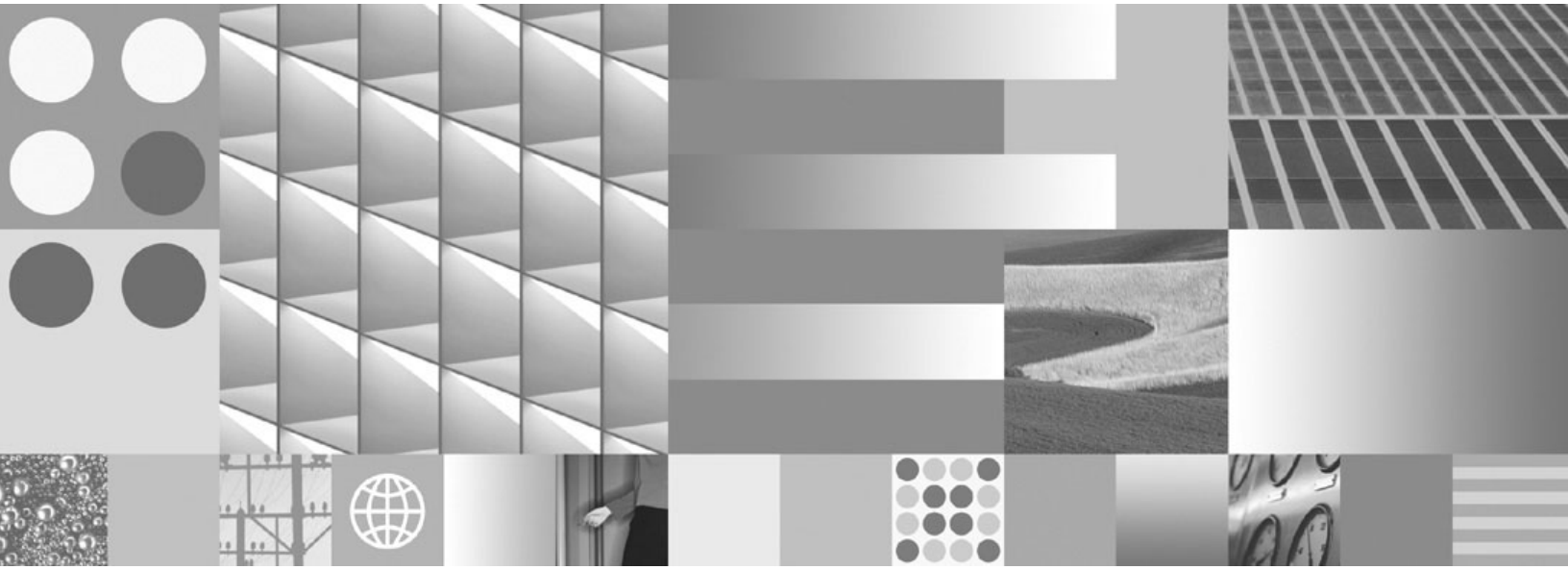
IBM Information Integration

バージョン 9.5

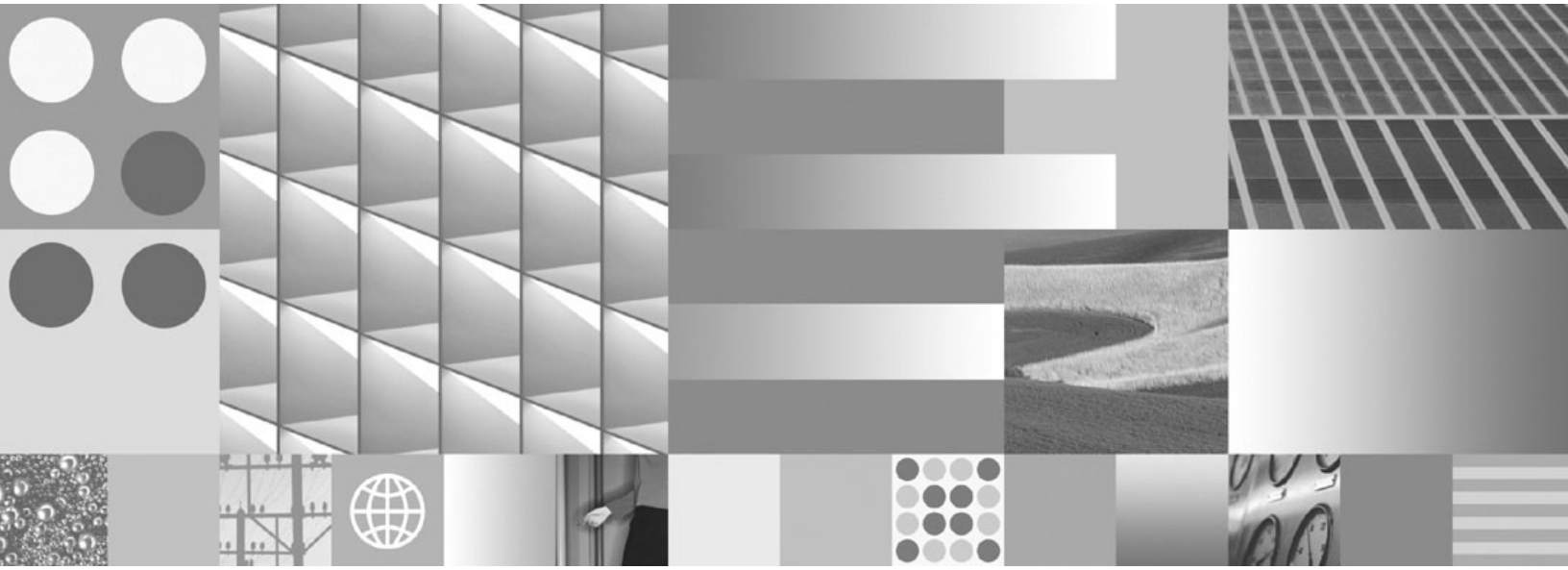


SQL レプリケーション ガイドおよびリファレンス

SC88-4168-01
(英文原典：SC19-1030-01)

IBM Information Integration

バージョン 9.5



SQL レプリケーション ガイドおよびリファレンス

SC88-4168-01
(英文原典：SC19-1030-01)

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、 531 ページの『特記事項』に記載されている情報をお読みください。

IBM 発行のマニュアルに関する情報のページ

http://www.ibm.com/jp/manuals/

こちらから、日本語版および英語版のオンライン・ライブラリーをご利用いただけます。また、マニュアルに関するご意見やご感想を、上記ページよりお送りください。今後の参考にさせていただきます。

(URL は、変更になる場合があります)

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： SC19-1030-01
IBM Information Integration Version 9.5
SQL Replication Guide and Reference

発　行： 日本アイ・ビー・エム株式会社

担　当： ナショナル・ランゲージ・サポート

第1刷 2007.10

この文書では、平成明朝体™W3、平成明朝体™W7、平成明朝体™W9、平成角ゴシック体™W3、平成角ゴシック体™W5、および平成角ゴシック体™W7を使用しています。この(書体*)は、(財) 日本規格協会と使用契約を締結し使用しているものです。フォントとして無断複製することは禁止されています。

注* 平成明朝体™W3、平成明朝体™W7、平成明朝体™W9、平成角ゴシック体™W3、
平成角ゴシック体™W5、平成角ゴシック体™W7

目次

第 1 章 SQL レプリケーションの計画

SQL レプリケーションをプランするときには、マイグレーション、メモリー、ストレージ、競合、ソース・システム、コード・ページ変換、およびパフォーマンスについてプランすることも考慮することが必要な場合があります。

移行計画

移行の計画には、レプリケーションのバージョン間で移行する際に生じる可能性がある問題に関する計画が含まれます。

既存のレプリケーション環境から移行している場合、特定の移行に関連する問題を考慮する必要があります。「*WebSphere Information Integration Migrating to Replication Version 9*」には、バージョン 9 レプリケーションに移行する方法が説明されています。バージョン 9 に移行するには、まずご使用のサーバーをバージョン 8 にしなければなりません。「*WebSphere Information Integration Migrating to SQL Replication Version 8*」には、バージョン 8 レプリケーションに移行する方法が説明されています。また、現在 DB2® DataJoiner® を使ってデータを複製しているレプリケーション環境を、非 DB2 リレーショナル・サーバーに、またはその逆に移行する方法についても説明されています。これらの資料は、ご使用の製品に関する WebSphere® Information Integration サポート・サイトからオンラインで入手できます。

メモリーの計画

メモリーの計画には、レプリケーションに必要なメモリーの量を計画することが含まれます。レプリケーションでは、必要なメモリーだけしか使用されません。必要なメモリー量は、ソースから複製されているデータの量とトランザクションの並行性に直接比例します。基本的に、複製されているデータの量が多ければ多いほど、また、同時実行されているトランザクションの数が多ければ多いほど、より多くのメモリーがレプリケーションに必要となります。

キャプチャー・プログラムとアプライ・プログラムを実行すると、かなり大量のメモリーが消費される可能性があります。

キャプチャー・プログラムによって使用されるメモリー

キャプチャー・プログラムは、DB2 リカバリー・ログを読み取る際にメモリーを使用します。関連したコミット・レコードまたはアボート・レコードを読み取るまで、キャプチャー・プログラムは個々のトランザクション・レコードをメモリーに保管します。アボートされたトランザクションに関連したデータはメモリーから消去され、コミット・レコードに関連したデータは CD 表および UOW 表に書き込まれます。コミットされたトランザクションは、キャプチャー・プログラムがコミット・インターバルに達して作業をコミットするまでメモリー内に留まります。

キャプチャー・プログラムが使用しているメモリー量をモニターするには、IBMSNAP_CAPMON 表の CURRENT_MEMORY 列の値を調べてください。

キャプチャー・プログラムの始動時に **memory_limit** パラメーターを設定すると、トランザクションと関連付けられているストレージとして、キャプチャー・プログラムが確実に指定された量のメモリーを使用するようにできます。このパラメーターによってその他のストレージ使用が限定されることはありません。また、**memory_limit** パラメーターは、キャプチャー・プログラムの実行中でも変更できます。メモリー限度に達すると、キャプチャーは一部のトランザクションを予備ファイルに書き込みます。キャプチャー・プログラムによって使用されるメモリー・リソースについては、このプログラムのストレージ・スペース要件と関連付けて検討する必要があります。

また、キャプチャー・プログラムのメモリー要件について計画する際には、ユーザー・トランザクションのサイズとコミット・インターバルも検討する必要があります。キャプチャー・プログラムの実行時にバッチ・ジョブを一時コミットなしで長時間実行すると、大量のメモリーが消費されます。一般に、コミット・インターバルの値が小さければ小さいほど、キャプチャー・プログラムが必要とするメモリーは少なくなります。

アクティブ登録に関する情報は、キャプチャー・プログラムを開始するとき、およびキャプチャー・プログラムの実行中に登録が動的に追加されるときに、読み取られてメモリーに保管されます。

z/OS Linux UNIX Windows

レプリケーションは、ログ・レコードを読み取る際にメモリー・バッファーを使用します。z/OS® オペレーティング・システムの場合は、デフォルトのサイズは 1 KB ページ x 66 で、これは ECSA (拡張共通サービス域) ストレージです。レプリケーションが ECSA を使用するのはこの場合だけです。Linux®、UNIX® および Windows® オペレーティング・システムの場合は、バッファーのデフォルトのサイズは 4 KB (K バイト) ページ x 50 です。

System i

CURRENT_MEMORY は、アクティブ CD 表のために標準入出力バッファーによって使用されるメモリーに入らないトランザクション・レコードを保留するために割り振られた追加のメモリーの最新のアカウントです。これは、多数のトランザクションを保留するために使用されている追加のメモリーの量を示す指標です。これは、特定のジャーナル・ジョブによって使用されているメモリーすべての正確な合計ではありません。

IBMSNAP_CAPMON 表に保管されている情報により、メモリーの使用を調整するのに役立つ稼働統計が提供されます。この表内の値は特定のキャプチャー・モニター・インターバルに関するものであり、各モニター・インターバルにわたった累積的な値ではないことに注意してください。 CURRENT_MEMORY 列のデータには、加算されるカウントは含まれません。レコードの作成時に、この値にはモニター・インターバルが終了した時点で使用中であったメモリーが反映されます。キャプチャー・モニター・インターバルは、キャプチャー・プログラムがこの表にデータを挿入する頻度を決定します。以下の方法のいずれかを使用して、キャプチャー・プログラムによって使用されているメモリーの量を調整してください。

予備の分も考慮したメモリー限度のチューニング:

1. キャプチャー・プログラムの始動時に、デフォルトのメモリー限度を使用する。

2. IBMSNAP_CAPMON 表の TRANS_SPILLED 列を調べて、データがメモリーから一時ファイルにあふれ出たかどうかをチェックする。この列は、メモリー制限のために特定のキャプチャー・モニター・インターバルの間にディスクにあふれたソース・システム・トランザクションの数を示しています。
3. データがメモリーからあふれた場合は、引き上げたメモリー限度または値を低くしたコミット・インターバルのいずれかを使用する。

あふれるのを防ぐためのメモリー限度のチューニング:

1. キャプチャー・プログラムの始動時に、メモリー限度を高く設定する (どのくらい高くするのかは、システム・リソースに応じて決まります)。
2. IBMSNAP_CAPMON 表の CURRENT_MEMORY 列を調べて、使用中のメモリー量をチェックする。この列は、キャプチャー・プログラムが特定のキャプチャー・モニター・インターバルの間に使用したメモリーの量を (バイト単位で) 示しています。
3. メモリー限度として指定した量よりもかなり少ないメモリーしか使用されていない場合は、メモリー限度としてもう少し下げた値を設定する。

アプライ・プログラムによって使用されるメモリー

アプライ・プログラムは、データを取り出すときにメモリーを使用します。使用されるメモリー量は、表の列のサイズおよび一度に取り出される行の数に比例します。例えば、アプライ・プログラムが LOB 列を取り出す場合は、このプログラムは 2 G バイトのメモリーを潜在的に使用する可能性があります。

アクティブ・サブスクリプション・セットに関する情報は、アプライ・プログラムの実行中に読み取られてメモリーに保管されます。通常、アプライ・プログラムによって一度に使用されるメモリー量は、ほとんどのメンバーが含まれているサブスクリプション・セットを処理するのに必要なメモリー量に比例します。

ストレージの計画

ストレージの計画は、DB2 ソース・サーバーのログの影響、ターゲット・サーバーのログの影響、ターゲット表とコントロール表のストレージ要件、診断ログ・ファイルのスペース所要量 (Linux、UNIX、Windows、z/OS)、キャプチャー・プログラムの予備ファイルのスペース所要量、およびアプライ・プログラムの予備ファイルのスペース所要量にとって重要です。

DB2 が必要とするストレージに加えて、以下のトピックのレプリケーションのためにストレージが確実に使用可能であるようにしておく必要があります。これらのトピックで指定されているサイズはすべて目安にすぎません。実動可能システムを準備して設計するには、障害防止などの要因も考慮に入れなければなりません。例えば、ネットワークが停止する可能性を考慮に入れて、データを保留する期間を延長しなければならない場合があります。

ヒント: ストレージ見積もりが過度に高いと思われる場合は、アプライ・プログラムがサブスクリプション・セットを実行する頻度とレプリケーション表の整理の頻度を再度調べてください。ストレージの使用量、最大限の障害許容度、および CPU オーバーヘッドの間のトレードオフについて検討する必要があります。

DB2 ソース・サーバーの場合のログの影響

一般に、レプリケーションに関係するすべての表のために、現行ログの 3 倍の量のログが追加で必要となります。基本的に、ソース表用、CD 表用、およびレプリケーション・コントロール表用のログ・スペースが必要です。このセクションでは、使用しているレプリケーション環境で予測可能なログの影響をより正確に見積もるのに役立つ可能性のあるその他の要因を示します。

アプリケーションによってソース・データベースに加えられる更新と、レプリケーション要件を検討してください。例えば、更新アプリケーションが一般に表の列の 60 パーセントを更新する場合、レプリケーション要件によっては、ログ・レコードの大きさが、類似の表が複製されない場合に比べて半分以上も増加する可能性があります。

z/OS Linux UNIX Windows

- DB2 は UPDATE ステートメントごとに完全な行イメージをログに記録します。これが発生するのは、 DATA CAPTURE CHANGES キーワードを使用して表を作成 (または変更) してからでなければ表をレプリケーションできないからです。
- ログへの追加量が最大になるレプリケーション要件の 1 つとして、変更前イメージと変更後イメージのキャプチャー (Update-anywhere レプリケーション・シナリオにおけるレプリケーション・ターゲット表の例が当てはまる) が挙げられます。ログの量を少なくする方法の 1 つとして、レプリケーション・ソースに対して定義されている列数を少なくすることができます。例えば、必要でない場合は変更前イメージをキャプチャーしないでください。

System i

- DB2 は UPDATE ステートメントごとに完全な行イメージをログに記録します。ログの量を少なくする方法の 1 つは、レプリケーション・ソースに対して定義されている列数を少なくすることであり、例えば、必要でない場合は変更前イメージをキャプチャーしないことです。
- CD 表用および UOW 表用として使用するストレージの量を最小化するには、これらの表を頻繁に再編成します。なぜなら、整理ではユーザーに代わって DASD を回復することはないからです。 ENDDPRCAP コマンドでキーワード RGZCTLTBL (コントロール表の再編成) を使用すれば、コントロール表を再編成できます。通常の操作条件で DASD の使用パターンを監視して、DASD 使用の予測と管理に役立ててください。ジャーナリングがオンになっている場合は、UOW 表および CD 表に対する DB2 ログの挿入および削除が行われるのに従ってログまたはジャーナルの量が増えることも考慮に入れてください。
- 現行レシーバーがいっぱいになると、システムは新しいレシーバーに切り替えられます。複製する必要がなくなった古いレシーバーについては、保管するか削除するかをオプションで指定できます。たくさんのトランザクションを扱うシステムの場合、キャプチャー・プログラムは場合によって遅れることがあります。キャプチャーが頻繁に遅れる場合は、ソース表を複数のジャーナルに分離して、ワークロードをキャプチャー・プログラムの複数インスタンスに分散できます。

ターゲット・サーバーの場合のログの影響

ソース・データベースのロギングに加えて、行が適用されるターゲット・データベースでもロギングされます。ログへの影響は、アプライ・プログラム用に選択したコミット・モードによって異なります。

表モード

表モード処理では、アプライ・プログラムはフェッチしたデータがすべて適用された後で単一コミットを発行します。アプライ・プログラムは一時チェックポイントを発行しません。このケースでは、アプライ・プログラムが 1 回のインターバルで処理する最大データ量を見積もり、そのデータ量を格納できるようにログ・スペースを調整する必要があります。

トランザクション・モード

トランザクション・モード処理では、アプライ・プログラムはそれぞれの更新をソース・トランザクションの順序どおりにすべてターゲット表にコピーし、それらの変更を 1 回のインターバルでトランザクション境界に達したときにまとめてコミットします。一時コミットのインターバルを設定するには、サブスクリプション・セット・オプション (**commit_count**(*x*)) の値 *x* を設定します。すべての応答セットを取り出すと、アプライ・プログラムは予備ファイルの内容をコミット・シーケンスの順序に従って適用します。このタイプの処理では、すべての予備ファイルを同時に開いて処理できます。例えば、コミット・カウントを 1 に設定と、アプライ・プログラムは 1 つのトランザクションが終わるたびにコミットします。コミット・カウントを 2 に設定した場合は、アプライ・プログラムは 2 つのトランザクションのセットが 1 つ終わるたびにコミットします。

System i ターゲット表のログ・スペース (ジャーナル・レシーバー・スペース) も検討する必要があります。System i™ では、ターゲット表用のジャーナル・レシーバーを **MNGRCV(*SYSTEM)** および **DLTRCV(*YES)** パラメーターで作成でき、さらにジャーナリングする必要があるのは変更後イメージ列だけであるため、以下の公式を使用して、ターゲット表用ジャーナル・レシーバーの大きさを見積もってください。

```
journal_receiver_volume=target_table_row_length X journal_receiver_threshold
```

ターゲット表およびコントロール表のストレージ要件

新しいターゲット表の大きさを見積もる必要があります。ターゲット表に必要なスペースはソース表のスペース以下であるのが普通ですが、ターゲット表が非正規化されたり、その中に変更前イメージ (変更後イメージに加えて) または履歴データが含まれていると、ソース表のスペースよりもずっと大きくなる場合があります。ターゲット表のサイズは、複製する際に何を選択するかによって異なります。選択するものとしては、例えば、複製するソース表のパーセント、複製する列のデータ・タイプ、変更前および変更後のイメージを複製するかどうか、算出列を追加するかどうか、行をサブセット化するかどうか、レプリケーション時になんらかのトランスフォーメーションを行うかどうかなどが挙げられます。

CD 表および一部のレプリケーション・コントロール表 (IBMSNAP_UOW、IBMSNAP_CAPTRACE、IBMSNAP_APPLYTRACE、IBMSNAP_APPLYTRAIL、IBMSNAP_CAPMON、IBMSNAP_ALERTS) も、DB2 ソース・データベース用として必要なディスク・スペースに影響を与えます。レプリ

ケーション環境のセットアップの仕方によっては、これらの表が非常に大きくなる可能性があります。上記以外のレプリケーション・コントロール表の場合は、必要なスペースは通常少なく、かつ静的です。

CD 表のサイズは、ソース表に変更が加えられるたびに大きくなり、やがてキャプチャー・プログラムによって整理されます。 CD 表に必要なスペースを見積もるには、データをどのくらい保持してから整理するのかについて、その期間を最初に決定した後、キャプチャー・プログラムがこれらの表を自動的に整理する頻度、またはユーザー自身がコマンドを使用してこれらの表を整理する頻度を指定します。

レプリケーション済みデータのバイト数を計算する場合は、キャプチャー・プログラムが CD 表に追加したオーバーヘッド・データの分として 21 バイトをそれぞれの行ごとに組み込む必要があります。キャプチャー・プログラムが CD 表にデータをキャプチャーし続けられる期間 (例えばネットワークが停止するなどしてデータが適用不能なときも含む) を決定してください。上記の偶発的な障害が発生した期間内にソース表で通常キャプチャーされる可能性のある挿入、更新、および削除の数を見積もってください。

CD 表の推奨サイズを判別するために、次のガイドラインを使用してください。

```
recommended_CD_size =
  ( (21 bytes) + sum(length of all registered columns) ) X
  (number of inserts, updates, and deletes to source table
   during the contingency period)
```

例

CD 表の行の長さが 100 バイト (これにオーバーヘッドの分の 21 バイトを加える) で、24 時間にわたる偶発的障害の期間内に 100,000 個の更新がキャプチャーされるとすると、CD 表に必要なストレージは約 12 MB となります。

この公式では、登録済み列には変更前イメージと変更後イメージの両方が含まれるものとしています。更新が INSERT 操作と DELETE 操作の対に変換されるようになっている場合は、挿入、更新、および削除の合計数を判別するときにそれらを勘定に入れてください。例えば、ソース表に対する 1 つの更新は、それぞれ CD 表の 2 つの行としてカウントしてください。

UOW 表は、特定のコミット・インターバルの間にキャプチャー・プログラムによって挿入される行数、および整理される行数に基づいて、拡大および縮小します。アプリケーション・トランザクションが COMMIT を発行するたび、およびトランザクションが登録済みのレプリケーション・ソース表に対して INSERT、DELETE、または UPDATE 操作を実行するたびに、行が 1 つ UOW 表に挿入されます。当初はこの表に必要なスペースを多めに見積もっておき、実際に使用されるスペースをモニターして、リカバリーできるスペースがあるかどうかを判別してください。

キャプチャー・プログラム用予備ファイルのスペース所要量

十分なメモリーがない場合、キャプチャー・プログラムはトランザクションを予備ファイルに書き込みます (つまり、メモリーに入りきらないトランザクションをここに入れます)。キャプチャー・プログラムは、最大のトランザクションをファイルに書き込みます。しかし、最大のトランザクションがメモリー限度を超える原因となったトランザクションであるとは限りません。

z/OS

予備ファイルは仮想入出力 (VIO) に送られます。

Linux UNIX Windows

予備ファイルは常にディスク上にあります。1 トランザクション当たり 1 つのファイルが **capture_path** ディレクトリーに作成されます。

System i

予備ファイルはライブラリー QTEMP の中に作成されます。予備ファイルが必要な各登録ごとに 1 つの予備ファイルが作成されます。

キャプチャー予備ファイルのサイズは、以下の要因によって決まります。

メモリー限度

memory_limit 稼働パラメーターを使用して、キャプチャー・プログラムが使用できるメモリー量を指定します。許可するメモリーが多ければ多いほど、キャプチャー・プログラムが入りきらない分をファイルに入れる可能性が少なくなります。

トランザクションのサイズ

トランザクションのサイズが大きいと、入りきらないためにファイルに入れる必要性が高まります。

同時実行されるトランザクションの数

キャプチャー・プログラムが同時に処理するトランザクションの数が増えた場合、またはキャプチャー・プログラムがインターリーブド・トランザクションを処理する場合は、キャプチャー・プログラムはより多くの情報をメモリーまたはディスクに保管する必要があります。

コミット・インターバル

通常、コミット・インターバルの値が小さければ小さいほど、必要とされるストレージは少なくなります。なぜなら、キャプチャーが情報をコミットする前にその情報をメモリーに保管しておく期間が短くなるからです。

アプライ・プログラム用予備ファイルのスペース所要量

アプライ・プログラムには、データを保管するための一時的なスペースが必要です。ASNLOAD ユーティリティーを使用する場合は、ロード予備ファイルではなくロード入力ファイルになります。アプライ・プログラムは、更新をターゲット表に適用するまでの間それらの更新を保留するために予備ファイルを使用します。一般に、予備ファイルはディスク・ファイルです。ただし、z/OS オペレーティング・システムでは、入りきらないデータをメモリーに入れるように指定できます。仮想メモリーの制約がない限り、予備ファイルはディスク上ではなく仮想メモリーに保管してください。

予備ファイルのサイズは、各レプリケーション・インターバル内で複製するよう選択されたデータのサイズに比例します。予備ファイルのサイズは、通常はこのデータのサイズのおよそ 2 倍です。予備ファイルのサイズを見積もるには、アプライ・プログラム用に計画した頻度インターバル (またはデータ・ブロック値) と、同じ期間内 (または変更のピーク時) になされた変更の量を比較します。

z/OS Linux UNIX Windows 予備ファイルの行サイズはターゲット行 のサイズであり、これにはあらゆるレプリケーション・オーバーヘッド列が含まれます。この行サイズは、DB2 パック内部フォーマットでのサイズではなく、拡張された解釈済みの文字フォーマット (SELECT から取り出された状態) でのサイズです。行には、行の長さと、個々の列ストリング上の NULL 終止符も含まれます。以下の例では、複製するよう選択されたデータに必要な予備ファイルのサイズを見積もります。この例では、その予備ファイルに保管される他のデータに必要な追加のスペースは勘定に入っていません。

System i 予備ファイルの行サイズは定数の 32 KB です。

例

変更量のピークが時間あたり 12,000 件の場合に、アプライ・プログラムの頻度が 1 時間インターバルで計画されていると、予備ファイルには 1 時間分の更新件数、つまり 12,000 件を保留しなければなりません。1 件の更新が 100 バイトのデータを表すとすれば、予備ファイルは最小限でも約 1.2 MB になります。これに加え、この予備ファイルに保管される他のデータ用のスペースも必要です。

診断ログ・ファイルのスペース所要量 (z/OS、Linux、UNIX、Windows)

診断ログ・ファイルには、レプリケーション・プログラムのアクティビティーに関する情報 (プログラムの始動時刻と停止時刻、およびプログラムが発行したその他の情報メッセージまたはエラー・メッセージなど) が保管されます。デフォルトでは、プログラムは再始動後もログ・ファイルにメッセージを付加します。これらのログ・ファイルを含むディレクトリーには、ファイルを保管するのに十分なスペースを確実に準備してください。

診断ログ・ファイルのロケーションは、キャプチャー・プログラム、アプライ・プログラム、およびレプリケーション・アラート・モニター・プログラムの始動時に始動パラメーター **capture_path**、**apply_path**、および **monitor_path** に設定された値によって決まります。

ストレージに関して不安がある場合、プログラム・ログを再利用するオプションがあります。これを使用すると、プログラムは始動するたびにログを削除して再作成します。プログラムの始動時に、ログを再利用するかどうかを指定できます。

競合検出の計画

標準または拡張競合検出を使用している場合は、レプリカ・ターゲット表用の CD (または CCD) 表に変更前イメージを保管する必要があります。また、参照整合性規則に制限があります。ピアツーピア・シナリオおよび Update-anywhere シナリオの場合、またはアプライ・プログラムがトランザクション・モード処理を使用する場合は、ソース規則に合わせた参照整合性規則を定義する必要があります。ピアツーピア・シナリオまたは Update-anywhere シナリオを使用し、かつ競合検出をオンにしない場合は、更新競合が防止されるようなアプリケーション環境を設計する必要があります。使用するアプリケーション環境では競合が起こり得ないのであれば、競合検出を使用しないことで処理サイクルを節約できます。

以下の方法のいずれかを使用して、ピアツーピア・レプリケーションおよび Update-anywhere レプリケーションでの競合を防止してください。

キー別のフラグメント化

レプリケーション・ソースが、特定のサイトのキー範囲別にレプリカによって更新されるようにアプリケーションを設計します。例えば、ニューヨークにあるサイトでは、米国東部のセールス・レコード (キー範囲として 49999 以下の ZIP コード[1]を使用) だけを更新できます。ただし、すべてのセールス・レコードの読み取りは行うことができます。

時間別のフラグメント化

特定のサイトで特定の期間においてだけ表を更新できるようにアプリケーションを設計します。それらの期間は、互いに十分間隔が空いていなければなりません。これは、これからマスター・バージョンになるサイトに対して加えられた、すべてのペンディング中の変更をレプリケーションできるようにするためです。夏時間 (またはサマータイム) などの時間の変更や、時間帯の相違にも対応します。

非 DB2 リレーショナル・ソースの計画

非 DB2 リレーショナル・データベースからのレプリケーションの場合は、キャプチャー・プログラムの代わりにキャプチャー・トリガーが使用されます。これらのトリガーは、変更済みのデータを非 DB2 リレーショナル・ソース表からキャプチャーし、その変更済みのデータを CCD 表にコミットします。

キャプチャー・トリガーは、トランザクションのスループット率とログ・スペース所要量に影響します。また、使用している環境に既存のトリガーがある場合は、それらを新しいキャプチャー・トリガーとマージしなければならない場合があります。詳細については、以下のセクションを参照してください。

キャプチャー・トリガーの場合のトランザクション・スループット率

トリガー・ベースの変更キャプチャーはトランザクション・スループット率に影響を与えるため、ソース・システムのトランザクション・ワークロードが増加します。

キャプチャー・トリガーは更新トランザクションの場合の応答時間を増加させます。この影響が最大となるのは、複製されるアプリケーション・ソース表をきわめて頻繁に更新するトランザクションの場合です。

非 DB2 リレーショナル・ソース・サーバーの場合のログの影響

非 DB2 リレーショナル・ソース・サーバーの場合、ソース・アプリケーションはより多くのアクティブ・ログ・スペースを必要とします。なぜなら、ログの量が複製されるソース表のおよそ 3 倍になるからです。変更は、トリガーによってソース表からキャプチャーされて、CCD 表に保管されます。変更済みのデータは変更中のソース表と同じコミット有効範囲内で書き込まれ、データは後でトリガー・ベースの整理機構によって削除されます。

1. 米国の郵便番号

1 回のソース INSERT、UPDATE、または DELETE 操作は、それぞれ 1 回の INSERT、UPDATE、または DELETE 操作に INSERT 操作と DELETE 操作をさらに 1 回ずつ加えたものとなります。更新を DELETE 操作と INSERT 操作の対に変更する場合は、ログの量はさらに増えます。

ログ・スペースを使い尽くしてしまい、キャプチャー・トリガーがレコードを CCD 表に挿入できなくなると、ユーザーまたはアプリケーション・プログラムによって試行されたトランザクションは正常に完了しません。

既存のトリガーとキャプチャー・トリガーとの共存

キャプチャー・トリガー・ロジックは、ソースの登録時にレプリケーション・センターによって生成される SQL スクリプトに含まれています。

デフォルトでは、INSERT トリガー、UPDATE トリガー、および DELETE トリガーは、これらのタイプの変更 (挿入、更新、および削除) をソース表からレプリケーションできるように作成されます。トリガー名は、CCD 表の名前と、トリガーのタイプを記述する先行文字 (I は INSERT、U は UPDATE、D は DELETE) から構成されます。例えば、CCD 表の名前が `undjr02.ccd001` である場合は、生成される DELETE トリガーの名前は `undjr02.dccd001` です。スクリプト内に生成されたトリガー名を変更してはなりません。

レプリケーション用に登録する表にトリガーがすでに存在しており、かつそのトリガーの名前が生成されたスクリプト内の名前と同じである場合は、そのスクリプトの生成時に警告を受け取ります。その生成されたスクリプトを実行しないでください。なぜなら、既存のトリガーが RDBMS によって上書きされる可能性があるからです。既存のトリガーと新しいトリガーとをどのようにマージするのかを決定し、既存のロジックとレプリケーション・センターによって生成されたトリガー・ロジックとをマージするスクリプトを作成してください。

作成するトリガー・タイプがレプリケーション用に登録する表にすでに存在しており、かつ RDBMS がそのタイプのトリガーを 1 つの表当たり 1 つだけしか許可しない場合は、生成されたスクリプトを実行する前にロジックをマージする必要があります。

Oracle ソース・サーバーの場合のロック

現在 Oracle ソースを更新中のアプリケーションがすべて終了してからでなければ、アプライ・プログラムはデータの適用を開始できません。

データを処理して同期点を設定できるように、アプライ・プログラムは CCD 表をロックする必要があります。 CCD 表に対するロックが保留されるのは、アプライ・サイクル全体を通してではなく、アプライ・プログラムが同期点を設定するまでの間だけです。ソース表を更新する必要のあるアプリケーションは、アプライ・プログラムが CCD 表をアンロックするまで待たなければなりません。

コード・ページ変換の計画

レプリケーション・コンポーネントは、異なるコード・ページを使用するデータの変換を各種のオペレーティング・システムの DB2 データベースに依頼するデータベース・アプリケーションです。

レプリケーション・コンポーネントは SQL SELECT、INSERT、UPDATE、および DELETE ステートメントを使用してデータを処理します。

互換コード・ページを持つデータベース間のレプリケーション

使用するレプリケーション構成において、異なるコード・ページを使用するシステム間で SQL ステートメントやデータをやり取りする必要がある場合は、DRDA® などの下層の DB2 プロトコルがコード・ページ変換を処理します。また、データが DB2 リレーショナル・データベースと非 DB2 リレーショナル・データベース間で渡される場合は、DB2 レプリケーションは必要なコード・ページ変換の処理をすべて下層のデータベース製品に依頼します。

異なるコード・ページを使用するデータベース間でデータを複製する予定である場合は、*IBM Information Management Software for zOS Solutions インフォメーション・センター*や *DB2 インフォメーション・センター*をチェックして、持っているコード・ページが互換性のあるものかどうかを判別してください。例えば、DB2 for Linux, UNIX and Windows を使用している場合は、文字データの変換に関するセクションを参照してください。

使用しているデータベースが互換コード・ページを持っていることを検査したら、次にそれらのデータベース間でコード・ページの使用方法に違いがあるかどうかを判別してください。例えば、あるデータベース製品では 1 つの表内のそれぞれの列ごとに異なるコード・ページを指定できる一方、別のデータベース製品ではコード・ページはデータベース・レベルでしか指定できないとします。最初の製品の複数のコード・ページが指定されている表は、2 番目の製品の単一データベースにはレプリケーションできません。したがって、各データベースによるコード・ページの処理方法は、使用している環境内の各種データベース間で必ず正常にデータがレプリケーションされるようにするためにはどのようにレプリケーションをセットアップする必要があるのかということに影響します。

レプリケーションに合わせた各国語サポート (NLS)

レプリケーション用の NLS 構成は、システム間のデータベース接続をセットアップする際に定義されます。ただし、キャプチャーまたはアプライ・プログラムを Linux、UNIX、または Windows オペレーティング・システムで実行している場合は、いくつかの構成ステップが必要になることがあります。

Linux、UNIX、および Windows では、キャプチャー・プログラムはデータの取り込み元となるデータベースと同じコード・ページを使用する必要があります。キャプチャー・プログラムがその同じコード・ページで稼働していない場合は、キャプチャーがデータベースと同じコード・ページを使用できるように、DB2CODEPAGE と呼ばれる DB2 環境変数またはレジストリー変数を設定する必要があります。

Linux、UNIX、またはWindowsでアプライ・プログラムを実行している際に、ソース表が UNICODE の場合は、アプライ・アプリケーション・コードが UNICODE である必要があります。ソース表のデータが ASCII の場合は、アプリケーション・コード・ページを ASCII または UNICODE にすることができます。アプライ・プログラムの DB2CODEPAGE 変数を設定することもできます。

コード・ページ変数の設定

DB2 は、アプリケーションのコード・ページをそのアプリケーションが実

行されているアクティブ環境から導き出します。通常、DB2CODEPAGE 変数が設定されていない場合は、コード・ページはオペレーティング・システムによって指定される言語 ID から導き出されます。ほとんどの場合、データベースの作成時にデフォルトのコード・ページを使用したのであれば、この値はキャプチャーまたはアプライ・プログラムにとって正しい値です。しかし、データベースの作成時にデフォルトのコード・ページ以外のコード・ページを明示的に指定した場合は、DB2CODEPAGE 変数を設定する必要があります。そうしない場合、データが正しく変換されない可能性があります。DB2CODEPAGE 変数に対して使用する値は、CREATE DATABASE ステートメントで指定する値と同じでなければなりません。 DB2CODEPAGE 変数の設定に関する詳細は、DB2 インフォメーション・センターを参照してください。

コード・ページからのレプリケーション

1 バイト文字セット (SBCS) コード・ページを使ったソース・データを Unicode UTF-8 を使ったターゲットに複製している場合、ソース・データベースの中のいくつかの 1 バイト文字は、DB2 によって、ターゲット・データベースで複数のバイトに変換される場合があります。 16 進値が 0x80 から 0xff であるすべての 1 バイト文字は、2 バイトの 1208 等価値に変換されます。つまり、場合によってはターゲット列はソース列よりも大きくなければならず、そうでない場合アプライ・プログラムが DB2 から SQL エラーを受け取ることがあります。

いくつかのデータベース製品のインプリメント・コード・ページのサポートは、他のコード・ページのサポートと異なることがあり、そのことがレプリケーション構成に影響する場合があります。例えば、System i 上の DB2 では、コード・ページの列レベルでの指定を許可していますが、DB2 for Linux, UNIX, and Windows ではデータベース・レベルでの指定しか許可されていません。したがって、異なるコード・ページを使用する複数の列を持つ System i 表がある場合、すべてのコード・ページに互換性がある場合を除き、これらの列を単一の DB2 forLinux,UNIX,and Windows データベースに対して複製することはできません。

LANG 変数の設定

Linux または UNIX システム上でキャプチャー・プログラムとアプライ・プログラムを実行している場合、LANG 環境変数を設定する必要が生じることがあります。キャプチャーおよびアプライ・プログラムは、この環境変数の内容を使って、使用されている言語のメッセージ・ライブラリーを検索します。例えば、LANG 環境変数が en_US に設定されると、キャプチャー・プログラムは DB2 インスタンスの /sqllib/msg/en_US サブディレクトリーにある英語のメッセージ・ライブラリーを検索します。キャプチャーがメッセージ・ライブラリーを見つけられなかった場合、IBMSNAP_CAPTRACE 表に書き込まれるすべてのメッセージは ASN0000S です。

DB2 for z/OS のレプリケーションの計画

SQL replication for DB2 for z/OS は、スキーマ名と表名を 128 バイトまでサポートします。

長い名前のサポートを利用するには、次のようにします。

- DB2 for z/OS バージョン 8 以降の新規関数モードで、キャプチャー・コントロール表、アプライ・コントロール表、およびモニター・コントロール表を作成する。
- DB2 for z/OS バージョン 8 以降の新規関数モードで、キャプチャー・サーバー、アプライ・サーバー、およびモニター・サーバーを実行する。

制約事項: DB2 for z/OS の新規関数モードのサブシステムと、DB2 for Linux, UNIX, and Windows、または DB2 for iSeries™ との間で複製する場合は、 30 バイト以下のスキーマ名を使用する必要があります。 DB2 for z/OS バージョン 8 の新規関数モードで 30 文字を超えるスキーマ名を使用すると、そのプラットフォームと DB2 for Linux, UNIX, and Windows、または DB2 for iSeries との間で複製を実行できません。

パフォーマンス・チューニング

パフォーマンス・チューニングには、パフォーマンスを最適化するためにレプリケーション環境のチューニングを行うことが含まれます。

「*WebSphere Information Integration Tuning for SQL Replication Performance*」は、DB2 レプリケーション環境の主なコンポーネントを調整して、最適なパフォーマンスを得る方法を説明しています。この資料は、ご使用の製品に関するオンラインの WebSphere Information Integration サポート・サイトで入手できます。

第 2 章 SQL レプリケーション用のユーザー ID とパスワードのセットアップ

SQL レプリケーション・プログラムを使用するには、ローカルおよびリモート・システム上の DB2 サーバーにアクセスするためのユーザー ID とパスワードをセットアップする必要があります。

以下のトピックでは、必要な権限と特権について説明し、SQL レプリケーション・プログラムが共用できる暗号化されたパスワード・ファイル中に必要なユーザー ID とパスワードを保管する方法について解説します。

管理の許可要件

レプリケーションをセットアップするには、生成済みの SQL を実行してオブジェクトを作成し、プランをバインドして SQL パッケージを作成します (System i)。これらのタスクに必要な権限は、オペレーティング・システムによって異なります。

レプリケーションを管理するためには、レプリケーション構成に関与するすべてのデータベースに対するユーザー ID を少なくとも 1 つは持っている必要があり、さらにそのユーザー ID にはレプリケーションをセットアップする権限がなければなりません。使用するユーザー ID は、すべてのシステムで同じである必要はありません。ただし、同じであった方が楽です。

z/OS Linux UNIX Windows

レプリケーションのセットアップに使用するユーザー ID で以下のタスクを実行できることを確認してください。

- すべてのサーバー (ソース・サーバー、キャプチャー・コントロール・サーバー、アプライ・コントロール・サーバー、モニター・コントロール・サーバー、およびターゲット・サーバー) への接続
- ソース・サーバー、キャプチャー・コントロール・サーバー、モニター・コントロール・サーバー、およびターゲット・サーバーに置かれているカタログ表からの選択
- ソース・サーバー、モニター・コントロール・サーバー、キャプチャー・コントロール・サーバー、およびアプライ・コントロール・サーバーでの表 (レプリケーション・コントロール表も含む)、表スペース、およびビューの作成
- ターゲット・サーバーでの表および表スペースの作成 (新しいターゲット表の作成にレプリケーション・プログラムを使用する場合) (既存の表をターゲットとして使用する場合は必須ではありません)
- レプリケーションに関与するそれぞれの DB2 データベース (ソース・サーバー、ターゲット・サーバー、モニター・コントロール・サーバー、およびアプライ・コントロール・サーバーも含む) でのプランのバインドまたはパッケージの作成

- 共有ライブラリーを使用したストアード・プロシージャーの作成およびストアード・プロシージャーの呼び出し (Linux、UNIX と Windows のみ)。

非 DB2 リレーショナル・データベースの場合は、使用するユーザー ID で以下のアクションを実行できなければなりません。

- 表の作成
- ソース表およびコントロール表に対するキャプチャー・トリガーの作成
- プロシージャーの作成
- フェデレーテッド・データベースでのニックネームの作成
- シーケンスの作成 (Oracle データベースの場合のみ)
- カタログ表からの選択

ほとんどのレプリケーション管理者は DBADM 特権または SYSADM 特権を持っています。 DB2 for z/OS の場合、レプリケーション管理者は、少なくともカタログからの選択を許可されている必要があり、さらに索引作成特権も含め、ASN スキーマを使用した表の作成、およびソース表の特性を持つ CD 表とターゲット表の作成を行うために必要なすべての特権を持っている必要があります。

System i

レプリケーションのセットアップに使用するユーザー ID で以下のタスクを実行できることを確認してください。

- すべてのサーバー (ソース・サーバー、キャプチャー・コントロール・サーバー、アプライ・コントロール・サーバー、モニター・コントロール・サーバー、およびターゲット・サーバー) への接続
- ソース・サーバー、キャプチャー・コントロール・サーバー、モニター・コントロール・サーバー、およびターゲット・サーバーに置かれているカタログ表からの選択
- ソース・サーバー、モニター・コントロール・サーバー、キャプチャー・コントロール・サーバー、およびアプライ・コントロール・サーバーでの表 (レプリケーション・コントロール表も含む) とビューの作成
- ターゲット・サーバーでの表の作成 (新しいターゲット表の作成に DB2 レプリケーション・プログラムを使用する場合) (既存の表をターゲットとして使用する場合は必須ではありません)
- レプリケーションに関与するそれぞれの DB2 データベース (ソース・サーバー、ターゲット・サーバー、モニター・コントロール・サーバー、およびアプライ・コントロール・サーバーも含む) でのプランのバインドまたはパッケージの作成

ほとんどのレプリケーション管理者は DBADM 特権または SYSADM 特権を持っています。

DPR 権限付与 (GRTDPRAUT) コマンドは、ユーザーがソースの登録、それらのソースへのサブスクライブ、およびコントロール表の作成を行うのを許

可するために使用します。System i システム間でのみレプリケーションを行う場合は、すべてのサーバーに対して同じユーザー ID を使用する必要があります。

DPR 権限付与 (GRTDPRAUT) コマンドがマシンにインストールされていない場合は、オブジェクト権限付与 (GRTOBJAUT) コマンドを使用する必要があります。

キャプチャー・プログラムの許可要件

キャプチャー・プログラムを実行するユーザー ID は、DB2 システム・カタログにアクセスできなければなりません。また、キャプチャー・コントロール・サーバー上のすべてのレプリケーション・コントロール表へのアクセスと更新が可能で、さらにキャプチャー・プログラム・パッケージを実行できなければなりません。

レプリケーション管理者のユーザー ID を使用してキャプチャー・プログラムを実行できますが、これは要件ではありません。

z/OS

キャプチャー・プログラムを実行するために使用するユーザー ID は、USS にアクセスできるものとして登録する必要があります。これは、z/OS UNIX または OS/390® UNIX (OMVS セグメントを持っていなければならない) を使用するためのユーザー ID を定義する必要があることを意味します。

また、キャプチャー・ロード・ライブラリーに APF 許可が与えられていること、およびキャプチャー・プログラムを実行するユーザー ID が以下の特権を持っていることも確認してください。

- 一時ディレクトリー (/tmp ディレクトリーか、 TMPDIR 環境変数によって指定されたディレクトリーのいずれか) への WRITE アクセス権。
- キャプチャー・コントロール・サーバー上のすべてのレプリケーション表に対する SELECT、UPDATE、INSERT、および DELETE 特権。
- DB2 カタログ (SYSIBM.SYSTABLES、SYSIBM.SYSCOLUMNS、および SYSIBM.SYSPLAN) に対する SELECT 特権。
- TRACE 特権。
- MONITOR1 および MONITOR2 特権。
- キャプチャー・プログラム・パッケージに対する EXECUTE 特権。

また、そのユーザー ID にキャプチャー・パス・ディレクトリーに対する WRITE アクセス権があること (USS)、または上位修飾子が付けられていること (z/OS) を確認してください。キャプチャー・プログラムを USS シェルで実行するためには、STEPLIB システム変数が設定されていて、さらにこの変数にキャプチャー・ロード・ライブラリーが組み込まれている必要があります。 PATH には HFS パス (/usr/lpp/db2repl_09_01/bin) が含まれていなければなりません。

Linux UNIX Windows

キャプチャー・プログラムを実行するユーザー ID が以下の権限および特権を持っていることを確認してください。

- DBADM または SYSADM 権限。

- **capture_path** パラメーターで指定されたディレクトリーに対する WRITE 特権。キャプチャー・プログラムは、このディレクトリー内に診断ファイルを作成します。

System i

DPR 権限付与 (GRTDPRAUT) コマンドは、ユーザーにローカル・システムでのキャプチャー・プログラムの実行を許可するために使用します。System i システム間でのみレプリケーションを行う場合は、すべてのサーバーに対して同じユーザー ID を使用する必要があります。GRTDPRAUT コマンドがマシンにインストールされていない場合は、オブジェクト権限付与 (GRTOBJAUT) コマンドを使用する必要があります。

アプライ・プログラムの許可要件

アプライ・プログラムを実行するユーザー ID は、DB2 システム・カタログにアクセスできなければなりません。また、キャプチャー・コントロールおよびターゲット・サーバー上のすべてのレプリケーション・コントロール表へのアクセスと更新が可能で、さらにアプライ・プログラム・パッケージを実行できなければなりません。

使用するレプリケーション環境内のそれぞれのサーバーごとに異なるユーザー ID を使用できます。レプリケーション管理者のユーザー ID を使用してアプライ・プログラムを実行できますが、これは要件ではありません。

z/OS

アプライ・プログラムを実行するユーザー ID が以下の権限および特権を持っていることを確認してください。

- 一時ディレクトリー (/tmp ディレクトリーか、 TMPDIR 環境変数によって指定されたディレクトリーのいずれか) への WRITE アクセス権。
- アプライ・コントロール・サーバー上のすべてのレプリケーション表に対する SELECT、UPDATE、INSERT、および DELETE 特権。
- DB2 カタログ (SYSIBM.SYSTABLES、SYSIBM.SYSCOLUMNS、および SYSIBM.SYSPLAN) に対する SELECT 権限。

注: アプライ・プログラムを実行するために使用するユーザー ID は、 USS にアクセスできるものとして登録する必要があります。これは、z/OS UNIX または OS/390 UNIX (OMVS セグメントを持っていなければならない) を使用するためのユーザー ID を定義する必要があることを意味します。ロード・ライブラリーに APF 許可が必要となるのは、アプライ・プログラムが ARM の指定付きで登録される場合だけです。アプライ・プログラムを USS シェルで実行するためには、STEPLIB システム変数が設定されていて、さらにこの変数にアプライ・ロード・ライブラリーが組み込まれている必要があります。 PATH には HFS パス (/usr/lpp/db2repl_09_01/bin) が含まれていなければなりません。

Linux UNIX Windows

アプライ・プログラムを実行するユーザー ID が以下の権限および特権を持っていることを確認してください。

- アプライ・パス・ディレクトリーに対する書き込み特権

- レプリケーション・ソース表 (関連した CD 表および CCD 表も含む) へのアクセス特権
- レプリケーション・ターゲット表に対するアクセスおよび更新特権
- レプリケーション・プログラムによって生成されて、キャプチャー・コントロール・サーバーおよびアプライ・コントロール・サーバーでビルドされたすべてのコントロール表に対するアクセスと更新特権
- アプライ・プログラムによって使用される任意のパスワード・ファイルの読み取り特権

注: 使用するソース表が非 DB2 リレーショナル・データベース管理システム上にある場合は、ユーザー ID はフェデレーテッド・データベースおよび非 DB2 リレーショナル・データベースの両方において、フェデレーテッド・データベースで定義されているニックネームを使用してソース表にアクセスできるだけの十分な特権を持っている必要があります。

System i

DPR 権限付与 (GRTDPRAUT) コマンドは、ユーザーにローカル・システムでのアプライ・プログラムの実行を許可するために使用します。System i システム間でのみレプリケーションを行う場合は、すべてのサーバーに対して同じユーザー ID を使用する必要があります。GRTDPRAUT コマンドがマシンにインストールされていない場合は、オブジェクト権限付与 (GRTOBJAUT) コマンドを使用する必要があります。

非 DB2 データベース

コントロール表が非 DB2 データベース上にある場合、変更されたデータを非 DB2 リレーショナル・ターゲットに入れたり、そこからデータを取り出したりするユーザー ID は、フェデレーテッド・データベースおよび非 DB2 リレーショナル・データベースにおいて十分な特権を持っている必要があります。

非 DB2 リレーショナル・ターゲットの場合、アプライ・プログラムを実行中のユーザー ID には、フェデレーテッド・データベース上のニックネームに対して WRITE を実行できる特権と、ユーザー・マッピングを介して実際の非 DB2 ターゲットへの WRITE を実行できる特権が必要です。

非 DB2 リレーショナル・ソースの場合は、アプライ・プログラムを実行中の ID には以下の特権が必要です。

- フェデレーテッド・データベース上のニックネームに対して READ と WRITE を実行できる特権、およびユーザー・マッピングを介してキャプチャー CONTROL 表に対し READ と WRITE を実行できる特権
- フェデレーテッド・データベース上のニックネームからの READ を実行できる特権、およびユーザー・マッピングを介して非 DB2 サーバー上の実際の CCD 表からの READ を実行できる特権
- フェデレーテッド・データベース上のニックネームからの READ を実行できる特権、およびユーザー・マッピングを介して非 DB2 サーバー上の実際のソース表からの READ を実行できる特権

非 DB2 リレーショナル・データベースでのキャプチャー・トリガーの場合の許可要件

非 DB2 データベースから複製する場合は、ソースからの変更のキャプチャーにはキャプチャー・トリガーが使用されます。リモート・ソース表を変更するリモート・ユーザー ID (例えば、ユーザー・アプリケーションからの) には、CCD 表への挿入を行う権限が必要です。

ほとんどの場合、INSERT、UPDATE、または DELETE トリガーを実行するのに明示的な権限は必要ありません。なぜなら、表に対してトリガーが定義されると、それ以降トリガーは INSERT、UPDATE、または DELETE を実行中のアプリケーションにとって透過的に実行されるからです。 Informix® データベースの場合は、登録済みのソース表に対して INSERT、UPDATE、および DELETE アクションを実行するリモート・ユーザー ID には EXECUTE PROCEDURE 特権が必要です。

SQL レプリケーション用のユーザー ID およびパスワードの保管 (Linux、UNIX、Windows)

asnpwd コマンドは、アプライ・プログラム、レプリケーション・アラート・モニター、およびレプリケーション・アナライザーがリモート・サーバー上のデータにアクセスできるように、パスワード・ファイルを作成して保守するために使用します。キャプチャー・プログラムはパスワード・ファイルを必要としません。

パスワード・ファイル内の情報は、機密性を確保するために暗号化されます。

レプリケーション環境が複数のサーバーに分散されていない場合は、ユーザー ID とパスワードを保管する必要はありません。しかし、ほとんどのレプリケーション環境では、データは複数のサーバーに分散されています。そのような環境がある場合は、データベースへの接続を試行する時に有効なユーザー ID とパスワードを入力して、DB2 がユーザーの ID を検証できるようにする必要があります。

レプリケーション・センターの場合とそれ以外のレプリケーション・プログラムの場合とでは、パスワード情報の保管方法が異なります。レプリケーション・センターの場合のパスワード要件に関する情報は、オンライン・ヘルプを参照してください。

第 3 章 SQL レプリケーション用のサーバーの構成

データを複製できるようにするには、その前にサーバーの作成と構成を行って、サーバー同士が接続できるようにしなければなりません。

z/OS z/OS でのサーバーの構成に関する詳細は、「*IBM WebSphere Information Integration z/OS 版 レプリケーション、インストール、およびカスタマイズ・ガイド*」を参照してください。

SQL レプリケーションのための接続要件

アプライ・プログラム、レプリケーション・センター、またはレプリケーション・コマンドを実行するワークステーションは、必ずソース・サーバー、キャプチャー・コントロール・サーバー、アプライ・コントロール・サーバー、およびターゲット・サーバーのデータベースに接続できなければなりません。

レプリケーション・アラート・モニターを使用する場合、これを実行するワークステーションは、モニター・コントロール・サーバーとこのモニターがモニターするあらゆるサーバーに接続できなければなりません。レプリケーション・センターを使用してモニターをセットアップする場合は、レプリケーション・センターがモニター・コントロール・サーバーに接続できることを確認してください。

レプリケーションの設計で、ソース・データベースとは異なるサーバーでデータのステージングが関係しているなら、さまざまなサーバー間の通信について慎重に考慮する必要があります。エミュレーション層の数、LAN ブリッジの数、およびルーター・リンクの数については、どれもレプリケーションのパフォーマンスに影響を与えるものなので、必ず制限してください。

データベースがネットワークに接続される場合のコネクティビティーは、接続されるオペレーティング・システムによって異なります。

以下のトピックでは、接続要件を詳しく説明しています。

Windows から System i サーバーへの接続

System i サーバー上のレプリケーションの管理を、Windows ワークステーションから接続して実行できます。

始める前に

- DB2 または DB2 Connect™ がワークステーションにインストールされていなければならない。
- TCP/IP がワークステーションにセットアップされている必要があること。

手順

DB2 for Windows ワークステーションから System i サーバーに接続する場合

1. System i サーバーにログオンして、リレーショナル・データベースを探します。

a. 接続先となる System i サーバーにログオンします。
b. dsprdbdire コマンドをサブミットしてから、 *LOCAL に local と指定します。
c. 出力の中から、リレーショナル・データベースの名前を探します。 例えば、以下の出力では、データベースの名前は DB2400E です。

```
MYDBOS2             9.112.14.67
RCHASDPD            RCHASDPD
DB2400E             *LOCAL
RCHASLJN            RCHASLJN
```

2. System i データベースを DB2 for Windows のカタログに入れます。
   a. Windows コマンド・プロンプトで、db2clp を入力します。 DB2 CLP コマンド・ウィンドウが開きます。
   b. コマンド・ウィンドウで、以下の 3 つのコマンドを正確な順序で入力します。

```
db2 catalog tcpip node server_name remote server_name server 446 system
server_name ostype OS400

db2 catalog dcs database rdb_name AS rdb_name

db2 catalog database rdb_name AS rdb_name at node server_name
authentication dcs
```

   ここで、*server_name* は System i システムの TCP/IP ホスト名で、*rdb_name* はステップ 1 で説明されている System i リレーショナル・データベースの名前です。
3. コマンド・ウィンドウで、以下のコマンドを発行します。

```
db2 terminate
```

4. System i システムにログオンするために使用する System i ユーザー・プロファイルが CCSID37 を使用していることを確認します。
   a. System i システムにログオンします。
   b. 以下のコマンドを入力します。ここで、*user* はユーザー・プロファイルです。

```
CHGUSRPRF USRPRF (user) CCSID(37)
```

   c. 次のように入力して、 DDM サーバーが System i システムで始動していることを確認します。

```
STRTCPSVR SERVER(*DDM)
```

5. DB2 for Windows および DB2 for System i が接続されていることを確認します。

```
db2 connect to rdb_name user user_name using password
```

非 DB2 リレーショナル・サーバーへの接続

非 DB2 リレーショナル・サーバーとの間で相互にデータのレプリケーションを行う場合は、その非 DB2 リレーショナル・サーバーにアクセスして接続できなければなりません。

非 DB2 リレーショナル・ソース・サーバーからのレプリケーションを試行する前に、使用するフェデレーテッド・サーバーとフェデレーテッド・データベースをセットアップする必要があります。セットアップには以下の 3 つのメイン・ステップがあります。

1. ラッパーを定義して、DB2 データベースが他の非 DB2 リレーショナル・データベースにアクセスできるようにする。
2. サーバー・マッピングを使用して非 DB2 リレーショナル・データベースを定義する。
3. DB2 データベースへの接続に使用されるユーザー ID とパスワードの組み合わせが非 DB2 リレーショナル・データベースへのアクセスに使用されるものとは異なる場合は、ユーザー・マッピングを作成する必要がある。

フェデレーテッド環境の構成に関する詳細は、DB2 インフォメーション・センターまたは「*WebSphere Information Integration フェデレーテッド・システム・ガイド*」を参照してください。

SQL レプリケーション用のコントロール表の作成

レプリケーション・プログラムは、コントロール表を使用して、登録済みの表、サブスクリプション・セット、稼働パラメーター、およびユーザー設定に関する情報を保管します。コントロール表は、レプリケーションのソースとターゲットを定義する前に作成します。

以下のトピックでは、SQL レプリケーション用のコントロール表の作成について説明しています。

SQL レプリケーション用のコントロール表の作成

ASNCLP コマンド行プログラムまたはレプリケーション・センターを使用して、キャプチャー・プログラム用とアプライ・プログラム用のコントロール表を作成できます。

制約事項

- レプリケーション・センターまたは ASNCLP が、コントロール表を作成するサーバーに接続できなければなりません。
- 複数のデータベース・パーティション環境では、コントロール表によって使用される表スペースすべてはカタログ・パーティション上になければなりません。既存の表スペースを使用する場合、その表スペースは、非パーティションであり、カタログ・パーティション上にある必要があります。

このタスクについて

コントロール表の作成方法をカスタマイズしない場合は、2 つの表スペース (1 つは UOW 表用、もう 1 つはその他のコントロール表用) が作成されます。それらのデフォルトのレプリケーション表スペースを使用したくない場合は、既存の表スペースを指定するか、新しい表スペースを作成するか、または現行の DB2 デフォルト表スペースを使用できます。

複数データベース・パーティション環境でキャプチャーを開始する場合、キャプチャーは IBMSNAP_RESTART 表と同じ表スペースに追加のコントロール表 (IBMSNAP_PARTITIONINFO) を作成します。

両方のレプリケーション管理ツールとも、特定のオペレーティング・システムまたは環境用のコントロール表を作成するときに使用すべきデフォルトが識別されるように、プロファイルを作成できます。それらのコントロール表に対してプロファイルを設定してしまえば、それ以降はコントロール表のセットを作成するたびにプロファイルを設定する必要がなくなります。コントロール表を作成する際に、デフォルトをオーバーライドできます。また、プロファイルもいつでも変更できます。ただし、その変更の影響を受けるのは、プロファイルの変更後に作成したコントロール表だけです。

手順

コントロール表を作成するには、次の方法のいずれかを使用します。

方法	説明

ASNCLP コマンド行プログラム	CREATE CONTROL TABLES FOR コマンドを使用して、新しいキャプチャーまたはアプライ・コントロール表のセットを作成します。例えば、以下のコマンドは環境を設定してキャプチャー・コントロール表を作成します。
	`SET SERVER CAPTURE TO DB SAMPLE`
	`SET OUTPUT CAPTURE SCRIPT "capctrl.sql";`
	`SET LOG "capctrl.err";`
	`SET RUN SCRIPT LATER;`
	`CREATE CONTROL TABLES FOR CAPTURE SERVER`
	`IN UW UOW TSUOW100 OTHERS TSASN100;`

レプリケーション・センター	キャプチャーおよびアプライに関する「コントロール表の作成」または「コントロール表の作成」-「クイック」ウィンドウを使用します。ウィンドウをオープンするには、オブジェクト・ツリー内の「キャプチャー・コントロール・サーバー」または「アプライ・コントロール・サーバー」フォルダーを右クリックして、以下のいずれかのオプションをクリックします。 • **「キャプチャー・コントロール表の作成」** • **「キャプチャー・コントロール表の作成」 → 「クイック」** • **「アプライ・コントロール表の作成」** • **「アプライ・コントロール表の作成」 → 「クイック」**

コントロール表の作成 (System i)

レプリケーション・コントロール表は、DB2 DataPropagator™ for System i のインストール時に自動的に作成されます。コマンドを使用して、コントロール表を作成することもできます。

このタスクについて

インストール時に、コントロール表は、DataPropagator のデフォルト・スキーマ (ASN) 内に作成されます (まだ存在していない場合)。コントロール表が誤って削除されたり破壊されたりした場合にも、コントロール表の追加のセットを作成できま

す。キャプチャーの場合は、別のスキーマでコントロール表の新しいセットを作成できます。スキーマは、最大 25 個まで作成できます。

ユーザー定義のファイル・システムでは、レプリケーション・コントロール表を基本補助記憶域プール (ASP) または独立補助記憶域プール (IASP) グループに作成できますが、両方に作成することはできません。コントロール表を IASP グループに作成する場合は、まず最初にすべてのキャプチャーおよびアプライ・コントロール表を基本 ASP から除去する必要があります。キャプチャーまたはアプライ・プログラムを開始する前に、ASN ライブラリー (またはキャプチャー・スキーマ用のライブラリー) を含む ASP グループに対して SETASPGRP コマンドを発行します。

手順

System i でコントロール表を作成するには、DPR 表の作成 (CRTDPRTBL) コマンドを使用します。

制約事項: CRTDPRTBL コマンドのみを使用して、System i でコントロール表を作成します。ASNCLP コマンド行プログラムとレプリケーション・センターは、System i でのコントロール表の作成をサポートしていません。

非 DB2 リレーショナル・ソース用のコントロール表の作成

レプリケーション・ソースとして Informix などの非 DB2 データベースを使用する場合は、レプリケーション・センターまたは ASNCLP コマンド行プログラムを使用して、コントロール表を作成できます。

このタスクについて

このようなタイプのソースの場合、レプリケーション・センターは非 DB2 リレーショナル・データベースに以下のキャプチャー・コントロール表を作成します。

- IBMSNAP_PRUNCNTL
- IBMSNAP_PRUNE_SET
- IBMSNAP_REG_SYNCH
- IBMSNAP_REGISTER
- IBMSNAP_SEQTABLE (Informix の場合のみ)
- IBMSNAP_SIGNAL

フェデレーテッド・データベース内に IBMSNAP_SEQTABLE 以外のすべての表のニックネームが作成されます。(この表は、Informix トリガーだけが使用します。アプライ・プログラムはこれを使用しません。)トリガーは、IBMSNAP_SIGNAL 表および IBMSNAP_REG_SYNCH 表に自動的に作成されます。

重要: IBMSNAP_SIGNAL 表および IBMSNAP_REG_SYNCH 表に作成されたトリガーは、除去または変更をしないでください。

キャプチャー・コントロール表のセットを複数作成する

1 台のサーバー上で複数のキャプチャー・プログラムを使用する場合は、キャプチャー・コントロール表のセットを複数作成して、それぞれの表セットが必ずユニークなキャプチャー・スキーマを持つようにする必要があります。

このタスクについて

このスキーマにより、ある表のセットを使用するキャプチャー・プログラムが識別されます。複数のキャプチャー・スキーマを使用すると、複数のキャプチャー・プログラムを並行実行できます。

次の状況で、複数のキャプチャー・プログラムを実行することができます。

- 待ち時間が短い表を他の表と異なる方法で扱うことにより、パフォーマンスを最適化するため。待ち時間が短い表がある場合は、それらの表のレプリケーションはそれら専用のキャプチャー・プログラムを使用して行うのが望ましいことがあります。この方法を使用すると、それらに異なるランタイム優先順位を与えることができます。また、キャプチャー・プログラムのパラメーター (整理インターバルやモニター・インターバルなど) を、それらの表の短い待ち時間に合うように設定することも可能です。
- より高いキャプチャー・スループットの可能性を提供するため。これは、複数の CPU を使用しているソース環境において、多大な利点が得られることがあります。スループット増加のトレードオフは、ログ・リーダーが複数存在することに関連した CPU オーバーヘッドの増加です。

同じフェデレーテッド・データベース内の複数の非 DB2 ソース・データベースから複製したい場合は、キャプチャー・コントロール表の複数のセットを作成し、各セットは独自のスキーマを持つ必要があります。または、希望する場合は、別のフェデレーテッド・データベースを使用することもできます。この場合、各サーバー上のキャプチャー・コントロール表はデフォルトの ASN スキーマを使用できます。

z/OS UNICODE コード化スキームと EBCDIC コード化スキームを別々に使用して作業する場合、または 1 つのサブシステムでキャプチャー・プログラムのインスタンスを複数実行する場合は、複数のキャプチャー・スキーマを使用できます。

System i DPR 表の作成 (CRTDPRTBL) コマンドを使用してキャプチャー・コントロール表の追加のセットを作成します。その際には、CAPCTLLIB パラメーターを使用してスキーマ名を指定します。

複数のデータベース・パーティション上でのキャプチャー・コントロール表

複数のパーティション・データベースにキャプチャー・コントロール表を作成する場合、これらのコントロール表によって使用される表スペースすべてはカタログ・パーティション上になければなりません。

既存の表スペースを使用する場合、その表スペースは、非パーティションであり、カタログ・ノード上にある必要があります。

キャプチャー・プログラムを初めて始動し、WARMSI スタート・モードを選択する場合は、 IBMSNAP_PARTITIONINFO 表はまだ存在していません。キャプチャー・プログラムはこの表および表のユニーク索引を、 IBMSNAP_RESTART 表があるの

と同じ表スペースに作成します。 IBMSNAP_PARTITIONINFO 表の作成後、キャプチャー・プログラムは、各データベース・パーティションについて表の中に 1 つの行を挿入します。

キャプチャー・プログラムを開始するのが初めてではなく、いずれかのウォーム・スタート・モードを選択する場合は、 IBMSNAP_PARTITIONINFO 表はすでに存在しています。レプリケーション・センターで、「**キャプチャーを最後に実行した後で 1 つまたは複数のパーティションが追加された**」というチェック・ボックスを選択している場合は、キャプチャー・プログラムは、キャプチャー・プログラムが最後に実行された後で追加された各データベース・パーティションについて、IBMSNAP_PARTITIONINFO 表内に 1 つの行を挿入します。

レプリケーション・プログラムのセットアップ

複製できるようにするには、その前にキャプチャー・プログラム、アプライ・プログラム、およびその他のレプリケーション・プログラムをご使用の環境のサーバー用にセットアップする必要があります。

以下のトピックでは、レプリケーション・プログラムにとって必要なセットアップについて説明しています。

レプリケーション・プログラムのセットアップ (Linux、UNIX、Windows)

環境変数の設定に必要なレプリケーション・プログラムをセットアップするには、キャプチャー・プログラム用にデータベースを準備し、オプションでパッケージをバインドしてください。

レプリケーション・プログラムのための環境変数の設定 (Linux、UNIX、Windows)

キャプチャー・プログラム、アプライ・プログラム、またはレプリケーション・アラート・モニター・プログラムを始動する前と停止する前、およびレプリケーション・センターまたはレプリケーション・システム・コマンドを使用する前に、環境変数を設定する必要があります。

手順

環境変数を設定するには、以下のようにします。

1. 以下のように入力して、DB2 インスタンス名 (DB2INSTANCE) の環境変数を設定します。

オペレーティング・システム	コマンド
Linux UNIX	`export DB2INSTANCE=`*`db2_instance_name`*
Windows	`SET DB2INSTANCE=`*`db2_instance_name`*

2. デフォルトのコード・ページ値以外のコード・ページを使用してソース・データベースを作成した場合は、 DB2CODEPAGE 環境変数をそのコード・ページに設定します。 **注:** キャプチャーは、データを取り込んでいるデータベースのコー

ド・ページと同じコード・ページで実行されている必要があります。DB2 は、キャプチャーが実行されているアクティブ環境からキャプチャー・コード・ページを導き出します。DB2CODEPAGE が設定されていない場合は、DB2 はコード・ページ値をオペレーティング・システムから取得します。データベースの作成時にデフォルトのコード・ページを使用したのであれば、オペレーティング・システムから取得した値はキャプチャーにとって正しいものです。

3. オプション: 環境変数 DB2DBDFT をソース・サーバーに設定します。
4. Linux UNIX 使用しているシステムに固有のライブラリー・パス・システム変数および実行可能モジュール・パス・システム変数に、レプリケーション用のライブラリーと実行可能モジュールがインストールされているディレクトリーが組み込まれていることを確認します。

キャプチャー・プログラムを実行するための DB2 データベースの準備 (Linux、UNIX、Windows)

キャプチャー・プログラムを実行するために DB2 データベースを準備するには、アーカイブ・ロギングを有効にします。他のデータベース構成パラメーターを設定することもできます。

手順

キャプチャー・プログラムを実行するために DB2 データベースを準備するには、以下のようにします。

1. 以下のようなコマンドを入力して、キャプチャー・コントロール・サーバーのデータベースに接続します。

   ```
   db2 connect to database
   ```

   ここで、*database* はキャプチャー・コントロール・サーバーのデータベースです。
2. アーカイブ・ロギングを有効にして、キャプチャー・コントロール・サーバーのデータベースを、ロールフォワード・リカバリーのために準備します。これを行うには、update database configuration (データベース構成の更新) コマンド (ログ保存リカバリー) と backup database (データベースのバックアップ) コマンドを発行します。

   注: キャプチャー・プログラムは、ロールフォワード・リカバリーを指定するために LOGRETAIN と LOGARCHMETH1 パラメーターの両方をサポートしています。LOGARCHMETH1=LOGRETAIN を指定すると、キャプチャー・プログラムは自動的に LOGRETAIN RECOVERY を設定します。

   複数データベース・パーティション環境の場合、キャプチャーが変更を取り込む各パーティションに対してロールフォワード・リカバリーができるように各パーティションを設定しなければなりません。
3. インストール要件に基づいて構成値を大きくする必要がある場合があります。
   - 多数の行または非常に大きな行を使用するトランザクションの場合は、キャプチャーの **memory_limit** パラメーターの値を大きくすることが推奨されます。

- 多くの大規模ワークステーション・シナリオの場合、適切なデータベース構成値は以下のとおりです: APPLHEAPSZ 1000、LOGFILSIZ 4000、LOGPRIMARY 8、LOGSECOND 40、DBHEAP 1000、LOGBUFSZ 16、MAXAPPLS 200。

オプション : キャプチャー・プログラム・パッケージのバインディング (Linux、UNIX、Windows)

Linux、UNIX、および Windows では、キャプチャー・プログラムは実行時に自動的にバインドされます。バインド・オプションを指定する場合、バインドをスケジュールする場合、またはすべての BIND 処理が正常に完了したことをチェックする場合は、手動でパッケージをバインドできます。

手順

キャプチャー・プログラム・パッケージをバインドするには、以下のようにします。

1. 以下のようなコマンドを入力して、キャプチャー・コントロール・サーバーのデータベースに接続します。

   ```
   db2 connect to database
   ```

   ここで、*database* はキャプチャー・コントロール・サーバーのデータベースです。
2. キャプチャー・プログラムのバインド・ファイルがあるディレクトリーに変更します。

   Linux UNIX

   db2homedir/sqllib/bnd

   ここで、*db2homedir* は DB2 インスタンスのホーム・ディレクトリーです。

   Windows

   drive:¥...¥sqllib¥bnd

   ここで、*drive:* は DB2 のインストール先のドライブです。
3. 以下のようなコマンドを入力して、キャプチャー・プログラム・パッケージを作成してソース・サーバー・データベースにバインドします。

   ```
   db2 bind @capture.lst isolation ur blocking all
   ```

   ただし ur は、パフォーマンスを向上するための非コミット読み取り形式のリストを指定します。

これらのコマンドは、パッケージを作成します。その名前は、capture.lst ファイル内で見つけることができます。

オプション : アプライ・プログラム・パッケージのバインディング (Linux、UNIX、Windows)

Linux、UNIX、および Windows では、アプライ・プログラムは実行時に自動的にバインドされます。バインド・オプションを指定する場合、バインドをスケジュール

する場合、またはすべての BIND 処理が正常に完了したことをチェックする場合は、手動でパッケージをバインドできます。

手順

アプライ・プログラム・パッケージをバインドするには、以下のようにします。

1. アプライ・プログラムのバインド・ファイルがあるディレクトリーに変更します。

   Linux UNIX

   db2homedir/sqllib/bnd

   ここで、*db2homedir* は DB2 インスタンスのホーム・ディレクトリーです。

   Windows

   drive:¥...¥sqllib¥bnd

   ここで、*drive:* は DB2 のインストール先のドライブです。

2. アプライ・プログラムが接続するそれぞれのソース・サーバー、ターゲット・サーバー、キャプチャー・コントロール・サーバー、およびアプライ・コントロール・サーバーごとに、以下のステップを実行します。

   a. 以下のコマンドを入力して、データベースに接続します。

      ```
      db2 connect to database
      ```

      ここで、*database* はソース・サーバー、ターゲット・サーバー、キャプチャー・コントロール・サーバー、またはアプライ・コントロール・サーバーです。データベースがリモート・データベースとしてカタログされている場合は、db2 connect to コマンドでユーザー ID とパスワードを指定する必要があります。以下に例を示します。

      ```
      db2 connect to database user userid using password
      ```

3. 次の 2 つのコマンドを両方入力して、アプライ・プログラム・パッケージを作成してデータベースにバインドします。

   ```
   db2 bind @applycs.lst isolation cs blocking all grant public
   db2 bind @applyur.lst isolation ur blocking all grant public
   ```

   なお、cs はカーソル固定形式でリストを指定するのに対し、ur は非コミット読み取り形式でリストを指定します。

これらのコマンドは、パッケージを作成します。その名前は、applycs.lst および applyur.lst ファイル内で見つけることができます。

Sybase ソースでのアプライ・プログラム・パッケージのバインド

Sybase データ・ソースから複製する場合、アプライ・プログラム・パッケージを手動でバインドして、カーソル固定 (CS)の分離レベルを指定する必要があります。

このタスクについて

Sybase データ・ソースへ接続するために SQL レプリケーションが使用する Sybase ラッパーでは、CS の分離レベルが必要です。アプライ・パッケージを手動でバインドせずに Sybase から複製すると、SQL1822N エラーが表示されます。

手順

Sybase ソースでアプライ・パッケージを手動でバインドするには、以下のようにします。

1. Sybase ソース・データベース、アプライ・コントロール・サーバー、およびターゲット・データベースに接続します。
2. 以下のコマンドを使用して、分離レベルを CS にセットします。

```
db2 bind ~/sqllib/bnd/@applyur.lst isolation cs
db2 bind ~/sqllib/bnd/@applycs.lst isolation cs
```

リモート・システムで使用するための SQL パッケージの作成 (System i)

System i では、CRTSQLPKG コマンドを使用してパッケージを作成しなければならない場合があります。

このタスクについて

このコマンドを使用してパッケージを作成する場合は、以下のとおりです。

- リモート・ジャーナリングを使用する場合。キャプチャー・プログラムが実行されているシステムで CRTSQLPKG コマンドを実行します。その際に、ソース表が配置されているシステムを参照します。
- ADDDPRSUB または ADDDPRSUBM コマンドを使用してサブスクリプション・セットまたはサブスクリプション・セットのメンバーを追加する前。ターゲット・サーバーで CRTSQLPKG コマンドを実行します。その際に、以下のガイドラインに従います。
  - ソース表が別のマシン上にある場合は、そのソース表が配置されているシステムを指示する。
  - アプライ・コントロール・サーバーが別のマシン上にある場合は、そのアプライ・コントロール・サーバーを指示する。

SQL パッケージを使用すると、レプリケーション・プログラムが分散レプリケーション環境で稼働できるようになります。その場合、レプリケーションが System i システム間で行われる環境であっても、System i システムと他のいずれかのオペレーティング・システム (Linux、 UNIX、または Windows など) との間で行われる環境であっても構いません。

CRTSQLPKG コマンドの使用法については、 「*DB2 for i5/OS SQL プログラミング*」を参照してください。

パッケージは ASN 修飾子を使って作成されます。 System i では、パッケージは ASN ライブラリーに作成されます。他のオペレーティング・システムでは、パッケージは ASN スキーマに作成されます。

以下のトピックでは、SQL パッケージを作成するための詳細が記載されています。

アプライ・プログラム用の SQL パッケージの作成 (System i)

System i では、アプライ・プログラムの SQL パッケージを作成して、接続する必要のあるすべてのリモート・サーバーとアプライ・プログラムが対話できるようにする必要があります。

手順

アプライ・プログラムの SQL パッケージを作成するには、次のようにします。

アプライ・プログラムがリモート・システムに接続できるようにするために、アプライ・プログラムが実行されているシステムで以下のコマンドを実行します。

```
CRTSQLPKG PGM(QDP4/QZSNAPV2) RDB(remote_system)
```

remote_system は、アプライ・プログラムが接続する必要のあるリモート・システムでのリレーショナル・データベース入り口名です。

レプリケーション・アナライザー用の SQL パッケージの作成 (System i)

System i では、レプリケーション・アナライザーの SQL パッケージを作成して、キャプチャー・コントロール・サーバーやターゲット・サーバーといった、分析の対象としているサーバーとレプリケーション・アナライザーが対話できるようにする必要があります。

手順

レプリケーション・アナライザーの SQL パッケージを作成するには、次のようにします。

レプリケーション・アナライザーが実行されているシステムで以下のコマンドを実行してください。

```
CRTSQLPKG PGM(QDP4/QZSNANZR) RDB(remote_system)
```

ここで、*remote_system* は分析の対象としているシステムの名前です。

SQL パッケージへの特権の付与 (System i)

System i では、SQL パッケージを作成した後に、ソース・データベースに登録されているファイルにサブスクライブするすべてのユーザーに対して *EXECUTE 特権を付与する必要があります。

手順

SQL パッケージに関する特権を付与するには、以下のようにします。

ソース・データベースが常駐している System i システムにログオンし、以下の方法のいずれかを使用してください。

方法	説明

GRTOBJAUT コマンド	オブジェクト権限付与 (GRTOBJAUT) コマンドを使用します。例えば、以下のようにします。
	`GRTOBJAUT OBJ(ASN/package_name) OBJTYPE(*SQLPKG)`
	`USER(subscriber_name) AUT(*OBJOPR *EXECUTE)`

方法	説明

GRANT SQL ステートメント	SQL を使用してソース・データベースに接続し、以下の GRANT SQL ステートメントを実行する。
	`CONNECT TO data_server_RDB_name`
	`GRANT EXECUTE ON PACKAGE ASN/package_name TO subscriber_name`

GRTDPRAUT コマンド	GRTDPRAUT コマンドを使用する (このコマンドがローカル・システムにインストールされている場合)。

レプリケーション・プログラムのセットアップ (z/OS)

z/OS に SQL レプリケーションをインストールする際に、レプリケーション・プログラムをセットアップしてカスタマイズする必要があります。

「*WebSphere Information Integration Replication Installation and Customization Guide for z/OS*」に記載されている説明を参照してください。

複数データベース・パーティションのキャプチャー

DB2 Enterprise Server Edition でデータを複製している場合、複数のデータベース・パーティションにまたがるソース表への変更をキャプチャーできます。

キャプチャー・プログラムは、同じパーティション・グループに属するデータベース・パーティションのリストを IBMSNAP_PARTITIONINFO 表に保持します。この表は、キャプチャー・プログラムが初めて開始したときに同じパーティション・グループに 1 つ以上のデータベース・パーティションがあることを検出したときに、キャプチャー・プログラムによって作成されます。

キャプチャー・プログラムがウォーム・スタートされるたびに、キャプチャーはそのコントロール表があるパーティション・グループのデータベース・パーティションのリストを読み取ります。キャプチャーは DB2 に認識されているデータベース・パーティションの数と、 IBMSNAP_PARTITIONINFO 表にリストされているデータベース・パーティションの数を比較します。IBMSNAP_PARTITIONINFO 表にリストされているデータベース・パーティションの数が DB2 に認識されている数と一致していなければなりません。一致していない場合、キャプチャー・プログラムは実行しません。

最後にキャプチャー・プログラムを実行した後で 1 つ以上のデータベース・パーティションを追加した場合には、新規データベース・パーティションをキャプチャー・プログラムが認識するようにしなければなりません。これはレプリケーション・センターで、「キャプチャーの開始」ウィンドウ上でいずれかのウォーム・スタート・モードに対して **STARTMODE** パラメーターを設定する際に「**キャプチャーを最後に実行した後で 1 つまたは複数のパーティションが追加された**」チェック・ボックスを選択することで行えます。

ジャーナルのセットアップ (System i)

DB2 DataPropagator for System i は、データに対する変更についてジャーナルから受け取る情報を使用することによって、レプリケーションのための CD および UOW 表にデータを入れます。

DB2 DataPropagator for System i は、大部分の操作のコミットメント・コントロール下で実行するため、コントロール表のジャーナリングを必要とします。(QSQJRN ジャーナルは、CRTDPRTBL コマンドが集合を作成するときに作成されます。)

管理者は、ソース表、CD 表、およびターゲット表が含まれているライブラリーにジャーナルが含まれていることを確認する必要があります。すべてのソース表のジャーナル処理が正しく行われるのを見届ける責任もあります。

System i でレプリケーション用の表を登録する前に、その表が変更前イメージと変更後イメージの両方についてジャーナリングされている必要があります。

以下のトピックでは、レプリケーションに必要なジャーナル・セットアップについて説明しています。

ソース表用ジャーナルのセットアップ (System i)

ソース表のジャーナリングをセットアップするには、ジャーナル・レシーバーを作成し、ジャーナルを作成してから、ジャーナリングを開始します。

始める前に

ソース表のジャーナルおよびジャーナル・レシーバーを定義するために作成する権限を所有している必要があります。

制約事項

ソース表のために使用するジャーナルは、DB2 DataPropagator for System i によって ASN スキーマ (またはこれ以外のキャプチャー・スキーマ) のライブラリーに作成されたジャーナル以外のものにしてください。

手順

ソース表のジャーナルを作成するには、以下のようにしてください。

1. ジャーナル・レシーバーの作成 (CRTJRNRCV) コマンドを使用して、選択したライブラリーにジャーナル・レシーバーを作成します。 定期的に保管されるライブラリーにジャーナル・レシーバーを置きます。将来ジャーナル・レシーバーの命名規則を作成するのに使用できるようなジャーナル・レシーバー名 (RCV0001 など) を選択します。 *GEN オプションを使用して、ジャーナル・レシーバーを変更するときに命名規則を継続することができます。このタイプの命名規則は、ジャーナル・レシーバーの変更をシステム管理に許可する場合にも役立ちます。 以下の例では、ジャーナル・レシーバー用に JRNLIB という名前のライブラリーを使用します。

   ```
   CRTJRNRCV  JRNRCV(JRNLIB/RCV0001)
              THRESHOLD(100000)
              TEXT('DataPropagator Journal Receiver')
   ```

2. 以下の例のように、ジャーナルの作成 (CRTJRN) コマンドを使用して、ジャーナルを作成します。

   ```
   CRTJRN  JRN(JRNLIB/DJRN1)
           JRNRCV(JRNLIB/RCV0001)
           MNGRCV(*SYSTEM) DLTRCV(*YES)
           TEXT('DataPropagator Journal')
   ```

   - ステップ 1 で作成したジャーナル・レシーバーの名前を指定します。

- アタッチされたレシーバーが大きくなりすぎる場合、システムがジャーナル・レシーバーを変更して、新しいレシーバーをアタッチするために、レシーバーの管理 (MNGRCV) パラメーターを使用します。このオプションを選択すると、CRTJRN コマンドを使って、手動でレシーバーを切り離したり、新しいレシーバーを作成およびアタッチしたりする必要がなくなります。
- デフォルトの属性 MINENTDTA(*NONE) を使用します。このキーワードには、他の値は無効です。
- オーバーライドする理由がある場合 (例えば、リカバリーさせる目的でこれらのジャーナル・レシーバーを保管する必要がある場合) にのみ、DLTRCV(*NO) を指定します。 DLTRCV(*YES) を指定すると、これらのレシーバーは、保管する機会もなく削除されてしまう可能性があります。

CRTJRN コマンドの RCVSIZOPT パラメーターに 2 つの値 (*RMVINTENT および *MINFIXLEN) を使用して、ストレージの可用性やシステム・パフォーマンスを最適化することができます。詳細については、「*System i Programming: Performance Tools Guide*」を参照してください。

3. 以下の例で示す方法で、物理ファイルのジャーナル開始 (STRJRNPF) コマンドを使って、ソース表のジャーナリングを開始します。

```
STRJRNPF FILE(library/file)
         JRN(JRNLIB/DJRN1)
         OMTJRNE(*OPNCLO)
         IMAGES(*BOTH)
```

