETAILLOG 434, 574
SNAPDYN_SQL 437, 577
SNAPFCM 442, 582
SNAPFCM_PART 445, 585
SNAPHADR 447, 587
SNAPLOCK 451, 591
SNAPLOCKWAIT 457, 597
SNAPSTMT 464, 604
SNAPSTORAGE_PATHS 471, 611
SNAPSUBSECTION 473, 613
SNAPSWITCHES 477, 617
SNAPTAB 480, 620
SNAPTAB_REORG 483, 623
SNAPTBSP 488, 628
SNAPTBSPPART 494, 634
SNAPTBSP_QUIESCER 499, 639
SNAPTBSP_RANGE 503, 643
SNAPUTIL 507, 647
SNAPUTIL_PROGRESS 510, 650
TBSP_UTILIZATION 655
TOP_DYNAMIC_SQL 657
表
サイズおよび状態の検索 232, 809
表関数
管理ビューとの 5
表関数 *(続き)*
サポートされている 7
使用すべきでない機能 805
ADMIN_GET_TAB_INFO 809
SNAP_GET_APPL_INFO 828
SNAP_GET_BP 836
SNAP_GET_DBM 849
SNAP_GET_DB_V91 852
SNAP_GET_DYN_SQL_V91 863
ADMIN_GET_DBP_MEM_USAGE 223
ADMIN_GET_MSGS 225
ADMIN_GET_TAB_INFO_V95 232
AUDIT_ARCHIVE 254
AUTH_LIST_GROUPS_FOR_AUTHID 343
HEALTH_GET_ALERT_ACTION_CFG 301
HEALTH_GET_ALERT_CFG 304
HEALTH_GET_IND_DEFINITION 308
PD_GET_DIAG_HIST 780
PD_GET_LOG_MSGS 787
SNAP_GET_AGENT 375, 515
SNAP_GET_AGENT_MEMORY_POOL 378, 518
SNAP_GET_APPL_INFO_V95 382, 522
SNAP_GET_APPL_V95 390, 530
SNAP_GET_BP_PART 404, 544
SNAP_GET_BP_V95 399, 539
SNAP_GET_CONTAINER_V91 407, 547
SNAP_GET_DBM_MEMORY_POOL 431, 571
SNAP_GET_DBM_V95 427, 567
SNAP_GET_DB_MEMORY_POOL 423, 563
SNAP_GET_DB_V95 411, 551
SNAP_GET_DETAIL_LOG_V91 434, 574
SNAP_GET_DYN_SQL_V95 437, 577
SNAP_GET_FCM 442, 582
SNAP_GET_FCM_PART 445, 585
SNAP_GET_HADR 447, 587
SNAP_GET_LOCK 451, 591
SNAP_GET_LOCKWAIT 457, 597
SNAP_GET_STMT 464, 604
SNAP_GET_STORAGE_PATHS 471, 611
SNAP_GET_SUBSECTION 473, 613
SNAP_GET_SWITCHES 477, 617
SNAP_GET_TAB_REORG 483, 623
SNAP_GET_TAB_V91 480, 620
SNAP_GET_TBSP_PART_V91 494, 634
SNAP_GET_TBSP_QUIESCER 499, 639
SNAP_GET_TBSP_RANGE 503, 643
SNAP_GET_TBSP_V91 488, 628
SNAP_GET_UTIL 507, 647
SNAP_GET_UTIL_PROGRESS 510, 650
WLM_GET_QUEUE_STATS 714
プロシージャー
使用すべきでない機能 805
要約 7
ADMIN_CMD
説明 37
ADMIN_COPY_SCHEMA 746