ステップ 2 で作成したジャーナルの名前を指定します。キャプチャー・プログラムには、IMAGES パラメーターの *BOTH という値が必要です。

4. ソース表のジャーナリングのセットアップを以下のように変更します。
   a. IMAGES(*BOTH) を使用して、ソース表に対するジャーナリングが確実に変更前イメージと変更後イメージの両方について行われるようにする。
   b. そのジャーナルに属性 MNGRCV(*SYSTEM) および DLTRCV(*YES) が指定されていることを確認する。
   c. そのジャーナルに MINENTDTA(*NONE) 属性が指定されていることを確認する。
   d. リモート・システムのジャーナルの場合、ソース・ジャーナルに MNGRCV(*SYSTEM)、DLTRCV(*YES)、および MINENTDTA(*NONE) 属性を指定する。リモート・ジャーナルを定義するには、ADDRMTJRN コマンドに DLTRCV(*YES) 属性を指定する。

ジャーナルおよびジャーナル・レシーバーの管理 (System i)

キャプチャー・プログラムは、ジャーナル項目の受信 (RCVJRNE) コマンドを使って、ジャーナルを受信します。

以下のトピックでは、ジャーナルおよびジャーナル・レシーバーを管理する方法が説明されています。

ジャーナル・レシーバーのシステム管理の指定 (System i):

System i システムによってジャーナル・レシーバーの変更が管理されるようにする必要があります。これは、システム変更ジャーナル管理と呼ばれています。

制約事項

RTVJRNE コマンドを使用してジャーナル項目を検索する場合、同一のジャーナルおよびキャプチャー・スキーマを使用できるのは、最大 299 個のソース物理ファイルまでです。同じジャーナルに 299 個以上のファイルを登録することが必要な場合は、ソースの登録を複数のキャプチャー・スキーマに分割してください。

手順

ジャーナル・レシーバーのシステム管理を指定するには、ジャーナルを作成するときに MNGRCV(*SYSTEM) を指定するか、ジャーナルをその値に変更します。 システム変更ジャーナル管理を使用する場合、システムに変更させるジャーナル・レシーバーのしきい値を指定するジャーナル・レシーバーを作成する必要があります。このしきい値は、最低でも 5,000 KB にし、システムでのトランザクションの数に基づいて決定する必要があります。システムは、レシーバーがしきい値のサイズに達して、新しいジャーナル・レシーバーが作成およびアタッチされると (可能な場合)、そのレシーバーを自動的に切り離します。

実行管理機能オブジェクトの定義の変更 (System i):

DB2 DataPropagator for System i のインストール時に作成される 3 つのタイプの実行管理機能オブジェクトについては、そのデフォルトの定義を変更することも、独自の定義を用意することもできます。

このタスクについて

このライブラリー用の SQL ジャーナルと SQL ジャーナル・レシーバー、および実行管理機能オブジェクトがインストール・プログラムによって作成されます。

表 1に、作成されるオブジェクトをリストします。

表 1. 実行管理機能オブジェクト

説明	オブジェクト・タイプ	名前
サブシステム記述	*SBSD	QDP4/QZSNDPR
ジョブ・キュー	*JOBQ	QDP4/QZSNDPR
ジョブ記述	*JOBD	QDP4/QZSNDPR

独自のサブシステム記述を作成する場合は、サブシステムの名前として QZSNDPR を指定して、そのサブシステムを QDP4 以外のライブラリーに作成する必要があります。これらの定義の変更についての詳細は、「*System i Work Management Guide*」(SC41-5306) を参照してください。

ジャーナル・レシーバーのユーザー管理の指定 (System i):

ジャーナルの作成時に MNGRCV(*USER) を指定すると (独自にジャーナル・レシーバーの変更管理を行いたい場合)、ジャーナル・レシーバーがストレージのしきい値 (レシーバーに指定した場合) に達すると、ジャーナルのメッセージ・キューにメッセージが送信されます。

このタスクについて

CHGJRN コマンドを使用して、古いジャーナル・レシーバーを切り離し、新しいレシーバーをアタッチします。このコマンドを使用すると、ジャーナルされない項目のエラー状態を避け、ジャーナルが使用するストレージ・スペースの量を制限することができます。パフォーマンスに影響が出ないようにするため、システムが最大使用回数に達していないときにこのことを行ってください。

CHGJRN MNGRCV(*SYSTEM) を指定すると、ジャーナル・レシーバーの管理をシステムが行うよう切り替えることができます。

次の 2 つの理由で、現行のジャーナル・レシーバーを定期的に切り離し、新しいレシーバーをアタッチする必要があります。

- 各ジャーナル・レシーバーに特定の管理可能な期間が含まれていると、ジャーナル項目のアナライズがより簡単になるため。
- 大きなジャーナル・レシーバーは、システム・パフォーマンスに影響を及ぼし、補助記憶域の貴重なスペースをふさいでしまう場合があるから。

ジャーナルのデフォルトのメッセージ・キューは QSYSOPR です。 QSYSOPR メッセージ・キューに大きなボリュームのメッセージがあると、別のメッセージ・キュー (DPRUSRMSG など) とジャーナルを関連付けることができます。メッセージ処理プログラムを使って、DPRUSRMSG メッセージ・キューをモニターすることができます。ジャーナル・メッセージ・キューに送信できるメッセージに関する説明は、「*System i Backup and Recovery*」を参照してください。

ジャーナル・レシーバー削除出口ルーチン (System i):

ジャーナル・レシーバー*削除*出口ルーチン (DLTJRNRCV) を使用すると、ジャーナル・レシーバーに含まれているすべての項目をキャプチャー・プログラムがまだ処理していない場合に、ジャーナル・レシーバーを削除することがないように制御できます。

DB2 DataPropagator for System i をインストールした場合、この出口ルーチンは自動的に登録されます。この出口ルーチンは、ジャーナル・レシーバーの削除時に、そのレシーバーがソース表のジャーナリングに使用されているかどうかにかかわらず、常に呼び出されます。この出口ルーチンは、ジャーナル・レシーバーを削除できるかどうかを決定します。

ジャーナル・レシーバー削除出口ルーチンを使用して、ジャーナル管理をシステムに任せるには、CHGJRN または CRTJRN コマンドに対して DLTRCV(*YES) および MNGRCV(*SYSTEM) を指定します。

重要: ジャーナル・レシーバー削除出口ルーチン用の登録を除去する場合は、ソース表用に使用されるすべてのジャーナルを変更して、それらが DLTRCV(*NO) 属性を持つようにする必要があります。

レシーバーが関連付けられているジャーナルがどのソース表にも関連付けられていない場合、この出口ルーチンはそのレシーバーの削除を*承認* します。

ジャーナル・レシーバーが 1 つかそれ以上のソース表で使われている場合、この出口ルーチンは、キャプチャー・プログラムによって処理されなかった項目が、削除

されているレシーバーに含まれていないことを確認します。キャプチャー・プログラムがそのレシーバーの項目をさらに処理する必要がある場合には、出口ルーチンはレシーバーの削除を承認しません。

ジャーナル・レシーバーを削除する必要があるのに、ジャーナル・レシーバー出口ルーチンが削除を承認しない場合には、DLTJRNRCV DLTOPT(*IGNEXITPGM) を指定して、その出口ルーチンをオーバーライドします。

第 4 章 表およびビューを SQL レプリケーション・ソースとして登録する

SQL レプリケーションの場合、レプリケーション・ソースとして使用する表とビューは、それらを登録することによって識別されます。

レプリケーション用として特定の表またはビューを登録する際には、使用可能なデータのソースを作成します。これは、後でさまざまな目的に合わせて別々のターゲットに対して使用できるものです。このセクションで説明されている管理タスクは、レプリケーション・ゴールに基づいてデータが各ソースからどのようにキャプチャーされるのかを定義するコントロール情報をセットアップするのに役立ちます。

ソースの登録時には、レプリケーション・ソースとして使用する表またはビュー、表内でレプリケーションのために使用可能にする列、および SQL レプリケーションがソースからデータと変更をキャプチャーする方法に関するプロパティーを示してください。

SQL レプリケーションでは、以下のオブジェクトをソースとして登録できます。

- DB2 表
- 非 DB2 リレーショナル表 (ニックネームを使用)
- 表内のデータのサブセット (DB2 または非 DB2 リレーショナル)
- 単一表のビュー (DB2)
- 複数の表の内部結合を表すビュー (DB2)

このセクションには、以下のトピックが含まれています。

DB2 表をソースとして登録する

DB2 表をレプリケーション・ソースとして登録する場合は、ソース・サーバー、ソース表の名前、キャプチャー・スキーマを指定します。CD (変更データ) 表が作成されます。

始める前に

- System i の場合を除くすべての DB2 ソースについては、ソース表 DDL は DATA CAPTURE CHANGES オプションを必要とします。このオプションをソースから除去しないでください。
- ソースとして登録する表を処理させるキャプチャー・コントロール・サーバー上に、キャプチャー・コントロール表がすでに存在していなければなりません。

制約事項

System i

- SQL ステートメントは長さが 32,000 文字までに制限されているため、1 つの表当たりおよそ 2000 列までしか登録できない。正確な列数は、列名の長さによって決まります。
- 単一キャプチャー・スキーマに対し、同じジャーナルを使用するソース表を 300 個より多く登録してはならない。
- ソース表、CD 表およびソース表のジャーナルは、すべてこれらのソース表の登録情報を含むキャプチャー・コントロール表と同じ補助記憶域プール (ASP) になければなりません。

Linux UNIX Windows

- 範囲でパーティション化された (CREATE TABLE ステートメントの PARTITION BY 文節の使用による) ソース表からのレプリケーションはサポートされていません。
- 表スペースが、複数パーティションのデータベース・パーティション・グループに置かれた場合は、レプリケーションはサポートされません。

このタスクについて

SQL レプリケーションでは、次のタイプの DB2 表がソースとしてサポートされています。

z/OS

- ユーザー・アプリケーションが保守する DB2 表
- カタログ表
- 外部 CCD 表

System i

- ユーザー・アプリケーションが保守する DB2 表 (ローカルまたはリモート側でジャーナリングされる)
- 外部 CCD 表

Linux UNIX Windows

- ユーザー・アプリケーションが保守する DB2 表
- カタログ表 (フル・リフレッシュのみのレプリケーション)
- マテリアライズ照会表
- 外部 CCD 表
- CREATE TABLE ステートメントの DISTRIBUTE BY 文節でパーティション化された表

別々のキャプチャー・スキーマを使用して、同じ表を複数回登録できます。

手順

DB2 表を登録するには、次の方法のいずれかを使用します。

方法	説明

ASNCLP コマンド行プログラム	CREATE REGISTRATION コマンドを使用して、ソース表、ソース・ビュー、ソース・ニックネームを登録します。例えば、以下のコマンドは、環境を設定し、STAFF 表を DB2 SAMPLE データベースに登録します。
	`SET SERVER CAPTURE TO DB SAMPLE;`
	`SET OUTPUT CAPTURE SCRIPT "register.sql";`
	`SET LOG "register.err";`
	`SET RUN SCRIPT LATER;`
	`CREATE REGISTRATION (DB2ADMIN.STAFF)`
	`DIFFERENTIALREFRESH STAGE CDSTAFF;`

レプリケーション・センター	「表の登録」ウィンドウを使用します。オブジェクト・ツリーで、選択したキャプチャー・スキーマを展開し、「**登録済み表**」フォルダーを右クリックし、「**表の登録**」をクリックします。. **ヒント:** 登録の際の時間を節約するために、キャプチャー・コントロール・サーバーに対してあらかじめソース・オブジェクト・プロファイルをセットアップしておけます。そのようにした場合は、表の登録時に、レプリケーション・センターはレプリケーション・センターのデフォルトではなく、そのソース・オブジェクト・プロファイルに定義されたデフォルトを使用します。これによって登録の際の時間を節約できます。なぜなら、それぞれの表を一度に 1 つずつ選択してデフォルト設定を手動で変更する代わりに、デフォルトを一度に上書きできるからです。

System i
ADDDPRREG システム・コマンド
	ADDDPRREG コマンドを使用して、System i 上で表を登録します。
	例えば、BSN キャプチャー・スキーマの下に、HR ライブラリーから EMPLOYEE という名前のソース表を登録し、HRCDLIB ライブラリーの下に CDEMPLOYEE という名前の CD 表を作成するには、以下のようにします。
	`ADDDPRREG SRCTBL(HR/EMPLOYEE) CAPCTLLIB(BSN)`
	`CDLIB(HRCDLIB) CDNAME(CDEMPLOYEE)`

ソースとして表を登録すると、登録された表に関連付けられているキャプチャー・プログラムが、そのソース用のログを読み取って、登録された列に対して発生した未完了の変更を、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までメモリーに保管します。ロールバックの場合は、それらの変更はメモリーから削除されます。コミットの場合は、それらの変更はキャプチャー・プログラムがコミット・ログ・レコードを読み取ると同時に CD 表に挿入されます。これらの変更は、キャプチャー・プログラムが各キャプチャー・サイクルの後でコミットするまでメモリー内に残されます。キャプチャー・プログラムは、CAPSTART シグナルがユーザーまたはアプライ・プログラムによって発行されるまで、DB2 ソース表のデータのキャプチャーを開始しません。

非リレーショナル・ソース表の場合: IMS™ などの非リレーショナル・データベース管理システムからのデータを含む DB2 表を登録できます。これを行うには、IMS DataPropagator や Data Refresher などといった、非リレーショナル・データベースからのデータを含む CCD 表を移植するためのアプリケーションが必要です。このアプリケーションは、IMS データベース内の非リレーショナル・セグメントへの変

更をキャプチャーし、CCD 表を移植します。この CCD 表はコンプリートでなければなりませんが、コンデンスされていてもコンデンスされていなくても構いません。他の CCD ソースと同様に、CCD ソース表には関連付けられているキャプチャー・プログラムがありません。なぜなら、この表には非リレーショナル・ソース表からの変更済みデータがすでに保管されているからです。 IMS DataPropagator および Data Refresher 製品は、IBMSNAP_REGISTER 表内の値を保守して、アプライ・プログラムがこのソース表からの読み取りを正しく行えるようにします。

非 DB2 リレーショナル表をソースとして登録する

非 DB2 リレーショナル表の登録時には、登録するソース表のニックネームを指定します。 CCD (整合変更データ) 表が作成されます。

始める前に

このソースを処理させるキャプチャー・コントロール・サーバー上に、キャプチャー・コントロール表がすでに存在していなければなりません。

制約事項

- 複数の非 DB2 リレーショナル・ソース・サーバーにアクセスするのに単一のフェデレーテッド・データベースを使用している場合は、その単一フェデレーテッド・データベースで、それぞれの非 DB2 リレーショナル・ソース・サーバーごとに異なるキャプチャー・スキーマを使用する必要があります。2 つの同一のスキーマを使用することはできません。各非 DB2 リレーショナル表は、それぞれ 1 つのキャプチャー・スキーマの下にしか登録できません。
- 非 DB2 リレーショナル表の LOB 列は登録できません。このデータ・タイプを含む表を登録する場合は、列のサブセットを登録する必要があります。

このタスクについて

デフォルトでは、CCD 所有者はソース表のスキーマ名から導き出されます。 CCD 所有者をスキーマ名と一致しないように変更する場合は、ソース表の所有者が CCD 表への書き込みを許可されていることを確認してください。ソース表の所有者が CCD 表を更新できない場合は、ソース表のトリガーは変更を CCD 表に書き込めません。

手順

非 DB2 リレーショナル表を登録するには、次の方法のいずれかを使用します。

方法	説明

ASNCLP コマンド行プログラム	CREATE REGISTRATION コマンドを使用して、ソース表、ソース・ビュー、ソース・ニックネームを登録します。例えば、以下のコマンドは、以下の特性で環境を設定し、登録を作成します。
	• Oracle データベースを含んだ IBM 以外のサーバー V9ORA
	• フェデレーテッド・サーバー FEDORADB
	• Oracle データベース内の CCD 表 undjr09.ccdtest
	• フェデレーテッド・サーバー内の CCD ニックネーム repldba.ccdtestnk
	• 登録するソース・ニックネーム repldba.tesnk
	• repldba.tesnk のすべての列を変更後イメージで登録
	`SET SERVER CAPTURE TO DB FEDORADB NONIBM SERVER V9ORA`
	`ID repldba PASSWORD "passw0rd";`
	`SET OUTPUT CAPTURE SCRIPT "ora_reg.sql";`
	`SET CAPTURE SCHEMA SOURCE ASNORA;`
	`SET LOG "orareg.out";`
	`CREATE REGISTRATION (repldba.testnk)`
	`DIFFERENTIALREFRESH STAGE repldba.ccdtestnk`
	`CONDENSED OFF NONIBM undjr09.ccdtest`
	`COLS ALL IMAGE AFTER;`
	
	フェデレーテッド・ソースの場合は、CONDENSED OFF オプションが必要です。

レプリケーション・センター	「ニックネームの登録」ウィンドウを使用します。オブジェクト・ツリーから、登録するニックネームを含んだ非 DB2 リレーショナル・データベースを展開します。「**登録済みニックネーム**」フォルダーを右クリックして、「**ニックネームの登録**」を選択します。 **ヒント:** 登録の際の時間を節約するために、キャプチャー・コントロール・サーバーに対してあらかじめソース・オブジェクト・プロファイルをセットアップしておけます。そのようにした場合は、表の登録時に、レプリケーション・センターはレプリケーション・センターのデフォルトではなく、CCD 表用のソース・オブジェクト・プロファイルに定義されたデフォルトと CCD 表のニックネームを使用します。これによって登録の際の時間を節約できます。なぜなら、それぞれの表を一度に 1 つずつ選択してデフォルト設定を手動で変更する代わりに、デフォルトを一度に上書きできるからです。

登録済みの非 DB2 リレーショナル表に対する変更が発生すると、キャプチャー・トリガーがキャプチャー・プログラムをシミュレートして、その変更を CCD 表に挿入します。キャプチャー・トリガーは、非 DB2 リレーショナル・ソース表への変更のキャプチャーを、そのソースが登録された時点から開始します。

ソース表用の登録オプション

SQL レプリケーションには、レプリケーション・ソースとして表を登録するときのために多くのオプションが用意されています。これらのオプションは、表の登録に関するタスクのうち、より大きな方のタスクの一部です。

登録する表の選択を済ませたら、レプリケーションのためにどの列を使用可能にするのかを示して、このソースから登録されたデータの処理方法と保管方法を決定するプロパティーを定義できます。また、キャプチャー・プログラムによる CD 表へのソース・データの保管方法 (またはキャプチャー・トリガーによる CCD 表へのデータの保管方法) などといった、他の登録オプションも指定できます。

以下のトピックでは、表を登録するためのオプションを詳しく取り上げています。

列のサブセットの登録 (垂直方向のサブセット化)

レプリケーションのために、ソース表の列のサブセットを登録できます。例えば、ターゲットからサブスクライブできる列のすべてを対象とする訳ではない場合や、ソース表に定義されているデータ・タイプのすべてがターゲット表でサポートされている訳ではない場合などが考えられます。

デフォルトでは、すべての列が登録されます。列のサブセットを登録するには、ターゲット表へのレプリケーションのために使用可能にする列だけを選択します。

CD および CCD 表には一部のタイプのターゲット表にとって十分なキー・データ (ポイント・イン・タイムなど) が含まれていなければならないため、使用するサブセットにターゲットに対してキー列 (主キーまたはユニーク索引) として機能する列が含まれていることを確認してください。

ヒント: ソースの列のサブセットを登録するのは、未登録の列を絶対に複製しない場合だけにしてください。後になって登録しなかった列を複製する必要が生じた場合は、登録を変更して未登録の列を追加しなければなりません (非 DB2 リレーショナル・ソースの場合は、1 つの登録に新しい列を追加するのにすべての登録をまとめて再定義する必要があります)。このソースに関連した内部 CCD を持つ予定である場合は、後で列を追加するのはさらに困難になる可能性があります。なぜなら、新しい列を登録すると、それらは内部 CCD にではなく、CD 表に追加されるからです。これらの問題を回避するためには、すべての列を登録し、アプライ・プログラムを使用して、ターゲットに複製する列をサブセット化するという方法もあります。

変更キャプチャーによるレプリケーションとフル・リフレッシュによるコピー

デフォルトでは、最後のレプリケーション・サイクル以降にソース表で発生した変更だけが複製されます (変更キャプチャーによるレプリケーション)。一方、毎回のサイクルでソース表にあるすべてのデータを複製することも可能です (フル・リフレッシュのみのレプリケーション)。

変更キャプチャー・レプリケーション

変更キャプチャー・レプリケーション時には、変更されたデータだけがターゲット表に複製されます。このソース用に選択したターゲット表のタイプによっては、この表の初期ロードを実行する必要があります。ほとんどの場合、アプライ・プログラムは初期フル・リフレッシュを実行してから、継続して変更キャプチャー・レプリケーションを実行します。

ターゲット表に対するフル・リフレッシュを許可しないことを選択すると、ソース表とターゲット表の再同期化が必要になった場合にターゲット表を手動で再ロードする必要があります。ターゲットが初期ソース・データを含んでいる状態でロードされた後、キャプチャー・プログラムはソースで発生した変更をキャプチャー CD 表に保管します。非 DB2 リレーショナル・ソースの場合の変更キャプチャー・レプリケーションでは、ソースで発生した変更をキャプチャー・トリガーがキャプチャー CCD 表に保管します。アプライ・プログラムは、CD 表または CCD 表から変更を読み取って、その登録されているソースにサブスクライブしているターゲットにそれらの変更を適用します。

DB2 ソース表を変更キャプチャー・レプリケーション用として定義したときに、ソースで発生した変更すべてを CD 表に保管しない方がよい場合があります。行 (水平方向) サブセットを登録して変更をフィルター操作し、実際にソースで発生している変更より少ない変更が CD 表にキャプチャーされるようにできます。以下の 2 つの行キャプチャー規則のいずれかを選択して、キャプチャー・プログラムがソース表からのどの変更済み行を CD 表に記録するのかを決定できます。

- すべての行に対する変更をキャプチャーする
- 変更が登録済みの列に発生した場合にのみ変更をキャプチャーする (DB2 のみ)

デフォルトでは、ソース表で行のいずれかの列 (登録済みでも未登録でも) が更新されるたびに、必ず変更がキャプチャーされます。列のサブセットだけを登録した場合は、ソース表に対して変更が発生するたびに、キャプチャー・プログラムは登録済み列の値を行単位で CD 表に記録します。これは、変更された列が登録済みの列とは異なる列であった場合でも実行されます。このデフォルト・オプションは、ソース表へのすべての変更の履歴を保持する場合に使用してください。非 DB2 リレーショナル・ソースの場合に使用可能なオプションはこれだけです。キャプチャー・トリガーは、変更が未登録の列に対して発生した場合であっても、ソースで変更のあったすべての行をキャプチャーします。

例: 表には 100 個の列があり、それらの列うちの 50 個をレプリケーション用に登録しているとします。デフォルトでは、表内の 100 個の列のどれか 1 つにでも変更が加えられると、いつでもキャプチャー・プログラムによって行が CD 表に書き込まれます (あるいは、キャプチャー・トリガーによって行が CCD 表に書き込まれます)。

DB2 ソースがある場合は、登録済みの列への変更だけをキャプチャー・プログラムにキャプチャーさせることがあります。このケースでは、キャプチャー・プログラムは登録済みの列に対する変更が発生したときだけ行を CD 表に書き込みます。

ヒント: 監査を目的とする情報が必要な場合、または表ではほとんどいつも登録済みの列に対してのみ変更が発生している場合は、すべての行への変更のキャプチャーを選択してください。未登録の列にのみ影響する変更が頻繁に発生している場合は、登録済みの列だけに対する変更のキャプチャーを選択してください。このオプションは、ソース表へのすべての変更の履歴を保持するのは望ましくない場合に使用してください。

フル・リフレッシュのみのレプリケーション

ターゲットがフル・リフレッシュのみのレプリケーション用に登録されているソースに対してサブスクライブしている場合、アプライ・プログラムは毎回のレプリケ

ーション・サイクルで、ターゲット表からすべてのデータを削除し、ソースの登録済み列に含まれているデータをコピーして、ソース・データをターゲットに移植します。キャプチャー・プログラムは関与せず、CD 表もありません。アプライ・プログラムは直接ソース表からデータを読み取ります。

小さな表
コピーするのに時間もリソースもあまり要しない非常に小さなソース表を使用している場合は、フル・リフレッシュのみのレプリケーションを選択するとよいでしょう。

大きな表
表が比較的大きく、かつフル・リフレッシュのみのレプリケーションを使用する場合は、表をより迅速にロードするために ASNLOAD 出口ルーチンを使用できます。

制約事項: このソースにサブスクライブするコンデンスされたターゲット表を準備する予定であるが、そのターゲット表のユニーク索引が用意できない場合は、ソースをフル・リフレッシュのみのレプリケーション用として登録する必要があります。

変更後イメージ列と変更前イメージ列

変更キャプチャーによるレプリケーションのソースを登録すると、デフォルトでは、列の変更後の (変更後イメージの) 値だけがキャプチャーされます。さらに、以前の (変更前イメージの) 値もキャプチャーするように選択することも可能です。

z/OS Linux UNIX Windows

表の中の個々の列について、変更前イメージ値のキャプチャーを行うか行わないかを選択できます。

System i

変更前イメージのキャプチャーを表内のすべての列について行うか、あるいはどの列に対しても行わないかのどちらかを選択できます。個々の列についてこのオプションを選択することはできません。

Sybase または Microsoft® SQL Server
1 つの表に、タイプが TIMESTAMP の列を 1 つしか組み込めません。 データ・ソースが Sybase または Microsoft SQL Server であり、かつソース表にタイプが TIMESTAMP の列があるときは、この列をレプリケーション・ソースの一部として定義する場合、この列については変更後イメージのみを選択してください。

制約事項: LOB データ・タイプの列については、CD 表の変更前イメージ値を組み込むことはできません。

以下の各セクションでは、どのような場合に各オプションを選択する必要があるのかについて説明します。

変更後イメージ値のみのキャプチャー

変更キャプチャー・レプリケーション用に登録するそれぞれの列ごとに、変更が行われるたびにキャプチャー・プログラムまたはキャプチャー・トリガーに対して、変更後イメージ値だけを記録させることを選択することができます。変更後イメー

ジ値だけを取り組むことを選択した場合は、CD (または CCD) 表にはそれぞれの変更された値ごとに 1 つの列が含められ、その列には変更発生後のソース列の値が保管されます。

このソースに対しては基礎集約ターゲット表タイプおよび変更集約ターゲット表タイプだけを使用する予定である場合は、変更前イメージは必要ありません。ターゲット表を計算済みの値のために使用する予定である場合は、変更前イメージ列は無意味です。なぜなら、計算された列には変更前イメージがないからです。他のすべてのターゲット表タイプでは、変更前イメージ列を利用することができます。

変更前イメージ値と変更後イメージ値のキャプチャー

変更キャプチャー・レプリケーション用に登録するそれぞれの列ごとに、変更が行われるたびにキャプチャー・プログラムまたはキャプチャー・トリガーに対して、変更前イメージ値と変更後イメージ値の両方を記録させることを選択することができます。変更前イメージ値と変更後イメージ値をキャプチャーすることを選択した場合は、CD (または CCD) 表にはそれぞれの変更された値ごとに 2 つの列が含められます。一方の列は変更が発生する前にソース列に入っていた値用で、もう一方の列は変更発生後の値用です。

変更前イメージと変更後イメージの両方を CD (または CCD) 表に保管することを選択した場合は、以下に示すように、変更前イメージ列と変更後イメージ列には、ソース表に対して実行されたそれぞれ異なるアクションの種類に応じて異なる値が含められます。

挿入 変更前イメージ列には NULL 値が含まれます。変更後イメージ列には挿入された値が含まれます。

更新 変更前イメージ列には、変更が発生する前の列値が含まれます。変更後イメージ列には、変更が発生した後の列値が含まれます。

更新を削除と挿入の対としてキャプチャーすることを選択した場合は、削除行ではその行の変更前イメージ列と変更後イメージ列の両方に更新の変更前イメージが含められ、挿入行では変更前イメージ列には NULL 値が、変更後イメージ列には変更後イメージが含められます。

削除 変更前イメージ列と変更後イメージ列に、変更が発生する前の列値が含められます。

z/OS DB2 for z/OS の表であれば 18 文字の列名を使用できますが、レプリケーションは、18 番目の文字をターゲット表では変更前イメージ列 ID に置き換えるため、列名の最初の 17 文字が必ず固有になるようにする必要があります。

Linux UNIX Windows System i 定義によって変更前イメージが含まれている列の場合、DB2 レプリケーションでは列名が 29 文字までに制限されます。なぜなら、列名全体で許容される文字数が 30 文字だけだからです。列名がそれより長いと、左方から切り捨てるようにプロファイルを設定していないかぎり、デフォルトでは余分な文字がレプリケーションによって右方から切り捨てられます。レプリケーションによってターゲット列に変更前イメージ列 ID (通常は *X*) が追加される上に、各列名は必ずユニークでなければならないため、29 文字より長い列名は使用

できません。複製するつもりがない表にはさらに長い列名を使用できますが、その列を将来複製する可能性がある場合は 29 文字の名前を使用することを検討してください。

以下のリストでは、変更前イメージ値のキャプチャーが必要となる可能性のあるケースについて説明しています。

ソース・データの履歴を保持する場合
監査目的でデータを保持する必要がある場合は、ある期間内にデータがどのように変更されたかについてのレコードを持てるように、変更前イメージと変更後イメージの両方を選択します。監査やアプリケーション・ロールバック機能を必要とする業界では、変更前イメージと変更後イメージのコピーのセットが役立ちます。

競合検出を使用する Update-anywhere 構成の場合
レプリカ表 (競合検出が None 以外に設定されているもの) の間で競合が起こり得る Update-anywhere 構成では、レプリカの CD 表に変更後イメージ列と変更前イメージ列の両方を登録して、競合が発生した場合には変更をロールバックできるようにする必要があります。

ターゲットでキー列が更新の対象となっている場合
ソースを登録する際には、この表をソースとして使用して定義する可能性のあるターゲット表があるかどうかを検討してください。通常ターゲット表はコンデンスされており、そのターゲット表内の各行をユニークにする列または列のセットを必要とします。それらのユニーク列によっていわゆるターゲット・キーが構成されます。これらのターゲット・キー列のいずれかがソースで更新される可能性がある場合は、SQL レプリケーションではターゲット表で必ず正しい行が更新されるようにするための特殊な処理が必要となります。SQL レプリケーションがターゲット表で必ず正しい行を新しいキー値で更新するようにするために、ターゲット・キーを構成する列については変更後イメージと変更前イメージの両方をキャプチャーすることを選択できます。これらの登録済み列の変更前イメージ値は、アプライ・プログラムが非キー・ソース列の変更をターゲット表のターゲット・キー列に適用するときに必要となります。変更を適用するときには、アプライ・プログラムはこの行をターゲット表でソースの CD (または CCD) 表内の変更前イメージ値と一致するターゲット・キー値を探すという方法で検索し、次にそのターゲット行をソースの CD (または CCD) 表内の変更後イメージ値で更新します。

これらの変更前イメージ値をソース表またはビューの登録時に登録しても、レプリケーションはユーザー・アプリケーションがターゲット・キーに対して更新を行うことを知りません。後で (サブスクリプション・セットを作成して) どのターゲットをこのソースにサブスクライブするのかを定義するときに、アプライ・プログラムに対し、変更をソースの非キー列からターゲットのキー列に適用する際に特殊な更新を実行するよう指定できます。

変更前イメージ接頭部

変更後イメージ列と変更前イメージ列をキャプチャーする場合、変更後イメージ列はソース表の列の名前になり、変更前イメージ列はソース表の列の名前に 1 文字の接頭部を付けた名前になります。

ASNCLP コマンド行プログラムとレプリケーション・センターによって割り当てられるデフォルトの変更前イメージ接頭部は X です。System i コマンドの場合のデフォルトは @ です。

デフォルトの接頭部は、変更が可能です。変更前イメージ接頭部と CD (または CCD) 列名の組み合わせは、CD (または CCD) 表内の現行の列名または潜在的な列名と同じであってはなりません。

例: 変更前イメージ接頭部として X を使用していて、さらに COL という名前のソース列を登録している場合は、XCOL という名前の列は登録できません。なぜなら、XCOL が別のソース列の実際の列名なのか、それとも COL という列名と変更前イメージ接頭部 X を持つ変更前イメージ列の名前なのかがはっきりしないからです。

制約事項: 変更前イメージ接頭部に、ブランク文字は使用できません。

表の変更前イメージ列を一切複製していない場合は、変更前イメージ接頭部を持たないことを選択して、このプロパティーを NULL に設定できます。

エラー発生時におけるキャプチャー・プログラムの停止

キャプチャー・プログラムは、登録の処理中に特定の問題を検出すると、デフォルトで停止します。プログラムの実行を継続するように選択することも可能です。

以下のリストでは、それぞれの環境に合った最適なオプションを選択するために役立つ詳細情報を提供します。

エラー発生時にキャプチャーを停止

このオプションの場合、キャプチャー・プログラムは、IBMSNAP_CAPTRACE 表にエラー・メッセージを書き込んで終了します。

キャプチャー・プログラムは、以下の致命的エラーが発生したときに停止します。

- CD 表スペースがいっぱいになった。
- SQLCODE-911 エラーが 1 行で 10 回発生した。
- 予期しない SQL エラーが発生した。

キャプチャー・プログラムは、以下のような特定の致命的ではないエラーが発生しても停止しません。

- SQLCODES が無効なデータ長を示す。
- z/OS コンプレッション・ディクショナリーが存在しない。

これらの非致命的エラーが発生すると、キャプチャー・プログラムは登録を無効にし、実行を継続します。

エラー発生時にキャプチャーを停止しない

特定のエラーが発生した場合、キャプチャー・プログラムは実行を継続します。ソース処理の初回の試行中にエラーを検出した場合は、キャプチャー・プログラムは登録を活動化しません。登録されたソースがすでにアクティブ化されていた場合は、登録の処理を停止します。いずれの場合も登録は停止します。停止された登録は、 IBMSNAP_REGISTER コントロール表の STATE 列に「S」(停止された) の値を取ります。

このオプションは、以下の非致命的エラーが発生してもキャプチャー・プログラムを停止しません。

- 登録が正しく定義されていない。
- キャプチャー・プログラムが変更済みデータの行の挿入を試行したときに、CD 表が検出されなかった。
- キャプチャー・プログラムが始動または再初期化されたときに、(非 System i) ソース表の DATA CAPTURE CHANGES オプションがオフにされた状態で検出された。

エラーのためにサブスクリプション・セット・メンバーの登録状態が停止状態にあると、アプライ・プログラムはサブスクリプション・セットを処理できません。

キャプチャー・プログラムが更新を保管する方法に関するオプション

デフォルトでは、ソース表に対する更新は、CD 表の中で 1 つの行に保管されます。ただし、状況によっては、キャプチャー・プログラムまたはキャプチャー・トリガーによって、ソース表への更新を DELETE と INSERT の対としてキャプチャーし、2 行で保管することが必要になります。

ソース・アプリケーションがサブスクリプション・セット・メンバーの述部で参照される列を 1 つ以上更新する場合は、更新を DELETE ステートメントと INSERT ステートメントとしてキャプチャーする必要があります。

例: ソース・データだけにサブスクライブするターゲットを、特定の列値を基にした述部を使用して定義する (例えば、WHERE DEPT = 'J35') 予定であるとします。その列を変更すると (例えば、DEPT='FFK' に変更)、キャプチャーされた変更は、ターゲットに複製する対象として選択されません。なぜなら、その列は述部の基準に合致していないからです。つまり、新しい FFK 部門は、サブスクリプション・セット・メンバーが部門 J35 に基づいているため複製されません。更新を DELETE および INSERT の対に変換すると、そのターゲット表行は確実に削除されます。

キャプチャーされたそれぞれの更新は、CD (または CCD) 表の 2 つの行に変換されます。この変換は、すべての列について実行されます。この取り込んだデータの増加に合わせて CD (または CCD) 表のスペース割り振りの調節が必要になる場合もあります。

変更の再キャプチャーの防止 (Update-anywhere レプリケーション)

Update-anywhere レプリケーションの場合は、再キャプチャー・オプションを使用することによって、あるサイトから複製した変更を 2 番目のサイトで再キャプチャーする (つまり、さらに別のサイトへの複製を可能にする) かどうかを制御できます。

制約事項: 非 DB2 リレーショナル・データベースの表は Update-anywhere に含まれません。このオプションは、DB2 ソース専用です。

Update-anywhere レプリケーションの場合は、変更はマスター表または関連したレプリカ表で発生する可能性があります。 Update-anywhere レプリケーションで使用する予定の表を登録する場合、SQL レプリケーションはその表がユーザーの構成の中でマスター表となることを前提とします。

登録時に、再キャプチャー・オプションをマスター表に対して設定します。その後、そのマスター・ソース表をレプリカ・ターゲットにマップする際に、レプリカの変更を再キャプチャーして他の表に転送するかどうかを設定できます。

Update-anywhere 構成でマスターとして機能するソース表を登録する場合は、以下の 2 つのオプションのうち、どちらかを選択できます。

マスターで変更を再キャプチャーする
: あるレプリカで発生したマスターに対する更新は、マスターで再キャプチャーされて、他のレプリカに転送されます。

マスターで変更を再キャプチャーしない
: あるレプリカで発生してマスターに加えられた更新は、マスターで再キャプチャーされず、他のレプリカに転送されません。

Update-anywhere 構成のレプリカ表を登録する場合は、以下の 2 つのオプションのうち、どちらかを選択できます。

レプリカで変更を再キャプチャーする
: マスターで発生してレプリカに加えられた更新は、そのレプリカで再キャプチャーされて、このレプリカにサブスクライブしている他のレプリカに転送されます。

レプリカで変更を再キャプチャーしない
: マスターで発生してレプリカに加えられた更新は、そのレプリカで再キャプチャーされず、このレプリカにサブスクライブしている他のレプリカに転送されません。

変更の再キャプチャーを防止すると、パフォーマンスを向上させ、さらにストレージ・コストを削減できます。なぜなら、キャプチャー・プログラムが同じ変更をそれぞれのレプリカごとに再キャプチャーしないからです。

以下のトピックでは、Update-anywhere 構成に基づいて変更を再キャプチャーするかどうかを決定する方法について説明しています。

レプリカを 1 つだけ持つマスター

Update-anywhere 構成内にレプリカを 1 つしか持たない予定である場合は、変更がマスター表でもレプリカ表でも再キャプチャーされないように登録を作成します。

マスター表が他のレプリカ表にとってのソースではなく、レプリカが他のレプリカにとってのソースではない場合 (multi-tier 構成において) は、これが最良の設定です。関与するのがこの 2 つの表だけである場合は、レプリカで発生した変更をマスターで再キャプチャーする必要は無く、またマスターで発生したいかなる変更も単一レプリカで再キャプチャーする必要がありません。

マスターの相互に排他的なパーティションである複数のレプリカ

マスターの相互に排他的なパーティションである複数のレプリカの場合は、変更がマスター表でもレプリカ表でも再キャプチャーされないように登録を作成します。

マスター表のパーティションである複数のレプリカを持つ予定である場合は、マスター表および各レプリカ表の両方で変更が再キャプチャーされないようにすることができます。どのレプリカも他のレプリカ表にとってのソースではない場合は、これが最良の設定です。レプリカがマスターのパーティションである場合は、複数のレプリカがマスターの同じデータにサブスクライブすることがあってはなりません。したがって、どのレプリカで発生したどの変更もマスターで再キャプチャーする必要は無く、他のレプリカに転送する必要もありません。なぜなら、そのソース・データにサブスクライブしているのはその変更が発生したレプリカだけだからです。

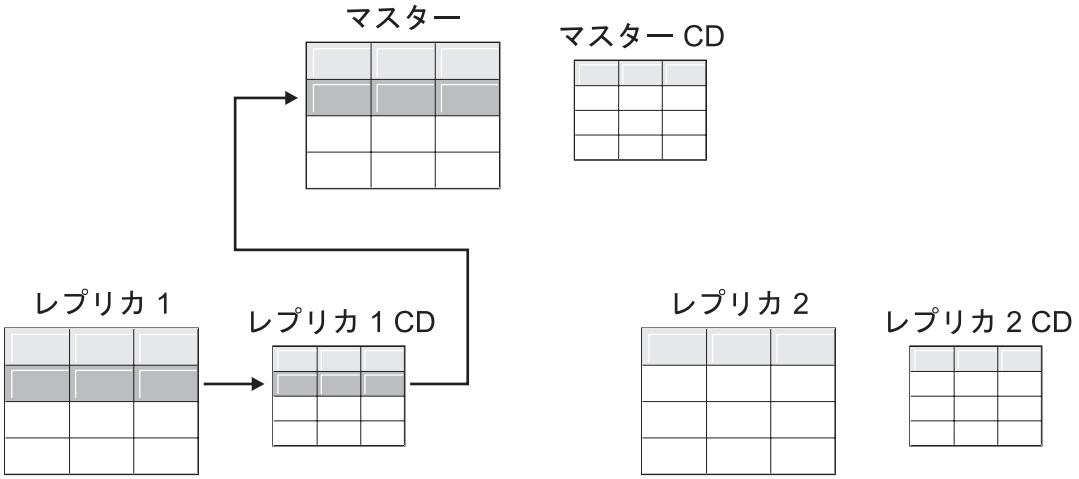


図 1. マスターの相互に排他的なパーティションであるレプリカ用の再キャプチャー・オプション： マスターの同じデータにサブスクライブしていない複数のレプリカを持っている場合は、どの表に対してもこの再キャプチャー・オプションを使用する必要はありません。

変更を複数のレプリカに複製するマスター

変更を複数のレプリカに複製するマスターの場合は、変更がマスター表では再キャプチャーされ、レプリカ表では再キャプチャーされないように登録を作成します。

そのようにすると、レプリカで発生した変更はマスターで再キャプチャーされて、その更新されたマスター・データにサブスクライブしている他のレプリカへと複製されます。

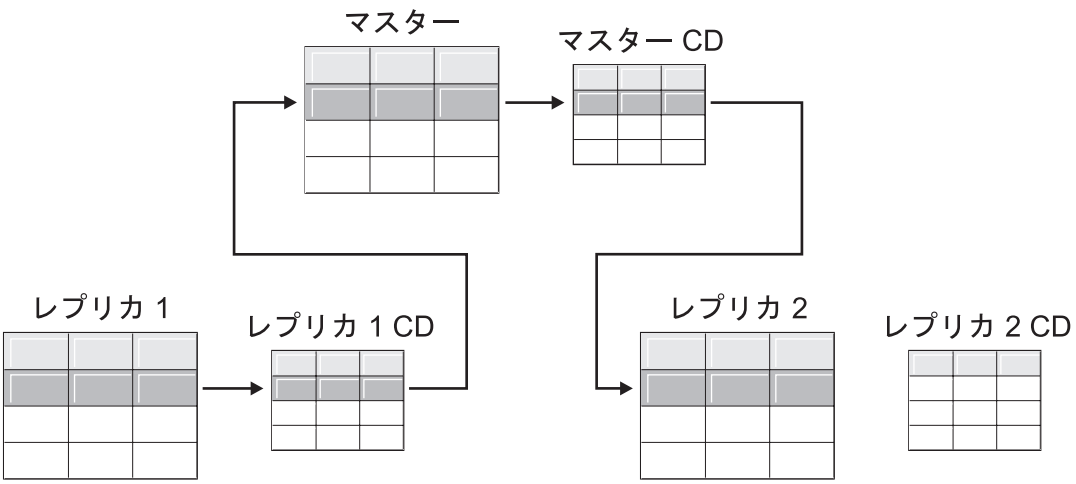


図 2. 変更を複数のレプリカに複製するマスター用の再キャプチャー・オプション： マスターの同じデータにサブスクライブしている複数のレプリカを持っている場合は、マスターで再キャプチャー・オプションを使用して、あるレプリカで発生した変更がマスターで再キャプチャーされて他のレプリカ表に転送されるようにできます。

他のレプリカに変更を複製するレプリカ (複数層)

他のレプリカに変更を複製するレプリカ (複数層) の場合は、変更がマスター表では再キャプチャーされず、レプリカ表では再キャプチャーされるように登録を作成します。

マスター (層 1) があるレプリカ (層 2) に対してソースとして機能し、次にそのレプリカが同様に別のレプリカ (層 3) に対してソースとして機能する multi-tier 構成を持てます。このタイプの構成を持つ予定である場合は、キャプチャー・プログラムに中間のレプリカ (層 2) で変更を再キャプチャーさせて、マスターで発生した変更がその次のレプリカ (層 3) に転送されるようにすることができます。

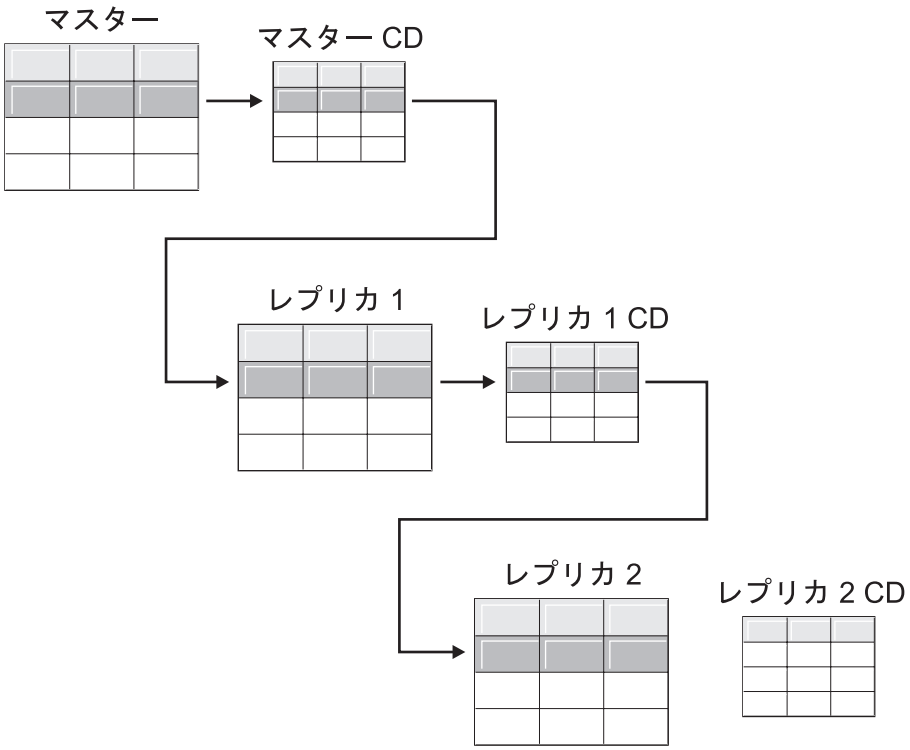


図 3. 層 2 で再キャプチャー・オプションを使用すると、層 1 での変更が層 3 にまで複製されるようにできます : multi-tier 構成で中間層として機能するレプリカ表がある場合は、そのレプリカで再キャプチャー・オプションを使用して、マスターで発生した変更が中間層のレプリカで再キャプチャーされて下層のレプリカに転送されるようにできます。

また、中間のレプリカ (層 2) に再キャプチャーを設定してある場合は、最終のレプリカ (層 3) で発生した変更が中間のレプリカ (層 2) で再キャプチャーされてマスター (層 1) に転送されます。

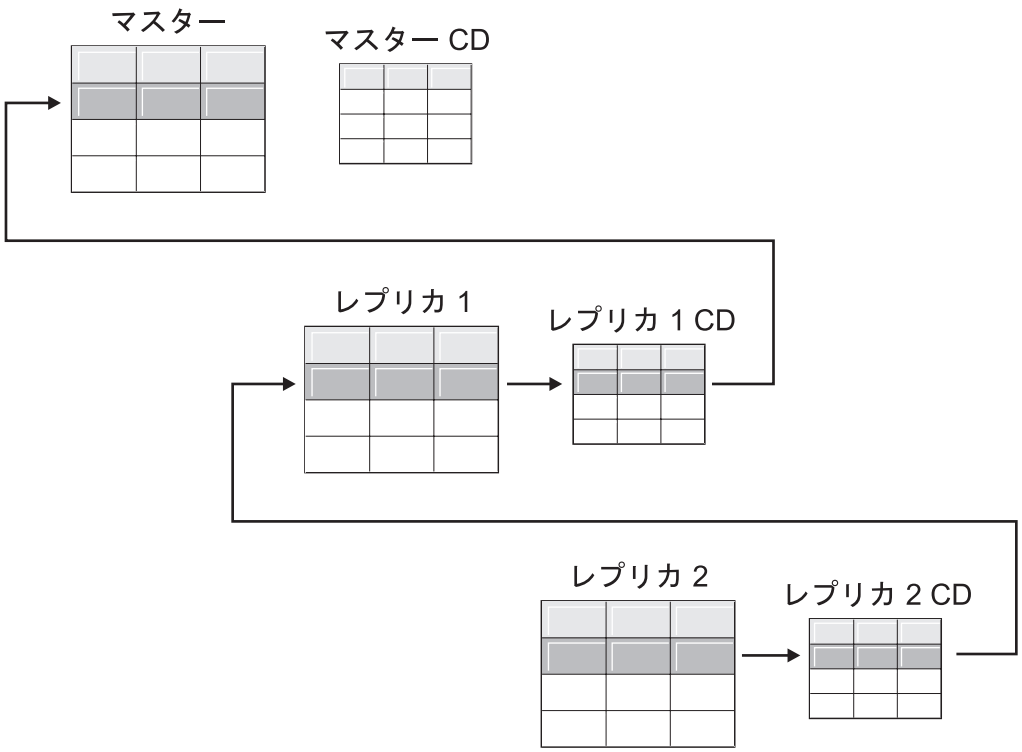


図 4. 層 2 で再キャプチャー・オプションを使用すると、層 3 での変更が層 1 にまで複製されるようにできます : multi-tier 構成で中間層として機能するレプリカ表がある場合は、そのレプリカで再キャプチャー・オプションを使用して、下層のレプリカで発生した変更が中間層のレプリカで再キャプチャーされてマスターに転送されるようにできます。

競合検出のためのオプション (Update-anywhere レプリケーション)

Update-anywhere 構成では、マスターとそのレプリカの間で時々競合が発生する可能性があります。ソースの登録時に、「なし」、「標準」、「拡張」という 3 つのレベルの中から競合検出のレベルを選択できます。

以下に、競合が起こる可能性のある場合を示します。

- マスター表である行が更新されたが、1 つ以上のレプリカ表で同じ行にそれとは異なる更新が加えられ、さらにアプライ・プログラムがそれらの競合する変更を同一サイクルの間に処理した。
- 制約に違反した。

競合検出レベルを個々のレプリケーション・ソースに設定した場合でも、アプライ・プログラムはサブスクリプション・セットのすべてのメンバーのレベルとして、サブスクリプション・セット・メンバーの競合検出レベルのうち最も高いものを使用します。

制約事項:

- 非 DB2 リレーショナル・データベースからの表は Update-anywhere に含まれない。したがって、非 DB2 リレーショナル・ソースでは競合検出は行われません。
- LOB 列を含む Update-anywhere 構成がある場合は、競合検出レベルとして「なし」を指定する必要がある。

逸失またはリジェクトされたトランザクションの許容度とパフォーマンス要件に基づいて、どのタイプの検出を使用するかを決定できます。

なし 競合検出を行いません。マスター表とレプリカ表の間で競合している更新は検出されません。このオプションは、Update-anywhere レプリケーションではお勧めしません。

標準 適度な競合検出。

各アプライ・サイクルの間に、アプライ・プログラムはマスターの CD 表のキー値とレプリカの CD 表のキー値を比較します。両方の CD 表に同じキー値が存在している場合、それは競合です。競合が発生すると、アプライ・プログラムは直前にレプリカでコミットされたトランザクションを取り消します。これは、レプリカの CD 表からの読み取りを実行し、マスターで発生した変更だけを保持するという方法で行われます。

拡張 マスターとそのレプリカとの間での最良のデータ保全性を実現する競合検出。

標準検出の場合と同様に、アプライ・プログラムは各アプライ・サイクルの間にマスターの CD 表のキー値とレプリカの CD 表のキー値を比較します。両方の CD 表に同じキー値が存在している場合、それは競合です。ただし拡張検出では、競合をチェックする前に、アプライ・プログラムはすべての未完了トランザクションがコミットされるまで待ちます。すべての未完了トランザクションを確実に捕らえられるように、アプライ・プログラムはこれから処理されるトランザクションに対してサブスクリプション・セット内のすべてのターゲット表をロックし、 CD 表内のすべての変更がキャプチャーされた後で競合検出を開始します。競合が発生すると、アプライ・プログラムは直前にレプリカでコミットされたトランザクションを取り消します。これは、レプリカの CD からの読み取りを実行し、マスターで発生した変更だけを保持するという方法で行われます。

制約事項: 拡張競合検出を指定しても、アプライ・プログラムが不定期接続環境で実行されている (COPYONCE キーワードを使用して始動された) 場合は、アプライ・プログラムは標準競合検出を使用します。

アプライ・プログラムは、読み取り従属関係を検出することはできません。例えば、後で (DELETE ステートメントによって、またはロールバック・トランザクションによって) 除去されることになる情報をアプリケーションが読み取る場合、アプライ・プログラムは従属関係を検出することはできません。

競合が発生し得るレプリケーション構成を (検出なしまたは標準検出のいずれかを選択して) セットアップする場合は、発生したあらゆる競合を識別して処理するための方法を組み込む必要があります。競合のあったトランザクション更新はレプリケーションのインフラストラクチャーによってすでに検出されてバックアウトされているとしても、アプリケーション・デザイナーは、いったんコミットされたが現在はバックアウト済みのトランザクションをどのように処理するのかを決定する必

要があります。 ASNDONE 出口ルーチンは各サブスクリプション・サイクルの最後で実行されるため、アプリケーション・デザイナーはこのルーチンを上記のようなアプリケーション固有のロジックのためのランチ点として使用できます。バックアウト済みの競合する更新に関する情報は、保持制限整理の対象として適格となるまで CD 表および UOW 表に残されます。

リモート・ジャーナリングを使用する表の登録 (System i)

リモート・ジャーナリングを使用する System i 表を登録する場合は、レプリケーション・ソースとしてローカル・ジャーナルの代わりにリモート・ジャーナルを指定できます。

レプリケーション用のリモート・ジャーナリング・オプションを選択して、CD 表、キャプチャー・プログラム、およびキャプチャー・コントロール表を、ソース表が置かれている System i サーバーとは別個の System i データベース・サーバーに移動してください。

System i で表をソースとして登録する場合、デフォルトではリモート・ジャーナリングを使用しないことが前提とされます。

推奨: ある System i 表から別の System i 表にデータを複製する場合、セットアップ済みのリモート・ジャーナルがあるときは常に、登録時にリモート・ジャーナリング関数を使用することを強く推奨します。レプリケーションにリモート・ジャーナリングを使用すると、パフォーマンスが大幅に向上します。リモート・ジャーナル関数を使用すると、登録、キャプチャー・プログラム、およびキャプチャー・コントロール表をソース表が置かれているシステムから遠く離れたところに移動できるため、そのシステムにはより多くのリソースが使用可能な状態で残されます。これにより、プロセッサー使用量が削減され、ディスク・スペースが節約されます。また、ターゲット・サーバーにあるリモート・ジャーナルを使用する場合は CD 表がターゲット表と同じシステム上に存在するため、アプライ・プログラムは予備ファイルを使用せずに直接 CD 表からターゲット表に変更を適用できます。予備ファイルを使用しないことで、アプライ・プログラムによって使用されるリソースの量が削減されます。

推奨: リモート・ジャーナルを使用する表をソースとして登録するのは、登録がレプリケーション・ターゲットと同じ System i システム上にある場合だけにしてください。SQL レプリケーションを使用すると、登録がターゲットと同じ System i システム上になくてもリモート・ジャーナルをソースとして登録できますが、その場合にはジャーナルをターゲット・システムに置くことで得られるパフォーマンス上の利点を得られません。

リモート・ジャーナリングを使用する System i 表を登録する前に、使用するリモート・ジャーナルがアクティブ状態になっていることを確認してください。

制約事項: リモート・ジャーナル構成ではレプリカ・ターゲット表タイプはサポートされません。

リモート・ジャーナル機能の詳細については、 「*バックアップおよび回復*」 (SD88-5008) および 「*System i Remote Journal Function for High Availability and Data Replication*」 (SG24-5189) を参照してください。

主キーの代わりに相対レコード番号 (RRN) を使用する (System i)

主キー、ユニーク索引、またはユニーク索引として使用できる列の組み合わせを含んでいない System i 表を登録する場合は、その表を相対レコード番号 (RRN) を使用して登録する必要があります。

RRN を使用して複製することを選択すると、CD 表とターゲット表の両方に、それぞれの行ごとにユニークな値を含む INTEGER 型の列 (IBMQSQ_RRN) が追加されます。この列に含まれているのは、ソース表の各行に対応する RRN です。

ソース表が再編成されないかぎり、RRN はソース表の行に対する主キーとして使用されます。ソース表が再編成されると、ソース表の各行の RRN が変更されます。したがって、CD 表およびターゲット表の行に含まれている RRN の値は、当該の行のソース表内での新しい位置を反映した正しい値ではなくなります。

ソース表を (例えば、削除された行を圧縮するために) 再編成すると、必ず DB2 DataPropagator for System i はそのソース表のセットになっているすべてのターゲット表に対してフル・リフレッシュを実行します。この理由から、RRN を主キーとして使用するターゲット表をやはり RRN を使用する他のターゲットと一緒にサブスクリプション・セットに入れて、RRN 以外で一意性を表すなんらかの因子を使用する表とはセットにしないでください。

レプリケーション・ソースとしてのビューの動作

レプリケーションのためにビューを登録すると、ビューは、基礎表の登録オプション (特に、変更キャプチャーのレプリケーションかフル・リフレッシュのレプリケーションかに関するオプション) を継承します。

以下のトピックでは、さまざまなシナリオでの登録済みビューの動作を説明しています。

単一の表に対するビュー

基礎表がレプリケーション用に登録されている場合は、単一表に対するビューを登録できます。ビューは、基礎表からレプリケーションのタイプを継承します。

フル・リフレッシュのみ
基礎表がフル・リフレッシュのみのレプリケーション用に登録されている場合は、ビューはフル・リフレッシュのみのレプリケーションを持ちます。そのビューを変更キャプチャー・レプリケーション用に登録することはできません。なぜなら、その基礎表は変更をトラッキングするための関連した CD 表を持っていないからです。

変更キャプチャー
基礎表が変更キャプチャー・レプリケーション用に登録されている場合は、ビューのレプリケーションは変更キャプチャー・レプリケーションとなるため、フル・リフレッシュのみのレプリケーション用に登録することはできません。

変更キャプチャー・レプリケーション用に登録されている表に対するビューを登録すると、基礎表の CD 表に対するビューが作成されます。この CD ビューには、登録したビューから参照される列だけが含まれています。

ビュー内の列のサブセットは登録できません。ビュー内のすべての列が自動的に登録されます。

複数の表の結合に対するビュー

複数の表の結合に対するビューを登録する場合は、少なくとも、結合に使用する基礎表のいずれか 1 つが登録されている必要があります。また、ソースとして登録されている CCD 表の内部結合も持てます。

結合をレプリケーション・ソースとして登録すると、SQL レプリケーションは同一の SOURCE_OWNER 値と SOURCE_TABLE 値が含まれている複数の行を IBMSNAP_REGISTER 表に追加します。これらの行は、各自の SOURCE_VIEW_QUAL 値によって区別されます。これらの各項目によって結合のコンポーネントが識別されます。

制約事項: CCD 表が組み込まれた結合を定義する場合は、その結合内の他のすべての表が CCD 表である必要があります。

ある結合ビューがレプリケーション・ソースとして存続できるようにするためには、その結合ビューを相関 ID を使用して作成する必要があります (単一表に対するビューの場合には相関 ID は不要です)。

例:

```
create view REGRES1.VW000  (c000,c1001,c2001,c2002,c1003) as
  select a.c000,a.c001,b.c001,b.c002,a.c003
  from REGRES1.SRC001 a, REGRES1.SRC005 b
  where a.c000=b.c000;
```

VW000 はビューの名前です。SRC001 および SRC005 は、ビューの一部である表です。C000、C001、C002、および C003 は、両方の表 (SRC001 と SRC005) の C000 列が等しいという条件の下でビューの一部となる列です。

ビューが継承するレプリケーションのタイプは、そのビューの基礎表の組み合わせによって決まります。各基礎表は次のいずれかです。

- 変更キャプチャー・レプリケーション用に登録されているもの
- フル・リフレッシュのみのレプリケーション用に登録されているもの
- 未登録のもの

表 2 には、基礎表のさまざまな組み合わせと、それぞれの組み合わせからソース・ビューと CD ビューが結果的にどのタイプになるのかが示されています。

表 2. ビューの場合の基礎表の組み合わせ

表 1	表 2	結合ビューおよび CD ビューの記述
変更キャプチャー用に登録されているもの	変更キャプチャー用に登録されているもの	このビューは、変更キャプチャー・レプリケーション用に登録されたものである。この CD ビューには、表 1 の CD 表と表 2 の CD 表から参照される列が含まれている。

表 2. ビューの場合の基礎表の組み合わせ (続き)

表 1	表 2	結合ビューおよび CD ビューの記述
変更キャプチャー用に登録されているもの	フル・リフレッシュのみ用に登録されているもの	このビューは、変更キャプチャー・レプリケーション用に登録されたものである。この CD ビューには、表 1 の CD 表から参照される列と、表 2 から参照される列が含まれている。各レプリケーション・サイクルでは、表 1 に入っている列に対する変更だけが登録済みビューのターゲットに複製される。
フル・リフレッシュのみ用に登録されているもの	フル・リフレッシュのみ用に登録されているもの	このビューは、フル・リフレッシュのみのレプリケーション用に登録されたものである。 CD ビューはない。
フル・リフレッシュのみ用に登録されているもの	未登録のもの	このビューは、フル・リフレッシュのみのレプリケーション用に登録されたものである。 CD ビューはない。
変更キャプチャー用に登録されているもの	未登録のもの	このビューは、変更キャプチャー・レプリケーション用に登録されたものである。この CD ビューには、表 1 の CD 表から参照される列と、表 2 から参照される列が含まれている。各レプリケーション・サイクルでは、表 1 に入っている列に対する変更だけが登録済みビューのターゲットに複製される。
未登録のもの	未登録のもの	このビューは有効なレプリケーション・ソースではないため登録できない。

二重削除の防止

複数のソース表がレプリケーション・ソースとして組み込まれているビューを定義する場合は、二重削除を防ぐための配慮が必要です。二重削除は、同一レプリケーション・サイクルの間に両方とも 1 つのビューの一部である表から行を削除した場合に発生します。例えば、CUSTOMERS 表と CONTRACTS 表を含むビューを作成したとします。二重削除は、同一のレプリケーション・サイクルで CUSTOMERS 表から 1 つの行を削除し、同様に (ビューの結合点から) それに対応する行を CONTRACTS 表からも削除した場合に発生します。ここで問題となるのは、その行が結合の 2 つのソース表から削除されているために、その行はビューに (基本ビューにも CD 表ビューにも) 表示されず、したがってこの二重削除がターゲットに複製できないことです。

二重削除を防ぐためには、結合内のソース表のいずれか 1 つに CCD 表を定義する必要があります。この CCD 表は、コンデンスされた非コンプリートの表でなければならず、さらにターゲット・サーバー上になければなりません。 結合内のいずれかのソース表に対し、コンデンスされた非コンプリートの CCD 表を定義すれば、ほとんどの場合の二重削除問題は解決されます。なぜなら、この CCD 表の IBMSNAP_OPERATION 列を使用すれば削除を検出できるからです。サブスクリプション・サイクルの後に実行するサブスクリプション・セットの定義に、単に SQL ステートメントを追加してください。この SQL ステートメントは、 CCD 表内の IBMSNAP_OPERATION が 『D』 と等しくなるターゲット表からすべての行を除去します。

同一アプライ・サイクルで、CCD を持つソース表のある行が更新されたが、結合内のもう一方の表ではそれに対応する行が削除されたという場合には、更新と削除に関する問題がまだ発生する可能性があります。その結果、アプライ・プログラムは結合された表でその対応する行を検出できず、更新された値を複製できません。

表のビューをソースとして登録する

レプリケーションのソースとしてビューを登録すると、そのビューは、ビューの基礎になっているソース表の登録オプションを継承します。

始める前に

- ソースとして登録するビューを処理させるキャプチャー・コントロール・サーバー上に、キャプチャー・コントロール表がすでに存在していなければならない。
- ソース・ビューの名前は DB2 表の命名規則に従う必要がある。
- ビューの基礎となる基礎表を少なくとも 1 つソースとして登録しなければならない。基礎表を登録するときは、ビューの登録時に使用するのと同じキャプチャー・スキーマを使用します。

制約事項

- 非 DB2 リレーショナル表のビューは登録できない。
- 別のビューの上層のビューは登録できない。
- ビューを定義されているすべての CCD 表は、レプリケーション・ソースとして登録するためにはコンプリートでコンデンスされたものでなければならない。
- System i SQL ステートメントは長さが 32,000 文字までに制限されているため、1 つのビューあたりおよそ 2000 列までしか登録できない。正確な列数は、列名の長さによって決まります。

手順

以下の方法のいずれかを使用して、ビューを登録してください。

方法	説明
ASNCLP コマンド行プログラム	CREATE REGISTRATION コマンドを使用し、*objowner* (オブジェクト所有者) および *objname*(オブジェクト名) としてビュー名を指定します。 ビューの場合は、このコマンドによって、差分リフレッシュとフル・リフレッシュのどちらのためにソースを登録するのかを指定します。
レプリケーション・センター	「ビューの登録」ウィンドウを使用します。ビューを登録するキャプチャー・スキーマを展開します。「**登録済みビュー**」フォルダーを右クリックして、「**ビューの登録**」をクリックします。.
System i ADDDPRREG システム・コマンド	System i の場合は、ADDDPRREG コマンドを使用してビューを登録します。

CCD 表をソースとして保守する (IMS)

IMS DataPropagator や DataRefresher™ などのプログラムによって保守され、外部でデータが取り込まれる CCD 表がある場合は、アプライ・プログラムがソースとして CCD 表を読み取ることができるように、これらの表を保守する必要があります。

手順

外部のツールによってデータが取り込まれている CCD 表を保守するには、以下のようにします。

以下のそれぞれのタイプのイベントに関して、IBMSNAP_REGISTER 表の 3 つの列 (CCD_OLD_SYNCHPOINT、SYNCHPOINT、SYNCHTIME) を更新します。

イベント	必要な更新
CCD 表の初回のフル・リフレッシュまたはロード	• CCD_OLD_SYNCHPOINT を CCD 表の IBMSNAP_COMMITSEQ の最小値を表す値に設定する。 • SYNCHPOINT を CCD 表の IBMSNAP_COMMITSEQ の最大値を表す値に設定する。 SYNCHPOINT を 0 に設定しないでください。順序付け用に独自の値を作成する場合は、初回用の SYNCHPOINT 値は 1 にしてください。 • SYNCHTIME を CCD 表の IBMSNAP_LOGMARKER の最大タイム・スタンプ値を表す値に設定する。
フル・リフレッシュまたはロードの後の CCD 表の更新	• CCD_OLD_SYNCHPOINT 値を変更してはならない。 • SYNCHPOINT を CCD 表の IBMSNAP_COMMITSEQ の新しい最大値を表す値に設定する。 • SYNCHTIME を CCD 表の IBMSNAP_LOGMARKER の新しい最大タイム・スタンプ値を表す値に設定する。
CCD 表の 2 回目以降のフル・リフレッシュまたはロード	• CCD_OLD_SYNCHPOINT を CCD 表の IBMSNAP_COMMITSEQ の最小値を表す値に設定する。 • SYNCHPOINT を CCD 表の IBMSNAP_COMMITSEQ の最大値を表す値に設定する。 • SYNCHTIME を CCD 表の IBMSNAP_LOGMARKER の最大タイム・スタンプ値を表す値に設定する。

重要: 上記では、CCD 表で IBMSNAP_COMMITSEQ および IBMSNAP_LOGMARKER に使用されている値は常に増えてゆく値であることを前提としています。アプライ・プログラムは、CCD_OLD_SYNCHOINT 値が最新に適用された SYNCHPOINT 値より大きくならないかぎり、ソース CCD 表でフル・リフレッシュが実行されたことを検出しません。

第 5 章 SQL レプリケーションのソースのサブスクライブ

レプリケーション・ソースとして表またはビューを登録した後に、ターゲット表またはターゲット・ビューが初期のソース・データとそれ以降の変更を受け取れるように、ターゲット表またはターゲット・ビューのサブスクリプションを定義できます。

このセクションで説明する管理タスクは、キャプチャー・プログラムとアプライ・プログラムが、ソース・データをコピーするため、または変更データを取り込んでそれをターゲット表に適切なインターバルで複製するために使用するコントロール情報をセットアップするときに役立ちます。

以下のトピックでは、ソースにサブスクライブする方法を詳しく説明しています。

ソースおよびターゲットの分類方法の計画

どのターゲットがどのソースをサブスクライブするかを定義する前に、ソースとターゲットの分類方法を計画する必要があります。

SQL レプリケーションは、グループ単位でソースからターゲットへのマッピングを処理します。これらのグループは、同一のキャプチャー・プログラムによって処理される 1 つ以上のソースと、ソース・データのすべてまたは一部をサブスクライブし、同一のアプライ・プログラムによって処理される 1 つ以上のターゲットで構成されています。これらのグループを、サブスクリプション・セットと呼び、ソースからターゲットへのマッピングをサブスクリプション・セット・メンバーと呼びます。

サブスクリプション・セットを計画するときは、以下の規則と制約に注意してください。

- サブスクリプション・セットは、ソース・サーバーをターゲット・サーバーにマップする。サブスクリプション・セット・メンバーは、ソース表またはビューを、ターゲット表またはビューにマップします。サブスクリプション・セットとそのメンバーは、アプライ・コントロール・サーバーに保管されます。
- アプライ・プログラムは、サブスクリプション・セットのすべてのメンバーを単一グループとして処理する。このため、サブスクリプション・セットのいずれかのメンバーで、何らかの理由でフル・リフレッシュ・コピーが必要な場合、セット全体のすべてのメンバーがリフレッシュされます。
- 1 つのセットのメンバーにおいて、すべてのソース表およびビューのキャプチャー・スキーマは同じでなければならない。
- System i では、1 つのサブスクリプション・セットのメンバーにおいて、すべてのソース表は同じジャーナルに記録されなければならない。
- IMS DataPropagator で作成され、サブスクリプション・セットのメンバーである外部 CCD 表すべてのキャプチャー・スキーマは同じでなければならない。

ユニークなアプライ修飾子を持つ単一のアプライ・プログラムは、サブスクリプション・セットを 1 つでもあるいは多数でも処理できます。単一のサブスクリプション・セットには、サブスクリプション・セット・メンバーを 1 つでもあるいは多数でも入れることができます。

以下のトピックでは、サブスクリプション・セットをアプライ・プログラムごとにグループ化する構成と、サブスクリプション・セット・メンバーをサブスクリプション・セットごとにグループ化する構成を比較し、それぞれのメリットとデメリット (トレードオフ) について説明しています。

サブスクリプション・セット・メンバー数の計画

サブスクリプション・セットにメンバーを追加するとき、ソースとターゲットのすべてのペア (サブスクリプション・セット・メンバー) を 1 つのサブスクリプション・セットにまとめるか、それぞれのペアごとに別々のサブスクリプション・セットを作成するか、あるいはそれぞれにかなりの数のペアが含まれた、少数のサブスクリプション・セットを作成するかを決定しなければなりません。

アプライ・プログラムは、サブスクリプション・セットのメンバーを 1 つの (論理) トランザクションで複製するため、以下のどちらの状態においても、複数のメンバーを 1 つのサブスクリプション・セットにまとめる必要があります。

- ソース表に相互に論理的な関連がある場合。
- ターゲット表に参照整合性制約がある場合。

複数のメンバーを 1 つのサブスクリプション・セットにまとめると、すべてのメンバーのレプリケーションを必ず同時に開始することができます。さらに、サブスクリプション・セットを処理するのに必要なデータベース接続の数と、レプリケーション環境を保守するための管理オーバーヘッドが削減されます。サブスクリプション・セットに SQL ステートメントまたはストアード・プロシージャーが含まれている場合、これらのステートメントまたはプロシージャーを使用して、そのサブスクリプション・セットのすべてのメンバーを処理できます。

サブスクリプション・セットの表の間に、論理的または参照整合性のリレーションシップがない場合、その表を 1 つまたはいくつかのサブスクリプション・セットにまとめることができます。サブスクリプション・セットの数を制限する主な理由は、レプリケーション環境の管理を簡単にすることです。ただし、サブスクリプション・セットの数を増やすと、レプリケーションの失敗による影響は最小になります。

アプライ・プログラムが失敗する原因となるエラーをより簡単に突き止められるようにするには、少数のメンバーしかサブスクリプション・セットに追加しないようにします。メンバーが少数なら、多数のメンバーがセットに含まれている場合より迅速に問題のソースを見つけることができます。サブスクリプション・セットの 1 つのメンバーで障害が起こった場合、そのセットの他のメンバーに適用されたデータはすべてロールバックされます。そのため、すべてのメンバーがサイクルを完了しない限り、どのメンバーも正常にサイクルを完了できません。アプライ・プログラムは、失敗したサブスクリプション・セットをその最後の正常なコミット・ポイントまでロールバックします。このコミット・ポイントは、アプライ・プログラムを始動したときに commit_count キーワードを指定していれば、現行のアプライ・サイクル内にある可能性があります。

アプライ修飾子ごとのサブスクリプション・セット数の計画

サブスクリプション・セットを定義するとき、そのサブスクリプション・セット用のアプライ修飾子を指定します。アプライ修飾子は、アプライ・プログラムのインスタンスを 1 つ以上のサブスクリプション・セットに関連付けます。

それぞれのサブスクリプション・セットを処理するのは、ただ 1 つのアプライ・プログラムですが、アプライ・プログラムはそれぞれ 1 つ以上のサブスクリプション・セットを、 1 つ 1 つのアプライ・サイクル内で処理できます。

アプライ・プログラムのインスタンス (それぞれに独自のアプライ修飾子があります) は必要な数だけ実行でき、アプライ・プログラムはそれぞれサブスクリプション・セットを必要な数だけ処理できます。基本オプションが 2 つあります。

各アプライ修飾子を 1 つのサブスクリプション・セットに関連付ける

各アプライ・プログラムが 1 つのサブスクリプション・セットを処理します。

速度が重要な場合は、いくつかのアプライ修飾子間にセットを分散でき、こうすることで、アプライ・プログラムのいくつかのインスタンスを同時に実行できます。

1 つのアプライ・プログラム・インスタンスで 1 つのサブスクリプション・セットを処理するように決めている場合は、アプライ・プログラムの OPT4ONE 始動オプションを使用できます。このオプションは、サブスクリプション・セットのコントロール表の情報をメモリーにロードします。

このオプションを使用すると、アプライ・プログラムは、サブスクリプション・セット情報のコントロール表をアプライ・サイクルごとには読み取りません。したがって、アプライ・プログラムのパフォーマンスが向上します。ただし、実行するアプライ・プログラム・インスタンスが多くなれば、それらが使用するシステム・リソースも多くなり、全体としてのパフォーマンスは低下する可能性があります。

アプライ修飾子をそれぞれ複数のサブスクリプション・セットに関連付ける

各アプライ・プログラムが多数のサブスクリプション・セットを処理します。

複数のアプライ修飾子を使用すれば、シングル・ユーザー ID からアプライ・プログラムの複数のインスタンスを実行できます。

アプライ・プログラムでは、指定されたアプライ修飾子に対応するセットすべてを、可能な限り現状のままで保持しようとします。アプライ・サイクルが開始すると、アプライ・プログラムは含まれる現行データが最も少ないサブスクリプション・セットを判別し、まずそのセットの処理を開始します。

速度を主な目的と考えない場合は、 1 つのアプライ修飾子を指定して、膨大な数のサブスクリプション・セットを複製することができます。例えば、営業時間終了後まで待ってから複製する場合に、これは大変有効なオプションとなります。

1 つのアプライ・プログラムで複数のサブスクリプション・セットを処理する欠点は、そのアプライ・プログラムがサブスクリプション・セットを順次処理することです。したがって、全体のレプリケーション待ち時間は増加する可能性があります。

一部のサブスクリプション・セットに対して特定の要件がある場合、この 2 つのオプションを組み合わせることができます。例えば、1 つのアプライ・プログラムにほとんどのサブスクリプション・セットを処理させることができるので、関連するサブスクリプション・セットをまとめて処理すれば、アプライ・プログラムを有効に利用できるでしょう。そして、別のアプライ・プログラムに単一のサブスクリプション・セットを処理させることができるので、そのサブスクリプション・セットのレプリケーション待ち時間を確実に最小にできます。さらに、アプライ・プログラムの 2 つのインスタンスを使用すれば、サブスクリプション・セットの並列処理全体が強化されます。

サブスクリプション・セットの作成

登録済みのソースからデータを複製する前に、アプライ・プログラムが 1 つの集合として処理するサブスクリプション・セット・メンバー (ソースからターゲットへのマッピング) のコレクションである、サブスクリプション・セットを作成しなければなりません。

始める前に

- サブスクリプション・セット用に、アプライ・コントロール表をアプライ・コントロール・サーバーで作成します。
- サブスクリプション・セットにサブスクリプション・セット・メンバーを追加する前に、ソースとして使用する表またはビューを登録します。 セットの分類方法について検討する必要もあります。

このタスクについて

サブスクリプション・セットを作成するときには、ソース・サーバーとターゲット・サーバー、使用するキャプチャー・プログラムとアプライ・プログラム、アプライ・プログラムでセットを処理するタイミングと方法を指定します。

サブスクリプション・セットにサブスクリプション・セット・メンバーを追加する必要はありません。ソースからターゲットへのマッピングを含まない空のセットを作成できます。以下のような理由から、空のセットの作成が必要になることがあります。

- 後でメンバーをセットに追加する計画があり、メンバーを追加するまではサブスクリプション・セットをアクティブにする計画はない。
- 空のサブスクリプション・セットが処理に適格であるとき、いつでも SQL ステートメントまたはストアード・プロシージャーを呼び出すために、アプライ・プログラムでそのセットを処理する。

手順

サブスクリプション・セットを作成するには、次の方法のいずれかを使用します。

方法	説明

ASNCLP コマンド行プログラム	CREATE SUBSCRIPTION SET コマンドを使用します。このコマンドで作成できるのは空のサブスクリプション・セットだけであるのに対し、レプリケーション・センターを使用した場合は、セットの作成時にメンバーを追加できます。
	
	以下のコマンドは、環境を設定し、アプライ修飾子 AQ00 を使用して SET00 という名前のサブスクリプション・セットを作成します。
	
	`SET SERVER CAPTURE TO DB SAMPLE;`
	`SET SERVER CONTROL TO DB TARGET;`
	`SET OUTPUT CAPTURE SCRIPT "capsubset.sql"`
	`CONTROLSCRIPT "appsubset.sql";`
	`SET LOG "subset.err";`
	`SET RUN SCRIPT LATER;`
	`CREATE SUBSCRIPTION SET SETNAME SET00 APPLYQUAL AQ00`
	`ACTIVATE YES TIMING INTERVAL 1 START DATE "2006-10-22"`
	`TIME "09:00:00.000000";`

レプリケーション・センター	「サブスクリプション・セットの作成」ノートブックを使用します。そのノートブックを開くには、セットを定義するアプライ・コントロール・サーバーを展開し、「**サブスクリプション・セット**」フォルダーを右クリックし、「**作成**」をクリックします。

System i
 	
ADDDPRSUB システム・コマンド
	1 つのメンバーを持つ、またはメンバーを持たないサブスクリプション・セットを作成するには、DPR サブスクリプション・セットの追加 (ADDDPRSUB) コマンドを使用します。
	
	例えば、AQHR アプライ修飾子の下に SETHR という名前のサブスクリプション・セットを作成するには、以下のようにします。
	
	`ADDDPRSUB APYQUAL(AQHR) SETNAME(SETHR) SRCTBL(HR/EMPLOYEE)`
	`TGTTBL(TGTLIB/TGTEMPL)`
	
	このサブスクリプション・セットには 1 つのサブスクリプション・セット・メンバーが含まれ、 HR ライブラリー下の EMPLOYEE という名前の登録済みソース表から、 TGTLIB ライブラリー下の TGTEMPL という名前のターゲット表にデータを複製します。

以下の基本的な特性を指定します。

アプライ・コントロール・サーバーの別名

サブスクリプション・セットを処理するアプライ・プログラム用のコントロール表を含むサーバーのローカルの別名。すべてのデータベースに、レプリケーション・センター、ASNCLP、アプライ・プログラムを実行するアプライ・コントロール・サーバーの同一の別名を定義します。そのようにすると、管理ツールはアプライ・コントロール表にデータを正しく取り込むことができ、どのアプライ・プログラムも標準の別名を使用して適切なサーバーに接続できるようになります。

サブスクリプション・セット名

サブスクリプション・セットの名前。サブスクリプション・セットを処理するアプライ・コントロール・サーバーにおいて、セット名は指定されたアプライ修飾子に対してユニークでなければなりません。名前の長さは 18 文字まで可能です。

アプライ修飾子

新規または既存のアプライ修飾子の名前。サブスクリプション・セットを処理するアプライ・プログラムを示します。同じアプライ修飾子を使用して、複数のサブスクリプション・セットを処理できます。同じアプライ修飾子を持つサブスクリプション・セットは、同じアプライ・コントロール・サーバーで定義しなければなりません。

キャプチャー・コントロール・サーバーの別名

サブスクリプション・セットに登録済みのソースを処理するキャプチャー・プログラム用のコントロール表を含むサーバーの別名。すべてのデータベースに、レプリケーション・センター、ASNCLP、アプライ・プログラムを実行するキャプチャー・コントロール・サーバーの同一の別名を定義します。そのようにすると、管理ツールはキャプチャー・コントロール表とアプライ・コントロール表にデータを正しく取り込むことができ、どのアプライ・プログラムも標準の別名を使用して適切なサーバーに接続できるようになります。

キャプチャー・スキーマ

キャプチャー・スキーマの名前。サブスクリプション・セットに登録済みのソースを定義するキャプチャー・コントロール表のセットを示します。サブスクリプション・セットのすべてのソース表は、同じサーバーに常駐していなければならず、 1 つのキャプチャー・プログラムだけがソース表への変更をキャプチャーすることができます。

ターゲット・サーバーの別名

アプライ・プログラムがソースからの変更を複製する先の表またはビューが含まれるターゲット・サーバーの名前。すべてのデータベースに、レプリケーション・センター、ASNCLP、アプライ・プログラムを実行するターゲット・サーバーの同一の別名を定義します。そのようにすると、管理ツールはアプライ・コントロール表にデータを正しく取り込むことができ、どのアプライ・プログラムも標準の別名を使用して適切なサーバーに接続できるようになります。

サブスクリプション・セットを作成するときは、アプライ・プログラムがセットを処理する方法について、デフォルトの設定値を使用できます。あるいは、サブスクリプションのプロパティーを、レプリケーションの要求を満たすように変更できます。

サブスクリプション・セットのオプション処理

サブスクリプション・セットを作成するときは、アプライ・プログラムがセットを処理する方法に関するオプションを定義します。

以下のトピックは、レプリケーションのニーズに基づいてどの設定値を選択するべきかを決定するときに役立ちます。

サブスクリプション・セットがアクティブかどうかの指定

アプライ・プログラムがサブスクリプション・セットの処理を開始するかどうかを指定できます。サブスクリプション・セットを活動化すると、アプライ・プログラムは、そのセットのフル・リフレッシュを開始します。

活動化レベルは以下の 3 つから選択します。

アクティブ
アプライ・プログラムは、それの次のサイクル内でセットを処理します。アプライ・プログラムに次の実行時でセットを処理させる場合は、セットを活動化します。後になっても、メンバーをセットに追加できます。セットをアクティブ化すると、ユーザーがセットを非アクティブ化するまで、セットはアクティブのままで、アプライ・プログラムはその処理を続けます。

非アクティブ
アプライ・プログラムはセットを処理しません。アプライ・プログラムでセットを処理する準備ができていない場合は、セットを非アクティブにしておきます。

1 回のみアクティブ
アプライ・プログラムは次のサイクル内でセットを処理し、その後セットを非活動化します。セットを 1 回だけ実行したい場合は、このオプションを指定します。このオプションを選択する前に、必ずすべてのサブスクリプション・セット・メンバーを追加してください。その理由は、サブスクリプション・セットを再活動化しない限り、アプライ・プログラムは後で追加したメンバーを処理しないためです。

アプライ・プログラムが取り出すデータに相当する分数の指定

アプライ・プログラムが毎回のアプライ・サイクルでレプリケーション・ソースから取り出すデータの量をおおよその分数で指定できます。

このオプションは、以下のような状況で役立ちます。

- 1 回のサブスクリプション・セット・サイクル内で処理されるデータ量が大量の場合。

  1 回のアプライ・サイクルで大量の変更ブロックを複製するサブスクリプション・セットは、予備ファイルまたはログ (ターゲット・データベースの場合) でのオーバーフローの原因となります。 例えば、アプライ・プログラムをバッチ処理するシナリオでは、レプリケーションを必要とするエンキューされたトランザクションのバックログが大量に生成される可能性があります。
- ネットワークの停止が長引くと、大量のデータ・ブロックが CD 表に累積され、アプライ・プログラムの予備ファイルとターゲットのログがオーバーフローする可能性がある。

指定する分数を、データ・ブロックと呼びます。指定するデータ・ブロッキングの値は、 IBMSNAP_SUBS_SET の表で MAX_SYNCH_MINUTES 列に保管されます。データの累積がデータ・ブロックのサイズより大きいと、アプライ・プログラムは 1 つのアプライ・サイクルをいくつかのミニサイクルに変換します。リソースが、指定されたブロッキング因数を処理するにはまだ不十分な場合、アプライ・プログラムはデータ・ブロックのサイズを削減し、使用可能なシステム・リソースに

一致させます。より小さいデータ集合を取り出すことにより、アプライ・プログラムはネットワーク負荷と取り出されたデータ用に一時的に必要なスペースの両方を減少させます。

各アプライ・サイクル中に、サブスクリプション・セットの MAX_SYNCH_MINUTES 値が NULL であるか、または 1 未満の数値に設定された場合、アプライ・プログラムはその設定に適格なすべてのデータを 1 つのアプライ・サイクルで処理します。 CD 表と UOW 表に大容量のデータが含まれている場合は、その状況のためにデータベース・トランザクション・ログがいっぱいになったり、または予備ファイルがオーバーフローするという問題が起きることがあります。以下のガイドラインを使用して、MAX_SYNCH_MINUTES を NULL 以外の値に変更することができます。

- ASN.IBMSNAP_SUBS_SET 表の SLEEP_MINUTES 列が特定のサブスクリプション・セットに対して 5 分 (またはそれ未満) に設定されている場合は、MAX_SYNCH_MINUTES を 5 分に設定します。
- 特定のサブスクリプション・セットに対して、 SLEEP_MINUTES が 30 分 (またはそれ以上) に設定されている場合は、 MAX_SYNCH_MINUTES を 60 分に設定します。
- SLEEP_MINUTES が 5 ～ 30 分の間に設定されている場合は、MAX_SYNCH_MINUTES を SLEEP_MINUTES と等しい値に設定します。

レプリケーション環境をモニターして、必要に応じて MAX_SYNCH_MINUTES を調整します。 MAX_SYNCH_MINUTES の数値がゼロより大きいことを確認してください。

例: アプライ・プログラムがミニサイクル当たりせいぜい 10 分相当のデータしか取り出さないように指定すると、最後のミニサイクルの約 10 分間以内で、アプライ・プログラムはかなりの量のコミット済みデータをソースの CD 表から取り出すことになります。

ログ・ファイルと予備ファイルをオーバーフローさせない点に加えて、ミニサイクルには、他にもいくつかの利点があります。レプリケーション・サイクルでエラーが発生している場合、アプライ・プログラムは、失敗したミニサイクルで行った変更のみをロールバックしなければなりません。ミニサイクルでレプリケーションが失敗した場合、アプライ・プログラムは、最後に正常に実行されたミニサイクルからサブスクリプション・セットを処理しようとしますが、これで大量の変更データを処理できる場合は、かなりの時間を節約できます。 71 ページの図 5 に、変更データが変更のサブセットに分割される仕組みを示します。

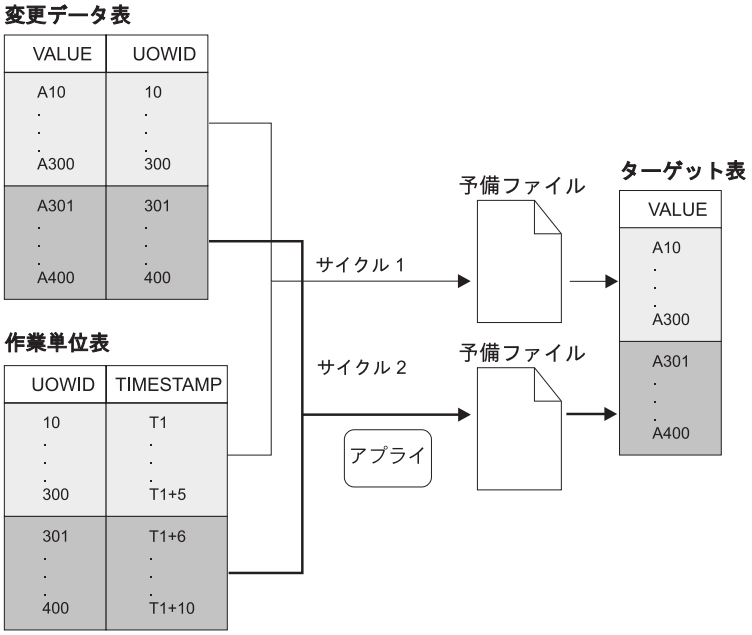


図 5. データ・ブロッキング: ネットワークのトラフィック量を少なくするには、データ・ブロック値を指定します。

設定する分数が小さければ、インターバル内に生じるサブスクリプション・セットのトランザクションをすべてコピーし、ミニサイクル内での予備ファイルやログのオーバーフローを防ぐことができます。

データを処理するとき、アプライ・プログラムは以下のどのアクションも行いません。

- 作業単位の分割 (実行時間の長い、コミットなしのバッチ・ジョブが、データ・ブロッキング係数で分割されることはありません)
- 前にコミット済みの小サブスクリプション・サイクルのロールバック
- フル・リフレッシュ中の、データ・ブロッキング係数の使用

参照整合性のあるターゲット表のためのロード・オプション

ターゲット表にソース・データをロードするまで、ターゲット表どうしの間で参照整合性制約を追加する操作を延期する場合もあります。

アプライ・プログラムの始動パラメーターを設定するときに、ターゲットをロードする方法を決定します。ターゲット表どうしの間の参照整合性リレーションシップを作成する方法として、以下のような代案も検討できます。

ターゲット表をロードする前

この場合は、ターゲット表の抽出およびロードの全段階を通じて、ソース表への変更が行われないことが必要です。また、ロード時の参照制約チェックをバイパスするため、LOADX 始動オプションを使用してアプライ・プログラムを始動しなければなりません。LOADX オプションを使用しない場合は、ターゲット表への挿入が失敗する可能性があります。始動オプション LOADX を使用するよりも、一般的にフル・リフレッシュはかなり速度があります。

ロードが完了し、アプライがターゲットに変更を適用する 1 つのサイクルが完了した後
このオプションの場合は、ターゲット表のロード中にも、ソース表を変更できます。バイパスする必要があるという制約がないため、始動オプション LOADX を使用してもしなくても、アプライ・プログラムを始動できます。ターゲット表の初期の移植中、ターゲットは参照整合性リレーションシップの点で相互に同期がとれていないことがあります。表のロード時に、セットのすべての変更がキャプチャーされます。アプライ・プログラムが変更の入った最初のセットを複製した後、すべてのターゲット表には同じトランザクションが入り、参照整合性を持つようになります。この時点でセットを非活動化して参照整合性制約を追加し、その後セットを再び活動化できます。

アプライ・プログラムがサブスクリプション・セット・メンバーの変更を複製する方法の指定

サブスクリプション・セットについて、変更キャプチャーによるレプリケーションを実行する場合は、アプライ・プログラムがターゲット表またはターゲット・ビューの変更をそれぞれのサブスクリプション・セット・メンバーごとに 1 回ずつコミットするか、トランザクションをいくつか適用した後でコミットするかを指定できます。

ターゲット表の初期ロード後、アプライ・プログラムは CD (または CCD) 表の読み取りを開始し、変更を予備ファイルに収集します。その後、プログラムは 2 つの方法のいずれかで変更を適用します。

表モード
アプライ・プログラムは、それぞれのサブスクリプション・セット・メンバーごとに 1 回ずつ変更をコミットします。

アプライ・プログラムは CD (または CCD) 表の予備ファイルからすべての変更を読み取り、対応するターゲット表に変更を適用してから、次の CD (または CCD) 表の処理を開始します。セット内のすべての CD (または CCD) 表からの変更の読み取りと適用が完了したら、アプライ・プログラムは DB2 コミットを発行して、サブスクリプション・セット内のすべてのターゲット表への変更をすべてコミットします。

トランザクション・モード
アプライ・プログラムは、指定した数のトランザクションを適用した後に変更をコミットします。サブスクリプション・セットのターゲット表に参照整合性制約がある場合は、トランザクション・モード処理を使用してください。

このモードの場合、アプライ・プログラムは、すべての予備ファイルを一度に開き、変更を同時に処理します。変更は、ソース表での発生順のとおりに適用されます。IBMSNAP_SUBS_SET 表の COMMIT_COUNT 列は、そのサブスクリプション・セットのすべてのターゲット表に変更が適用されコミットされる方法を制御します。

トランザクション・モード処理では、ユーザー・コピーとポイント・イン・タイムのターゲット表のセットに関するアプライ・プログラムの動作だけが変更されます。以下の制限にも注意する必要があります。

- CCD 表を含むセットは常に表モードで処理されます。

- レプリカ表を含むセットは常にトランザクション・モードで処理されます。

コミットが 1 回の場合、サブスクリプション・セットの待ち時間を削減できますが、コミットが複数の場合は、アプライ・プログラムがオリジナルのコミット・シーケンスのデータを適用できます。

サブスクリプション・セットのターゲット表のタイプによっては、表モードとトランザクション・モードを混合した処理を使用することも可能です。

サブスクリプション・セット用の SQL ステートメントまたはストアード・プロシージャーの定義

アプライ・プログラムがサブスクリプション・セットを処理するたびに実行される SQL ステートメントまたはストアード・プロシージャーを定義できます。これらのステートメントは、CCD 表の整理、またはターゲットに適用される前のソース・データの取り扱いに役立ちます。

SQL ステートメントまたはストアード・プロシージャーの実行時期と実行場所は次のとおりです。

- アプライ・プログラムがデータを適用する前にキャプチャー・コントロール・サーバーで。
- アプライ・プログラムがデータを適用する前にターゲット・サーバーで。
- アプライ・プログラムがデータを適用した後にターゲット・サーバーで。

レプリケーション・センターを使用して SQL ステートメントをサブスクリプション・セットに追加するときは、「SQL ステートメントまたはプロシージャー呼び出しの追加」ウィンドウで「**ステートメントの準備**」をクリックし、構文を検証します。

サブスクリプション・セットのレプリケーションをスケジューリングするためのオプション

アプライ・プログラムがサブスクリプション・セットを処理する頻度を指定して、ターゲット表のデータをどの程度最新に保つかをコントロールできます。時間に基づくスケジューリング、イベントに基づくスケジューリング、またはこれらのオプションの組み合わせを使用できます。

例えば、インターバルをアプライ・サイクルの間の 1 日に設定し、さらにこのサイクルを起動するイベントを指定することもできます。これらのスケジューリング・オプションの両方を使用する場合、サブスクリプション・セットはスケジュールされた時刻とイベント発生時刻の両方で処理対象として適格になります。

Update-anywhere レプリケーションでは、 master-to-replica サブスクリプション・セットと replica-to-master サブスクリプション・セットに使用するタイミングは同じであっても異なっていても構いません。

インターバル内またはイベントとイベントの間で複製されるデータ量が膨大な場合、アプライ・プログラムがサブスクリプション・セットを処理できるのは、先行するインターバルまたはイベントで、すべてのセットへのデータの適用が終了して

からです。このケースでは、レプリケーション待ち時間が予想とは異なるかもしれませんが、失われるデータはありません。

時間に基づくスケジューリング

セットを処理する時期をコントロールする最も単純な方法は、時間に基づくスケジューリング (相対タイミングまたはインターバル・タイミングとしても知られています) を使用することです。特定の開始日付、時間、およびインターバルを決定します。インターバルは、特定の値 (1 分 ～ 1 年) または連続した値にすることができますが、時間間隔はおおよその値になります。

アプライ・プログラムは、ワークロードとリソースの可用性に基づいて、可能な限りすみやかにサブスクリプション・セットの処理を開始します。ある時間間隔を選択しても、レプリケーションが正確にその頻度で行われるとは限りません。連続タイミングを指定すると、アプライ・プログラムは可能な限り頻繁にデータを複製します。

イベントに基づくスケジューリング

イベントに基づくスケジューリング (イベント・タイミングとしても知られています) を使用してデータを複製するために、サブスクリプション・セットを定義するときにイベント名を指定します。イベント名に対応するタイム・スタンプを IBMSNAP_SUBS_EVENT 表に移植することも必要です。アプライ・プログラムはイベントを検出するとレプリケーションを開始します。

IBMSNAP_SUBS_EVENT 表には、表 3 で示されているように、4 つの列があります。

表 3. IBMSNAP_SUBS_EVENT 表に保管されるデータの例

EVENT_NAME	EVENT_TIME	END_OF_PERIOD	END_SYNCHPOINT
END_OF_DAY	2002-05-01-17.00.00.000000	2002-05-01-15.00.00.000000	

EVENT_NAME 列は、サブスクリプション・セットの定義時に指定するイベントの名前を保管します。 EVENT_TIME は、アプライ・プログラムがセットの処理を開始する時刻を示すタイム・スタンプです。 END_OF_PERIOD は、該当時刻より後の更新が後のイベントまたは時刻まで据え置かれることを指定するオプション値です。 END_SYNCHPOINT は、該当するログ・シーケンス番号より後の更新が後のイベントまたは時刻まで据え置かれることを指定するオプション値です。 END_OF_PERIOD と END_SYNCHPOINT の両方の値を指定すると、 END_SYNCHPOINT の値が優先されます。 EVENT_TIME の値はアプライ・コントロール・サーバーのクロックにより設定されますが、 END_OF_PERIOD の値はソース・サーバーのクロックにより設定されます。 2 つのサーバーが別の時間帯にある場合、この区別は重要です。

表 3 によると、END_OF_DAY というイベントの場合、 EVENT_TIME のタイム・スタンプ値 (2002-05-01-17.00.00.000000) は、アプライ・プログラムがサブスクリプション・セットの処理を開始する時刻です。END_OF_PERIOD のタイム・スタンプ値 (2000-05-01-15.00.00.000000) は、更新が複製されなかった後、次の日のサイクル

で複製される時刻です。つまり、イベントはこの時刻の前に作成されたすべての未確定の更新を複製して、それに続くすべての更新を延期します。

ユーザーまたはアプリケーションは、行を表に挿入する SQL の INSERT ステートメントを使用してイベントを IBMSNAP_SUBS_EVENT 表に通知し、イベントを活動化しなければなりません。例えば、現行のタイム・スタンプに 1 分加算した値を使用して、 EVENT_NAME に指定したイベントを起動します。このイベントに結び付けられたサブスクリプション・セットはすべて、 1 分以内で実行するのに適格なものとなります。フル・リフレッシュと変更キャプチャー・レプリケーションの両方について、イベントを手動で通知しなければなりません。

イベントは、翌週、翌年、または毎週土曜日のように、前もって通知することができます。アプライ・プログラムが実行されている場合、アプライ・プログラムは指定されたおおよその時刻に処理を開始します。アプライ・プログラムは、指定された時刻に停止しており、後で再始動されると、サブスクリプション・イベント表をチェックして、通知されたイベントのサブスクリプション・セットの処理を開始します。

アプライ・プログラムは表を整理しません。自力で表にデータを追加して保守しなければなりません。さらに、レプリケーション・センターを使用してサブスクリプション・イベント表を更新することはできません。この表にイベントを追加するには、SQL ステートメントを発行するか自動手順を定義しなければなりません。

例:

```
INSERT INTO ASN.IBMSNAP_SUBS_EVENT
       (EVENT_NAME, EVENT_TIME)
       VALUES ('EVENT01', CURRENT TIMESTAMP + 1 MINUTES)
```

アプライ・プログラムがサブスクリプション・セットを処理した最新の時点 (サブスクリプション・セットのコントロール表の LASTRUN 列にある値で指定) より前に発生するイベントは、期限切れのイベントと見なされて無視されます。そのため、アプライ・プログラムが実行中である場合は、期限切れのイベントを通知することを避けるため、時間的にわずかに先のイベントを通知してください。

サブスクリプション・セットのスケジューリング

ソースからターゲットへのマッピングを行った後 (または空のサブスクリプション・セットを作成した後)、サブスクリプション・セットのタイミング情報を定義します。

ソースからターゲットへのマッピングを行った後 (または空のサブスクリプション・セットを作成した後)、サブスクリプション・セットのタイミング情報を定義します。「サブスクリプション・セットの作成」ウィンドウの「スケジュール」ページで、サブスクリプション・セットを最初に処理の対象とするのはいつかを指定します。デフォルトは、ローカル・マシンの現在の日付と時刻です。また、サブスクリプション・セットをどのような頻度で処理の対象にするかも指定します。次のような頻度を指定できます。

- 時間に基づくレプリケーション

  アプライ・プログラムは、定期的な時間間隔を使用してこのサブスクリプション・セットを処理します。

- イベントに基づくレプリケーション

  アプライ・プログラムは、あるイベントが起こるたびにこのサブスクリプション・セットを処理します。
- 時間に基づくレプリケーションとイベント・ベースのレプリケーションの両方

  アプライ・プログラムは、定期的な時間インターバルおよびイベントが起こる都度の両方を使用して、このサブスクリプション・セットを処理します。この場合、サブスクリプション・セットは、スケジュールされた時刻および、イベントが起こった時の両方で処理の対象になります。

サブスクリプション・セット・メンバーの作成

アプライ・プログラムがグループ処理を行うために、サブスクリプション・セットの中にソースからターゲットへのマッピングを追加できます。ソースからターゲットへのこれらのマッピングを、サブスクリプション・セット・メンバーと呼びます。

始める前に

ソースでの変更をサブスクライブするターゲットをセットアップする前に、ソースとして使用する表またはビューを登録しなければなりません。さらに、サブスクリプション・セットを作成したり、セットに追加するメンバーの数を計画する必要もあります。

制約事項

- SQL レプリケーションは非 DB2 リレーショナル表のビューをソースとしてサポートしない。
- ターゲット・ビューを定義する場合、そのビューは挿入可能なビューでなければならない。すなわち、ビューのすべての列は更新可能でなければならず、ビューの全選択にキーワード UNION ALL を組み込むことができません。
- レプリケーション・センターを使用する場合、該当する列がターゲット表にまだ存在していなければ、サブスクリプション・セット・メンバーにその列を追加できない。
- **z/OS:** ROWID 列がレプリケーション用に指定された唯一のユニーク索引である場合を除き、ROWID 列をレプリケーションのために選択しないでください。

  推奨: レプリケーションのためのユニーク索引には、ROWID 列ではなく IDENTITY 列を使用してください。
- z/OS Linux UNIX Windows それぞれのサブスクリプション・セットごとに、最大 200 のメンバーを定義できます。
- System i それぞれのサブスクリプション・セットごとに、最大 78 のメンバーを定義できます。

このタスクについて

サブスクリプション・セット・メンバーを定義するとき、ソース・データをサブスクライブするターゲット表またはビューを指定し、複製されたデータのターゲットでの表示方法を定義できます。

手順

サブスクリプション・セット・メンバーを追加するには、次の方法のいずれかを使用します。

方法	説明

ASNCLP コマンド行プログラム	CREATE MEMBER コマンドを使用して、既存のサブスクリプション・セットにサブスクリプション・セット・メンバーを追加します。例えば、以下のような特性を持ったコマンドを使用します。
	• 環境を設定します。
	• ターゲット表が使用する表スペースのオプションを格納するためのプロファイル TBSPROFILE を作成します。
	• SET00 サブスクリプション・セット、AQ00 アプライ修飾子、STAFF ソース表を指定します。
	• 新しいターゲット表 TRGSTAFF を全列登録済みのユーザー・コピーとして作成するように指定します。
	`SET SERVER CAPTURE TO DB SAMPLE;`
	`SET SERVER CONTROL TO DB TARGET;`
	`SET SERVER TARGET TO DB TARGET;`
	`SET OUTPUT CAPTURE SCRIPT "capmember.sql"`
	`CONTROLSCRIPT "appmember.sql"`
	`SET LOG "member.err";`
	`SET RUN SCRIPT LATER;`
	`SET PROFILE TBSPROFILE FOR OBJECT TARGET TABLESPACE`
	`OPTIONS UW USING FILE "/tmp/db/ts/TSTRG.TS" SIZE 700 PAGES;`
	`CREATE MEMBER IN SETNAME SET00 APPLYQUAL AQ00`
	`ACTIVATE YES SOURCE STAFF TARGET NAME TRGSTAFF`
	`DEFINITION IN TSTRG00 CREATE USING PROFILE TBSPROFILE`
	`TYPE USERCOPY COLSALL REGISTERED;`

レプリケーション・センター	以下のノートブックのいずれかを使用します。 • サブスクリプション・セットの作成。サブスクリプション・セットを作成するときは、このノートブックを使用します。セットを定義するアプライ・コントロール・サーバーを展開し、「**サブスクリプション・セット**」フォルダーを右クリックし、「**作成**」をクリックします。 • サブスクリプション・セット・プロパティー。すでにサブスクリプション・セットを作成済みで、このセットに 1 つ以上のサブスクリプション・セット・メンバーを追加する場合は、このノートブックを使用します。サブスクリプション・セットを右クリックし、「**プロパティー**」を選択します。 • サブスクリプション・セットにメンバーを追加する。1 つのメンバーを複数のサブスクリプション・セットに追加するには、このノートブックを使用します。各メンバーは同じソースを使用する必要があります。メンバーを追加するサブスクリプション・セットを右クリックし、「**メンバーの追加**」を選択します。

System i
ADDDPRSUBM システム・コマンド
	ADDDPRSUBM コマンドを使用して、サブスクリプション・セットにメンバーを追加します。 例えば、サブスクリプション・セット・メンバーを AQHR アプライ修飾子の下の SETHR という名前のサブスクリプション・セットに追加するには、以下のようにします。
	`ADDDPRSUBM APYQUAL(AQHR) SETNAME(SETHR) SRCTBL(HR/YTDTAX)`
	`TGTTBL(TGTHR/TGTTAX)`

ソースをターゲットにマップするには、ソースとして使用する登録済みの表またはビューについて、以下の情報を指定します。

- ソース表またはビュー、およびターゲット表またはビュー (ターゲット表の表スペースと索引も含まれる)。
- ターゲット表のタイプ。
- ターゲット表に複製するソース表の登録済みの列。

  レプリケーション・センターを使用してソースをターゲットにマップするとき、LOB 列は列マッピングに自動的には組み込まれません。これらの列は明示的に選択しなければなりません。
- ターゲット表に複製するソース表の行 (行を指定するには WHERE 文節を組み込む)。

選択したソースを DB2 ターゲットに追加するには、以下のようにします。

ターゲット表またはターゲット・ビューについて、以下の情報を指定します。

- スキーマ。
- ターゲットとして使用する表またはビューの名前。

  デフォルト: デフォルト名は、ターゲット・サーバーのターゲット・オブジェクト・プロファイルがあれば、そこから参照されます。このプロファイルを設定していないと、デフォルトは TG にソース表またはビューの名前が続いたものになります。 (例えば、ソース表の名前が EMPLOYEE なら、ターゲット表のデフォルト名は TGEMPLOYEE になります。)
- ターゲット表のタイプ

  デフォルト: ユーザー・コピー

指定したターゲット表が存在しない場合は、管理ツールまたは ADDDPRSUBM システム・コマンドがターゲット表を作成します。

選択したソースを非 DB2 リレーショナル・ターゲットにマップするには、以下のようにします。

ターゲット表について、以下の情報を指定します。

- ニックネーム・スキーマ
- ニックネーム
- リモート・スキーマ
- リモート表の名前

  デフォルト: デフォルト名は、ターゲット・サーバーのターゲット・オブジェクト・プロファイルがあれば、そこから参照されます。このプロファイルを設定していないと、デフォルトは TG にソース表またはビューの名前が続いたものになります。 (例えば、ソース表の名前が EMPLOYEE なら、ターゲット表のデフォルト名は TGEMPLOYEE になります。)
- ターゲット表のタイプ

  デフォルト: ユーザー・コピー

サブスクリプション・セット・メンバーを追加するとき、ターゲット表のデフォルト・タイプであるユーザー・コピーが使用できます。あるいはレプリケーションの要求を満たす別のターゲット表タイプを選択することができます。

まだ存在しないターゲット表のサブスクリプション・セット・メンバーを追加するときは、デフォルトの設定値を使用することができます。あるいは、メンバーのプロパティーを、レプリケーションの要求を満たすように変更できます。まず、使用するターゲット表のタイプを選出してから、アプライ・プログラムがそのターゲットにデータを複製する方法に合うようにプロパティーを設定することができます。

ターゲット表タイプ

ターゲット表のタイプは、データの表示方法とレプリケーション構成に左右されます。既存の表をターゲットとして使用できますが、新規の表も作成できます。

制約事項

- 変更後イメージ・ターゲットの列の NULL 属性は、ソース表またはビューのその列の NULL 属性と互換性がなければならない。既存の列に互換性を持たせるには、SQL の COALESCE 式を使用します。
- 非 DB2 リレーショナル・データベースのソース表の場合、ターゲット表は以下のタイプのみ定義できる。
  - ユーザー・コピー表
  - ポイント・イン・タイム表
  - 外部 CCD 表
- すべての非 DB2 のリレーショナル・ターゲット表と索引の名前は、 DB2 の表と索引の命名規則に従わなければなりません。
- System i RRN 列をキー列として使用する System i のソース表の場合、ターゲット表は以下のタイプのみ定義できる。
  - ポイント・イン・タイム表
  - 外部 CCD 表
- z/OS z/OS サブシステムのソース表については、ユーザー・コピー表用のサブスクリプション・セットの where 文節を満たすために、アプライ・プログラムが CD 表および UOW 表を結合する場合、 CD 表および UOW 表のコード化スキームは同じでなければなりません。

ターゲット・タイプ

ターゲット表のタイプは以下から選択できます。

ユーザー・コピー

サブスクリプション・セット・メンバーで定義された列のみが組み込まれる読み取り専用ターゲット表です。ユーザー・コピー表の構造はソース表と同じにすることができます。あるいは、ソース列のサブセットを組み込むことができますが、この場合、変更前イメージまたは算出列が含まれるときと含まれないときがあります。 SQL レプリケーションは、ユーザー・コピー・ターゲット表へのアプリケーション書き込みのみを想定しています。エンド・ユーザーまたはアプリケーションがユーザー・コピー表を直接変更しても、 SQL レプリケーションによって上書きされることがあり、そのために

ソース表とターゲット表のデータが一致しなくなってしまうことが起こりえます。ソース表とターゲット表の両方を更新する必要がある場合は、Update-anywhere レプリケーションを使用することを考えてください。

ポイント・イン・タイム
サブスクリプション・セット・メンバーで定義された列とタイム・スタンプ列を含む読み取り専用ターゲット表です。ポイント・イン・タイム表の構造はソース表と同じにすることができます。あるいは、ソース列のサブセットを組み込むことができますが、この場合、変更前イメージまたは算出列が含まれるときと含まれないときがあります。

基礎集約
SQL 列関数 (SUM、AVG など) を使用して、ソース表の内容全体のサマリーを計算する読み取り専用ターゲット表です。

基礎集約表は、ソース表の内容を要約します。基礎集約表には、アプライ・プログラムが集約を実行したタイム・スタンプも組み込まれます。基礎集約表は、ソース表の状態を定期的にトラッキングする場合に使用します。

変更集約
SQL 列関数 (SUM、AVG など) を使用して、ソース表に加えられた最近の変更の内容 (CD 表または内部 CCD 表に保管されています) 全体に関するサマリーを計算する読み取り専用ターゲット表です。

変更集約表は、ソース表ではなく、CD 表または内部 CCD 表の内容を要約します。変更集約表には、変更がキャプチャーされた (CD 表または CCD 表に書き込まれた) 時間間隔にマークを付けるための 2 つのタイム・スタンプも組み込まれます。レプリケーション・サイクルの合間に行われた変更 (UPDATE、INSERT、および DELETE 操作) をトラッキングする場合は、変更集約表を使用します。

CCD (整合変更データ)
レプリケーション・コントロール情報用の列が追加された読み取り専用ターゲット表です。これらの列に組み込まれるのは、ログ・レコード番号 (またはジャーナル・レコード番号)、ソース表が SQL の INSERT、DELETE、または UPDATE ステートメントを使用して変更されたかどうかの標識、および挿入、削除、更新に関連したコミット・ステートメントのログ・レコード番号とタイム・スタンプです。オプションで、変更前イメージ列と UOW 表からの列を組み込むこともできます。

レプリカ
Update-anywhere レプリケーション用の読み取り/書き込みターゲット表です。ユーザー・アプリケーション・プログラムまたはユーザーが直接更新できるタイプのターゲット表は、レプリカ表のみです。したがって、レプリカ表は、マスター表から、そしてローカル・アプリケーション・プログラムまたはユーザーから変更を受け取ります。レプリカ表の構造はソース表と同じにすることができ、そうでなければ、ソース列のサブセットを組み込むことができます。しかし、追加のレプリケーション・コントロール列 (タイム・スタンプなど) は組み込みません。レプリカ表は DB2 データベースでのみサポートされています。

以下のトピックでは、ターゲット・タイプごとの使用法と、レプリケーションの要求を満たすターゲット表のプロパティーの設定方法について説明します。

読み取り専用ターゲット表

ソース・データをターゲットでどのように表示するかによって、読み取り専用ターゲット表にソース表またはソース・ビューのコピー、変更の履歴、または計算済みのサマリーが含まれるように定義できます。

以下のトピックには、これらのタイプの読み取り専用ターゲットの詳細について記述されています。

ユーザー・コピーとポイント・イン・タイムのターゲット:

デフォルトでは、サブスクリプション・セット・メンバーを定義するときに、ユーザー・コピー表がターゲット・タイプとして作成されます。ターゲットに変更が適用されたポイント・イン・タイムをトラッキングするには、ポイント・イン・タイムをターゲット・タイプとして選択します。

ユーザー・コピー
: コピー時にターゲット表をソース表に突き合わせる場合は、このデフォルト・タイプを使用します。ユーザー・コピー表に追加のレプリケーション・コントロール列は含まれませんが、ソース表の行または列のサブセット、あるいは複製されない追加の列を入れることができます。

ポイント・イン・タイム
: ターゲットに変更が適用されたポイント・イン・タイムをトラッキングする場合は、ポイント・イン・タイムをターゲット・タイプとして選択します。ポイント・イン・タイム・ターゲットにはソース表と同じデータが含まれ、アプライ・プログラムがそれぞれの行をいつターゲットにコミットしたかを知らせるために追加されたタイム・スタンプ列もあります。タイム・スタンプ列は、当初は NULL です。ポイント・イン・タイム表には、ソース表の行または列のサブセット、あるいは複製されない追加の列を入れることができます。

制約事項: DB2 は、AS IDENTITY GENERATED ALWAYS と定義されている DB2 表の列には、値が挿入されないようにします。この制約を回避するには、次のようにすることができます。

- IDENTITY CLAUSE が指定されていないターゲット表を作成する
- AS IDENTITY GENERATED BY DEFAULT が指定された列を持つターゲット表を作成する

基礎集約または変更集約のターゲット:

ソース表の内容全体またはソース表データに加えられた最新の変更のサマリーを含むターゲット表を作成することができます。

集約ターゲット表タイプでは、COUNT、 SUM、MIN、MAX、AVG などの集約 SQL 列関数を使用して、ターゲットの列を定義できます。これらの列には、オリジナルのソース・データは含まれません。含まれるのは、定義した SQL 関数の算出値です。アプライ・プログラムはフル・リフレッシュの間、集約を作成しません。行は、アプライ・プログラムがセットを処理する経過において付加されます。集約表を使用すると、SQL レプリケーションが 1 行 1 行ではなくサマリー情報だけをレプリケーションできるため、ネットワーク帯域幅とターゲット表内のスペース両方を節約できるという利点があります。

基礎集約のターゲット

基礎集約ターゲット表を使用して、各レプリケーション・サイクル内でソース表の状態をトラッキングします。基礎集約ターゲット表では、アプライ・プログラムがソース表から集約します (読み取って計算を実行します)。基礎集約表には、アプライ・プログラムが集約を実行したタイム・スタンプも組み込まれます。

基礎集約表のみが登録済みソース表のターゲットである場合、ソース表への変更をキャプチャーする必要はありません。

例: 週ごとの平均カスタマー数を知りたいとします。ソース表にカスタマーごとの行があれば、アプライ・プログラムが週ごとにソース表の行数を合計し、結果を基礎集約表に保管します。集約を毎週実行すると、ターゲット表にはその年の週ごとのカスタマー数を示す 52 の項目が含まれるようになります。

変更集約のターゲット

レプリケーション・サイクルの合間にソース表で行われた変更
(UPDATE、INSERT、および DELETE 操作) をトラッキングする場合は、変更集約表を使用します。変更集約ターゲット表では、アプライ・プログラムが CD 表または内部 CCD 表から集約します (読み取って計算を実行します)。変更集約表には、キャプチャー・プログラムが CD 表または CCD 表に変更を挿入した時間間隔にマークを付けるための 2 つのタイム・スタンプも組み込まれます。

例: 毎週新たに獲得したカスタマー数 (INSERT) と、失った既存のカスタマー数 (DELETE) を知りたいとします。 CD 表の挿入された行と削除された行の数を週ごとに数え、その数を変更集約表に保管します。

重要: サブスクリプション・セット・メンバーのソース表がフル・リフレッシュのみのレプリケーション用に登録済みの場合、ソースで CD 表または CCD 表が必須である変更集約ターゲット表を持つことはできません。

CCD ターゲット:

ソース・データの監査を行ったり、データの使用状況の履歴を保持したりすることが必要な場合もあります。ターゲット・タイプとして整合変更データ (CCD) 表を使用することで、ソースの変更に関する履歴を追跡管理できます。

例えば、データの変更が発生したときの変更前後の比較や、ソース表への更新を行ったユーザー ID をトラッキングできます。

ソース表の履歴を保持する読み取り専用ターゲット表を定義するには、以下の属性を持つようにターゲット CCD 表を定義します。

非コンデンス
: ソースの変更すべてに関するレコードを保持するには、CCD 表を非コンデンスに定義し、発生した変更ごとに 1 行が保管されるようにします。非コンデンスの表には同じキー値を持つ複数の行が含まれるため、ユニーク索引は定義しないでください。非コンデンス CCD 表は、 UPDATE、INSERT、または DELETE 操作ごとに 1 行を保有することで、ソース表に対して実行された操作の履歴を保留します。 UPDATE 操作を INSERT および

DELETE 操作 (パーティション化キー列用) としてキャプチャーする場合、CCD 表は、それぞれの更新ごとに 2 つの行、すなわち DELETE に 1 行、INSERT に 1 行を指定します。

コンプリートまたは非コンプリート

CCD 表をコンプリートにするか、あるいは非コンプリートにするかを選択できます。未完成の CCD 表にはソース行のコンプリート・セットが最初含まれていないため、ソース表への更新 (アプライ・プログラムが CCD 表の移植を開始してからの更新) 履歴を保持する未完成の CCD 表を作成します。

UOW (作業単位) 列の組み込み

監査機能の改善のため、UOW 表からの追加の列を組み込んでください。ユーザー指向の識別がさらに必要であれば、UOW 表で、DB2 for z/OS の相関 ID に関する列、 1 次許可 ID、または System i のジョブ名およびユーザー・プロファイルを使用することができます。

内部 CCD ターゲット:

ソース表で変更が頻繁に発生する場合、最後のアプライ・サイクル以降にソースで発生したコミット済みの変更を要約するために、内部 CCD 表を作成できます。

キャプチャー・プログラムがログからの変更を追加するとき、CD 表は常に流動的であるため、 CCD のソース変更のローカル・キャッシュが、ターゲット用のより継続的なソースとして機能します。

オリジナルのソース表が更新されるとき、キャプチャー・プログラムはソースのログでの頻繁な変更を読み取り、その変更をソースの CD 表に追加します。その CD 表から、アプライ・プログラムは CD 表での変更を読み取り、内部 CCD 表に移植します。内部 CCD 表を、最終のサイクル内で発生した CD 表のそれぞれの行ごとの最新の変更のみが含まれるように定義できます。したがって、CCD 表はアプライ・サイクルの合間では静的であるため (CD 表から CCD 表に複製するアプライ・プログラムの場合)、ターゲット用のより継続的なソースになります。ソースからのコンデンス変更を行うことで、同じ行の多数の変更をターゲット表に複製しなくなるため、レプリケーションのパフォーマンス全体が改善されます。

キャプチャー・プログラムは新しい変更を常に CD 表に追加しているため、 2 番目のアプライ・プログラムは CD 表の代わりに内部 CCD 表から変更を読み取ります。そのため、様々な変更を様々なターゲットに複製することはなく、ターゲット相互の同期がとれた状態を保つことができます。 2 番目のアプライ・プログラムは、オリジナルのソース表をフル・リフレッシュのために使用し、内部 CCD 表を変更キャプチャー・レプリケーションのために使用します。

重要 (Update-anywhere の場合): 内部 CCD 表を定義すると、アプライ・プログラムはレプリカをターゲットとしてサブスクリプション・セットを処理するときにそれを無視します。マスター・ソースの CD 表からレプリカへの変更を適用します。

推奨事項

- ソース表と内部 CCD 表との間でサブスクリプション・セット・メンバーを定義するのは、ソース表と他のターゲット表との間で他のサブスクリプション・セット・メンバーを定義する前に行うこと。そうするとアプライ・プログラムは、ソ

ース表から変更を複製するために CD 表ではなく内部 CCD 表を使用するようになります。ソース表に内部 CCD 表を定義する前に他のサブスクリプション・セット・メンバーを定義し、それらのメンバーを使用してレプリケーションを開始すると、ソース表のすべてのターゲットに対してフル・リフレッシュの実行が必要になる可能性があります。

- すべての内部 CCD 表を 1 つのサブスクリプション・セットに結合して、ソース・データベースのすべてのターゲット表相互の同期が必ずとれるようにする。
- 頻繁に変化するソース列のサブセットを他のターゲットに適用するだけの場合でも、すべての登録済みソース列が内部 CCD に複製されるというデフォルトを使用する。そうすると、オリジナルのソース表の他の登録済み列のデータを今後必要とする可能性のあるターゲット表のために、内部 CCD 表をソースとして使用できます。今後のどのようなターゲットへの変更キャプチャー・レプリケーションにおいても使用可能なのは、内部 CCD 表の列のみです。

内部 CCD 表の属性

内部 CCD 表は、レプリケーションにおける暗黙的なソースとして使用します。レプリケーション・ソースとして明示的に定義することはできません。サブスクリプション・セット・メンバーを追加する場合、オリジナルのソース表 (内部 CCD 表ではありません) をターゲット表にマップします。内部 CCD 表には以下の属性があります。

内部
CCD 表はソースの CD 表の代わりとして機能します。内部 CCD 表についての情報は、IBMSNAP_REGISTER 表で、この内部 CCD 表のソース表と同じ行に保管されます。内部 CCD 表独自の行は、登録表内にはありません。アプライ・プログラムは、CD 表からではなく、内部 CCD 表 (存在する場合) から変更を自動的に複製します。それぞれのレプリケーション・ソースごとに、内部 CCD 表は 1 つだけ存在できます。

制約事項: ユーザー表には算出列は含まれないため、 CCD サブスクリプションに算出列を組み込まないでください。

ローカル
CCD 表はソース表と同じデータベース内にあります。

非コンプリート
アプライ・プログラムは、フル・リフレッシュのために内部 CCD ではなくオリジナルのソース表を使用するため、後続ターゲットにすべてのソース行の初期コピーがすでに含まれるという理由から、CCD は非コンプリートです。

コンデンス
内部 CCD はコンデンスされています。すなわち、その表にはキー値ごとに 1 行が含まれ、アプライ・プログラムは変更ごとに 1 行をアプライする代わりに、 CCD 表のそれぞれの行ごとに最新の変更をアプライするようになります。

UOW 列なし
内部 CCD 表では、UOW 表の列の追加はサポートされません。 UOW 列を組み込んでいるターゲット CCD 表をすでに定義している場合は、内部 CCD 表を使用できません。

複数層構成における中間層の定義

基本のレプリケーション・モデルは、ソースが 1 つでターゲットが 1 つ以上の 2 層モデルです。3 つ以上の層で構成をセットアップすることもできます。

制約事項

複数層構成内の中間層は、DB2 表でなければなりません。

このタスクについて

複数層構成にはソース表とターゲット表が含まれ、そのターゲット表は他のターゲット表にはソースとして機能します。

複数層レプリケーション環境をセットアップする 1 つの理由は、分散のオーバーヘッドをソース・システムから 2 番目のシステムに移動することです。しかも、ソース・システムへのデータベース接続の多くを回避できるので、接続コストを第 2 層に移動できます。さらに、変更を層 1 から層 2 の CCD 表に収集できるため、各層への変更を複製する頻度をコントロールでき、ターゲット (層 3) に複製される変更の数を削減できます。

例えば、3 層モデルで、第 1 層 (層 1) はソース・データベース、第 2 層 (層 2) は層 1 のターゲットです。しかも層 2 はターゲットを含む第 3 層 (層 3) のソースであり、変更を 1 つまたは多数の層 3 のデータベースに分散できます。レプリケーション構成内に 2 つより多い層が含まれる場合、ソースとターゲットの両方として機能する中間層は CCD 表です。

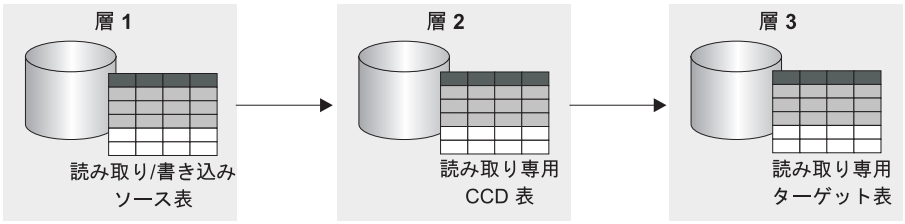


図 6. 3 層レプリケーションのモデル : データを、ソース表からターゲット表に、さらにその表から別のターゲット表にレプリケーションできます。

この手順はレプリカ表にも適用されます。 CCD 表は通常読み取り専用レプリケーションに使用されますが、レプリカ表は Update-anywhere レプリケーションに使用されます。

手順

複数層レプリケーションをセットアップして、ターゲット表が後続のターゲットのソースになるように構成するには、以下のようにします。

1. レプリケーションのソース表 (層 1) を登録します。 このソースにおいて、キャプチャー・プログラムは層 1 で発生する変更をキャプチャーし、それを層 1 の CD 表に保管します。
2. ソース・サーバーとターゲット・サーバー (層 2) の間のサブスクリプション・セットを作成します。 このサブスクリプション・セットにおいてアプライ・プログラムは、層 1 から層 2 の CCD 表に変更を適用します。

3. ソース表 (層 1) と CCD ターゲット表 (層 2) をマップするサブスクリプション・セット・メンバーを定義します。

   このメンバーにターゲット表を定義するとき、以下の属性を持つ CCD 表になるようなターゲット表を選択します。

   外部登録済みソース

   ソースを外部ターゲット表として定義しなければならず、この表が後続の層のためのソースとして機能できるように登録しなければなりません。他の登録済みソースと同様に、外部 CCD 表にも独自の行が IBMSNAP_REGISTER 表内にあります。ソースとしても機能可能な外部 CCD 表は、ただ 1 つのソース表によってのみ移植されます。

   すべての外部 CCD 表は、同じキャプチャー・スキーマを使用してサブスクリプション・セットに登録しなければなりません。

   変更データ (CD) 表と IBMSNAP_UOW 表を結合させることなく、外部 CCD 表にレプリカを生成できます。新規表タイプは、IBMSNAP_SUBS_MEMBR 表の TARGET_STRUCTURE 列の値 9 で指定されます。タイプ 9 CCD 表には IBMSNAP_LOGMARKER 列が含まれますが、アプライ・プログラムは、この列のソース・コミットのタイム・スタンプの入手に CD 表と IBMSNAP_UOW 表の結合を必要としません。むしろ、アプライ・プログラムは、IBMSNAP_LOGMARKER 列の同じ値を同じサイクル内のすべての行に対して生成します。新規 CCD 表タイプの構造は、タイプ 3 CCD 表と同じになります。表には、ユーザー列に加えて、以下の 4 つの IBM® 必須列が含まれます。

   ```
   IBMSNAP_COMMITSEQ
   IBMSNAP_INTENTSEQ
   IBMSNAP_OPERATION
   IBMSNAP_LOGMARKER
   user_columns
   ```

   このターゲット表タイプは、3 層レプリケーション構成のソース表として登録することができます。

   重要: タイプ 9 CCD 表の場合、データ・ブロッキング因数 (IBMSNAP_SUBS_SET コントロール表の MAX_SYNCH_MINUTES) を設定解除 (NULL) する必要があります。

   コンプリート

   後続の層においてアプライ・プログラムがフル・リフレッシュと変更キャプチャー・レプリケーションの両方の実行に使用するため、完全な CCD 表を使用しなければなりません。

   コンデンス

   コンデンスされた CCD を使用します。すなわち、この表にはキー値ごとに 1 行が含まれ、必ず最新の変更のみが後続の層に複製されます。アプライ・プログラムは変更ごとに 1 行を適用する代わりに、CCD 表のそれぞれの行ごとに最新の変更を適用します。コンデンスされた表はそれぞれの行ごとにユニーク・キー値が必須であるため、ユニーク索引を定義しなければなりません。

4. CCD 表が登録されるので、中間層のデータベースにキャプチャー・コントロール表がまだない場合は、キャプチャー・コントロール表を作成します。

5. 登録済み CCD 表が含まれる層 2 のサーバーと (層 3 の) 後続のターゲット・サーバーとの間で、サブスクリプション・セットを作成する。 このセットにおいてアプライ・プログラムは、CCD 表から後続の層のターゲット表に変更を適用します。アプライ・プログラムは、CCD 表をフル・リフレッシュと変更キャプチャー・レプリケーションの両方に使用します。アプライ修飾子は通常、CCD にデータを入れるのに使った修飾子とは別のものを使用しますが、同一のものを使用することも可能です。
6. CCD ソース表 (層 2) と後続のターゲット表 (層 3) をマッピングするサブスクリプション・セット・メンバーを定義する。 この CCD ソース表をサブスクライブするターゲット表を組み込んで、複数のメンバーをセットアップできます。これが複数層構成の最終層であれば、ターゲット表のタイプは何でも可能です。ただし、計画している層が 3 つより多い場合は、層 3 のターゲット表をステップ 3 で示されているように定義し、ステップ 4 から 5 を繰り返して後続の層を追加します。

重要: フル・リフレッシュが外部 CCD (中間層) で発生した場合、ソースとしてその外部 CCD を使用する、すべての後続の層におけるアプライ・プログラムはフル・リフレッシュを実行します。これをカスケード・フル・リフレッシュと呼びます。

読み取り/書き込みターゲットの定義 (Update-anywhere)

Update-anywhere レプリケーションでは、マスター・ソース表における変更は従属ターゲット表に複製され、さらにそのレプリカ表における変更はマスター・ソース表に逆に複製されることが可能です。

始める前に

- 単一アプリケーション・プログラムはマスター表とレプリカ表の両方を更新しないため、参照整合性制約を宣言して使用しなければならない。参照整合性違反はアプリケーションのロジックでは検出できません。
- 参照整合性違反を防止するため、複数のマスター表を対象に存在するすべての参照制約をレプリカ表に含めなければならない。いくつかの参照制約を省略すると、レプリカ表に加えられた更新がマスター表に複製されるとき、参照整合性違反を引き起こす可能性があります。管理ツールは、ソース表からターゲット表への参照制約定義のコピーを行わず、新しい制約を生成することもできません。
- フル・リフレッシュ中の参照整合性チェックをバイパスするには、ASNLOAD 出口ルーチンを使用しなければならない。

制約事項

- リモート・ジャーナル構成ではレプリカ・ターゲット表タイプはサポートされません。
- Update-anywhere レプリケーションで、CCD 表をソースまたはターゲットとして使用することはできない。
- LOB データ・タイプの列を Update-anywhere レプリケーションの対象にできるようにするには、登録表の CONFLICT_LEVEL を 0 に設定しなければならない。
- 非 DB2 データベースのタイプをレプリカ・ターゲット表にすることはできない。そのため、Update-anywhere レプリケーションの対象にできません。

このタスクについて

Update-anywhere レプリケーションで、マスター表とそのレプリカのすべては、ソースとターゲットの両方として機能する読み取り/書き込み表です。

手順

マスター表と 1 つ以上のレプリカ表との間で (それぞれのレプリカ表が別々のデータベースにある場合) Update-anywhere 構成をセットアップするには、以下のようにします。

1. レプリカ表を入れるデータベースそれぞれにキャプチャー・コントロール表がまだない場合は、キャプチャー・コントロール表を作成する。
2. レプリケーションのためにソース表を登録する (マスター表) 。
3. 1 つ以上のレプリカを入れるマスター・データベースとターゲット・データベースとの間に、サブスクリプション・セットを作成する。

   すべてのレプリカ表が同じデータベースに、そしてすべてのマスター表が別のデータベースに含まれている場合、サブスクリプション・セットは 1 つだけ必要です。レプリカ表が複数のデータベースに含まれている場合、保有するレプリカ・データベースと同数のサブスクリプション・セットが必要です。
4. マスター表とそのおのおのに関連したレプリカ表との間のマッピングごとに、サブスクリプション・セット・メンバーを定義する。

   この構成では、レプリカ表が含まれたサーバーで通常実行されるアプライ・プログラムは 1 つしかありません。このセットにおけるアプライ・プログラムは、マスターの CD 表から変更をプルして、それをレプリカ表に適用します。また、アプライ・プログラムはレプリカ表の CD 表から変更をプッシュして、それをマスター表に適用します。

   重要: Update-anywhere 構成において、マスター表とレプリカ表はデータを往復させて互いに複製するため、レプリカ・ターゲット表にはソース表と同じ列が含まれている必要があります。脱落した列がマスター・サイトで NULL 可能、または NOT NULL WITH DEFAULT として定義されている場合に限り、マスター表の列のサブセットが含まれるレプリカ・ターゲットを作成できますが、レプリカで新しい列の追加または列の名前変更は行わないでください。
5. レプリカ表にソースのプロパティーを定義する。 レプリカ表を組み込んでサブスクリプション・セット・メンバーを作成するとき、 SQL レプリケーションはそのレプリカ表をレプリケーション・ソースとして自動的に登録します。レプリカ・ターゲット表はソースとして機能するため、ターゲット表に共通するプロパティーとは別に設定可能なプロパティーがあります。これらのプロパティーにより、キャプチャー・プログラムがレプリカへの変更を処理する方法が決定されます。ただし、マスター表から継承されても、レプリカ表に合うように変更できないプロパティーが 2 つあります。競合検出レベルとフル・リフレッシュの可否のプロパティーです。このソースにおいて、キャプチャー・プログラムはレプリカ表での変更をキャプチャーし、それをレプリカの CD 表に保管します。

   重要: マスターとレプリカがソースとターゲットの両方として機能するとしても、フル・リフレッシュ・コピーは、マスターからレプリカへのみ行われ、レプリカからマスターへは行われません。

競合を防止するため、レプリカ表のターゲット・キーを、マスター・ソース表の主キーまたはユニーク索引と同じにしなければなりません。マスター表はレプリカを更新でき、レプリカはマスターを更新できるため、アプライ・サイクルの合間にマスター表のある行が更新され、 1 つ以上のレプリカ表の同じ行にそれとは異なる更新が加えられた場合 (変更がマスター CD 表とレプリカ CD 表で行われます)、競合が発生する可能性があります。レプリカ表は競合検出レベルをマスター・ソース表またはビューから継承します。最善なのは、マスターからすべてのレプリカ表へデータを複製するときに、競合が決して発生しないようにアプリケーションを設計することです。マスター・ソースを登録したとき、競合検出のレベルを 3 つの選択肢から選びました。

ソース表における参照整合性制約を定義した場合、保全違反を防止するために、同じ参照整合性制約をレプリカ表にも定義しなければなりません。参照整合性違反が発生した場合、サブスクリプション・サイクルは自動的に再試行されます。

既存の表をターゲット表として使用する

SQL レプリケーションの外部で定義されている既存のターゲット表を組み込むサブスクリプション・セット・メンバーを定義できます。

このようなユーザー定義のターゲット表は、表の構造が有効である限り、レプリケーションで有効なターゲット表タイプ (ユーザー・コピー、ポイント・イン・タイム、基礎または変更集約、CCD、あるいはレプリカ) のいずれかになります。例えば、ユーザー定義のポイント・イン・タイム表には、 IBMSNAP_LOGMARKER という TIMESTAMP タイプの列が組み込まれなければなりません。

要件

- サブスクリプション・セット・メンバーの定義に含まれる列が既存のターゲット表より少ない場合、レプリケーションに関連するターゲット表の列は NULL が許可されているか、あるいは NOT NULL WITH DEFAULT として定義されていなければならない。
- ポイント・イン・タイム CCD 表、ユーザー・コピー CCD 表、レプリカ CCD 表、およびコンデンス CCD 表にはユニーク索引が必要である。既存のターゲット表を使用してサブスクリプション・セット・メンバーを定義するときは、既存のユニーク索引を使用するか、または新規のユニーク索引を指定することができます。

制約事項

- サブスクリプション・セット・メンバーの定義に、既存のターゲット表より多くの列を含めることはできない。
- レプリケーション・センターを使用する場合、該当する列がターゲット表にまだ存在していなければ、サブスクリプション・セット・メンバーにその列を追加できない。

レプリケーションは、既存のターゲット表とサブスクリプション・セット・メンバー定義との間の矛盾をチェックします。

重要 (複数層の場合): ソース表が層 1、CCD 表が層 2、そして既存の表が層 3 であるような複数層構成をセットアップする場合、層 1 と層 2 との間でサブスクリプション・セット・メンバーを定義するときに、 CCD 表を既存のターゲット表に

指定されている属性に一致するように定義します。そして、この CCD 表をソース表とする既存のターゲット表のサブスクリプション・セット・メンバーを定義します。

すべてのターゲット表タイプに共通のプロパティー

ターゲット表を作成するときに、タイプに関係なく、それぞれのレプリケーション環境に基づいてプロパティーを設定できます。

以下のトピックでは、ソース・データをターゲット表にマップする方法に関して定義できる共通の特性について説明しています。

ソース列のサブセットの複製

デフォルトでは、LOB 列以外のすべての登録されたソース列がターゲット表に含まれます。すべての列を複製しない方がよい場合や、ターゲット表がソースで定義したすべてのデータ・タイプをサポートしていない場合もあります。

この場合は、ターゲット表に複製するソース列のみを選択します。選択しないソース表の登録済みの列は、他のサブスクリプション・セット・メンバーに引き続き使用できますが、現行のソースからターゲットへのマッピングには組み込まれません。

算出列をターゲット表に追加することもできます。これらの列は、SUBSTR のような SQL スカラー関数で定義するか、あるいは列 A の値を列 B の値で除算する (colA/colB) ような派生列にすることができます。これらの算出列は、ソース表のすべての列を参照します。

ソース行のサブセットの複製

デフォルトでは、ターゲット表にはソース表のすべての行が含まれます。すべての行を複製しない方がよい場合があります。つまり、異なる種類のデータを含む行を別のターゲット表に複製できるということです。

一定の条件 (SQL WHERE 文節) と一致する行が含まれる行 (水平方向) サブセットをサブスクリプション・セット・メンバーに定義できます。

SQL の述部には、通常 ID または区切り ID を入れることができます。 WHERE 文節についての詳細は、「*DB2 SQL リファレンス*」を参照してください。

例えば、社内のある 1 部門に関するすべての行を複製する WHERE 文節を定義できます。または、すべての LOB 列 (および主キー列) を 1 つのターゲット表にレプリケーションする WHERE 文節をサブスクリプション・セット・メンバーに定義し、その他の列すべてを別のターゲット表にレプリケーションする WHERE 文節を別のサブスクリプション・セット・メンバーに定義できます。このようにして、ターゲット・データベースはソース表からのデータすべてを保有できますが、ターゲット・データベースのソース表を非正規化して、データウェアハウスへの照会パフォーマンスを調整します。

行述部の制約事項

- この文節に WHERE を入力しない。これは暗黙で指定されます。副選択ステートメントの文節にのみ WHERE を入力してください。

- 文節をセミコロン (;) で終わらせない。
- WHERE 文節にブール式 OR を含める場合、例えば、(COL1=X OR COL2=Y) のように、述部を括弧で囲む。
- ターゲット表が変更集約表であって、変更前イメージ列を含んでいる場合、変更前イメージ列を GROUP BY 文節に含めなければならない。

例

次に示す例には、ターゲット表の行をフィルターに掛けるために使用できる WHERE 文節が含まれます。これらの例は一般的なものであり、モデルとして使用するためのものです。

特定の値を持つ行を指定する WHERE 文節

特定の値 (例えば、管理職にある社員を表す MGR) をもつ行だけをコピーするには、次のような WHERE 文節を使用します。

```
EMPLOYEE = 'MGR'
```

ある範囲の値を持つ行を指定する WHERE 文節

ある範囲の値 (例えば従業員番号 5000 ～ 7000) の行をターゲット表にコピーするには、以下のような WHERE 文節を使用します。

```
EMPID BETWEEN 5000 AND 7000
```

ソース列からターゲット列にマップする方法

デフォルトでは、SQL レプリケーションによって作成されたターゲット表の列名は、ソース表の列名と一致します。ほとんどのターゲット列は、名前とデータ長を変更しても、ソース列にマップできます。

レプリケーション・コントロール列 (IBMSNAP または IBMQSQ で始まる) を除くターゲット表のすべての列の名前を変更できます。ターゲット表が存在していれば、レプリケーション・センターは列を名前でマップします。

ターゲット表の列を、ソース列とは異なる長さにすることができます。ターゲット列がソース列より短い場合、サブスクリプション・セット・メンバーの式を使用して文字を長い列から短い列にマップしたり、その式を含むビューを登録できます。例えば、ソース列が char(12) でターゲット列が char(4) である場合、次の式を使ってレプリケーション中に COL1 からの値を切り捨てることができます。

```
substr(col1, 1,4)
```

ターゲット列名が長い場合は、ターゲット列にブランクを埋め込みます。

注: DB2 for Linux, UNIX, and Windows から DB2 for z/OS と DB2 for i5/OS® の両方に対して LONG VARCHAR 列をマップする場合は、いくつかの制約事項があります。

レプリケーション・センターの使用

レプリケーション・センターを使用してターゲット表を作成する際に、ターゲット表のタイプに関係なく、ターゲット表で列の名前を変更できます。さらに、列の属性 (データ・タイプ、長さ、位取り、精度、NULL 可能かどうか) に互換性があれば、それらを変更することもできます。

レプリケーション・センターを使用して、既存のターゲット表の列を名前変更することはできません。ソース列とターゲット列が一致しない場合は、レプリケーション・センターを使用してソースからターゲットに列をマップするか、あるいはソースの列名に一致する名前を含むターゲット表のビューを作成するかのどちらかが可能です。

非 DB2 リレーショナル表に対するマッピング

DB2 表を、既存のニックネームを持つ非 DB2 リレーショナル表にマップする場合、一部の列のデータ・タイプに互換性がないことがあります。ソース列のデータ・タイプにターゲット列のデータ・タイプとの互換性がない場合、以下のようにターゲットでデータ・タイプを変更して、ソースとの互換性を持たせることができます。

- ソースのデータ・タイプがターゲットで必要とするデータ・タイプに一致するように調整するための算出列を追加できる。
- 非 DB2 リレーショナル・ターゲット表のニックネームを変更して、データ・タイプ変換を変更できる。

例: データ・タイプが DATE の DB2 列を含む DB2 ソース表から、データ・タイプが DATE の Oracle 列を含む Oracle ターゲット表にデータを複製するとします。

表 4. DB2 DATE 列の Oracle DATE 列へのマッピング

DB2 列	ニックネーム・データ・マッピング	Oracle 列
A_DATE DATE	A_DATE TIMESTAMP A_DATE DATE	A_DATE DATE

Oracle データ・タイプ DATE を含む Oracle ターゲット表が作成されます (日付データとタイム・スタンプ・データの両方が入ります)。フェデレーテッド・データベースにある Oracle の DATE データ・タイプの初期のニックネームは、 DB2 データ・タイプを TIMESTAMP としてマップします。DB2 レプリケーション・センターとレプリケーション用の System i コマンドは、ニックネームのデータ・タイプを DATE に変更するので、 DATE は Oracle に複製されますが、TIMESTAMP はされません。

ターゲット・キー

コンデンス・ターゲット表が変更キャプチャー・レプリケーションに関係している場合、アプライ・プログラムは、そのターゲット表にターゲット・キー と呼ばれる主キーまたはユニーク索引がなければなりません。

ターゲット表のユニーク索引として使用する列は選択できます。以下のタイプのターゲット表はコンデンスされており、ターゲット・キーが必須です。

- ユーザー・コピー
- ポイント・イン・タイム
- レプリカ
- コンデンス CCD

ターゲット表を新規に作成する場合は、デフォルトの索引名とスキーマを使用できますが、デフォルトをユーザーの命名規則に一致するように変更することもできます。

デフォルト名は、ターゲット・サーバーのターゲット・オブジェクト・プロファイルがあれば、そこから参照されます。このプロファイルを設定していないと、デフォルトは「IX + ターゲット表の名前」となります。例えば、ターゲット表の名前が TGEMPLOYEE なら、ターゲット表の索引のデフォルト名は IXTGEMPLOYEE になります。

ユニーク索引のオプション

ユニーク索引を作成するためのオプションは、新しいターゲット表を作成するのか、それとも既存のターゲット表を使用するのかによって異なります。

新しいターゲット表

新しいターゲット表のユニーク索引を作成するためのオプションが 2 つあります。

- ターゲット表のユニーク索引として使用する列を指定する。
- SQL レプリケーションでユーザー向けのユニーク索引を選択する。

  ユニーク索引用の列を選択しない場合、 SQL レプリケーションではソース表に以下の定義のいずれかがあるかどうかをこの順序でチェックします。

  1. 主キー
  2. ユニーク制約
  3. ユニーク索引

  SQL レプリケーションで、ソース表のこれらの定義のいずれかが検出され、関連した列が登録されていてターゲット表の一部であることが検出された場合、 SQL レプリケーションはソース表の主キー (またはユニーク索引か RRN) をターゲット・キーとして使用します。ユニーク制約の場合、SQL レプリケーションは制約列を使用して、ターゲット表のユニーク索引を作成します。

  System i 主キーまたはユニーク索引を持たない System i ソース表の場合は、相対レコード番号 (RRN) を一意性の因子として使用するように表の登録を変更します。サブスクリプション・セット・メンバーを定義するときに、RRN 列をターゲット表のユニーク索引として指定します。

  System i ターゲット・キーとして RRN を使用する System i のターゲット表の場合、これらのターゲット表を複製するには、アプライ・プログラムを System i で実行します。

既存のターゲット表

既存のターゲット表の場合、ユニーク索引を選択する必要があります。以下のオプションのいずれかを選択することができます。

- ターゲット表の既存の索引を使用する。

既存の索引を使用するには、レプリケーション・センターで、索引を表す列を選択します。レプリケーション・センターが厳密に一致するものを検出すれば、アプライ・プログラムが使用するようにターゲット・キーを設定するだけですが、検出しなかった場合は、ユニーク索引を作成してから、ターゲット・キーをアプライ・プログラムが使用するように設定します。

- ターゲット表の別の索引を作成する。

  ユニーク索引がまだない場合は作成され、アプライ・プログラムが使用するようにターゲット・キーが設定されます。

重要: ソース表で更新される可能性のある列が組み込まれたターゲット表のキーを選択する場合、ターゲット・キー列に特殊な更新を加えるようアプライ・プログラムに指示しなければなりません。

アプライ・プログラムがターゲット・キー変更オプションを使用してターゲット・キー列を更新する方法

サブスクリプション・セット・メンバーを定義するときにターゲット・キー変更オプションを選択すると、ターゲット・キーの変更時、アプライ・プログラムはターゲット・キー列に特殊な更新を加えます。

前提条件

アプライ・プログラムでターゲット・キー列を更新するためには、ターゲット・キーの一部になっているソース列を、CD (または CCD) 表の変更前イメージ列に登録する必要があります。ターゲット・キーを構成する列の変更前イメージ値をキャプチャーするためのソース登録を定義していない場合は、異なるキーを持つターゲット表をサブスクライブする前に、登録を変更して変更前イメージ値を組み込まなければなりません。

制約事項

- 更新を削除および挿入のペアとしてキャプチャーするために登録されたソース表において、ターゲット・キー変更オプションを使用することはできません。
- アプライ・プログラムが、ターゲット・キー列の変更前イメージを基にして、ターゲット表を更新する場合、ソース表内の式を、ターゲット表内のキー列にマップすることはできません。(すなわち、IBMSNAP_SUBS_MEMBR 表の TARGET_KEY_CHG 列にそのターゲット表に対する値 Y がある場合です。)

ターゲット・キー列の変更前イメージ値が CD (または CCD) 表にあることを確認後、ターゲット・キー列の更新時にアプライ・プログラムが変更前イメージ値を使用するように、サブスクリプション・セット・メンバーのオプションを選択します。

ターゲット・キー列の更新時にアプライ・プログラムが変更前イメージ値を使用するように指定しない場合、ターゲット・キーの一部となっているソース表の列を更新しても、SQL レプリケーションはデータを正しく複製しません。

アプライ・プログラムは、新しい値を使用してターゲット表の行の更新を試みますが、更新のための新しいキー値がターゲット表で検出されません。そこでアプラ

イ・プログラムはこの更新を INSERT に変換して、新しいキー値をターゲット表に挿入します。この場合、古いキー値をもつ古い行はターゲット表内に残ります。これは不要な行です。

ターゲット・キー列が変更前イメージ値を使用して処理されるような変更を指定すると、アプライ・プログラムは古いキー値が指定された行を検出でき、その行を新しい値で更新します。例えば、*target_key_chg* 変数が N に設定されている場合、更新操作の SQL ステートメントは次のようになります。

```
UPDATE targettable SET <non-key columns>= after-image values
WHERE <key columns> = after-image values
```

target_key_chg 変数が Y に設定されている場合は、更新操作の SQL ステートメントは次のようになります。

```
UPDATE targettable SET <all columns> = after-image values
WHERE <key columns> = before-image values
```

第 6 章 SQL レプリケーションでの特殊なデータ・タイプのレプリケーション

LOB、ROWID、または DB2 以外のデータ・タイプなど、特殊なデータ・タイプを複製するときには、一定の条件および制約事項がありますので注意してください。場合によっては、SQL レプリケーションでこれらのデータ・タイプを処理できるように追加のセットアップ・ステップを実行する必要があります。

以下のトピックでは、特殊なデータ・タイプのレプリケーションについて説明します。

レプリケーションにおける一般的なデータの制約事項

SQL レプリケーションでは、特定のデータ・タイプについて、データ暗号化の制約事項やデータ・タイプの制約事項などがあります。

データ暗号化の制約事項

SQL レプリケーションでは、いくつかのタイプの暗号化データを複製できます。

EDITPROC

SQL レプリケーションでは、データ・セキュリティーを強化するために、 DB2 for z/OS の編集ルーチン (EDITPROC) で定義されたソース表をサポートしています。その表をレプリケーションのソースとして使用するには、その表を含んでいる DB2 サブシステムがバージョン 8 (APAR PK13542) でなければなりません。

DB2 for Linux, UNIX, and Windows の暗号化スカラー関数

DB2 for Linux, UNIX, and Windows では、暗号化スカラー関数を使用して、列データの暗号化と暗号化解除を実行できます。この機能をレプリケーションで使用するには、ソースのデータ・タイプが VARCHAR FOR BIT DATA でなければなりません。ソースとターゲットが同じコード・ページを使用していて、暗号化解除機能を使用できる状況であれば、そのデータを正常に複製できます。暗号化されたデータが入っている列のレプリケーションは、DECRYPT_BIN 機能または DECRYPT_CHAR 機能をサポートしているサーバーでのみ使用するべきです。

データ・タイプの制約事項

SQL レプリケーションでは、以下のデータ・タイプを複製できません。

- DB2 以外のリレーショナル・ソースからの LOB 列
- 以下のいずれかのプロシージャーが定義されている場所の列
  - FIELDPROC
  - VALIDPROC

SQL レプリケーションでは、特定の状況下で以下のデータ・タイプを複製することができます。

- ソース表およびターゲット表が DB2 for z/OS にある場合は、長い可変 GRAPHIC (LONG VARGRAPHIC) データ。
- 長い可変文字 (LONG VARCHAR および LONG VARGRAPHIC) データの場合は、ソース・データベース表が DB2 for z/OS にあるか、ソース表とターゲット表の両方が DB2 for Linux, UNIX, and Windows にあることが必要です。表の作成時に DATA CAPTURE CHANGES をソース表に指定した場合、LONG VARCHAR 列および LONG VARGRAPHIC 列は、自動的にすべてレプリケーション有効に設定されます。 ALTER TABLE ステートメントを使用して、それまで LONG 列がなかった表に LONG VARCHAR 列を追加する場合には、ALTER TABLE ステートメントを使用し、新しい LONG VARCHAR または LONG VARGRAPHIC 列に対して DATA CAPTURE CHANGES INCLUDE LONGVAR COLUMNS を有効にする必要があります。

SQL レプリケーションでは、要約データ・タイプを含む表を複製することはできません。

SQL レプリケーションでは、空間データ・タイプ列を含む表を複製することはできますが、実際の空間データ・タイプ列を複製することはできません。

ユーザー定義のデータ・タイプ (DB2 の特殊データ・タイプ) は、複製される前に変更データ (CD) 表で基本データ・タイプに変換されます。さらに、SQL レプリケーションがターゲット表をサブスクリプション・セット・メンバー定義として作成した場合、ユーザー定義タイプは CD 表内と同様に、ターゲット表内で基本データ・タイプに変換されます。

ラージ・オブジェクト・データ・タイプ

SQL レプリケーションは、ラージ・オブジェクト (LOB) のデータ・タイプをサポートします。このデータ・タイプには、バイナリー LOB (BLOB)、文字 LOB (CLOB)、2 バイト文字 LOB (DBCLOB) が含まれます。

このトピックでは、BLOB、CLOB、DBCLOB の各データ・タイプをまとめて LOB データといいます。

キャプチャー・プログラムはログ・レコード内の LOB 記述子を読み、変更されているために複製する必要があるデータが LOB 列にあるかどうかを判別しますが、変更データ (CD) 表には LOB データをコピーしません。LOB 列が変更されると、キャプチャー・プログラムは CD 表に標識を設定します。アプライ・プログラムはこの標識を読み取ると、次に LOB 列全体 (LOB 列の変更部分だけではない) をソース表からターゲット表に直接コピーします。

LOB 列には最大 2 GB のデータを入れることができるので、アプライ・プログラムには十分なネットワーク帯域幅を指定できるようにしてください。同じように、ターゲット表にも LOB データが入るだけのディスク・スペースを指定する必要があります。

制約事項:

- アプライ・プログラムは、常に最新バージョンの LOB 列をソース表 (CD 表ではない) から直接コピーします。これはその列が CD 表の他の列よりも新しい場合にも当てはまります。 このため、ターゲット行の LOB 列が変更された場合この LOB 列は、そのターゲット行の残りのデータと矛盾するものになる可能性があります。ターゲット行でデータの矛盾が発生する可能性を最小化するために、アプライ・サイクルのインターバルは、アプリケーションで実用上の問題が起きない範囲で、できるだけ短くしてください。
- 1 つの表でレプリケーションできる LOB 列は 10 以下です。10 よりも多い LOB 列を含む表を登録すると、アプライ・プログラムからエラー・メッセージが戻されます。 1 つの表に 10 よりも多くの LOB 列を登録しようとすると、レプリケーション・センターからエラー・メッセージが戻されます。
- 競合検出が使用不可の場合、LOB データをレプリカ表にコピーできます。
- DB2 for OS/390 バージョン 6 (またはそれ以降) と DB2 for Linux, UNIX, and Windows の間で LOB データをコピーするには、DB2 Connect バージョン 7 以降が必要です。
- ニックネームを使って LOB データを参照することはできません。
- LOB または ROWID 列の変更前イメージ値はサポートされていません。
- テキスト、音声、ビデオ、イメージ、その他の DB2 エクステンダー™の LOB 列データに関連付けられた追加のコントロール・ファイルがデータベースの外部で保守されている場合、DB2 エクステンダーのレプリケーションはサポートされていません。
- SQL レプリケーションでは、LOB 全体の複製だけが可能です。LOB の部分的な複製はできません。
- System i 上のレプリケーション環境でリモート・ジャーナル・セットアップを使用する場合は、LOB 列を複製できません。

第 7 章 SQL レプリケーション環境におけるデータのサブセット化

通常、レプリケーションではサブセット化が行われます。レプリケーション・ソースを登録するときに、ソース表から複製する列と行が選択されることもあります。サブスクリプション・セットを作成するときに、それぞれのターゲット表に複製する特定の登録済みの列が選択されることもあります。

レプリケーション要件に応じて、ソースの場合は登録時に、ターゲットの場合はサブスクリプションの際に、データをサブセット化できます。

- ソースに対するターゲットが 1 つだけの場合、または複数のターゲットがまったく同じデータを必要とする場合は、ニーズがターゲットごとに異なる可能性について考慮する必要がないため、登録時にデータをサブセット化したり操作したりすることができます。
- 1 つのソースに対して複数のターゲットが存在し、アプライされるデータに関する要件がその複数のターゲット間でそれぞれ異なるときは、登録時のサブセット化は不可能な場合がある。このケースでは、データのサブセット化はサブスクリプションの際に行うことになります。

レプリカ・ターゲット表に対するレプリケーションの場合は、これらの技法はどれも使用しないでください。Update-anywhere 構成では、マスター表と各レプリカ表は互いにデータを複製し合います。ソース表の使用されていない列が NULL 可能であるかぎり、レプリカ表はソース表の列のサブセットを持てます。そうでない場合は、レプリカ表はソース表と同じ列を含んでいなければならず、そのために列のサブセット化、新しい列の追加、または列の名前変更が不可能となります。

以下のトピックで、詳しく説明します。

登録時におけるデータのサブセット化

登録済みのソースからデータをキャプチャーする前または後に、特定の高度な技法を使用して、データをサブセット化できます。これらの技法は、データの同じサブセットを一度キャプチャーし、そのサブセットを多数のターゲット表に複製する場合に特に便利です。

データのサブセット化を、登録済みのソースからそのデータをキャプチャーする前と後のどちらに行うのかを選択できます。このセクションで示されている技法は、Update-anywhere レプリケーションまたはピアツーピア・レプリケーション以外のすべてのレプリケーション構成で使用できます。

登録時にデータをサブセット化すると、レプリケーションのパフォーマンスが向上する可能性があります。なぜなら、キャプチャー・プログラムが CD 表に追加するデータの量が削減されて、アプライ・プログラムが読み取るデータ量が削減されるからです。また、CD 表内の行数が少なくなるため、ストレージも削減されます。

以下のトピックでは、登録時にデータをサブセット化するための方法を取り上げています。

ビューを使用したソース・データのサブセット化

ソースの登録時には、レプリケーションに使用できるようにする列を選択します。選択した列はレプリケーションのためにキャプチャーされます。いくつかのケースでは、変更レプリケーション用のソースを登録した後で、そのソースのビューを登録する必要が生じることがあります。

例えば、人事部が、給与情報を含む人事データが入った表を保守しているとします。バックアップ・データベースを保守するために、この人事表全体がバックアップ・サイトに登録され、このサイトからサブスクライブされています。しかし、別のターゲット・サイトがこの人事表にサブスクライブする必要がある場合、この 2 番目のサブスクライバーから給与情報を隠蔽することができます。解決策は、この人事表に対するビューを登録し、2 番目のサブスクライバー用に登録されたビューだけにアクセス権を許可して、給与情報がアクセスから保護されるようにすることです。サブスクリプションは、この登録されたビューに対して作成できます。

また、複数のソース表が組み込まれたビューも登録できます。例えば、カスタマー表と支店表がある場合、カスタマーを適切にサブセット化してターゲットに正しく複製する唯一の方法は、この 2 つの表を結合して、特定の支店のカスタマーだけが特定のターゲットに複製されるようにすることです。このケースでは、二重削除を防止するための配慮が必要です。

特定の行のキャプチャーを防止するために CD 表にトリガーを定義する

レプリケーションのシナリオによっては、行内の特定の変更がキャプチャーされてターゲット表にレプリケーションされるのを防止することができます。特定の変更がキャプチャーされるのを抑制するには、CD 表にトリガーを定義してください。

ソースの登録時に管理ツールを使用すると、キャプチャーする列を選択できますが、それらの行における特定の変更がレプリケーションされるのを防止する機能は提供されません。レプリケーションのシナリオによっては、行内の特定の変更がキャプチャーされてターゲット表にレプリケーションされるのを防止することができます。例えば、ターゲット表にはすべての行が含まれているようにし、それらの表からはどの行も削除されないようにしたい場合は、ソースから削除が複製されないようにする必要があります。

特定の変更についてはキャプチャーしないように抑制するには、CD 表にトリガーを定義します。これらのトリガーにより、キャプチャー・プログラムが無視する必要のある変更が指定され、CD 表に加えられた変更に対応する行が追加されないようにされます。これらのトリガーは、レプリケーション・センターでは作成できませんが、既存の CD 表に対して (つまり、ソースを登録した後で) 手動で作成できます。キャプチャー・プログラムは、値が 99999 の SQLSTATE が表示されるトリガーの失敗をすべて無視するので、その行は CD 表に挿入されません。

例えば、表 SAMPLE.TABLE (CD 表は SAMPLE.CD_TABLE) からのレプリケーション中はソース表でのすべての DELETE 操作を抑制するとします。以下のトリガーは、DELETE 操作であるすべての行が CD 表に挿入されるのを抑制します。

```
CREATE TRIGGER SAMPLE.CD_TABLE_TRIGGER
NO CASCADE BEFORE INSERT ON SAMPLE.CD_TABLE
REFERENCING NEW AS CD
FOR EACH ROW MODE DB2SQL
WHEN (CD.IBMSNAP_OPERATION = 'D')
SIGNAL SQLSTATE '99999' ('CD INSERT FILTER')
```

登録時に生成された SQL にこの CREATE TRIGGER ステートメントを追加する場合は、登録を完了して CD 表にトリガーを作成するために、その変更した SQL を実行する必要があります。

これらのトリガーは、キャプチャー・プログラムが CD 表に対する行の挿入を試行するたびに実行されます。そのため、ここでのトリガーの使用がレプリケーション環境での最高のパフォーマンスをもたらすものなのかどうかについて検討する必要があります。 CD 表にトリガーを追加すると、データ・スループットが上がる場合と下がる場合とがあります。 CD 表のトリガーは、ソースで著しく多発する変更を抑制する場合に使用するようにしてください。変更の大部分をキャプチャーする予定だが、そのうちの一部の変更については複製されるのを抑制したいという場合は、不要な行をサブスクリプションの際に抑制します。

サブスクリプションの際のデータのサブセット化

サブスクリプションの際にデータをサブセット化すると、アプライ・プログラムが取り出すデータの量が減るので、レプリケーションのパフォーマンスが向上する可能性があります。また、ターゲット表の行を少なくするほど、ストレージ要件を減らすことができます。

アプライ・プログラムは、フル・リフレッシュ・レプリケーション時および変更キャプチャー・レプリケーション時にどのデータをコピーするのかを述部を使用して判別します。レプリケーション・センターおよび ASNCLP では、フル・リフレッシュ・レプリケーション用および変更キャプチャー・レプリケーション用の述部値を指定できます。変更キャプチャー・レプリケーションの場合にのみ使用する補足的な述部情報を追加したいときがあります。なぜなら、この情報はフル・リフレッシュ時には使用不可であるからです。この補足的な述部情報は、ユーザーが提供する SQL を使用して、UOW_CD_PREDICATES 列の IBMSNAP_SUBS_MEMBR 表に追加する必要があります。

例えば、ALL.CUSTOMERS という名前の登録済みの表があり、それに関連した CD 表の名前が ALL.CD_CUSTOMERS であるとします。サブスクリプション・ターゲットには ALL.CUSTOMERS のサブセットだけを含めたいとします (ALL.CUSTOMERS の ACCT_BALANCE 列は 50000 を超えます)。さらに、ターゲット表の履歴データを保守する必要がある (つまり、ターゲット表内のデータはどれも削除されたくない) とします。 PREDICATES 値が 'ACCT_BALANCE > 50000' のサブスクリプション・セット・メンバーを作成できます。

レプリケーション・センターまたは ASNCLP を使用する場合は、ターゲット表での削除は防止できません。なぜなら、操作のタイプに関する情報は CD 表に保管されるため、ソース表またはビューではこの情報が使用不可だからです。したがっ

て、以下の情報が組み込まれた SQL ステートメントを使用して、追加の変更キャプチャー述部を生成する必要があります。シナリオによっては、この更新ステートメントに列を追加して、 IBMSNAP_SUBS_MEMBR 表の単一行が確実に更新されるようにしなければならない場合があります。

```
UPDATE ASN.IBMSNAP_SUBS_MEMBR SET UOW_CD_PREDICATES = 'IBMSNAP_OPERATION <>''D'''
   WHERE APPLY_QUAL = 'apply_qual' AND SET_NAME = 'set_name' AND
   SOURCE_OWNER = 'ALL' AND SOURCE_TABLE = 'CUSTOMERS'
```

CD 表内の変更前イメージ列、CD 表からのあらゆるオーバーヘッド列、または UOW 表からのあらゆる列などといった、フル・リフレッシュ時に使用不可である列を参照するサブスクリプション・セット・メンバー述部の場合は、必ず UOW_CD_PREDICATES 列を手動でセットアップする必要があります。

デフォルトでは、ターゲット表がユーザー・コピーである場合でも、アプライ・プログラムは UOW 表と CD 表を結合しません。このプログラムは CD 表から直接データをフェッチして適用します。述部が UOW 表を参照する必要があり、かつターゲット表がユーザー・コピーである場合は、 IBMSNAP_SUBS_MEMBR 表の JOIN_UOW_CD 列の値を Y に設定する必要があります。このフラグを設定すれば、UOW 表と CD 表がアプライ・プログラムによって確実に結合されます。

行のサブセットに対して 1024 バイト (IBMSNAP_SUBS_MEMBR 表の PREDICATES 列の容量) を超える述部を指定する場合は、ソース・ビューを使用する必要があります。

サブスクリプション・セットに複合述部ステートメントを使用している場合は、式全体を括弧で囲みます。例えば、述部ステートメントに AND および OR 文節を使用している場合には、式を次のように囲みます。

```
((TOSOURCE = 101 AND STATUS IN (202,108,109,180,21,29,32,42))
OR (SOURCE = 101))
```

第 8 章 SQL レプリケーション環境におけるデータ操作

ソース・データは、ターゲット表に複製される前にトランスフォームまたは拡張することができます。

例えば、以下のような方法でデータを操作することがあります。

- データ・クレンジングを実行する操作
- データ集約を実行する操作
- ソースに存在しない列にターゲット表でデータを取り込む操作

アプライ・プログラムを使用すれば、データをターゲットに適用する前または後に、以下の方法でデータを操作できます。

- ストアード・プロシージャーまたは SQL ステートメントの使用
- 107 ページの『名前が異なるソース列とターゲット列のマッピング』
- 107 ページの『算出列の作成』

キャプチャーの前または後にデータを操作できます。データを 1 回操作して、トランスフォームされたデータを多数のターゲット表に複製する場合は、サブスクリプションではなく登録でデータを操作します。すべてのソース・データをキャプチャーし、トランスフォームされたデータを選択して個々のターゲットに適用する場合は、登録ではなくサブスクリプション中にデータを操作します。

レプリケーションのシナリオによっては、CD 表に保管されたソース・データの内容を操作することになります。トリガー、サブスクリプション中の式、あるいはソース・ビューのどれを使用しても、同じジョブを実行できます。それぞれの方式に、良い点と悪い点があります。トリガーは、CPU サイクルの使用コストがかかりすぎる可能性があります。関数をセットアップするのに、サブスクリプションは複数回必要ですが、ビューを使用すると 1 回でセットアップできます。

例えば、ソース表で特定の値が脱落している場合、キャプチャー・プログラムで NULL 値をキャプチャーする必要はないでしょう。

CD 表に対してトリガーを使用して、データを CD 表に挿入するときにキャプチャー・プログラムがデータを拡張する条件を指定できます。この場合、キャプチャー・プログラムがソースで NULL 値を発見したら、デフォルト値を CD 表に挿入するように指定できます。データがソース表の更新から脱落している場合、以下のコードを使用して、確定したデフォルトを提供するトリガーを作成できます。

```
CREATE TRIGGER ENHANCECD
NO CASCADE BEFORE INSERT ON CD_TABLE
REFERENCING NEW AS CD
FOR EACH ROW MODE DB2SQL
WHEN (CD.COL1 IS NULL)
SET CD.COL1 ='MISSING DATA'
END
```

トリガーの代わりに、登録済みソース・ビューまたはサブスクリプションの式で DB2 の COALESCE スカラー関数を使用できます。登録済みビューで、この合体関数が最初の非 NULL 値を戻します。

ソース・ビューの使用例の一部:

```
CREATE VIEW SAMPLE.SRCVIEW  (columns) AS SELECT
       ... COALESCE(A.COL1, 'MISSING DATA') ...
       FROM SAMPLE.TABLE A
```

式の使用例の一部:

```
COALESCE(CD.COL1, 'MISSING DATA')
```

ストアード・プロシージャーまたは SQL ステートメントを使用したデータ拡張

サブスクリプション・セット情報を定義するとき、特定のセットを処理するたびにアプライ・プログラムに実行させるランタイム処理ステートメントを、SQL ステートメントまたはストアード・プロシージャーを使って定義することもできます。これらのランタイム処理により、レプリケーション中にデータを操作できるようになります。

サブスクリプション・セット情報を定義するとき、特定のセットを処理するたびにアプライ・プログラムに実行させるランタイム処理ステートメントを、SQL ステートメントまたはストアード・プロシージャーを使って定義することもできます。これらのランタイム処理により、レプリケーション中にデータを操作できるようになります。このようなステートメントは、 CCD 表の整理やサブスクリプション・セット処理でのシーケンス・コントロールにも役立ちます。ランタイム処理ステートメントは、サブスクリプション・セットの処理前ならキャプチャー・コントロール・サーバーで、あるいはサブスクリプション・セットの処理前または後であればターゲット・サーバーで実行できます。例えば、データを取り出す前、またはターゲット表にデータを複製した後、あるいはその両方の場合に SQL ステートメントを実行することができます。

ニックネームの制約事項: フェデレーテッド DB2 表 (ニックネームを使用する) は、通常単一の作業単位内で更新されます。アプライ・プログラムがすべてのデータをターゲットに適用した後、SQL ステートメントをサブスクリプション・セットに追加する場合、以下の 2 つの状況のいずれかで、SQL COMMIT ステートメントをその SQL ステートメントより先行させる必要があります。

- この SQL ステートメントは、サブスクリプション・セットのターゲット表またはターゲット・ニックネームがあるサーバー以外のサーバー上のニックネームに対して挿入、更新、または削除を行う。
- この SQL ステートメントは、アプライ・コントロール・サーバーのローカルの表に対して、挿入、更新、または削除を行うが、サブスクリプション・セットのターゲット・ニックネームはリモート・サーバーにある。

追加の COMMIT ステートメントは、追加された SQL ステートメントを処理する前に、アプライ・プログラムの作業をコミットします。

ストアード・プロシージャーは、パラメーターなしの SQL CALL ステートメントを使用します。プロシージャー名の長さは、18 文字以下でなければなりません

(System i の場合は、最大 128 文字です)。ソースまたはターゲット表が非 DB2 リレーショナル・データベースにある場合、 SQL ステートメントはフェデレーテッド DB2 データベースを対象として実行されます。 SQL ステートメントが非 DB2 データベースを対象として実行されることは決してありません。各タイプのランタイム・プロシージャーは単一トランザクションとして一緒に実行されます。さらに、それぞれのステートメントごとに受け入れ可能 SQLSTATE を定義することもできます。

(特定のセットの処理の完了後ではなく) それぞれのセットの処理の完了後にデータを操作する場合は、ASNDONE 出口ルーチンを使用します。

名前が異なるソース列とターゲット列のマッピング

レプリケーション・センターまたは ASNCLP コマンド行プログラムを使用してサブスクリプション・セット・メンバーを定義する場合に、参照先のターゲット表が存在しなければ、ターゲット表のタイプに関係なく、ターゲット表で列を名前変更できます。互換性のある列の属性を変更することも可能です。

さらに、互換性があれば、列の属性 (データ・タイプ、長さ、位取り、精度、NULL 可能かどうか) を変更することもできます。レプリケーション管理ツールを使用して、既存のターゲット表の列を名前変更することはできません。

サブスクリプション・セット・メンバーの参照先のターゲット表がすでに存在している場合、管理ツールは名前によって列をマップしようとします。ソース列とターゲット列が一致しない場合は、ツールを使用してソースからターゲットに列をマップするか、ソースの列名に一致する名前を含むターゲット表のビューを作成するかのどちらかが可能です。

算出列の作成

既存のターゲット表の列名を変更することはできませんが、ソース列の式を変更して、既存のターゲット表の列に正しくマップするようにしたり、あるいはそれとの互換性を持たせることができます。

SQL 式を使用すれば、既存のソース列から新しい列を派生させることもできます。集約ターゲット表タイプでは、COUNT や SUM といった集約関数を使って新しい列を定義できます。ターゲット表の他のタイプでは、式でスカラー関数を使用して新しい列を定義できます。ソース表とターゲット表の列の名前だけが違って、他は互換性がある場合、レプリケーション・センターまたは ASNCLP を使用すれば列をもう一方の表の列にマップできます。

例えば、既存のソース表 (SRC.TABLE) とターゲット表 (TGT.TABLE) があるとします。

```
CREATE TABLE SRC.TABLE (SRC_COL1 CHAR(12) NOT NULL, SRC_COL2 INTEGER,
     SRC_COL3 DATE, SRC_COL4 TIME, SRC_COL5 VARCHAR(25))

CREATE TABLE TGT.TABLE (TGT_COL1 CHAR(12) NOT NULL,
     TGT_COL2 INTEGER NOT NULL, TGT_COL3 TIMESTAMP, TGT_COL4 CHAR(5))
```