プロシージャー *(続き)*
ADMIN_DROP_SCHEMA 751
ADMIN_REMOVE_MSGS 231
ADMIN_TASK_ADD 239
ADMIN_TASK_REMOVE 247
ADMIN_TASK_UPDATE 251
ALTOBJ 753
AM_DROP_TASK 24
AM_GET_LOCK_CHNS 25
AM_GET_LOCK_CHN_TB 24
AM_GET_LOCK_RPT 26
AM_GET_RPT 34
AM_SAVE_TASK 35
ANALYZE_LOG_SPACE 663
AUDIT_ARCHIVE 254
AUDIT_DELIM_EXTRACT 255
AUTOMAINT_GET_POLICY 257
AUTOMAINT_GET_POLICYFILE 258
AUTOMAINT_SET_POLICYFILE 261
CAPTURE_STORAGEMGMT_INFO 673
CREATE_STORAGEMGMT_TABLES 674
DROP_STORAGEMGMT_TABLES 675
GENERATE_DISTFILE 665
GET_DBSIZE_INFO 777
GET_ROUTINE_SAR 659
GET_SWRD_SETTINGS 666
HEALTH_HI_REC 310
PUT_ROUTINE_SAR 660
REBIND_ROUTINE_PACKAGE 661
REORGCHK_IX_STATS 794
REORGCHK_TB_STATS 796
SET_ROUTINE_OPTS 662
SET_SWRD_SETTINGS 669
SNAPSHOT_FILEW 901
SNAP_WRITE_FILE 513, 653
STEPWISE_REDISTRIBUTE_DBPG 671
SYSINSTALLOBJECTS 800
SYSINSTALLROUTINES 921
SYSTS_ADMIN_CMD 676
WLM_CANCEL_ACTIVITY 702
WLM_CAPTURE_ACTIVITY_IN_PROGRESS 703
WLM_COLLECT_STATS 705
WLM_SET_CLIENT_INFO 744
分割ミラー
データベース・パスの検索 766
ヘルス・アラート
アラート構成 304
アラート・アクション構成 301
ヘルス・インディケーター
ヘルス・インディケーター定義の検索 308
ヘルプ
言語の構成 930
SQL ステートメント 930

［マ行］

問題判別
チュートリアル 934
通知ログ・メッセージ 787
利用できる情報 934

［ラ行］

履歴ファイル
情報の検索 761
ルーチン
SQL 管理 805
サポートされている 7
レジストリー変数
使用中の設定の検索 267
連絡先
連絡先グループの検索 759
連絡先リストの検索 760
連絡先リスト
連絡先グループのリストの検索 759
連絡先の検索 760

A

ADD CONTACT コマンド
構文 (ADMIN_CMD を使用) 39
ADD CONTACTGROUP コマンド
構文 (ADMIN_CMD を使用) 41
ADMINTABCOMPRESSINFO 226
ADMINTABINFO 管理ビュー 232
ADMIN_CMD プロシージャー 37
サポートされているコマンド
ADD CONTACT 39
ADD CONTACTGROUP 41
AUTOCONFIGURE 42
BACKUP DATABASE 46
DESCRIBE 55
DROP CONTACT 66
DROP CONTACTGROUP 67
EXPORT 68
FORCE APPLICATION 80
GET STMM TUNING DBPARTITIONNUM 82
IMPORT 83
INITIALIZE TAPE 111
LOAD 112
PRUNE HISTORY/LOGFILE 156
QUIESCE DATABASE 158
QUIESCE TABLESPACES FOR TABLE 160
REDISTRIBUTE DATABASE PARTITION GROUP 163
REORG INDEXES/TABLE 174
RESET ALERT CONFIGURATION 185
RESET DATABASE CONFIGURATION 187
RESET DATABASE MANAGER CONFIGURATION 189
REWIND TAPE 190