以下のステップで、サブスクリプション中に算出列を使用して必要なターゲット表をマップします。

1. レプリケーション・センターを使用して、ソース表の SRC_COL1 をターゲット表の TGT_COL1 にマップする。これらの列には互換性があるので、一方から他方にマップするのに式を使う必要はありません。
2. 式 COALESCE(SRC_COL2, 0) を使用して列値を計算し、TGT_COL2 に提供するためにマップする。 SRC_COL2 は NULL 可能で TGT_COL2 は NOT NULL のため、TGT_COL2 の値が必ず非 NULL になるようにこのステップを実行する必要があります。
3. 式 TIMESTAMP(CHAR(SRC_COL3) CONCAT CHAR(SRC_COL4)) を使用して列値の計算をし、TGT_COL3 に提供するためにマップする。この列式はターゲット・データベースのタイム・スタンプ列にマップするデータを提供します。
4. 式 SUBSTR(SRC_COL5, 1,5) を使用して列値の計算をし、TGT_COL4 に提供するためにマップする。

第 9 章 SQL レプリケーションに関するキャプチャー・プログラムの操作

このセクションは、DB2 データベースに対するログ・ベースのキャプチャーに関するものです。トリガー・ベースのキャプチャーを使用している場合は、登録時にトリガーが作成されるため、ユーザーはこのセクションで説明されている操作を実行しません。

キャプチャー・プログラムの始動 (Linux、UNIX、Windows、および z/OS)

キャプチャー・プログラムを始動して、DB2 データベースのログからのデータのキャプチャーを開始してください。非 DB2 リレーショナル・ソースに対してトリガー・ベースのキャプチャーを使用している場合は、登録時にトリガーが作成されるため、キャプチャー・プログラムを始動する必要はありません。

始める前に

- ソース・サーバーおよびキャプチャー・コントロール・サーバーへの接続を構成します。
- 正しい許可を受けていることを確認します。
- 該当するキャプチャー・スキーマのためのコントロール表を作成します。
- 登録を定義します。
- キャプチャー・プログラムとアプライ・プログラムを構成します。

このタスクについて

注: DB2 ユーティリティーは、変更をキャプチャー・プログラムから見える方法ではログに記録しません。そのため、キャプチャー・プログラムは DB2 ユーティリティーによって行われた変更を一切キャプチャーしません。

キャプチャー・プログラムの始動時に、始動パラメーターを指定することもできます。

キャプチャー・プログラムの始動後、キャプチャー・プログラムがデータのキャプチャーをすぐには開始しないことがあります。このプログラムがデータのキャプチャーを開始するのは、ターゲット表を完全にリフレッシュしたことをアプライ・プログラムがキャプチャー・プログラムにシグナルで通知した後だけです。このシグナルを受け取ると、キャプチャー・プログラムは所定のソース表用のログからの変更のキャプチャーを開始します。

手順

Linux、UNIX、Windows、 z/OS、でキャプチャー・プログラムを始動するには、次の方法のいずれかを使用します。

方法	説明
レプリケーション・センター	「キャプチャーの開始」ウィンドウを使用します。このウィンドウを開くには、オブジェクト・ツリーの「**操作**」ブランチから「**キャプチャー・コントロール・サーバー**」フォルダーをクリックし、内容ペインで、始動するキャプチャー・プログラムが配置されているキャプチャー・コントロール・サーバーを右クリックします。「**キャプチャーの開始**」を選択します。
z/OS Linux UNIX Windows asncap システム・コマンド	このコマンドを使用してキャプチャー・プログラムを始動します。始動パラメーターを指定することもできます。
z/OS z/OS コンソールまたは TSO	z/OS の場合は、JCL を使用してキャプチャー・プログラムを始動したり、システム始動タスクとしてキャプチャー・プログラムを始動したりすることもできます。JCL でキャプチャー・プログラムを始動する場合は、新しい呼び出しパラメーター値を指定できます。JCL を使用する場合に呼び出しパラメーターを指定する最適な方法は、IBMQREP_CAPPARMS 表に呼び出しパラメーターを格納しておくことです。EXEC ステートメントの PARM パラメーターには、100 文字を超えるサブパラメーターを指定できません。
Windows Windowsサービス	Windows オペレーティング・システムの場合は、システムの始動時にキャプチャー・プログラムを自動的に始動するための DB2 レプリケーション・サービスを作成できます。

キャプチャー・プログラムが始動したかどうかを確認するには、次の方法のいずれかを使用します。

- z/OS バッチ・モードで実行している場合は、z/OS コンソールまたは z/OS ジョブ・ログを調べて、プログラムが始動したことを示すメッセージを探します。
- キャプチャーの診断ログ・ファイル (z/OS の場合は *capture_server.capture_schema*.CAP.log、Linux、UNIX、Windows の場合は *db2instance.capture_server.capture_schema*.CAP.log) を調べて、プログラムが変更をキャプチャーしていることを示すメッセージを探します。以下に例を示します。

  ```
  ASN0104I Change capture has been started for the source
  table "REGRESS.TABLE1" for changes found in the log beginning
  with log sequence number "0000:0275:6048".
  ```

- IBMSNAP_CAPTRACE 表を調べて、プログラムが変更をキャプチャーしていることを示すメッセージを探します。
- レプリケーション・センターの「キャプチャー・メッセージ」ウィンドウを使用して、プログラムが始動したことを示すメッセージを探します。そのウィンドウを開くには、メッセージを表示するキャプチャー・プログラムが配置されているキャプチャー・サーバーを右クリックして、「**レポート**」 → 「**キャプチャー・メッセージ**」を選択します。
- レプリケーション・センターの「状況のチェック」ウィンドウまたは asnccmd status コマンドを使用して、すべてのキャプチャー・スレッドの状況を表示しま

す。そのウィンドウを開くには、チェックするキャプチャー・プログラムが配置されているキャプチャー・サーバーを右クリックして、「**状況のチェック**」を選択します。

キャプチャー・プログラムの始動 (System i)

キャプチャー・プログラムを始動して、ジャーナルからのデータのキャプチャーを開始してください。

始める前に

キャプチャー・プログラムを始動する前に、以下の前提条件が満たされていることを必ず確認してください。

- 正しい許可を受けている。
- 適切なキャプチャー・スキーマ用のコントロール表が作成されており、登録が定義済みである。
- レプリケーション・プログラムが構成済みである (キャプチャー・プログラムがリモート・ジャーナルを読み取っている場合)。

このタスクについて

キャプチャー・プログラムの始動後、キャプチャー・プログラムがデータのキャプチャーをすぐには開始しないことがあります。このプログラムがデータのキャプチャーを開始するのは、アプライ・プログラムがキャプチャー・プログラムに対し、所定のソース表用のログからの変更のキャプチャーを開始するよう求めるシグナルを送った後だけです。

手順

System i の場合にキャプチャー・プログラムを始動するには、次の方法のいずれかを使用します。

方法	説明
STRDPRCAP システム・コマンド (System i)	変更のキャプチャーを開始するには、 DPR キャプチャー・プログラムの始動 (STRDPRCAP) コマンドを使用します。
レプリケーション・センター	「キャプチャーの開始」ウィンドウを使用します。このウィンドウを開くには、オブジェクト・ツリーの「**操作**」ブランチから「**キャプチャー・コントロール・サーバー**」フォルダーをクリックし、内容ペインで、始動するキャプチャー・プログラムが配置されているキャプチャー・コントロール・サーバーを右クリックします。「**キャプチャーの開始**」を選択します。

キャプチャー・プログラムのデフォルトの操作パラメーター

キャプチャー・コントロール表を作成すると、キャプチャー・プログラムの操作パラメーターのデフォルト値が IBMSNAP_CAPPARMS 表に保管されます。

112 ページの表 5と 112 ページの表 6にデフォルト値を示します。

Linux UNIX Windows

表 5. キャプチャーの稼働パラメーターのデフォルト設定 (Linux、UNIX、Windows、z/OS)

稼働パラメーター	デフォルト値	IBMSNAP_CAPPARMS 表の列名
capture_server	DB2DBDFT[1]	該当せず
capture_schema	ASN[2]	該当せず
add_partition	n[4]	該当せず
asynchlogrd	n[4]	該当せず
retention_limit	10080 分	RETENTION_LIMIT
lag_limit	10080 分	LAG_LIMIT
commit_interval	30 秒	COMMIT_INTERVAL
prune_interval	300 秒	PRUNE_INTERVAL
trace_limit	10080 分	TRACE_LIMIT
monitor_limit	10080 分	MONITOR_LIMIT
monitor_interval	300 秒	MONITOR_INTERVAL
memory_limit	32 MB	MEMORY_LIMIT
autoprune	y[3]	AUTOPRUNE
term	y[3]	TERM
autostop	n[4]	AUTOSTOP
logreuse	n[4]	LOGREUSE
logstdout	n[4]	LOGSTDOUT
sleep_interval	5 秒	SLEEP
capture_path	キャプチャーが始動されたディレクトリー[5]	CAPTURE_PATH
startmode	warmsi[6]	STARTMODE

注:

1. キャプチャー・コントロール・サーバーは、Windows、Linux、および UNIX の場合の DB2DBDFT 環境変数の値である (変数が指定されている場合)。z/OS にはデフォルトはない。
2. キャプチャー・スキーマのデフォルトは変更できない。別のキャプチャー・スキーマを使用するには、**capture_schema** 始動パラメーターを使用する。
3. Yes
4. No
5. キャプチャーが Windows サービスとして始動する場合、そのキャプチャー・パスは ¥sqllib¥bin である。
6. キャプチャー・プログラムはウォーム・スタートする。これがこのプログラムの初回の始動である場合にのみ、コールド・スタートに切り替わる。

System i

表 6. キャプチャー稼働パラメーターのデフォルト設定 (System i)

稼働パラメーター	デフォルト値	IBMSNAP_CAPPARMS 表の列名
CAPCTLLIB	ASN[1]	該当せず
JOBD	*LIBL/QZSNDPR	該当せず

表 6. キャプチャー稼働パラメーターのデフォルト設定 (System i) (続き)

稼働パラメーター	デフォルト値	IBMSNAP_CAPPARMS 表の列名
JRN	*ALL	該当せず
RETAIN	10080 分	RETENTION_LIMIT
LAG	10080 分	LAG_LIMIT
FRCFRQ	30 秒	COMMIT_INTERVAL
CLNUPITV	*IMMED [2]	該当せず
CLNUPITV	86400 秒[2]	PRUNE_INTERVAL
CLNUPITV	*IMMED [2]	該当せず
TRCLMT	10080 分	TRACE_LIMIT
MONLMT	10080 分	MONITOR_LIMIT
MONITV	300 秒	MONITOR_INTERVAL
MEMLMT	32 MB	MEMORY_LIMIT
WAIT	120 秒	該当せず
RESTART	*YES[3]	該当せず

注:

1. キャプチャー・スキーマのデフォルトは変更できない。別のキャプチャー・スキーマを使用するには、キャプチャー・プログラムの始動時に **CAPCTLLIB** パラメーターを指定する。他のほとんどの稼働パラメーターのデフォルト値は、 IBMSNAP_CAPPARMS 表に保管されている。
2. **CLNUPITV** には 2 つのサブパラメーターがある。デフォルトでは、キャプチャー・プログラムは実行開始直後に整理を実行し、整理インターバルに達するたび (デフォルトでは 24 時間ごと) に再び整理を実行する。
3. デフォルトでは、キャプチャー・プログラムはウォーム・スタートする。

キャプチャーの操作パラメーターの説明

キャプチャー・プログラムの始動時に、始動パラメーターを選択することもできます。ここでは、それぞれの始動パラメーターを取り上げ、各パラメーターで選択するべき値に関する推奨事項を示します。

特に断りのない限り、すべてのパラメーターは z/OS、Linux、UNIX、および Windows に適用されます。

- 114 ページの『add_partition (Linux、UNIX、Windows)』
- 114 ページの『autoprune』
- 114 ページの『autostop』
- 115 ページの『capture_path』
- 115 ページの『capture_schema』
- 116 ページの『capture_server』
- 117 ページの『commit_interval』
- 117 ページの『lag_limit』
- 117 ページの『logreuse』
- 118 ページの『logstdout』

- 118 ページの『memory_limit』
- 119 ページの『monitor_interval』
- 119 ページの『prune_interval』
- 120 ページの『retention_limit』
- 120 ページの『sleep_interval』
- 121 ページの『startmode』
- 122 ページの『term』
- 122 ページの『trace_limit』

add_partition (Linux、UNIX、Windows) Linux UNIX Windows

デフォルト: **add_partition**=n

add_partition パラメーターは、最後にキャプチャー・プログラムが再始動されてから、新しく追加されたパーティションのログ・ファイルの読み取りを、キャプチャー・プログラムが開始するかどうかを指定します。

add_partition=y と設定し、キャプチャー・プログラムがログ・ファイルを読み取るようにします。各新規パーティションにおいて、ウォーム・スタート・モードでキャプチャー・プログラムが開始されると、キャプチャーがログ・ファイルを読み取ります。読み取りは、最初のデータベース CONNECT ステートメントが DB2 インスタンスに対して実行された後に DB2 が使用した最初のログ・シーケンス番号 (LSN) から開始されます。

autoprune

デフォルト: **autoprune**=y

autoprune パラメーターは、キャプチャー・プログラムがそのコントロール表のいくつかを自動的に整理するかどうかを指定します。デフォルト (**autoprune**=y を使用) では、キャプチャー・プログラムは CD 表と UOW 表の行、ならびに IBMSNAP_CAPTRACE 表、IBMSNAP_CAPMON 表、および IBMSNAP_SIGNAL 表の行を自動的に整理します。 **autoprune**=nを設定した場合は、prune コマンドを使用してこれらの表を整理する必要があります。

自動整理をオンに設定してキャプチャーを始動する場合は、整理インターバルを設定して、使用しているレプリケーション環境での整理頻度を最適化してください。キャプチャー・プログラムは以下のパラメーターを使用して、どの行が整理されてもよいほど古いのかを判別します。

- **retention_limit** (CD 表、UOW 表、およびシグナル表の場合)
- **monitor_limit** (モニター表の場合)
- **trace_limit**(キャプチャー・トレース表の場合)

autostop

デフォルト: **autostop**=n

autostop パラメーターは、ログの終わりに達した後にキャプチャー・プログラムが稼働し続けるか終了するかをコントロールします。

デフォルト (**autostop**=n) では、キャプチャー・プログラムはトランザクションの検索後に終了しません。

モバイル環境または時々しか接続しない環境でレプリケーションを行っている場合は **autostop**=y オプションを使用してください。 autostop を使用すると、キャプチャー・プログラムは確実にすべての適格トランザクションを検索し、ログの終わりに達した時点で必ず停止します。それ以外のトランザクションも検索する場合は、キャプチャーを再始動する必要があります。 **autostop**=y オプションは、テスト環境でも使用されます。

推奨: ほとんどのケースでは、**autostop**=y を使用しないでください。これを使用すると、レプリケーションの管理にオーバーヘッドが加わるからです (例えば、キャプチャー・プログラムを再始動し続ける必要があります)。

capture_path

キャプチャー・パスは、キャプチャー・プログラムが自分用の作業ファイルとログ・ファイルを保管するディレクトリーです。デフォルトでは、キャプチャー・パスはキャプチャー・プログラムが始動されるディレクトリーです。

z/OS

キャプチャー・プログラムは POSIX アプリケーションであるため、デフォルトのキャプチャー・パスは、次に示す方法のどちらでこのプログラムを開始するかによって決まります。

USS コマンド行プロンプト
プログラムを開始したディレクトリー。

開始済みタスクまたは JCL を使用して
その開始済みタスクまたはジョブと関連したユーザー ID のホーム・ディレクトリー。

パス名または //CAPV9 などの高位修飾子 (HLQ) のいずれかを指定できます。HLQ を使用すると、z/OS 順次データ・セット・ファイル名のファイル命名規則に準拠する順次ファイルが作成されます。順次データ・セットは、プログラムを実行中のユーザー ID と関連付けられています。それ以外の場合は、これらのファイル名は、明示的に名前を指定されたディレクトリー・パスに保管されるファイル名 (HLQ がファイル名の最初の部分として連結されているもの) に類似します。例えば、`sysadm.CAPV8.`*`filename`* のようになります。

Linux UNIX Windows

キャプチャー・パスを変更して、キャプチャー・プログラムがファイルを保管する場所を指定できます。パス名 (例えば、/home/db2inst/capture_files) を指定できます。キャプチャー・プログラムを Windows サービスとして始動する場合、デフォルトではキャプチャー・プログラムは ¥sqllib¥bin ディレクトリーで始動されます。

capture_schema

デフォルト: **capture_schema**=ASN

capture_schema パラメーターにより、どのキャプチャー・プログラムを始動するのかが識別されます。デフォルトでは、キャプチャー・スキーマは ASN です。

別のスキーマをすでにセットアップしてある場合は、**capture_schema** パラメーターを使用してそのスキーマを指定すれば、当該のキャプチャー・プログラムを始動できます。

次のような場合には、複数のキャプチャー・スキーマを使用することがあります。

アプリケーションの独立性を保ってアーカイブする場合

アプリケーション A 用のキャプチャー・プログラムとアプリケーション B 用のキャプチャー・プログラムを別個に持てるように、複数のキャプチャー・スキーマを作成してください。各キャプチャー・プログラムは、それぞれ独自のコントロール表を使用します。キャプチャー・プログラムのいずれかがダウンしても、1 つのアプリケーションだけしか影響を受けません。それ以外のアプリケーションは、別のキャプチャー・プログラムからサービスを受けているために影響されません。

各アプリケーションのそれぞれ異なる要件を満たす場合

同じソース表を使用するが、データ要件がそれぞれ異なる複数のアプリケーションがある場合は、複数のキャプチャー・スキーマを作成してください。例えば、給与計算アプリケーションは機密従業員データを必要としますが、内部従業員登録にはこのようなデータは必要ありません。その機密情報を 1 つのキャプチャー・スキーマだけに登録して、それ以外のキャプチャー・スキーマには登録しないようにできます。同様に、一部のアプリケーションではキャプチャー・プログラムに異なる動作をさせる必要がある場合は、1 つの表を複数回登録できます。例えば、一部のアプリケーションではキャプチャー・プログラムが更新の保管を削除と挿入の対として実行する必要があるというような場合です。

登録に関する問題を分離する場合

1 つの登録に問題がある場合は、別のキャプチャー・スキーマを作成して、作業登録をそのスキーマに移動できます。この方法により、その問題のある登録をオリジナルのスキーマ内でデバッグし、影響を受けていない登録を別のスキーマを使用して実行できます。

capture_server

z/OS **デフォルト: capture_server=**None

Linux UNIX Windows **デフォルト: capture_server=** DB2DBDFT 環境変数の値 (設定されている場合)

capture_server パラメーターは、キャプチャー・コントロール・サーバーを指定します。

z/OS キャプチャー・コントロール表は DB2 サブシステム名で配置されます。キャプチャー・プログラムは DB2 ログを読み取るため、キャプチャー・プログラムはソース・データベースと同じサーバーで実行しなければなりません。

Linux UNIX Windows キャプチャー・コントロール表 (登録表など) はソース表の登録情報を含み、キャプチャー・コントロール・サーバー上に配置されます。

commit_interval

デフォルト: **commit_interval**=30

commit_interval パラメーターは、キャプチャー・プログラムがキャプチャー・コントロール表 (UOW 表および CD 表も含む) に対してデータをコミットする頻度を秒単位で指定します。デフォルトでは、キャプチャー・プログラムは UOW 表および CD 表にデータをコミットする前に 30 秒待ちます。コミット・インターバル内に更新された表に対して、ロックが保留されます。 **commit_interval** パラメーターに比較的大きな値を指定すると、キャプチャー・プログラムの CPU 使用量が削減されますが、それと同時に、頻繁に実行されているサブスクリプション・セットの待ち時間が増える可能性があります。なぜなら、アプライ・プログラムが取り出せるのはコミット済みのデータだけだからです。

lag_limit

デフォルト: **lag_limit**=10 080

lag_limit パラメーターは、キャプチャー・プログラムが DB2 ログからのレコードを処理する際に許される遅れの分数を表します。

デフォルトでは、ログ・レコードが 10,080 分 (7 日) より古い場合、キャプチャー・プログラムがコールド・スタートに切り替えるのを許可する値が **startmode** パラメーターに指定されないかぎり、キャプチャー・プログラムは始動しません。

遅れが限度に達したためにキャプチャー・プログラムが始動しない場合は、キャプチャー・プログラムがログの読み取りで遅れを出している理由を判別する必要があります。この遅れの限度に関するパラメーターを実際に使用することのないテスト環境では、遅れの限度をより高く設定して、キャプチャー・プログラムの再始動を試行することができます。また、使用するテスト環境のソース表にはほんの少しのデータしか入っていない場合には、コールド・スタートを使用して、すべてのターゲット表内のデータを完全にリフレッシュすることもできます。

logreuse

デフォルト: **logreuse**=n

キャプチャー・プログラムは、操作情報をログ・ファイルに保管します。

z/OS ログ・ファイル名には DB2 インスタンス名は含まれません。例えば、SRCDB1.ASN.CAP.log。このファイルは、**capture_path** パラメーターで指定されているディレクトリーに保管されます。 **capture_path** パラメーターが高位修飾子 (HLQ) として指定されている場合は、z/OS 順次データ・セット・ファイルのファイル命名規則が適用されます。したがって、ログ・ファイル名を作成するのに使用される **capture_schema** 名は、その名前の最初の 8 文字までの長さに切り捨てられます。

Linux UNIX Windows ログ・ファイルの名前は *db2instance.capture_server.capture_schema*.CAP.log です。例えば、DB2INST.SRCDB1.ASN.CAP.log。

デフォルト (**logreuse=n**) では、キャプチャー・プログラムはメッセージをログ・ファイルに付加します。これは、キャプチャー・プログラムが再始動された後であっても同様に行われます。メッセージの履歴が必要な場合は、デフォルトのままにしておいてください。次のような場合には、キャプチャー・プログラムが再始動時にログを削除して再作成するようにさせる (**logreuse=y**) 必要があります。

- ログが大きくなったためにログを消去する場合。
- ログに保管されている履歴が不要な場合。
- スペースを節約する場合。

logstdout

デフォルト: **logstdout=n**

logstdout パラメーターが使用可能なのは、asncap コマンドを使用する場合だけです。このパラメーターは、レプリケーション・センターでは使用不可です。

デフォルトでは、キャプチャー・プログラムは一部の警告メッセージと通知メッセージをログ・ファイルにのみ送信します。トラブルシューティング中や、テスト環境でキャプチャー・プログラムの動作をモニターしている場合には、このようなメッセージは標準出力 (STDOUT) に送信されるようにする (**logstdout=y**) ことができます。

memory_limit

デフォルト: **memory_limit=32**

memory_limit パラメーターは、キャプチャー・プログラムが使用できるメモリーの量を MB (M バイト) 単位で指定します。

デフォルトでは、キャプチャー・プログラムは 32 MB のメモリーを使用してトランザクション情報を保管します。その後あふれた分の情報は、**capture_path** ディレクトリーに配置されているファイルに入れられます。このメモリーの限度は、パフォーマンスに関するニーズに基づいて変更できます。メモリーの限度を高めに設定すると、キャプチャーのパフォーマンスは向上しますが、システム上のその他のユーザーが使用可能なメモリーが少なくなります。メモリーの限度を低めに設定すると、その他のユーザーのためにメモリーが解放されます。メモリーの限度をあまりに低く設定して、キャプチャー・プログラムが入りきらない分をファイルに入れるようになると、システム上で使用するスペースが増え、さらに入出力のためにシステムがスローダウンすることになります。

メモリーの限度は、レプリケーション・アラート・モニターを使用してモニターできます。また、CAPMON 表内のデータを使用すると、メモリー制限のためにディスクにあふれたソース・システム・トランザクションの数も判別できます。 CAPMON 表の TRANS_SPILLED 列の値を合計してください。

monitor_interval

デフォルト: **monitor_interval=**300

monitor_interval パラメーターは、キャプチャー・プログラムが情報を IBMSNAP_CAPMON 表に書き込む頻度を指定します。

デフォルトでは、キャプチャー・プログラムはキャプチャー・モニター表に 300 秒 (5 分) ごとに行を挿入します。この稼働パラメーターは、コミット・インターバルと関連して機能します。データを細分化されたレベルでモニターしてみたい場合は、コミット・インターバルに近いモニター・インターバルを使用してください。

monitor_limit

デフォルト: **monitor_limit=**10080

monitor_limit パラメーターは、整理の対象にする前に、行がどのくらい古くなるまでモニター表に残しておく必要があるのかを指定します。

デフォルトでは、IBMSNAP_CAPMON 表内の 10,080 分 (7 日) より古い行が整理されます。IBMSNAP_CAPMON 表には、キャプチャー・プログラムに関する稼働統計が含まれています。 1 週間分より少ない統計しか必要でない場合は、デフォルトのモニター限度を使用してください。統計を頻繁にモニターする場合は、1 週間分の統計を保持する必要性はないと考えられるので、モニター限度を低めに設定し、キャプチャー・モニター表がより頻繁に整理されるようにして、古い統計が除去されるようにできます。この統計を履歴の分析に使用する場合、および 1 週間より長い期間の統計が必要である場合は、モニター限度の値を大きくしてください。

prune_interval

デフォルト: **prune_interval=**300

prune_interval パラメーターは、キャプチャー・プログラムが一部のコントロール表から古い行を削除しようとする、その試行頻度を指定します。このパラメーターが有効なのは、**autoprune=y** の場合だけです。

デフォルトでは、キャプチャー・プログラムは CD 表および UOW 表を 300 秒 (5 分) ごとに整理します。表の整理頻度が不十分であると、それらの表が含まれている表スペース用のスペースが使い尽くされる可能性があります。そうなった場合は、キャプチャー・プログラムは強制停止されます。表の整理があまりに頻繁にまたはピーク時に実行される場合は、そのような整理によって同じシステム上で実行されているアプリケーション・プログラムに支障が生じる可能性があります。オプションとしての整理頻度をレプリケーション環境で設定する場合は、実際のレプリケーション環境に最適な値を使用してください。一般に、パフォーマンスは表のサイズが小さく保たれている場合に最適となります。

整理インターバルの値を低くする前に、整理が発生する可能性があるほどデータが頻繁にアプライされていることを確認してください。アプライ・プログラムがデータを頻繁にアプライしていない場合は、整理インターバルの値を低く設定しても無益です。なぜなら、CD 表と UOW 表を整理するためには、その前にアプライ・プログラムがすべてのターゲットに対してデータを複製する必要があるからです。

整理インターバルにより、キャプチャー・プログラムがどのくらいの頻度で表の整理を試行するのかが決定されます。整理間隔は、データが整理の対象となるのに十分なだけ古くなったと見なす時点を決定するパラメーター (**trace_limit**、**monitor_limit**、**retention_limit**) と関連して機能します。例えば、**prune_interval** が 300 秒、**trace_limit** が 10080 秒である場合は、キャプチャー・プログラムは 300 秒ごとに整理を試行します。トレース表で 10080 分 (7 日) より古い行を検出すると、キャプチャー・プログラムはそれらをすべて整理します。

retention_limit

デフォルト: **retention_limit=**10 080

retention_limit パラメーターは、保持制限整理の対象として適格と見なすまでに、古いデータをどれだけの期間 CD 表、UOW 表、および IBMSNAP_SIGNAL 表に残しておくのかを決定します。

サブスクリプション・セットが非活動にされているか、またはときたましか実行されないために、通常の整理処理が使用禁止になっている場合は、データは長い間 CD 表および UOW 表に残されます。このデータが現行 DB2 タイム・スタンプから保持制限の値を引いたものより古くなると、このデータは保持制限整理処理によって表から削除されます。サブスクリプション・セットの実行が極めてまれであったり、アプライ・プログラムを停止すると、使用している CD 表と UOW 表が非常に大きくなり、保持制限整理の対象となる可能性があります。

整理される行のいずれか 1 つでもレプリケーションの候補になっており、それにもかかわらずなんらかの理由でそれらがターゲット表にまだアプライされていない場合は、ターゲット表をリフレッシュして、ソースと同期化させる必要があります。比較的高い保持制限を使用すれば、フル・リフレッシュが発生するのを回避できます。ただし、使用している CD 表および UOW 表が大きくなり、システム上のスペースを使用します。

Update-anywhere レプリケーションを実行している場合は、保持制限整理により、リジェクトされたトランザクションが確実に削除されます。レプリカ・ターゲット表に対して競合検出を使用している場合、競合しているトランザクションが検出されると、結果的にトランザクションがリジェクトされます。それらのリジェクトされたトランザクションと関係のある CD 表および UOW 表の行は複製されず、保持制限に達した時点で整理されます。リジェクトされたトランザクションと関係のあるすべての古い行が削除された場合は、フル・リフレッシュは不要です。

また、保持整理により、もはや不要となったシグナル情報も IBMSNAP_SIGNAL 表から確実に削除されます。

sleep_interval

デフォルト: **sleep_interval=**5

スリープ・インターバルは、キャプチャー・プログラムがログの終わりに到達し、バッファーが空になった後、キャプチャー・プログラムは何秒待ってから再びログを読み取るかという、その秒数を表します。 z/OS オペレーティング・システムで

のデータ共有の場合は、スリープ・インターバルは、バッファーがその全容量の半分より少ない領域しか使用されていない状態に戻った後、キャプチャー・プログラムがスリープする秒数を表します。

デフォルトでは、キャプチャー・プログラムは 5 秒間スリープします。キャプチャー・プログラムがログを読み取るために生じるオーバーヘッドを削減する場合は、スリープ・インターバルを変更してください。スリープ・インターバルの値を小さくするということは、その分遅延が生じる機会が少なくなるということを意味します。スリープ・インターバルの値を大きくすると、頻繁には更新が行われないシステムで CPU 使用量を節約できる可能性があります。

startmode

デフォルト: **startmode**=warmsi

キャプチャーは、以下の始動モードのいずれかを使用して始動できます。

warmsi (ウォーム・スタート、初回はコールド・スタートに切り替える)
キャプチャー・プログラムはウォーム・スタートします。ただし、今回はじめてキャプチャー・プログラムを始動する場合は、コールド・スタートに切り替えられます。この始動モードは、コールド・スタートになるのは必ずキャプチャー・プログラムの最初の始動時だけにする場合に使用してください。

warmns (ウォーム・スタート、決してコールド・スタートに切り替えない)
キャプチャー・プログラムはウォーム・スタートします。ウォーム・スタートできない場合、コールド・スタートに切り替わりません。 **warmns** を日常のレプリケーション環境で使用する場合は、ウォーム・スタートの実行を妨げているあらゆる問題 (データベースや表スペースが使用不能など) を修復する機会があります。この始動モードを使用して、予期せぬコールド・スタートが実行されるのを防いでください。ウォーム・スタートした場合、キャプチャー・プログラムは終了したところから処理を再開します。キャプチャー・プログラムの始動後にエラーが発生した場合は、キャプチャー・プログラムは終了し、すべての表はそのまま残されます。

ヒント: **warmns** は、キャプチャー・プログラムをはじめて始動するときには使用できません。なぜなら、キャプチャー・プログラムの最初の始動時にはウォーム・スタート情報がないからです。キャプチャー・プログラムの最初の始動時に **cold** 始動モードを使用してから、**warmns** 始動モードを使用してください。始動モードを切り替えたくない場合は、代わりに **warmsi** を使用できます。

cold コールド・スタート時には、キャプチャー・プログラムは初期化の間に CD 表および UOW 表のすべての行を削除します。これらのレプリケーション・ソースへのすべてのサブスクリプション・セットは、次のアプライ処理サイクルの間に完全にリフレッシュされます (つまり、すべてのデータがソース表からターゲット表にコピーされます)。キャプチャー・プログラムがコールド・スタートを試行したときにフル・リフレッシュが使用不可にされていた場合、キャプチャー・プログラムは始動しますが、アプライ・プログラムは失敗して、エラー・メッセージを発行します。

キャプチャー・プログラムがコールド・スタートするよう明示的に要求する必要がある場合はめったにありません。コールド・スタートが必要となるのは、キャプチャー・プログラムの最初の始動時、および **warmsi** が推奨始動モードである場合だけです。

重要: 変更データの履歴を正確なものにしておきたい場合は、キャプチャー・プログラムのコールド・スタートは行わないでください。キャプチャー・プログラムをシャットダウンされた後でアプライ・プログラムが変更をレプリケーションできないと、ギャップが生じることがあります。 また、コールド・スタートを避けたい以上、 IBMSNAP_CAPPARMS 表で STARTMODE のデフォルトとしてコールド・スタート (cold) を指定しないでください。

term

デフォルト: **term**=y

term パラメーターにより、DB2 の状況がキャプチャー・プログラムの動作にどのように影響するかが決定されます。

デフォルトでは、キャプチャー・プログラムは DB2 が終了すると終了します。

term=n は、DB2 がアクティブでない場合に DB2 が始動するのをキャプチャー・プログラムが待つようにする場合に使用してください。DB2 が静止している場合は、キャプチャーは終了しません。これは、キャプチャーはアクティブなままであるが、データベースは使用しないということです。

trace_limit

デフォルト: **trace_limit**10080

trace_limit は、整理の対象にする前に、行がどのくらい古くなるまで IBMSNAP_CAPTRACE 表に残しておく必要があるのかを指定します。

キャプチャーが整理を実行する場合、デフォルトでは、 IBMSNAP_CAPTRACE 表内の行は 10080 分 (7 日) ごとに整理の対象として適格となります。CAPTRACE 表には、キャプチャー・プログラム用の監査証跡情報が含まれています。キャプチャーが実行した内容はすべてこの表に記録されます。そのため、キャプチャー・プログラムが非常にアクティブであると、この表が急速に大きくなる可能性があります。監査情報に関するニーズに合わせてトレース限度を変更してください。

キャプチャー・パラメーターの変更方法

キャプチャーの操作パラメーターの保管済みの値を変更することも、プログラムの始動時または実行中にこれらの値を一時的にオーバーライドすることも可能です。

IBMSNAP_CAPPARMS 表での新しいデフォルト値の設定

IBMSNAP_CAPPARMS 表には、キャプチャー・プログラムの動作を制御するために変更できるパラメーターが含まれています。この表のスキーマ名はキャプチャー・スキーマです。この表が作成されると、その中にはキャプチ

ャー・プログラム用の出荷時のデフォルト値が入れられます。
IBMSNAP_CAPPARMS 表の列値が設定されていない場合は、デフォルト値が使用されます。

キャプチャー・プログラムの始動時に行うパラメーター値の指定

キャプチャー・プログラムの始動時に、このプログラム用の値を指定できます。始動時に設定された値により、現行セッションの間のキャプチャーの動作がコントロールされ、デフォルトの稼働パラメーター値およびキャプチャー・パラメーター表内に存在する可能性のあるあらゆる値がオーバーライドされます。キャプチャー・パラメーター表内の値については、これらの値による更新は行われません。キャプチャー・プログラムを始動する前にキャプチャー・パラメーター表を変更せず、さらにキャプチャー・プログラムの始動時にパラメーターをまったく指定しなかった場合は、稼働パラメーターに対してデフォルト値が使用されます。

キャプチャー・プログラムの実行中に行うパラメーター値の変更

キャプチャーの実行中に、その稼働パラメーターを一時的に変更できます。キャプチャー・プログラムは、ユーザーが値を再び変更するまで、またはキャプチャー・プログラムを停止して再始動するまで、その新しい値を使用します。キャプチャー・パラメーターは、セッションの間必要なだけ何度でも変更できます。

例 1 Linux UNIX Windows

キャプチャー・スキーマ ASNPROD のキャプチャー・コミット・インターバルのデフォルト設定を使用しない場合。

1. ASNPROD キャプチャー・スキーマ用のキャプチャー・パラメーター表を更新する。コミット・インターバルを 60 秒に設定してください。これにより、今後のキャプチャー・プログラムの始動時にはデフォルト・コミット・インターバルが 60 秒に設定されます。

   ```
   update asnprod.ibmsnap_capparms set commit_interval=60;
   ```

2. 場合によっては、パフォーマンス・チューニングを行うことができます。その場合、比較的小さな値を指定したコミット・インターバルを使用してキャプチャーを始動してみてください。キャプチャー・パラメーター表内の値を変更する代わりに、単に 20 秒に設定したコミット・インターバルのパラメーターを使用してキャプチャー・プログラムを始動してください。キャプチャー・プログラムが 20 秒のコミット・インターバルを使用して稼働している間、このプログラムのパフォーマンスをモニターします。

   ```
   asncap capture_server=srcdb1 capture_schema=asnprod commit_interval=20
   ```

3. 値をさらに小さくしたコミット・インターバルを試してみる。キャプチャー・プログラムを停止する代わりに、コミット・インターバルを 15 秒に設定するパラメーターの変更要求をサブミットしてください。キャプチャー・プログラムは稼働を継続しますが、ただし 15 秒ごとにデータをコミットするようになります。

   ```
   asnccmd capture_server=srcdb1 capture_schema=asnprod chgparms
   commit_interval=15
   ```

   重要: 変更するパラメーターは、**chgparms**の直後に指定する必要があります。

4. 引き続きパフォーマンスのモニターとコミット・インターバルのパラメーターの変更を行う。キャプチャー・プログラムを停止する必要はありません。最終的

に、ニーズに合ったコミット・インターバルを見つけたときは、キャプチャー・パラメーター表を (ステップ 1 (123 ページ) で説明されている方法で) 更新して、次回の始動時にキャプチャー・プログラムがその新しい値をデフォルト・コミット・インターバルとして使用するようにできます。

例 2 System i

キャプチャー・スキーマ ASNPROD のキャプチャー・コミット・インターバルのデフォルト設定を使用しない場合。

1. ASNPROD キャプチャー・スキーマ用のキャプチャー・パラメーター表を更新する。コミット・インターバルを 90 秒に設定してください。これにより、今後のキャプチャー・プログラムの始動時にはデフォルト・コミット・インターバルが 90 秒に設定されます。

   ```
   CHGDPRCAPA CAPCTLLIB(ASNPROD) FRCFRQ(90)
   ```

2. 場合によっては、パフォーマンス・チューニングを行うことができます。その場合、比較的小さな値を指定したコミット・インターバルを使用してキャプチャーを始動してみてください。キャプチャー・パラメーター表内の値を変更する代わりに、45 秒に設定したコミット・インターバルのパラメーターを使用してキャプチャー・プログラムを始動してください。キャプチャー・プログラムが 45 秒のコミット・インターバルを使用して稼働している間、このプログラムのパフォーマンスをモニターしてください。

   ```
   STRDPRCAP CAPCTLLIB(ASNPROD) FRCFRQ(45)
   ```

3. 値をさらに小さくしたコミット・インターバルを試してみる。キャプチャー・プログラムを停止する代わりに、コミット・インターバルを 30 秒に設定するパラメーターの変更要求をサブミットしてください。キャプチャー・プログラムは稼働を継続しますが、ただし 30 秒ごとにデータをコミットするようになります (注: System i では、30 秒より短いコミット・インターバルは設定できません)。

   ```
   OVRDPRCAPA CAPCTLLIB(ASNPROD) FRCFRQ(30)
   ```

4. 最終的に、ニーズに合ったコミット・インターバルを見つけたときは、キャプチャー・パラメーター表を (ステップ 1 で説明されている方法で) 更新して、次回の始動時にキャプチャー・プログラムがその新しい値をデフォルト・コミット・インターバルとして使用するようにできます。

実行中のキャプチャー・プログラムの動作の変更

キャプチャーの 1 つ以上の操作パラメーターの値を動的に変更できます。その変更は IBMSNAP_CAPPARMS 表には保管されませんが、キャプチャー・プログラムを停止するか新しい値を指定するまで、使用されます。

このタスクについて

z/OS Linux UNIX Windows キャプチャー・プログラムの実行中に、以下のキャプチャー・パラメーターを変更できます。

- **autoprune**
- **autostop**
- **commit_interval**

- **lag_limit**
- **logreuse**
- **logstdout**
- **memory_limit**
- **monitor_interval**
- **monitor_limit**
- **prune_interval**
- **retention_limit**
- **sleep_interval**
- **term**
- **trace_limit**

制約事項: z/OS メッセージを作成するためにキャプチャー・プログラムが使用できるメモリーの量は、JCL で指定されている **memory_limit** パラメーターおよび REGION のサイズの値に基づいて、キャプチャー・プログラムの開始時に決定されます。**memory_limit** の値は、キャプチャー・プログラムの実行中は変更することができません。値を変更するには、まずキャプチャー・プログラムを停止してください。

System i 所定のキャプチャー・スキーマについて、以下の稼働パラメーターの値をオーバーライドできます。

- **CLNUPITV**
- **FRCFRQ**
- **MEMLMT**
- **MONLMT**
- **MONITV**
- **PRUNE**
- **RETAIN**
- **TRCLMT**

値を変更した場合、その影響がすべてのパラメーターに即時に及ぶとは限りません。

手順

実行中のキャプチャー・プログラムの動作を変更するには、次の方法のいずれかを使用します。

方法	説明
レプリケーション・センター	「実行中のキャプチャー・プログラムのパラメーターの変更」ウィンドウを使用します。この方法を使用すると、パラメーターの現行値を変更前に参照できます。そのウィンドウを開くには、オブジェクト・ツリーの 「**操作**」ブランチを開き、「**キャプチャー・コントロール・サーバー**」をクリックし、内容ペインでキャプチャー・コントロール・サーバーを右クリックし、 「**パラメーターの変更**」 → 「**実行中のキャプチャー・プログラム (Running Capture Program)**」をクリックします。
z/OS Linux UNIX Windows asnccmd chgparms システム・コマンド	この方法では、パラメーターの現行値は表示されません。
System i OVRDPRCAPA システム・コマンド	実行中のキャプチャー・プログラムの動作を変更するには、 DPR キャプチャー属性のオーバーライド (OVRDPRCAPA) コマンドを使用します。

IBMSNAP_CAPPARMS 表に保管されている操作パラメーターの変更

IBMSNAP_CAPPARMS 表には、キャプチャー・プログラムの操作パラメーターが保管されています。キャプチャー・プログラムを始動すると、始動パラメーターを使用してその表の値を一時的にオーバーライドした場合や、プログラムの実行中にその表の値を一時的にオーバーライドした場合を除き、キャプチャー・プログラムはその表の値を使用します。

このタスクについて

IBMSNAP_CAPPARMS 表には 1 つの行しか入れられません。デフォルト値のいずれかを変更する場合は、行を挿入する代わりに列を更新することができます。この行を削除しても、デフォルトが始動パラメーターによってオーバーライドされないかぎり、キャプチャー・プログラムはそれらのデフォルトを使用して始動します。

キャプチャー・プログラムがこの表を読み取るのは、始動時だけです。キャプチャー・プログラムの実行中にキャプチャー・パラメーター表を変更し、キャプチャー・プログラムを再初期化しても、キャプチャー・プログラムの操作は変更されません。

手順

IBMSNAP_CAPPARMS 表に保管されているパラメーターを変更するには、次の方法のいずれかを使用します。

方法	説明
レプリケーション・センター	「パラメーターの変更 - 保管済み」ウィンドウを使用します。そのウィンドウを開くには、オブジェクト・ツリーの「**操作**」ブランチを開き、「**キャプチャー・コントロール・サーバー**」をクリックし、内容ペインでキャプチャー・コントロール・サーバーを右クリックし、「**パラメーターの変更**」 → 「**保管**」をクリックします。
System i CHGDPRCAPA システム・コマンド	DPR キャプチャー属性の変更 (CHGDPRCAPA) コマンドは、キャプチャー・プログラムにより使用され、IBMSNAP_CAPPARMS 表に保管されるグローバル操作パラメーターを変更するために使用されます。

キャプチャー・プログラムの停止と始動を行った後でのみ、パラメーターの変更は有効になります。

キャプチャー・プログラムの停止

特定のキャプチャー・スキーマ用のキャプチャー・プログラムを停止できます。キャプチャー・プログラムを停止すると、それ以降当該のソースからはデータがキャプチャーされなくなります。

このタスクについて

System i キャプチャー・プログラムが停止した時点において開かれた状態にあった UOW 表とすべての CD 表を再編成する場合、キャプチャー・プログラムはシャットダウンに時間を要します (このプログラムは即時にはシャットダウンしません)。

手順

キャプチャー・プログラムを停止するには、次の方法のいずれかを使用します。

方法	説明
レプリケーション・センター	「キャプチャーの停止」ウィンドウを使用します。そのウィンドウを開くには、オブジェクト・ツリーの「**操作**」ブランチを開き、「**キャプチャー・コントロール・サーバー**」をクリックし、内容ペインでキャプチャー・コントロール・サーバーを右クリックし、「**キャプチャーの停止**」をクリックします。
z/OS Linux UNIX Windows asnccmd stop システム・コマンド	このコマンドを使用して、キャプチャーを停止します。
System i ENDDPRCAP システム・コマンド	DPR キャプチャー・プログラムの終了 (ENDDPRCAP) コマンドを使用して、キャプチャー・プログラムを停止します。

整理中にキャプチャー・プログラムを停止または中断した場合は、整理も中断されます。キャプチャー・プログラムを再開または再始動すると、整理は **autoprune** パラメーターに基づいて再開されます。

登録をドロップするためにキャプチャー・プログラムを停止する必要はありません。登録をドロップするときには、必ずその前にその登録を非活動化してください。

キャプチャーの再初期化

キャプチャー・プログラムの実行中に既存の登録済みオブジェクトの属性を変更する場合は、必ずキャプチャー・プログラムを再初期化してください。

このタスクについて

例えば、IBMSNAP_REGISTER 表内の値 CONFLICT_LEVEL、 CHGONLY、RECAPTURE、 CHG_UPD_TO_DEL_INS などを変更する場合は、キャプチャー・プログラムを再初期化する必要があります。

System i でのキャプチャーの場合は、以前キャプチャーされていなかったジャーナルを対象としてデータのキャプチャーを開始する場合にも再初期化が必要です。

手順

キャプチャー・プログラムを再初期化するには、次の方法のいずれかを使用します。

方法	説明
レプリケーション・センター	「キャプチャーの再初期化」ウィンドウを使用します。そのウィンドウを開くには、オブジェクト・ツリーの「**操作**」ブランチを開き、「**キャプチャー・コントロール・サーバー**」をクリックし、内容ペインでキャプチャー・コントロール・サーバーを右クリックし、「**キャプチャーの再初期化**」をクリックします。
z/OS Linux UNIX Windows asnccmd reinit システム・コマンド	このコマンドを使用して、キャプチャーを再初期化します。
System i INZDPRCAP システム・コマンド	DPR キャプチャー・プログラムの初期化 (INZDPRCAP) コマンドを使用して、キャプチャー・プログラムを初期化します。

キャプチャー・プログラムの中断 (Linux、UNIX、Windows、z/OS)

キャプチャー・プログラムを中断すると、キャプチャー・プログラム環境を損なうことなく、ピーク時にオペレーティング・システムのリソースを解放できます。

始める前に

特定のキャプチャー・スキーマを指定したキャプチャー・プログラムを始動しておく必要があります。

このタスクについて

進行中の作業の終了後にキャプチャー・プログラムをシャットダウンしたくない場合にも、キャプチャー・プログラムを停止する代わりに中断できます。キャプチャーを再開する場合には、キャプチャーを再始動する場合に必然的に生じるオーバーヘッドが生じません。

重要: レプリケーション・ソースを除去する前に、キャプチャー・プログラムを中断しないでください。その代わりに、レプリケーション・ソースを非活動化してから除去してください。

手順

キャプチャー・プログラムを中断するには、次の方法のいずれかを使用します。

方法	説明
レプリケーション・センター	「キャプチャーの中断」ウィンドウを使用します。そのウィンドウを開くには、オブジェクト・ツリーの「**操作**」ブランチを開き、「**キャプチャー・コントロール・サーバー**」をクリックし、内容ペインでキャプチャー・コントロール・サーバーを右クリックし、「**キャプチャーの中断**」をクリックします。
z/OS Linux UNIX Windows asnccmd suspend システム・コマンド	このコマンドを使用して、キャプチャーを中断します。

整理中にキャプチャー・プログラムを停止または中断した場合は、整理も中断されます。キャプチャー・プログラムを再開または再始動すると、整理は **autoprune** パラメーターに基づいて再開されます。

キャプチャーの再開 (Linux、UNIX、Windows、z/OS)

中断されたキャプチャー・プログラムに再びデータのキャプチャーを開始させる場合は、このプログラムを再始動する必要があります。

手順

中断したキャプチャー・プログラムを再開するには、次の方法のいずれかを使用します。

方法	説明
レプリケーション・センター	「キャプチャーの再開」ウィンドウを使用します。そのウィンドウを開くには、オブジェクト・ツリーの「**操作**」ブランチを開き、「**キャプチャー・コントロール・サーバー**」をクリックし、内容ペインでキャプチャー・コントロール・サーバーを右クリックし、「**キャプチャーの再開**」をクリックします。

方法	説明
z/OS Linux UNIX Windows asnccmd resume システム・コマンド	このコマンドを使用して、キャプチャーを再開します。

整理中にキャプチャー・プログラムを停止または中断した場合は、整理も中断されます。キャプチャー・プログラムを再開または再始動すると、整理は **autoprune** パラメーターに基づいて再開されます。

キャプチャー・プログラムにトランザクションを無視するように指示する

キャプチャー・プログラムに、DB2 リカバリー・ログ内のトランザクションを無視するよう指示することができます。これにより、これらのトランザクションはキャプチャーされなくなります。

このタスクについて

以下の 1 つ以上の ID を使用して、無視するトランザクションを指定できます。

- トランザクション ID
- 許可 ID
- z/OS 許可トークン
- z/OS プラン名

トランザクション ID に基づいてトランザクションを無視するようキャプチャー・プログラムに指示するには、キャプチャー・プログラムの開始時に asncap コマンド・パラメーターを使用します。

許可 ID、許可トークン、またはプラン名に基づいてトランザクションを無視するには、SQL を使用して、その ID をキャプチャー・サーバーの IBMQREP_IGNTRAN コントロール表に挿入します。

手順

キャプチャー・プログラムにトランザクションを無視するよう指示するには、使用する ID に応じて、以下の方法のいずれかを使用します。

ID	手順

トランザクション ID	**ignore_transid** パラメーターを指定した asncap コマンドを使用して、無視する 1 つのトランザクションを指定します。コマンドの形式は以下のとおりです。
	
	`asnqap capture_server=q_capture_server`
	`capture_schema=q_capture_schema`
	`ignore_transid=transaction_ID`
	
	ここで、*transaction_ID* は 10 バイトの 16 進 ID で、形式は以下のとおりです。
	
	z/OS
	
	0000:*xxxx*:*xxxx*:*xxxx*:*mmmm*
	
	ここで、*xxxx*:*xxxx*:*xxxx* はトランザクション ID、*mmmm* はデータ共用メンバー ID です。メンバー ID は、LOGP 出力内にあるログ・レコード・ヘッダーの末尾の 2 バイトにあります。データ共用が有効になっていない場合、メンバー ID は 0000 です。
	
	例えば、以下のコマンドは、データ共用環境で、つまりメンバー ID に 0001 を指定して 1 つのトランザクションを無視することを指定します。
	
	`asncap capture_server=sample`
	`capture_schema=ASN`
	`ignore_transid=0000:BD71:1E23:B089:0001`
	
	Linux UNIX Windows
	
	nnnn:0000:*xxxx*:*xxxx*:*xxxx*
	
	ここで、*xxxx*:*xxxx*:*xxxx* はトランザクション ID、*nnnn* はパーティション・データベースのパーティション ID です (非パーティション・データベースの場合、この値は 0000 です)。
	
	例えば、以下のコマンドは、非パーティション・データベースで 1 つのトランザクションを無視することを指定します。
	
	`asncap capture_server=sample`
	`capture_schema=ASN`
	`ignore_transid=0000:0000:0000:0000:BE97`
	
	複数のトランザクションを無視するには、キャプチャー・プログラムを停止し、別のトランザクション ID を指定してからキャプチャー・プログラムをウォーム・モードで開始します。

ID	手順

許可 ID、許可トークン (z/OS)、またはプラン名 (z/OS)	SQL を使用して 1 つ以上の ID を IBMQREP_IGNTRAN コントロール表に挿入します。適切な ID を以下の列に挿入します。
	AUTHID
	許可 ID。
	z/OS **AUTHTOKEN**
	許可トークン (ジョブ名)。
	z/OS **PLANNAME**
	プラン名。
	例えば、以下のステートメントは、*repldba* という許可 ID を持つトランザクションをキャプチャー・プログラムが無視することを指定します。
	`insert into schema.IBMQREP_IGNTRAN (`
	`  AUTHID,`
	`  AUTHTOKEN,`
	`  PLANNAME)`
	`values (`
	`  'repldba',`
	`  NULL,`
	`  NULL);`
	z/OS 以下のステートメントは、許可トークン IGN1 およびプラン名 ZPLAN を持つトランザクションが z/OS 上で無視されることを指定します。
	`insert into schema.IBMQREP_IGNTRAN (`
	`  AUTHID,`
	`  AUTHTOKEN,`
	`  PLANNAME)`
	`values (`
	`  NULL,`
	`  'IGN1',`
	`  'ZPLAN');`

キャプチャー・プログラムは、トランザクションを無視する際に、無視されたトランザクションの ID を記録する IBMQREP_IGNTRANTRC 表に行を挿入します。

第 10 章 SQL レプリケーションに関するアプライ・プログラムの操作

アプライ・プログラムの操作には、開始、停止、ASNDONE および ASNLOAD 出口ルーチンの使用などのタスクが含まれます。

アプライ・プログラムの始動 (Linux、UNIX、Windows、z/OS)

アプライ・プログラムのインスタンスを始動して、ターゲットへのデータのアプライを開始できます。

始める前に

以下の点を確認してください。

- 必要なレプリケーション・サーバーのすべてへの接続が構成されている。
- 正しい許可を受けている。
- 必要なアプライ修飾子に対してソースおよびコントロール・データを含むコントロール表が作成されている。
- レプリケーション・プログラムが構成済みである。
- z/OS 必要なすべてのサーバーにアプライ・プログラムが手動でバインドされている。
- Linux UNIX Windows リモート・サーバーに対するエンド・ユーザー認証用のパスワード・ファイルが存在している。

また、以下の条件が満たされていることを確認してください。

- アプライ修飾子に対して少なくとも 1 つのアクティブなサブスクリプション・セットが存在し、サブスクリプション・セットが 1 つまたは複数の以下の項目を含んでいる必要があります。
  - サブスクリプション・セット・メンバー
  - SQL ステートメント
  - 手順
- コンデンスされているターゲット表はすべてターゲット・キーを必要とします。ターゲット・キーは、アプライ・プログラムが各アプライ・サイクル中に複製する変更のトラッキングに使用する主キーまたはユニーク索引のいずれかである、ユニーク列のセットです。 (非コンデンス CCD 表には主キーまたはユニーク索引はありません。)

このタスクについて

アプライ・プログラムを始動するときには、始動パラメーターも指定できます。

手順

アプライ・プログラムを始動するには、次のようにします。

以下のいずれかの方法を使用します。

オプション	説明
レプリケーション・センター	「アプライの開始」ウィンドウを使用します。このウィンドウを開くには、オブジェクト・ツリーの 「**操作**」ブランチから「**アプライ・コントロール・サーバー**」フォルダーを開き、「**アプライ修飾子**」フォルダーをクリックします。内容ペインで、始動するアプライ・プログラムを表すアプライ修飾子を右クリックして、「**アプライの開始**」をクリックします。
z/OS Linux UNIX Windows asnapply システム・コマンド	このコマンドを使用して、アプライを始動します。
z/OS z/OS コンソールまたは TSO	z/OS では、JCL を使用するか、またはシステム開始タスクとして、アプライ・プログラムを始動できます。JCL を使用してアプライ・プログラムを始動する場合は、新しい呼び出しパラメーター値を指定できます。これらの値は、IBMSNAP_APPPARMS 表に保管されているパラメーター値をオーバーライドします。EXEC ステートメントの PARM パラメーターには、100 文字を超えるサブパラメーターを指定できません。
Windows Windows サービス	Windows オペレーティング・システム上で DB2 レプリケーション・サービスを作成して、システムの始動時に自動的に Q アプライ・プログラムを始動できます。

アプライ・プログラムは始動された後、以下のイベントのいずれかが発生するまで、連続的に実行されます (**copyonce** 始動パラメーターが使用されない場合)。

- ユーザーがレプリケーション・センターまたはコマンドを使用してアプライ・プログラムを停止する。
- アプライ・プログラムがアプライ・コントロール・サーバーに接続できない。
- アプライ・プログラムは処理用のメモリーを割り振ることができない。

アプライ・プログラムが始動済みかどうかを検証するには、次の方法のいずれかを使用します。

- z/OS バッチ・モードで実行している場合は、z/OS コンソールまたは z/OS ジョブ・ログを調べて、プログラムが始動したことを示すメッセージを探します。
- アプライ診断ログ・ファイル (z/OS では *apply_server.apply_qualifier*.APP.log、Linux、UNIX、および Windows では *db2instance.apply_server.apply_qualifier*.APP.log) を検査して、プログラムが変更をキャプチャーしていることを示すメッセージについて調べます。
- IBMSNAP_APPLYTRACE 表を調べて、プログラムが変更を適用していることを示すメッセージを探します。
- レプリケーション・センター内の「アプライ・メッセージ」ウィンドウを使用して、プログラムが始動したことを示すメッセージを探します。そのウィンドウを

オープンするには、内容ペインで、メッセージを表示する対象のアプライ・プログラムを識別するアプライ修飾子を右クリックし、「**レポート**」 → 「**アプライ・メッセージ**」を選択します。

- レプリケーション・センターの「状況のチェック」ウィンドウまたは asnacmd 状況コマンドを使用して、すべてのアプライ・スレッドの状況を表示します。そのウィンドウをオープンするには、内容ペインで、チェックする対象のアプライ・プログラムを識別するアプライ修飾子を右クリックし、「**状況のチェック**」を選択します。

アプライ・プログラムの始動 (System i)

アプライ・プログラムのインスタンスを始動して、ターゲットへのデータのアプライを開始できます。

始める前に

システムが正しくセットアップされていることを確認してください。

- すべてのレプリケーション・サーバーへの接続が構成されている。
- 正しい許可を受けている。
- コントロール表が作成されている。
- レプリケーション・プログラムが構成済みである。

また、以下の条件が満たされていることを確認してください。

- アプライ修飾子に対して少なくとも 1 つのアクティブなサブスクリプション・セットが存在し、サブスクリプション・セットが、1 つ、または複数の以下の項目を含んでいる必要があります。
  - サブスクリプション・セット・メンバー
  - SQL ステートメント
  - 手順
- コンデンスされているターゲット表はすべてターゲット・キーを必要とします。ターゲット・キーは、アプライ・プログラムが各アプライ・サイクル中に複製する変更のトラッキングに使用する主キーまたはユニーク索引のいずれかである、ユニーク列のセットです。 (非コンデンス CCD 表には主キーまたはユニーク索引はありません。)

手順

アプライ・プログラムを始動するには、次の方法のいずれかを使用します。

方法	説明
STRDPRAPY システム・コマンド	DPR アプライ・プログラムの始動 (STRDPRAPY) コマンドは、ローカル・システム上でアプライ・プログラムを始動するために使用されます。

方法	説明
レプリケーション・センター	「アプライの開始」ウィンドウを使用します。このウィンドウを開くには、オブジェクト・ツリーの「**操作**」ブランチから「**アプライ・コントロール・サーバー**」フォルダーを開き、「**アプライ修飾子**」フォルダーをクリックします。内容ペインで、始動するアプライ・プログラムを表すアプライ修飾子を右クリックして、「**アプライの開始**」をクリックします。

アプライ・プログラムは始動された後、以下の条件のいずれか発生するまで、連続的に実行されます。

- ユーザーが COPYONCE(*YES) 始動パラメーターを使用してプログラムを始動する。
- ユーザーが ALWINACT(*NO) を指定し、処理するべきデータがない。
- ユーザーがレプリケーション・センターまたはコマンドを使用してアプライ・プログラムを停止する。
- アプライ・プログラムがアプライ・コントロール・サーバーに接続できない。
- アプライ・プログラムは処理用のメモリーを割り振ることができない。

アプライ・プログラムのデフォルトの操作パラメーター

アプライ・コントロール表を作成すると、アプライの操作パラメーターのデフォルト値が IBMSNAP_APPPARMS 表に保管されます。

表 7と 137 ページの表 8にデフォルト値を示します。 z/OS Linux UNIX Windows

表 7. アプライ稼働パラメーターのデフォルト設定 (z/OS、Linux、UNIX、Windows)

稼働パラメーター	デフォルト値	IBMSNAP_APPPARMS 表の列名
apply_qual	デフォルトなし	APPLY_QUAL
apply_path	アプライが始動されたディレクトリー [1]	APPLY_PATH
control_server	DB2DBDFT[2]	該当せず
copyonce	n[3]	COPYONCE
db2_subsystem	デフォルトなし[4]	該当せず
delay	6 秒	DELAY
errwait	300 秒	ERRWAIT
inamsg	y[5]	INAMSG
loadxit	n[3]	LOADXIT
logreuse	n[3]	LOGREUSE
logstdout	n[3]	LOGSTDOUT
notify	n[3]	NOTIFY
opt4one	n[3]	OPT4ONE
pwdfile	asnpwd.aut	該当せず
spillfile	disk[6]	SPILLFILE

表 7. アプライ稼働パラメーターのデフォルト設定 (z/OS、Linux、UNIX、Windows) (続き)

稼働パラメーター	デフォルト値	IBMSNAP_APPPARMS 表の列名
sleep	y[5]	SLEEP
sqlerrcontinue	n[3]	SQLERRCONTINUE
term	y[5]	TERM
trlreuse	n[3]	TRLREUSE

注:

1. アプライが Windows サービスとして始動する場合、そのパスは sqllib¥bin である。
2. アプライ・コントロール・サーバーは、DB2DBDFT 環境変数の値である (指定されている場合)。 Linux、UNIX、Windows の各オペレーティング・システムの場合に限る。
3. no
4. DB2 サブシステム名は最大 4 文字。このパラメーターは必須です。DB2 サブシステム名は、z/OS オペレーティング・システムにのみ適用される。
5. yes
6. z/OS オペレーティング・システムでは、デフォルトは MEM である。

System i

表 8. アプライ稼働パラメーターのデフォルト設定 (System i)

稼働パラメーター	(*value) の記述
USER (*CURRENT)	システムにサインオンしたユーザー。
JOBD (*LIBL/QZSNDPR)	製品ライブラリー名/ジョブ記述。
APYQUAL (*USER)	現行ユーザー名 (上記から)。
CTLSVR (*LOCAL)	ローカル RDB サーバー名。
TRACE (*NONE)	トレースを生成しない。
FULLREFPGM (*NONE)	ASNLOAD 出口ルーチンを実行しない。
SUBNFYPGM (*NONE)	ASNDONE 出口ルーチンを実行しない。
INACTMSG (*YES)	アプライ・プログラムは非アクティブ期間を開始すると、プログラムが非アクティブになる期間を示すメッセージ ASN1044 を生成する。
ALWINACT (*YES)	処理するものがない場合はスリープする。
DELAY (6)	再度処理する前に、アプライ・サイクルの後に 6 秒待機する。
RTYWAIT (300)	失敗した操作を再始動する前に 300 秒待機する。
COPYONCE (*NO)	1 つのコピー・サイクルを完了した後で終了せずに処理を続行する。
TRLREUSE (*NO)	アプライ・プログラムの始動時に IBMSNAP_APPLYTRAIL 表を空にしない。
OPTSNGSET (*NO)	アプライ・プログラムのパフォーマンスを 1 つのサブスクリプション・セットの処理用に最適化しない。

アプライの操作パラメーターの説明

アプライ・プログラムの始動時に、始動パラメーターを選択することもできます。ここでは、それぞれの始動パラメーターを取り上げ、各パラメーターで選択するべき値に関する推奨事項を示します。

以下のパラメーターは、特に指定がない限り、z/OS、Linux、UNIX、および Windows に適用されます。

- 『apply_path』
- 139 ページの『apply_qual』
- 139 ページの『control_server』
- 140 ページの『copyonce』
- 140 ページの『db2_subsystem (z/OS)』
- 140 ページの『delay』
- 141 ページの『errwait』
- 141 ページの『inamsg』
- 141 ページの『loadxit』
- 142 ページの『logreuse』
- 142 ページの『logstdout』
- 142 ページの『notify』
- 143 ページの『opt4one』
- 143 ページの『pwdfile』
- 143 ページの『sleep』
- 144 ページの『spillfile』
- 144 ページの『sqlerrcontinue』
- 145 ページの『term』
- 146 ページの『trlreuse』

apply_path

Linux UNIX Windows **デフォルト: apply_path=***current_directory*

Windows **デフォルト (Windows 上のサービス): apply_path**sqllib¥bin

アプライ・パスは、アプライ・プログラムがログおよび作業ファイルを保管するディレクトリーです。デフォルトのアプライ・パスはプログラムを始動するディレクトリーです。ログおよび作業ファイルを他の場所に保管するには、アプライ・パスを変更します (例えば、AIX® システムの場合は /home/db2inst/apply_files)。アプライ・ログ・ファイルにアクセスするには、このディレクトリーに移動する必要があるため、どのディレクトリーを選択したか覚えておいてください。

z/OS アプライ・パスを変更する方法に関する詳細は、SASNSAMP(ASNSTRA) ジョブを参照してください。

重要: 選択したディレクトリーに、アプライ・プログラムから使用される一時ファイルに十分なスペースがあることを確認してください。

Windows **Windows システムでのアプライ・プログラムのインスタンスの始動**: レプリケーション・センターまたは asnapply コマンドを使用してアプライ・プログラムを始動する場合に、もし大文字と小文字の区別以外は同じ複数のアプライ修飾子を使用するのであれば、必ずアプライ・パスを指定してください。Windows システムのファイル名は大文字小文字が区別されません。例えば、APPLYQUAL1、 ApplyQual1、applyqual1 という 3 つのアプライ修飾子があるとします。アプライ・プログラムの各インスタンスのログ・ファイルのファイル名が競合しないように、これらのアプライ・インスタンスはそれぞれ異なる **apply_path** を使用して開始する必要があります。

apply_qual

処理対象のサブスクリプション・セットのアプライ修飾子を指定する必要があります。 (アプライ修飾子は、サブスクリプション・セットを作成するときに定義します。) 1 つの始動コマンドで指定できるアプライ修飾子は 1 つだけです。

重要: アプライ修飾子では大文字小文字が区別されます。入力された値が、IBMSNAP_SUBS_SET 表の APPLY_QUAL 列の値と一致する必要があります。

複数のアプライ修飾子を定義してあれば、アプライ・プログラムの別のインスタンスを始動できます。始動されたアプライ・プログラムの各インスタンスは、同じアプライ・コントロール・サーバーで表される異なるサブスクリプション・セットを処理します。例えば、2 つのサブスクリプション・セットが定義され、それぞれのセットにユニークの修飾子、APPLY1 および APPLY2 があるとします。アプライ・プログラムの 2 つのインスタンス (各修飾子に対して 1 つずつ) を始動できます。各インスタンスは、 CNTRLSVR と呼ばれるアプライ・コントロール・サーバーにあるコントロール表を使用します。アプライ・プログラムの各インスタンスは、各自のサブスクリプション・セットを別個に処理するため、アプライ・プログラムの 1 つのインスタンスですべてのセットを処理する場合よりもパフォーマンスが向上します。

control_server

z/OS **デフォルト**: なし

Linux UNIX Windows **デフォルト:** DB2DBDFT 環境変数の値 (使用できる場合)

アプライ・コントロール・サーバーは、サブスクリプション定義およびアプライ・コントロール表が置かれているサーバーです。1 つのアプライ修飾子に対して、コントロール・サーバーを 1 つだけ指定してください。値が指定されない場合、アプライ・プログラムはデフォルト・サーバー上で始動されます。デフォルトはオペレーティング・システムによって異なります。

アプライ・プログラムがコントロール・サーバーに接続できない場合、アプライ・プログラムは終了します。アプライ・プログラムは他のサーバーに接続できなくても終了しません。この場合には、エラー・メッセージを発行して処理を続けます。

copyonce

デフォルト: **copyonce=n**

copyonce パラメーターは、アプライ・プログラムのコピー・サイクルを決定します。

copyonce=y を指定して開始されたアプライ・プログラムは、適格なサブスクリプション・セットをそれぞれ一度だけ処理した後で終了します。この場合、以下の条件のいずれかが満たされた場合、サブスクリプション・セットは処理に適格となります。

- サブスクリプション・セットは相対タイミングを使用し、時間が経過し、サブスクリプション・セットはアクティブである。
- サブスクリプション・セットはイベント・ベースのタイミングを使用し、アクティブであり、イベントが発生したが、アプライ・プログラムはまだサブスクリプション・セットを処理していない。

copyonce=n を使用してアプライ・プログラムを始動する典型的な状況は、アプライ・プログラムの実行を継続し、適格なサブスクリプションの処理を続ける必要があるときです。

時々ネットワークに接続するようなダイヤルイン環境でアプライ・プログラムを実行するときには、**copyonce=n** ではなく、**copyonce=y** を使用してください。また、テスト環境でアプライ・プログラムを実行するときにも、**copyonce=y** を使用することが考えられます。

ヒント: サブスクリプション・セットが適格であり、レプリケーションできるデータがあるかぎり、アプライ・プログラムが各サブスクリプション・セットの処理を何度も行うようにする場合は、**copyonce=y** ではなく、**sleep=n** を使用してください。 **copyonce=y** は、複製するべきデータが残っている場合も、各セットを一度しか処理しません。

db2_subsystem (z/OS)

db2_subsystem パラメーターは、アプライが z/OS 上で実行している場合に、DB2 サブシステムの名前を指定します。入力する DB2 サブシステム名は最大 4 文字です。このパラメーターにはデフォルトはありません。このパラメーターは必須です。

delay

デフォルト: **delay=6** 秒

delay パラメーターは、アプライ・プログラムが各アプライ・サイクルの最後に待機する秒数を設定します。

デフォルトでは、連続してレプリケーションを行う場合 (サブスクリプション・セットで **sleep=0** 分を使用する場合)、アプライ・プログラムはサブスクリプション・セットが正常に処理されてから、 6 秒間待った後にサブスクリプション・セットを

再試行します。複製すべきデータベース・アクティビティーがない場合は、ゼロ以外の値を使用して CPU サイクルを節約してください。待ち時間を少なくするには遅延値を小さくします。

注: copyonce が指定されている場合、**delay** パラメーターは無視されます。

errwait

デフォルト: **errwait**=300 秒 (5 分)

errwait パラメーターは、サブスクリプション・サイクルが失敗した後に、アプライ・プログラムがサブスクリプション・セットを再試行するまでに待つ秒数を指定します。

デフォルトでは、アプライ・プログラムはサブスクリプション・サイクルが失敗した後 300 秒待機してから、サブスクリプション・セットを再試行します。テスト環境ではより小さい値を使用することが考えられます。最小値は 1 秒です。実稼働環境では、このパラメーターのデフォルトを変更する前に、トレードオフを考慮してください。

- 小さい値を使用すると、アプライ・プログラムがハード・エラーを繰り返し再試行することにより、CPU サイクルが無駄になります。例えば、ターゲット表に問題があるときにアプライ・プログラムがサブスクリプション・セットの処理を繰り返し再試行した場合は、 CPU サイクルを不必要に消費することになります。ログ・ファイルに、またはオペレーター・コンソールに (アプライ・プログラムを z/OS で実行している場合) に、大量のメッセージが送られることもあります。
- 大きな値を使用すると、アプライ・プログラムが一時的エラー条件を再試行するまで待つ必要がある場合の待ち時間が増えることになります。例えば、迅速に解決されるネットワーク・エラーを検出したときにもアプライ・プログラムは不必要に待つことになるため、**errwait** パラメーターに大きな値を使用すると、待ち時間が増加します。

注: copyonce が指定されている場合、**errwait** パラメーターは無視されます。

inamsg

デフォルト: **inamsg**=y

inamsg パラメーターは、アプライ・プログラムが非アクティブになったときにメッセージを発行するかどうかを指定します。

デフォルトでは、アプライ・プログラムは非アクティブになるとメッセージを発行します。サブスクリプション・セットの処理間のアプライ・プログラムの待ち時間が長くない場合は特にそうですが、メッセージはアプライ・ログ・ファイルのスペースを多量に消費します。このため、アプライ・プログラムが非アクティブになった場合のメッセージを発行させたくないことも考えられます。これらのメッセージをオフにするには、**inamsg**=n を使用します。

loadxit

デフォルト: **loadxit**=n

loadxit パラメーターは、アプライ・プログラムが ASNLOAD 出口ルーチンを使用してターゲット表をリフレッシュするかどうかを指定します。

デフォルトでは、アプライ・プログラムはターゲット表をリフレッシュするために ASNLOAD 出口ルーチンを使用しません (**loadxit**=n)。アプライ・プログラムで ASNLOAD 出口ルーチンを呼び出してターゲット表をリフレッシュする場合は、**loadxit**=y を使用します。フル・リフレッシュ時にターゲット表に大量のデータをコピーするときには、 ASNLOAD 出口を使用することを考えてみてください。

logreuse

デフォルト: **logreuse**=n

アプライ・プログラムは、操作情報をログ・ファイルに保管します。このパラメーターは、ログ・ファイルに追加するか、上書きするかを指定します。

z/OS

ログ・ファイルの名前は *control_server.apply_qualifier*.APP.log になります。

Linux UNIX Windows

ログ・ファイルの名前は *db2instance.control_server.apply_qualifier*.APP.logになります。

デフォルトでは、アプライ・プログラムは開始されるたびに、ログ・ファイルにメッセージを追加します (**logreuse**=n)。アプライ・プログラムから発行されたメッセージの履歴を保存する場合は、デフォルトのままにします。以下のような状況では、**logreuse**=y を使用して、アプライ・プログラムが開始時にログを削除し、再作成するようにできます。

- ログが大きくなったため、ログを消去してスペースを節約する場合。
- ログに保管されている履歴が不要な場合。

logstdout

デフォルト: **logstdout**=n

logstdout パラメーターは、asnapply コマンドを使用した場合にしか使用できません。**logstdout** はレプリケーション・センターからは使用できません。

logstdout パラメーターは、アプライ・プログラムが、完了メッセージ (ASN10251) をログ・ファイルと標準出力の両方に送信するかどうかを指定します。

デフォルトでは、アプライ・プログラムは完了メッセージを標準出力 (STDOUT) に送信しません。 **logstdout**=y と指定すると、アプライ・プログラムは、完了メッセージをログ・ファイルと標準出力 (STDOUT) の両方に送信します。トラブルシューティングの場合や、アプライ・プログラムの稼働状態をモニターする場合は、メッセージを標準出力に送信するように選択できます。

notify

デフォルト: **notify**=n

notify パラメーターは、アプライ・プログラムがサブスクリプションを処理した後に、ASNDONE 出口ルーチンに通知するかどうかを指定します。

デフォルトでは、アプライ・プログラムはサブスクリプションの処理が完了した後で ASNDONE 出口ルーチンに通知しません。**notify=y** が指定されると、アプライ・プログラムはサブスクリプション・サイクルが完了した後、アプライ・コントロール表の検査や、E メール・メッセージの送信など、追加処理を実行するために、ASNDONE を呼び出します。

opt4one

デフォルト: **opt4one=**n

opt4one パラメーターは、アプライ・プログラムの処理が、1 つのサブスクリプション・セット用に最適化されるかどうかを指定します。

注: copyonce が指定されている場合、**opt4one** パラメーターは無視されます。

デフォルトでは、アプライ・プログラムは複数のサブスクリプション・セット用に最適化されています。アプライ・プログラムは各コピー・サイクルの最初にレプリケーション・コントロール表から情報を読み取ります。アプライ修飾子に対してサブスクリプション・セットが 1 つである場合は、アプライ・プログラムがメモリー・キャッシュ内にサブスクリプション・セットのメンバーおよび列の情報を入れてそれを再利用するように、**opt4one=**y を使用してアプライ・プログラムを始動してください。アプライ・プログラムを 1 つのサブスクリプション・セット用に最適化すると、アプライ・プログラムから使用される CPU が少なくなるため、スループット率が向上します。

重要: **opt4one=**y を使用した場合に、セットにメンバーを追加するか、その他の方法でセットを変更したときには、アプライ・プログラムがコントロール表の中の変更を入手できるように、アプライ・プログラムを停止してから再度開始する必要があります。

pwdfile

デフォルト: **pwdfile=**asnpwd.aut

データを複数のサーバーに分散する場合は、アプライ・プログラムがリモート・サーバー上のデータにアクセスできるように、暗号化されたパスワード・ファイルにユーザー ID とパスワードを保管できます。

sleep

デフォルト: **sleep=**y

sleep パラメーターは、アプライ・プログラムが適格なサブスクリプション・セットを処理した後で、スリープ・モードで実行を続けるか、終了するかを指定します。

デフォルトでは、アプライ・プログラムは **sleep=**y で始動します。アプライ・プログラムは適格なサブスクリプション・セットがあるかどうかをチェックします。適格なサブスクリプション・セットが検出されると、アプライ・プログラムはセットを処理し、別の適格なセットがないか探します。アプライ・プログラムは適格なセ

ットが検出されると、その処理を続けます。適格なセットが検出されない場合、アプライ・プログラムはスリープ・モードで実行を続け、定期的に「ウェイクアップ」して、適格なサブスクリプション・セットがあるかどうかをチェックします。通常の場合は、長期間にわたって更新を適用するために、アプライ・プログラムをアクティブな状態で実行しておきたいため、この方法でアプライ・プログラムを始動します。

注: **copyonce** が指定されている場合、**sleep** パラメーターは無視されます。

sleep=n を指定して始動したアプライ・プログラムは、適格なサブスクリプション・セットがあるかどうかをチェックし、それを処理します。アプライ・プログラムは適格なサブスクリプション・セットが見つからなくなるまでセットの処理を続け、複製するべきデータがなくなるまで適格なセットの処理を繰り返した後、終了します。**sleep=**n を使用する典型的な状況は、適格なサブスクリプション・セットが検出されたときにのみアプライ・プログラムが実行され、その後にプログラムが終了される、モバイル環境またはテスト環境です。この場合は、アプライ・プログラムをスリープ・モードで待機させ、適格なセットがないかどうかをチェックするために定期的にウェイクアップさせることはしません。このような環境では、アプライ・プログラムを無期限に実行するのではなく、アプライ・プログラムをいつ実行するかをユーザーがコントロールできるようにします。

ヒント: 各サブスクリプション・セットを一度だけ処理する場合は、**sleep=**n ではなく、**copyonce=**y を使用してください。

spillfile

z/OS **デフォルト:** **spillfile=**MEM

Linux UNIX Windows **デフォルト:** **spillfile=**disk

アプライ・プログラムはソース表からデータをリトリーブし、アプライ・プログラムが実行されているシステム上の予備ファイルに入れます。

z/OS USS を含む z/OS オペレーティング・システムの場合は、予備ファイルはデフォルトでメモリー内に保管されます。ディスク上に予備ファイルを保管するように指定する場合、アプライ・プログラムは ASNASPL DD ステートメント上の指定を使用して、予備ファイルを割り振ります。ASNASPL DD ステートメントが指定されない場合、VIO が使用されます。

Linux UNIX Windows **spillfile** の唯一有効な設定は disk です。予備ファイルは、必ず **apply_path** パラメーターで指定されたロケーションのディスクに置かれるからです。

sqlerrcontinue

デフォルト: **sqlerrcontinue=**n

sqlerrcontinue パラメーターは、アプライ・プログラムが特定の SQL エラーに対処する方法を指定します。

デフォルトでは、アプライ・プログラムは SQL エラーを検出すると、そのサブスクリプション・セットの処理を停止し、エラー・メッセージを生成します。通常の場合、実稼働環境ではデフォルトを使用します。

テスト環境の場合は、ターゲット表にデータを挿入するときに、特定の SQL エラーが発生することが予想されます。これらのエラーはユーザーにとっては許容できる場合もありますが、エラーにより現行のサブスクリプション・サイクルは停止します。このような状況では、アプライ・プログラムがエラーを無視し、そのサイクルで複製したデータをロールバックしないように、**sqlerrcontinue=y** を使用してアプライ・プログラムを始動できます。アプライ・プログラムはターゲット表へのデータ挿入時に SQL エラーを受け取ると、*apply_qualifier*.sqs ファイルの中の値をチェックします。一致するものがあると、エラーに関する詳細をエラー・ファイル *apply_qualifier*.errに書き込み、処理を続行します。アプライ・プログラムは、<*apply_qualifier*.sqs ファイルにリストされていない SQL エラーを検出すると、セットの処理を停止し、次のセットに進みます。

sqlerrcontinue=y オプションを使用してアプライ・プログラムを始動する前に、*apply_qualifier*.sqs ファイルを作成し、アプライ・プログラムの呼び出しに使用したディレクトリーにこのファイルを保管する必要があります。このファイルには、最大 20 の 5 バイトの値を続けてリストしてください。アプライ・プログラムの実行中にこのファイルの内容を変更したときには、アプライ・プログラムが新しい値を認識できるように、アプライ・プログラムを停止してから再度開始してください。

例: ターゲット表で次のエラーを受け取ったときにアプライ・プログラムがサブスクリプション・セットの処理を続けるようにするとします (sqlstate/code)。

42704/-803
重複索引違反

以下の SQL 状態を含む SQL 状態ファイルを作成します。

```
42704
```

ターゲット表の更新時にこの SQL 状態が戻された場合、アプライ・プログラムはセット内のその他のターゲット表に変更を適用し、エラーおよびリジェクトされた行の両方を示すエラー・ファイルを作成します。

ヒント: IBMSNAP_APPLYTRAIL 表の STATUS 列をチェックします。16 という値は、アプライ・プログラムがサブスクリプション・セットを正常に処理したが、*apply_qualifier*.sqs ファイルで定義された、許容されるなんらかのエラーが発生したことを意味します。

term

デフォルト: **term**=y

term パラメーターは、DB2 の状況が、アプライ・プログラムの稼働にどのような影響を与えるかを決定します。

デフォルトでは、アプライ・プログラムは DB2 が終了または静止すると終了します。

DB2 がアクティブでないときに、アプライ・プログラムが DB2 の始動を待つようにするには、**term**=n を使用します。アプライ・プログラムはアクティブなまま残り、DB2 が開始または静止解除されるまで再接続します。

注: copyonce が指定されている場合、**term** パラメーターは無視されます。

trlreuse

デフォルト: **trlreuse**=n

trlreuse パラメーターは、アプライ・プログラムの始動時に、IBMSNAP_APPLYTRAIL 表を再利用する (追加する) か、上書きするかを指定します。

デフォルトでは、アプライ・プログラムは開始されると、アプライ・トレール表に項目を追加します。この表は、アプライ・コントロール・サーバーにあるすべてのアプライ・インスタンスの操作の履歴を保持します。この表は、診断およびパフォーマンス統計のリポジトリーです。更新の履歴を保持する場合は、デフォルトを使用します。以下のような状況では、アプライ・プログラムの始動時に、アプライ・トレール表に追加するのではなく、表を空にすることができます (**trlreuse**=y)。

- アプライ・トレール表が大きくなったため、消去してスペースを節約する場合。
- 表に保管されている履歴が不必要な場合。

ヒント: trlreuse=y を使用する代わりに、アプライ・プログラムがサブスクリプション・セットを正常に処理した後で (**status**=0)、SQL 処理を使用してアプライ・トレール表から行を削除できます。

アプライの操作パラメーターの変更方法

稼働パラメーターのデフォルト値を、ご使用の環境で通常使用している値に変更できます。アプライ・プログラムの開始時に、これらのデフォルト値をオーバーライドすることもできます。

IBMSNAP_APPPARMS 表での新しいデフォルト値の設定

IBMSNAP_APPPARMS 表には、アプライ・プログラムの動作を制御するために変更できるパラメーターが含まれています。この表が作成されると、その中にはアプライ・プログラム用のデフォルト値が入れられます。

アプライ・プログラムの始動時に行うパラメーター値の指定

アプライ・プログラムの始動時に、このプログラム用の値を指定できます。始動時に設定された値により、現行セッションの間のアプライの動作がコントロールされ、デフォルトの稼働パラメーター値およびアプライ・パラメーター表内に存在する可能性のあるあらゆる値がオーバーライドされます。アプライ・パラメーター表内の値については、これらの値による更新は行われません。アプライ・プログラムを始動する前にアプライ・パラメーター表を変更せず、さらにアプライ・プログラムの始動時にどの稼働パラメーターも指定しなかった場合は、稼働パラメーターに対してデフォルト値が使用されます。

例 Linux UNIX Windows

アプライ修飾子 ASNPROD の **errwait** のデフォルト設定を使用しない場合。ASNPROD アプライ修飾子用のアプライ・パラメーター表を更新します。 **errwait** インターバルを 600 秒に設定します。

```
update asn.ibmsnap_appparms set errwait=600 where apply_qual='ASNPROD'
```

IBMSNAP_APPPARMS 表に保存されているアプライ・パラメーターの変更 (z/OS、Linux、UNIX、Windows)

IBMSNAP_APPPARMS 表には、アプライ・プログラムの操作パラメーターが保管されています。アプライ・プログラムを始動すると、始動パラメーターを使用してその表の値を一時的にオーバーライドした場合を除き、アプライ・プログラムはその表の値を使用します。

このタスクについて

各アプライ修飾子には 1 つの行しか入れられません。デフォルト値のいずれかを変更する場合は、行を挿入する代わりに列を更新することができます。この行を削除しても、出荷時のデフォルトが始動パラメーターによってオーバーライドされないかぎり、アプライ・プログラムはそれらのデフォルトを使用しても始動します。

アプライ・プログラムはこの表を始動時にしか読み取らないため、新しい設定でアプライ・プログラムを実行する場合は、アプライ・プログラムの停止と始動を行う必要があります。アプライ・プログラムの実行中にアプライ・パラメーター表を変更しても、アプライ・プログラムの動作は変更されません。

アプライ・プログラムの停止

ユーザーがアプライ・プログラムを停止すると、アプライ・プログラムはターゲット表にデータをコピーしなくなり、次にプログラムが開始されたときに正しく開始されるように、コントロール表を更新します。

手順

アプライ・プログラムを停止するには、次のようにします。

以下のいずれかの方法を使用します。

オプション	説明
レプリケーション・センター	「アプライの停止」ウィンドウを使用します。このウィンドウを開くには、オブジェクト・ツリーの 「**操作**」ブランチから「**アプライ・コントロール・サーバー**」フォルダーを開き、「**アプライ修飾子**」フォルダーをクリックします。内容ペインで、始動する対象のアプライ・プログラムを表すアプライ修飾子を右クリックして、「**アプライの停止**」をクリックします。

オプション	説明
z/OS Linux UNIX Windows asnacmd stop システム・コマンド	このコマンドを使用して、アプライを停止します。
System i ENDDPRAPY システム・コマンド	DPR アプライ・プログラムの終了 (ENDDPRAPY) コマンドは、ローカル・システム上のアプライ・プログラムを停止するために使用されます。

ASNDONE 出口ルーチンの変更 (z/OS、Linux、UNIX、Windows)

Linux 、UNIX、Windows、z/OS の各オペレーティング・システムでは、ASNDONE 出口ルーチンをカスタマイズして、サブスクリプションの処理の完了後にアプライ・プログラムの動作を変更できます。

このタスクについて

notify=y パラメーターを使用してアプライ・プログラムを始動すると、アプライ・プログラムはサブスクリプションの処理が成功したかどうかに関係なく、サブスクリプションの処理が終了した後、ASNDONE 出口ルーチンを呼び出します。以下のリストは、レプリケーション環境で使用するために ASNDONE 出口ルーチンをどのように変更できるかの例を示しています。