ADMIN_CMD プロシージャー *(続き)*
サポートされているコマンド *(続き)*
RUNSTATS 191
SET TAPE POSITION 203
UNQUIESCE DATABASE 204
UPDATE ALERT CONFIGURATION 205
UPDATE CONTACT 211
UPDATE CONTACTGROUP 212
UPDATE DATABASE CONFIGURATION 213
UPDATE DATABASE MANAGER CONFIGURATION 217
UPDATE HEALTH NOTIFICATION CONTACT LIST 219
UPDATE HISTORY 220
UPDATE STMM TUNING DBPARTITIONNUM 222
メッセージの検索 225
メッセージの削除 231
ADMIN_COPY_SCHEMA プロシージャー 746
ADMIN_DROP_SCHEMA プロシージャー 751
ADMIN_GET_DBP_MEM_USAGE
表関数 223
ADMIN_GET_MSGS 表関数 225
ADMIN_GET_TAB_COMPRESS_INFO 表関数 226
ADMIN_GET_TAB_INFO 表関数 809
ADMIN_GET_TAB_INFO_V95 表関数 232
ADMIN_REMOVE_MSGS プロシージャー 231
ADMIN_TASK_ADD プロシージャー 239
ADMIN_TASK_LIST 管理ビュー 245
ADMIN_TASK_REMOVE プロシージャー 247
ADMIN_TASK_STATUS 管理ビュー 248
ADMIN_TASK_UPDATE プロシージャー 251
ALTOBJ プロシージャー 753
AM_BASE_RPTS 表関数 22
AM_BASE_RPT_RECOMS 表関数 21
AM_DROP_TASK プロシージャー 24
AM_GET_LOCK_CHNS プロシージャー 25
AM_GET_LOCK_CHN_TB プロシージャー 24
AM_GET_LOCK_RPT プロシージャー 26
AM_GET_RPT プロシージャー 34
AM_SAVE_TASK プロシージャー 35
ANALYZE_LOG_SPACE プロシージャー 663
APPLICATIONS 管理ビュー 351
APPLICATION_ID スカラー関数 756
APPL_PERFORMANCE 管理ビュー 350
AUDIT_ARCHIVE 254
AUDIT_DELIM_EXTRACT 255
AUDIT_LIST_LOGS 256
AUTHORIZATIONIDS 管理ビュー 347
AUTH_LIST_AUTHORITIES_FOR_AUTHID 表関数 340
AUTH_LIST_GROUPS_FOR_AUTHID 表関数 343
AUTH_LIST_ROLES_FOR_AUTHID 345
AUTOCONFIGURE コマンド
ADMIN_CMD の使用 42
AUTOMAINT_GET_POLICY ストアード・プロシージャー 257
AUTOMAINT_GET_POLICYFILE ストアード・プロシージャー 258
AUTOMAINT_SET_POLICY ストアード・プロシージャー 260
AUTOMAINT_SET_POLICYFILE ストアード・プロシージャー 261

B

BACKUP DATABASE コマンド
ADMIN_CMD の使用 46
BP_HITRATIO 管理ビュー 356
BP_READ_IO 管理ビュー 358
BP_WRITE_IO 管理ビュー 360

C

CAPTURE_STORAGEMGMT_INFO プロシージャー 673
COMPILATION_ENV 表関数 756
CONTACTGROUPS 管理ビュー 759
CONTACTS 管理ビュー 760
CONTAINER_UTILIZATION 管理ビュー 361
CREATE_STORAGEMGMT_TABLES プロシージャー 674

D

DB2 インフォメーション・センター
言語 930
更新 931
バージョン 930
別の言語で表示する 930
DB2 資料の印刷方法 929
DBCFG 管理ビュー 263
DBMCFG 管理ビュー 265
DBPATHS 管理ビュー 766
DB_HISTORY 管理ビュー
説明 761
DB_PARTITIONS 表関数 262
DESCRIBE コマンド
説明
ADMIN_CMD プロシージャーを使用 55
DROP CONTACT コマンド
ADMIN_CMD の使用 66
DROP CONTACTGROUP コマンド
ADMIN_CMD の使用 67
DROP_STORAGEMGMT_TABLES プロシージャー 675

E

ENV_FEATURE_INFO 管理ビュー
説明 272
ENV_INST_INFO 管理ビュー 269
ENV_PROD_INFO 管理ビュー 270
ENV_SYS_INFO 管理ビュー 273
ENV_SYS_RESOURCES 管理ビュー 274
EXPLAIN_FORMAT_STATS スカラー関数 770

EXPLAIN_GET_MSGS 表関数 774
EXPORT コマンド
説明
ADMIN_CMD プロシージャーを使用 68

F

FORCE APPLICATION コマンド
ADMIN_CMD の使用 80

G

GENERATE_DISTFILE プロシージャー 665
GET STMM TUNING DBPARTITIONNUM コマンド
ADMIN_CMD の使用 82
GET_DBM_CONFIG 表関数 819
GET_DBSIZE_INFO プロシージャー 777
GET_DB_CONFIG 表関数 818
GET_ROUTINE_OPTS スカラー関数 658
GET_ROUTINE_SAR プロシージャー 659
GET_SWRD_SETTINGS プロシージャー 666

H

HEALTH_CONT_HI 表関数 277
HEALTH_CONT_HI_HIS 表関数 279
HEALTH_CONT_INFO 表関数 281
HEALTH_DBM_HI 表関数 296
HEALTH_DBM_HI_HIS 表関数 298
HEALTH_DBM_INFO 表関数 300
HEALTH_DB_HI 表関数 283
HEALTH_DB_HIC 表関数 290
HEALTH_DB_HIC_HIS 表関数 292
HEALTH_DB_HI_HIS 表関数 286
HEALTH_DB_INFO 表関数 295
HEALTH_GET_ALERT_ACTION_CFG 表関数 301
HEALTH_GET_ALERT_CFG 表関数 304
HEALTH_GET_IND_DEFINITION 表関数 308
HEALTH_HI_REC プロシージャー 310
HEALTH_TBS_HI 表関数 312
HEALTH_TBS_HI_HIS 表関数 314
HEALTH_TBS_INFO 表関数 318

I

IMPORT コマンド
ADMIN_CMD の使用 83
INITIALIZE TAPE コマンド
ADMIN_CMD の使用 111

L

LOAD コマンド
ADMIN_CMD の使用 112
LOCKS_HELD 管理ビュー 363
LOCKWAIT 管理ビュー 366
LOG_UTILIZATION 管理ビュー 370
LONG_RUNNING_SQL 管理ビュー 371

M

MON_GET_FCM 表関数 801
MON_GET_FCM_CONNECTION_LIST 表関数 802
MQPUBLISH スカラー関数 320
MQREAD スカラー関数 322
MQREADALL 表関数 323
MQREADALLCLOB 表関数 325
MQREADCLOB スカラー関数 327
MQRECEIVE スカラー関数 328
MQRECEIVEALL 表関数 329
MQRECEIVEALLCLOB 表関数 332
MQRECEIVECLOB スカラー関数 334
MQSEND スカラー関数 335
MQSUBSCRIBE スカラー関数 337
MQUNSUBSCRIBE スカラー関数 338

N

NOTIFICATIONLIST 管理ビュー 779

O

OBJECTOWNERS 管理ビュー 348

P

PDLOGMSGS_LAST24HOURS 管理ビュー
説明 787
PD_GET_DIAG_HIST 表関数
説明 780
PD_GET_LOG_MSGS 表関数
説明 787
PRIVILEGES 管理ビュー 349
PRUNE HISTORY/LOGFILE コマンド
ADMIN_CMD プロシージャーを使用 156
PUT_ROUTINE_SAR プロシージャー 660

Q

QUERY_PREP_COST 管理ビュー 374
QUIESCE DATABASE コマンド
ADMIN_CMD の使用 158
QUIESCE TABLESPACES FOR TABLE コマンド
ADMIN_CMD の使用 160