- トランザクションがリジェクトされたことが判明した場合は、この出口ルーチンを使用して、リジェクトされたトランザクションを UOW 表で確認し、その後のアクション (例えば、 E メールをレプリケーション・オペレーターに自動的に送信する、メッセージを発行する、またはアラートを生成する) を開始します。
- この出口ルーチンを使用して、失敗したサブスクリプション・セットが訂正されるまでアプライ・プログラムが再試行を続けないように、失敗したサブスクリプション・セットを非活動化します。失敗したサブスクリプション・セットを検出するには、IBMSNAP_APPLYTRAIL 表で STATUS= -1 を探すように出口ルーチンを変更します。サブスクリプション・セットを非活動化するには、IBMSNAP_SUBS_SET 表に ACTIVATE=0 を設定するように出口ルーチンを構成します。
- データが各サブスクリプション・セットに適用された後で出口ルーチンを使用してデータを操作します。(この代わりに、 SQL ステートメントまたはストアード・プロシージャーを使用して、アプライ・プログラムによる特定のサブスクリプション・セットの処理の前または後で実行されるランタイム処理ステートメントを定義することもできます)

手順

サンプルの ASNDONE 出口ルーチンの変更したバージョンを使用するには、以下のようにします。

1. 要件に合うように、ASNDONE ルーチンを調整します。

- z/OS サンプル・プログラム SASNSAMP(ASNDONE) の PROLOG セクションを参照してください。
- Linux UNIX Windows この出口ルーチンの変更方法については、サンプル・プログラム (¥sqllib¥samples¥repl¥asndone.smp) の PROLOG セクションを参照してください。

2. プログラムをコンパイル、リンク、バインドし、実行可能ファイルを適切なディレクトリーに入れます。
3. **notify**=y パラメーターを使用してアプライ・プログラムを始動し、 ASNDONE 出口ルーチンを呼び出します。

ASNDONE 出口ルーチンの変更 (System i)

System i オペレーティング・システムでは、ASNDONE 出口ルーチンをカスタマイズして、サブスクリプションの処理の完了後にアプライ・プログラムの動作を変更できます。

このタスクについて

SUBNFYPGM パラメーターを ASNDONE 出口ルーチンの名前に設定してアプライ・プログラムを始動すると、アプライ・プログラムは、サブスクリプションの処理が成功したかどうかに関係なく、サブスクリプションの処理が終了した後、ASNDONE 出口ルーチンを呼び出します。以下のリストは、レプリケーション環境で使用するために ASNDONE 出口ルーチンをどのように変更できるかの例を示しています。

- トランザクションがリジェクトされたことが判明した場合は、この出口ルーチンを使用して、リジェクトされたトランザクションを UOW 表で確認し、その後のアクション (例えば、 E メールをレプリケーション・オペレーターに自動的に送信する、メッセージを発行する、またはアラートを生成する) を開始します。
- この出口ルーチンを使用して、失敗したサブスクリプション・セットが訂正されるまでアプライ・プログラムが再試行を続けないように、失敗したサブスクリプション・セットを非活動化します。失敗したサブスクリプション・セットを検出するには、IBMSNAP_APPLYTRAIL 表で STATUS= -1 を探すように出口ルーチンを変更します。サブスクリプション・セットを非活動化するには、IBMSNAP_SUBS_SET 表に ACTIVATE=0 を設定するように出口ルーチンを構成します。
- データが各サブスクリプション・セットに適用された後で出口ルーチンを使用してデータを操作します。(SQL ステートメントまたはストアード・プロシージャーを使用して、アプライ・プログラムによる特定のサブスクリプション・セットの処理の前または後で実行されるランタイム処理ステートメントを定義することもできます。)

手順

サンプルの ASNDONE 出口ルーチンの変更したバージョンを使用するには、以下のようにします。

1. 要件に合うように、ASNDONE 出口ルーチンを調整します。 150 ページの表 9 は、C、COBOL、および RPG 言語のこのルーチンのソース・コードがどこにあ

るかを示しています。

表 9. ASNDONE のソース・コード

コンパイラー言語	ライブラリー名	ソース・ファイル名	メンバー名
C	QDP4	QCSRC	ASNDONE
COBOL	QDP4	QCBLLESRC	ASNDONE
RPG	QDP4	QRPGLESRC	ASNDONE

プログラムの変更時に、以下の活動化グループに関連することを考慮してください。

このプログラムが新しい活動化グループで実行するために作成される場合
アプライ・プログラムおよび ASNLOAD プログラムは、リレーショナル・データベースの接続およびオープン・カーソルなどの SQL リソースを共有しません。System i オペレーティング・システムにある活動化処理コードは、コントロールがアプライ・プログラムに戻される前に、ASNLOAD プログラムによって割り振られたリソースをすべて解放します。アプライ・プログラムが ASNLOAD プログラムを呼び出すたびに、追加のリソースが使用されます。

このプログラムが呼び出し側の活動化グループで実行するために作成される場合
アプライ・プログラムとの間で SQL リソースを共有します。アプライ・プログラムへの影響が最小限で済むようプログラムを設計します。例えば、現行のリレーショナル・データベース接続を変更すると、予期しないアプライ・プログラムの処理を引き起こす可能性があります。

このプログラムが名前付き活動化グループで実行するために作成される場合
アプライ・プログラムとの間でリソースを共有しません。名前付き活動化グループを使用すると、ASNLOAD プログラムが呼び出されるたびに、活動化グループのオーバーヘッドは回避されます。ランタイムのデータ構造および SQL リソースは、呼び出し間で共有されます。アプライ・プログラムが終了するまで、アプリケーションの終結処理は実行されません。したがって、サブスクリプション通知プログラムは、コントロールがアプライ・プログラムに戻されるとき、ソース表、ターゲット表、またはコントロール表をロックすることにより、アプライ・プログラムでロックの競合が生じることがないように設計します。

2. プログラムをコンパイル、リンク、バインドし、実行可能ファイルを適切なディレクトリーに入れます。
3. アプライ・プログラムを始動し、STRDPRAPY コマンドのパラメーター SUBNFYPGM を使用して、ASNDONE プログラムの名前を指定します。

例えば、プログラムの名前が ASNDONE_1 でライブラリー APPLIB にある場合には、以下のコマンドを使用します。

```
SUBNFYPGM(APPLIB/ASNDONE_1)
```

ASNLOAD 出口ルーチンを使ったターゲット表のリフレッシュ

ASNLOAD 出口ルーチンを使用すれば、アプライ・プログラムがデータをターゲットにロードする通常の方法よりも、ターゲット表のフル・リフレッシュを効率的に実行できます。

デフォルトでは、アプライ・プログラムはサブスクリプション・セット内の各ターゲット表のリフレッシュを実行するときに ASNLOAD 出口ルーチンを使用しません。アプライ・プログラムはソース表に対して全選択を行い、アプライ・プログラムが実行されているサーバー上の予備ファイルにデータを持ち込み、INSERT ステートメントを使用してターゲット表にデータを取り込みます。ソース表が大きい場合には、代わりに ASNLOAD 出口ルーチンを使用できます。

以下のように、サンプルの出口ルーチンは DB2 プラットフォームによってそれぞれ異なり、そのプラットフォームで提供されるユーティリティー・オプションを利用します。

z/OS Linux UNIX Windows

ASNLOAD 出口ルーチンは、サンプルの出口ルーチンとして用意されており、ソース形式とコンパイル済み形式の両方があります。

System i

ASNLOAD はソース形式だけで用意されています。

アプライ・プログラムが ASNLOAD 出口ルーチンを呼び出したときにエラーが発生すると、アプライ・プログラムがメッセージを発行し、現行のサブスクリプション・セットの処理を停止し、次のサブスクリプション・セットを処理します。

ASNLOAD 出口ルーチンを使ったターゲット表のリフレッシュ (Linux、 UNIX、Windows)

Linux、UNIX、Windows の各オペレーティング・システムでは、ASNLOAD 出口ルーチンを使用して、ターゲット表を効率的にリフレッシュできます。このルーチンは、変更してから使用することも可能です。

始める前に

- ターゲット表の列は、ソース表の順序およびデータ・タイプの両方と一致しなければなりません。
- ターゲット表は、レプリケーション・マッピングの一部である列だけを含んでいなければなりません。
- アプライを実行するユーザー ID は、ASNLOAD を実行する DB2 インスタンスのユーザー ID でなければなりません。例えば、Linux と UNIX の場合は、DB2 インスタンスとアプライの両方のユーザー ID が共通のグループのメンバーであることを確認してください。次に、DB2 インスタンスに書き込みアクセス権限を提供するために、chmod 775 コマンドを使用して、アプライ始動ディレクトリーの許可ビットを設定します。

制約事項

ASNLOAD 出口ルーチンは、EXPORT、IMPORT、LOAD の各ユーティリティー、および LOAD FROM CURSOR 機能と連動します。サブスクリプション・セット・メンバーのソースがニックネームの場合や、ターゲット・データベースがソース・データベースと同じ場合は、LOAD FROM CURSOR が ASNLOAD 出口のデフォルト・オプションになります。以下の操作を実行した場合は、LOAD FROM CURSOR を DB2 データ・ソースに対して使用することもできます。

- ソース表のニックネームがターゲット・データベースで作成された場合。

- サブスクリプション・セット・メンバーの IBMSNAP_SUBS_MEMBR 表の列が、LOAD FROM CURSOR 機能が使用されることを示すように設定された場合。これらの列の値は、レプリケーション・センターを使って設定できます。
  - LOADX_TYPE 列は、LOAD FROM CURSOR 機能を使用することを示すように設定する必要があります。
  - LOADX_SRC_N_OWNER 列と LOADX_SRC_N_TABLE 列では、ソース表を含むサブスクリプション・セット・メンバーのソース・ニックネーム情報を指定する必要があります。

このタスクについて

ユーザーがサンプルの出口ルーチンを呼び出すと、デフォルトでは、ソース・サーバー、ターゲット・サーバー、およびランタイム環境に基づいて、使用されるユーティリティーが選択されます。このルーチンは、DB2 IMPORT ユーティリティーまたは DB2 LOAD ユーティリティーと一緒に DB2 EXPORT ユーティリティーを使用するか、LOAD FROM CURSOR ユーティリティーを使用できます。

コンパイル済みの出口ルーチンを使用したり、レプリケーション構成をカスタマイズして動作を構成したり、出口コード自体を変更することができます。レプリケーション構成は、IBMSNAP_SUBS_MEMBR 表の列を更新するか、サンプルの構成ファイル (asnload.ini) を更新することによりカスタマイズできます。

提供されたままの ASNLOAD ルーチンを使用する場合は、**loadxit=y** パラメーターを使用してアプライ・プログラムを始動します。

手順

ASNLOAD 出口ルーチンの変更したバージョンを使用するには、以下のようにします。

1. サイトの要件に合うように、ASNLOAD ルーチンを調整します。 この出口ルーチンの変更方法については、サンプル・プログラム (sqllib¥samples¥repl¥asnload.smp) の PROLOG セクションを参照してください。

   重要: サンプルのソースは、asnload.ini ファイルからユーザー ID とパスワードの組み合わせを使用します。asnload.ini ファイルに特定のサーバーのユーザー ID とパスワードがない場合や、asnload.ini ファイルを使用できない場合、この出口は、 **user**パラメーターまたは **using**パラメーターを使用しないで接続しようとします。
2. プログラムをコンパイル、リンク、バインドし、実行可能ファイルを適切なディレクトリーに入れます。
3. ユーザーが指定したコードを使用して取り込みが行われるメンバーについては、LOADX_TYPE を 2 に設定してください。
4. loadxit=y パラメーターを使用してアプライ・プログラムを始動し、 ASNLOAD 出口ルーチンを呼び出します。

ASNLOAD 出口ルーチンは、ASNLOAD 出口ルーチンを呼び出したアプライ・インスタンスの **apply_path** ディレクトリーに以下のファイルを生成します。

asnload *apply_qualifier***.trc**
トレースがオンの場合、このファイルはトレース情報を保持します。

ASNLOAD 出口ルーチンがこのファイルを作成します。ファイルが存在する場合は、情報がファイルに追加されます。

asnload *apply_qualifier*.**msg**
このファイルは、ロード統計を含めて、一般出口障害、警告メッセージおよび情報メッセージを保持します。ASNLOAD 出口ルーチンがこのファイルを作成します。ファイルが存在する場合は、情報がファイルに追加されます。

asnaEXPT *apply_qualifier*.**msg**
このファイルは、DB2 の EXPORT ユーティリティーが発行したエラー・メッセージ、警告メッセージ、または通知メッセージを保持します。ASNLOAD 出口ルーチンがこのファイルを作成します。ファイルが存在する場合は、情報がファイルに追加されます。

asnaIMPT *apply_qualifier*.**msg**
このファイルは、DB2 の IMPORT ユーティリティーが発行したエラー・メッセージ、警告メッセージ、または通知メッセージを保持します。ASNLOAD 出口ルーチンがこのファイルを作成します。ファイルが存在する場合は、情報がファイルに追加されます。

asnaLOAD *apply_qualifier*.**msg**
このファイルは、DB2 の LOAD ユーティリティーが発行したエラー・メッセージ、警告メッセージ、または通知メッセージを保持します。ASNLOAD 出口ルーチンがこのファイルを作成します。ファイルが存在する場合は、情報がファイルに追加されます。

ASNLOAD 出口ルーチンを使ったターゲット表のリフレッシュ (z/OS)

z/OS オペレーティング・システムでは、ASNLOAD 出口ルーチンを使用することにより、ターゲット表をより効率的にリフレッシュできます。このルーチンは、変更してから使用することも可能です。

始める前に

- ターゲット表の列は、ソース表の順序およびデータ・タイプの両方と一致しなければなりません。
- ターゲット表は、レプリケーション・マッピングの一部である列だけを含んでいなければなりません。

このタスクについて

ASNLOAD 出口ルーチンは、DB2 V7 (またはそれ以降) Utilities Suite で使用可能な LOAD FROM CURSOR ユーティリティーを呼び出します。このユーティリティーは、カーソルに基づくフェッチを行い、ソースからデータを入手し、ターゲットにデータをロードします。

ASNLOAD 出口ルーチンは LOG NO を指定して LOAD を使用し、表スペースの COPYPEND 状況をリセットします。サンプルの ASNLOAD ソース・コードに変更を加えて、ロード・オプションを変更できます。ソースは、2 つのヘッダー・ファイルと 3 つの C++ プログラムから構成されています。

提供されたままの ASNLOAD ルーチンを使用する場合は、**loadxit**=y パラメーターを使用してアプライ・プログラムを始動します。

手順

ASNLOAD 出口ルーチンの変更したバージョンを使用するには、以下のようにします。

1. サイトの要件に合うように、ルーチンを調整します。この出口ルーチンの変更方法については、サンプル・プログラム SASNSAMP(ASNLOAD) の PROLOG セクションを参照してください。
2. プログラムをコンパイル、リンク、バインドし、実行可能ファイルを適切なディレクトリーに入れます。
   a. 以下の条件が満たされていることを確認してください。
      - DB2 Universal Database™ for z/OS and OS/390 バージョン 7 以降 (ユーティリティー・サポートを含む) がインストールされている。
      - DSNUTILS ストアード・プロシージャーが実行されている。DSNUTILS は WLM 環境で実行する必要があります。DSNUTILS の使用方法の詳細は、「*DB2 for z/OS V8 Utility Guide and Reference*」を参照してください。
   b. サンプル zmak ファイル (SASNSAMP(ASNCMPLD)) を使用して、USS の ASNLOAD ユーザー出口プログラムをコンパイルし、リンク・エディットします。
   c. ASNLOAD 出口ルーチンを DSNUTILS およびアプライ・パッケージとバインドします。 サンプルの ASNLOAD は LOG NO を指定してロードを実行した後、表スペースを修復して nocopypend を設定します。表スペースのバックアップは行いません。デフォルトでは、 ASNLOAD は、APPLY_PATH=// オプションが指定された **apply_path** パラメーターがそのアプライ・インスタンスに指定されていない限り、アプライ・プログラムのインスタンスを実行中のユーザー ID の下に 2 つの一時ファイルを作成します。この場合、2 つの一時ファイルは、APPLY_PATH で指定された上位修飾子の下に作成されます。このルーチンは、ロードに関するすべての情報を含むファイルも作成します。
3. ユーザーが指定したコードを使用して取り込みが行われるメンバーについては、loadx_type = 2 を設定します。
4. **loadxit**=y パラメーターを使用してアプライ・プログラムを始動し、ASNLOAD 出口ルーチンを呼び出します。

ASNLOAD 出口ルーチンは、ASNLOAD 出口ルーチンを呼び出したアプライ・インスタンスの **apply_path** ディレクトリーまたは HLQ に、以下のファイルを生成します。

userid.*apply_qual*.**LOADMSG**
: このファイルは、ロード統計を含めて、障害、警告メッセージおよび情報メッセージを保持します。ASNLOAD 出口ルーチンがこのファイルを作成します。ファイルが存在する場合は、情報がファイルに追加されます。

userid.*apply_qual*.**LOADTRC**
: トレースがオンの場合、このファイルはトレース情報を保持します。

ASNLOAD 出口ルーチンがこのファイルを作成します。ファイルが存在する場合は、情報がファイルに追加されます。

ASNLOAD 出口の動作のカスタマイズ (z/OS、Linux、UNIX、Windows)

出口コードそのものをカスタマイズするだけでなく、ASNLOAD 出口ルーチンの動作をカスタマイズすることも可能です。そのためには、IBMSNAP_SUBS_MEMBR 表の列を更新するか、構成ファイルを更新します。

IBMSNAP_SUBS_MEMBR 表の使用による ASNLOAD オプションの設定

IBMSNAP_SUBS_MEMBR 表の列を使用して、 ASNLOAD 出口ルーチンの動作をカスタマイズできます。

このタスクについて

LOADX_TYPE 列を使用してロード・オプションを指定します。LOADX_TYPE の有効な値は以下のとおりです。

NULL (デフォルト)

z/OS LOAD FROM CURSOR ユーティリティーを使用します。

Linux UNIX Windows ASNLOAD 出口ルーチンが、最適なユーティリティーを決定します。(オプション 3、4 または 5)

1 このメンバーについては ASNLOAD 出口ルーチンを呼び出しません。

このメンバーについて ASNLOAD 出口ルーチンを呼び出したくない場合は、 LOADX_TYPE を 1 に設定します。

2 ユーザーの作成した出口ロジックを提供します。

ユーザーの作成したロジックを ASNLOAD 出口ルーチンで提供する場合は、 ASNLOAD 出口ルーチンにより取り込みを行うサブスクリプション・セットのメンバーに関して LOADX_TYPE を 2 に設定してください。LOADX_TYPE を 2 に設定しているのに出口ロジックを提供しない場合、出口は失敗します。

3 LOAD FROM CURSOR ユーティリティーを使用します。

Linux UNIX Windows LOAD FROM CURSOR 機能を使用するには、ターゲット表にロードするデータを取り出すための SELECT ステートメントが必要です (ターゲット表は、ローカル・データベースに存在している必要があります)。このステートメントは、DB2 表またはニックネームのどちらかを参照でき、セットアップは以下のようにします。

IBM 以外のソースから、登録済みソース・ニックネームがターゲット・データベースとは別のデータベース上にある DB2 表に複製している場合、または DB2 表から別の DB2 表へ複製していてソース・データベースがターゲット・データベースと異なる場合、以下のステップを実行する必要があります。

1. ターゲット・サーバー・データベースのソース表にニックネームを作成します。
2. IBMSNAP_SUBS_MEMBR 表のニックネーム所有者およびテーブル名列 (LOADX_SRC_N_OWNER および LOADX_SRC_N_TABLE) を更新します。

DB2 表から別の DB2 表へ複製していて、ソース・データベースとターゲット・データベースが同じである場合、または IBM 以外のソースから、登録済みソース・ニックネームがターゲット・データベースと同じデータベース上にある DB2 表に複製している場合、 LOAD FROM CURSOR ユーティリティーを使用するための追加のアクションは必要ありません。

Linux UNIX Windows **4**

EXPORT ユーティリティーと LOAD ユーティリティーの組み合わせを使用します。

Linux UNIX Windows **5**

EXPORT ユーティリティーと IMPORT ユーティリティーの組み合わせを使用します。

ASNLOAD の構成ファイルの使用 (Linux、UNIX、Windows)

ASNLOAD 出口ルーチンへの入力を構成するために、オプションの構成ファイルを使用できます。このファイルは ASNLOAD の実行に必須ではありません。

このタスクについて

構成ファイルのファイル名は asnload.ini にする必要があります。 ASNLOAD 出口ルーチンは、**apply_path** パラメーターで指定されているディレクトリーでこのオプションの構成ファイルを探します。

手順

ASNLOAD 構成ファイルを使用するには、以下のようにします。

1. サンプル・ファイル sqllib/samples/repl/asnload.ini を編集します。
2. ASNLOAD 出口ルーチンを呼び出したアプライ・インスタンスの **apply_path** パラメーターで指定されているディレクトリーにそのファイルを格納します。

ASNLOAD 出口ルーチンを使ったターゲット表のリフレッシュ (System i)

System i では、ASNLOAD 出口ルーチンを使用して、ターゲット表を効率的にリフレッシュできます。また、使用前にルーチンを変更することもできます。

始める前に

- ターゲット表の列は、ソース表の順序およびデータ・タイプの両方と一致しなければなりません。
- ターゲット表には、レプリケーション・マッピングの一部である列だけを組み込めます。

このタスクについて

例えば、ソース表の各列および各行をターゲット表にコピーしている場合、分散データ管理 (DDM) ファイルおよびファイルのコピー (CPYF) CL コマンドを使用する出口ルーチンのフル・リフレッシュを、ソース表からターゲット表へとファイル全体をコピーするように設計することができます。

提供されたままの ASNLOAD 出口ルーチンを使用する場合は、FULLREFPGM パラメーターを使用してアプライ・プログラムを始動します。

手順

ASNLOAD 出口ルーチンの変更したバージョンを使用するには、以下のようにします。

1. サイトの要件に合うように、ASNLOAD 出口ルーチンを調整します。 この出口ルーチンの変更方法については、サンプル・プログラムの PROLOG セクションを参照してください。ソースは、表 10にあるとおり、C、COBOL、RPG の各言語で用意されています。

表 10. ASNLOAD のソース・コード

コンパイラー言語	ライブラリー名	ソース・ファイル名	メンバー名
C	QDP4	QCSRC	ASNLOAD
COBOL	QDP4	QCBLLESRC	ASNLOAD
RPG	QDP4	QRPGLESRC	ASNLOAD

2. プログラムをコンパイル、リンク、バインドし、実行可能ファイルを適切なディレクトリーに入れます。 アプライ・プログラムを妨害しないようにするため、出口ルーチンが (呼び出し側の活動化グループではなく) 新しい活動化グループを使うようにコンパイルします。

   名前付き活動化グループまたは新しい活動化グループを使って出口ルーチンをコンパイルすることができます。パフォーマンスを向上させるには、名前付き活動化グループを使用します。名前付き活動化グループを使用すると、出口ルーチンは必要に応じて変更をコミットまたはロールバックする必要があります。アプライ・プログラムは、(終了するまで) 変更をコミットまたはロールバックすることはありません。出口ルーチンは、完了時に変更を明示的にコミットするか、または変更を暗黙的にコミットするためにコンパイルされる必要があります。出口ルーチンの完了時にコミットされていない変更はすべて、以下のいずれかが行われるまでコミットされることはありません。

   - アプライ・プログラムが、同じ活動化グループを使って別の出口ルーチンを呼び出す。
   - アプライ・プログラムに応じて開始したジョブが終了する。

3. FULLREFPGM パラメーターを ASNLOAD プログラムの名前に設定してアプライ・プログラムを始動します。 始動されたアプライ・プログラムは、ユーザーから指定された ASNLOAD 出口ルーチンを使用します。別の ASNLOAD 出口ルーチンを使用させたい場合は、アプライ・プログラムを終了してから再始動します。

ASNLOAD 出口ルーチンを実行すると、すべてのターゲット表が 1 つずつリフレッシュされます。

第 11 章 レプリケーション・プログラムの操作 (z/OS)

以下のトピックでは、z/OS オペレーティング・システム上でのレプリケーション・プログラムの操作について説明しています。

システム開始タスクを使用してレプリケーション・プログラムを操作する方法

システム開始タスクを使用して、キャプチャー・プログラム、アプライ・プログラム、およびレプリケーション・アラート・モニターを操作できます。

手順

システム開始タスクを使用してレプリケーション・プログラムを操作するには、キャプチャー・プログラムのこの例を使用してください。

1. PROCLIB にプロシージャー *procname* を作成します。
2. *procname* について、RACF® STARTED クラス内に項目を作成します。この項目は、*procname* を、キャプチャー・プログラムを始動するために使用される RACF ユーザー ID と関連付けます。キャプチャー・プログラムを始動する前に、このユーザー ID に必要な DB2 許可が付与されるようにしてください。
3. MVS™ システム・コンソールから、コマンド start *procname* を実行します。

以下のサンプル・プロシージャーはキャプチャー・プログラム用です。

```
//CAPJAYC PROC
//ASNCAP EXEC PGM=ASNCAP,REGION=M,
//PARM='V71A autostop LOGSTDOUT startmode=COLD
//capture_schema=JAY logreuse'
//STEPLIB DD DISP=SHR,DSN=DPROPR.ASN81 .SASNLOAD
//DD DISP=SHR,DSN=SYS1.SCEERUN
//DD DISP=SHR,DSN=DSN7.SDSNLOAD
//CEEDUMP DD SYSOUT=
//SYSPRINT DD SYSOUT=
//SYSTERM DD DUMMY
//
```

JCL を使用したレプリケーション・プログラムの操作

z/OS では、JCL を使用して、レプリケーション・プログラムの実行の開始、停止、および変更を行えます。操作を繰り返し実行する場合は、スクリプトを保管することができます。

このタスクについて

SQL レプリケーション V9 サンプル・ライブラリーには、サンプル JCL およびスクリプトが入っています。

推奨: 変更を加える前に、SASNSAMP ライブラリーから別のライブラリーにジョブをコピーしてください。SASNSAMP ライブラリーにあるサンプル・ジョブの完全なリストに関しては、プログラム・ディレクトリーを参照してください。

手順

JCL を使用してレプリケーション・プログラムを操作する方法

1. レプリケーション・プログラムを始動します。

オプション	説明
バッチ・ジョブを使用してキャプチャー・プログラムを始動する	ASNSTRC バッチ・ジョブの PARM フィールドに、オプションの適切な呼び出しパラメーターを指定することにより、z/OS 用の JCL を準備します。 TSO または z/OS コンソールからジョブを実行します。ジョブは SASNSAMP サンプル・ライブラリーにあります。
バッチ・ジョブを使用してアプライ・プログラムを始動する	ASNSTRA バッチ・ジョブの PARM フィールドに、オプションの適切な呼び出しパラメーターを指定することにより、z/OS 用の JCL を準備します。 TSO または z/OS コンソールからジョブを実行します。ジョブは SASNSAMP サンプル・ライブラリーにあります。
バッチ・ジョブを使用してレプリケーション・アラート・モニターを始動する	ASNSTRM バッチ・ジョブの PARM フィールドに、オプションの適切な呼び出しパラメーターを指定することにより、z/OS 用の JCL を準備します。 TSO または z/OS コンソールからジョブを実行します。ジョブは SASNSAMP サンプル・ライブラリーにあります。
JCL を使用してレプリケーション・アラート・モニターを始動する	レプリケーション・アラート・モニター・ジョブの PARM フィールドに、適切な呼び出しパラメーターを指定することにより、z/OS 用の JCL を準備します。サイトの要件に合うように、JCL を調整します。ライブラリー SASNSAMP(ASNMON#) にある呼び出し JCL のサンプルは、レプリケーション・アラート・モニター (z/OS 版) に含まれています。 呼び出し JCL でのこの行の例を次に示します。 `//monasn EXEC PGM=ASNMON,PARM='monitor_server=DSN monitor_qual=monqual'` この `DSN` はサブシステム名で、`monqual` はモニター修飾子です。

2. オプション: 開始済みのレプリケーション・プログラムを変更します。

   キャプチャー・プログラム、アプライ・プログラム、またはレプリケーション・アラート・モニター・プログラムを開始した後で、MODIFY コマンドを使用して、プログラムを停止したり、関連するタスクを実行することができます。MODIFY コマンドは MVS コンソールから実行する必要があります。以下の構文例に示すように、F という省略形を使用できます。

   基本的に、F *jobname*, は、asnacmd、 asnccmd、または asnmcmd など、実際のコマンド名に置き換わります。例えば、キャプチャー・プログラムを停止するには、以下のコマンドを使用します。

```
F capjfa,stop
```

MODIFY の情報については、「*z/OS MVS システム・コマンド*」を参照してください。

JCL を使用した z/OS でのアプライ・プログラムの始動

サンプル・ディレクトリーに用意されているサンプル・スクリプトを変更して実行することにより、z/OS 上のアプライ・プログラムを始動できます。

手順

JCL を使用して z/OS 上でアプライ・プログラムを始動する方法

1. アプライ・ジョブの PARM フィールドに、適切な呼び出しパラメーターを指定することにより、z/OS 用の JCL を準備します。
2. サイトの要件に合うように、JCL を調整します。

   z/OS オペレーティング・システムの場合の、呼び出し JCL でのこの行の例を次に示します。

   ```
   //apyasn EXEC PGM=ASNAPPLY,PARM='control_server=CTLDB1
                              DB2_SUBSYSTEM=DSN
                              apply_qual=myqual spillfile=disk'
   ```

   UNIX および Windows オペレーティング・システムの場合の、呼び出し JCL でのこの行の例を次に示します。

   ```
   //apyasn EXEC PGM=ASNAPPLY,PARM='control_server=CTLDB1
                              apply_qual=myqual spillfile=disk'
   ```

3. TSO または MVS コンソールから JCL をサブミットします。

JCL を使用した z/OS でのキャプチャー・プログラムの始動

サンプル・ディレクトリーに用意されているサンプル・スクリプトを変更して実行することにより、z/OS 上のキャプチャー・プログラムを始動できます。

手順

JCL を使用して z/OS 上でキャプチャー・プログラムを始動する方法

1. z/OS 用の JCL を準備します。
   a. キャプチャー・ジョブの PARM フィールドに、オプションの適切な呼び出しパラメーターを指定します。
   b. システム全体の /etc/profile ファイル、またはレプリケーション・プログラムを実行しているユーザーのホーム・ディレクトリーにある .profile ファイルで TZ 環境変数を指定しなかった場合、JCL で TZ 環境変数と言語環境変数を設定しなければなりません。TZ 変数の設定に関する詳細は、「*WebSphere Information Integration Replication Installation and Customization Guide for z/OS*」を参照してください。

   次に示す呼び出し JCL の行の例では、TZ および LANG 変数を設定しています。

   ```
   //CAPJFA EXEC PGM=ASNCAP, PARM='ENVAR('TZ=PST8PDT','LANG=en_US')/
                             DSN6 cold capture_schema=JFA autostop'
   ```

2. TSO または MVS コンソールから JCL をサブミットします。

レプリケーションおよび発行を自動的に再始動するために自動リスタート・マネージャー (ARM) を使用する方法 (z/OS)

自動リスタート・マネージャー (ARM) のリカバリー・システムを z/OS で使用して、Q キャプチャー、Q アプライ、キャプチャー、アプライ、およびレプリケーション・アラート・モニターの各プログラムを再始動することができます。

始める前に

ARM がインストール済みであり、レプリケーション・プログラムが正しく設定されていることを確認してください。ARM をレプリケーション・プログラムで使用するには、このプログラムが必ず APF 許可されるようにしてください。例えば、Q アプライ、アプライ、またはレプリケーション・アラート・モニターの各プログラムで ARM を使用する場合は、適切なロード・モジュールを APF 許可ライブラリーにコピーする必要があります。(Q キャプチャー・プログラムおよびキャプチャー・プログラムは、ARM を使用するかしないかに関係なく、必ず APF 許可にする必要があります。)

このタスクについて

ARM とは z/OS のリカバリー機能で、特定のバッチ・ジョブや開始済みタスクの可用性を向上させることができます。ジョブまたはタスクが失敗するか、ジョブやタスクを実行しているシステムに障害が発生した場合、ARM はオペレーターの介入なしに、ジョブまたはタスクを再始動できます。

ARM は、処理対象のアプリケーションを識別するためにエレメント名を使用します。ARM 対応の各アプリケーションは、それ自身のユニークなエレメント名を生成し、ARM とのすべてのコミュニケーションにこの名前を使用します。 ARM はエレメント名をトラッキングし、エレメント名に対して再始動ポリシーを定義します。ARM の設定についての詳細は、「*z/OS MVS プログラミング: シスプレックス・サービス・ガイド*」を参照してください。

手順

ARM を使用してレプリケーション・プログラムと発行プログラムを自動的に再始動するには、以下のようにします。

1. ARM を構成するときに、以下のエレメント名のいずれかを指定します。

プログラム	エレメント名
Q キャプチャー	ASNQC*xxxxyyyy*
Q アプライ	ASNQA*xxxxyyyy*
キャプチャー	ASNTC *xxxxyyyy*
アプライ	ASNTA *xxxxyyyy*
レプリケーション・アラート・モニター	ASNAM *xxxxyyyy*

この *xxxx* は DB2 サブシステム名であり、*yyyy* はデータ共有メンバー名です (後者はデータ共有構成の場合にのみ必要)。エレメント名の長さは常に 16 文字であり、ブランクが埋め込まれます。

2. オプション: レプリケーション・プログラムまたは発行プログラムの複数のインスタンスをデータ共用メンバーの中で実行している場合は、プログラムを始動して、各プログラム・インスタンスにユニークな ARM エレメント名を作成するときに **arm** パラメーターを指定します。

   arm パラメーターは、以前の表にリストされているエレメント名に付加された 3 文字の値をとります。構文は **arm=***zzz* で、*zzz* には任意の長さの英数字ストリングが可能です。ただしレプリケーション・プログラムは、ユニークな 16 バイト名を作成するために、現在の名前に 3 文字以内の文字のみを連結し、必要に応じてブランクを埋め込みます。

レプリケーション・プログラムは初期化時にエレメント名を使用して ARM に登録されます。登録時にイベント出口を ARM に提供することはありません。レプリケーション・プログラムは z/OS サブシステムとして実行されるのではないため、イベント出口は必要ありません。登録済みプログラムが異常終了した場合 (例えばセグメント違反の発生)、 ARM は登録済みプログラムを再始動します。登録されたレプリケーション・プログラムは、通常終了した場合 (例えば、STOP コマンドによる終了)、または無効な登録を検出した場合は、登録解除されます。

ヒント: パラメーター **term=**n を使用して Q キャプチャー、Q アプライ、キャプチャー、アプライ、またはレプリケーション・アラート・モニターの各プログラムを始動すると、DB2 が静止または停止してもプログラムは停止しません。この場合、プログラムは ARM から登録解除されません。プログラムは実行を続けますが、DB2 が静止解除されるか始動するまでは、実際の作業を実行しません。

データ共有モードへのレプリケーション環境の移行 (z/OS)

キャプチャー・プログラムが非データ共有モードで実行されているときに、システムをデータ共有モードに移行するときには、ASNPLXFY ユーティリティーを一度実行して、システムが Sysplex で実行されるように、準備する必要があります。

始める前に

キャプチャー・プログラムを実行するために使用したものと同じユーザー ID を使用するか、同じ特権を持つものを使用します。ASNPLXFY ユーティリティーが APF 許可であることを確認してください。 ASNPLXFY プランは、サブシステムにバインドする必要があります。また、サブシステムはデータ共有モードで実行されている必要があります。このユーティリティーのバインディングの詳細は、プログラム・ディレクトリーを参照してください。

このタスクについて

キャプチャー・プログラムが正しい LRSN から開始するように、キャプチャー・プログラムをウォーム・スタートする前に、このユーティリティーをデータ共用構成で実行してください。このユーティリティーは、IBMSNAP_RESTART 表の中のデータを移行します。非データ共有のログ・シーケンス番号 (RBA) は、データ共有環境での同等のシーケンス番号 (LRSN) に変換されます。

手順

USS データ共有環境で ASNPLXFY ユーティリティーを実行するには、以下のようにします。

1. キャプチャー・プログラムを停止します。
2. コマンド行から ASNPLXFY コマンドを発行します。 例えば、次のようにします。

   ```
   ASNPLXFY yoursubsystem captureschema
   ```

   サブシステムの名前が必要です。キャプチャー・スキーマはオプションです。デフォルトのキャプチャー・スキーマは ASN です。
3. キャプチャー・プログラムをウォーム・スタートします。

第 12 章 SQL レプリケーション環境の変更

以下のトピックでは、Q レプリケーション環境に変更を加える日常の操作に関する手順や問題点について説明しています。

新規オブジェクトの登録

レプリケーション環境では、いつでも新規の表、ビュー、またはニックネームを登録できます。キャプチャー・プログラムを再初期化する必要はありません。

このタスクについて

新しく登録されたオブジェクトは、そのオブジェクトを参照するサブスクリプション・セットがアプライ・プログラムによって処理されたときに、キャプチャー・プログラムによって自動的に初期化されます。アプライ・プログラムは、この新規オブジェクトの変更のキャプチャーを開始するようにキャプチャー・プログラムにシグナルを送ります。

手順

新規オブジェクトを登録するには、以下のようにします。

以下の方法のいずれかを使用して、新規オブジェクトを登録します。

方法	説明

ASNCLP コマンド行プログラム	CREATE REGISTRATION コマンドを使用して、ソース表、ソース・ビュー、ソース・ニックネームを登録します。例えば、以下のコマンドは、環境を設定し、DEPARTMENT 表をフル・リフレッシュ・レプリケーションのために DB2 SAMPLE データベースに登録します。
	`SET SERVER CAPTURE TO DB SAMPLE;`
	`SET OUTPUT CAPTURE SCRIPT "registernew.sql";`
	`SET LOG "registernew.err";`
	`SET RUN SCRIPT LATER;`
	`CREATE REGISTRATION (DB2ADMIN.DEPARTMENT) FULL REFRESH ONLY;`

レプリケーション・センター	以下のいずれかのウィンドウを使用します。 • 「登録済み表のプロパティー」ノートブック • 「登録済みビューのプロパティー」ノートブック • 「登録済みニックネーム・プロパティー」ノートブック これらのウィンドウを開くには、オブジェクト・ツリーのキャプチャー・コントロール・サーバーの下にある「**登録済み表**」、「**登録済みビュー**」、「**登録済みニックネーム**」のいずれかのフォルダーをクリックし、内容ペインで登録済みのオブジェクトを右クリックし、「**プロパティー**」を選択します。

方法	説明
System i ADDDPRREG システム・コマンド	System i の場合は、DPR 登録の追加 (ADDDPRREG) コマンドを使用して、新しい表を登録します。

登録済みオブジェクトの登録属性の変更

既存の登録済みオブジェクトの登録属性はいつでも変更できます。

手順

登録済みのオブジェクトの登録属性を変更するには、以下のようにします。

1. 次の方法のいずれかを使用して、属性を変更します。

方法	説明

ASNCLP コマンド行プログラム	ALTER REGISTRATION コマンドを使用して、登録済みのオブジェクトのプロパティーを変更します。例えば、以下のコマンドは、環境を設定し、DB2 SAMPLE データベースの STAFF 表の登録を変更することによって、削除と挿入の対として更新をキャプチャーするようにします。
	
	`SET SERVER CAPTURE TO DB SAMPLE;`
	`SET OUTPUT CAPTURE SCRIPT "register.sql";`
	`SET LOG "register.err";`
	`SET RUN SCRIPT LATER;`
	`ALTER REGISTRATION (DB2ADMIN.STAFF)`
	`UPDATE AS DELETE INSERT ON;`

レプリケーション・センター	以下のいずれかのウィンドウを使用します。 • 「登録済み表のプロパティー」ノートブック • 「登録済みビューのプロパティー」ノートブック • 「登録済みニックネーム・プロパティー」ノートブック これらのウィンドウを開くには、オブジェクト・ツリーのキャプチャー・コントロール・サーバーの下にある「**登録済み表**」、「**登録済みビュー**」、「**登録済みニックネーム**」のいずれかのフォルダーをクリックし、内容ペインで登録済みのオブジェクトを右クリックし、「**プロパティー**」を選択します。

2. 属性を変更した後に、キャプチャー・プログラムを再初期化します。

ソース表への列の追加

登録済みソース表に列を追加する必要がある場合は、最初に DB2 レプリケーションがこの表をどのように使用しているかを考えてください。このソース表の中の新しい列を複製する必要がある場合には、既存のキャプチャー・プログラムおよびアプライ・プログラムが新規列を認識し、中断なしに処理を続行できるようにしてください。

始める前に

この手順を使用する前に、ソース表、変更データ (CD) 表、およびターゲット表の構造、そしてシステムで定義されている登録およびサブスクリプション・セットについてよく調べておいてください。

制約事項

System i 主キーとして相対レコード番号 (RRN) を使用する System i の表に列を追加する場合は、これらのステップを使用しないでください。RRN は CD 表の最後の列である必要があります。RRN を含む System i の表に列を追加するときには、登録を除去し、ソース表に列を追加してから、RRN がキャプチャーされることを指定して、新しい登録として再度この表を追加してください。

DB2 以外のリレーショナル・データベース上の登録済みソースに列を追加するためにこれらのステップを使用することはできません。DB2 以外のリレーショナル・ソースの登録には、変更のキャプチャーに使用されるトリガーのセットが含まれます。これらのトリガーを変更することはできません。このため、このソース表に新規列を追加し、これらの列の中のデータを複製する必要があるときには、既存の登録済みソースをドロップしてから、再作成する必要があります。

このタスクについて

新規列の中のデータを複製するかどうかによって、特別の処理ステップを実行する必要があります。

複製しない

新規列の中のデータを複製する必要がない場合には、特別な処理ステップを実行する必要はありません。キャプチャー・プログラムは即時に変更を認識し、実行を続けます。

複製する

これらの新規列の中のデータを複製する必要がある場合には、新規列のデータがキャプチャーされ、キャプチャー・プログラムおよびアプライ・プログラムがエラーなしに実行を継続できるように、以下のステップを実行します。

手順

ソース表に列を追加するには、以下のようにします。

1. 変更するソース表に対するすべてのアクティビティーを静止します。
2. キャプチャー・プログラムを停止します。
3. オプション: この手順の間キャプチャー・プログラムをアクティブにしておく必要がある場合には、ソース表に対するアクティビティーを停止した後、IBMSNAP_SIGNAL 表に USER シグナルを挿入します。キャプチャー・プログラムが USER シグナルを処理するまで待ちます。 キャプチャー・プログラムは USER シグナルを処理すると、関連する CD 表に対して処理を必要とするアクティビティーがなくなるため、この CD 表へのアクセスを必要としなくなります。
4. レプリケーション・センターから、このソース表にサブスクライブするすべてのサブスクリプション・セットを非活動化します。

注: この処理中にサブスクリプション・セットを非活動化したくない場合は、新規列を追加しているときに、このソース表に対して、これらのサブスクリプション・セットに関連付けられたアプライ・プログラムが実行されていないことを確認してください。この代わりに、これらのアプライ・プログラムが、前の USER シグナルに関連付けられたシグナルのログ・シーケンス番号 (LSN) までデータの処理を終了していることを確認することもできます。

このステップの方法では、表を変更できるように、CD 表に対して排他的アクセスを確保しています。

5. SQL の ALTER TABLE ADD ステートメントを実行して、ソース表に新しい列を追加します。
6. ASNCLP コマンド行プログラムの ALTER REGISTRATION コマンドまたはレプリケーション・センターの「登録済み表のプロパティー」ノートブックを使用して、CD 表に新しい列を追加します。 キャプチャー・プログラムは自動的に登録を再初期化し、新規列で初めてログ・データを読み取ったときに、これらの新規列の変更をキャプチャーします。
7. SQL の ALTER TABLE ADD ステートメントを実行して、ターゲット表に新しい列を追加します。
8. レプリケーション・センターから、まだ非活動化していないすべての関連サブスクリプション・セットを非活動化します。 絶対に必要である場合は、ここでこのソース表に対するアクティビティーを再開できます。しかし、関連するサブスクリプション・セットがまだ変更されていないため、これらの新規列に対して行われた変更が失われないように、これらのサブスクリプション・セットは非活動化しておく必要があります。
9. ASNCLP コマンド行プログラムの ALTER MEMBER ADD COLS コマンドまたはレプリケーション・センターの「ターゲット表への列の追加」ウィンドウを使用して、関連するサブスクリプション・セット・メンバーに新しい列を追加します。
10. z/OS Linux UNIX Windows **opt4one** を y に設定してアプライ・プログラムを実行中である場合は、アプライ・プログラムをいったん停止してから再始動します。
11. サブスクリプション・セットを再活動化します。

登録済みオブジェクトの変更のキャプチャーの停止

キャプチャー・プログラムがオブジェクトに必要なすべての処理を終了できるように、登録済みオブジェクトを削除する前に、オブジェクトを非活動化する必要があります。また、このオブジェクトで一時的に変更のキャプチャーを停止しても、他の登録済みオブジェクトに対してはキャプチャー・プログラムを実行し続けておきたい場合も、登録済みオブジェクトを非活動化できます。

制約事項

非活動化できる DB2 登録済みオブジェクトは、キャプチャー・プログラム・ソースとして定義されているものだけです。

キャプチャー・トリガーから使用される DB2 以外のリレーショナル・データベース・オブジェクトを非活動化することはできません。

このタスクについて

キャプチャー・プログラムは、非活動化されたソース・オブジェクトについては変更のキャプチャーを停止します。しかし、これらのソース・オブジェクトに関連する変更データ (CD) 表、登録属性、およびサブスクリプション・セットは、システム上に残ります。

登録済みオブジェクトを非活動化する前に、この登録済みオブジェクトに関連付けられたすべてのサブスクリプション・セットを非活動化する必要があります。これにより、ユーザーがオブジェクトを削除する、または再度活動化する準備が整う前に、アプライ・プログラムがオブジェクトを自動的に再活動化し、非活動化処理に介入してくることを防止できます。

オブジェクトが非活動化され、SQL レプリケーションがそのオブジェクトに対する変更のキャプチャーを停止すると、登録済みオブジェクトに関連付けられたすべてのサブスクリプション・セットが影響を受けます。これらのサブスクリプション・セットの実行を続けたい場合は、この登録済みオブジェクトをソースとして使用するサブスクリプション・セット・メンバーを、非活動化されたサブスクリプション・セットから除去する必要があります。

手順

登録済みオブジェクトを非活動化するには、次のようにします。

1. レプリケーション・センターを使用して、関連するすべてのサブスクリプション・セットを非活動化します。 「**サブスクリプション・セット**」フォルダーをクリックし、内容ペインでアクティブなサブスクリプション・セットを右クリックし、「**非アクティブ化**」を選択します。
2. 次の方法のいずれかを使用して、登録済みオブジェクトを非活動化します。

方法	説明
レプリケーション・センター	「**登録済み表**」フォルダーをクリックし、内容ペインで登録済み表を右クリックし、「**変更の取り込みを停止する**」を選択します。
SQL	手動で IBMSNAP_SIGNAL 表に CAPSTOP シグナルを挿入します。

登録の再活動化を可能にする

登録を再活動化すると、キャプチャー・プログラムは、アプライ・プログラムが CAPSTART シグナルを送信した後に登録を再活動化します。しかし、予期しないエラーのためにキャプチャー・プログラムが登録を非活動化した場合は、登録を再活動化するために特別の処理を行う必要があります。

始める前に

非活動化された登録に関してキャプチャー・プログラムから生成されたエラー・メッセージを読みます。

キャプチャー・コントロール表の構成、およびシステム上で実行中のキャプチャー・プログラムについて調べておいてください。

このタスクについて

予期しないエラーが発生すると、この登録の STOP_ON_ERROR 列の値が N に設定されている場合、キャプチャー・プログラムは、IBMSNAP_REGISTER 表の中の STATE 列の値を S (Stopped) に設定することがあります。この STATE 列の値は、キャプチャー・プログラムがこの登録の処理を停止したこと、そして登録の修復が必要であることを意味します。アプライ・プログラムは停止状態の登録に対して CAPSTART シグナルを発行することはありません。

手順

予期しないエラーを修正し、登録の再活動化を可能にするには、以下のようにします。

1. エラー・メッセージに含まれた情報を使用して、登録を変更します。
2. キャプチャー・コントロール・サーバーから、次の SQL スクリプトを実行して、 IBMSNAP_REGISTER 表の中の STATE 列をリセットします。

```
UPDATE schema.IBMSNAP_REGISTER
   SET STATE             = 'I'
 WHERE
       SOURCE_OWNER      = 'SrcSchema'  AND
       SOURCE_TABLE      = 'SrcTbl'     AND
       SOURCE_VIEW_QUAL  =  SrcVwQual   AND
       STATE             = 'S';
```

   この *schema* はキャプチャー・スキーマの名前であり、*SrcSchema* は登録済みソース表スキーマであり、*SrcTbl* は登録済みソース表の名前であり、 *SrcVwQual* はこのソース表のソース・ビュー修飾子です。 STATE 列が I (Inactive) に設定されると、キャプチャー・プログラムは、通常はアプライ・プログラムから出される CAPSTART シグナルを受け取りしだい、データのキャプチャーを開始することができます。

アクティブな登録のソース表が誤って DATA CAPTURE NONE に変更されたとします (本来は DATA CAPTURE CHANGES である必要があります)。また、エラーを検出してもキャプチャー・プログラムが停止しないことを指定する STOP_ON_ERROR = 'N' を使用してこの登録が定義されていたとします。キャプチャー・プログラムは次回の再始動、または再初期化時に、ソース表のこの誤った条件を認識し、この登録について、 IBMSNAP_REGISTER 表の中の STATE 列を S (Stopped) に設定します。登録は停止状態になるため、アプライ・プログラムが対応するサブスクリプション・セットを処理しようとすると、エラー・メッセージが発行されます。以下のことを行う必要があります。

- 表のオプションを DATA CAPTURE CHANGES にリセットする ALTER TABLE ステートメントをサブミットして、 SQL によりソース表の設定を訂正します。
- 上記の SQL スクリプトを使用して、手動で登録を停止状態から非活動状態にリセットします。

アプライ・プログラムはサブスクリプション・セット全体に対してフル・リフレッシュを実行します。

登録の除去

ユーザーが登録をドロップすると、SQL レプリケーションはオブジェクトの登録をドロップし、関連する変更データ (CD) 表または整合変更データ (CCD) 表をドロップし、 DB2 以外のリレーショナル・データベース・ソースの CCD オブジェクト・ニックネームおよびキャプチャー・トリガーをドロップします。実際のソース表またはビューはデータベース内に残ります。

始める前に

- キャプチャー・プログラムがこのオブジェクトの現行の処理を終了できるように、登録を非活動化します。
- ソースに関連したサブスクリプション・セットを非活動化します。

このタスクについて

重要: 非活動化は非同期処理です。オブジェクトの除去前に必ず非活動化処理が終了するようにしてください。

キャプチャー・プログラムの実行中に変更が加えられた場合は、キャプチャー・プログラムを再初期化するか、いったん停止して再始動するまで、キャプチャー・プログラムはその変更を認識しません。

手順

登録を除去するには、次の方法のいずれかを使用します。

方法	説明

ASNCLP コマンド行プログラム	DROP REGISTRATION コマンドを使用して、1 つ以上の登録をドロップします。例えば、以下のコマンドは、環境を設定し、DB2 SAMPLE データベースでの DEPARTMENT 表の登録をドロップします。
	`SET SERVER CAPTURE TO DB SAMPLE;`
	`SET OUTPUT CAPTURE SCRIPT "dropregis.sql";`
	`SET LOG "dropregis.err";`
	`SET RUN SCRIPT LATER;`
	`DROP REGISTRATION (DB2ADMIN.DEPARTMENT);`

レプリケーション・センター	「登録済み表の削除」または「登録済みビューの削除」ウィンドウを使用します。これらのウィンドウを開くには、オブジェクト・ツリーのキャプチャー・コントロール・サーバーの下にある「**登録済み表**」フォルダーまたは「**登録済みビュー**」フォルダーをクリックし、内容ペインで登録済みのオブジェクトを右クリックし、「**削除**」を選択します。
System i RMVDPRREG システム・コマンド	DPR 登録の除去 (RMVDPRREG) コマンドを使用して、IBMSNAP_REGISTER 表から 1 つのソース表を除去します。

キャプチャー・スキーマの変更

既存のキャプチャー・スキーマを変更できます。

始める前に

- SQL レプリケーション・コントロール表、およびシステムに定義されているサブスクリプション・セットについてよく調べておいてください。
- 新しいキャプチャー・スキーマ名を決定します。
- キャプチャー・コントロール・サーバー、およびこのキャプチャー・コントロール・サーバーに関連するすべてのアプライ・コントロール・サーバーが、バージョン 8 以降に移行済みであることを確認してください。

制約事項

ソース・サーバーが DB2 以外のリレーショナル・データベースである場合は、この手順は使用しないでください。

このタスクについて

ヒント: モニター定義が設定されているか、あるいは、変更しようとしているキャプチャー・スキーマの下でレプリケーション・アラート・モニター・プログラムを始動している場合には、これらのモニター定義をドロップしてください。キャプチャー・スキーマを変更した後、新しいキャプチャー・スキーマ名を使用してモニター定義を再作成します。次に、asnmcmd reinit システム・コマンドを使用して、関連するモニターを再初期化できます。asnmcmd stop システム・コマンドを使用してモニターを停止してから、asnmon システム・コマンドを使用してプログラムを再始動することもできます。

手順

キャプチャー・スキーマを変更するには、以下のようにします。

1. 新しいキャプチャー・スキーマのためのコントロール表を作成します。
2. キャプチャー・プログラムを停止します。
3. レプリケーション・センターを使用して、関連するすべてのサブスクリプション・セットを非活動化します。
4. アプライ・コントロール・サーバーから次の SQL ステートメントを実行して、このキャプチャー・スキーマに属するソース表を持つ関連するサブスクリプション・セットのキャプチャー・スキーマ名を変更します。

   ```
   UPDATE ASN.IBMSNAP_SUBS_SET
      SET CAPTURE_SCHEMA    = 'NewSchema'
    WHERE
          CAPTURE_SCHEMA    = 'ExistingSchema';
   ```

   この *NewSchema* は新しいキャプチャー・スキーマ名であり、*ExistingSchema* は、変更しようとしているキャプチャー・スキーマの名前です。
5. このキャプチャー・スキーマに登録されたターゲット表 (例えば CCD 表またはレプリカ・タイプ表) を持つサブスクリプション・セットを作成してある場合には、アプライ・コントロール・サーバーから次の SQL ステートメントを実行して、これらのサブスクリプション・セットのターゲット・スキーマ名を変更します。

```
UPDATE ASN.IBMSNAP_SUBS_SET
   SET TGT_CAPTURE_SCHEMA    = 'NewSchema'
 WHERE
       TGT_CAPTURE_SCHEMA    = 'ExistingSchema';
```

この *NewSchema* は新しいキャプチャー・スキーマ名であり、*ExistingSchema* は、変更しようとしているキャプチャー・スキーマの名前です。

6. キャプチャー・コントロール・サーバーから SQL ステートメントを実行して、既存のキャプチャー・コントロール表のそれぞれから、ステップ 1 で作成した対応する新しいキャプチャー・コントロール表のそれぞれにアクティブ情報をコピーします。例えば、 IBMSNAP_REGISTER 表にアクティブ情報をコピーするには、次のようにします。

```
INSERT INTO NewSchema.IBMSNAP_REGISTER
       SELECT * FROM
              ExistingSchema.IBMSNAP_REGISTER;
```

この *NewSchema* は新しいキャプチャー・スキーマ名であり、*ExistingSchema* は、変更しようとしているキャプチャー・スキーマの名前です。

次の表を含めて、既存のキャプチャー・コントロール表のそれぞれに対してこのステップを繰り返します。

- IBMSNAP_CAPMON
- IBMSNAP_CAPPARMS
- IBMSNAP_CAPTRACE
- IBMSNAP_PRUNCNTL
- IBMSNAP_PRUNE_SET
- IBMSNAP_REG_EXT (System i のみ)
- IBMSNAP_REGISTER
- IBMSNAP_RESTART
- IBMSNAP_SIGNAL
- IBMSNAP_UOW

IBMSNAP_CAPENQ (UNIX、Windows、z/OS の場合) または IBMSNAP_PRUNE_LOCK コントロール表には行が含まれていないため、これらの表についてはこのステップを繰り返す必要はありません。CD 表は変更しないでください。

7. レプリケーション・センターまたは ASNCLP を使用して、既存のスキーマ、および関連するキャプチャー・コントロール表をドロップします。
8. 新しいスキーマ名を使用してキャプチャー・プログラムを再始動します。
9. レプリケーション・センターを使用して、関連するサブスクリプション・セットを再び活動化します。

新規サブスクリプション・セットの作成

既存の登録済みオブジェクトについては、いつでも新しいサブスクリプション・セットを作成し、新しいサブスクリプション・セット・メンバーをセットに追加できます。

始める前に

新しいサブスクリプション・セットを作成する前に、ソースとして使用する表またはビューを登録します。

制約事項

対応するアプライ・プログラムがアクティブである場合は、サブスクリプション・セットが完全に定義されるまで、新しいサブスクリプション・セットをアクティブ化しないでください。

このタスクについて

この手順は、サブスクリプション・セット・メンバーを含む、または含まない、新しいサブスクリプション・セットの追加に使用されます。

手順

新しいサブスクリプション・セットを作成するには、次の方法のいずれかを使用します。

方法	説明
ASNCLP コマンド行プログラム	CREATE SUBSCRIPTION SET コマンドを使用して、空のセットを作成します。
レプリケーション・センター	「**サブスクリプション・セットの作成**」ノートブックを使用して、セットを作成してメンバーを追加するか、空のセットを作成します。 そのノートブックを開くには、セットを定義するアプライ・コントロール・サーバーを展開し、「**サブスクリプション・セット**」フォルダーを右クリックし、「**作成**」をクリックします。
System i ADDDPRSUB システム・コマンド	1 つのメンバーを持つ、またはメンバーを持たないサブスクリプション・セットを作成するには、DPR サブスクリプション・セットの追加 (ADDDPRSUB) コマンドを使用します。

既存のサブスクリプション・セットに新しいサブスクリプション・セット・メンバーを追加する

1 つ以上の既存のサブスクリプション・セットに、それぞれ同じソース表を使用する 1 つ以上のメンバーを追加できます。例えば、3 つのサブスクリプション・セットを選択した場合は、それぞれのサブスクリプション・セットに、同じレプリケーション・ソースを使用する 1 つのメンバーを追加できます。

このタスクについて

サブスクリプション・セットにメンバーを追加すると、アプライ・コントロール表にその新しいメンバーに関する情報を挿入することになります。ほとんどの場合、アプライ・プログラムは、次回のアプライ・サイクルの始めにその情報を読み取ります。

ただし、OPT4ONE オプション (Linux、UNIX、Windows、z/OS) または OPTSNGSET オプション (System i) を設定した状態で処理が進行しているサブスクリプション・セットにメンバーを追加する場合は、そのサブスクリプション・セットに関してアプライ・プログラムをいったん停止してから再始動する必要があります。 OPT4ONE オプションを設定してセットを処理する場合、アプライ・プログラムは、セットに関するコントロール表の情報をメモリーに読み込むことによって、毎回のアプライ・サイクルの始めにセットに関する情報をコントロール表から読み取らなくてもよいようにします。

メンバーのソース表が差分レプリケーション用に登録されている状態で、キャプチャー・プログラムがすでに実行中になっている場合は、メンバーを追加する前にキャプチャー・プログラムを停止したり再初期化したりする必要はありません。追加されたメンバーは、ソースとして登録済みの表を使用しなければならないので、キャプチャー・プログラムはすぐにそのメンバーに関する変更をキャプチャーし始めます。

手順

既存のサブスクリプション・セットに新しいサブスクリプション・セット・メンバーを追加するには、次の方法のいずれかを使用します。

方法	説明
ASNCLP コマンド行プログラム	CREATE MEMBER コマンドを使用して、既存のサブスクリプション・セットにサブスクリプション・セット・メンバーを追加します。
レプリケーション・センター	「**サブスクリプション・セットにメンバーを追加する**」ノートブックを使用します。 そのノートブックを開くには、「**登録済み表**」フォルダーをクリックします。内容ペインで、使用する登録済み表を右クリックし、「**メンバーの追加**」をクリックします。
System i ADDDPRSUBM システム・コマンド	既存のサブスクリプション・セットにメンバーを追加するには、DPR サブスクリプション・セット・メンバーの追加 (ADDDPRSUBM) コマンドを使用します。

既存のサブスクリプション・セットでサブスクリプション・セット・メンバーを使用不可にする

アプライ・プログラムが障害のあるサブスクリプション・セット・メンバーを無視して、残りのサブスクリプション・セットの処理を続行できるようにするには、障害のあるサブスクリプション・セット・メンバーを使用不可にする必要があります。

このタスクについて

サブスクリプション・セットに含まれている表に対するレプリケーションで問題があれば、アプライ・プログラムは、IBMSNAP_APPLYTRAIL 表にエラー・メッセージを挿入して、そのアプライ・サイクルで他のメンバーの処理を続行します。

手順

サブスクリプション・セット・メンバーを使用不可にするには、以下の SQL UPDATE ステートメントを実行します。

```
UPDATE ASN.IBMSNAP_SUBS_MEMBR
 SET MEMBER_STATE = 'D'
 WHERE APPLY_QUAL= apply_qualifier
                SET_NAME = set_name
                 WHOS_ON_FIRST = whos_on_first
                 SOURCE_OWNER = source_owner
   SOURCE_TABLE = source_table
  SOURCE_VIEW_QUAL = source_view_qualifier
  TARGET_OWNER = target_owner
  TARGET_TABLE = target_table
```

アプライ・プログラムは、メンバーが再度使用可能になるまでこのメンバーを処理しません。

既存のサブスクリプション・セットに対してサブスクリプション・セット・メンバーを使用可能にする

MEMBER_STATE を N (新規) に変更することによって、サブスクリプション・セットの使用不可のメンバーを追加したり再度使用可能にしたりすることができます。

手順

サブスクリプション・セット・メンバーを再び使用可能にするには、以下の SQL UPDATE ステートメントを実行します。

```
UPDATE ASN.IBMSNAP_SUBS_MEMBR
 SET MEMBER_STATE = 'N'
 WHERE APPLY_QUAL= apply_qualifier
                SET_NAME = set_name
                 WHOS_ON_FIRST = whos_on_first
                 SOURCE_OWNER = source_owner
   SOURCE_TABLE = source_table
  SOURCE_VIEW_QUAL = source_view_qualifier
  TARGET_OWNER = target_owner
  TARGET_TABLE = target_table
```

サブスクリプション・セットのプロパティーの変更

サブスクリプション・セットのプロパティーは、アプライが実行中で他のサブスクリプション・セットを処理しているときでも変更できます。その場合は、次回のアプライ・サイクルの前にセットを再び活動化します。

このタスクについて

以下のリストでは、変更しなければならない可能性がある属性について説明します。

- 更新を適用するスケジュール (時間ベースのレプリケーションまたはイベント・ベースのレプリケーション)
- サブスクリプション・ステートメント
- サブスクリプション・セット・メンバーの WHERE 文節述部
- コミット・カウント
- データ・ブロック値 (MAX_SYNCH_MINUTES)

最初にサブスクリプション・セットを非活動化することによって、変更を入力している間にアプライ・プログラムがセットを処理する状況を回避できます。サブスクリプション・セットを再び活動化した後に、アプライ・プログラムは、次のアプライ・サイクルでサブスクリプション・セットの変更を認識します。

手順

サブスクリプション・セットのプロパティーを変更するには、以下のようにします。

1. レプリケーション・センターを使用して、サブスクリプション・セットを非活動化します。
2. 次の方法のいずれかを使用して、サブスクリプション・セットを変更します。

方法	説明

ASNCLP コマンド行プログラム	ALTER SUBSCRIPTION SET コマンドを使用します。
	
	以下のコマンドは、環境を設定し、サブスクリプション・セット SET00 を変更して、タイミング・インターバルを 15 分に下げます。
	
	`SET SERVER CAPTURE TO DB SAMPLE;`
	`SET SERVER CONTROL TO DB TARGET;`
	`SET OUTPUT CAPTURE SCRIPT "capsubsetchg.sql"`
	`CONTROLSCRIPT "appsubsetchg.sql";`
	`SET LOG "subsetchg.err";`
	`SET RUN SCRIPT LATER;`
	`ALTER SUBSCRIPTION SET SETNAME SET00`
	`APPLYQUAL AQ00 SETTYPE R ACTIVATE YES`
	`TIMING INTERVAL 15 COMMIT COUNT NULL;`

レプリケーション・センター	「サブスクリプション・セット・プロパティー」ノートブックを使用します。そのノートブックを開くには、アプライ・コントロール・サーバーの中で「**サブスクリプション・セット**」フォルダーをクリックし、内容ペインでサブスクリプション・セットを右クリックし、「**プロパティー**」をクリックします。

3. サブスクリプション・セットを再活動化します。

z/OS Linux UNIX Windows アプライ・プログラムの **opt4one** パラメーターを y に設定した場合は、アプライ・プログラムをいったん停止してから再始動しないと、変更が認識されません。

サブスクリプション・セット名の変更

サブスクリプション・セットとそのすべてのメンバーをドロップしてから再作成する操作を実行しなくても、サブスクリプション・セットの名前を変更できます。

始める前に

これらの SQL ステートメントを実行する前に、SQL レプリケーション・コントロール表の構造、およびシステムで定義されているサブスクリプション・セットについてよく調べておいてください。

ヒント: モニター定義が設定されているか、サブスクリプション・セットでアラート条件を検出するようにレプリケーション・アラート・モニター・プログラムを始動してある場合には、これらの定義をドロップしてください。サブスクリプション・セット名を変更した後、レプリケーション・センターまたは ASNCLP を使用してモニター定義を再作成します。次に、asnmcmd reinit システム・コマンドを使用して、モニターを再初期化できます。asnmcmd stop コマンドを使用してモニターを停止してから、asnmon コマンドを使用してプログラムを再始動することもできます。

手順

サブスクリプション・セットの名前を変更するには、以下のようにします。

1. レプリケーション・センターを使用して、サブスクリプション・セットを非活動化します。
2. アプライ・コントロール・サーバーから次の SQL ステートメントを実行して、IBMSNAP_SUBS_SET 表、IBMSNAP_SUBS_MEMBR 表、および IBMSNAP_SUBS_COLS 表の中のサブスクリプション・セットの名前を変更します。

```
UPDATE ASN.IBMSNAP_SUBS_SET
   SET SET_NAME      = 'NewSetName'
 WHERE
       APPLY_QUAL    = 'ApplyQual'     AND
       SET_NAME      = 'ExistSetName'  AND
       WHOS_ON_FIRST = 'Val';

UPDATE ASN.IBMSNAP_SUBS_MEMBR
   SET SET_NAME      = 'NewSetName'
 WHERE
       APPLY_QUAL    = 'ApplyQual'      AND
       SET_NAME      = 'ExistSetName'   AND
       WHOS_ON_FIRST = 'Val';

UPDATE ASN.IBMSNAP_SUBS_COLS
   SET SET_NAME      = 'NewSetName'
 WHERE
       APPLY_QUAL    = 'ApplyQual'      AND
       SET_NAME      = 'ExistSetName'   AND
       WHOS_ON_FIRST = 'Val';
```

   この *NewSetName* は新しいサブスクリプション・セット名であり、*ApplyQual* はアプライ修飾子であり、*ExistSetName* はサブスクリプション・セットの既存の名前であり、*Val* は F または S のいずれかです。
3. このサブスクリプション・セットが事前または事後に実行される SQL ステートメントまたはプロシージャー呼び出しを使用する場合は、アプライ・コントロー

ル・サーバーから次の SQL スクリプトを実行して、IBMSNAP_SUBS_STMTS 表の中のサブスクリプション・セット名を変更します。

```
UPDATE ASN.IBMSNAP_SUBS_STMTS
   SET SET_NAME      = 'NewSetName'
 WHERE
       APPLY_QUAL    = 'ApplyQual'      AND
       SET_NAME      = 'ExistSetName'   AND
       WHOS_ON_FIRST = 'Val';
```

この *NewSetName* は新しいサブスクリプション・セット名であり、*ApplyQual* はアプライ修飾子であり、*ExistSetName* はサブスクリプション・セットの既存の名前であり、*Val* は F または S のいずれかです。

4. キャプチャー・コントロール・サーバーから次の SQL ステートメントを実行して、 IBMSNAP_PRUNE_SET 表および IBMSNAP_PRUNCNTL 表の中のサブスクリプション・セット名を変更します。

```
UPDATE Schema.IBMSNAP_PRUNE_SET
   SET SET_NAME      = 'NewSetName'
 WHERE
       APPLY_QUAL    = 'ApplyQual'     AND
       SET_NAME      = 'ExistSetName'  AND
       TARGET_SERVER = 'Target_Server';

UPDATE Schema.IBMSNAP_PRUNCNTL
   SET SET_NAME      = 'NewSetName'
 WHERE
       APPLY_QUAL    = 'ApplyQual'     AND
       SET_NAME      = 'ExistSetName'  AND
       TARGET_SERVER = 'Target_Server';
```

この *Schema* はキャプチャー・スキーマの名前であり、*NewSetName* は新しいサブスクリプション・セットの名前であり、*ApplyQual* はアプライ修飾子であり、*ExistSetName* はサブスクリプション・セットの既存の名前であり、*Target_Server* はターゲット表のデータベース・ロケーションです。

5. Linux、UNIX、Windows、z/OS でアプライ・プログラムを実行していて、**opt4one** を y に設定している場合には、アプライ・プログラムをいったん停止してから再始動します。
6. レプリケーション・センターから、サブスクリプション・セットを再び活動化します。

サブスクリプション・セットの分割

サブスクリプション・セットを複数のセットに分割できます。そのときに、サブスクリプション・セット情報を除去してから再作成する必要はありません。

始める前に

- これらの SQL ステートメントを実行する前に、SQL レプリケーション・コントロール表の構造、およびシステムで定義されているサブスクリプション・セットについてよく調べておいてください。
- 分割するサブスクリプション・セットのサブスクリプション・セット・メンバーを識別し、これらのサブスクリプション・セット・メンバーに関連するソース表およびターゲット表を判別します。
- 分割するサブスクリプション・セットのキャプチャー・コントロール・サーバー、ターゲット・サーバー、およびアプライ・コントロール・サーバーを識別し

ます。この手順を使用して作成する新しいサブスクリプション・セットでは、これらのキャプチャー・コントロール・サーバー、ターゲット・サーバー、およびアプライ・コントロール・サーバーのロケーションを使用する必要があります。

このタスクについて

ヒント: モニター定義が設定されているか、サブスクリプション・セットでアラート条件を検出するようにレプリケーション・アラート・モニター・プログラムを始動してある場合には、これらの定義をドロップしてください。サブスクリプション・セットを分割した後、レプリケーション・センターまたは ASNCLP を使用して、モニター定義を再作成します。次に、asnmcmd reinit システム・コマンドを使用して、モニターを再初期化できます。asnmcmd stop コマンドを使用してモニターを停止してから、asnmon コマンドを使用してプログラムを再始動することもできます。

手順

サブスクリプション・セットを分割するには、次のようにします。

1. レプリケーション・センターから、分割するサブスクリプション・セットを非活動化します。 「サブスクリプション・セット」フォルダーで内容ペインの中のアクティブなサブスクリプション・セットを右クリックし、「非アクティブ化」を選択します。
2. 新しいサブスクリプション・セットを作成します。 新しいセットは、IBMSNAP_SUBS_SET 表の中に新しい行で表されます。この新しいサブスクリプション・セットは非アクティブなままおいておきます。
3. アプライ・コントロール・サーバーから次の SQL ステートメントを実行して、既存のサブスクリプション・セットから IBMSNAP_SUBS_SET 表の中の新しいサブスクリプション・セットの行に情報をコピーします。

```
UPDATE ASN.IBMSNAP_SUBS_SET
   SET STATUS     =
                  (SELECT STATUS FROM ASN.IBMSNAP_SUBS_SET B
                    WHERE APPLY_QUAL    = 'ApplyQual' AND
                          SET_NAME      = 'ExistName' AND
                          WHOS_ON_FIRST = 'Val'),
       LASTRUN    =
                  (SELECT LASTRUN FROM ASN.IBMSNAP_SUBS_SET B
                    WHERE APPLY_QUAL    = 'ApplyQual' AND
                          SET_NAME      = 'ExistName' AND
                          WHOS_ON_FIRST = 'Val'),
       SYNCHPOINT  =
                   (SELECT SYNCHPOINT FROM ASN.IBMSNAP_SUBS_SET B
                    WHERE APPLY_QUAL    = 'ApplyQual' AND
                          SET_NAME      = 'ExistName' AND
                          WHOS_ON_FIRST = 'Val'),
       SYNCHTIME   =
                   (SELECT SYNCHTIME FROM ASN.IBMSNAP_SUBS_SET B
                    WHERE APPLY_QUAL    = 'ApplyQual' AND
                          SET_NAME      = 'ExistName' AND
                          WHOS_ON_FIRST = 'Val'),
       LASTSUCCESS =
                   (SELECT LASTSUCCESS FROM ASN.IBMSNAP_SUBS_SET B
                    WHERE APPLY_QUAL    = 'ApplyQual' AND
                          SET_NAME      = 'ExistName' AND
                          WHOS_ON_FIRST = 'Val')
```

```
 WHERE
       APPLY_QUAL         = 'ApplyQual' AND
       SET_NAME           = 'NewName'   AND
       WHOS_ON_FIRST      = 'Val';
```

この *ApplyQual* はアプライ修飾子であり、*ExistName* は分割する既存のサブスクリプション・セットの名前であり、*Val* は F または S のいずれかであり、*NewName* は、ユーザーが作成している新しいサブスクリプション・セットの名前です。

4. キャプチャー・コントロール・サーバーから次の SQL ステートメントを実行して、 IBMSNAP_PRUNE_SET 表に新しいサブスクリプション・セット用の新しい行を挿入します。

```
INSERT INTO Schema.IBMSNAP_PRUNE_SET
              (APPLY_QUALIFIER,
               SET_NAME,
               TARGET_SERVER,
               SYNCHTIME,
               SYNCHPOINT
       VALUES ('ApplyQual',
               'NewName',
               'Target_Server',
                NULL,
                x'00000000000000000000');
```

この *Schema* はキャプチャー・スキーマの名前であり、*ApplyQual* はアプライ修飾子であり、*NewName* は作成している新しいサブスクリプション・セットの名前であり、*Target_Server* はターゲット表のデータベース・ロケーションです。

5. キャプチャー・コントロール・サーバーから次の SQL ステートメントを実行して、 IBMSNAP_PRUNE_SET 表の中の既存のサブスクリプション・セットの行から新しいサブスクリプション・セットの行に情報をコピーします。

```
UPDATE Schema.IBMSNAP_PRUNE_SET
   SET SYNCHPOINT       =
             (SELECT SYNCHPOINT FROM Schema.IBMSNAP_PRUNE_SET B
               WHERE APPLY_QUAL    = 'ApplyQual' AND
                     SET_NAME      = 'ExistName' AND
                     TARGET_SERVER = 'Target_Server'),
       SYNCHTIME =
             (SELECT SYNCHTIME FROM  Schema.IBMSNAP_PRUNE_SET B
               WHERE APPLY_QUAL    = 'ApplyQual' AND
                     SET_NAME      = 'ExistName' AND
                     TARGET_SERVER = 'Target_Server')
 WHERE
       APPLY_QUAL    = 'ApplyQual' AND
       SET_NAME      = 'NewName'   AND
       TARGET_SERVER = 'Target_Server';
```

この *Schema* はキャプチャー・スキーマの名前であり、*ApplyQual* はアプライ修飾子であり、*ExistName* は分割する既存のサブスクリプション・セットの名前であり、*Target_Server* はターゲット表のデータベース・ロケーションであり、*NewName* はユーザーが作成しようとしている新しいサブスクリプション・セットの名前です。

6. アプライ・コントロール・サーバーから次の SQL ステートメントを実行して、 IBMSNAP_SUBS_MEMBR 表および IBMSNAP_SUBS_COLS 表で、新しいサブスクリプション・セットに移動する各サブスクリプション・セット・メンバーのサブスクリプション・セット名を変更します。

```
UPDATE ASN.IBMSNAP_SUBS_MEMBR
   SET SET_NAME          = 'NewName'
 WHERE
       APPLY_QUAL        = 'ApplyQual' AND
       SET_NAME          = 'ExistName' AND
       WHOS_ON_FIRST     = 'Val'       AND
       SOURCE_OWNER      = 'SrcSchema' AND
       SOURCE_TABLE      = 'SrcTbl'    AND
       SOURCE_VIEW_QUAL  =  SrcVwQual  AND
       TARGET_OWNER      = 'TgtSchema' AND
       TARGET_TABLE      = 'TgtTbl';

UPDATE ASN.IBMSNAP_SUBS_COLS
   SET SET_NAME          = 'NewName'
 WHERE
       APPLY_QUAL        = 'ApplyQual' AND
       SET_NAME          = 'ExistName' AND
       WHOS_ON_FIRST     = 'Val'       AND
       TARGET_OWNER      = 'TgtSchema' AND
       TARGET_TABLE      = 'TgtTbl';
```

この *NewName* はユーザーが作成している新しいサブスクリプション・セットであり、*ApplyQual* はアプライ修飾子であり、*ExistName* は分割する既存のサブスクリプション・セットであり、*Val* は F または S のいずれかであり、*SrcSchema* はソース表スキーマであり、*SrcTbl* はソース表名であり、*SrcVwQual* はこのソース表のソース・ビュー修飾子であり、 *TgtSchema* はターゲット表のスキーマであり、*TgtTbl* はターゲット表名です。

新しいサブスクリプション・セットに移動するサブスクリプション・セット・メンバーごとにこのステップを繰り返します。

7. 分割するサブスクリプション・セットが事前または事後に実行される SQL ステートメントまたはプロシージャー呼び出しを使用する場合は、IBMSNAP_SUBS_STMTS 表の中で適切なステートメントを新しいサブスクリプション・セットに移動します。

   a. アプライ・コントロール・サーバーから次の SQL スクリプトを実行して、ステートメントを移動します。

      ```
      UPDATE ASN.IBMSNAP_SUBS_STMTS
         SET SET_NAME      = 'NewName'
       WHERE
             APPLY_QUAL    = 'ApplyQual' AND
             SET_NAME      = 'ExistName' AND
             WHOS_ON_FIRST = 'Val'       AND
             STMT_NUMBER   in (Stmt1,Stmt2,..Stmtn);
      ```

      ここで、*NewName* はユーザーが作成している新しいサブスクリプション・セットの名前、*ApplyQual* はアプライ修飾子、*ExistName* は分割する既存のサブスクリプション・セットの名前、*Val* は F または S のいずれかの値、*Stmt1*、*Stmt2*、および *Stmtn* は、新しいサブスクリプション・セットに移動するステートメントの番号に対応します。

   b. 両方のサブスクリプション・セットについて、ステートメントの新しいカウントを反映するように、IBMSNAP_SUBS_SET 表の中の AUX_STMTS 列の値を調整します。必要であれば、重複しないようにステートメントの番号を変更します。

8. キャプチャー・コントロール・サーバーから次の SQL ステートメントを実行して、移動した各サブスクリプション・セット・メンバーについて、IBMSNAP_PRUNCNTL 表の中のサブスクリプション・セットの名前を変更します。

```
UPDATE Schema.IBMSNAP_PRUNCNTL
   SET SET_NAME          = 'NewName'
 WHERE
       APPLY_QUAL        = 'ApplyQual'      AND
       SET_NAME          = 'ExistName'      AND
       TARGET_SERVER     = 'Target_Server'  AND
       SOURCE_OWNER      = 'SrcSchema'      AND
       SOURCE_TABLE      = 'SrcTbl'         AND
       SOURCE_VIEW_QUAL  =  SrcVwQual       AND
       TARGET_OWNER      = 'TgtSchema'      AND
       TARGET_TABLE      = 'TgtTbl';
```

この *Schema* はキャプチャー・スキーマの名前であり、*NewName* はステップ 2 で作成した新しいサブスクリプション・セットの名前であり、*ApplyQual* はアプライ修飾子であり、*ExistName* は分割された既存のサブスクリプション・セットの名前であり、*Target_Server* はターゲット表のデータベース・ロケーションであり、*SrcSchema* はソース表スキーマであり、*SrcTbl* はソース表名であり、*SrcVwQual* はこのレプリケーション・ソース表のソース・ビュー修飾子であり、*TgtSchema* はターゲット表スキーマであり、*TgtTbl* はターゲット表名です。

新しいサブスクリプション・セットに移動したサブスクリプション・セット・メンバーごとにこのステップを繰り返します。

9. z/OS Linux UNIX Windows **opt4one** を y に設定してアプライ・プログラムを実行中である場合は、アプライ・プログラムをいったん停止してから再始動します。
10. レプリケーション・センターから、両方のサブスクリプション・セットを再活動化します。

サブスクリプション・セットのマージ

2 つのサブスクリプション・セットを 1 つにマージできます。2 つのサブスクリプション・セットの中のターゲット表が同じトランザクション整合性を持つようにする場合に、サブスクリプション・セット情報を削除して再作成したくなければ、サブスクリプション・セットをマージできます。

始める前に

これらの SQL ステートメントを実行する前に、SQL レプリケーション・コントロール表の構造、およびシステムで定義されているサブスクリプション・セットについてよく調べておいてください。

マージする各サブスクリプション・セットのキャプチャー・コントロール・サーバー、ターゲット・サーバー、およびアプライ・コントロール・サーバーを識別します。マージするすべてのサブスクリプション・セットが、同じキャプチャー・コントロール・サーバー、ターゲット・サーバー、およびアプライ・コントロール・サーバーを使用して作成されていることを確認します。

制約事項

マージされる 2 つのサブスクリプション・セットは、同じキャプチャー・サーバーから、そして同じキャプチャー・スキーマを通じてソース・データを得ている必要があります。

重要: サブスクリプション・セットがマージされたときにデータが失われないように、 2 つのサブスクリプション・セットでは、同じ同期点値までのソース・データが処理済みである必要があります。

手順

サブスクリプション・セットをマージするには、次のようにします。

1. 関連したキャプチャー・プログラムを停止します。 両方のサブスクリプション・セットが、IBMSNAP_SUBS_SET 表に示されたものと同じ同期点および同期時刻に達するまで待ちます。

   ヒント: キャプチャー・プログラムを停止したくない場合は、IBMSNAP_SIGNAL 表の中に USER シグナルを挿入し、END_SYNCHPOINT (IBMSNAP_SUBS_EVENT 表の中にある) を IBMSNAP_SIGNAL 表の中の SIGNAL_LSN 列の値に設定してイベントを生成し、このエンドポイントまでのデータのみが適用されるようにします。
2. レプリケーション・センターから、両方のサブスクリプション・セットを非活動化します。
3. アプライ・コントロール・サーバーから次の SQL ステートメントを実行して、別のサブスクリプション・セット内に移動するサブスクリプション・セットに対応する行を、 IBMSNAP_SUBS_SET 表から削除します。

   ```
   DELETE FROM ASN.IBMSNAP_SUBS_SET
    WHERE
          APPLY_QUAL    = 'ApplyQual'      AND
          SET_NAME      = 'Subset_To_Move' AND
          WHOS_ON_FIRST = 'Val';
   ```

   この *ApplyQual* はアプライ修飾子であり、*Subset_To_Move* は別の既存のサブスクリプション・セット内に移動するサブスクリプション・セットの名前であり、*Val* は F または S のいずれかです。
4. キャプチャー・コントロール・サーバーから次の SQL ステートメントを実行して、別のサブスクリプション・セット内に移動するサブスクリプション・セットに対応する行を、 IBMSNAP_PRUNE_SET 表から削除します。

   ```
   DELETE FROM Schema.IBMSNAP_PRUNE_SET
    WHERE
          APPLY_QUAL    = 'ApplyQual'      AND
          SET_NAME      = 'Subset_To_Move' AND
          TARGET_SERVER = 'Target_Server' ;
   ```

   この *Schema* はキャプチャー・スキーマの名前であり、*ApplyQual* はアプライ修飾子であり、*Subset_To_Move* は別の既存のサブスクリプション・セット内に移動するサブスクリプション・セットの名前であり、*Target_Server* はターゲット表のデータベース・ロケーションです。

5. アプライ・コントロール・サーバーから次の SQL ステートメントを実行して、IBMSNAP_SUBS_MEMBR 表および IBMSNAP_SUBS_COLS 表の中で、移動するサブスクリプション・セットの名前を、別のサブスクリプション・セットの名前に変更します。

```
UPDATE ASN.IBMSNAP_SUBS_MEMBR
   SET SET_NAME      = 'Existing_Merged_Subset'
 WHERE
       APPLY_QUAL    = 'ApplyQual'      AND
       SET_NAME      = 'Subset_To_Move' AND
       WHOS_ON_FIRST = 'Val';

UPDATE ASN.IBMSNAP_SUBS_COLS
   SET SET_NAME      = 'Existing_Merged_Subset'
 WHERE
       APPLY_QUAL    = 'ApplyQual'      AND
       SET_NAME      = 'Subset_To_Move' AND
       WHOS_ON_FIRST = 'Val';
```

この *Existing_Merged_Subset* は、移動するサブスクリプション・セットとマージされる既存のサブスクリプション・セットの名前であり、*ApplyQual* はアプライ修飾子であり、*Subset_To_Move* は既存のサブスクリプション・セット内に移動するサブスクリプション・セットの名前であり、 *Val* は F または S のいずれかです。

6. 移動するサブスクリプション・セットが事前または事後に実行される SQL ステートメントまたはプロシージャー呼び出しを使用する場合は、IBMSNAP_SUBS_STMTS 表の中でサブスクリプション・セットの名前を変更します。

   a. アプライ・コントロール・サーバーから次の SQL スクリプトを実行して、サブスクリプション・セットの名前を変更します。

```
UPDATE ASN.IBMSNAP_SUBS_STMTS
   SET SET_NAME      = 'Existing_Merged_Subset'
 WHERE
       APPLY_QUAL    = 'ApplyQual'      AND
       SET_NAME      = 'Subset_To_Move' AND
       WHOS_ON_FIRST = 'Val';
```

   この *Existing_Merged_Subset* は、移動するサブスクリプション・セットとマージされる既存のサブスクリプション・セットの名前であり、*ApplyQual* はアプライ修飾子であり、*Subset_To_Move* は既存のサブスクリプション・セット内に移動するサブスクリプション・セットの名前であり、 *Val* は F または S のいずれかです。

   b. 既存のマージされたサブスクリプション・セット内のステートメントの新しいカウントを反映するように、IBMSNAP_SUBS_SET 表の中の AUX_STMTS 列の値を調整します。必要であれば、重複しないようにステートメントの番号を変更します。

7. キャプチャー・コントロール・サーバーから次の SQL ステートメントを実行して、 IBMSNAP_PRUNCNTL 表の中で、移動したサブスクリプション・セットの名前を、マージされたサブスクリプション・セットの名前に変更します。

```
UPDATE Schema.IBMSNAP_PRUNCNTL
   SET SET_NAME      = 'Existing_Merged_Subset'
 WHERE
       APPLY_QUAL    = 'ApplyQual'      AND
       SET_NAME      = 'Subset_To_Move' AND
       TARGET_SERVER = 'Target_Server' ;
```

この *Schema* はキャプチャー・スキーマの名前であり、*Existing_Merged_Subset* は移動するサブスクリプション・セットとマージされる既存のサブスクリプション・セットの名前であり、*ApplyQual* はアプライ修飾子であり、*Subset_To_Move* は別の既存のサブスクリプション・セット内に移動するサブスクリプション・セットの名前であり、*Target_Server* はターゲット表のデータベース・ロケーションです。