R

REBIND_ROUTINE_PACKAGE プロシージャー 661

REDISTRIBUTE DATABASE PARTITION GROUP コマンド
ADMIN_CMD の使用 163
REG_VARIABLES 管理ビュー 267
REORG INDEXES/TABLE コマンド
ADMIN_CMD の使用 174
REORGCHK_IX_STATS プロシージャー 794
REORGCHK_TB_STATS プロシージャー 796
RESET ALERT CONFIGURATION コマンド
ADMIN_CMD の使用 185
RESET DATABASE CONFIGURATION コマンド
ADMIN_CMD の使用 187
RESET DATABASE MANAGER CONFIGURATION コマンド
ADMIN_CMD の使用 189
REWIND TAPE コマンド
ADMIN_CMD の使用 190
RUNSTATS コマンド
ADMIN_CMD の使用 191

S

SET TAPE POSITION コマンド
ADMIN_CMD の使用 203
SET_ROUTINE_OPTS プロシージャー 662
SET_SWRD_SETTINGS プロシージャー 669
SNAPAGENT 管理ビュー 375, 515
SNAPAGENT_MEMORY_POOL 管理ビュー 378, 518
SNAPAPPL 管理ビュー 390, 530
SNAPAPPL_INFO 管理ビュー 382, 522
SNAPBP 管理ビュー 399, 539
SNAPBP_PART 管理ビュー 404, 544
SNAPCONTAINER 管理ビュー 407, 547
SNAPDB 管理ビュー 411, 551
SNAPDBM 管理ビュー 427, 567
SNAPDBM_MEMORY_POOL 管理ビュー 431, 571
SNAPDB_MEMORY_POOL 管理ビュー 423, 563
SNAPDETAILLOG 管理ビュー 434, 574
SNAPDYN_SQL 管理ビュー 437, 577
SNAPFCM 管理ビュー 442, 582
SNAPFCM_PART 管理ビュー 445, 585
SNAPHADR 管理ビュー 447, 587
SNAPLOCK 管理ビュー 451, 591
SNAPLOCKWAIT 管理ビュー 457, 597
SNAPSTMT 管理ビュー 464, 604
SNAPSTORAGE_PATHS 管理ビュー 471, 611
SNAPSUBSECTION 管理ビュー 473, 613
SNAPSWITCHES 管理ビュー 477, 617
SNAPTAB 管理ビュー 480, 620
SNAPTAB_REORG 管理ビュー 483, 623
SNAPTBSP 管理ビュー 488, 628
SNAPTBSPPART 管理ビュー 494, 634
SNAPTBSP_QUIESCER 管理ビュー 499, 639
SNAPTBSP_RANGE 管理ビュー 503, 643
SNAPUTIL 管理ビュー 507, 647
SNAPUTIL_PROGRESS 管理ビュー 510, 650
SNAP_GET_AGENT 表関数 375, 515
SNAP_GET_AGENT_MEMORY_POOL 表関数 378, 518
SNAP_GET_APPL_INFO 表関数 828
SNAP_GET_APPL_INFO_V95 表関数 382, 522
SNAP_GET_APPL_V95 表関数 390, 530
SNAP_GET_BP_PART 表関数 404, 544
SNAP_GET_BP_V95 表関数 399, 539
SNAP_GET_CONTAINER_V91 表関数 407, 547
SNAP_GET_DBM_MEMORY_POOL 表関数 431, 571
SNAP_GET_DBM_V95 表関数 427, 567
SNAP_GET_DB_MEMORY_POOL 表関数 423, 563
SNAP_GET_DB_V95 表関数 411, 551
SNAP_GET_DETAIL_LOG_V91 表関数 434, 574
SNAP_GET_DYN_SQL_V91 表関数 863
SNAP_GET_DYN_SQL_V95 表関数 437, 577
SNAP_GET_FCM 表関数 442, 582
SNAP_GET_FCM_PART 表関数 445, 585
SNAP_GET_HADR 表関数 447, 587
SNAP_GET_LOCK 表関数 451, 591
SNAP_GET_LOCKWAIT 表関数 457, 597
SNAP_GET_STMT 表関数 464, 604
SNAP_GET_STORAGE_PATHS 表関数 471, 611
SNAP_GET_SUBSECTION 表関数 473, 613
SNAP_GET_SWITCHES 表関数 477, 617
SNAP_GET_TAB_REORG 表関数 483, 623
SNAP_GET_TAB_V91 表関数 480, 620
SNAP_GET_TBSP_PART_V91 表関数 494, 634
SNAP_GET_TBSP_QUIESCER 表関数 499, 639
SNAP_GET_TBSP_RANGE 表関数 503, 643
SNAP_GET_TBSP_V91 表関数 488, 628
SNAP_GET_UTIL 表関数 507, 647
SNAP_GET_UTIL_PROGRESS 表関数 510, 650
SNAP_WRITE_FILE プロシージャー 513, 653
SQL 管理ルーチン 686
使用すべきでないルーチン 805
SQL ステートメント
ヘルプを表示する 930
SQLERRM スカラー関数 797
STEPWISE_REDISTRIBUTE_DBPG プロシージャー 671
SYSINSTALLOBJECTS プロシージャー 800
SYSTS_ADMIN_CMD プロシージャー 676
SYSTS_ALTER ストアード・プロシージャー 677
SYSTS_CLEAR_COMMANDLOCKS ストアード・プロシージャー 681
SYSTS_CLEAR_EVENTS ストアード・プロシージャー 683
SYSTS_CREATE 686
SYSTS_CREATE プロシージャー 686
SYSTS_DISABLE 693
SYSTS_DISABLE プロシージャー 693
SYSTS_DROP ストアード・プロシージャー 695
SYSTS_ENABLE ストアード・プロシージャー 698
SYSTS_UPDATE ストアード・プロシージャー 699