8. z/OS Linux UNIX Windows **opt4one** を y に設定してアプライ・プログラムを実行中である場合は、アプライ・プログラムをいったん停止してから再始動します。
9. レプリケーション・センターから、マージしたサブスクリプション・セットを再び活動化します。

サブスクリプション・セットのアプライ修飾子の変更

サブスクリプション・セットのアプライ修飾子を変更する必要がある場合には、サブスクリプション・セットを削除および再作成することなく、 SQL を使用して変更を行うことができます。

始める前に

これらの SQL ステートメントを実行する前に、SQL レプリケーション・コントロール表の構造、およびシステムで定義されているサブスクリプション・セットについてよく調べておいてください。

また、次の情報も確認してください。

- 新しいアプライ修飾子の名前。
- 既存のアプライ修飾子から新しいアプライ修飾子に移動するサブスクリプション・セット。
- これらのサブスクリプション・セットに定義されている、事前または事後に実行される SQL ステートメントまたはプロシージャー呼び出し。

このタスクについて

同じアプライ修飾子を使用する複数のサブスクリプション・セットがある場合には、アプライ・プログラムのワークロードのバランスを取るために、いくつかのサブスクリプション・セットを新しいアプライ修飾子に移動することも考えられます。

ヒント: モニター定義が設定されているか、アプライ修飾子のアラート条件を検出するようにレプリケーション・アラート・モニター・プログラムを始動してある場合には、これらの定義をドロップしてください。修飾子を変更した後、レプリケーション・センターまたは ASNCLP を使用してモニター定義を再作成します。次に、asnmcmd reinit システム・コマンドを使用して、モニターを再初期化できます。asnmcmd stop コマンドを使用してモニターを停止してから、asnmon コマンドを使用してプログラムを再始動することもできます。

移動するサブスクリプション・セットごとにこの手順の SQL ステートメントを実行する必要があります。

手順

サブスクリプション・セットのアプライ修飾子を変更するには、以下のようにします。

1. レプリケーション・センターを使用して、変更するサブスクリプション・セットを非活動化します。
2. アプライ・コントロール・サーバーから次の SQL ステートメントを実行して、IBMSNAP_SUBS_SET 表、IBMSNAP_SUBS_MEMBR 表、および IBMSNAP_SUBS_COLS 表の中のサブスクリプション・セットのアプライ修飾子を変更します。

```
UPDATE ASN.IBMSNAP_SUBS_SET
   SET APPLY_QUAL    = 'NewApplyQual'
 WHERE
       APPLY_QUAL    = 'ExistApplyQual' AND
       SET_NAME      = 'Name'           AND
       WHOS_ON_FIRST = 'Val';

UPDATE ASN.IBMSNAP_SUBS_MEMBR
   SET APPLY_QUAL    = 'NewApplyQual'
 WHERE
       APPLY_QUAL    = 'ExistApplyQual' AND
       SET_NAME      = 'Name'           AND
       WHOS_ON_FIRST = 'Val';

UPDATE ASN.IBMSNAP_SUBS_COLS
   SET APPLY_QUAL    = 'NewApplyQual'
 WHERE
       APPLY_QUAL    = 'ExistApplyQual' AND
       SET_NAME      = 'Name'           AND
       WHOS_ON_FIRST = 'Val';
```

   この *NewApplyQual* は新しいアプライ修飾子であり、*ExistApplyQual* は既存のアプライ修飾子であり、*Name* はサブスクリプション・セットの名前であり、*Val* は F または S のいずれかです。
3. このサブスクリプション・セットが事前または事後に実行される SQL ステートメントまたはプロシージャー呼び出しを使用する場合は、アプライ・コントロール・サーバーで次の SQL ステートメントを実行して、IBMSNAP_SUBS_STMTS 表の中のサブスクリプション・セットのアプライ修飾子を変更します。

```
UPDATE ASN.IBMSNAP_SUBS_STMTS
   SET APPLY_QUAL    = 'NewApplyQual'
 WHERE
       APPLY_QUAL    = 'ExistApplyQual' AND
       SET_NAME      = 'Name'           AND
       WHOS_ON_FIRST = 'Val';
```

   この *NewApplyQual* は新しいアプライ修飾子であり、*ExistApplyQual* は既存のアプライ修飾子であり、*Name* はサブスクリプション・セットの名前であり、*Val* は F または S のいずれかです。
4. キャプチャー・コントロール・サーバーから次の SQL ステートメントを実行して、 IBMSNAP_PRUNE_SET 表および IBMSNAP_PRUNCNTL 表の中のサブスクリプション・セットのアプライ修飾子を変更します。

```
UPDATE Schema.IBMSNAP_PRUNE_SET
   SET APPLY_QUAL    = 'NewApplyQual'
 WHERE
       APPLY_QUAL    = 'ExistApplyQual' AND
       SET_NAME      = 'Name'           AND
       TARGET_SERVER = 'Target_Server';

UPDATE Schema.IBMSNAP_PRUNCNTL
   SET APPLY_QUAL    = 'NewApplyQual'
 WHERE
       APPLY_QUAL    = 'ExistApplyQual' AND
       SET_NAME      = 'Name'           AND
       TARGET_SERVER = 'Target_Server';
```

この *Schema* はキャプチャー・スキーマの名前であり、*NewApplyQual* は新しいアプライ修飾子であり、*ExistApplyQual* は既存のアプライ修飾子であり、*Name* はサブスクリプション・セットの名前であり、 *Target_Server* はターゲット表のデータベース・ロケーションです。

5. 移動する残りのサブスクリプション・セットのそれぞれについて、ステップ 2 から 4 を繰り返します。
6. Linux、UNIX、Windows z/OS でアプライ・プログラムを実行していて、**opt4one** を y に設定している場合には、アプライ・プログラムをいったん停止してから再始動します。
7. レプリケーション・センターを使用して、サブスクリプション・セットを再び活動化します。

サブスクリプション・セットの非活動化

サブスクリプション・セットは、除去することなく、非活動化できます。サブスクリプション・セットを非活動化すると、アプライ・プログラムは、現在の処理サイクルを完了させてから、サブスクリプション・セットの処理を停止します。

始める前に

これらの SQL ステートメントを実行する前に、SQL レプリケーション・コントロール表の構造、およびシステムで定義されているサブスクリプション・セットについてよく調べておいてください。

このタスクについて

サブスクリプション・セットを非活動化させる時間の長さによっては、これらの非活動化したサブスクリプション・セットに関して特別の保守が必要になります。

短期間 一時的に非活動化したサブスクリプション・セットについては、特別な処理要件はありません。サブスクリプション・セットの属性を変更するとき、またはターゲット表の障害を修復するときには、サブスクリプション・セットを一時的に非活動化する必要があります。

サブスクリプション・セットを非活動化、変更、および再活動化するには、レプリケーション・センターを使用します。

長期間 現在は必要ないが、将来使用する可能性のあるサブスクリプション・セットは非活動化しておくことができます。しかし、このサブスクリプション・セットを長期間にわたって非活動化しておく必要がある場合には、累積した変

更データによってキャプチャー・プログラムおよびアプライ・プログラムのパフォーマンスが影響を受ける可能性があるため、追加の処理が必要になります。

キャプチャー・プログラムは、整理プロセス時にはアプライ・プログラムからの情報を使用します。長い時間にわたりアプライ・プログラムが非アクティブになるか、サブスクリプション・セットが非アクティブ化されると、整理情報が不整合になり、非アクティブ化されたサブスクリプション・セットに関連してアクティブな登録が残っている場合には、作業単位 (UOW) 表および変更データ (CD) 表で迅速かつ効果的な整理を行うことができなくなります。この不整合な情報により、残りのアクティブなアプライ・プログラムのパフォーマンスが大幅に低下したり、整理処理が高価な CPU を不必要に消費してしまう可能性があります。UOW 表および CD 表は、最終的にはキャプチャー・プログラムの保持制限 (デフォルト値は 7 日) に基づいて整理されます。しかし、レプリケーション環境の規模によっては、この期間に大量のデータが累積される可能性があります。

このような整理の問題を防止するために、長期間にわたり非活動化しておく必要のあるサブスクリプション・セットについては、SQL を使用して整理情報をリセットすることができます。

登録済みオブジェクトに関連付けられたすべてのサブスクリプション・セットを非活動化したときには、登録済みオブジェクトも非活動化して、キャプチャー・プログラムが不必要にデータをキャプチャーしないようにする必要があります。

手順

1. レプリケーション・センターから、セットを非活動化します。 「**サブスクリプション・セット**」フォルダーをクリックし、内容ペインでアクティブなサブスクリプション・セットを右クリックし、「**非アクティブ化**」を選択します。
2. キャプチャー・コントロール・サーバーから次の SQL ステートメントを実行して、 IBMSNAP_PRUNE_SET 表および IBMSNAP_PRUNCNTL 表で、非活動化したサブスクリプション・セットの整理情報をリセットします。

```
UPDATE Schema.IBMSNAP_PRUNE_SET
   SET SYNCHPOINT    = x'00000000000000000000' AND
       SYNCHTIME     = NULL
 WHERE
       APPLY_QUAL    = 'ApplyQual' AND
       SET_NAME      = 'Name'      AND
       TARGET_SERVER = 'Target_Server';

UPDATE Schema.IBMSNAP_PRUNCNTL
   SET SYNCHPOINT    = NULL  AND
       SYNCHTIME     = NULL
 WHERE
       APPLY_QUAL    = 'ApplyQual' AND
       SET_NAME      = 'Name'      AND
       TARGET_SERVER = 'Target_Server';
```

この *Schema* はキャプチャー・スキーマの名前であり、*ApplyQual* はアプライ修飾子であり、*Name* はサブスクリプション・セットの名前であり、*Target_Server* はターゲット表のデータベース・ロケーションです。

サブスクリプション・セットの除去

特定のサブスクリプション・セットでデータのレプリケーションが必要なくなったときには、サブスクリプション・セットを除去できます。しかし、除去するサブスクリプション・セットに対してアプライ・プログラムの処理が行われている場合には、アプライ・プログラムのジョブはアベンドし、このジョブの中の他のサブスクリプション・セットは、ユーザーがジョブを再始動するまで処理されません。

手順

サブスクリプション・セットを除去するには、次のようにします。

1. サブスクリプション・セットに対するアプライ・プログラムの現在の処理がすべて完了していることを確認するために、レプリケーション・センターから、サブスクリプション・セットを除去する前にサブスクリプション・セットを非活動化してください。 「**サブスクリプション・セット**」フォルダーをクリックし、内容ペインでアクティブなサブスクリプション・セットを右クリックし、「**非アクティブ化**」を選択します。
2. 次の方法のいずれかを使用して、非活動化したサブスクリプション・セットを除去します。

方法	説明

ASNCLP コマンド行プログラム	DROP SUBSCRIPTION SET コマンドを使用します。
	
	以下のコマンドは、環境を設定し、名前が SET00 でアプライ修飾子が AQ00 のサブスクリプション・セットをドロップします。
	
	`SET SERVER CAPTURE TO DB SAMPLE;`
	`SET SERVER CONTROL TO DB TARGET;`
	`SET OUTPUT CAPTURE SCRIPT "drpcapsubset.sql"`
	`CONTROLSCRIPT "drpappsubset.sql";`
	`SET LOG "drpsubset.err";`
	`SET RUN SCRIPT LATER;`
	`DROP SUBSCRIPTION SET SETNAME SET00 APPLYQUAL AQ00;`

レプリケーション・センター	「サブスクリプション・セットの削除」ウィンドウを使用します。このウィンドウをオープンするには、「**サブスクリプション・セット**」フォルダーをクリックし、内容ペインでアクティブなサブスクリプション・セットを右クリックし、「**削除**」を選択します。
System i RMVDPRSUB システム・コマンド	サブスクリプション・セットを除去するには、 DPR サブスクリプション・セットの除去 (RMVDPRSUB) コマンドを使用します。

キャプチャー・プログラムは、登録済みオブジェクトのすべてのサブスクリプション・セットが除去されても、変更データ (CD) 表でのデータのキャプチャーおよび行の書き込みを続けます。このキャプチャー・プログラムによる処理の続行を防止するために、サブスクリプション・セットを除去した後で、登録済みオブジェクトを非活動化または除去してください。

データベース・アプリケーション・イベントとレプリケーション・イベントの調整

IBMSNAP_SIGNAL 表に手動で行を挿入すれば、データベース・イベントとレプリケーション・イベントを調整できます。シグナルと呼ばれるこれらの行の指示に基づいて、実行中のキャプチャー・プログラムは具体的なアクションを実行します。

USER タイプ・シグナルを使用したイベント END_SYNCHPOINT の設定

SIGNAL_TYPE 列の値を USER に設定することにより、DB2 リカバリー・ログ内の正確な時点を確立し、データベース・アプリケーション・イベントとレプリケーション・イベントを調整できます。

このタスクについて

例えば、オンライン・トランザクション処理 (OLTP) データを、別個に保守されるデータウェアハウスに複製する場合、ウェアハウスのデータは、随時の照会のために比較的安定したものにしておく必要があります。このため、ウェアハウスのデータでは、 OLTP アプリケーションの労働日の特定の時点までに発生した変更のみを更新することにします。この場合のデータベース・アプリケーション・イベントは、労働日の論理的な終了時です。レプリケーション・イベントは、特定の日のビジネスがクローズしてから、次の日のビジネスがクローズするまでの変更の適用ということになります。サブスクリプション・セットはイベント処理に対してのみ構成されていると仮定します。

手順

USER タイプのシグナルを作成するには、次のようにします。

1. IBMSNAP_SIGNAL 表に次の行を挿入して、キャプチャー USER タイプのシグナルを作成します。

```
INSERT INTO Schema.IBMSNAP_SIGNAL
            (signal_type,
             signal_subtype,
             signal_state)
      VALUES('USER',
             'USER APPLY EVENT SIGNAL',
             'P');
```

   データベース・アプリケーション・イベントが発生したとき (この場合はアプリケーションの労働日の終わり) に、この SQL INSERT ステートメントを実行します。

   キャプチャー・プログラムは、データベース・リカバリー・ログでこのレコードを検出した後で、このシグナル表のログ・レコードを処理します。キャプチャー・プログラムが反応するのは、この挿入に対応するコミット・レコードが検出された場合、つまりこのイベントがコミットされたことが検証された場合だけです。

USER タイプのシグナルがコミットされると、キャプチャー・プログラムは、処理される挿入ログ・レコードに対応する、次の IBMSNAP_SIGNAL 列値を更新します。

- SIGNAL_STATE = 'R' (キャプチャー・プログラムが受け取り済み)
- SIGNAL_LSN = このシグナル行の挿入を含む DB2 の作業単位のコミット・ログ・レコードからのログ・シーケンス番号

2. 挿入されたシグナル行の SIGNAL_LSN 列にある値を使用して、IBMSNAP_SUBS_EVENT コントロール表に END_SYNCHPOINT 値を挿入します。この新しい値は、新しい労働日のすべてのデータがキャプチャー・プログラムによってキャプチャーされたこと、そしてアプライ・プログラムが SIGNAL_LSN 列の値までのデータのみをフェッチしてアプライする必要があることをアプライ・プログラムに知らせます。

   IBMSNAP_SIGNAL 表に対する更新トリガーを作成すれば、IBMSNAP_SUBS_EVENT への挿入を自動化できます。

```
CREATE TRIGGER EVENT_TRIG
    NO CASCADE AFTER UPDATE ON Schema.IBMSNAP_SIGNAL
       REFERENCING NEW AS N
          FOR EACH ROW MODE DB2SQL
              WHEN (N.SIGNAL_SUBTYPE = 'USER APPLY EVENT SIGNAL')
                    INSERT INTO ASN.IBMSNAP_SUBS_EVENT VALUES
                         ('WH_APPLY_EVENT',
                          (CURRENT TIMESTAMP + 2 MINUTES),
                           N.SIGNAL_LSN,
                           null);
```

   このトリガーは、キャプチャー・プログラムによって IBMSNAP_SIGNAL 表が更新されるたびに起動されます。 SIGNAL_SUBTYPE 列が USER APPLY EVENT SIGNAL に更新されると、トリガーは IBMSNAP_SUBS_EVENT 表に行を挿入します。この行は、2 分間が経過した後、最後の労働日 (キャプチャー・プログラムにより算出された SIGNAL_LSN 値の以前にコミットされている) 以降のフェッチおよび適用を処理する必要があることをアプライ・プログラムに指示します。

キャプチャーの CMD STOP シグナルを使用するべき状況

SIGNAL_TYPE 列の値を CMD に設定し、SIGNAL_SUBTYPE 列の値を STOP に設定することにより、キャプチャー・プログラムの処理を、DB2 リカバリー・ログの正確な時点で停止できます。

キャプチャーの CMD STOP シグナルは、以下のような目的に使用できます。

- 以前のログ・レコードを読み取れない状態にした、ソース表の変更に対してキャプチャー・プログラムを調整するため。このような状態は、ユーザーが表をドロップしてから再作成した場合、またはユーザーが KEEPDICTIONARY オプションを YES に設定しないで表を再編成したときに発生する可能性があります。
- 複製された分散データベース・システム間で共通のリカバリー点を調整するため。

ソース表の変更とキャプチャー・プログラムの調整

キャプチャー CMD タイプ STOP サブタイプのシグナルを使用して、キャプチャー・プログラムをシャットダウンして、ソース表の変更を調整できます。

手順

ソース表の変更を調整するには次のようにします。

1. 次の SQL ステートメントを使用して、IBMSNAP_SIGNAL 表に行を挿入してキャプチャー CMD タイプ STOP サブタイプのシグナルを作成します。

```
INSERT INTO Schema.IBMSNAP_SIGNAL
            (signal_type,
             signal_subtype,
             signal_state)
      VALUES('CMD',
             'STOP',
             'P');
```

   この行は、データベース・アプリケーション・イベントが発生したときに、ソース表のアクティビティーが静止した後、そしてログ・レコードの問題の原因となったアクティビティーが変更される前に、挿入する必要があります。

   キャプチャー・プログラムは、データベース・リカバリー・ログでこのレコードを検出した後で、このシグナル表のログ・レコードを処理します。キャプチャー・プログラムが反応するのは、この挿入に対応するコミット・レコードが検出された場合、つまりこのイベントがコミットされたことが検証された場合だけです。

   キャプチャー・プログラムは、この挿入された IBMSNAP_SIGNAL 行を含む DB2 作業単位のコミット・ログ・レコードより以前の、ログ上のトランザクションからのすべてのキャプチャーされたデータをコミットした後で、すべてのキャプチャー・スレッドを順番にシャットダウンします。キャプチャー・プログラムは終了する前に、処理される挿入ログ・レコードに対応する IBMSNAP_SIGNAL 表の行の中の次の値も更新します。

   - SIGNAL_STATE = 'R' (キャプチャー・プログラムが受け取り済み)
   - SIGNAL_LSN = このシグナル行の挿入を含む DB2 の作業単位のコミット・ログ・レコードからのログ・シーケンス番号

   変更のあるソース表のすべてのログ・レコードは、終了時にキャプチャー・プログラムにより処理されます。

2. ユーザーのシナリオに従って、ソース表をドロップして再作成するか、KEEPDICTIONARY オプションを YES に設定せずにソース表を再編成および圧縮します。
3. 複製された列をドロップまたは変更した場合は、このソース表に対して作成された、対応する登録およびサブスクリプション・セットをここで変更する必要があります。必要であれば、影響を受けたサブスクリプション・セットが、現在停止中のキャプチャー・プログラムに追いつくのを待つことにより、アプライ・プログラムとの間でこのような変更をさらに調整することができます。サブスクリプション・セットは、IBMSNAP_SUBS_SET 表の中の SYNCHPOINT 列の値が *Schema*.IBMSNAP_RESTART 表の中の MAX_COMMITSEQ 列の値と等しくなると、キャプチャー・プログラムと同期します。

分散リカバリー点の設定

キャプチャー CMD タイプ STOP サブタイプのシグナルを使用して、ソース・データベースおよびターゲット・データベースを同じリカバリー点に設定し、共通整合点でデータベースをリカバリーできます。

始める前に

この手順を使用する前に、ターゲット・データベースにアプライ・コントロール表が作成されていることを確認してください。

また、IBMSNAP_SIGNAL 表に行を挿入する前に、ソース・データベースに対するすべてのアクティビティーが静止していることを確認します。しかし、IBMSNAP_SIGNAL 表に行を挿入するまでは、データベース表のバックアップまたはイメージ・コピーを作成しないでください。

サブスクリプション・セットがイベント処理用の典型的な構成になっていない場合には、サブスクリプション・セットを一時的にイベント・ベースのタイミング用に設定する必要があります。次の SQL ステートメントを使用して、サブスクリプション・イベント IBMSNAP_SUBS_EVENT 表に行を挿入します。

```
INSERT INTO ASN.IBMSNAP_SUBS_EVENT
       VALUES('RECOVERY_EVENT',
              CURRENT TIMESTAMP + 2 MINUTES,
              SIGNAL_LSN_value,
              NULL);
```

この *SIGNAL_LSN_value* は、キャプチャー・プログラムによって設定され、IBMSNAP_SIGNAL 表の中に保管されるログ・シーケンス番号です。

手順

分散リカバリー点を設定するには、次のようにします。

1. 次の SQL ステートメントを使用して、IBMSNAP_SIGNAL 表に行を挿入してキャプチャー CMD タイプ STOP サブタイプのシグナルを作成します。

   ```
   INSERT INTO Schema.IBMSNAP_SIGNAL
              (signal_type,
               signal_subtype,
               signal_state)
         VALUES('CMD',
                'STOP',
                'P');
   ```

   キャプチャー・プログラムは、データベース・リカバリー・ログでこのレコードを検出した後で、このシグナル表のログ・レコードを処理します。キャプチャー・プログラムが反応するのは、この挿入に対応するコミット・レコードが検出された場合、つまりこのイベントがコミットされたことが検証された場合だけです。

   キャプチャー・プログラムは、この挿入された IBMSNAP_SIGNAL 行を含む DB2 作業単位のコミット・ログ・レコードより以前の、ログ上のトランザクションからのすべてのキャプチャーされたデータをコミットした後で、すべてのキャプチャー・スレッドを順番にシャットダウンします。キャプチャー・プログラムは終了する前に、処理される挿入ログ・レコードに対応する IBMSNAP_SIGNAL 表の行の中の次の値も更新します。

- SIGNAL_STATE = 'R' (キャプチャー・プログラムが受け取り済み)
- SIGNAL_LSN = このシグナル行の挿入を含む DB2 の作業単位のコミット・ログ・レコードからのログ・シーケンス番号

ソース・データベースのすべてのログ・レコードは、終了時にキャプチャー・プログラムにより処理されます。

2. ソース・データベースのバックアップまたはイメージ・コピー・ユーティリティーを実行します。
3. 挿入した IBMSNAP_SIGNAL 表の行の SIGNAL_LSN 列の値を、IBMSNAP_SUBS_EVENT 表の中の END_SYNCHPOINT 値として使用します。この値は、バックアップ時点以前にコミットされたすべてのデータがキャプチャー・プログラムによってキャプチャーされたこと、そしてアプライ・プログラムが SIGNAL_LSN 列の値までのデータのみをフェッチしてアプライする必要があることをアプライ・プログラムに知らせます。 サブスクリプション・セットは、SIGNAL_LSN 値までのすべてのデータを処理します。
4. ターゲット・データベースのバックアップまたはイメージ・コピー・ユーティリティーを実行します。これでソース・データベースとターゲット・データベースは同じリカバリー点を持つことになるため、共通整合点で両方のデータベースをリカバリーできます。

アプライ・イベントが設定され、ソース・データベースのバックアップまたはイメージ・コピー・ユーティリティーのアクティビティーが完了したらすぐに、ソース・データベースのすべてのアクティビティーを再開できます。キャプチャー・プログラムも開始することができます。 ターゲット・データベースのバックアップまたはイメージ・コピー・ユーティリティーのアクティビティーが完了したら、サブスクリプション・セットのスケジューリング・オプションを元の設定 (時間ベース、イベント・ベースまたはその両方) に戻すことができます。

System i STOP シグナルを送信して、1 つのジャーナル・ジョブまたはすべてのジャーナル・ジョブを停止できます。単一のジャーナル・ジョブを停止するには、そのジャーナルに指定されたシグナル表 (IBMSNAP_SIGNAL_*xxxx_yyyy* 表。*xxxx* はジャーナル・ライブラリーで、*yyyy* はジャーナル名) にシグナルを挿入します。すべてのジャーナル・ジョブを停止するには、シグナルを *schema*.IBMSNAP_SIGNAL 表に挿入します。リモート・ジャーナル構成中の単一のジャーナル・ジョブを停止するには、ソース・サーバー上のジャーナル・シグナル表にシグナルを挿入します。リモート・ジャーナル構成でジャーナル・シグナル表を作成する方法に関する記述を確認してください。

アプライ・プログラム外部の CAPSTART ハンドシェーク・シグナルの実行

アプライ・プログラムがサブスクリプション・セットを使用して CD 表の変更をフェッチおよびアプライできるようにするためには、そのサブスクリプション・セット内の各サブスクリプション・セット・メンバーのキャプチャー・プログラムとアプライ・プログラムの間でハンドシェーク (同期化された通信) が必要になります。

このタスクについて

アプライ・プログラムは、CMD タイプ CAPSTART サブタイプのシグナルを IBMSNAP_SIGNAL 表に挿入することにより、ハンドシェークを開始します。アプライ・プログラムは、コンプリートとして定義されているターゲット表でサブスクリプション・メンバーのフル・リフレッシュを実行する前に、このシグナルを挿入します。

手順

アプライ・プログラムの外部で CAPSTART ハンドシェーク・シグナルを実行するには、以下のようにします。

次の SQL ステートメントを使用して、IBMSNAP_SIGNAL 表に行を挿入してキャプチャー CMD タイプ CAPSTART サブタイプのシグナルを作成します。

```
INSERT INTO Schema.IBMSNAP_SIGNAL
            (signal_type,
             signal_subtype,
             signal_input_in,
             signal_state)
       VALUES('CMD',
              'CAPSTART',
               mapid,
              'P');
```

この *mapid* は *Schema*.IBMSNAP_PRUNCNTL 表の MAP_ID 列の値であり、ハンドシェークを必要とするサブスクリプション・セット・メンバーの行に対応します。

注: 必要であれば、サブスクリプション・セット・メンバーのフル・リフレッシュを実行する前に、この SQL INSERT ステートメントを実行してください。

キャプチャー・プログラムは、データベース・リカバリー・ログでこのレコードを検出した後で、このシグナル表のログ・レコードを処理します。キャプチャー・プログラムが反応するのは、この挿入に対応するコミット・レコードが検出された場合、つまりこのイベントがコミットされたことが検証された場合だけです。

キャプチャー・プログラムは、登録済み表の以前の使用に基づいて、関連する登録がすでにメモリー内に入れられているかどうかをチェックします。登録済み表が使用されていない場合、キャプチャー・プログラムは関連する登録情報をメモリー内に読み込み、この登録済み表がアクティブであり、使用中であることを示す値を IBMSNAP_REGISTER 表の中に設定します。

キャプチャー・プログラムは登録済み表が使用中であるかどうかに関係なく、*Schema*.IBMSNAP_PRUNCNTL 表の中の関連する行の SYNCHPOINT 列および SYNCHTIME 列の値を、この挿入されたシグナル行を含む DB2 作業単位のコミット・ログ・レコードからのログ・シーケンス番号、およびこの同じコミット・ログ・レコードからのタイム・スタンプにそれぞれ設定します。

キャプチャー・プログラムは、処理される挿入ログ・レコードに対応する IBMSNAP_SIGNAL 表の行の中の次の値を更新します。

- SIGNAL_STATE = 'C' (キャプチャー・プログラムにより受け取り済みでコンプリート)

- SIGNAL_LSN = このシグナル行の挿入を含む DB2 の作業単位のコミット・ログ・レコードからのログ・シーケンス番号

CAPSTOP シグナルの実行

登録の変更のキャプチャーを手動で停止する場合は、CAPSTOP シグナルを開始できます。このシグナルは、登録を非活動化する場合、または登録を除去する前に使用できます。

手順

CAPSTOP シグナルを実行するには、次のようにします。

1. 次の SQL ステートメントを使用して、IBMSNAP_SIGNAL 表に行を挿入してキャプチャー CMD タイプ CAPSTOP サブタイプのシグナルを作成します。

```
INSERT INTO Schema.IBMSNAP_SIGNAL
            (signal_type,
             signal_subtype,
             signal_input_in,
             signal_state)
       VALUES('CMD',
              'CAPSTOP',
               source_owner.source_table,
              'P');
```

   この *Schema* はキャプチャー・スキーマの名前であり、*source_owner.source_table* はキャプチャーされた変更を必要としなくなった表の完全修飾名です。

   キャプチャー・プログラムは、データベース・リカバリー・ログでこのレコードを検出した後で、このシグナル表のログ・レコードを処理します。キャプチャー・プログラムが反応するのは、この挿入に対応するコミット・レコードが検出された場合、つまりこのイベントがコミットされたことが検証された場合だけです。

   キャプチャー・プログラムは、登録済み表の以前の使用に基づいて、関連する登録がすでにメモリー内に入れられているかどうかをチェックします。登録済み表が現在使用中でない場合には、キャプチャー・プログラムは CAPSTOP シグナルを無視します。

   登録済み表が使用されている場合は、キャプチャー・プログラムはこの登録に関連するメモリーをクリアし、登録を非活動化します (IBMSNAP_REGISTER 表の STATE 列を 'I' に設定することにより)。 その後キャプチャー・プログラムはこの登録済み表について変更のキャプチャーを停止します。

   キャプチャー・プログラムは、処理される挿入ログ・レコードに対応する IBMSNAP_SIGNAL 表の行の中の次の列値を更新します。

   - SIGNAL_STATE = 'C' (キャプチャー・プログラムにより受け取り済みでコンプリート)
   - SIGNAL_LSN = このシグナル行の挿入を含む DB2 の作業単位のコミット・ログ・レコードからのログ・シーケンス番号

2. オプション: オプション: 登録を除去します。

3. System i オプション: CAPSTOP シグナルを送信して、登録の変更のキャプチャーを停止することもできます。そのためには、IBMSNAP_SIGNAL_*xxxx_yyyy* 表 (*xxxx* はジャーナル・ライブラリー、*yyyy* は対象ジャーナルのジャーナル名) にシグナルを挿入します。リモート・ジャーナル構成中の登録のキャプチャー変更を停止するには、ソース・サーバーで CAPSTOP シグナルを挿入します。

夏時間調整 (System i)

System i の場合、キャプチャー・プログラムは、ジャーナルの変更内容を読み取るときに、タイム・スタンプとジャーナルのシーケンス番号を使用します。春と秋にアメリカの夏時間調整に合わせてシステム・クロックを調整する必要がある場合は、このプロセスによって問題が発生する可能性があります。

このタスクについて

System i システムには、夏時間調整を行うための 2 つの方法が用意されています。

V5R3 システムは、タイム・スタンプの抜けや重複を回避するために、クロックを遅らせるか (秋)、クロックを早めます (春)。System i V5R3 でキャプチャー・プログラムを実行する場合に、この新しい方法で時間変更を行うのであれば、以下の手順を使用する必要はありません。

V5R3 より前のリリース
秋には、システムのすべてのアクティビティーを 1 時間停止してから、クロックを 1 時間戻す必要があります。この方法の場合は、以下の手順を使用しなければなりません。

手順

夏時間調整を行うには、以下のようにします。

1. 秋にクロックを 1 時間戻さなければならない場合は、以下の手順を実行します。
   a. ソース表を更新するキャプチャー・プログラムや他のアプリケーションをすべて停止します。
   b. ソース・ジャーナルに新しいジャーナル項目を追加することなく、システム時刻が少なくとも 1 時間進むのを待ちます。
   c. システム時刻を 1 時間戻すように設定します。
   d. キャプチャー・プログラムを再始動します。

   その手順を、以下の例で示します。
   a. 12:00 に、キャプチャー・プログラムおよびすべてのアプリケーションを停止します。
   b. 13:00 まで待ちます。ジャーナル項目のタイム・スタンプの値を 12:00 までのものだけにするためです。
   c. システム時刻を 12:00 に戻します。
   d. 変更を加えます。その変更に対するジャーナル項目のタイム・スタンプは 12:01 になります。

e. キャプチャーを再始動します。キャプチャーは 12:00 から開始するので、12:01 (夏時間) (標準時では 13:01) に行われた変更はキャプチャーされます。

キャプチャー・プログラムは、現在のシステム時刻よりも早いタイム・スタンプで再始動します。新しいシステム時刻が時間変更の直前のシステム時刻を超えるまで、ジャーナル項目は追加されないので、データが失われる可能性はありません。

推奨: 時間変更はアプライ・プログラムには影響しませんが、時間変更を実施するときには、アプライ・プログラムもいったん停止してから再始動してください。

2. 春にクロックを 1 時間進めなければならない場合は、以下の手順を実行します。
   a. キャプチャー・プログラムを停止して、時間変更を行います。 キャプチャー・プログラムは、ソース表に何の変更もなく 1 時間が経過したかのような動作になります。

レプリケーション構成を別のシステムにプロモートするためのオプション

あるシステム (例えばテスト・システム) で登録済みオブジェクトまたはサブスクリプション・セットを定義し、レプリケーション環境を別のシステム (例えば実動システム) にコピーする必要がある場合には、レプリケーション・センターのプロモート関数を使用できます。

プロモート関数は、登録済みオブジェクトまたはサブスクリプション・セットをリバース・エンジニアリングして、適切なデータ定義言語 (DDL) およびデータ操作言語 (DML) 付きのスクリプト・ファイルを作成します。ソースの再登録またはサブスクリプション・セットの再作成を行う必要なしに、レプリケーション定義を別のデータベースにコピーできます。

例えば、プロモート関数を使用して、リモートのターゲット・データベース用のサブスクリプション・セットを定義します。テスト環境でモデルになるターゲット・システムを定義した後、リモート・ターゲット・システム用のサブスクリプション・セットのスクリプトを作成できます (そして使用するアプライ修飾子の変更などを行うことができます)。こうしない場合、中央のコントロール・ポイントからはサポートされません。

重要: プロモート関数は宛先ターゲット・システムとは接続せず、そのシステムのレプリケーション構成パラメーターの検証を行いません。

以下のリストでは、レプリケーション構成を別のシステムにプロモートするための 3 つのオプションについて説明します。

登録済み表のプロモート

この関数は、指定された表の登録情報をプロモートします。この関数は、オプショナルで、基礎表、索引および表スペース定義のプロモートも行います。プロモートする複数の表に対して異なるキャプチャー・スキーマおよび異なるサーバー名を指定できます。また、プロモートされたソース表に関連する変更データ (CD) 表のスキーマ名も変更できます。

複数の登録済み表を同時にプロモートできます。ユーザーが指定した新しいスキーマ名は、プロモートされるすべての表に適用されます。

この関数は、DB2 バージョン 8 以降で登録された表しかプロモートしません。

登録済みビューのプロモート

この関数は、指定されたビューの登録情報をプロモートします。この関数は、オプショナルで、基本ビュー、登録抹消された基礎表 (ビューのベースである)、索引および表スペース定義のプロモートも行います。プロモートする複数のビューに対して異なるキャプチャー・スキーマおよび異なるサーバー名を指定できます。また、プロモートされたソース・ビューに関連する CD ビュー、およびこれらの CD ビューのベースである CD 表のスキーマ名も変更できます。

複数の登録済みビューを同時にプロモートできます。ユーザーが指定した新しいスキーマ名は、プロモートされるすべてのビューに適用されます。

重要: プロモートするビューが登録済みソース表に基づくものである場合は、登録済み表のプロモート関数を使用して、登録済みのソース表を別個にプロモートする必要があります。これらの登録済みソース表が、登録済みビューのプロモート関数によって自動的にプロモートされることはありません。しかし、このビューのベースである、登録抹消された基礎表は、必要であればこの関数によってプロモートされます。

サブスクリプション・セットのプロモート

この関数はサブスクリプション・セットをプロモートします。この関数により、データベース間でサブスクリプション・セット (すべてのサブスクリプション・セット・メンバーを含めて) をコピーできます。

サブスクリプション・セットのプロモート関数は、登録済み表のプロモート関数と一緒に使用する必要があります。

重要: プロモート関数を使用して、サポートされているすべてのオペレーティング・システム上の登録済みオブジェクトおよびサブスクリプション・セットをプロモートできます。プロモート関数は、同種のシステム間でのみレプリケーション定義をコピーします。例えば、1 つの DB2 for z/OS システムから別の DB2 for z/OS システムへコピーします。

プロモート関数を使用して、DB2 以外のリレーショナル・データベースとの間でレプリケーション定義をコピーすることはできません。また、プロモート関数を使用して、System i リモート・ジャーナルを含むレプリケーション定義をコピーすることもできません。

第 13 章 SQL レプリケーション環境の保守

データベース内にあり、SQL レプリケーションに使用される、ソース・システム、コントロール表、およびターゲット表を保守する必要があります。

SQL レプリケーションはデータベース・システムと共同で処理を行うため、既存のデータベース・アクティビティーの変更は最小限ですみます。しかし、システム全体の円滑な実行を保証し、潜在的な問題を回避するためには、レプリケーション環境の処理要件を判別し、これらの要件がデータベース・システムに影響を与える可能性を判別する必要があります。

以下のトピックでは、ソース・システム、コントロール表、およびターゲット表の保守要件について説明します。

ソース・システムの保守

レプリケーション・ソース・システムは、変更キャプチャー・メカニズム、複製するソース表 (System i で使用されるリモート・ジャーナルを含む)、キャプチャー・プログラムから使用されるログ・データ、および DB2 以外のリレーショナル・データベース・ソースで使用されるキャプチャー・トリガーで構成されます。

これらのトピックでは、ソース表とログ・ファイルを正しく保守する方法、そしてこれらの表およびファイルが常に SQL レプリケーションからアクセス可能であるようにする方法を説明します。

ソース表とソース・ビューへのアクセス

キャプチャー・プログラムおよびアプライ・プログラムが常に処理を進められるように、 SQL レプリケーションでのソース表の可用性を考慮する必要があります。

レプリケーション・ソース・オブジェクトは、システム上の他のデータベース表およびビューと同じ保守を必要とする、データベース表およびビューです。これらのオブジェクトに関して、既存のユーティリティーおよび保守ルーチンを引き続き実行してください。

SQL レプリケーションは、ほとんどのレプリケーション処理においては、ソース表に直接アクセスする必要はありません。しかし、SQL レプリケーションは、次の 2 つのアクションのいずれかが発生したときには、ソース表または表スペースにアクセスする必要があります。

- アプライ・プログラムがフル・リフレッシュを実行したとき。
- ログ・マネージャーが圧縮されたログ・レコードを読み取ろうとしたとき (z/OS のみ)。

フル・リフレッシュ時にアプライ・プログラムが阻害されないように、ソース表に対して読み取りアクセスが可能であることを確認してください。また、z/OS の場合は、ソース表が圧縮されている場合に、DB2 が圧縮されたログ・レコードの表スペースに対してラッチを入手できるように、ユーティリティーがオンライン・モード

で実行されることを確認してください。ユーティリティーおよび保守ルーチンが、データベース (または z/OS の場合は圧縮された表スペース) をオフラインにする必要のある排他モードで実行される場合は、レプリケーションでソース・オブジェクトを使用することはできません。

ソース・ログとジャーナル・レシーバー

DB2 リカバリー・ログには、DB2 リカバリー機能の提供と、実行中のキャプチャー・プログラムへの情報の提供という 2 つの目的があります。

DB2 リカバリー、および SQL レプリケーションの両方についてログ・データを保存する必要があります。また、このデータを削除する前に、キャプチャー・プログラムおよび DB2 が、ログまたはジャーナル・レシーバーのセットの処理を完全に終了していることを確実に確認する必要があります。

注: SQL レプリケーションは、DB2 以外のリレーショナル・データベースからのログ・データは使用しません。

ログ・データの保存 (Linux、UNIX、Windows)

ログ・データはログ・バッファー、アクティブ・ログ、またはアーカイブ・ログに入っています。キャプチャー・プログラムはウォーム・スタートのたびに、プログラムの停止後に作成されたすべての DB2 ログと、処理が完全に終了していないすべての DB2 ログを要求します。

始める前に

注: キャプチャー・プログラムがアーカイブ・ログからデータをリトリーブできるように、ユーザー出口アーカイブを使用するようにデータベースを構成する必要があります。

このタスクについて

DB2 の実行時は常にキャプチャー・プログラムを実行するようにしておけば、キャプチャー・プログラムは一般的に DB2 のリカバリー・ログに合わせた最新の状態になります。DB2 がアクティブになっている間は常にキャプチャー・プログラムを実行するか、ログ・レコードを 1 週間以上保存するのであれば、既存のログ保存手順を使用し続けることができます。しかし、次の場合には、 SQL レプリケーションに合わせて、ログ保存手順を変更する必要があります。

- DB2 がバックアップを完了したら即時にログ・レコードを削除する場合、そしてこれらのログ・レコードが順方向リカバリーに必要とされない場合。
- ストレージの制約があるため、アーカイブしたリカバリー・ログを頻繁に削除する必要がある場合。

手順

キャプチャー・プログラムで使用するために保存する必要のあるログ・レコードと、削除できるログ・レコードを判別するには、次のようにします。

1. 次の SQL ステートメントを実行し、IBMSNAP_RESTART 表から MIN_INFLIGHTSEQ 値を入手します。

パーティション・データベースの場合: マルチパーティション環境では、各パーティションはログ・ファイルの独自のセットを保守するため、この手順は各パーティションに拡張される必要があります。 IBMSNAP_PARTITIONINFO 表の SEQUENCE 列を使用して、各パーティションごとにこの情報を判別してください。

```
SELECT MIN_INFLIGHTSEQ
FROM ASN.IBMSNAP_RESTART
WITH UR;
```

MIN_INFLIGHTSEQ 値が表示されます。MIN_INFLIGHTSEQ 値は、CHAR(10) の FOR BIT DATA 列であり、16 進の 20 文字のように見えます。以下に例を示します。

```
00000000123456123456
```

MIN_INFLIGHTSEQ 値の最後の 12 文字に注目してください。この例では次のようになっています。

```
123456123456
```

重要: キャプチャー・プログラムは、データをコミットするたびに、**commit_interval** パラメーターの値に基づいて IBMSNAP_RESTART を更新します。この手順で使用する SELECT ステートメントでは非コミット読み取り (UR) が指定されているので、MIN_INFLIGHTSEQ の非コミット値を受け取る可能性があります。最も正確な値を得るようにするには、SELECT ステートメントを実行し、コミット・インターバルが経過するのを待ってから、もう一度 SELECT を実行します。この手順の残りの部分では、小さな値の MIN_INFLIGHTSEQ を使用してください。

2. コマンド行から db2 get db cfg コマンドを入力し、アクティブ・ログ・ファイルのパスを入手します。以下に例を示します。

   ```
   db2 get db cfg for yourdbname
   ```

   この *yourdbname* はデータベース名です。 画面に表示された出力からアクティブ・ログ・ファイルのパスを確認します。以下に例を示します。

   ```
   Path to log files   =C:¥DB2¥NODE0000¥SQL00001¥SQLOGDIR¥
   ```

3. DB2 コマンド行から db2flsn コマンドを入力し、MIN_INFLIGHTSEQ 値の最後の 12 文字を入力します。 以下に例を示します。

   ```
   C:¥DB2¥NODE0000¥SQL00001¥>db2flsn 123456123456
   ```

   db2flsn コマンドを実行するには、SQLOGCTL.LFH.1 ファイルかそのミラー・コピーである SQLOGCTL.LFH.2 にアクセスできる必要があります。これらのファイルは両方ともデータベース・ディレクトリーにあります。 システムは、ログ・シーケンス番号により識別されるログ・レコードを含むファイルの名前をリトリーブして表示します。以下に例を示します。

   ```
   Given LSN is contained in the log file S000123.LOG
   ```

ジャーナル・レシーバーへのアクセス (System i)

キャプチャー・プログラムから必要とされるすべてのジャーナル・レシーバーを保存することが重要です。

RESTART(*YES) パラメーターを指定してキャプチャー・プログラムを再始動すると、キャプチャー・プログラムは以前に終了した場所から処理を続行し、 1 つまたは複数のソース表により使用されるすべてのジャーナル・レシーバーを必要とします。

キャプチャー・プログラムが必要なジャーナル・レシーバーのすべてにアクセスできるように、DB2 DataPropagator for System i のインストール時に自動的に登録された、ジャーナル・レシーバー削除出口プログラムを使用してください。この出口プログラムは、ユーザーまたはユーザーのアプリケーション・プログラムの 1 つがジャーナル・レシーバーの削除を試みるたびに呼び出されます。ようとこの出口プログラムは、ジャーナル・レシーバーを削除できるかどうかを決定します。

推奨: CHGJRN または CRTJRN コマンドで DLTRCV(*YES) および MNGRCV(*SYSTEM) を指定し、ジャーナル・レシーバー削除出口プログラムを使用し、ジャーナル管理をシステムに任せるようにします。

ジャーナル・レシーバーが 1 つまたは複数のソース表で使用されている場合、ジャーナル・レシーバー削除出口プログラムは、キャプチャー・プログラムによってまだ処理されていない項目が、削除対象のレシーバーに含まれていないことを確認します。キャプチャー・プログラムがそのレシーバーの項目をさらに処理する必要がある場合には、出口プログラムはレシーバーの削除を承認しません。

コンプレッション・ディクショナリーの管理に関する考慮事項 (z/OS)

DB2 のコンプレッション・ディクショナリー・ユーティリティーを使用する場合は、キャプチャー・プログラムとの間でこれらのユーティリティーの使用を調整する必要があります。

DB2 コンプレッション・ディクショナリーの更新 (z/OS)

キャプチャー・プログラムがログ・レコードを要求した場合、DB2 は圧縮された表スペースに保管されている表のログ・レコードを圧縮解除する必要があります。DB2 は現行のコンプレッション・ディクショナリーを使用して圧縮を解除します。コンプレッション・ディクショナリーが使用できない場合もあります。以下の場合、キャプチャー・プログラムはそれぞれ異なるアクションを実行します。

コンプレッション・ディクショナリーが一時的に使用できない場合
DB2 はキャプチャー・プログラムにエラーを戻します。キャプチャー・プログラムは何度か処理の続行を試みます。ディクショナリーが引き続き使用できない状態である場合は、キャプチャー・プログラムは ASN0011E メッセージを発行して終了します。

コンプレッション・ディクショナリーが永続的に使用できない場合
KEEPDICTIONARY=YES を指定しないで REORG ユーティリティーを使用すると、コンプレッション・ディクショナリーが失われる場合があります。この場合、キャプチャー・プログラムは、登録の STOP_ON_ERROR オプションで指定されたエラー・アクションに従います。 STOP_ON_ERROR=N (no) の場合、キャプチャー・プログラムは登録を非活動化します。STOP_ON_ERROR=Y (yes) の場合、キャプチャー・プログラムは ASN0011E メッセージを発行して終了します。

APAR PK19539 (DB2 for z/OS バージョン 8) を適用すると、KEEPDICTIONARY=YES を指定しないで REORG ユーティリティーを使用した場合でも、DB2 はメモリーにコンプレッション・ディクショナリーのバックアップを 1 つ保持します。したがって、以下の場合を除いて KEEPDICTIONARY=YES を指定する必要はありません。

- DB2 を再始動する。
- キャプチャー・プログラムが表のすべての古いログ・レコードを読み取る前に、その表スペースに対して REORG ユーティリティーを 2 回使用する。

DB2 for z/OS バージョン 7 では、こうした状況を防止するために、表のコンプレッション・ディクショナリーに影響を与えるアクティビティーを実行する前に、キャプチャー・プログラムが表のすべてのログ・レコードを処理するようにしてください。以下のアクティビティーの中には、コンプレッション・ディクショナリーに影響を与えるものがあります。

- 圧縮設定を変更する表スペースの変更
- データ共有環境から非データ共有環境へのコピーを含めて、DSN1COPY を使用したサブシステム間での圧縮した表スペースのコピー
- 表スペースに対する REORG ユーティリティーの実行

DB2 コンプレッション・ディクショナリーのラッチ (z/OS)

コンプレッション・ディクショナリーの可用性についても考慮する必要があります。キャプチャー・プログラムが圧縮されたログ・レコードを読み取るときには、 DB2 はディクショナリーにアクセスするために、ソースの圧縮された表スペースのラッチを取ります。 DB2 のログ読み取りインターフェースがこのラッチを必要としたときに、ソース・システム上の圧縮された表スペースが STOPPED 状態にある場合は、キャプチャー・プログラムは停止します。その逆に、ソースの表スペースへの完全アクセスを必要とする、または表スペースが STOPPED 状態であることを必要とするユーティリティーは、キャプチャー・プログラムがディクショナリーの読み取り中にラッチを保持しているために締め出されることがあります。

ラッチが使用できないために一時的なロックアウトが発生しないように、ソースの圧縮された表スペースを DB2 (またはベンダー) のユーティリティーで排他的に使用する必要があるときには、キャプチャー・プログラムを中断してください。

コントロール表の保守

SQL レプリケーションは、ソース定義、サブスクリプション・セット定義、およびその他のレプリケーション固有のコントロール情報を保管するためにコントロール表を使用します。コントロール表のサイズは静的なものもあれば、データベースおよびレプリケーションのサイズの要件に従って動的に拡大 (または縮小) するものもあります。

以下の表のサイズは、通常処理時に頻繁に変更されます。

- System i IBMSNAP_APPLY_JOB
- IBMSNAP_APPLYTRACE

- IBMSNAP_APPLYTRAIL
- IBMSNAP_CAPMON
- IBMSNAP_CAPTRACE
- CD 表
- CCD 表
- IBMSNAP_ALERTS
- IBMSNAP_MONTRACE
- IBMSNAP_MONTRAIL
- IBMSNAP_SIGNAL
- BMSNAP_SUBS_EVENT
- IBMSNAP_UOW

これらの動的なコントロール表のサイズおよび拡大により、システムのパフォーマンスが影響を受ける可能性があります。

SQL レプリケーションのための RUNSTATS ユーティリティー (Linux、UNIX、Windows、z/OS)

RUNSTATS ユーティリティーは、表および関連する索引の物理的特性に関する統計を更新します。

以前に SQL レプリケーションで使用していたのと同じ頻度で、既存の表に対して引き続き RUNSTATS ユーティリティーを実行する必要があります。しかし、表に含まれるデータの量が大量である場合には、変更データ (CD)、IBMSNAP_UOW、およびその他の動的なコントロール表に対して RUNSTATS ユーティリティーを一度だけ実行してください。RUNSTATS が動的な表に関して有用な情報を報告するのは、これらの表が実動レベルで最大のサイズにある場合です。オプティマイザーは必要な統計を獲得して、データへのアクセスの最良のストラテジーを判断します。

パッケージとプランの再バインド (z/OS、Linux、UNIX、Windows)

分離レベルを UR (非コミット読み取り) に設定してパッケージとプランを再バインドすると、システム・パフォーマンスを最適化できます。

SQL レプリケーションのパッケージおよびプランの多くは、分離 UR を使用してバインドされます。パッケージおよびプランの自動再バインドに使用される内部の保守プログラムは、キャプチャー・プログラムおよびアプライ・プログラムがカーソル固定などの標準オプションを使用してレプリケーション・パッケージを再バインドした場合に、プログラム間で競合の問題を発生させる可能性があります。パッケージおよびプランの再バインドが必要な場合は注意してください。SQL レプリケーション・パッケージは、最適なシステム・パフォーマンスを維持するために、分離 UR にバインドしておく必要があります。

コントロール表の再編成

頻繁に更新される動的なコントロール表は定期的に再編成する必要があります。

このタスクについて

CD 表および IBMSNAP_UOW 表は、変更キャプチャー時には多数の INSERTS を受け取り、整理時には多数の DELETES を受け取ります。IBMSNAP_CAPMON 表、 IBMSNAP_CAPTRACE 表、および IBMSNAP_APPLYTRAIL 表のサイズは、レプリケーションのソース表の更新率によって、大きく変化する可能性があります。

推奨: 以下の動的なコントロール表は週に一度は再編成してください。

- CD 表
- IBMSNAP_ALERTS
- IBMSNAP_APPLYTRACE
- IBMSNAP_APPLYTRAIL
- IBMSNAP_CAPMON
- IBMSNAP_CAPTRACE
- IBMSNAP_MONTRAIL
- IBMSNAP_MONTRACE
- IBMSNAP_UOW

その他のコントロール表に対しては、未使用のスペースを再利用するためのユーティリティーを実行したり、頻繁に更新されるオプティマイザーの統計を生成したりする必要はありません。

手順

コントロール表を再編成するには、次の方法のいずれかを使用します。

方法	説明
z/OS **PREFORMAT オプションを指定した REORG ユーティリティー**	このユーティリティーの PREFORMAT オプションは、キャプチャー・プログラムの挿入処理を迅速化します。
System i **RGZPFM (物理ファイル・メンバーの再編成) コマンド**	ENDDPRCAP コマンドで **RGZCTLTBL(*YES)** パラメーターを指定すると、キャプチャー・プログラムが終了したときに UOW 表およびアクティブな CD 表を再編成できます。
Linux UNIX Windows **REORG コマンド**	このコマンドを使用して、フラグメント化されたデータを除去して、スペースを再利用します。

キャプチャー・プログラムによって保守される動的なコントロール表の整理 (Linux、 UNIX、Windows、z/OS)

サイズが変動する表を手動で整理することも、自動的に整理することも可能です。

このタスクについて

以下の動的なコントロール表の拡張をモニターし、これらの表で使用可能な各種の整理方法を考慮してください。

- CD 表
- IBMSNAP_UOW
- IBMSNAP_CAPMON
- IBMSNAP_CAPTRACE
- IBMSNAP_SIGNAL
- System i IBMSNAP_AUTHTKN

これらの表で整理が一定インターバルで自動的に行われるように、キャプチャー・プログラムを設定できます。また、整理プロセスを立ち上げることにより、要求時に一度だけ整理を行うこともできます。ユーザーが次に整理コマンドを入力するまで、キャプチャー・プログラムは整理を行いません。

手順

キャプチャー・プログラムによって保守される動的なコントロール表を整理するには、以下のようにします。

1. 動的なコントロール表を自動的に整理する場合は、以下のいずれかの方法で **autoprune** パラメーターを yes に設定します。

方法	説明
自動整理を指定してキャプチャー・プログラムを始動します。	**autoprune**=y を指定して、asncap システム・コマンドを実行します。 **prune_interval** パラメーターを設定して、自動整理プロセスの発生頻度を指定します。
実行中のキャプチャー・プログラムで自動整理を使用可能にします。	**autoprune**=y を指定して、asnccmd chgparms コマンドを実行します。 **prune_interval** パラメーターを設定して、自動整理プロセスの発生頻度を指定します。

2. 動的なコントロール表を 1 回整理するには、以下のいずれかの方法を使用します。

方法	説明
レプリケーション・センター	「キャプチャー・コントロール表の整理」ウィンドウを使用して、表の整理を一度行います。そのウィンドウを開くには、オブジェクト・ツリーの「**操作**」ブランチで「**キャプチャー・コントロール・サーバー**」フォルダーをクリックし、内容ペインでサーバーを右クリックし、「**キャプチャーの整理**」をクリックします。

方法	説明
実行中のキャプチャー・プログラムから整理操作を 1 回開始します。	整理パラメーターを指定して、 asnccmd システム・コマンドを実行します。

CD 表と UOW 表の整理

キャプチャー・プログラムは、自動的に呼び出された場合も、要求時に実行する場合も、アプライ・プログラムから報告された進行に基づいて、各整理サイクルで CD 表および UOW 表の整理を行います。

整理の進行状況は、IBMSNAP_PRUNE_SET 表の SYNCHPOINT 列の値に示されます。この通常の整理は、各 CD 表をサブスクライブするすべてのアプライ・プログラムを通じて最小の同期点値に基づいて、また UOW 表の場合は全体を通じて最小の同期点値に基づいて行われます。

しかし通常の整理では、関連するサブスクリプション・セットが非常にまれにしか実行されない場合は、CD 表および UOW 表の効果的な整理を行うことはできません。関連するアプライ・プログラムの実行頻度を決定するとき、これらのアプライ・プログラムを停止するとき、そしてサブスクリプション・セットを比較的長い間非活動化するときには、整理の効率性を考慮してください。

サブスクリプション・セットの実行が極めてまれであったり、アプライ・プログラムを停止すると、使用している CD 表と UOW 表が非常に大きくなり、保持制限整理の対象となる可能性があります。保持制限は、キャプチャー・プログラムの稼働パラメーターであり、このデフォルト値は 1 週間です。この値は、表の中の古いデータがどれくらいたつと保持制限整理に適格になるかを決定します。

サブスクリプション・セットが非アクティブになるか、まれにしか実行されないために、通常の整理処理が使用禁止になると、データが非常に長い時間表の中に留まる可能性があります。このデータが DB2 の現行タイム・スタンプから保持制限値を引いた値よりも古くなると、保持制限整理処理は、このデータを表から整理します。

保持制限整理を必要とするような条件の発生は回避してください。古いデータが累積されることにより、ストレージのオーバーフローが発生し、性能低下を招く可能性があります。

推奨: すべてのサブスクリプション・セットに対して、アプライ・プログラムを少なくとも 1 日に 1 回実行してください。

ソース・サーバーが多様なターゲット・システムに変更データを提供している場合、そして、各ターゲットの要件が大きく異なるものであり、また、ターゲットによっては、少数の登録済みソースに対してアプライ・プログラムがまれにしか実行されない場合は、複数のキャプチャー・プログラムを使用することを考えてみてください。複数のキャプチャー・プログラムを実行することにより、異なるキャプチャー・スキーマを使用して、多様な処理要件を管理できます。1 つのキャプチャ

ー・スキーマを使用して、サブスクリプション・セットの固有のタイミング要件によりまれにしか整理されない表を分離し、残りのソース表に対しては、別のキャプチャー・スキーマを使用できます。

その他の動的コントロール表の整理に関する推奨事項

古いデータを除去し、システム・パフォーマンスを向上させるために、レプリケーション・コントロール表を定期的に整理してください。

キャプチャー・プログラムは、自分が保守している表に対してのみ整理操作を実行します。整合変更データ (CCD) 表はアプライ・プログラムによって保守されているため、キャプチャー・プログラムはこれらの表を自動的に整理しません。CCD 表のタイプによっては、整理の必要のないものもあります。完全なコンデンス CCD 表は、同じ場所で更新されます。

完全なコンデンス CCD 表からユーザーが除去できる唯一のレコードは、従属するターゲット表にすでに複製済みで、 IBMSNAP_OPERATION 列の値が D (削除) のものです。非コンデンス CCD 表は、履歴データを含むものであり、非常に大きくなる可能性があります。このデータは監査のために保存しておく必要があるため、非コンデンス CCD 表に対しては整理操作を実行しないでください。

しかし、内部 CCD 表の整理は考慮に値します。これらの表は、システムの更新アクティビティーが多い場合は、すぐに大きくなる可能性があります。内部 CCD 表からは最新の変更だけがフェッチされるため、古い行を保存しておく必要はありません。

内部 CCD 表の整理を使用可能にするために、従属するすべてのターゲットへのアプライがすでに完了している変更データを整理する、事後に実行される SQL ステートメントを、関連するサブスクリプション・セットに追加することを考えてみてください。また、自動スケジューリング機能に必要な SQL DELETE ステートメントを追加して、これらの表から行を削除することもできます。

IBMSNAP_APPLYTRAIL 表と IBMSNAP_APPLYTRACE 表については、手動での整理も行ってください。頻繁に実行されるアプライ・プログラムで複数のサブスクリプション・セットを定義および使用する場合、IBMSNAP_APPLYTRAIL 表は急速に拡張するため、頻繁な整理が必要です。これらの表の拡張を管理する最良の方法は、サブスクリプション・セットの 1 つに、事後に実行される SQL ステートメントまたはプロシージャー呼び出しを追加することです。また、自動スケジューリング機能に SQL DELETE ステートメントを追加することもできます。

レプリケーションの失敗の防止およびエラーからのリカバリー

これらのトピックでは、コントロール表およびレプリケーション・データに影響を与える可能性のある、レプリケーションの失敗を防止し、失敗から回復する方法について説明します。

キャプチャー・プログラムのコールド・スタートの防止

キャプチャー・プログラムのコールド・スタートは、プログラムを初めて開始するとき、またはコントロール表およびターゲット表のリフレッシュが必要な場合にだけ実行してください。キャプチャー・プログラムをコールド・スタートすると、レプリケーション環境内のすべてのターゲット表がリフレッシュされます。

z/OS Linux UNIX Windows warmns または warmsi オプションを指定してキャプチャー・プログラムを始動した場合、プログラムは IBMSNAP_RESTART 表の中の再始動点に基づいてログ・レコードのリトリーブを試みます。キャプチャー・プログラムがログを検出できないと、キャプチャー・プログラムのウォーム・スタートは失敗します。

キャプチャー・プログラムのコールド・スタートを防止するために、以下の推奨事項を考慮してください。

- System i RESTART(*YES) パラメーターを指定してキャプチャー・プログラムを始動してください。キャプチャー・プログラムは、以前に終了したときに停止したポイントから処理を継続します。システム上に十分な DB2 ログ・データまたはジャーナル・レシーバーを保存し、そのデータを SQL レプリケーションで使用できるようにしてください。
- レプリケーション・アラート・モニター、またはその他のメカニズムを使用して、キャプチャー・プログラムからの履歴データの状況を確認してください。次にこの情報を使用して、 DB2 がアクティブのときには必ずキャプチャー・プログラムが実行されていることを検証できます。
- システム上に十分な DB2 ログ・データおよびジャーナル・レシーバーが保存され、このデータが SQL レプリケーションから使用可能であることを確認してください。

コントロール表の入出力エラーおよび接続障害からのリカバリー

レプリケーションがコントロール表への接続を失った場合は、表のリカバリーが可能です。他のエラーの場合は、レプリケーション・プログラムがシャットダウンします。

このタスクについて

キャプチャー・プログラムは入出力エラーまたは接続障害を検出すると、適切なエラー・メッセージを発行してシャットダウンします。

アプライ・プログラムは、コントロール表に重大なエラーを検出するとシャットダウンします。アプライ・プログラムはターゲット表のエラー、またはネットワーク接続のエラーを検出すると、IBMSNAP_APPLYTRAIL 表にエラーを書き込み、処理を続行します。

手順

コントロール表のエラーや接続障害からリカバリーするには、以下のようにします。

1. コントロール表で入出力エラーまたは接続障害が発生した場合には、DB2 の標準リカバリー手順を使用して、表の順方向リカバリーを行います。 表はデータを失いません。
2. プログラムがシャットダウンした場合は、障害ポイントからキャプチャー・プログラムを再始動し、アプライ・プログラムを再始動します。

脱落したソース・データのリトリーブ

ソースを失った場合は、リカバリー・ポイントを使用する方法またはフル・リフレッシュによってデータをリトリーブできることがあります。

このタスクについて

ソース表で障害時点まで順方向リカバリーが行われると、SQL レプリケーションは正常に進行します。表がリカバリーされると、キャプチャー・プログラムは表のデータ変更の収集を続行します。

しかし、キャプチャー・プログラムおよびアプライ・プログラムは、読み取り専用ターゲット表のポイント・イン・タイム・リカバリーは検出しません。ユーザーがソース表を回復した場合、アプライ・プログラムが、ソース表にもはや存在していない変更をターゲット表に複製した可能性があるため、ユーザーがターゲット表を同じ論理ポイント・イン・タイムまで戻せない場合は、ソース表とターゲット表の間で不整合が生じる可能性があります。

複数レベルでレプリケーションが行われる場合は、この状況がより複雑なものになります。各レベル間のリカバリー点を照合するためのメカニズムを提供するか、リカバリー方式としてフル・リフレッシュを使用する必要があります。

手順

次の方法のいずれかを使用して、ソース・データをリカバリーします。

方法	説明
リカバリー・ポイントのメカニズム	各レベルのレプリケーションの中から一致するリカバリー・ポイントを用意するためのメカニズムを開発します。
フル・リフレッシュ	リカバリー方式としてフル・リフレッシュを選択します。

IBMSNAP_CAPMON 表と IBMSNAP_CAPTRACE 表の整理

IBMSNAP_CAPMON 表と IBMSNAP_CAPTRACE 表の整理は、操作パラメーターの値によって制御します。

キャプチャー・プログラムは、キャプチャー・プログラムの以下の稼働パラメーターの値に基づいて、各整理サイクルで IBMSNAP_CAPMON 表および IBMSNAP_CAPTRACE 表の整理を行います。

- IBMSNAP_CAPMON 表に行を保持する時間の長さを示す **monitor_limit** パラメーター (Linux、UNIX、Windows、z/OS) と **MONLMT** パラメーター (System i)
- IBMSNAP_CAPTRACE 表に行を保持する時間の長さを示す **trace_limit** パラメーター (Linux、UNIX、Windows、z/OS) と **TRCLMT**パラメーター (System i)

モニター限度パラメーターおよびトレース限度パラメーターのデフォルト値は両方ともに 1 週間です。IBMSNAP_CAPMON 表の中にキャプチャー・プログラムの待ち時間およびスループットの履歴情報をどのくらい長く保持するか、IBMSNAP_CAPTRACE 表の中に監査およびトラブルシューティングのデータをどれくらい長く保持するかに応じて、これらの値を変更できます。

IBMSNAP_SIGNAL 表の整理

レプリケーションの実行時には常に行が追加されていくので、IBMSNAP_SIGNAL 表は自動的に整理されます。

IBMSNAP_SIGNAL 表もまた、各整理サイクル中に整理されます。シグナル行は、SIGNAL_STATE 列の値が C になると整理に適格になります。 C という値は、シグナル情報が完成しており、キャプチャー・プログラム、またはその他のユーザー処理から必要とされることはないため、整理に適格であることを意味します。SIGNAL_TIME 列の値が、DB2 の現行タイム・スタンプから保持制限パラメーターの値を引いたものよりも古いシグナル行は、保持制限整理に適格です。

ターゲット表の保守

ターゲット・サーバー上の表は、データベース・システムの他の表を保守するのと同じ方法で保守してください。

ターゲット表が既存のデータベース表である場合も、 SQL レプリケーションにより自動的に生成されるように指定された表である場合も、これらの表に対して現在のバックアップおよび保守のルーチンを使用してください。

注: ユーティリティーを実行するためにターゲット表をオフラインにする前に、アプライ・プログラムを非活動化してください。

第 14 章 ソース表とターゲット表間の相違検出および修復

asntdiff および asntrep ユーティリティーを使用すると、手動でソース表とターゲット表を比較したり、ターゲットのロード (フル・リフレッシュ) を行ったりしなくても、Q レプリケーションおよび SQL レプリケーションで、この 2 つの表の相違を検出し、修復することができます。

このタスクについて

ターゲット表にユーザーまたはアプリケーションが予期しない変更を行った場合、または、システムが拡張ネットワークや、ターゲット・システムの障害を起こしている場合などは、ソース表とターゲット表の同期ができません。

asntdiff および asntrep ユーティリティーは、Q キャプチャー、Q アプライ、キャプチャー、およびアプライ・プログラムとは別に実行されます。 DB2 SQL を使用してソース表とターゲット表からデータを取り出しますが、WebSphere MQ キューは使用しません。このユーティリティーは、ログ、トリガー、または分離レベルには依存していません。

手順

ソース表とターゲット表の相違を検出して修復するには、asntdiff ユーティリティーを実行してから、asntrep ユーティリティーを実行します。

表相違検出ユーティリティー (asntdiff)

asntdiff ユーティリティーは、ソース表にあるすべての列を、ターゲット表の対応する列と比較し、2 つの表の相違リストを DB2 表の形式で生成します。

asntdiff ユーティリティーを使用する場合、asntdiff コマンドを実行し、比較するソースおよびターゲット表を含む Q サブスクリプション (Q レプリケーション) の名前またはサブスクリプション・セット・メンバー (SQL レプリケーション) の名前を指定します。

asntdiff コマンドは、Linux、 UNIX、 Windows、および z/OS オペレーティング・システム上で実行可能です。コマンドは、Linux、 UNIX、 Windows、 z/OS、または System i オペレーティング・システム上で表を比較します。 asntdiff コマンドは、2 つの表のそれぞれ対応する列のデータ・タイプが同じであれば、フェデレーテッド・ソースとフェデレーテッド・ターゲットで使用できます。

Q レプリケーションの場合、ターゲットは表でなければならず、ストアード・プロシージャーにすることはできません。SQL レプリケーションの場合、ターゲットはユーザー表、ポイント・イン・タイム表、レプリカ表、ユーザーがコピーした表でなければなりません。

コマンドを実行して、Q サブスクリプションまたはサブスクリプション・セット・メンバーを一意的に識別する SQL WHERE 文節を以下のように指定します。

Q レプリケーション

WHERE 文節は、Q キャプチャー・サーバーのある IBMQREP_SUBS コントロール表の行を、SUBNAME 列の値に基づいて識別します。例:

```
where="subname = 'my_qsub'"
```

SQL レプリケーション

WHERE 文節は、アプライ・コントロール・サーバーにある IBMSNAP_SUBS_MEMBR 表の行を、SET_NAME 列の値に基づいて識別します。例:

```
where="set_name = 'my_set' and source_table='EMPLOYEE'"
```

WHERE 文節でさらに多くの述部を使用して、一意的にサブスクリプション・セット・メンバーを識別することが必要になる場合があります。例えば、IBMSNAP_SUBS_MEMBR 表からの APPLY_QUAL、SOURCE_OWNER、TARGET_OWNER、または TARGET_TABLE 列を、文節に追加することが必要になる場合があります。

相違検出表

asntdiff コマンドを使用して、Q レプリケーションおよび SQL レプリケーションのソース・データベースまたはサブシステムに相違検出表を作成します。

相違検出表の名前は、*schema*.ASNTDIFF になります (*schema* は、DIFF_SCHEMA パラメーターに指定されている値)。スキーマを指定しない場合は、デフォルトで ASN になります。 DIFF パラメーターを使用して、表名を指定することもできます。

デフォルトでは、相違検出表はデフォルトの DB2 ユーザー表スペース中に作成されます。DIFF_TABLESPACE パラメーターを使用して、相違検出表スペースを指定できます。相違検出表を作成して相違検出表スペースを指定することもできます。相違検出表を作成すると、その表に含まれている行は、asntdiff コマンドを使用する際に削除されます。

相違検出表には 2 列以上の列があります。1 つの列の名前は「DIFF 」で、末尾にブランク・スペースがあります。DIFF 列の値は、挿入、更新、または削除操作を示す文字の後に、どの表に相違のある行が入っているかを示す数値が続いたものです。他の列には、レプリケーション・キー列の値が入っています。相違検出表には、ターゲット表の一致していない行ごとに 1 行があります。

相違検出表は、以下の 3 つの ID を使用して、ターゲット表に変更を加えてソース表と一致させるのに必要な操作を示します。

D (削除)

キー値のある行がターゲットのみにあり、ソースにないことを示します。

U (更新)

同じキー値のある行がソースとターゲットの両方にあるが、ターゲットの 1 つ以上の非キー列が違っていることを示します。

I (挿入)

キー値のある行がソースのみにあり、ターゲットにないことを示します。

値 ? 1 は、1 列以上のソース列に無効文字があることを示しています。

値 ? 2 は、1 列以上のターゲット列に無効文字があることを示しています。

例

以下のリストは、ソースの EMPLOYEE 表を同じ表のターゲット・コピーと比較することによって戻される値です。レプリケーションのキー列は、以下のように従業員番号 EMPNO です。

```
DIFF  EMPNO
U 2   000010
I 2   000020
I 2   000040
D 2   000045
I 2   000050
D 2   000055
```

例の先頭行は、キー値 000010 のある行がソース表とターゲット表の両方にあるが、ターゲットの 1 つ以上の非キー列の値が違っていることを示します。次の 2 行は、キー値 000020 および 000040 のある行がソースのみにあることを示します。4 番目の行は、キー値 000045 のある行がターゲットのみにあることを示します。

値? 1 および ? 2 はこの例にはありません。

削除操作の抑制

Q レプリケーションでは、ソース表からの削除操作のレプリケーションを抑制することを選択できます。削除操作を複製しないと、ターゲット表にある行がソース表にない可能性があります。Q サブスクリプションの SUPPRESS_DELETES 値が Y の場合は、asntdiff ユーティリティーは、ターゲット固有の行を無視し、相違なしと報告します。警告が発行されて、抑制された行の数が示されます。

ソースとターゲットのデータ・タイプの相違

asntdiff ユーティリティーは、サブスクリプションの記述に基づいて 2 つの SELECT SQL ステートメントを構築します。このユーティリティーは、両方のステートメントの実行結果のデータを比較して、ソース表とターゲット表の間の相違を入手します。両方の SQL ステートメントの列のデータ・タイプは同じでなければなりません。

SQL レプリケーション
このユーティリティーは、IBMSNAP_SUBS_COLS 表の EXPRESSION 列を使用して、ソースの SQL ステートメントを構築します。

Q レプリケーション
ソースとターゲットの両方のデータ・タイプは同じでなければなりません。

GRAPHIC データ・タイプの比較

asntdiff ユーティリティーを使用してソース表とターゲット表を比較する際に、ソースとターゲットの GRAPHIC グラフィック・データの列が一致しないことがあります。DB2 の GRAPHIC データ・タイプの列は、グラフィック・データの後にブランクが埋め込まれます。この埋め込みは、データベースを作成した際のコード・ページに応じて、単一バイトまたは 2 バイトのスペースになります。この埋め込みのために、ソース表とターゲット表の間でデータが一致しないことがあります (特に、

ソース表とターゲット表のコード・ページが違う場合)。この埋め込みは GRAPHIC データ・タイプのみに適用され、VARGRAPHIC や LONG VARGRAPHIC などの他のグラフィック・データ・タイプには適用されません。

GRAPHIC データ・タイプの列を比較するには、DB2 スカラー関数 rtrim(<column> を使用してソース表とターゲット表を比較する前に、データ中のブランク埋め込みを除去しなければなりません。この関数は、単一バイト・スペースか 2 バイト・スペースかによるコード・ページの相違を除去し、asntdiff ユーティリティーが一貫性のある方法で GRAPHIC データを比較することを保証します。

述部

ソース表とターゲット表の相違が計画的な場合もあります。例えば、Q レプリケーションで検索条件を使用して、複製される行をフィルターに掛ける場合などが該当します。このユーティリティーは、述部の結果によるソース表とターゲット表の相違は示しません。

SQL レプリケーション
このユーティリティーは IBMSNAP_SUBS_MEMBR 表の PREDICATES 列を使用して、ソース表から行を選択します。UOW_CD_PREDICATES 列の値は無視されます (asntdiff はソース表を直接参照しますが、アプライ・プログラムは CD 表を参照します)。

Q レプリケーション
このユーティリティーは IBMQREP_SUBS 表の SEARCH_CONDITION 列の値を使用して、SELECT ステートメントの WHERE 文節を構築します。

asntdiff ユーティリティーの使用時期

asntdiff ユーティリティーは、ソースおよびターゲット表が安定しているときに使用するのが最適です。このユーティリティーは、Q キャプチャー・プログラムおよび Q アプライ・プログラムまたはキャプチャー・プログラムおよびアプライ・プログラムがアイドル状態の時に実行することができます。たとえば、Q キャプチャー・プログラムが、DB2 リカバリー・ログの末尾に到達し、すべての変更がターゲットにアプライされると、ユーティリティーの実行が可能になります。アプリケーションがソースを更新中の場合、比較は正確ではない場合があります。

レプリケーション・プログラムが実行中であると、asntdiff コマンドを 1 回より多く (2 回以上) 実行して、ソースおよびターゲット表との間の進行中の相違の全体像を取得しなければならないことがあります。

表修復ユーティリティー (asntrep)

asntrep ユーティリティーは、ターゲット表で行の削除、挿入、および更新を行うことにより、すべての DB2 サーバー上のソース表とターゲット表の相違を修復します。このユーティリティーは、Linux、UNIX、または Windows オペレーティング・システム上で実行されます。

asntrep ユーティリティーは、asntdiff ユーティリティーで生成された相違検出表を使用して、以下の作業を行います。

- ソース表に一致するキーがない行をターゲット表から削除します。

- ソース表にあるが、一致するキーがない行をターゲット表に挿入します。
- ソースに一致するキーがあるが、非キー・データが違うターゲット行を更新します。

Q レプリケーションの場合、ターゲットは表でなければならず、ストアード・プロシージャーにすることはできません。SQL レプリケーションの場合、ターゲットはユーザー表、ポイント・イン・タイム表、レプリカ表、ユーザーがコピーした表でなければなりません。Q サブスクリプションでピアツーピア・レプリケーションを行うときに、asntrep ユーティリティーを使用する場合、論理表のすべてのコピーを一度に 2 コピーずつ修復しなければなりません。

asntrep ユーティリティーを使用する場合は、asntdiff コマンドの実行後に asntrep コマンドを実行します。 asntrep コマンドは、ソース・データベースまたはサブシステムからターゲットへ相違検出表をコピーし、この表を使用して、ターゲット表を修復します。

asntrep コマンドは、ターゲット・データベースまたはサブシステムから相違検出表をドロップしません。ユーザーが手動で、表をドロップしなければなりません。

asntrep コマンドを使用する場合、asntdiff コマンドに使用したものと同じ WHERE 文節を使用して、同期させるソース表とターゲット表のある Q サブスクリプションまたはサブスクリプション・セットのメンバーを識別します。

修復プロセス中に、ターゲット表の参照整合性制約はドロップされません。挿入または削除が、参照整合性制約に違反しているため、ターゲット表からの行の挿入または削除操作が失敗する可能性があります。また、重複したソース行があると、ターゲットにユニーク索引がある場合にターゲットの修復が不可能になる可能性があります。

第 15 章 レプリケーション・アラート・モニター

レプリケーション・アラート・モニターを使用して、SQL レプリケーション環境、Q レプリケーション環境、またはイベント発行環境をモニターできます。

レプリケーション・アラート・モニターは、Classic レプリケーション・ソースの状況をチェックできませんが、Classic レプリケーション構成で DB2 またはフェデレーテッド・ターゲット・サーバーをモニターできます。

以下のトピックでは、レプリケーション・アラート・モニターの動作や、レプリケーション/パブリッシング環境のためのモニターの構成や操作の方法について説明しています。

レプリケーション・アラート・モニターによるレプリケーションのモニター

レプリケーション・アラート・モニターは、レプリケーション環境の状況の変化についてアラートを出すことができるプログラムです。

レプリケーション・アラート・モニターは、実行中にレプリケーションの状況を自動的にチェックし、レプリケーション環境で発生した特定の条件について通知します。例えば、SQL レプリケーションの場合、レプリケーション・アラート・モニターは、アプライ・プログラムの終了時に通知できます。同様に、Q レプリケーションの場合、レプリケーション・アラート・モニターは、Q キャプチャー・プログラムが Q サブスクリプションを非アクティブ化すると、そのことをユーザーに通知できます。

制約事項: レプリケーション・アラート・モニターは、Classic レプリケーション・ソースの状況をチェックできませんが、Classic レプリケーション構成で DB2 またはフェデレーテッド・ターゲット・サーバーをモニターできます。

レプリケーション・アラート・モニターを構成する方法は 2 つあります。

1 つのモニター

一般的には、少数のレプリケーション・プログラムをモニターする場合には、 1 つのモニターを使用します。 1 つのモニターをセットアップすると、すべてのコントロール情報が 1 つのサーバー上に保管されます。個々のモニターは複数のレプリケーション・プログラムをモニターできますが、モニターが個々のサーバー上で一度にチェックするアラートは 1 つのみです。いずれか 1 つのサーバーに戻る前に、モニター対象の他のサーバーをすべてチェックしなければなりません。

複数のモニター

多数のレプリケーション・プログラムをモニターしたり、特定のプログラムのモニターを優先順位付けしたり、モニターの作業負荷を分割したりするには、追加のモニターを使用します。システム内の各サーバーをチェックするために、それぞれ独立したモニターを作成します。これらのモニターは相互に通信しませんが、サーバーに関するアラートを送信します。複数のモニタ

ーをセットアップすると、個々のモニターのコントロール情報は、モニターに割り当てられているサーバー上に保管されます。以下のようにするには、複数のモニターを使用してください。

- **レプリケーション・プログラムごとにモニター頻度を変える。** レプリケーション・プログラムでアラート条件をチェックする頻度を上げるには、monitor_interval を小さくしてモニターをセットアップします。例えば、あるモニターを、あるキャプチャー・サーバーで 15 分おきに CAPTURE_WARNINGS アラート条件をモニターするよう割り当てることができます。別のモニターを、別のキャプチャー・サーバーで 50 分おきに CAPTURE_WARNINGS アラート条件をモニターするよう割り当てることができます。
- **さまざまなアプリケーションを別々にモニターする。** レプリケーション・アプリケーションごとにモニターをセットアップしてください。例えば、さまざまなグループに別々のモニターがアラートを送信したり、管理者が 2 種類のアプリケーション用にアラートを区別するのに役立てることができます。同様に、さまざまな条件をチェックするのに、別々のアラート条件を割り当てることができます。
- **アラート条件の優先順位付けを行う。** 例えば、QAPPLY_STATUS アラート条件を使用して、 10 分おきに Q アプライ・プログラムの状況をモニターすることもできます。一方、QAPPLY_MEMORY アラート条件を使用して、 300 分おきに同じ Q アプライ・プログラムのメモリーをモニターすることもできます。

レプリケーション・アラート・モニターのコンポーネントについて説明する用語を以下に示します。

モニター

モニターは、レプリケーション・アラート・モニターの 1 つのインスタンスまたはオカレンスです。モニターをセットアップして、 1 つ以上のサーバー上で実行しているレプリケーション・プログラムの状況をチェックできます。個々のモニターは、割り当て先の 1 つ以上のサーバー上のレプリケーション・アクティビティーをチェックします。

モニター修飾子

モニター修飾子は、ユーザー指定のモニター名です。すべてのモニターにユニークなモニター修飾子があります。

モニター・コントロール・サーバー

モニター・コントロール・サーバーとは、レプリケーション・アラート・モニターのコントロール情報が含まれるサーバーのことです。

アラート

アラートとは、レプリケーション環境中のイベントや条件に関して知らせる通知のことです。レプリケーション・アラート・モニターは、 E メールかページャーを使用してアラートを送信します。

アラート条件

アラート条件とは、レプリケーション・アラート・モニターがアラートを送信するレプリケーション環境の条件のことです。アラート条件には 3 種類

あり、それは状況によって起動されるアラート条件、イベントによって起動されるアラート条件、およびしきい値によって起動されるアラート条件です。

状況によって起動されるアラート条件
状況アラート条件は、レプリケーション・プログラムの状況について通知します。例えば、APPLY_STATUS アラート条件を指定すると、レプリケーション・アラート・モニターは、アプライ・プログラムが実行していない場合にアラートを送信します。

イベントによって起動されるアラート条件
イベント・アラート条件は、レプリケーション中に特定のイベントが発生した時点で通知します。例えば、QAPPLY_ERRORS アラート条件を指定すると、レプリケーション・アラート・モニターは、Q アプライ・プログラムが IBMQREP_APPLYTRACE 表にエラーを記録した時点でアラートを送信します。

しきい値によって起動されるアラート条件
しきい値アラート条件は、レプリケーション環境中でしきい値を超過した時点で通知します。例えば、QCAPTURE_MEMORY アラート条件を指定すると、レプリケーション・アラート・モニターは、Q キャプチャー・プログラムの使用メモリーがしきい値で許可されている量を超えた時点でアラートを送信します。

連絡先 連絡先とは、レプリケーション・アラート・モニターから送信されるアラートを受信するための E メール・アドレスかページャー・アドレスのことです。アラートは、z/OSコンソールに送信することも可能です。 ASNMAIL 出口ルーチンは、モニターに関する E メール通知を送信します。この出口ルーチンに変更を加えて、問題管理システムなどにアラートを書き込むこともできます。

連絡先グループ
連絡先グループとは、同じアラートを受信する連絡先の集合のことです。

z/OS

アラートを z/OS コンソールに送信するように指定することもできます。

レプリケーション・アラート・モニターは、Linux、 UNIX、 Windows、 z/OS の各オペレーティング・システムに対応した DB2 上のサーバーをモニターします。コントロール表がバージョン 8 以降のアーキテクチャーに配置されているレプリケーション・プログラムをモニターできます。

制約事項

- プラットフォーム固有の制約事項の中には、Q キャプチャー・プログラム、Q アプライ・プログラム、キャプチャー・プログラム、アプライ・プログラムの状況のモニターに該当するものがあります。これらの制約事項は、以下のアラート条件に適用されます。
  - QCAPTURE_STATUS
  - QAPPLY_STATUS
  - CAPTURE_STATUS
  - APPLY_STATUS

z/OS

z/OS サーバーから実行されるモニターは、レプリケーション・プログラムがそのモニターと同じ z/OS サブシステムで実行されていないと、レプリケーション・プログラムの状況をモニターできません。

Linux UNIX Windows

分散プラットフォームでモニターを実行しようとしている場合や、レプリケーション・プログラムの状況をモニターする場合は、DB2 Administration Server を、状況をモニターする Q キャプチャー、Q アプライ、キャプチャー、アプライ・プログラムと同じシステムで実行しなければなりません。レプリケーション・プログラムが分散プラットフォームまたは z/OS のどちらで実行されているかにかかわらず、状況をモニターするには DAS が必要です。

例えば、MONITOR1 が SERVER_LINUX1 サーバー上で実行されており、DB2 Administration Server が SERVER_ZOS1 にインストールされて実行されている場合、このモニターは、SERVER_ZOS1 サーバー上で実行されている Q アプライ・プログラムの状況をモニターできます。詳細については、「*IBM WebSphere Information Integration Replication Installation and Customization Guide for z/OS*」を参照してください。

- 非 DB2 リレーショナル・データベースの場合、レプリケーション・アラート・モニターは、フェデレーテッド・データベース・システムでソースとして使用される、そうしたデータベースに関連するトリガーはモニターしません。
- z/OS レプリケーション・アラート・モニターは、SMTP サーバーを使用して E メール通知を送信できますが、ASNMAIL 出口ルーチンを使用して通知を処理することはできません。
- System i System i サーバーをモニターするには、レプリケーション・アラート・モニターは、Linux、UNIX、Windows のサーバーで実行し、System i サーバーをリモートにモニターしなければなりません。 DB2 for i5/OS サーバー上でモニター・コントロール・サーバーをセットアップすることはできません。

レプリケーション・アラート・モニターのアラート条件および通知

レプリケーション・アラート・モニターは、特定のアラート条件が発生したときに通知を送信できます。

レプリケーション・アラート・モニターのアラート条件

アラート条件とは、モニターがアラートを送信するレプリケーション環境の条件のことです。アラートとは、アラート条件を引き起こした状況、イベント、またはしきい値を記述したメッセージのことです。

関連するパラメーター値を報告するアラートもあります。例えば、QCAPTURE_MEMORY アラート条件のメッセージは、Q キャプチャー・プログラムが使用しているメモリーの量と、超過したメモリーしきい値を報告します。

以下のセクションでは、レプリケーション環境をモニターするために使用できるアラート条件について説明します。

- 『Q キャプチャー・プログラムのアラート条件』
- 『Q アプライ・プログラムのアラート条件』
- 226 ページの『キャプチャー・プログラムのアラート条件』
- 227 ページの『アプライ・プログラムのアラート条件』

Q キャプチャー・プログラムのアラート条件

表 11 は、Q キャプチャー・プログラムのアラート条件を説明しています。

表 11. Q キャプチャー・プログラムのアラート条件

アラート条件	説明
QCAPTURE_STATUS	レプリケーション・アラート・モニターは、Q キャプチャー・プログラムが実行していないとアラートを送信します。
QCAPTURE_ERRORS	レプリケーション・アラート・モニターは、 IBMQREP_CAPTRACE 表の OPERATION 列中に値が ERROR の行を検出するとアラートを送信します。
QCAPTURE_WARNINGS	レプリケーション・アラート・モニターは、 IBMQREP_CAPTRACE 表の OPERATION 列中に値が WARNING の行を検出するとアラートを送信します。
QCAPTURE_LATENCY	Q キャプチャー待ち時間とは、データがデータベースに書き込まれた時刻と、 Q キャプチャー・プログラムがそのデータを渡した時刻との間の差を測定したものです。レプリケーション・アラート・モニターは、 Q キャプチャー待ち時間がユーザー指定のしきい値を超えているとアラートを送信します。 Q キャプチャー待ち時間は秒単位で測定されます。
QCAPTURE_MEMORY	レプリケーション・アラート・モニターは、 Q キャプチャー・プログラムが使用するメモリーがユーザー指定のしきい値を超えているとアラートを送信します。メモリーは MB 単位で測定されます。
QCAPTURE_TRANSIZE	レプリケーション・アラート・モニターは、 Q キャプチャー・プログラムが処理しているトランザクションの使用メモリーがユーザー指定のしきい値を超えているとアラートを送信します。メモリーは MB 単位で測定されます。
QCAPTURE_SUBSINACT	レプリケーション・アラート・モニターは、Q キャプチャー・プログラムが Q サブスクリプションを非アクティブ化するとアラートを送信します。

Q アプライ・プログラムのアラート条件

表 12 は、Q アプライ・プログラムのアラート条件を説明しています。

表 12. Q アプライ・プログラムのアラート条件

アラート条件	説明
QAPPLY_STATUS	レプリケーション・アラート・モニターは、Q アプライ・プログラムが実行していないとアラートを送信します。
QAPPLY_ERRORS	レプリケーション・アラート・モニターは、 IBMQREP_APPLYTRACE 表の OPERATION 列中に値が ERROR の行を検出するとアラートを送信します。
QAPPLY_WARNINGS	レプリケーション・アラート・モニターは、 IBMQREP_APPLYTRACE 表の OPERATION 列中に値が WARNING の行を検出するとアラートを送信します。

表 12. Q アプライ・プログラムのアラート条件 (続き)

アラート条件	説明
QAPPLY_LATENCY	Q アプライ待ち時間とは、 Q アプライ・プログラムが受信キューからトランザクションを受け取った後に、トランザクションがターゲット表にアプライされる時間を測定したものです。レプリケーション・アラート・モニターは、 Q アプライ待ち時間がユーザー指定のしきい値を超えているとアラートを送信します。 Q アプライ待ち時間はミリ秒単位で測定されます。
QAPPLY_EELATENCY	Q アプライ・エンドツーエンド待ち時間とは、レプリケーションが変更をキャプチャーし、それをターゲット・データベースにアプライするために必要な合計時間を測定したものです。レプリケーション・アラート・モニターは、Q アプライ・エンドツーエンド待ち時間がユーザー指定のしきい値を超えているとアラートを送信します。 Q アプライ・エンドツーエンド待ち時間は秒単位で測定されます。
QAPPLY_MEMORY	レプリケーション・アラート・モニターは、 Q アプライ・プログラムが使用するメモリーがユーザー指定のしきい値を超えているとアラートを送信します。メモリーは MB 単位で測定されます。
QAPPLY_EXCEPTIONS	レプリケーション・アラート・モニターは、IBMQREP_EXCEPTIONS 表に行が挿入されると、ターゲットでの競合や SQL エラーのためにアラートを送信します。
QAPPLY_RECVQINACT	レプリケーション・アラート・モニターは、受信キューが非アクティブ化されるとアラートを送信します。
QAPPLY_SPILLQDEPTH	レプリケーション・アラート・モニターは、予備キューの満杯率がユーザー指定のしきい値を超えているとアラートを送信します。満杯率はパーセントで表されます。
QAPPLY_QDEPTH	レプリケーション・アラート・モニターは、いずれかのキューの満杯率がユーザー指定のしきい値を超えているとアラートを送信します。満杯率はパーセントで表されます。

キャプチャー・プログラムのアラート条件

表 13 は、キャプチャー・プログラムのアラート条件を説明しています。

表 13. キャプチャー・プログラムのアラート条件

アラート条件	説明
CAPTURE_STATUS	レプリケーション・アラート・モニターは、キャプチャー・プログラムが実行していないとアラートを送信します。
CAPTURE_ERRORS	レプリケーション・アラート・モニターは、 IBMSNAP_CAPTRACE 表の OPERATION 列中に値が ERROR の行を検出するとアラートを送信します。
CAPTURE_WARNINGS	レプリケーション・アラート・モニターは、 IBMSNAP_CAPTRACE 表の OPERATION 列中に値が WARNING の行を検出するとアラートを送信します。
CAPTURE_LASTCOMMIT	レプリケーション・アラート・モニターは、キャプチャー・プログラムの最後のコミットから経過した時間がユーザー指定のしきい値を超えているとアラートを送信します。経過時間は秒単位で測定されます。

表 13. キャプチャー・プログラムのアラート条件 (続き)

アラート条件	説明
CAPTURE_CLATENCY	現行キャプチャー待ち時間とは、データがデータベースに書き込まれた時刻と、 Q キャプチャー・プログラムがそのデータを渡した時刻との間の差を測定したものです。レプリケーション・アラート・モニターは、現行キャプチャー待ち時間がユーザー指定のしきい値を超えているとアラートを送信します。
CAPTURE_HLATENCY	履歴キャプチャー待ち時間とは、最後にモニターがサーバーでアラート条件をチェックした後に測定した、すべてのキャプチャー待ち時間を複合したものです。レプリケーション・アラート・モニターは、履歴キャプチャー待ち時間がユーザー指定のしきい値を超えているとアラートを送信します。
CAPTURE_MEMORY	レプリケーション・アラート・モニターは、キャプチャー・プログラムが使用するメモリーがユーザー指定のしきい値を超えているとアラートを送信します。メモリーは MB 単位で測定されます。

アプライ・プログラムのアラート条件

表 14 は、アプライ・プログラムのアラート条件を説明しています。

表 14. アプライ・プログラムのアラート条件

アラート条件	説明
APPLY_STATUS	レプリケーション・アラート・モニターは、アプライ・プログラムが実行していないとアラートを送信します。
APPLY_SUBSFAILING	レプリケーション・アラート・モニターは、サブスクリプションが失敗するとアラートを送信します。
APPLY_SUBSINACT	レプリケーション・アラート・モニターは、サブスクリプションが非アクティブ化されるとアラートを送信します。
APPLY_ERRORS	レプリケーション・アラート・モニターは、 IBMSNAP_APPLYTRACE 表の OPERATION 列中に値が ERROR の行を検出するとアラートを送信します。
APPLY_WARNINGS	レプリケーション・アラート・モニターは、 IBMSNAP_APPLYTRACE 表の OPERATION 列中に値が WARNING の行を検出するとアラートを送信します。
APPLY_FULLREFRESH	レプリケーション・アラート・モニターは、フル・リフレッシュがあるとアラートを送信します。
APPLY_REJTRANS	レプリケーション・アラート・モニターは、サブスクリプション・セット中でトランザクションがリジェクトされるとアラートを送信します。
APPLY_SUBSDELAY	レプリケーション・アラート・モニターは、サブスクリプション処理の遅延がユーザー指定のしきい値より長くなるとアラートを送信します。
APPLY_REWORKED	レプリケーション・アラート・モニターは、アプライ・プログラムが再処理するサブスクリプション・セット中の行が、ユーザー指定のしきい値より多くなるとアラートを送信します。

表 14. アプライ・プログラムのアラート条件 (続き)

アラート条件	説明
APPLY_LATENCY	アプライ・エンドツーエンド待ち時間とは、レプリケーションが変更をキャプチャーし、それをターゲット・データベースにアプライするために必要な合計時間を測定したものです。レプリケーション・アラート・モニターは、アプライ・エンドツーエンド待ち時間がユーザー指定のしきい値を超えているとアラートを送信します。アプライ・エンドツーエンド待ち時間は秒単位で測定されます。

レプリケーション・アラート条件の E メール通知

レプリケーション・アラート・モニターは、アラート条件が発生した際に E メール通知できます。

E メール通知の内容は、指定された E メール・アドレスがページャー用のものかどうかによって異なります。次の例は、1 セットのアラートについて、それぞれの場合に予想される情報のタイプを示しています。ページャー以外の装置宛てに送信される E メールは、特定のサーバーで各アラート条件が発生した時刻を示しています。また、各アラート条件が発生した回数、および関連するメッセージも示しています。レプリケーション・アラート・モニターからページャー宛てに送信される E メールには、完全なメッセージの代わりに、アラートを起動したパラメーターのサマリーが含まれます。アラート条件が何回も発生した場合、タイム・スタンプはアラート条件が最後に発生した時刻を反映しています。

E メールのフィルターを回避するための ASNSENDER 変数の設定

ページャー・サービスなどの一部のプロバイダーでは、非送信請求メッセージのフィルターを実行するために完全な有効リターン・アドレスが必要です。有効なリターン・アドレスが指定されない場合は、E メールがブロックされる場合があります。

E メール・アドレスやページャーに対するアラートを受信していない場合は、以下の手順を実行してください。

1. レプリケーション・アラート・モニターを停止します。
2. ASNSENDER 環境変数を有効な E メール・アドレスに設定します。例:

   ```
   SET ASNSENDER=replmon@server.com
   ```

3. モニターを開始します。

ページャー以外の装置宛ての E メール通知の例 (SQL レプリケーション)

```
To:      repladmin@company.com
From:    replmon@server.com
Subject: Monitor: "MONQUAL" Alerts issued

ASN5129I MONITOR "MONQUAL". The Replication Alert Monitor on
server "WSDB" reports an e-mail alert

2002-01-20-10.00.00    1 ASN0552E Capture : "ASN" The program
encountered an SQL error. The server name is "CORP". The SQL
request is "PREPARE". The table name "PROD1.INVOICESCD".
The SQLCODE is "-204". The SQLSTATE is "42704".The SQLERRMC
is "PROD1.INVOICESCD". The SQLERRP is "readCD"
```

```
2002-01-20-10.05.00    2 ASN5152W Monitor "MONQUAL".The current
Capture latency exceeds the threshold value. The Capture control
server is "CORP". The schema is "ASN". The Capture
latency is "90" seconds.The threshold is "60" seconds

2002-01-20-10.05.00    4 ASN5154W Monitor "MONQUAL". The memory
used by the Capture program exceeds the threshold value. The
Capture control server is "CORP".  The schema is "ASN".
The amount of memory used is "34" bytes. The threshold is
"30" megabytes.
```

ページャー宛ての E メール通知の例 (SQL レプリケーション)

```
To:      repladmin@company.com
From:    replmon@server.com
Subject: Monitor: "MONQUAL" Alerts issued

MONQUAL - MONDB

2002-01-20-10.00.00 ASN0552E  1 CAPTURE-ERRORS - CORP - ASN
2002-01-20-10.05.00 ASN5152W  2 CAPTURE_CLATENCY - CORP - ASN - 90 - 60
2002-01-20-10.05.00 ASN5154W  4 CAPTURE_MEMORY - CORP - ASN - 34 - 30
```

SQL レプリケーションでは、モニターは通知を送信する際に、キャプチャー・コントロール・サーバー別およびアプライ・コントロール・サーバー別にアラートをグループ化します。 1 つのサーバーがキャプチャー・コントロール・サーバーとアプライ・コントロール・サーバーの両方である場合、モニターはそのサーバーのすべてのアラートをともにグループ化します。

Q レプリケーションでは、モニターは通知を送信する際に、 Q キャプチャー・サーバー別および Q アプライ・サーバー別にアラートをグループ化します。 1 つのサーバーが Q キャプチャー・サーバーと Q アプライ・サーバーの両方である場合、モニターはそのサーバーのすべてのアラートをともにグループ化します。

E メール通知のサイズがその E メールのタイプの制限を超えた場合は、モニターは複数の E メールに分けて通知を送信します。通常の E メール通知の最大サイズは 1024 文字です。ページャーの E メール・アドレスの場合の制限は 250 文字です。

ASNMAIL 出口ルーチンは、モニターに関する E メール通知を送信します。別のアラート処理を行うように、この出口ルーチンを変更できます。例えば、ASNMAIL ユーザー出口ルーチンが、アラートを問題管理システムに保管するようにできます。

z/OS コンソールへのアラートの送信

レプリケーション・アラート・モニターは、E メール・アドレスやページャーだけでなく、z/OS コンソールにもアラートを送信できます。z/OS コンソールに出力するには、モニターは z/OS 上で実行されている必要があります。

このタスクについて

モニターからアラートを受け取るように E メール・サーバーをすでに構成してある場合に、その後 z/OS コンソールにアラートを送信することを選択すると、z/OS コンソールと E メール・サーバーの両方にアラートが送信されます。

手順

z/OS コンソールにアラートを送信するようにレプリケーション・アラート・モニターをセットアップするには、次のようにします。

1. z/OS コンソールを指定するモニター・アラートを受信する連絡先を作成します。次の方法のいずれかを使用します。

方法	説明
ASNCLP コマンド行プログラム	CREATE ALERT CONDITIONS FOR コマンドでは、NOTIFY OPERATOR CONSOLE キーワードを使用して、 z/OS コンソールを、作成する条件で起動されるアラートの宛先として定義します。例: `CREATE ALERT CONDITIONS FOR QCAPTURE SCHEMA ASN1 MONITOR QUALIFIER MONQUAL NOTIFY OPERATOR CONSOLE (STATUS DOWN, ERRORS, WARNINGS)`
レプリケーション・センター	「アラート条件」ウィンドウの、「**オペレーター・コンソールに通知を送信**」チェック・ボックスにチェック・マークを付けます。

2. 以下のいずれかのオプションを使用して、**console** パラメーターを `Y` に設定します。

オプション	説明
JCL	モニターを開始するジョブで `CONSOLE=Y` と指定します。
asnmon コマンド (USS)	USS のコマンド・プロンプトから、**console** パラメーターを `Y` に設定して asnmon コマンドを実行します。例えば、次のようにします。 `asnmon monitor_server=SAMPLE monitor_qualifier=monqual console=y`
レプリケーション・センター	「今実行するか、コマンドを保管する」ウィンドウを使用し、モニターを開始する操作コマンドを編集して、**console** パラメーターを `Y` に設定してから、モニターを開始します。

レプリケーション中のアラートを送信する ASNMAIL 出口ルーチン (Linux、UNIX、Windows)

ASNMAIL 出口ルーチンは、レプリケーション環境内で発生した特定の条件について通知するアラートを配布します。

z/OS レプリケーション・アラート・モニターは、z/OS 上で ASNMAIL 出口ルーチンを使用して通知を処理することはできません。代わりに SMTP サーバーを使用できます。

この出口ルーチンは次の入力を受け取ります。

```
asnmail email_serverto_addresssubjectalert_messagealert_message
```

表 15では、ASNMAIL 出口ルーチンの入力について説明しています。

表 15. ASNMAIL 出口ルーチンの入力

入力	説明
email_server	これは、SMTP プロトコルを使用する E メール・サーバーのアドレスです。このサーバー・アドレスは、asnmon コマンドの始動時に指定された email_server パラメーターから渡されます。
to_address	これは、通知を受ける連絡先の E メール・アドレスです。
subject	これは通知の件名です。
alert_message	これは、アラート・メッセージを含むストリングです。

アラートを E メールで送信する代わりに、ASNMAIL 出口ルーチンに変更を加えて、問題管理システムなどにアラートを入力することもできます。¥sqllib¥samples¥repl¥ ディレクトリーには、ASNMAIL 出口ルーチンのサンプルが含まれます。asnmail.smp サンプルには、サンプル・プログラムを使用する場合の入力パラメーターと指示が含まれています。

レプリケーション・アラート・モニターのセットアップ

レプリケーション環境は、サーバー上で実行されるレプリケーション・プログラムと、それらのプログラムをサポートするコントロール表で構成されます。レプリケーション・アラート・モニターは、この環境をモニターします。

このタスクについて

以下のトピックでは、モニターをセットアップする前に検討するべき事柄について説明します。

- 232 ページの『レプリケーション・アラート・モニターの許可要件』
- 232 ページの『レプリケーション・アラート・モニターによって使用されるメモリー』
- 232 ページの『オプション : レプリケーション・アラート・モニター・プログラム・パッケージのバインディング (Linux、UNIX、Windows)』

手順

モニターをセットアップするには、次のようにします。

1. モニター・コントロール・サーバーごとのコントロール表の作成。
2. レプリケーション・アラート・モニターの連絡先情報の定義。
3. 1 つ以上のモニターの作成。
4. アラート条件の選択。
5. モニターの操作。
6. オプション: モニターの中断期間の定義。

レプリケーション・アラート・モニターによって使用されるメモリー

レプリケーション・アラート・モニターは、定義を格納したり、アラートを通知として送信する前にメモリー内で保持したりするために、メモリーを使用します。

定義用に必要なメモリーの量は、定義の数に直接比例します。レプリケーション・アラート・モニターは、アラート通知の保管用として 32 KB のメモリーを予約します。それより多いメモリーが、必要に応じて要求されたり、不要になった場合にはリリースされたりします。

推奨: レプリケーション・アラート・モニターにはメモリー割り当て量を設定しないことをお勧めします。設定する必要がある場合は、3 MB に設定してください。

レプリケーション・アラート・モニターの許可要件

レプリケーション・アラート・モニターを実行するすべてのユーザー ID には、モニターする Q キャプチャー・サーバーまたは Q アプライ・サーバーにアクセスする権限が必要です。またユーザー ID には、モニター・コントロール・サーバー上のモニター・コントロール表へのアクセス権も必要です。

モニターを実行するユーザー ID は、以下の権限および特権を持っている必要があります。

- モニター CONTROL 表に対する SELECT、UPDATE、INSERT、および DELETE 特権
- モニターするサーバー上の Q キャプチャー・コントロール表および Q アプライ・コントロール表に対する SELECT 権限
- BINDADD 権限 (モニター・パッケージに対して自動バインド機能を使用する場合にのみ必要)
- モニター・プログラム・パッケージに対する EXECUTE 特権
- レプリケーション・アラート・モニターが診断ファイルを保管する **monitor_path** ディレクトリーに対する WRITE 特権
- Linux UNIX Windows レプリケーション・アラート・モニターによって使用されるパスワード・ファイルへの読み取りアクセス

オプション : レプリケーション・アラート・モニター・プログラム・パッケージのバインディング (Linux、UNIX、Windows)

Linux、UNIX、Windowsの場合、レプリケーション・アラート・モニター・プログラムは実行時に自動的にバインドされます。バインド・オプションを指定したり、バインドのスケジュールを設定したり、すべてのバインド・プロセスが正常に完了したことを確認したりする場合は、パッケージを手動でバインドできます。

手順

モニター・プログラム・パッケージをバインドするには、以下のようにします。

1. モニター・プログラムのバインド・ファイルが配置されているディレクトリーに変更します。

プラットフォーム	バインド・ファイルの場所
Windows	*drive:*¥...¥sqllib¥bnd *drive:* は、DB2のインストール先のドライブです。
Linux UNIX	*db2homedir*/sqllib/bnd ここで、*db2homedir* は DB2 インスタンスのホーム・ディレクトリーです。

2. 各モニター・コントロール・サーバーについて、以下の手順を実行します。
   a. 以下のコマンドを入力して、データベースに接続します。

      ```
      db2 connect to database
      ```

      ここで、*database* はモニター・コントロール・サーバーです。データベースがリモート・データベースとしてカタログされている場合は、db2 connect to コマンドでユーザー ID とパスワードを指定する必要があります。例:

      ```
      db2 connect to database user userid using password
      ```

   b. 以下のコマンドを入力し、レプリケーション・アラート・モニター・プログラム・パッケージを作成してデータベースにバインドします。

      ```
      db2 bind @asnmoncs.lst isolation cs blocking all grant public
      db2 bind @asnmonur.lst isolation ur blocking all grant public
      ```

      なお、csはカーソル固定形式のリストを意味し、urは非コミット読み取り形式のリストを意味しています。

   これらのコマンドは、パッケージを作成します。その名前は、asnmoncs.lst および asnmonur.lst ファイル内で見つけることができます。
3. 現在モニターの対象としており、レプリケーション・アラート・モニター・プログラムの接続先となっているそれぞれのサーバーごとに、以下の手順を実行してください。
   a. 以下のコマンドを入力して、データベースに接続します。

      ```
      db2 connect to database
      ```

      database は、モニター対象のサーバーです。データベースがリモート・データベースとしてカタログされている場合は、db2 connect to コマンドでユーザー ID とパスワードを指定する必要があります。例:

      ```
      db2 connect to database user userid using password
      ```

   b. 以下のコマンドを入力し、レプリケーション・アラート・モニター・プログラム・パッケージを作成してデータベースにバインドします。

      ```
      db2 bind @asnmonit.lst isolation ur blocking all grant public
      ```

      この ur は、非コミット読み取り形式のリストを意味しています。

   これらのコマンドは、パッケージを作成します。その名前は、asnmonit.lst ファイル内で見つけることができます。

レプリケーション・アラート・モニターのコントロール表の作成

レプリケーション・アラート・モニターを使用できるようにするには、その前にモニター管理表を作成しなければなりません。この表には、アラート条件、連絡先情報、ランタイム・パラメーター、およびモニター用の他のメタデータが保管されます。

このタスクについて

モニター管理表を作成するサーバーは、モニター・コントロール・サーバーと呼ばれます。

DB2 for Linux, UNIX, Windows データベース、または DB2 for z/OS サブシステムをモニター・コントロール・サーバーにすることができます。たいていの場合、必要とされるモニター・コントロール・サーバーは 1 つだけですが、レプリケーション環境に応じて複数のサーバーを使用することができます。例えば、モニターするレプリケーション・プログラムと同じシステム上でモニターを実行する場合は、モニターを実行するサーバー上のローカル・モニターごとにコントロール表のセットを作成してください。

手順

モニター・コントロール表を作成するには、次の方法のいずれかを使用します。

方法	説明
ASNCLP コマンド行プログラム	CREATE CONTROL TABLES FOR コマンドを使用します。例: `CREATE CONTROL TABLES FOR MONITOR CONTROL SERVER;`
レプリケーション・センター	「モニター・コントロール表の作成」ウィンドウを使用します。このウィンドウをオープンするには、「**モニター・コントロール・サーバー**」フォルダーを右クリックして、「**モニター・コントロール表の作成**」を選択します。

レプリケーション・アラート・モニターの連絡先情報の定義

初めてレプリケーション・アラート・モニターを使用する場合は、その前にアラート条件の通知先の個人またはグループに関する連絡先情報を定義する必要があります。

このタスクについて

連絡先情報はモニター・コントロール・サーバーに保管されます。同じモニター・コントロール・サーバー上で実行するモニターは、連絡先を共有できます。複数のモニター・コントロール・サーバーがある場合には、それぞれのサーバーに連絡先を定義する必要があります。モニターの実行後に連絡先情報を変更できます。

各連絡先の E メール・アドレスと名前を指定して連絡先を定義した後で、連絡先をグループにできます。例えば、すべてのレプリケーション管理者に関する連絡先情報を含む **レプリケーション管理者**という連絡先グループをセットアップすることもできます。サーバー間で連絡先やグループの情報をコピーすることもできます。

レプリケーション・センターでレプリケーション・アラート・モニター用に作成した連絡先は、タスク・センターやヘルス・センターなどの他の DB2センターでは使用できません。他の DB2センターで作成された連絡先は、レプリケーション・アラート・モニターでは使用できません。

手順

レプリケーション・アラート・モニターの連絡先情報を定義するには、次のようにします。

1. 以下のいずれかの方法を使用して、モニター・コントロール・サーバー上で、モニターの連絡先と連絡先グループを作成します。

方法	説明

ASNCLP コマンド行プログラム	CREATE CONTACT コマンドを使用します。例:
	`CREATE CONTACT REPLADMIN`
	`EMAIL "repladmin@us.ibm.com"`
	`DESCRIPTION "replication administration";`

レプリケーション・センター	「連絡先の作成」ウィンドウまたは「連絡先グループの作成」ウィンドウを使用します。このウィンドウをオープンするには、連絡先または連絡先グループを追加するモニター・コントロール・サーバーを展開し、「**連絡先**」フォルダーを右クリックして、「**連絡先の作成**」 → 「**担当者**」 または「**連絡先の作成**」 → 「**グループ**」を選択します。

2. オプション: レプリケーション・センターの「連絡先およびグループのコピー」ウィンドウを使用して、モニター・コントロール・サーバー間で連絡先情報をコピーします。 このウィンドウをオープンするには、連絡先または連絡先グループが置かれているモニター・コントロール・サーバーを展開します。「**連絡先**」フォルダーを選択します。目次ペインで、コピーする連絡先または連絡先グループを右クリックして、「**コピー**」を選択します。
3. オプション: ASNCLP コマンド行プログラムで DELEGATE CONTACTS コマンドを使用して、一定期間、既存の連絡先を新しい連絡先に委任します。 例:

```
DELEGATE CONTACT REPLADMIN TO PERFORMACE FROM "2007-11-22" TO "2007-12-06"
```

レプリケーションまたは公開用のモニターの作成

モニター管理表を作成した後に、レプリケーション・センターの「モニターの作成ウィザード」を使用してモニターを作成し、レプリケーションまたは公開環境のモニターに使うアラート条件を選択します。

始める前に

モニターを作成するには、その前にレプリケーション・アラート・モニターをセットアップしなければなりません。

手順

モニターを作成するには、次のようにします。

1. レプリケーション・センターで、「モニターの作成ウィザード」をオープンし、モニターの名前と、モニターがアラート条件をチェックするレプリケーション・プログラムまたは公開プログラムを指定します。
   a. このウィザードを開くには、モニターを作成するモニター・コントロール・サーバーを展開し、「**モニター**」フォルダーを右クリックして、「**作成**」を選択します。
   b. 「開始」ページで、モニター修飾子を指定します。このモニターがアラート条件をチェックするプログラムを指定します。 SQL レプリケーションで使用するサブスクリプション・セットをモニターすることもできます。

   ウィザードでは以下の 1 つ以上のページが表示されるので、このモニターにアラート条件をチェックさせたいレプリケーション・プログラムに応じてアラート条件を選択できます。
   - Q キャプチャー・プログラムのアラート条件の選択。
   - Q アプライ・プログラムのアラート条件の選択。
   - キャプチャー・プログラムのアラート条件の選択。
   - アプライ・プログラムのアラート条件の選択。
   - サブスクリプション・セットのアラート条件の選択。

   詳細については、オンライン・ヘルプを参照してください。 例えば、Q キャプチャー・プログラムと Q アプライ・プログラムをモニターするように指定する場合、「モニターの作成」ウィザードには、「Q キャプチャー・プログラムのアラート条件の選択」ページと、「Q アプライ・プログラムのアラート条件の選択」ページが表示されます。
2. 上記のページの 1 つから、2 次ダイアログを開きます。そこでは以下のことを実行できます。
   a. モニターするプログラムまたはサブスクリプション・セットを指定します。
   b. チェックするアラート条件、および適切なアラート条件のパラメーターを指定します。 例えば、モニターでアラート条件を毎分チェックするには、**monitor_interval** パラメーター値を 60 に設定できます。
3. 「サマリー」ページで、「**完了**」をクリックします。

レプリケーション・アラート・モニターのアラート条件の選択

モニターの作成時に、モニターにアラートの送信を求めるアラート条件を選択します。モニター対象の各 Q キャプチャー・プログラム、Q アプライ・プログラム、キャプチャー・プログラム、アプライ・プログラム、またはサブスクリプション・セットごとにアラート条件を選択することができます。

このタスクについて

レプリケーション・アラート・モニターは、以下の時点でレプリケーション・プログラムと公開プログラムのアクティビティーをモニターします。

- 開始時に、個々のモニターが即時にアラート条件をチェックする。
- 指定した時間間隔で、個々のモニターが定期的にアラート条件をチェックする。

手順

レプリケーション・アラート・モニターのアラート条件を選択するには、次の方法のいずれかを使用します。

方法	説明
ASNCLP コマンド行プログラム	以下のいずれかのコマンドを使用します。 • CREATE ALERT CONDITIONS FOR APPLY • CREATE ALERT CONDITIONS FOR CAPTURE • CREATE ALERT CONDITIONS FOR Q CAPTURE • CREATE ALERT CONDITIONS FOR Q APPLY
レプリケーション・センター	モニター対象として選択したプログラムに応じて、レプリケーション・センターの「モニターの作成」ウィザードの以下のページのいずれか 1 つを使用します。 • Q キャプチャー・プログラムのアラート条件の選択。 • Q アプライ・プログラムのアラート条件の選択。 • キャプチャー・プログラムのアラート条件の選択。 • アプライ・プログラムのアラート条件の選択。 • サブスクリプション・セットのアラート条件の選択。

ユーザーの環境と互換性のあるしきい値を選択してください。例えば、キャプチャー・プログラムが 30 秒のコミット・インターバルで実行されている場合は、 30 秒より長いキャプチャー待ち時間のしきい値を指定してください。または、アプライ・プログラムが 10 分ごとにサブスクリプション・セットを処理するようにスケジュールする場合、APPLY_SUBSDELAY アラート条件のしきい値を 10 分より長い値に設定します。

レプリケーション・アラート・モニターのアラート条件の変更

モニターの実行中にアラート条件を変更できます。そのためには、アラート条件を変更してから、モニターを再初期化します。

手順

アラート条件を変更するには、次の方法のいずれかを使用します。

方法	説明
ASNCLP コマンド行プログラム	以下のいずれかのコマンドを使用します。 • ALTER ALERT CONDITIONS FOR APPLY • ALTER ALERT CONDITIONS FOR CAPTURE • ALTER ALERT CONDITIONS FOR Q CAPTURE • ALTER ALERT CONDITIONS FOR Q APPLY
レプリケーション・センター	Q キャプチャー・プログラム、Q アプライ・プログラム、キャプチャー・プログラム、アプライ・プログラム、またはサブスクリプション・セットの「アラート条件」ウィンドウを使用します。そのウィンドウを開くには、オブジェクト・ツリーでモニターを選択し、目次ペインでスキーマまたはサブスクリプション・セットを右クリックして、「**変更**」をクリックします。

アラート条件を変更してから、モニターを再初期化します。

アラート・モニターの中断期間の定義

レプリケーション・アラート・モニター・プログラムの中断期間を定義できます。反復的な中断期間 (例えば、毎週日曜日の午前中の 2 時間など) を作成することもできれば、1 つの期間だけモニターを中断することもできます。

このタスクについて

中断しているモニターは、すべての定義済みのアラート条件について、Q キャプチャー・サーバー、Q アプライ・サーバー、キャプチャー・コントロール・サーバー、アプライ・コントロール・サーバーのチェックを停止します。中断期間が終了すると、モニターはチェックを再開します。

反復的な中断を定義するには、*中断テンプレート* を作成します。テンプレートを作成すれば、複数のモニター対象サーバーでテンプレートを再利用できます。

テンプレートを作成しない場合は、サーバーのモニターを 1 回だけ中断するための開始日時と終了日時を指定できます。

モニターの中断に関するすべての日時は、モニターを実行しているシステムのクロックに基づきます。

制約事項

中断と中断テンプレートの定義は、ASNCLP コマンド行プログラムでのみ行えます。レプリケーション・センターでは、定義することも表示することもできません。

手順

定義済みの期間、モニターを中断するには、次のようにします。

1. オプション: ASNCLP コマンド行プログラムで CREATE MONITOR SUSPENSION TEMPLATE コマンドを使用して、反復的な中断を定義するためのテンプレートを作成します。

   例えば、毎週日曜日の 00:00:00 から 04:00:00 までモニター・プログラムを中断するテンプレートを作成するには、以下のコマンドを使用します。

   ```
   CREATE MONITOR SUSPENSION TEMPLATE SUNDAY START TIME 00:00:00
   REPEATS WEEKLY DAY OF WEEK SUNDAY FOR DURATION 4 HOURS
   ```

2. ASNCLP コマンド行プログラムで CREATE MONITOR SUSPENSION コマンドを使用して、1 回の中断の開始時点と終了時点を定義するか、中断テンプレートを使用します。

   例えば、テンプレート SUNDAY を使用してモニター・コントロール・サーバー QSRVR1 を中断する S1 という中断を作成するには、以下のコマンドを使用します。

   ```
   CREATE MONITOR SUSPENSION NAME S1 FOR SERVER QSRVR1 STARTING DATE 2006-12-10
   USING TEMPLATE SUNDAY ENDING DATE 2007-12-31
   ```

3. asnmcmd reinit コマンドを使用して、中断するモニターを再初期化します。

レプリケーション・センターの「モニターの再初期化」ウィンドウを使用することもできます。このウィンドウを開くには、目次ペインでモニター修飾子を右クリックして、「**モニターの再初期化**」をクリックします。

4. オプション: ASNCLP コマンド行プログラムで以下のいずれかのコマンドを使用して、モニターの中断または中断テンプレートをリスト、変更、またはドロップします。

コマンド	説明
LIST MONITOR SUSPENSION	モニター・コントロール・サーバー上の中断のリストを生成します。
ALTER MONITOR SUSPENSION	中断に関する以下のプロパティーを変更できます。 • 使用するテンプレート • テンプレートを使用する開始日と終了日 • モニター・プログラムを 1 回中断する開始日と終了日
DROP MONITOR SUSPENSION	モニター・コントロール表から中断を削除します。
LIST MONITOR SUSPENSION TEMPLATE	モニター・コントロール・サーバー上の中断テンプレートのリストを生成します。
ALTER MONITOR SUSPENSION TEMPLATE	中断テンプレートに定義されているモニター中断の頻度と長さを変更できます。
DROP MONITOR SUSPENSION TEMPLATE	モニター・コントロール表から中断テンプレートを削除します。

レプリケーション・アラート・モニターの操作

レプリケーション・アラート・モニターに対して、開始、停止、中断、再初期化、その他の操作を実行できます。

モニターの開始

モニターを開始するには、いくつかの方法を使用できます。モニターは、継続的に実行するか、1 モニター・サイクルだけ実行するかを決めることもできます。パラメーターの値を設定して、モニターが実行中にエラーを検出した場合に連絡する人物の E メール・アドレスを入力することもできます。

始める前に

- モニター・コントロール表とモニターを作成します。このときに、選択した連絡先とアラート条件も指定します。
- パスワード・ファイルの作成。
- モニター管理表と、モニターするプログラムが実行しているサーバーに対する許可を持っていることを確認してください。

手順

モニターを開始するには、次の方法のいずれかを使用します。

方法	説明
レプリケーション・センター	「モニターの開始」ウィンドウを使用します。このウィンドウをオープンするには、開始するモニターを示すモニター修飾子を右マウス・ボタンでクリックし、「**モニターの開始**」を選択します。
z/OS Linux UNIX Windows asnmon システム・コマンド	このコマンドを使用して、モニターを開始し、オプションで始動パラメーターを指定します。
z/OS 自動リスタート・マネージャー	ARM のリカバリー・システムをセットアップして、z/OS コンソールまたは TSO からモニターを開始します。
Windows Windowsサービス	モニターを Windows サービスとして実行するようにセットアップすることができます。

モニターの再初期化

実行中のモニターを再初期化できます。モニターを再初期化すると、連絡先、アラート条件、およびパラメーター値に加えたすべての更新が認識されます。例えば、モニターの実行時に連絡先の新規 E メール・アドレスを追加した場合は、モニターを再初期化します。

手順

モニターを再初期化するには、次の方法のいずれかを使用します。

方法	説明
asnmcmd システム・コマンド	実行中のモニターを再初期化するには、asnmcmd reinit コマンドを使用します。
レプリケーション・センター	「モニターの再初期化」ウィンドウを使用して、モニターを再初期化します。このウィンドウをオープンするには、再初期化するモニターを示すモニター修飾子を右クリックし、「**モニターの再初期化**」を選択します。

モニターの中断と再開

レプリケーション/パブリッシング環境のモニターを一時的に停止する場合は、モニターをいったん中断してから再開できます。

このタスクについて

以下のような状況では、モニターを停止して再始動する代わりに、モニターを中断して再開することを検討できます。

- モニターを停止して開始する権限を持っていない場合。

- レプリケーション環境内のサーバーがサービスの対象になっている場合。例えば、MONITOR1 というモニターが、Q キャプチャー・サーバーである SERVER_GREEN をモニターしている場合に、SERVER_GREEN をメンテナンスのために午後 4 時から午後 7 時までシャットダウンするのであれば、MONITOR1 を午後 4 時に中断し、午後 7 時に再開できます。そうすれば、MONITOR1 が QCAPTURE_STATUS アラート条件を出すことを回避できます。

キャプチャー・プログラム、アプライ・プログラム、Q キャプチャー・プログラム、Q アプライ・プログラムの実行中にモニターを中断すると、再開時には中断したところから実行を継続します。モニターを中断して再開した場合は、モニターの中断中に発生したアラート条件をチェックしたり、その条件に関するアラートを出したりすることはありません。

手順

モニターを中断して再開するには、次のようにします。

1. モニターを中断するには、asnmcmd suspend コマンドを実行します。 モニターは、アラート条件のチェックを停止します。
2. モニターを再開するには、asnmcmd resume コマンドを実行します。 モニターは、アラート条件のチェックを再開します。

モニター中断の終了

定期スケジュールによる有効期限の前にモニター中断を終了するには、モニター・コントロール表から中断を除去してから、モニターを再初期化します。

手順

モニター中断を終了するには、次のようにします。

1. ASNCLP コマンド行プログラムで DROP MONITOR SUSPENSION コマンドを使用して、モニター・サーバーのコントロール表から中断を除去します。

   例えば、以下のコマンドは、SUSP1 という名前の中断を除去します。

   ```
   DROP MONITOR SUSPENSION NAME SUSP1
   ```

2. 次の方法のいずれかを使用して、モニターを再初期化します。

方法	説明
asnmcmd コマンド	asnmcmd reinit コマンドを使用して、モニターがコントロール表から最新の変更内容を読み取るように要求します。以下のコマンドは、モニター・コントロール・サーバー wsdb で、myqual というモニター修飾子によって識別されるモニターを再初期化します。 `asnmcmd monitor_server=wsdb monitor_qual=myqual reinit`
レプリケーション・センター	「モニターの再初期化」ウィンドウを使用します。目次ペインで、再初期化するモニターを示すモニター修飾子を右クリックし、「**モニターの再初期化**」をクリックします。

注: モニターを停止し開始することによって、モニターがコントロール表を読み取るように要求することもできます。

モニターの停止

モニターを停止すると、アラート条件の対象のレプリケーション・プログラムまたは公開プログラムのチェックは停止します。モニターを停止するには、レプリケーション・センター、システム・コマンド、または DB2 レプリケーション・サービスを使用することができます。

このタスクについて

キャプチャー、アプライ、Q キャプチャー、または Q アプライ・プログラムの実行中にモニターが停止した場合、次回の開始時にモニターは以下のアクションを実行します。

- モニターが停止中に合致していたアラート条件をチェックする。
- 合致していたすべての条件に対してアラートを発行する。

手順

モニターを停止するには、次の方法のいずれかを使用します。

方法	説明
asnmcmd stop コマンド	このコマンドを使用して、モニターを停止します。
レプリケーション・センター	「モニターの停止」ウィンドウを使用して、モニターを停止します。このウィンドウをオープンするには、停止するモニターを示すモニター修飾子を右クリックし、 「**モニターの停止**」を選択します。
Windows Service Control Manager	DB2レプリケーション・サービスを停止します。レプリケーション・サービスの停止時にモニターは自動的に停止します。

モニター・プログラムのメッセージの検討

指定された一定期間内に IBMSNAP_MONTRACE 表に挿入されたメッセージを検討するには、「モニター・メッセージ」ウィンドウを使用します。
IBMSNAP_MONTRACE 表には、モニター・プログラムから発行された、アクション、警告、およびエラーなどの重要なイベントの行が入ります。

例えば、「モニター・メッセージ」ウィンドウでは、1 週間の間にモニター・プログラムから記録されたすべてのエラー・メッセージと警告メッセージを検討できます。「モニター・メッセージ」ウィンドウから、ファイルにデータを出力したり保管したりすることもできます。

レプリケーション・アラート・モニターのパラメーター

さまざまなパラメーターの値を設定して、レプリケーション・アラート・モニターの動作を決定できます。

レプリケーション・アラート・モニターのパラメーターのデフォルト値

レプリケーション管理ツールを使用してモニター・コントロール表を作成すると、モニター操作パラメーターのデフォルト値が設定されます。

各パラメーターのデフォルト値を表 16にまとめます。

表 16. レプリケーション・アラート・モニター操作パラメーターのデフォルト値

稼働パラメーター	デフォルト値
alert_prune_limit	10080 分
autoprune	Y
email_server	デフォルト値なし
max_notification_minutes	60 分
max_notifications_per_alert	3
monitor_errors	デフォルト値なし
monitor_interval	300 秒
monitor_limit	10080 分
monitor_path	asnmon コマンドが呼び出されたディレクトリーです。
runonce	N
trace_limit	10080 分

レプリケーション・アラート・モニターのパラメーターの説明

このトピックでは、レプリケーション・アラート・モニターの操作に使用できる以下のパラメーターについて説明します。

- 『alert_prune_limit』
- 244 ページの『autoprune』
- 244 ページの『email_server』
- 244 ページの『max_notification_minutes』
- 244 ページの『max_notifications_per_alert』
- 245 ページの『monitor_errors』
- 245 ページの『monitor_interval』
- 245 ページの『monitor_limit』
- 245 ページの『monitor_path』
- 245 ページの『runonce』
- 246 ページの『trace_limit』

alert_prune_limit

デフォルト: alert_prune_limit=10080 分 (7 日)

新規のモニター・サイクルの開始時に、レプリケーション・アラート・モニターは IBMSNAP_ALERTS 表から整理の対象となる行を整理します。デフォルトでは、レ

プリケーション・アラート・モニターは、10080 分 (7 日) よりも古い行を削除します。**alert_prune_limit** パラメーターは、レプリケーション・アラート・モニターが表に保管する古いデータの量を制御します。このパラメーターは、データがどの程度古くなったらレプリケーション・アラート・モニターによって整理されるかを指定します。

システム上の IBMSNAP_ALERTS 表のストレージ・スペースが小さいときには、**alert_prune_limit** パラメーターの値を小さくできます。除去の制限を下げるとスペースは節約できますが、処理コストが増加します。また、**alert_prune_limit** パラメーターの値を大きくして、すべてのアラート・アクティビティーの履歴を保持しておくことも考えられます。 SQL レプリケーションに限り、整理の制限が大きいほど、大きな変更データ (CD) 表と UOW 表のスペースが必要になりますが、処理コストは小さくなります。

autoprune

デフォルト: autoprune=y

autoprune パラメーターは、自動整理を制御します。レプリケーション・アラート・モニターは、モニター・コントロール表にコピー済みの行は、IBMSNAP_ALERTS 表から自動的に整理します。

email_server

email_server パラメーターは、ASNMAIL 出口ルーチンを使用できるようにします。デフォルトの ASNMAIL ルーチンを使用すると、レプリケーション・アラート・モニターが E メールを使用してアラートを送信できます。 Simple Mail Transfer Protocol (SMTP) を使用するよう設定されている E メール・サーバーのアドレスに、このパラメーターの値を設定してください。

max_notification_minutes

デフォルト: max_notifications_minutes=60

max_notifications_minutes パラメーターは、アラート条件が複数回発生したかどうかをモニターがトラッキングする期間を指定します。デフォルトでは、60 分の間にアラート条件が複数回発生した場合、レプリケーション・アラート・モニターは、60 分の間に最大 3 つのアラートを送信します。 **max_notifications_per_alert** パラメーターは、**max_notifications_minutes** パラメーターによって指定された時間内に送信するアラート条件の通知の数を、モニターに指示します。

max_notifications_per_alert

デフォルト: max_notifications_per_alert=3

max_notifications_per_alert パラメーターは、 1 つのアラートに関する通知の最大送信数を、レプリケーション・アラート・モニターに指示します。デフォルトでは、レプリケーション・アラート・モニターがアラート条件を複数回受信した場合、 60 分の間にそのアラート条件に関して最大 3 つの通知を送信します。

monitor_errors

レプリケーション・アラート・モニターは、モニター処理で発生するすべてのエラーを保管します。操作エラーの一例として、レプリケーション・アラート・モニターがモニター・コントロール・サーバーに接続できない場合があります。操作エラーの通知を受信する場合は、**monitor_errors** パラメーターに E メール・アドレスを指定しなければなりません。 E メール・アドレスを指定しないと、レプリケーション・アラート・モニターは操作エラーをログに記録しますが、エラーの通知を送信しません。

email_server パラメーターに有効な E メール・サーバーが記述されていない場合は、レプリケーション・アラート・モニターは **monitor_errors** パラメーターを無視します。

monitor_interval

デフォルト: monitor_interval=300 秒 (5 分)

monitor_interval パラメーターは、アラート条件をチェックする頻度をレプリケーション・アラート・モニターに指示します。デフォルトでは、レプリケーション・アラート・モニターは 300 秒ごとに、サーバー上の特定のモニターについて、すべてのアラート条件をチェックします。

monitor_limit

デフォルト: monitor_limit=10080 分 (7 日)

Q レプリケーションの場合、**monitor_limit** パラメーターは、 Q キャプチャー・プログラムが行を整理する前に、 IBMQREP_CAPMON 表と IBMQREP_CAPQMON 表に行を保持する期間を指定します。 SQL レプリケーションの場合、**monitor_limit** パラメーターは、 Q キャプチャー・プログラムが行を整理する前に、 IBMSNAP_CAPMON 表に行を保持する期間を指定します。整理インターバルのたびに、現行タイム・スタンプに基づいて行がこの制限より古い場合は、キャプチャー・プログラムと Q キャプチャー・プログラムはこれらの表中の行を整理します。

monitor_path

デフォルト: monitor_path=asnmon コマンドが呼び出されたディレクトリー

monitor_path パラメーターは、レプリケーション・アラート・モニターが使用するログ・ファイルのロケーションを指定します。

runonce

デフォルト: runonce=n

レプリケーション・アラート・モニターは開始されると、デフォルトとして、ユーザーから選択されたアラート条件をモニターするインターバルで実行されます。レプリケーション・アラート・モニターを 1 時間おきに実行したり、その他の時間間隔で実行したり、一度だけ実行するようにスケジュールすることもできます。

runonce=y が指定されると、レプリケーション・アラート・モニターはユーザーから選択されたすべてのアラート条件を一度チェックし、 **monitor_interval** パラメーターを無視します。バッチ処理中でレプリケーション・アラート・モニターを実行するときに **runonce**を使用できます。例えば、アプライ・プログラムが完了した後で、**runonce=y** を使用して、失敗したサブスクリプション・セットがあるかどうかを確認できます。失敗したサブスクリプション・セットがある場合には、レプリケーション・アラート・モニターは連絡先の個人またはグループに通知を送信します。

デフォルトでは **monitor_interval**は 300 秒 (5 分) です。レプリケーション・アラート・モニターは 300 秒ごとに、サーバー上の特定のモニターごとに、すべてのアラート条件をチェックします。レプリケーション・アラート・モニターはアラート条件を検出すると、通知を送信します。

trace_limit

デフォルト: trace_limit=10080 分 (7 日)

trace_limit パラメーターは、 IBMSNAP_MONTRACE 表と IBMSNAP_MONTRAIL 表を整理する頻度を、レプリケーション・アラート・モニターに指示します。レプリケーション・アラート・モニターは、 10080 分 (7 日) 間これらの表に行を保管します。レプリケーション・アラート・モニターは、**trace_limit** パラメーターに指定された値より古い行を整理します。

レプリケーション・アラート・モニターのランタイム・パラメーターの変更

レプリケーション・アラート・モニターのランタイム・パラメーターは、モニターの開始時またはモニターの実行中に変更できます。

このタスクについて

初期パラメーター値は、モニターの作成時に設定します。これらの値は、IBMSNAP_MONPARMS コントロール表に格納されます。モニターを開始すると、モニターはそのコントロール表を読み取り、そのパラメーター値を使用します。

その保管済みの値は、実行時に (モニターの開始時またはモニターの実行中に) オーバーライドできます。実行時に設定した値は、その現在の実行でしか継続しません。モニターを停止して再始動すると、モニターは、コントロール表に保管されている値を使用します。

手順

1. モニターの開始時にパラメーターを変更します。次の方法のいずれかを使用します。

方法	説明
asnmon システム・コマンド	このコマンドを使用してモニターを開始するときに、1 つ以上のパラメーターと値を指定します。

方法	説明
レプリケーション・センター	「モニター開始パラメーターの指定 (Specify Monitor Startup Parameters)」ウィンドウを使用します。このウィンドウを開くには、目次ペインで、開始するモニターを示すモニター修飾子を右クリックして、「**モニターの開始**」をクリックします。「モニターの開始」ウィンドウで「**パラメーター**」をクリックします。

2. asnmcmd chgparms システム・コマンドを使用すれば、モニターの実行中にパラメーターを変更できます。 変更できるパラメーターは、以下のとおりです。
   - **monitor_interval**
   - **autoprune**
   - **alert_prune_limit**
   - **trace_limit**
   - **max_notifications_per_alert**
   - **max_notifications_minutes**

レプリケーション・アラート・モニターの実行頻度の指定

レプリケーション・アラート・モニターがレプリケーション環境のアラート条件をチェックする頻度を決定しなければなりません。

手順

レプリケーション・アラート・モニターの実行頻度を指定するには、次の方法を使用します。

- asnmon コマンドの **runonce** パラメーターを使用して、レプリケーション・アラート・モニターを繰り返し実行するか、それとも 1 回のみ実行するかを指定します。
- asnmon コマンドの **monitor_interval** パラメーターを使用して、**runonce=**n の場合のレプリケーション・アラート・モニターの実行頻度を指定します。
- レプリケーション・センターを使用して、レプリケーション・アラート・モニターを開始する実行時間を指定することもできます。

選択されたアラート条件の通知基準の指定

選択したアラート条件は、レプリケーション・アラート・モニターによって保管されます。電子メール (E メール) によって自動的にアラート条件を連絡先に通知するように、通知パラメーターをセットアップできます。

手順

アラート条件の通知基準を指定するには、次の方法を使用します。

- **max_notifications_per_alert** パラメーターを設定して、特定の時間内の最大通知数を制御します。**max_notifications_minutes** パラメーターで指定された時間内の特定のアラート条件に関する、受け取りたい通知の最大数を指定します。
- **email_server** パラメーターを設定して、アラート条件が生じた時点で DB2 が E メール通知できるようにします。このパラメーターの値は、SMTP プロトコルを使用する E メール・サーバーのアドレスに設定してください。

- オプション: ASNMAIL 出口ルーチンに独自の拡張機能を作成して、アラート条件の処理方法をカスタマイズできます。このオプションは、問題管理やその他のシステムと統合するのに便利です。

操作エラーの通知基準の指定

操作中にエラーが発生すると、レプリケーション・アラート・モニターは通知を送信します。

手順

操作エラーの通知基準を指定するには、**monitor_errors** パラメーターの値として E メール・アドレスを設定します。モニターは、発生した操作エラーの通知をこのアドレスに送信します。 Simple Mail Transfer Protocol (SMTP) プロトコルを使用して、 E メール・アドレスを入力してください。

レプリケーション・アラート・モニターからのデータの整理インターバルの指定

レプリケーション・アラート・モニターは、モニター・コントロール表を自動的に整理できます。モニターが表を自動的に整理するかどうか、および整理する場合はモニターが表を整理する方法を決定しなければなりません。

手順

モニター表を整理する頻度を指定するには、次の方法を使用します。

- **autoprune** パラメーターを使用して、レプリケーション・アラート・モニターにコントロール表を自動的に整理させたいかどうかを指定します。
- **alert_prune_limit** パラメーターの値を変更して、レプリケーション・アラート・モニターが表に保管する履歴データの量を制御します。データがどの程度古くなったらレプリケーション・アラート・モニターによって IBMSNAP_ALERTS 表から整理されるかを指定します。
- **trace_limit** パラメーターの値を変更して、レプリケーション・アラート・モニターが行をモニター表に保管する期間を制御します。

第 16 章 レプリケーション・サービス (Windows)

Windows Service Control Manager (SCM) を使用することにより、Windows オペレーティング・システム上でレプリケーション・プログラムをシステム・サービスとして実行できます。

レプリケーションのための Windows サービスの説明

Windowsオペレーティング・システムでは、Q キャプチャー・プログラム、Q アプライ・プログラム、キャプチャー・プログラム、アプライ・プログラム、レプリケーション・アラート・モニター・プログラムを開始したり停止したりするプログラムとしてのレプリケーション・サービスがあります。

ユーザーが作成したレプリケーション・サービスは、自動モードで SCM に追加され、サービスが開始されます。Windows は固有のサービス名および表示名を使用してサービスを登録します。

以下の用語は、レプリケーション・サービスの命名規則を説明しています。

レプリケーション・サービス名

レプリケーション・サービス名は、各サービスを一意的に識別し、サービスを停止または開始するときにはこれを使用します。名前は以下のようなフォーマットになります。

`DB2.instance.alias.program.qualifier_or_schema`

表 17 では、レプリケーション・サービス名の入力について説明しています。

表 17. レプリケーション・サービス名の入力

入力	説明
instance	DB2 インスタンスの名前。
alias	Q キャプチャー・サーバー、Q アプライ・サーバー、キャプチャー・コントロール・サーバー、アプライ・コントロール・サーバー、モニター・コントロール・サーバーのデータベース別名。
program	以下の値のいずれか: QCAP (Q キャプチャー・プログラムの場合)、QAPP (Q アプライ・プログラムの場合)、CAP (キャプチャー・プログラムの場合)、APP (アプライ・プログラムの場合)、MON (レプリケーション・アラート・モニター・プログラムの場合)。
qualifier_or_schema	以下の ID のいずれか: Q キャプチャー・スキーマ、Q アプライ・スキーマ、キャプチャー・スキーマ、アプライ修飾子、モニター修飾子。

例: 以下のサービス名は、スキーマ ASN を持ち、INST1 という名前のインスタンスの下でデータベース DB1 を処理している Q アプライ・プログラムのものです。

```
DB2.INST1.DB1.QAPP.ASN
```

レプリケーション・サービスの表示名

表示名は、「サービス」ウィンドウで表示されるテキスト・ストリングで、読みやすい形式のサービス名です。例:

```
DB2 - INST1 DB1 QAPPLY ASN
```

サービスの記述を追加する場合は、レプリケーション・サービスを作成した後で、Service Control Manager (SCM) を使用してください。また、SCM を使用し、サービスのユーザー名およびパスワードを指定することもできます。

レプリケーション・サービスの作成

Windows オペレーティング・システム上で Q キャプチャー・プログラム、Q アプライ・プログラム、キャプチャー・プログラム、アプライ・プログラム、およびレプリケーション・アラート・モニター・プログラムを開始するために、DB2 レプリケーション・サービスを作成できます。

始める前に

レプリケーション・サービスを作成する前に、DB2 インスタンス・サービスが実行されていることを確認してください。レプリケーション・サービスを作成するときに DB2 インスタンス・サービスが実行されていないと、レプリケーション・サービスは作成されますが、自動的に開始されなくなります。

このタスクについて

サービスを作成するときには、Windows にログオンするために使用するアカウント名と、そのアカウント名のパスワードを指定する必要があります。

システムには複数のレプリケーション・サービスを追加できます。すべての Q キャプチャー・サーバー、Q アプライ・サーバー、またはキャプチャー・コントロール・サーバーのスキーマごとに 1 つのサービスを追加し、すべてのアプライ・コントロール・サーバーおよびモニター・コントロール・サーバーの修飾子ごとに 1 つのサービスを追加できます。 例えば、5 つのデータベースがあり、それぞれのデータベースが Q アプライ・コントロール・サーバーとモニター・コントロール・サーバーを兼ねている場合は、10 個のレプリケーション・サービスを作成できます。各サーバーに複数のスキーマまたは修飾子がある場合は、さらに多くのサービスを作成できます。

手順

レプリケーション・サービスを作成するには、以下のようにします。

asnscrt コマンドを使用します。

サービスを作成するときには、Windows にログオンするために使用するアカウント名と、そのアカウント名のパスワードを指定する必要があります。

ヒント: レプリケーション・サービスが正しくセットアップされた場合は、サービスが正常に開始された後、サービス名が STDOUT に送信されます。サービスが開始されない場合は、開始しようとしているプログラムのログ・ファイルを調べてください。デフォルトでは、ログ・ファイルは DB2PATH 環境変数で指定されたディレクトリー内にあります。このデフォルトは、サービスとして開始するプログラムのパス・パラメーター (**capture_path**、**apply_path**、**monitor_path**) を指定することによってオーバーライドできます。 また、Windows Service Control Manager (SCM) を使用して、サービスの状況を表示できます。

レプリケーション・サービスの開始

レプリケーション・サービスを作成したら、サービスを停止して、再度開始できます。

このタスクについて

重要: サービスからレプリケーション・プログラムを始動した場合、同じスキーマまたは修飾子を使用してプログラムを始動しようとすると、エラーが発生します。

手順

レプリケーション・サービスを開始するには、次のようにします。

次の方法のいずれかを使用します。

- Windows Service Control Manager (SCM)
- net stop コマンド

レプリケーション・サービスの停止

レプリケーション・サービスを作成したら、サービスを停止して、再度開始できます。

このタスクについて

レプリケーション・サービスを停止すると、サービスに関連するプログラムは自動的に停止します。しかし、レプリケーション・システム・コマンド (asnqacmd、asnqccmd、asnccmd、asnacmd、asnmcmd) を使用してプログラムを停止する場合、プログラムを開始したサービスは実行を続けます。これは明示的に停止しなければなりません。

手順

レプリケーション・サービスを停止するには、次のようにします。

次の方法のいずれかを使用します。

- Windows Service Control Manager (SCM)
- net stop コマンド

レプリケーション・サービスのリストの表示

asnlist コマンドを使用して、すべてのレプリケーション・サービスとそのプロパティーのリストを表示できます。

手順

レプリケーション・サービスのリストを表示するには、次のようにします。

1. asnlist コマンドを使用します。
2. オプション: asnlist コマンドで **details** パラメーターを使用すれば、レプリケーション・サービスのリストと各サービスの説明を表示できます。

レプリケーション・サービスのドロップ

レプリケーション・サービスが必要なくなった場合は、Windows Service Control Manager (SCM) から除去されるように、サービスをドロップできます。

このタスクについて

サービスによって開始されるプログラムの始動パラメーターを変更する場合は、サービスをドロップして、新しい始動パラメーターを使用する新しいサービスを作成する必要があります。

手順

レプリケーション・コマンドのサービスをドロップするには、asnsdrop コマンドを使用します。

第 17 章 各種オペレーティング・システムでの SQL レプリケーション・プログラムのスケジューリング

オペレーティング・システム (OS) のコマンドを使用してあらかじめ指定しておいた時刻にキャプチャー・プログラム、アプライ・プログラム、またはレプリケーション・アラート・モニター・プログラムが始動するようにスケジューリングできます。

Linux および UNIX オペレーティング・システムでのプログラムのスケジューリング

Linux、UNIX の各オペレーティング・システムでは、レプリケーション・プログラムを始動するスケジュールを設定できます。

手順

Linux と UNIX の場合にレプリケーション・プログラムのスケジュールを設定するには、以下のようにします。

レプリケーション・プログラムを特定の時刻に開始するには **at** コマンドを使用します。 表 18は、金曜日の午後 3:00 にレプリケーション・プログラムを始動するためのコマンドを示しています。

表 18. レプリケーション・プログラムのスケジューリング・コマンド (Linux、UNIX)

レプリケーション・プログラム	Linux または UNIX のコマンド
キャプチャー	`at 3pm Friday asncap autoprune=n`
アプライ	`at 3pm Friday asnapply applyqual=myqual`
レプリケーション・アラート・モニター	`at 3pm Friday asnmon monitor_server=db2srv1 monitor_qualifier=mymon`

Windows オペレーティング・システムでのプログラムのスケジューリング

Windows オペレーティング・システムでは、レプリケーション・プログラムを始動するスケジュールを設定できます。

手順

Windows Service Control Manager を使用していない場合は、AT コマンドを使用して、プログラムを特定の時刻に始動できます。 AT コマンドを入力する前に、Windows スケジュール・サービスを始動してください。 254 ページの表 19 は、金曜日の午後 3:00 にレプリケーション・プログラムを始動するためのコマンドを示しています。

表 19. レプリケーション・プログラムのスケジューリング・コマンド (Windows)

レプリケーション・プログラム	Windows のコマンド
キャプチャー	c:¥>at 15:00/ interactive"c:¥SQLLIB¥BIN¥db2cmd.exe c:¥CAPTURE¥asncap.exe"
アプライ	c:¥>AT 15:00 /interactive "c:¥SQLLIB¥BIN¥db2cmd.exe c:¥SQLLIB¥BIN¥asnapply.exe control_server=cntldb apply_qual=qualid1"
レプリケーション・アラート・モニター	c:¥>AT 15:00 /interactive "c:¥SQLLIB¥BIN¥db2cmd.exe c:¥CAPTURE¥asnmon.exe monitor_server=db2srv1 monitor_qualifier=mymon"

z/OS オペレーティング・システムでのプログラムのスケジューリング

z/OS オペレーティング・システム上のレプリケーション・プログラムをいつ開始するかを、以下の 2 種類のコマンドを使用してスケジュールできます。

手順

z/OS オペレーティング・システム上のプログラムをスケジュールするには、以下の方法を使用します。

1. z/OS 用のプログラムを呼び出すプロシージャーを PROCLIB に作成します。
2. ICHRIN03 RACF モジュール (または、MVS セキュリティー・パッケージの該当する定義) を修正し、プロシージャーをユーザー ID に関連付けます。
3. SYS1.LPALIB でモジュールをリンク・エディットします。
4. 特定の時点でキャプチャー・プログラムまたはアプライ・プログラムを始動するには、$TA JES2 コマンドまたは AT NetView コマンドを使用します。 $TA JES2 コマンドの使用法については、「*MVS/ESA JES2 コマンド*」を参照してください。AT NetView コマンドの使用については、「*NetView (MVS) コマンド・リファレンス*」を参照してください。

System i オペレーティング・システムでのプログラムのスケジューリング

System i オペレーティング・システムでは、レプリケーション・プログラムを始動するスケジュールを設定できます。

手順

1. アプライ・プログラムを始動する場合は、ADDJOBSCDE コマンドを実行します。
2. キャプチャー・プログラムを始動する場合は、SBMJOB コマンドを実行します。 以下に例を示します。

```
SBMJOB CMD('STRDPRCAP...')SCDDATE(...)SCDTIME(...)
```

第 18 章 SQL レプリケーション・プログラムに関するレポートの表示

以下のトピックでは、レプリケーション環境に関するレポートを生成して分析するための方法を説明します。その情報を活用して、レプリケーション・プログラムの現在の状況をチェックしたり、履歴データを検討して最近のメッセージやスループットや待ち時間の統計を確認したりできます。

レプリケーション・プログラムの状況のチェック (z/OS、Linux、UNIX、Windows)

キャプチャー・プログラム、アプライ・プログラム、またはレプリケーション・アラート・モニターの現在の状況は簡単に評価できます。

レプリケーション・プログラムの状況をチェックするには、次のコマンドのいずれかを使用します。

キャプチャー・プログラム
asnccmd システム・コマンド、**status** パラメーター

アプライ・プログラム
asnacmd システム・コマンド、**status** パラメーター

レプリケーション・アラート・モニター
asnmcmd システム・コマンド、**status** パラメーター

プログラムの状況を照会すると、そのプログラムに関連する各スレッドの状態を記述するメッセージが戻されます。

- キャプチャー・プログラムには、以下のスレッドがあります。
  - ワーカー・スレッド
  - 管理スレッド
  - 整理スレッド
  - シリアライゼーション・スレッド
  - トランザクション読み取りスレッド (**asynchlogrd** 開始パラメーターが yes に設定されている場合)
- アプライ・プログラムには、以下のスレッドがあります。
  - 管理スレッド
  - ワーカー・スレッド
  - シリアライゼーション・スレッド
- レプリケーション・アラート・モニター・プログラムには、以下の 3 つのスレッドがあります。
  - 管理スレッド
  - ワーカー・スレッド
  - シリアライゼーション・スレッド

受け取ったメッセージに基づいて、プログラムが正しく作動しているかどうかを確認できます。通常、ワーカー・スレッド、管理スレッド、整理スレッドは作動状態にあり、計画どおりのタスクを実行しています。シリアライゼーション・スレッド (グローバル・シグナル・ハンドラー) は、通常は待機状態にあり、シグナルを待機しています。整理スレッドは、CD 表、および次のレプリケーション・コントロール表を整理します。

- IBMSNAP_UOW 表
- IBMSNAP_CAPTRACE 表
- IBMSNAP_CAPMON 表
- IBMSNAP_SIGNAL 表

受け取ったメッセージではプログラムが機能していることが示されているのに、環境はその逆の証拠を示しているという場合には、さらに調査が必要です。例えば、アプライ・プログラムの状況を照会した結果、ワーカー・スレッドが作動中であることが判明しているのに、データが期待どおりにターゲット表にアプライされていない場合には、IBMSNAP_APPLYTRAIL 表の中で、データがアプライされない理由を示しているメッセージを探す必要があります。システム・リソースの問題が原因で、プログラムが正しく作動しない可能性もあります。

履歴データの傾向を検討

最近のレプリケーション操作からの履歴データを検討して、データの傾向を評価できます。一定期間にわたり認識される傾向から、安定した量のデータが複製されていることや、パフォーマンスを向上させるために調整の余地があることなどを確認できます。

最近のレプリケーション操作からの履歴データを検討して、データの傾向を評価できます。一定期間にわたり認識される傾向から、安定した量のデータが複製されていることや、パフォーマンスを向上させるために調整の余地があることなどを確認できます。

履歴データは、以下のコントロール表から派生します。

- IBMSNAP_APPLYTRAIL
- IBMSNAP_APPLYTRACE
- IBMSNAP_CAPMON
- IBMSNAP_CAPTRACE

これらの表の整理の頻度により、生成できるレポートが異なります。トラブルシューティングやパフォーマンス評価のためにデータを調べる目的で、これらの表には少なくとも 1 週間分のデータを保存しておくことをお勧めします。

表 20 は、ユーザーが表示できる履歴データを示しています。

表 20. 履歴情報の保管場所

質問:	**レプリケーション・センターで使用するウィンドウ:**
キャプチャー・プログラム、アプライ・プログラム、およびモニター・プログラムからの最新のメッセージはどのようなものか ?	キャプチャー・メッセージ アプライ・メッセージ モニター・メッセージ
平均で、 • 一定期間中に CD 表で処理された行数は ? • 整理される行数は ? • コミットされるトランザクション数は ? • キャプチャー・プログラムが使用するメモリー量は ?	キャプチャー・スループット分析
ソースでデータが更新されてから、キャプチャー・プログラムによってキャプチャーされるまでの平均時間はどれくらいか ?	キャプチャー待ち時間
アプライ・プログラムからの最新のメッセージはどのようなものか ?	アプライ・レポート
平均で、 • 一定期間中にターゲット表で処理された行数は ? • サブスクリプション・セットの処理の経過時間は ?	アプライ・スループット分析
ソース表が更新されてから、対応するターゲット表が更新されるまでの平均の経過時間はどれくらいか ?	エンドツーエンド待ち時間

分析対象とするデータ量を識別するために、時間の範囲を選択できます。時間範囲の開始と終了の両方の日時を指定してから、算出結果の平均値として結果を表示するように指定します。時間インターバル (1 秒、1 分、1 時間、1 日、または 1 週間) を選択して、結果をグループ化することもできます。例えば、9:00 p.m. から 9:59 p.m. までのアプライ・プログラムのスループットを分析するように選択した場合、データを 1 分インターバルで表示すると、結果は 60 行で表示され、それぞれの行に、60 分の範囲の間の各 1 分間のアクティビティーの要約が表示されます。また、1 時間というインターバルを選択した場合は、結果は 1 行に表示され、指定された 1 時間という期間の平均スループットが示されます。ユーザーがインターバルを指定しない場合は、 IBMSNAP_APPLYTRAIL 表のロー・データが表示されます。

レプリケーション・センターのウィンドウは、さまざまなコントロール表およびログ・ファイルに含まれた情報から結果を表示します。以下のトピックでは、レプリケーション・センターで、履歴データを使用してレプリケーション操作を評価する方法を説明しています。

キャプチャー・プログラムのメッセージの検討

特定の期間内に IBMSNAP_CAPTRACE 表に挿入されたメッセージを確認するには、「キャプチャー・メッセージ」ウィンドウを使用します。
IBMSNAP_CAPTRACE 表には、キャプチャー・プログラムから発行された、初期化、整理、警告、およびエラーなどの重要なイベントの行が入ります。

例えば、「キャプチャー・メッセージ」ウィンドウでは、1 週間の間にキャプチャー・プログラムによって記録されたすべてのエラー・メッセージと警告メッセージを確認できます。また、「キャプチャー・メッセージ」ウィンドウから、ファイルにデータを印刷または保管することもできます。

キャプチャー・プログラムのスループットの検査

特定の期間内のキャプチャー・プログラムのパフォーマンス結果を表示するには、「キャプチャー・スループット分析」ウィンドウを使用します。キャプチャー・プログラムは統計情報を定期的に IBMSNAP_CAPMON 表に記録し、整理時には、整理統計を IBMSNAP_CAPTRACE 表に記録します。

「キャプチャー・スループット分析」ウィンドウはこれらの表の情報を使用して、4 つの異なるタスクのパフォーマンス率の計算結果を表示します。 4 つのタイプの情報の結果をすべて使用して、キャプチャー・プログラムのスループット・パフォーマンスを検査できます。以下の結果を絶対値で表示するか平均値で表示するかを指定するためのオプションもあります。

- ログで挿入された、またはスキップされた行数
- CD 表で整理された行数
- コミットされたトランザクション数
- メモリー使用

例えば、「キャプチャー・スループット分析」ウィンドウでは、キャプチャー・プログラムのスループットの 1 週間の平均パフォーマンスを確認できます。そのためには、時刻範囲の開始と終了の両方の日時を指定してから、算出結果の平均値として結果を表示するように指定します。

キャプチャー・プログラムによって処理されるデータの待ち時間の表示

ソースで特定のデータが更新されてから、キャプチャー・プログラムによってキャプチャーされるまでのおおよその時間を表示するには、「キャプチャー待ち時間」ウィンドウを使用します。この経過時間は、CD 表の中のデータの新しさをある程度示しています。この平均待ち時間は IBMSNAP_CAPMON 表の情報から派生し、その表の情報は IBMSNAP_REGISTER 表から派生します。

現行のキャプチャー待ち時間は、IBMSNAP_REGISTER 表の中の SYNCHTIME 列の CURRENT_TIMESTAMP 値とグローバル・レコードとの差として計算できます。

```
(CURRENT_TIMESTAMP) - (SYNCHTIME)
```

表 21. 現行キャプチャー待ち時間を計算するための値の例

パラメーター	列値
CURRENT_TIMESTAMP	2006–10–20–10:30:25
SYNCHTIME	2006–10–20–10:30:00

例えば、表 21 の値を使用すると、現行待ち時間は 25 秒になります。

```
10:30:25 - 10:30:00 = 25
```

キャプチャー待ち時間は時間とともに変化し、これらの変更の履歴は IBMSNAP_CAPMON 表に保管されます。レプリケーション・センターもキャプチャー・モニター表の中の情報を使用して、平均待ち時間または履歴待ち時間を計算します。その場合の数式では、平均待ち時間を計算するために CURRENT_TIMESTAMP 値ではなく MONITOR_TIME 値を使用するので、現行の待ち時間を計算する数式とはその点が異なります。 MONITOR_TIME 値は、キャプチャー・プログラムによってキャプチャー・モニター表に行が挿入された時間を示すタイム・スタンプです。平均待ち時間は、秒、分、時間、日、または週で表示できます。例えば、「キャプチャー待ち時間」ウィンドウで、キャプチャー・プログラムの過去 1 週間の平均待ち時間を 1 時間単位で表示できます。

アプライ・プログラムのメッセージの検討

特定の期間内に IBMSNAP_APPLYTRACE 表に挿入されたメッセージを確認するには、「アプライ・メッセージ」ウィンドウを使用します。IBMSNAP_APPLYTRACE 表には、アプライ・プログラムから発行された可能性がある、初期化、警告、エラーなどの重要なイベントの行が入ります。

例えば、「アプライ・メッセージ」ウィンドウでは、1 週間の間にアプライ・プログラムによって記録された可能性があるすべてのエラー・メッセージと警告メッセージを確認できます。また、「アプライ・メッセージ」ウィンドウから、そのデータをファイルに出力したり保管したりすることもできます。

「アプライ・レポート」ウィンドウを使用して、IBMSNAP_APPLYTRAIL 表に挿入されたデータを検討して、一定期間について、特定のアプライ・プログラムが成功しているかどうかをチェックします。 IBMSNAP_APPLYTRAIL 表には、サブスクリプション・セットの実行に関するデータとして、サブスクリプション・セットの状況、エラー・メッセージ、処理された行数などが含まれています。

「アプライ・レポート」ウィンドウでは次のデータを表示できます。

- すべてのサブスクリプション・セット
- 失敗したサブスクリプション・セット
- 成功したサブスクリプション・セット
- 失敗したサブスクリプション・セットのエラーの要約

例えば、「アプライ・レポート」ウィンドウでは、アプライ・プログラムが先週サブスクリプション・セットを正常に処理したかどうかを確認できます。複製できなかったサブスクリプション・セットに関してアプライ・プログラムから出されたエラー・メッセージを表示できます。「アプライ・レポート」ウィンドウと「アプライ・スループット分析」ウィンドウを併用することも可能です。「アプライ・レポ

ート」ウィンドウでどのセットが正常に複製されたかを確認した後、「アプライ・スループット分析」ウィンドウを使用して、複製された行数と、レプリケーションに要した時間を確認できます。

「アプライ・レポート」ウィンドウを使用して、IBMSNAP_APPLYTRAIL 表で、特定の行からのすべてのデータを表示することもできます。

アプライ・プログラムのスループットの検査

特定のアプライ修飾子のパフォーマンス統計を検査するには「アプライ・スループット分析」ウィンドウを使用します。SQL ステートメントを作成しなくても、データをフィルターに掛け、グループ化できます。

特定のアプライ修飾子のパフォーマンス統計を検査するには「アプライ・スループット分析」ウィンドウを使用します。SQL ステートメントを作成しなくても、データをフィルターに掛け、グループ化できます。

例えば、特定のアプライ修飾子により処理されたサブスクリプション・セット内のターゲット表で挿入、更新、削除、および再処理された行数を表示できます。また、特定のアプライ修飾子について、アプライ・プログラムがサブスクリプション・セットの処理に費やした時間の長さも確認できます。

トランザクションの複製に要した平均時間の表示

特定のサブスクリプション・セット内のトランザクションの複製に使用された平均時間のおおよその値を表示するには、「エンドツーエンド待ち時間」ウィンドウを使用します。

「エンドツーエンド待ち時間」ウィンドウでは、例えば、一定期間内のアプライ・サイクルごとに、サブスクリプション・セットのおおよその待ち時間を表示できます。また、時間をインターバルで分けて、インターバルごとの平均待ち時間を表示することもできます。

レプリケーション・センターでは次の公式を使用してエンドツーエンド待ち時間を計算します。

```
(ENDTIME - LASTRUN) + (SOURCE_CONN_TIME - SYNCHTIME)
```

- ENDTIME は、アプライ・プログラムがサブスクリプション・セットの処理を終了した時間です。
- LASTRUN は、アプライ・プログラムがサブスクリプション・セットの処理を開始した時間です。
- SOURCE_CONN_TIME は、アプライ・プログラムがデータをフェッチするためにキャプチャー・コントロール・サーバーに接続した時間です。
- SYNCHTIME は、キャプチャー・プログラムによる、CD 表へのデータのコミットの最新の時刻です。

表 22. エンドツーエンド待ち時間の値の計算の例

パラメーター	列値
ENDTIME	2006–10–20–10:01:00
LASTRUN	2006–10–20–10:00:30

表 22. エンドツーエンド待ち時間の値の計算の例 (続き)

パラメーター	列値
SOURCE_CONN_TIME	2006–10–20–10:00:32
SYNCHTIME	2006–10–20–10:00:00

例えば、あるサブスクリプション・セットに、 260 ページの表 22 に示すような値があるとします。前述の式を使用すると、このサブスクリプション・セットのエンドツーエンドの平均待ち時間は 62 秒になります。

```
(10:01:00 - 10:00:30) + (10:00:32 - 10:00:00) = 62
```

キャプチャー・プログラムおよびアプライ・プログラムのジャーナル・ジョブの状況のチェック (System i)

DB2 for System iは、サブシステム・ジョブの処理 (WRKSBSJOB) システム・コマンドを使用して、キャプチャー・プログラムおよびアプライ・プログラムのジャーナル・ジョブの状況をチェックします。

手順

キャプチャー・プログラムとアプライ・プログラムのジャーナル・ジョブの状況をチェックするには、以下のようにします。

サブシステム・ジョブの処理 (WRKSBSJOB) システム・コマンドを以下のように使用します。

1. 次のコマンドを入力します。

   ```
   WRKSBSJOB subsystem
   ```

   この *subsystem* はサブシステム名です。ユーザーが自身のサブシステム記述を作成していないかぎり、多くの場合、サブシステムは QZSNDPR です。
2. 実行中として表示されるジョブのリストから対象のジョブを見つけます。

   ジャーナル・ジョブは、割り当てられているジャーナルに従って命名されています。このリストにジョブがない場合は、サブミットしたジョブの処理 (WRKSBMJOB) システム・コマンドまたはジョブの処理 (WRKJOB) システム・コマンドを使用して、ジョブを見つけます。ジョブのジョブ・ログを探して、ジョブが正常完了していること、またはジョブが失敗した理由を確認します。

キャプチャー・プログラムの進行のモニター (System i)

キャプチャー・プログラムが終了した場合は、IBMSNAP_RESTART 表を調べて、キャプチャー・プログラムが終了前にどこまで進行したかを確認できます。ソース表によって使用される各ジャーナルごとに 1 つの行があります。 LOGMARKER 列には、正常に処理された最後のジャーナル項目のタイム・スタンプが記されています。 SEQNBR 列には、そのジャーナル項目のシーケンス番号が記されています。

このタスクについて

キャプチャー・プログラムが終了した場合は、IBMSNAP_RESTART 表を調べて、キャプチャー・プログラムが終了前にどこまで進行したかを確認できます。ソース表によって使用される各ジャーナルごとに 1 つの行があります。 LOGMARKER 列には、正常に処理された最後のジャーナル項目のタイム・スタンプが記されています。 SEQNBR 列には、そのジャーナル項目のシーケンス番号が記されています。

手順

キャプチャー・プログラムの実行中に進行状況を確認するには、以下のようにします。

1. キャプチャーされる各ソース表の CD 表を開きます。
2. 各 CD 表の最後の行の COMMITSEQ 列にある 16 進値をメモします。
3. IBMSNAP_UOW 表の中で、COMMITSEQ の 16 進値がそれと同じ値になっている行を見つけます。一致する COMMITSEQ 値が IBMSNAP_UOW 表に存在しない場合には、 CD 表の最後から 2 番目の行で処理を繰り返してください。一致する 16 進値が見つかるまで、CD 表をさかのぼっていきます。
4. 一致する COMMITSEQ の 16 進値が見つかったら、その UOW 行の LOGMARKER 列にある値をメモします。これが、最後に処理されたジャーナル項目のタイム・スタンプです。その時間までソース表に加えられているすべての変更は、適用される準備ができています。
5. ジャーナルの表示 (DSPJRN) システム・コマンドを使用して、キャプチャー・プログラムによる処理待ちのジャーナル項目の数を確認します。以下の例のように、出力ファイル (またはプリンター) に出力を送信して、レポートを保存します。

```
DSPJRN FILE(JRNLIB/DJRN1)
       RCVRNG(*CURCHAIN)
       FROMTIME(timestamp)
       TOTIME(*LAST)
       JRNCDE(J F R C)
       OUTPUT(*OUTFILE)
       ENTDTALEN(1) OUTFILE(library/outfile)
```

この *timestamp* は、4 で識別されたタイム・スタンプです。

出力ファイルにあるレコード数が、キャプチャー・プログラムが処理しなければならない残っているジャーナル項目の大体の数です。

第 19 章 SQL レプリケーション用のレプリケーション SQL スクリプトのカスタマイズおよび実行

コントロール表の作成、ソース表の登録、サブスクリプション・セットとメンバーの作成のためには、レプリケーション・センターと ASNCLP コマンド行プログラムで生成される SQL スクリプトを実行しなければなりません。 SQL スクリプトは、レプリケーション・センターを使用して、あるいは DB2 コマンド行から実行できます。 SQL スクリプトは、必要に応じて、ユーザーの要求を満たすように変更できます。

始める前に

SQL スクリプトを DB2 コマンド行から実行する場合は、SQL スクリプトの実行時にサーバーに手動で接続しなければなりません。また、SQL ステートメントを編集して、ユーザー ID とパスワードを接続先のサーバーに指定することも必要です。例えば、以下の例に似た行を探し、プレースホルダー (XXXX) を上書きして情報を追加します。

```
CONNECT TO srcdb USER XXXX USING XXXX ;
```

このタスクについて

ASNCLP とレプリケーション・センターには、生成された SQL スクリプトをすぐに実行するためのオプションと、生成された SQL スクリプトを保管して後から実行するためのオプションがあります。 SQL をすぐに実行するオプションを選択した場合でも、今後の参照用に SQL を保管することは可能です。例えば、大規模なレプリケーション・サブスクリプション・セットの定義を SQL ファイルに保存しておけば、必要なときにその定義を再実行できます。

生成済みの SQL スクリプトを編集するときは、終了文字を変更しないように注意してください。また、ファイルに保管されるスクリプトが複数ある場合は、スクリプト区切り文字を変更しないでください。

以下のタスクを行うために、SQL スクリプトをユーザーの環境に合わせてカスタマイズすることがあります。

- 同一のレプリケーション・アクション (複数のサーバー用にカスタマイズされたもの) の複数のコピーを作成する。
- CD 表の表スペースまたはデータベースをサイズ変更する。
- サイト別の標準を定義する。
- 定義を結合してバッチ・ジョブとして実行する。
- 指定時刻までレプリケーション・アクションの実行を延期する。
- バックアップ用、サイト専用のカスタマイズ用、または (不定期接続の環境の場合等) 分散サイトでのスタンドアロン実行用に SQL スクリプトのライブラリーを作成する。
- 表や索引の作成ステートメントを編集し、データベース・オブジェクトを表すようにする。

- Informixなどの非 DB2 リレーショナル・データベースの場合は、対象のデータベース・スペースまたは表スペースに表が確実に作成されるようにする。
- Microsoft SQL Server の場合、既存のセグメントにコントロール表を作成する。
- 複数のサブスクリプション・セットを同時に定義する方法として、サブスクリプション・セット・メンバーの述部を表示また編集する。 述部に置換変数を使用して、その変数をプログラミング論理によって解決することができます。

手順

次の方法のどちらかを使用して、SQL スクリプトが含まれたファイルを DB2 コマンド行から実行します。

- SQL スクリプトにセミコロン (;) が終了文字として含まれている場合は、このコマンドを使用する。`db2 -tvf filename`
- SQL スクリプトに区切り文字として他の文字が含まれている場合は (この例では、異機種のレプリケーションなどでポンド記号 (#) が終了文字)、このコマンドを使用する。`db2 -td# -vf filename`

第 20 章 SQL レプリケーション・コンポーネントの通信方法

さまざまなレプリケーション・コンポーネントは、それぞれ独立して実行しますが、相互の通信のためにそれぞれがレプリケーション・コントロール表に格納する情報については、互いに依存し合っています。

管理ツール
レプリケーション・センターまたは ASNCLP コマンド行プログラムは、登録されたソース、サブスクリプション・セット、アラート条件についての初期情報をコントロール表に挿入する SQL スクリプトを作成します。

キャプチャー・プログラムまたはトリガー
キャプチャー・プログラムおよびキャプチャー・トリガーは、コントロール表を更新することにより、レプリケーションの進行状況を示し、変更の処理を調整します。

アプライ・プログラム
アプライ・プログラムは、コントロール表を更新することにより、レプリケーションの進行状況を示し、変更の処理を調整します。

レプリケーション・アラート・モニター
レプリケーション・アラート・モニターは、キャプチャー・プログラム、アプライ・プログラム、およびキャプチャー・トリガーにより更新されたコントロール表を読み取り、サーバーでの問題や進行状況を理解します。

レプリケーション・センター、ASNCLP、キャプチャー・プログラムまたはトリガー、およびアプライ・プログラム

レプリケーション・ソースとして、表、ビュー、ニックネームを登録すると、レプリケーション・センターまたは ASNCLP コマンド行プログラムは、すべての登録情報を含んだレプリケーション・コントロール表、つまり IBMSNAP_REGISTER 表にそのソースの情報を保管する SQL スクリプトを作成します。管理ツールによって生成される SQL スクリプトは、登録されたソース用の CD 表も作成します。

IBMSNAP_REGISTER 表には、登録されたソースごとに 1 つの行があり、また、登録されたビュー内の基礎表ごとに 1 つの行があります。この表は、登録されたそれぞれのソースについて、次の種類の情報を含んでいます。

- ソース表のスキーマ名および名前
- 登録されたそれぞれのソース表の構造タイプ
- CD 表のスキーマ名および名前
- このビュー内の基礎表の CD 表の名前 (登録されたビューの場合で、基礎表が登録されている場合のみ)
- 内部 CCD 表のスキーマ名および名前 (内部 CCD 表がある場合)
- Update-anywhere ソースの競合検出レベル

キャプチャー・プログラムとアプライ・プログラムは、IBMSNAP_REGISTER 表にある情報を使用して、それぞれの状況を互いに通知し合います。この表は関連する情報用にいくつかの列を持っています。

System i ソースの場合、リモート側でジャーナルに記録された表を含め、IBMSNAP_REGISTER 表への拡張があり、IBMSNAP_REG_EXT には、System i にユニークな追加の情報 (ジャーナル・ライブラリーやジャーナル名など) が含まれます。

サブスクリプション・セットを作成し、それにメンバーを追加すると、レプリケーション・センターは、すべてのサブスクリプション・セット情報が含まれる以下のようなレプリケーション・コントロール表にそのサブスクリプション・セット用の情報を保管する SQL スクリプトを作成します。

- IBMSNAP_SUBS_SET 表
- IBMSNAP_SUBS_MEMBR 表
- IBMSNAP_SUBS_COLS 表
- IBMSNAP_SUBS_STMTS 表

ターゲット表がまだ存在しない場合は、レプリケーション・センターによって生成される SQL スクリプトがターゲット表も作成します。

メインのサブスクリプション・セット表である IBMSNAP_SUBS_SET には、それぞれのサブスクリプション・セットごとに 1 つの行があります。この表は、それぞれのサブスクリプション・セットについて、次の種類の情報を含んでいます。

- アプライ修飾子
- サブスクリプション・セットの名前
- サブスクリプション・セットのタイプ: 読み取り専用か、または読み取り/書き込み (Update-anywhere)
- ソースおよびターゲット・データベースの名前と別名
- サブスクリプション・セットを処理するタイミング
- サブスクリプション・セットの現在の状況

この表には、関連する情報のための列もさらにいくつかあります。

その他のサブスクリプション・セット表には、サブスクリプション・セットのメンバー、列、およびセットを使用して処理される SQL ステートメント (またはストアード・プロシージャー) についての情報が含まれます。