T

TBSP_UTILIZATION 管理ビュー 655
TOP_DYNAMIC_SQL 管理ビュー 657

U

UNQUIESCE DATABASE コマンド
ADMIN_CMD の使用 204
UPDATE ALERT CONFIGURATION コマンド
ADMIN_CMD の使用 205
UPDATE CONTACT コマンド
ADMIN_CMD の使用 211
UPDATE CONTACTGROUP コマンド
ADMIN_CMD の使用 212
UPDATE DATABASE CONFIGURATION コマンド
ADMIN_CMD の使用 213
UPDATE DATABASE MANAGER CONFIGURATION コマンド
ADMIN_CMD の使用 217
UPDATE HEALTH NOTIFICATION CONTACT LIST コマンド
ADMIN_CMD の使用 219
UPDATE HISTORY コマンド
ADMIN_CMD の使用 220
UPDATE STMM TUNING DBPARTITIONNUM コマンド
ADMIN_CMD の使用 222

V

Visual Explain
チュートリアル 933

W

WLM_CANCEL_ACTIVITY プロシージャー 702
WLM_CAPTURE_ACTIVITY_IN_PROGRESS プロシージャー 703
WLM_COLLECT_STATS プロシージャー
説明 705
WLM_GET_ACTIVITY_DETAILS 表関数
説明 707
WLM_GET_QUEUE_STATS 表関数
説明 714
WLM_GET_SERVICE_CLASS_AGENTS 表関数
説明 718
WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES 表関数
説明 724
WLM_GET_SERVICE_SUBCLASS_STATS 表関数
説明 728
WLM_GET_SERVICE_SUPERCLASS_STATS 表関数
説明 734
WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES 表関数
説明 738
WLM_GET_WORKLOAD_STATS 表関数
説明 741
WLM_GET_WORK_ACTION_SET_STATS 表関数
説明 736
WLM_SET_CLIENT_INFO プロシージャー 744

Printed in Japan

SC88-4435-03

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:

DB2 Version 9.5 for Linux, UNIX, and Windows　バージョン 9 リリース 5

管理ルーチンおよびビュー