キャプチャー・プログラムおよびアプライ・プログラム

キャプチャー・プログラムは、レプリケーション・コントロール表のいくつかを使用して、ソース・データベースにどのような変更がなされたかを示し、アプライ・プログラムはそれらのコントロール表の値を使用して、ターゲット・データベースに何をコピーする必要があるかを検出します。

キャプチャー・プログラムは、アプライ・プログラムから指示を受けない限り何も情報をキャプチャーせず、また、アプライ・プログラムは、レプリケーション・ソースおよびそれに関連するサブスクリプション・セットが定義されるまで、キャプチャー・プログラムに変更のキャプチャーを開始する指示を出しません。

以下のリストは、データ保全性を保つために、通常のレプリケーション・シナリオでアプライ・プログラムとキャプチャー・プログラムがどのように通信するかを説明しています。

ソース・データベースからデータを取り込む

1. キャプチャー・プログラムは始動時に IBMSNAP_REGISTER 表を読み取り、登録されたレプリケーション・ソースの中のどれについて変更をキャプチャーする必要があるかを識別します。その後、登録情報をメモリー内に保持します。
2. キャプチャー・プログラムは DB2 ログまたはジャーナルを継続して読み取り、登録されたソース表またはビューの変更レコード (INSERT、UPDATE、および DELETE) を検出します。また、アプライ・プログラムまたはユーザーによって開始されたシグナル・アクションを拾い出すため、IBMSNAP_SIGNAL 表への挿入も検出します。アプライ・プログラムが CAPSTART シグナルを IBMSNAP_SIGNAL 表に挿入し、コミットされたシグナルをキャプチャー・プログラムが検出すると、キャプチャー・プログラムは登録を開始し、関連するソースの変更のキャプチャーを開始します。
3. 登録されたソースの変更のキャプチャーをキャプチャー・プログラムが開始すると、キャプチャー・プログラムは、DB2 ログまたはジャーナルで検出したコミット済みの変更ごとに、 CD 表に 1 行を書き込みます (更新 ID を DELETE または INSERT ステートメントとして保管するように指定した場合は 2 行)。キャプチャー・プログラムは、コミットされていない変更については、変更がコミットされるかまたは打ち切られるまで、メモリーに保持します。外部 CCD 表でない、登録済みのレプリケーション・ソースは、それぞれ 1 つの関連する CD 表を持ちます。
4. コミット・インターバルごとに、キャプチャー・プログラムは CD および UOW 表に書き込んだデータをコミットし、さらに IBMSNAP_REGISTER 表を更新して、どの CD 表に新しくコミットされた変更があるかを示すフラグを付けます。

データをターゲット・データベースに適用する

1. 新しく定義されたサブスクリプション・セットについてはすべて、アプライ・プログラムは最初に、キャプチャー・プログラムに変更をキャプチャーするようにシグナルを出します。その後、セットのそれぞれのメンバーについてフル・リフレッシュを実行します (不完全なターゲット表は除く)。
2. レプリケーションの対象として適格なサブスクリプション・セットがある場合、アプライ・プログラムは IBMSNAP_REGISTER 表を調べて、複製しなければならない変更があるかどうかを判別します。
3. アプライ・プログラムは CD 表からターゲット表に変更をコピーします。
4. アプライ・プログラムは IBMSNAP_SUBS_SET 表を更新し、アプライ・プログラムがそれぞれのサブスクリプション・セット用にコピーしたデータの量を記録します。

5. アプライ・プログラムは、CD 表から変更を読み取った時点を示す値で、 IBMSNAP_PRUNE_SET 表を更新します。

CD 表の整理

キャプチャー・プログラムは、CD 表を整理するときに、IBMSNAP_PRUNE_SET 表にある情報を使用して、どの変更が適用されたかを判別し、複製済みの変更を CD 表から削除します。

キャプチャー・トリガーおよびアプライ・プログラム

キャプチャー・トリガーは、レプリケーション・コントロール表のいくつかを使用して、ソース・データベースにどのような変更がなされたかを示し、アプライ・プログラムはそれらのコントロール表の値を使用して、ターゲット・データベースに何をコピーする必要があるかを検出します。

キャプチャー・トリガーは情報のキャプチャーを即時に開始します。キャプチャー・プログラムと異なり、アプライ・プログラムからのシグナルを待つことはしません。

以下のリストは、データ保全性を保つために、通常のレプリケーション・シナリオでキャプチャー・トリガーとアプライ・プログラムがどのように通信するかを説明しています。

ソースからデータを取り込む

1. 登録されたレプリケーション・ソース表で DELETE、UPDATE、または INSERT 操作が行われた場合はいつでも、キャプチャー・トリガーはその変更を、そのソース表用の CCD 表に記録します。

データをターゲットに適用する

1. 新しく定義されたサブスクリプション・セットについてはすべて、アプライ・プログラムは最初に、キャプチャー・トリガーにシグナルを出し、どの時点から変更データのフェッチを開始するかを示す、有効な開始点を CCD 表に記録するように指示します。その後、セットのそれぞれのメンバーについてフル・リフレッシュを実行します (不完全なターゲット表は除く)。
2. アプライ・プログラムは、DB2 以外のリレーショナル・ソース用のサブスクリプション・セットを処理する時に、 IBMSNAP_REG_SYNCH 表を更新します。この更新により、この表の UPDATE トリガーが起動されます。このトリガーは、 IBMSNAP_REGISTER 表の SYNCHPOINT 値を更新し、ターゲットにコピーした CCD 表内の最高の SYNCHPOINT 値を記録します。次のサイクルで、アプライ・プログラムは、この SYNCHPOINT より小さいか等しい SYNCHPOINT 値を持つ、CCD 表内の新しいデータを処理します。 IBMSNAP_REG_SYNCH 表は DB2 データベースにはないので、アプライ・プログラムはレプリケーション・センターが作成したニックネームを使用して表に書き込みます。
3. アプライ・プログラムは IBMSNAP_REGISTER 表を調べて、複製しなければならない変更があるかどうかを判別します。
4. アプライ・プログラムは CCD 表から得られる変更をターゲット表にコピーします。

5. アプライ・プログラムは IBMSNAP_SUBS_SET 表を更新し、アプライ・プログラムがそれぞれのサブスクリプション・セット用にコピーしたデータの量を記録します。
6. アプライ・プログラムは、CCD 表から変更を読み取った時点を示す値で、登録されたそれぞれのソースの IBMSNAP_PRUNCNTL 表を更新します。

CCD 表の整理

IBMSNAP_PRUNCNTL 表の UPDATE トリガーは、ソース・データベースにあるすべての CCD 表を調べて、すでに複製された変更を CCD 表から削除します。

管理ツールとレプリケーション・アラート・モニター

アラート条件を定義する際に、その条件の発生時にどの連絡先に通知するかを指定すると、レプリケーション・センターまたは ASNCLP コマンド行プログラムは、そのアラート条件と連絡先情報をレプリケーション・コントロール表に保管する SQL スクリプトを作成します。これらのコントロール表には、すべてのアラート条件と通知情報が含まれます。

以下のコントロール表が更新されます。

- IBMSNAP_CONDITIONS 表
- IBMSNAP_CONTACTS 表
- IBMSNAP_GROUPS 表
- IBMSNAP_CONTACTGRP 表

IBMSNAP_CONDITIONS 表には、モニターの対象となるそれぞれの条件ごとに 1 行が登録されています。この表は、それぞれのアラート条件について、次の種類の情報を保持します。

- モニター修飾子
- モニターするキャプチャー・サーバーまたはアプライ・サーバーの名前および別名
- モニターするコンポーネント (キャプチャー・プログラムまたはアプライ・プログラム)
- キャプチャー・スキーマまたはアプライ修飾子
- サブスクリプション・セットの名前 (セットをモニターする場合)
- モニターするアラート条件
- 条件が起こった時に知らせる連絡先

この表は関連する情報用にいくつかの列を持っています。

レプリケーション・アラート・モニターのその他の表には、アラート条件が発生した場合に誰に知らせるか (個別の連絡先、連絡先のグループ、z/OS コンソールのいずれか)、その連絡先にどのような方法で知らせるか (E メールやページャー)、その条件が発生し続ける場合にどの程度の頻度で連絡先に知らせるか、といった情報が含まれます。

レプリケーション・アラート・モニター、キャプチャー・プログラム、およびアプライ・プログラム

レプリケーション・アラート・モニターは、キャプチャー・コントロール表のいくつかを使用してキャプチャー・プログラムをモニターし、アプライ・コントロール表のいくつかを使用してアプライ・プログラムをモニターします。モニターは、それぞれのキャプチャー・コントロール・サーバーまたはアプライ・コントロール・サーバーで、何をモニターするかによって、異なるレプリケーション・コントロール表から読み取ります。

レプリケーション・アラート・モニターは、キャプチャー・プログラムまたはアプライ・プログラムへの介入や通信は行いません。

以下のステップは、レプリケーション・アラート・モニターがキャプチャー・プログラムやアプライ・プログラムの条件をどのようにモニターし、アラート条件が発生した場合にどのように連絡先に知らせるかを説明しています。

1. レプリケーション・アラート・モニターは、 IBMSNAP_CONDITIONS 表内のそれぞれの条件 (モニター修飾子について) のアラート条件および連絡先を読み取ります。
2. アラート条件が定義されているキャプチャー・コントロール・サーバーまたはアプライ・コントロール・サーバーごとに、レプリケーション・アラート・モニターは次のタスクを行います。
   a. レプリケーション・アラート・モニターはサーバーに接続し、そのサーバーの各アラート条件に関連付けられたレプリケーション・コントロール表を読み、条件のいずれかを満たすかどうかを調べます。
   b. いずれかの条件を満たした場合、レプリケーション・アラート・モニターはその条件に関するデータをメモリーに保管し、そのサーバーの残りのアラート条件の処理を続けます。
   c. そのサーバーのすべてのアラート条件の処理が終了したら、レプリケーション・アラート・モニターはキャプチャー・コントロール・サーバーまたはアプライ・コントロール・サーバーを切断し、アラートを IBMSNAP_ALERTS 表に挿入し、その条件について連絡先に知らせます。

第 21 章 SQL レプリケーション・オブジェクトの命名規則

以下の表は、レプリケーション・オブジェクトの名前の制限を示しています。

表 23. レプリケーション・オブジェクトの名前の制限

オブジェクト	名前の制限
ソース表とターゲット表	Linux UNIX Windows データベース管理システムの命名規則に従ってください。 System i 名前に、ブランク、アスタリスク (*)、疑問符 (?)、単一引用符 (')、二重引用符 (")、スラッシュ (/) を含めることはできません。
ソース列とターゲット列	データベース管理システムの命名規則に従ってください。 (すべての変更前イメージ列には、1 文字の接頭部が追加されます。変更前イメージ列の名前が未確定にならないようにするため、ソース列名が 29 文字までユニークなものになるようにし、列名に変更前イメージ文字接頭部を追加するときに、変更前イメージ列名が既存の列名と競合しないようにしてください。)
サブスクリプション・セット	サブスクリプション・セット名には、DB2 で可変文字 (VARCHAR) 列に許可されているすべての文字を使用できます。**推奨:** DB2 表および列名の命名規則に従ってください。DB2 レプリケーションは、サブスクリプション・セット名を各レプリケーション・コントロール・サーバーに保管するため、名前を 3 つのすべてのサーバーのコード・ページと互換性のあるものにしてください。
キャプチャー・スキーマ	Linux UNIX Windows キャプチャー・スキーマには 30 文字以下のストリングを使用できます[1]。 z/OS キャプチャー・スキーマには 18 文字以下のストリングを使用できます。DB2 UDB for z/OS バージョン 8 の新規関数モードのサブシステムでは、128 文字以下のストリングを使用できます[1]。 System i キャプチャー・スキーマ (CAPCTLLIB) には 10 文字以下の英数字を使用できます[1]。
アプライ修飾子	z/OS Linux UNIX Windows アプライ修飾子は、18 文字以下のストリングにできます[1]。 System i アプライ修飾子には 18 文字以下のストリングを使用できますが、アプライ・ジョブの長さは 10 文字までにしかできないため、どのアプライ修飾子も、最初の 10 文字はユニークなものでなければなりません[1]。

表 23. レプリケーション・オブジェクトの名前の制限 (続き)

オブジェクト	名前の制限
モニター修飾子	z/OS Linux UNIX Windows モニター修飾子は、18 文字以下のストリングにできます[1]。

注:

1. キャプチャー・スキーマ、アプライ修飾子、およびモニター修飾子では、これらのオブジェクトの名前に、以下の有効文字のみを使用してください。
   - A ～ Z (英大文字)
   - a ～ z (英小文字)
   - 数表示 (0 ～ 9)
   - 下線文字 "_"

   ブランクは使用できません。また、コロン「:」および正符号「+」などの特殊文字も使用できません。

レプリケーション・システム・コマンドおよびレプリケーション・センターは、デフォルトとして、ユーザーから指定されたすべての名前を大文字に変換します。入力された名前のとおりに正確に大文字と小文字を維持するには、大文字小文字混合文字の名前を二重引用符 (またはターゲット・システムで使用できるように構成されている他の文字) で囲んでください。例えば、`myqual` または `MyQual` または `MYQUAL` と入力すると、名前は `MYQUAL` として保管されます。これらの同じ名前を二重引用符で囲んで入力すると、`myqual` または `MyQual` または `MYQUAL` としてそれぞれ保管されます。オペレーティング・システムによっては二重引用符が認識されないことがあります。その場合は、エスケープ文字としてバックスラッシュまたは円記号 (¥) を使用する必要があります。

Windows Windows オペレーティング・システムの場合は、同じ名前を区別するためにユニークなパスを使用する必要があります。例えば、3 つのアプライ修飾子、`myqual`、`MyQual`、および `MYQUAL` を使用するとします。3 つの名前は同じ文字を使用していますが、大文字と小文字が異なります。これら 3 つの修飾子が同じアプライ・パスにあると、名前の競合が発生します。

重要: キャプチャー・プログラム、アプライ・プログラム、またはレプリケーション・アラート・モニター用に Windows サービスをセットアップするときには、キャプチャー・スキーマ、アプライ修飾子、およびモニター修飾子にユニークな名前を使用する必要があります。大文字と小文字を使用してこれらの名前を区別することはできません。

第 22 章 SQL レプリケーション用のシステム・コマンド (Linux、UNIX、Windows、z/OS)

このセクションでは、Linux、UNIX、Windows、および z/OS 上の UNIX System Services (USS) の、SQL レプリケーション・プログラムの始動、操作、変更、およびモニターを行うコマンドについて説明します。

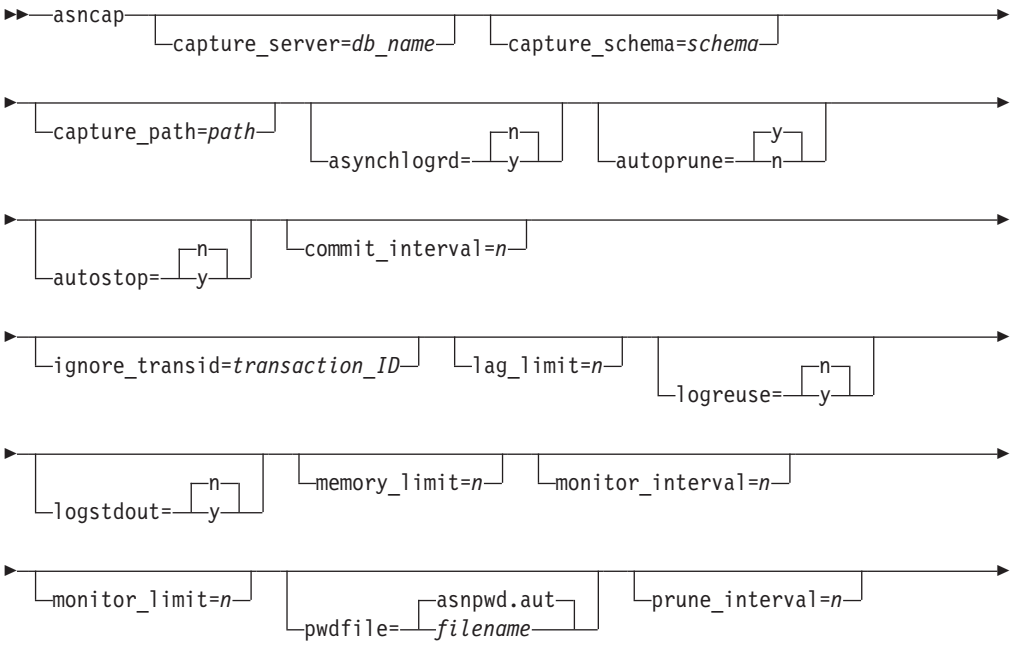
これらのコマンドはすべて asn という接頭部を持ち、オペレーティング・システムのコマンド・プロンプトか、シェル・スクリプトに入力されます。コマンドの 1 つ、asnanalyze は、System i にあるリモート・データにも使用できます。

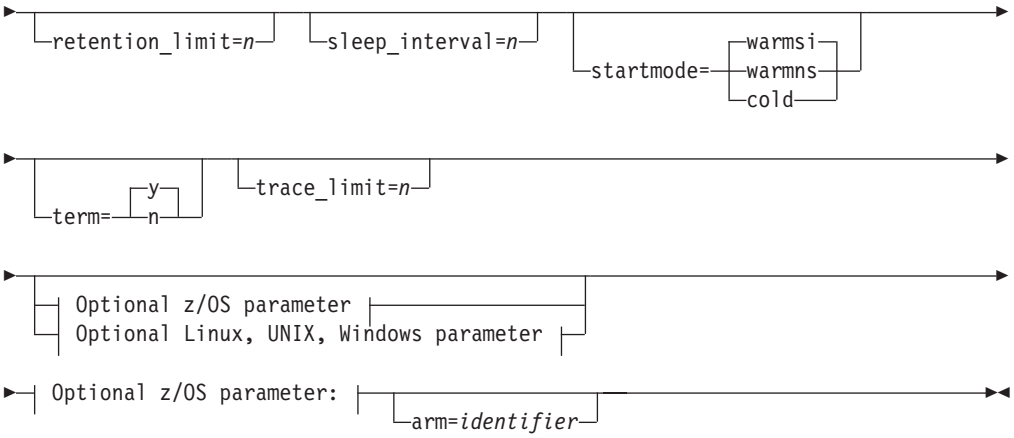
asncap: キャプチャーの始動

asncap コマンドを使用して Linux、 UNIX、 Windows および z/OS の UNIX System Services (USS) でキャプチャー・プログラムを開始します。このコマンドは、オペレーティング・システムのプロンプトまたはシェル・スクリプト内で実行します。

キャプチャー・プログラムを始動すると、停止されるかまたはリカバリー不能エラーが検出されるまで実行を続けます。

構文





Optional Linux, UNIX, Windows parameter:

パラメーター

表 24 は、呼び出しパラメーターを定義しています。

表 24. Linux、UNIX、Windows、および z/OS オペレーティング・システム用 asncap 呼び出しパラメーター定義

パラメーター	定義
capture_server=*db_name*	キャプチャー・コントロール・サーバーの名前を指定します。 z/OS キャプチャー・プログラムを実行する DB2 サブシステムの名前を指定します。データ共有の場合、グループ・アタッチ名を使用しないでください。その代わりに、メンバー・サブシステム名を指定してください。 Linux UNIX Windows キャプチャー・コントロール・サーバーを指定しない場合、このパラメーターはデフォルトで DB2DBDFT 環境変数の値になります。

表 24. *Linux*、*UNIX*、*Windows*、および *z/OS* オペレーティング・システム用 *asncap* 呼び出しパラメーター定義 *(続き)*

パラメーター	定義
add_partition=*y/n*	Linux UNIX Windows 最後にキャプチャー・プログラムが再始動されてから、新しく追加されたパーティションのログ・ファイルの読み取りを、キャプチャー・プログラムが開始するかどうかを指定します。 **n (デフォルト)** キャプチャー・プログラムが最後に再始動されてから、新規パーティションは追加されていません。 **y** キャプチャー・プログラムは、1 つ以上の新規パーティション上でログ・ファイルの読み取りを開始します。各パーティション上で、キャプチャー・プログラムはデータベースを最後に始動したときに最初に使用されたログ・シーケンス番号 (LSN) からログの読み取りを開始します。
arm=*identifier*	z/OS キャプチャー・プログラムの単一インスタンスを自動リスタート・マネージャーに対して識別するのに使用する 3 文字の英数字ストリングを指定します。指定された値が、キャプチャーがそれ自体のために生成する ARM 要素名 ASNTC*xxxxyyyy* (*xxxx* はデータ共用グループ・アタッチ名、*yyyy* は DB2 メンバー名) に追加されます。**arm** パラメーターにはどんな長さのストリングでも指定できますが、キャプチャー・プログラムは現在の名前に最大で 3 文字だけを連結します。必要なら、キャプチャー・プログラムは名前にブランクを埋め込んで、固有の 16 バイトの名前を作ります。

表 24. *Linux*、*UNIX*、*Windows*、および *z/OS* オペレーティング・システム用 *asncap* 呼び出しパラメーター定義 *(続き)*

パラメーター	定義
asynchlogrd=*y/n*	**n (デフォルト)** キャプチャー・プログラムが DB2 リカバリー・ログの読み取りと、ログからキャプチャーされたトランザクションの処理に同一のスレッドを使用することを指定します。 **y** キャプチャー・プログラムが DB2 リカバリー・ログからのトランザクションのキャプチャーに専用スレッドを使用することを指定します。トランザクション読み取りスレッドは、メモリー・バッファー内のコミット済みトランザクションをプリフェッチします。メモリー・バッファーからは、別のスレッドがトランザクションを取得して処理し、CD 表に挿入するための SQL ステートメントにします。この非同期モードでは、すべての環境でキャプチャー・パフォーマンスを向上させることができ、特にパーティション・データベースおよび z/OS データ共用に対して効果があります。非常に高いアクティビティー・レベルを持つシステムの場合、このプリフェッチによりメモリー使用量が増加する可能性があります。**memory_limit** パラメーターを適切に調整してください。
capture_schema=*schema*	特定のキャプチャー・プログラムを識別するために使用するキャプチャー・スキーマの名前を指定します。入力するスキーマ名の長さは 1 から 30 文字でなければなりません。デフォルトは ASN です。
capture_path=*path*	キャプチャー・プログラムが使用する作業ファイルのロケーションを指定します。デフォルトは、asncap コマンドが呼び出されたディレクトリーです。
autoprune=*y/n*	変更データ (CD) 表、作業単位 (UOW) 表、IBMSNAP_CAPMON 表、 IBMSNAP_CAPTRACE 表、および IBMSNAP_SIGNAL 表の中の行の自動整理を可能にするかどうかを指定します。 **y (デフォルト)** キャプチャー・プログラムは、IBMSNAP_CAPPARMS 表で指定されたインターバルで、適格な行の自動整理を行います。キャプチャー・プログラムは、行が複製されたものか否かに関係なく、保持制限より古い CD、UOW、および IBMSNAP_SIGNAL の行を削除します。 **n** 自動整理は使用不可になります。

表 24. *Linux*、*UNIX*、*Windows*、および *z/OS* オペレーティング・システム用 *asncap* 呼び出しパラメーター定義 *(続き)*

パラメーター	定義
autostop=*y/n*	キャプチャー・プログラムの始動前にログに記録されたすべてのトランザクションを検索した後、キャプチャー・プログラムを終了するかどうかを指定します。 **n (デフォルト)** キャプチャー・プログラムは、トランザクションを検索した後、終了しません。 **y** キャプチャー・プログラムは、トランザクションを検索した後、終了します。
commit_interval=*n*	キャプチャー・プログラムが、何秒待ってから、作業単位 (UOW) 表および変更データ (CD) 表に行をコミットするかを示す秒数を指定します。デフォルトは 30 秒です。
ignore_transid=*transaction_ID*	キャプチャー・プログラムが *transaction_ID* により識別されるトランザクションをキャプチャーしないことを指定します。 *transaction_ID* の値は、 10 バイトの 16 進 ID で、形式は以下のとおりです。 z/OS 0000:*xxxx*:*xxxx*:*xxxx*:*mmmm* ここで、*xxxx*:*xxxx*:*xxxx* はトランザクション ID、*mmmm* はデータ共用メンバー ID です。メンバー ID は、LOGP 出力内にあるログ・レコード・ヘッダーの末尾の 2 バイトにあります。データ共用が有効になっていない場合、メンバー ID は 0000 です。 Linux UNIX Windows *nnnn*:0000:*xxxx*:*xxxx*:*xxxx* ここで、*xxxx*:*xxxx*:*xxxx* はトランザクション ID、*nnnn* はパーティション・データベースのパーティション ID です (非パーティション・データベースの場合、この値は 0000 です)。
lag_limit=*n*	キャプチャー・プログラムがログ・レコードを処理するときに、許される遅れの分数を指定します。デフォルトは 10,080 分 (7 日) です。キャプチャー・プログラムは、ウォーム・スタートの場合にのみ、このパラメーターの値をチェックします。この限界を超えると、キャプチャー・プログラムは開始されません。

表 24. Linux、UNIX、Windows、および z/OS オペレーティング・システム用 asncap 呼び出しパラメーター定義 (続き)

パラメーター	定義
logreuse=*y/n*	キャプチャー・プログラムがログ・ファイルを再利用するか、またはログ・ファイルにメッセージを付加するかを指定します。 **n (デフォルト)** キャプチャー・プログラムは、キャプチャー・プログラムの再始動後であってもログ・ファイルにメッセージを付加します。 **y** キャプチャー・プログラムは、まず現行のログ・ファイルを切り捨て、次にキャプチャー・プログラムの再始動時に新しいログを開始して、ログ・ファイルを再利用します。 z/OS 以下のように、ログ・ファイル名には DB2 インスタンス名は含まれません。 *capture_server.capture_schema*.CAP.log Linux UNIX Windows 以下のように、ログ・ファイル名には DB2 インスタンス名が含まれます。 *db2instance.capture_server.capture_schema*.CAP.log
logstdout=*y/n*	キャプチャー・プログラムがログ・ファイル・メッセージを送る場所を指定します。 **n (デフォルト)** キャプチャー・プログラムは、ほとんどのログ・ファイル・メッセージをログ・ファイルにのみ送ります。初期化メッセージは、ログ・ファイルと標準出力 (STDOUT) の両方に送られます。 **y** キャプチャー・プログラムは、ログ・ファイルと標準出力 (stdout) の両方にメッセージを送信します。
memory_limit=*n*	トランザクションを作成するためにキャプチャー・プログラムが使用できるメモリーの最大サイズ (MB 単位) を指定します。このメモリー限度に達すると、キャプチャー・プログラムはトランザクションをファイルに書き出します。デフォルトは 32 MB です。 z/OS **memory_limit**=0 を指定した場合、キャプチャー・プログラムが、キャプチャー・ジョブの領域サイズ・パラメーターに基づいて、使用するメモリーの量を決定します。メモリー割り振りは、領域サイズの 80 パーセントになります。
monitor_interval=*n*	キャプチャー・プログラムが、 IBMSNAP_CAPMON 表に行を挿入する頻度 (秒単位) を指定します。デフォルトは 300 秒 (5 分) です。

表 24. *Linux*、*UNIX*、*Windows*、および *z/OS* オペレーティング・システム用 *asncap* 呼び出しパラメーター定義 *(続き)*

パラメーター	定義
monitor_limit=*n*	IBMSNAP_CAPMON 表内の行は、何分たったら、整理の対象として適格になるかを示す分数を指定します。**monitor_limit** パラメーターの値よりも古い、すべての IBMSNAP_CAPMON 行は、次の整理サイクルで削除されます。デフォルトは 10,080 分 (7 日) です。
pwdfile=*filename*	パスワード・ファイルの名前を指定します。パスワード・ファイルを指定しない場合、デフォルトは asnpwd.aut です。 このコマンドは、**capture_path** パラメーターで指定されたディレクトリー内でパスワード・ファイルを探します。**capture_path** パラメーターを指定しない場合、このコマンドは、コマンドを呼び出したディレクトリー内でパスワード・ファイルを探します。
prune_interval=*n*	変更データ (CD) 表、作業単位 (UOW) 表、IBMSNAP_CAPMON 表、 IBMSNAP_CAPTRACE 表、および IBMSNAP_SIGNAL 表の整理の頻度 (秒数) を指定します。autoprune パラメーターを n に設定した場合は、このパラメーターは無視されます。デフォルトは 300 秒 (5 分) です。
retention_limit=*n*	変更データ (CD) 表、作業単位 (UOW) 表、または IBMSNAP_SIGNAL 表の行は、何分たったら、整理の対象として適格になるかを示す分数を指定します。**retention_limit** パラメーターの値よりも古い行は、次の整理サイクルで削除されます。デフォルトは 10,080 分 (7 日) です。
sleep_interval=*n*	キャプチャー・プログラムが、アクティブ・ログの処理を終了し、バッファーが空であると判断するまで、何秒スリープするかを示す秒数を指定します。デフォルトは 5 秒です。 z/OS バッファーの使用率が半分に満たない状態に戻った後、キャプチャー・プログラムが何秒スリープするかを指定します。

表 24. Linux、UNIX、Windows、および zOS オペレーティング・システム用 asncap 呼び出しパラメーター定義 (続き)

パラメーター	定義
startmode=*mode*	キャプチャー始動時にキャプチャー・プログラムが使用する、処理プロシージャーを指定します。 **warmsi (デフォルト)** ウォーム・スタート情報を入手できる場合、キャプチャー・プログラムは、直前の実行の終了時点から処理を再開します。これがキャプチャー・プログラムの最初の始動である場合は、自動的にコールド・スタートに切り替えます。 ウォーム・スタート中は、キャプチャー・プログラムは、 IBMSNAP_CAPTRACE 表、変更データ (CD) 表、作業単位 (UOW) 表、および IBMSNAP_RESTART 表に手を付けずそのままにしておきます。キャプチャー・プログラムの始動後にエラーが起こった場合、キャプチャー・プログラムは終了します。 **warmns** ウォーム・スタート情報を入手できる場合、キャプチャー・プログラムは、直前の実行の終了時点から処理を再開します。キャプチャー・プログラムの始動後にエラーが起こった場合、キャプチャー・プログラムは終了します。キャプチャー・プログラムは、ウォーム・スタートできない場合、コールド・スタートに切り替わりません。 **cold** キャプチャー・プログラムを始動すると、CD、UOW 表内のすべての行を削除します。ほとんどの登録はリセットされ、それらのソースに対するすべてのサブスクリプションは、次回のアプライ処理のサイクルですべてリフレッシュされます。外部 CCD の登録とそれらのサブスクリプション (そのターゲットが非コンプリート CCD である) は、完全には更新されません。

表 24. Linux、UNIX、Windows、および z/OS オペレーティング・システム用 asncap 呼び出しパラメーター定義 (続き)

パラメーター	定義
term=*y*/*n*	DB2 が静止または停止した場合にキャプチャー・プログラムは終了するかどうかを指定します。 **y (デフォルト)** DB2 が静止または停止した場合、キャプチャー・プログラムは終了します。 **n** DB2 が静止または停止するとき、キャプチャー・プログラムは実行を継続します。 DB2 が初期化されると、キャプチャー・プログラムはウォーム・モードで始動し、 DB2 が静止または停止されて中断していた時点からのキャプチャーを開始します。 DB2 が FORCE または異常終了で終了した場合は、このパラメーターを n にしてもキャプチャー・プログラムは終了します。 このパラメーターを n にし、制限付きアクセス (ACCESS MAINT) を使用して DB2 を始動すると、キャプチャー・プログラムは接続できないので、結果として終了します。
trace_limit=*n*	IBMSNAP_CAPTRACE 表の行は、何分たったら、整理の対象として適格になるかを示す分数を指定します。**trace_limit** パラメーターの値よりも古い、すべての IBMSNAP_CAPTRACE 行が、次の整理サイクルで削除されます。デフォルトは 10,080 分 (7 日) です。

戻りコード

asncap コマンドは、正常終了したときにゼロの戻りコードを戻します。コマンドが失敗する場合、ゼロ以外の戻りコードが戻されます。

asncap の例

以下の例は、asncap コマンドの使用法を示しています。

例 1

db という名前のキャプチャー・コントロール・サーバーおよび、キャプチャー・スキーマ ASN を使用し、 /home/files/capture/logs/ ディレクトリーにある作業ファイルを使用して、初めてキャプチャー・プログラムを始動する例です。

```
asncap capture_server=db capture_schema=ASN
  capture_path=/home/files/capture/logs/ startmode=cold
```

例 2

キャプチャー・プログラムが停止した後、整理を行わずにキャプチャー・プログラムを再始動する例です。

```
asncap capture_server=db autoprune=n sleep_interval=10 startmode=warmsi
```

この例では、キャプチャー・プログラムは、対応するコントロール表内のすべての行を保存し、アクティブ・ログの処理を終了し、バッファーが空であると判断した後、10 秒スリープします。ウォーム・スタート情報を入手できない場合、キャプチャー・プログラムは、直前の実行の終了時点から処理を再開し、コールド・スタートに切り替えます。

例 3

warmns startmode で、変更されたパラメーター設定値を使用して、キャプチャー・プログラムを再始動します。

```
asncap capture_server=db autoprune=y prune_interval=60 retention_limit=1440
  startmode=warmns
```

このコマンドは、キャプチャー・プログラムを再始動し、新しいパラメーター設定値を使用するようにします。新しいパラメーター設定値は、CD、UOW、および IBMSNAP_SIGNAL 表が整理の対象として適格になるまでの時間を減らし、整理の頻度をデフォルトのパラメーター設定値よりも増やしています。ウォーム・スタート情報を入手できない場合、キャプチャー・プログラムは、直前の実行の終了時点から処理を再開しますが、自動的にコールド・スタートに切り替えることはしません。

例 4

その作業ファイルをすべて、capture_files という新しいサブディレクトリーに送信するキャプチャー・プログラムを始動します。

1. 該当のディレクトリーに行き、 capture_files という新しいサブディレクトリーを作成します。

   ```
   cd /home/db2inst
     mkdir capture_files
   ```

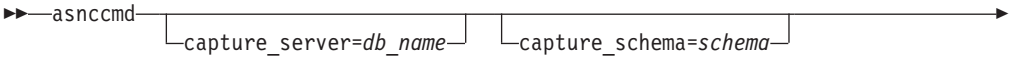
2. キャプチャー・プログラムを始動し、たった今作成した新しいサブディレクトリーにあるキャプチャー・パスを指定します。

   ```
   asncap capture_server=db capture_schema=ASN
     capture_path=/home/db2inst/capture_files startmode=warmsi
   ```

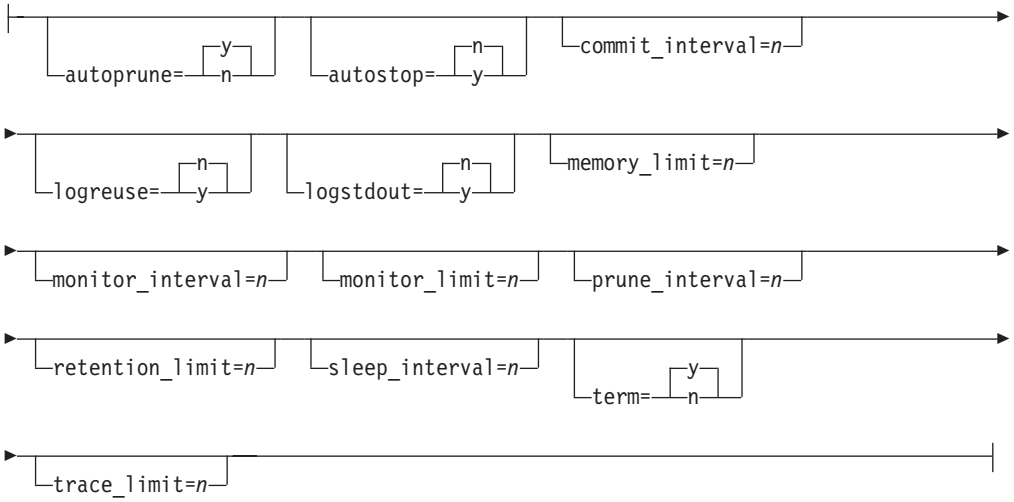
asnccmd: キャプチャーの操作

asnccmd コマンドを使用して Linux、UNIX、Windows および z/OS の UNIX System Services (USS) で実行中のキャプチャー・プログラムにコマンドを送ります。このコマンドは、オペレーティング・システムのプロンプトまたはシェル・スクリプト内で実行します。

構文



パラメーター:



パラメーター

表 25 は、 asnccmd コマンドの呼び出しパラメーターを定義します。

表 25. asnccmd 呼び出しパラメーターの定義

パラメーター	定義
capture_server=*y/n*	キャプチャー・コントロール・サーバーの名前を指定します。 z/OS コントロール・サーバーに接続するデータベース・サーバーの名前です。データ共用の場合、グループ・アタッチ名またはメンバー・サブシステム名を使用します。 Linux UNIX Windows キャプチャー・コントロール・サーバーを指定しない場合、このパラメーターはデフォルトで *DB2DBDFT* 環境変数の値になります。
capture_schema=*schema*	特定のキャプチャー・プログラムを識別するために使用するキャプチャー・スキーマの名前を指定します。スキーマ名の長さは 1 から 30 文字でなければなりません。デフォルトは ASN です。

表 25. asnccmd 呼び出しパラメーターの定義 (続き)

パラメーター	定義
chgparms	キャプチャー・プログラムの実行中に、以下の稼働パラメーターのうち 1 つ以上を変更することを指定します。 • **autostop** • **commit_interval** • **logreuse** • **logstdout** • **memory_limit** • **monitor_interval** • **monitor_limit** • **prune_interval** • **retention_limit** • **signal_limit** • **sleep_interval** • **term** • **trace_limit** **制約事項:** z/OS **memory_limit** の値は、キャプチャー・プログラムの実行中は変更することができません。値を変更するには、まずキャプチャー・プログラムを停止してください。 1 つの **asnccmd chgparms** コマンド中に複数のパラメーターを指定したり、必要に応じてこれらのパラメーター値を変更したりできます。変更内容は IBMSNAP_CAPPARMS 表の値を一時的にオーバーライドしますが、この表には書き込まれません。キャプチャー・プログラムを停止して再始動する際には、IBMSNAP_CAPPARMS 中の値が使用されます。 273 ページの『asncap: キャプチャーの始動』 には、このコマンドを使用してオーバーライドできるパラメーターの説明が記載されています。
prune	変更データ (CD) 表、作業単位 (UOW) 表、IBMSNAP_CAPMON 表、IBMSNAP_CAPTRACE 表、および IBMSNAP_SIGNAL 表をいったん整理する場合は、このパラメーターを指定します。コマンドが正常にキューに入れられると、キャプチャー・プログラムはメッセージを発行します。
qryparms	現行の稼働パラメーター値を標準出力 (stdout) に書き込む場合に指定します。
reinit	キャプチャー・プログラムが IBMSNAP_REGISTER 表から新しく追加されたレプリケーション・ソースを入手することを指定します。例えば、キャプチャー・プログラムの実行中に、新しいレプリケーション・ソースを追加する場合や、ALTER ADD ステートメントを使用してレプリケーション・ソースや変更データ (CD) 表に列を追加する場合に、このパラメーターを使用します。

表 25. *asnccmd 呼び出しパラメーターの定義 (続き)*

パラメーター	定義
suspend	キャプチャー・プログラム環境を損なうことなく、ピーク時にオペレーティング・システムのリソースを操作可能なトランザクションのために解放することを指定します。 **重要:** レプリケーション・ソースを取り消すには、キャプチャーを中断しないでください。代わりに、キャプチャー・プログラムを停止してください。
resume	中断されたキャプチャー・プログラムがデータのキャプチャーを再開することを指定します。
状況	各キャプチャー・スレッド (管理、整理、シリアライゼーション、およびワーカー) の状態を示すメッセージを受け取ることを指定します。
stop	キャプチャー・プログラムを通常の方法で停止し、その時点までに処理されたログ・レコードをコミットすることを指定します。

asnccmd の例

以下の例は、asnccmd コマンドの使用法を示しています。

例 1

新しく追加したレプリケーション・ソースを、実行中のキャプチャー・プログラムに認識させるには、次のように入力します。

```
asnccmd capture_server=db capture_schema=ASN reinit
```

例 2

CD、UOW、IBMSNAP_CAPMON、IBMSNAP_CAPTRACE、および IBMSNAP_SIGNAL 表の整理を 1 回行う例です。

```
asnccmd capture_server=db capture_schema=ASN prune
```

例 3

それぞれのキャプチャー・スレッドの状態についてメッセージを受け取る場合の例です。

```
asnccmd capture_server=db capture_schema=ASN status
```

例 4

キャプチャー・プログラムの現行操作値を標準出力に送信する例です。

```
asnccmd capture_server=db capture_schema=ASN qryparms
```

例 5

キャプチャー・プログラムの実行中に自動整理を行わないようにするには、次のように入力します。

```
asnccmd capture_server=db capture_schema=ASN chgparms autoprune=n
```

例 6

キャプチャー・プログラムの実行を停止するには、次のように入力します。

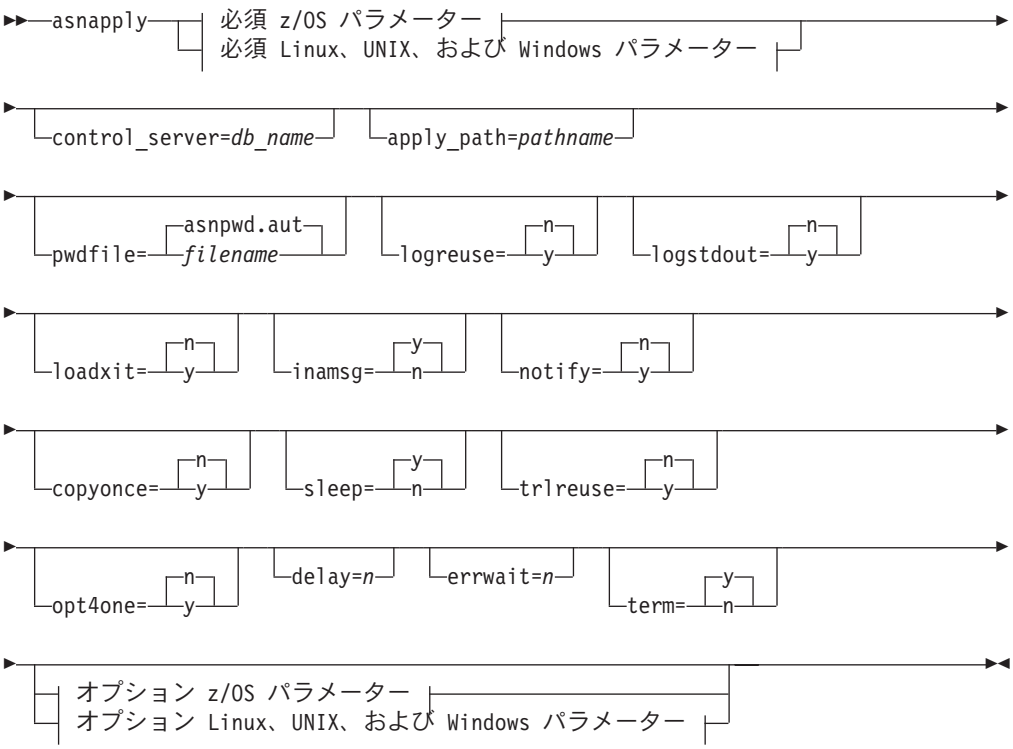
```
asnccmd capture_server=db capture_schema=ASN stop
```

asnapply: アプライの始動

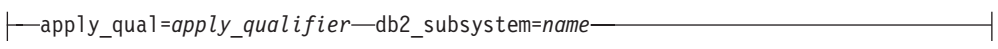
Linux、UNIX、Windows、および z/OS 上の UNIX System Services (USS) でアプライ・プログラムを操作するには、asnapply コマンドを使用します。このコマンドは、オペレーティング・システムのプロンプトまたはシェル・スクリプト内で実行します。

アプライ・プログラムを開始すると、停止されるかまたはリカバリー不能エラーが検出されるまで実行を続けます。

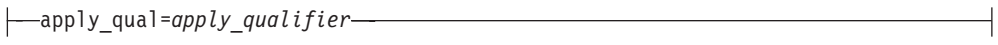
構文



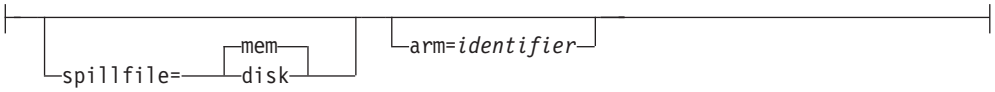
必須 z/OS パラメーター:



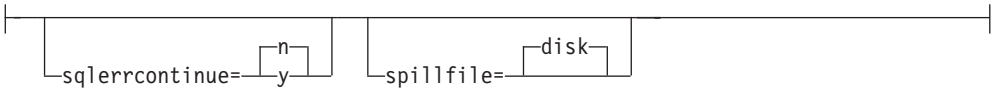
必須 Linux、UNIX、および Windows パラメーター:



オプション z/OS パラメーター:



オプション Linux、UNIX、および Windows パラメーター:



パラメーター

表 26 は、呼び出しパラメーターを定義しています。

表 26. Linux、UNIX、Windows、および z/OS オペレーティング・システム用 asnapply 呼び出しパラメーター定義

パラメーター	定義
apply_qual=*apply_qualifier*	アプライ・プログラムが、処理されるサブスクリプション・セットの識別に使用するアプライ修飾子を指定します。 入力する値は、IBMSNAP_SUBS_SET 表の APPLY_QUAL 列の値と一致する必要があります。アプライ修飾子名には大文字小文字の区別があり、最大 18 文字です。
db2_subsystem=*name*	z/OS アプライ・プログラムを実行する DB2 サブシステムの名前を指定します。入力するサブシステム名は最大 4 文字です。このパラメーターにはデフォルトはありません。このパラメーターは必須です。
control_server=*db_name*	サブスクリプション定義とアプライ・プログラム・コントロール表が存在する、アプライ・コントロール・サーバーの名前を指定します。 z/OS アプライ・コントロール・サーバーのロケーション名を指定します。 Linux UNIX Windows アプライ・コントロール・サーバーを指定しない場合、このパラメーターはデフォルトで DB2DBDFT 環境変数の値になります。
apply_path=*pathname*	アプライ・プログラムが使用する作業ファイルのロケーションを指定します。デフォルトは、**asnapply** コマンドが呼び出されたディレクトリーです。
pwdfile=*filename*	パスワード・ファイルの名前を指定します。パスワード・ファイルを指定しない場合、デフォルトは asnpwd.aut です。 このコマンドは、**apply_path** パラメーターで指定されたディレクトリー内でパスワード・ファイルを探します。**apply_path** パラメーターを指定しない場合、このコマンドは、コマンドを呼び出したディレクトリー内でパスワード・ファイルを探します。

表 26. *Linux*、*UNIX*、*Windows*、および *z/OS* オペレーティング・システム用 *asnapply* 呼び出しパラメーター定義 *(続き)*

パラメーター	定義
logreuse=*y/n*	アプライ・プログラムが、ログ・ファイルを再利用するか、またはメッセージを付加するかを指定します。 **n (デフォルト)** アプライ・プログラムは、アプライ・プログラムの再始動後であっても、ログ・ファイルにメッセージを付加します。 **y** アプライ・プログラムは、ログ・ファイルを削除し、アプライ・プログラムの再始動時にそれを再作成することにより、ログ・ファイルを再利用します。 z/OS ログ・ファイル名には DB2 インスタンス名 (*control_server.apply_qualifier*.APP.log) は含まれません。 Linux UNIX Windows ログ・ファイル名には以下の DB2 インスタンス名が含まれます。 *db2instance.control_server.apply_qualifier*.APP.log
logstdout=*y/n*	アプライ・プログラムがログ・ファイル・メッセージを送信する場所を指定します。 **n (デフォルト)** アプライ・プログラムは、ほとんどのログ・ファイル・メッセージをログ・ファイルにのみ送ります。初期化メッセージは、ログ・ファイルと標準出力 (STDOUT) の両方に送られます。 **y** アプライ・プログラムは、ログ・ファイルと標準出力 (stdout) の両方にログ・ファイル・メッセージを送信します。
loadxit=*y/n*	アプライ・プログラムが ASNLOAD を呼び出すかどうかを指定します。 ASNLOAD は IBM 提供の出口ルーチンであり、エクスポートおよびロード・ユーティリティーを使用して、ターゲット表をリフレッシュします。 **n (デフォルト)** アプライ・プログラムは ASNLOAD を呼び出しません。 **y** アプライ・プログラムは ASNLOAD を呼び出します。

表 26. Linux、UNIX、Windows、および z/OS オペレーティング・システム用 asnapply 呼び出しパラメーター定義 (続き)

パラメーター	定義
inamsg=*y/n*	アプライ・プログラムを非アクティブにしたとき、アプライ・プログラムからメッセージを出すかどうかを指定します。 **y (デフォルト)** アプライ・プログラムは非アクティブ時にメッセージを出します。 **n** アプライ・プログラムは非アクティブ時にメッセージを出しません。
notify=*y/n*	アプライ・プログラムが ASNDONE を呼び出すかどうかを指定します。 ASNDONE は、アプライ・プログラムがサブスクリプション・セットのコピーを終了した時に、ユーザーにコントロールを戻すための出口ルーチンです。 **n (デフォルト)** アプライ・プログラムは ASNDONE を呼び出しません。 **y** アプライ・プログラムは ASNDONE を呼び出します。
copyonce=*y/n*	アプライ・プログラムが呼びだされた時点で適格と見なされたサブスクリプション・セットごとに、アプライ・プログラムがコピー・サイクルを 1 回実行するかどうかを指定します。その後、アプライ・プログラムは終了します。適格と見なされるサブスクリプション・セットとは、以下の基準を満たすものです。 • IBMSNAP_SUBS_SET 表で (ACTIVATE > 0)。ACTIVATE 列の値がゼロより大きい場合、そのサブスクリプション・セットは無期限にアクティブであるか、または 1 回のみのサブスクリプション処理に使用されています。 • (REFRESH_TYPE = R または B) または (REFRESH_TYPE = E であり、指定されたイベントが発生)。 REFRESH_TYPE 列の値は IBMSNAP_SUBS_SET 表に保管されます。 サブスクリプション・セット表の MAX_SYNCH_MINUTES 限度および、 IBMSNAP_SUBS_EVENT 表の END_OF_PERIOD タイム・スタンプが指定されている場合は、これに従います。 **n (デフォルト)** アプライ・プログラムは、適格なサブスクリプション・セットごとにコピー・サイクルを 1 回実行しません。 **y** アプライ・プログラムは、適格なサブスクリプション・セットごとにコピー・サイクルを 1 回実行します。

表 26. Linux、UNIX、Windows、および z/OS オペレーティング・システム用 asnapply 呼び出しパラメーター定義 (続き)

パラメーター	定義
sleep=*y/n*	処理の対象として適格となる新しいサブスクリプションがない場合に、アプライ・プログラムがどうするかを指定します。 **y (デフォルト)** アプライ・プログラムはスリープします。 **n** アプライ・プログラムは停止します。
trlreuse=*y/n*	アプライ・プログラムの始動時に、アプライ・プログラムが IBMSNAP_APPLYTRAIL 表を空にするかどうかを指定します。 **n (デフォルト)** アプライ・プログラムは IBMSNAP_APPLYTRAIL 表に項目を付加します。アプライ・プログラムは表を空にしません。 **y** アプライ・プログラムはプログラム始動時に IBMSNAP_APPLYTRAIL 表を空にします。
opt4one=*y/n*	アプライ・プログラムに定義されているサブスクリプション・セットが 1 つだけの場合、アプライ・プログラムのパフォーマンスを最適化するかどうかを指定します。 **n (デフォルト)** サブスクリプション・セットが 1 つの場合、アプライ・プログラムのパフォーマンスを最適化しません。 **y** サブスクリプション・セットが 1 つの場合、アプライ・プログラムのパフォーマンスを最適化します。最適化を y に設定すると、アプライ・プログラムはサブスクリプション・セット・メンバーの情報をキャッシュに入れて再利用します。このようにサブスクリプション・セット・メンバーの情報を再利用すると、 CPU 使用率が減り、スループットが向上します。
delay=*n*	連続レプリケーションを使用する場合に、それぞれのアプライ・サイクルが終了した後、何秒待つかを示す遅延時間 (秒単位) を指定します。 n は、0、1、2、3、 4、 5、または 6 です。デフォルトは 6 で、これは連続レプリケーションで使用されます (つまりサブスクリプション・セットが sleep=0 分を使用する場合)。**copyonce** が指定されている場合、このパラメーターは無視されます。
errwait=*n*	アプライ・プログラムがエラー状態になった後、何秒待ってから再試行するかを示す秒数 (1 から 65535) を指定します。デフォルト値は 300 秒 (5 分) です。 **注:** アプライ・プログラムはほとんど切れ目なく稼働しており、 IBMSNAP_APPLYTRAIL 表に多くの行を生成するので、ここにあまり小さい数を指定しないでください。

表 26. *Linux*、*UNIX*、*Windows*、および *z/OS* オペレーティング・システム用 *asnapply* 呼び出しパラメーター定義 *(続き)*

パラメーター	定義
term=*y/n*	DB2 の状況がアプライ・プログラムの動作にどのように影響するかを指定します。 **y (デフォルト)** DB2 が静止または停止するとき、アプライ・プログラムは終了します。 **n** DB2 が静止または停止するとき、アプライ・プログラムは終了せず、DB2 が開始するまで待機します。
spillfile=*filetype*	フェッチした応答セットをどこに保管するかを指定します。 z/OS 有効な値は以下のとおりです。 **mem (デフォルト)** メモリー・ファイル。応答セット用の十分なメモリーがない場合、アプライ・プログラムは失敗します。 **disk** ディスク・ファイル。 Linux UNIX Windows 有効な値は以下のとおりです。 **disk (デフォルト)** ディスク・ファイル。
arm=*identifier*	z/OS 自動リスタート・マネージャーに対してアプライ・プログラムの単一インスタンスを示すために使用される 3 文字の英数字ストリングを指定します。この値は、アプライ自体が生成する ARM エレメント名: ASNTA*xxxxyyyy* (*xxxx* はデータ共用のグループ・アタッチ名、*yyyy* は DB2 メンバー名) に付加されます。 **arm** パラメーターには任意の長さのストリングを指定できますが、アプライ・プログラムは現行名に 3 文字までのみを連結します。必要であれば、アプライ・プログラムはその名前にブランクを埋め込んで 16 バイトの固有名にします。

表 26. Linux、UNIX、Windows、および z/OS オペレーティング・システム用 asnapply 呼び出しパラメーター定義 (続き)

パラメーター	定義
sqlerrcontinue=*y/n*	アプライ・プログラムがある種の SQL エラーを検出した場合、アプライ・プログラムが処理を継続するかどうかを指定します。 アプライ・プログラムは、失敗した SQLSTATE を SQLSTATE ファイルに指定された値に照らしてチェックします。この SQLSTATE ファイルは、アプライ・プログラムの実行前にユーザーが作成します。ファイルの内容と一致すれば、アプライ・プログラムは失敗した行についての情報をエラー・ファイル (apply_qualifier.ERR) に書き込み、処理を継続します。SQLSTATE ファイルには 5 バイト値を 20 個まで含めることができます。 **n (デフォルト)** アプライ・プログラムは SQLSTATE ファイルをチェックしません。 **y** アプライ・プログラムは処理中に SQLSTATE ファイルをチェックします。

戻りコード

asnapply コマンドは、正常終了したときにゼロの戻りコードを戻します。コマンドが失敗する場合、ゼロ以外の戻りコードが戻されます。

asnapply の例

以下の例は、asnapply コマンドの使用法を示しています。

例 1

アプライ修飾子 AQ1 を使用し、コントロール・サーバー名は dbx で、/home/files/apply/ ディレクトリーにある作業ファイルを使用して、アプライ・プログラムを始動する例です。

```
asnapply apply_qual=AQ1 control_server=dbx apply_path=/home/files/apply/
  pwdfile=pass1.txt
```

アプライ・プログラムは /home/files/apply/ ディレクトリーで pass1.txt という名前のパスワード・ファイルを探します。

例 2

ASNLOAD 出口ルーチンを呼び出すアプライ・プログラムを始動します。

```
asnapply apply_qual=AQ1 control_server=dbx pwdfile=pass1.txt loadxit=y
```

この例では、アプライ・プログラムは pass1.txt という名前のパスワード・ファイルを現行ディレクトリーで探します。

例 3

適格なサブスクリプション・セットごとにコピー・サイクルを 1 回実行するアプライ・プログラムを始動します。

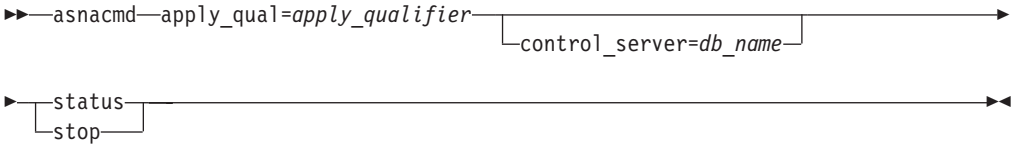
```
asnapply apply_qual=AQ1 control_server=dbx apply_path=/home/files/apply/
  copyonce=y
```

この例では、アプライ・プログラムは /home/files/apply/ ディレクトリーで、デフォルトのパスワード・ファイル (asnpwd.aut) を探します。

asnacmd: アプライの操作

Linux、UNIX、Windows、および z/OS 上の UNIX System Services (USS) でアプライ・プログラムを操作するには、asnacmd コマンドを使用します。このコマンドは、オペレーティング・システムのプロンプトまたはシェル・スクリプト内で実行します。

構文



パラメーター

表 27 は、呼び出しパラメーターを定義しています。

表 27. Linux、UNIX、Windows、および z/OS オペレーティング・システム用 asnacmd 呼び出しパラメーター定義

パラメーター	定義
apply_qual=*apply_qualifier*	アプライ・プログラムが、処理されるサブスクリプション・セットの識別に使用するアプライ修飾子を指定します。 アプライ修飾子を指定してください。入力する値は、IBMSNAP_SUBS_SET 表の APPLY_QUAL 列の値と一致する必要があります。アプライ修飾子名には大文字小文字の区別があり、最大 18 文字です。
control_server=*db_name*	サブスクリプション定義とアプライ・コントロール表が存在するアプライ・コントロール・サーバーの名前を指定します。 z/OS コントロール・サーバー・パラメーターは、コントロール・サーバーに接続するデータベース・サーバーの名前です。 Linux UNIX Windows アプライ・コントロール・サーバーを指定しない場合、このパラメーターはデフォルトで DB2DBDFT 環境変数の値になります。
status	アプライ内の各スレッド (管理およびワーカー) の状態を示すメッセージを受け取ることを指定します。

表 27. Linux、UNIX、Windows、および zOS オペレーティング・システム用 asnacmd 呼び出しパラメーター定義 (続き)

パラメーター	定義
stop	アプライ・プログラムを通常の方法で停止することを指定します。

asnacmd の例

次の例は、**asnacmd** コマンドの使用方法を示しています。

例 1

それぞれのアプライ・スレッドの状態についてメッセージを受け取る場合のコマンド例です。

```
asnacmd apply_qual=AQ1 control_server=dbx status
```

例 2

アプライ・プログラムを停止するには、次のようにします。

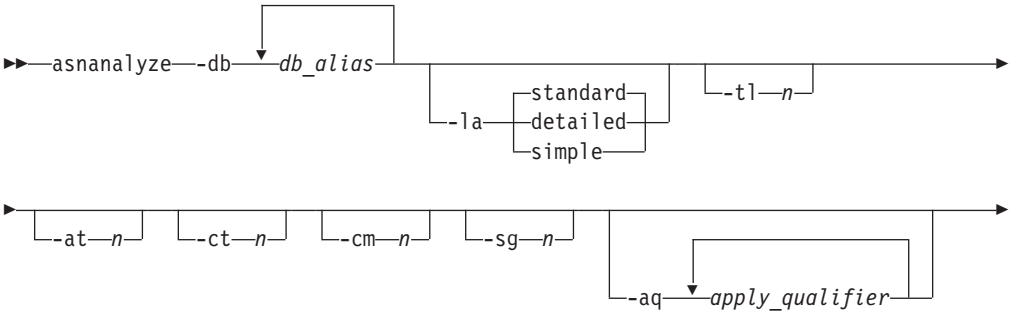
```
asnacmd apply_qual=AQ1 control_server=dbx stop
```

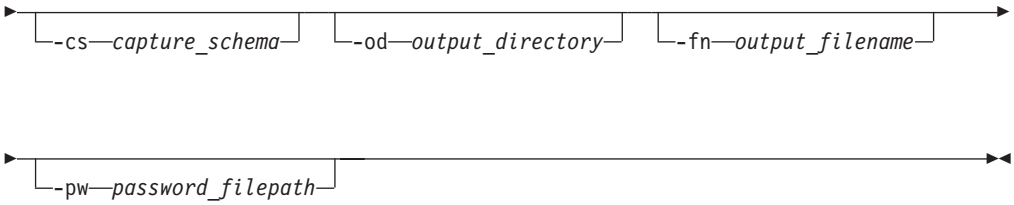
asnanalyze: アナライザーの操作

レプリケーション・コントロール表の状態についてのレポートを生成するには、asnanalyze コマンドを使用します。このコマンドは、任意のオペレーティング・システムにあるレプリケーション・コントロール表を分析します。System i でも可能ですが、コマンドは Linux、UNIX または Windows から呼び出す必要があります。

コマンドを呼び出す場合、asnanalyze コマンドと最初のパラメーターの間にはスペースを 1 つ入れる必要があります。パラメーターを指定しないでコマンドを発行すると、画面にコマンド・ヘルプが表示されます。

構文





パラメーター

表 28 は、呼び出しパラメーターを定義しています。

表 28. Linux、UNIX および Windows オペレーティング・システム用 asnanalyze 呼び出しパラメーター定義

パラメーター	定義
-db *db_alias*	キャプチャー・コントロール・サーバー、ターゲット・サーバー、およびアプライ・コントロール・サーバーを指定します。 データベース別名を少なくとも 1 つ指定する必要があります。複数のデータベース別名がある場合は、ブランク・スペースを使用して値を区切ります。
-la *level_of_analysis*	報告される分析のレベルを指定します。 **standard (デフォルト)** コントロール表の内容および、キャプチャー・プログラムおよびアプライ・プログラムからの状況情報を含むレポートを生成します。 **detailed** 標準レポートの情報に加えて以下の情報を生成します。 • 変更データ (CD) 表および作業単位 (UOW) 表の整理情報 • DB2 for z/OS 表スペースのパーティション化および圧縮情報 • サブスクリプション・キーのためのターゲット索引の分析 **simple** 標準レポートで情報を生成しますが、IBMSNAP_SUBS_COLS 表からの詳細情報は含まれません。
-tl *n*	IBMSNAP_APPLYTRAIL 表から取り出す項目の日付範囲 (0 から 30 日) を指定します。デフォルトは 3 日です。
-at *n*	アプライ・トレース IBMSNAP_APPLYTRACE 表から検索する項目の日付範囲 (0 から 30 日) を指定します。デフォルトは 3 日です。
-ct *n*	IBMSNAP_CAPTRACE 表から取り出す項目の日付範囲 (0 から 30 日) を指定します。デフォルトは 3 日です。
-cm *n*	IBMSNAP_CAPMON 表から取り出す項目の日付範囲 (0 から 30 日) を指定します。デフォルトは 3 日です。

表 28. Linux、UNIX および Windows オペレーティング・システム用 asnanalyze 呼び出しパラメーター定義 (続き)

パラメーター	定義
-sg *n*	IBMSNAP_SIGNAL 表から取り出す項目の日付範囲 (0 から 30 日) を指定します。デフォルトは 3 日です。
-aq *apply_qualifier*	分析する特定のサブスクリプション・セットを識別するアプライ修飾子を指定します。 複数のアプライ修飾子を指定することができます。複数のアプライ修飾子がある場合は、ブランク・スペースを使用して値を区切ります。アプライ修飾子を指定しない場合、指定されたデータベース別名のすべてのサブスクリプション・セットが分析されます。
-cs *capture_schema*	分析するキャプチャー・スキーマの名前を指定します。 このパラメーターを使用する場合、指定できるキャプチャー・スキーマは 1 つだけです。
-od *output_directory*	アナライザー・レポートを保管するディレクトリーを指定します。デフォルトは、現行ディレクトリーです。
-fn *output_filename*	アナライザー・レポート出力を含むファイルの名前を指定します。 アナライザーを実行するオペレーティング・システムのファイル命名規則を使用します。ファイル名がすでに存在する場合、ファイルは上書きされます。デフォルトのファイル名は、asnanalyze.htm です。
-pw *password_filepath*	パスワード・ファイルの名前とパスを指定します。このパラメーターを指定しない場合、アナライザーは現行ディレクトリーで asnpwd.aut ファイルを探します。

asnanalyze の例

以下の例は、asnanalyze コマンドの使用法を示しています。

例 1

proddb1 という名前のデータベース上のレプリケーション・コントロール表を分析する例です。

```
asnanalyze -db proddb1
```

例 2

proddb1 と proddb2 のデータベース上のレプリケーション・コントロール表について、詳細レベルの分析を入手する例です。

```
asnanalyze -db proddb1 proddb2 -la detailed
```

例 3

proddb1 と proddb2 のデータベース上の、IBMSNAP_APPLYTRAIL、IBMSNAP_APPLYTRACE、IBMSNAP_CAPTRACE、IBMSNAP_CAPMON、および IBMSNAP_SIGNAL 表から、最新の 2 日の情報を分析する例です。

```
asnanalyze -db proddb1 proddb2 -tl 2 -at 2 -ct 2 -cm 2 -sg 2
```

例 4

proddb1 および proddb2 データベース上の、IBMSNAP_APPLYTRAIL、IBMSNAP_APPLYTRACE、IBMSNAP_CAPTRACE、IBMSNAP_CAPMON、および IBMSNAP_SIGNAL 表から、qual1 および qual2 のアプライ修飾子についてのみ、最新の 2 日の情報を simple レベルで分析するには、以下のように指定します。

```
asnanalyze -db proddb1 proddb2 -la simple -tl 2 -at 2 -ct 2 -cm 2 -sg 2
  -aq qual1 qual2 -od c:¥mydir -fn anzout -pw c:¥SQLLIB
```

このコマンド例は、アナライザーの出力を c:mydir ディレクトリーの下の anzout ファイルに書き込み、 c:SQLLIB ディレクトリーからパスワード情報を使用します。

例 5

特定のキャプチャー・スキーマを分析する例です。

```
asnanalyze -db proddb1 proddb2 -cs BSN
```

例 6

コマンド・ヘルプを表示するには、以下のように入力します。

```
asnanalyze
```

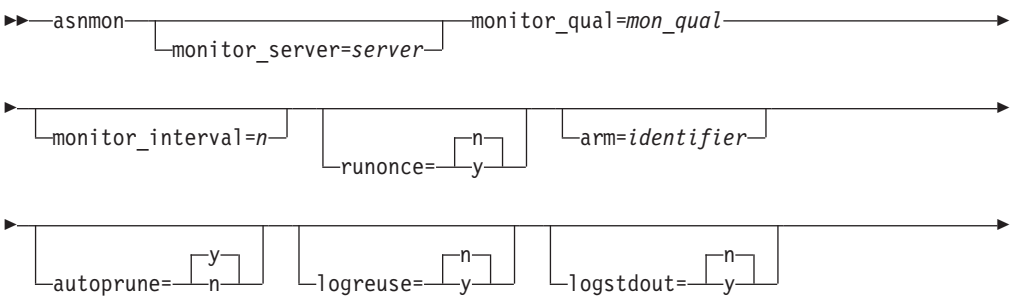
asnmon: レプリケーション・アラート・モニターの始動

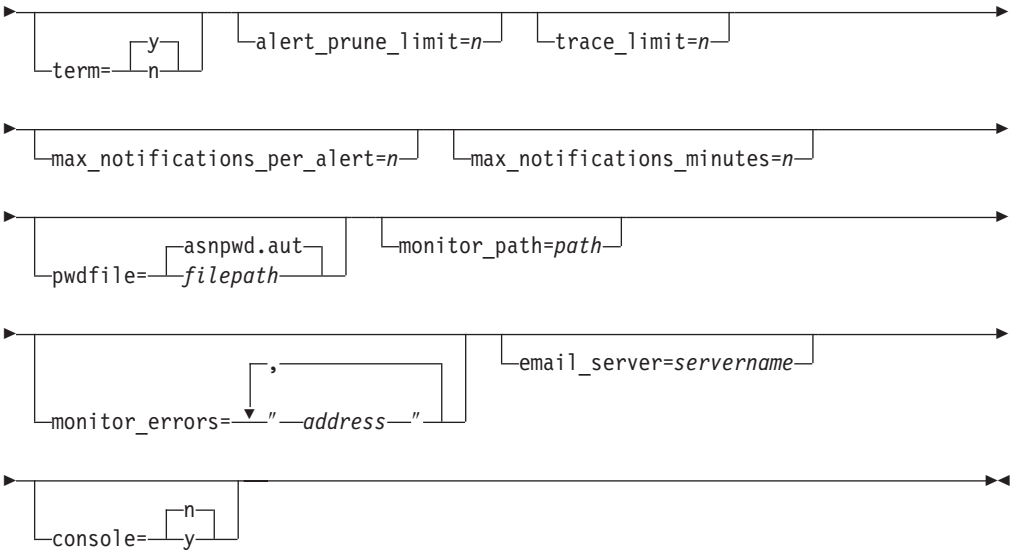
Linux、 UNIX、 Windows、および z/OS の UNIX System Services (USS) 上でレプリケーション・アラート・モニターを開始するには、asnmon コマンドを使用します。このコマンドは、オペレーティング・システムのプロンプトまたはシェル・スクリプト内で実行します。

レプリケーション・アラート・モニターは次の情報を記録します。

- Q キャプチャー、Q アプライ、キャプチャー、およびアプライの各プログラムの状況
- コントロール表に書き込まれたエラー・メッセージ
- しきい値

構文





パラメーター

表 29 は、asnmon コマンドの呼び出しパラメーターを定義します。

表 29. Linux、UNIX、Windows、および z/OS オペレーティング・システム用 asnmon 呼び出しパラメーター定義

パラメーター	定義
monitor_server=*server*	レプリケーション・アラート・モニター・プログラムを実行し、モニター・コントロール表が存在する、モニター・コントロール・サーバーの名前を指定します。これを入力する場合には、このパラメーターを必ず最初のパラメーターにします。 Linux UNIX Windows モニター・コントロール・サーバーを指定しない場合、このパラメーターはデフォルトで DB2DBDFT 環境変数の値になります。 z/OS デフォルトは DSN です。
monitor_qual=*mon_qual*	レプリケーション・アラート・モニター・プログラムが使用するモニター修飾子を指定します。モニター修飾子は、モニター対象のサーバーおよび関連するモニター条件を識別します。 モニター修飾子は必ず指定する必要があります。モニター修飾子名には大/小文字の区別があり、最大 18 文字です。

表 29. *Linux*、*UNIX*、*Windows*、および *z/OS* オペレーティング・システム用 *asnmon* 呼び出しパラメーター定義 *(続き)*

パラメーター	定義
monitor_interval=*n*	このモニター修飾子について、レプリケーション・アラート・モニター・プログラムを実行する頻度 (秒数) を指定します。デフォルトは 300 秒 (5 分) です。 **runonce** パラメーターを y に設定すると、レプリケーション・アラート・モニターはこのパラメーターを無視します。 **重要:** この **monitor_interval** パラメーターは、レプリケーション・アラート・モニター・プログラムにのみ影響を与えます。 Q キャプチャー、Q アプライ、キャプチャー、およびアプライの各プログラムには影響を与えません。
runonce=y/n	このモニター修飾子について、レプリケーション・アラート・モニター・プログラムを 1 回だけ実行するかどうかを指定します。 **n (デフォルト)** レプリケーション・アラート・モニター・プログラムは、**monitor_interval** パラメーターに指定された頻度で実行されます。 **y** レプリケーション・アラート・モニター・プログラムはモニター・サイクルを 1 回だけ実行します。 runonce パラメーターを y に設定すると、レプリケーション・アラート・モニターは **monitor_interval** パラメーターを無視します。
z/OS **arm=***identifier*	自動リストート・マネージャーに対してレプリケーション・アラート・モニター・プログラムの単一インスタンスを識別するのに使用される 3 文字の英数字ストリングを指定します。指定された値が、モニター・プログラムがそれ自体のために生成する ARM 要素名 ASNAM*xxxxyyyy* (*xxxx* はデータ共用グループ・アタッチ名、*yyyy* は DB2 メンバー名) に追加されます。 **arm** パラメーターにはどんな長さのストリングでも指定できますが、モニター・プログラムは現在の名前に最大で 3 文字だけを連結します。必要なら、モニター・プログラムは名前にブランクを埋め込んで、固有の 16 バイトの名前を作ります。
autoprune=y/n	レプリケーション・アラート・モニターのアラート (IBMSNAP_ALERTS) 表内の行の自動整理を使用可能にするかどうかを指定します。 **y (デフォルト)** レプリケーション・アラート・モニター・プログラムは、**alert_prune_limit** パラメーターの値よりも古い、IBMSNAP_ALERTS 表内の行を自動的に整理します。 **n** 自動整理は使用不可になります。

表 29. Linux、UNIX、Windows、および z/OS オペレーティング・システム用 asnmon 呼び出しパラメーター定義 (続き)

パラメーター	定義
logreuse=y/n	レプリケーション・アラート・モニター・プログラムが、診断ログ・ファイル (db2instance.monitor_server.mon_qual.MON.log) を再利用するか、またはメッセージを付加するかを指定します。 **n (デフォルト)** レプリケーション・アラート・モニター・プログラムはログ・ファイルにメッセージを付加します。 **y** レプリケーション・アラート・モニター・プログラムは、ログ・ファイルを削除し、レプリケーション・アラート・モニター・プログラムの再始動時にそれを再作成することにより、ログ・ファイルを再利用します。
logstdout=y/n	レプリケーション・アラート・モニター・プログラムがメッセージをどこに送信するかを指定します。 **n (デフォルト)** レプリケーション・アラート・モニター・プログラムはログ・ファイルにのみメッセージを送信します。 **y** レプリケーション・アラート・モニター・プログラムは、メッセージをログ・ファイルと標準出力 (stdout) の両方に送信します。
term=y/n	DB2 静止時にモニター・プログラムを実行し続けるかどうかを指定します。 **y (デフォルト)** モニター・プログラム、DB2 静止時に停止します。 **n** DB2 が静止モードで、すべてのアプリケーション (モニター・プログラムを含む) を強制的に切断している間に、モニター・プログラムは実行し続けます。 DB2 が静止モードではなくなると、モニター・プログラムは、レプリケーションのモニターに戻ります。 **term** パラメーターの設定に関係なく、モニター・プログラムは、DB2 のシャットダウン時に停止します。DB2 が再び始動するときに、モニター・プログラムを再始動する必要があります。
alert_prune_limit=*n*	レプリケーション・アラート・モニターのアラート (IBMSNAP_ALERTS) 表に行を保持する期間 (分) を指定します。この値よりも古い行はすべて整理されます。デフォルトは 10,080 分 (7 日) です。

表 29. *Linux*、*UNIX*、*Windows*、および *z/OS* オペレーティング・システム用 *asnmon* 呼び出しパラメーター定義 *(続き)*

パラメーター	定義
trace_limit=*n*	レプリケーション・アラート・モニターのトレース (IBMSNAP_MONTRACE) 表内に何分保持されるとその行が整理の対象になるかを示す分数を指定します。**trace_limit** パラメーターの値よりも古いすべての IBMSNAP_MONTRACE 行が、次の整理サイクルで整理されます。デフォルトは 10,080 分 (7 日) です。
max_notifications_per_alert=*n*	**max_notifications_minutes** パラメーター値で指定された期間中にアラートが生じた場合に、同じアラートをユーザーに送信する最大回数を指定します。このパラメーターは、同じアラートがユーザーに何回も再送されないようにするために使用します。デフォルトは 3 です。
max_notifications_minutes=*n*	このパラメーターは **max_notifications_per_alert** パラメーターと連動し、アラート条件が起きた期間を示します。デフォルトは 60 分です。
pwdfile=*filepath*	パスワード・ファイルの完全修飾名を指定します。このファイルは asnpwd コマンドを使用して定義します。デフォルトのファイル名は、asnpwd.aut です。
monitor_path=*path*	レプリケーション・アラート・モニター・プログラムが使用するログ・ファイルのロケーションを指定します。デフォルトは、asnmon コマンドが呼び出されたディレクトリーです。
monitor_errors=*address*	アラート・モニターがモニター・コントロール・サーバーに接続する前に致命的エラーが検出された場合、その通知を送信する宛先の E メール・アドレスを指定します。このパラメーターを使用して、開始パラメーターが無効である、モニター修飾子が誤っている、データベースがダウンしている、またはその他のエラーのために、モニター・コントロール・サーバー接続が失敗したという通知を送信します。 E メール・アドレスのテキストは二重引用符で囲みます。 複数の E メール・アドレスを入力することができます。E メール・アドレスはコンマで区切ってください。コンマの前後にスペースを入力してもかまいません。
email_server=*servername*	E メールのサーバー・アドレスを指定します。このパラメーターは、SMTP (Simple Mail Transfer Protocol) を指定して ASNMAIL 出口ルーチンを使用する場合にのみ入力します。

表 29. *Linux*、*UNIX*、*Windows*、および *z/OS* オペレーティング・システム用 *asnmon* 呼び出しパラメーター定義 *(続き)*

パラメーター	定義
z/OS **console=y/n**	レプリケーション・アラート・モニター・プログラムがアラート通知を z/OS コンソールに送信するかどうかを指定します。このパラメーターを y (はい) に設定し、E メール・サーバーがすでに構成されている場合には、アラートが z/OS コンソールと E メール・サーバーの両方に送信されます。 **n (デフォルト)** レプリケーション・アラート・モニター・プログラムはアラート通知を z/OS コンソールに送信しません。 **y** レプリケーション・アラート・モニター・プログラムはアラート通知を z/OS コンソールに送信します。

戻りコード

asnmon コマンドは、正常終了したときにゼロの戻りコードを戻します。コマンドが失敗する場合、ゼロ以外の戻りコードが戻されます。

asnmon の例

次の例は、asnmon コマンドの使用方法を示しています。

例 1

デフォルトのパラメーターを使用して、レプリケーション・アラート・モニターを開始します。

```
asnmon monitor_server=wsdb monitor_qual=monqual
```

例 2

指定したモニター修飾子について、120 秒 (2 分) おきに実行するレプリケーション・アラート・モニターを開始します。

```
asnmon monitor_server=wsdb monitor_qual=monqual monitor_interval=120
```

例 3

レプリケーション・アラート・モニターを開始し、指定したモニター修飾子について、1 回だけ実行することを指定します。

```
asnmon monitor_server=wsdb monitor_qual=monqual runonce=y
```

例 4

レプリケーション・アラート・モニターを開始し、モニター・エラーが検出された場合は E メールで通知を送信します。

```
asnmon monitor_server=wsdb monitor_qual=monqual
  monitor_errors="repladm@company.com, dbadmin@company.com"
```

例 5

120 秒 (2 分) おきに実行するレプリケーション・アラート・モニターを開始し、1440 分 (24 時間) 待ってからアラートを送信します。

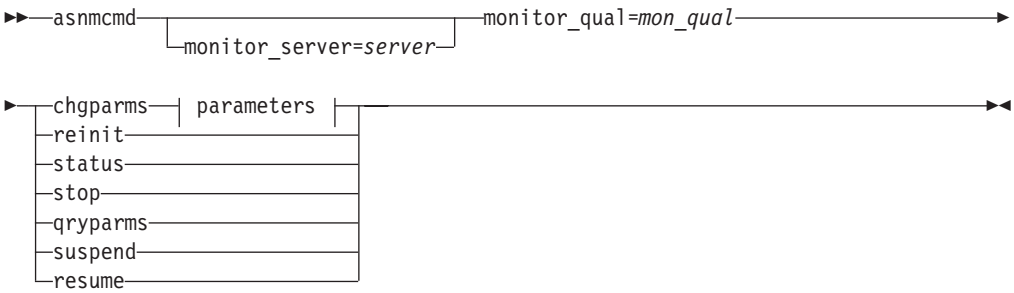
```
asnmon monitor_server=wsdb monitor_qual=monqual monitor_interval=120
  max_notifications_per_alert=2 max_notifications_minutes=1440
```

max_notifications_minutes パラメーター値 (1440 分) に指定された期間中にアラートが起こった場合、このレプリケーション・アラート・モニター・プログラムは最大 2 つのアラートを送信します。

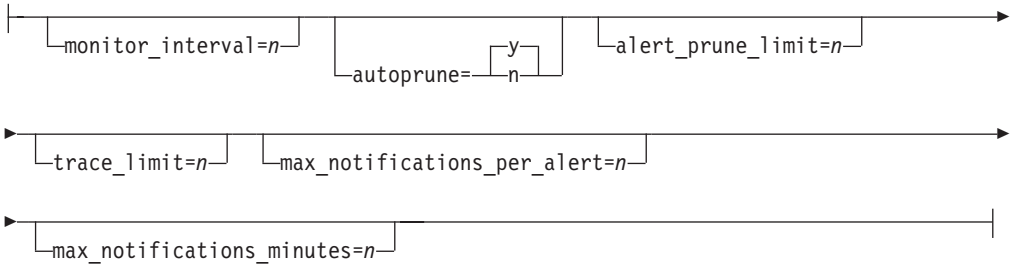
asnmcmd: 実行中のレプリケーション・アラート・モニターの処理

Linux、 UNIX、 Windows、および z/OS の UNIX System Services (USS) 上で実行中のレプリケーション・アラート・モニターにコマンドを送信するには、asnmcmd を使用します。 このコマンドは、オペレーティング・システムのプロンプトまたはシェル・スクリプト内で実行します。

構文



Parameters:



パラメーター

表 30は、asnmcmd コマンドの呼び出しパラメーターを定義します。

表 30. Linux、UNIX、Windows、および z/OS オペレーティング・システム用 asnmcmd 呼び出しパラメーター定義

パラメーター	定義
monitor_server=*server*	レプリケーション・アラート・モニター・プログラムを実行し、モニター・コントロール表が存在する、モニター・コントロール・サーバーの名前を指定します。これを入力する場合には、このパラメーターを必ず最初のパラメーターにします。 Linux UNIX Windows モニター・コントロール・サーバーを指定しない場合、このパラメーターはデフォルトで DB2DBDFT 環境変数の値になります。 z/OS デフォルトは DSN です。
monitor_qual=*mon_qual*	レプリケーション・アラート・モニター・プログラムが使用するモニター修飾子を指定します。モニター修飾子は、モニター対象のサーバーおよび関連するモニター条件を識別します。 モニター修飾子は必ず指定する必要があります。モニター修飾子名には大/小文字の区別があり、最大 18 文字です。
chgparms	レプリケーション・アラート・モニターの実行中に、以下の稼働パラメーターの 1 つ以上を変更することを指定します。 • **monitor_interval** • **autoprune** • **alert_prune_limit** • **trace_limit** • **max_notifications_per_alert** • **max_notifications_minutes** 1 つの chgparms サブコマンドに複数のパラメーターを指定したり、必要に応じてこれらのパラメーター値を変更したりできます。変更内容は IBMSNAP_MONPARMS 表の値を一時的にオーバーライドしますが、この表には保管されません。レプリケーション・アラート・モニターを停止して再始動する際には、 IBMSNAP_MONPARMS の値が使用されます。 297 ページの『asnmon: レプリケーション・アラート・モニターの始動』 には、このサブコマンドを使用してオーバーライドできるパラメーターの説明が記載されています。 **重要:** 変更するパラメーターは、chgparms サブコマンドの直後に指定する必要があります。

表 30. Linux、UNIX、Windows、および z/OS オペレーティング・システム用 asnmcmd 呼び出しパラメーター定義 (続き)

パラメーター	定義
reinit	レプリケーション・アラート・モニター・プログラムがコントロール表を読み取り、連絡先、アラート条件、およびパラメーター用にメモリー内に保持しているデータをリフレッシュすることを指定します。モニター・プログラムは、すべての値を読み取ると、サーバー上で条件チェックのサイクルを開始します。このサイクルが完了した後、**monitor_interval** に指定した時間が経過すると、次のモニター・サイクルが開始します。
status	レプリケーション・アラート・モニター内の各スレッド (管理、シリアライゼーション、およびワーカー) の状態を示すメッセージを受け取ることを指定します。
qryparms	レプリケーション・アラート・モニターの現行の稼働パラメーター値を標準出力 (stdout) に書き込む場合に指定します。
suspend	resume コマンドを発行するまで、レプリケーション・アラート・モニターによるサーバー上での条件チェックを一時的に停止する場合に指定します。
resume	レプリケーション・アラート・モニターが中断されていて、このモニター・プログラムによるサーバー上での条件チェックを再び開始する場合に指定します。
stop	レプリケーション・アラート・モニターを通常の方法で停止することを指定します。

asnmcmd の例

次の例は、asnmcmd コマンドの使用方法を示しています。

例 1

指定したモニター修飾子について、レプリケーション・アラート・モニターを停止します。

```
asnmcmd monitor_server=wsdb monitor_qual=monqual stop
```

例 2

レプリケーション・アラート・モニターのスレッドの状態を示すメッセージを受け取ります。

```
asnmcmd monitor_server=wsdb monitor_qual=monqual status
```

例 3

モニター・コントロール表からの現行値を使用して、レプリケーション・アラート・モニターをリフレッシュします。

```
asnmcmd monitor_server=wsdb monitor_qual=monqual reinit
```

例 4

指定期間中にレプリケーション・アラート・モニターが送信する通知の最大数を、デフォルトの 3 より少なくします。

```
asnmcmd monitor_server=wsdb monitor_qual=monqual
chgparms max_notifications_per_alert=2
```

例 5

レプリケーション・アラート・モニターの現行稼働パラメーター値を標準出力に送信します。

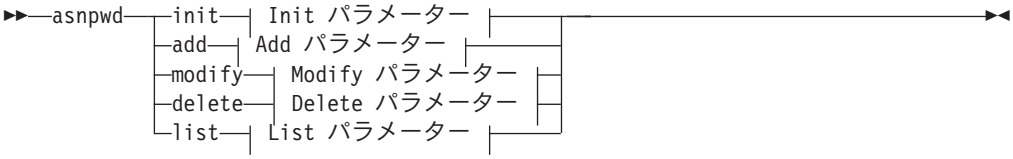
```
asnmcmd monitor_server=wsdb monitor_qual=monqual qryparms
```

asnpwd: パスワード・ファイルの作成および保守

Linux、UNIX、および Windows のパスワード・ファイルの作成および変更は、asnpwd コマンドを使用して行います。このコマンドは、コマンド行またはシェル・スクリプト内で実行します。

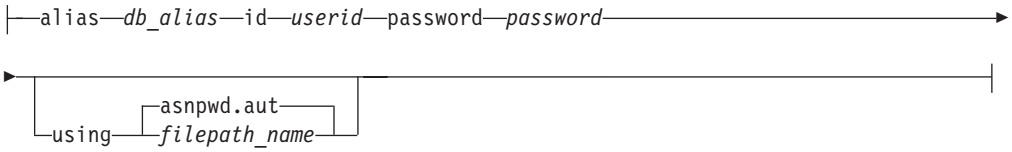
パラメーターを指定せずに asnpwd コマンドを入力したり、このコマンドの後に *?* を指定して実行したり、コマンドの後に誤ったパラメーターを指定して実行すると、コマンド・ヘルプが表示されます。

構文

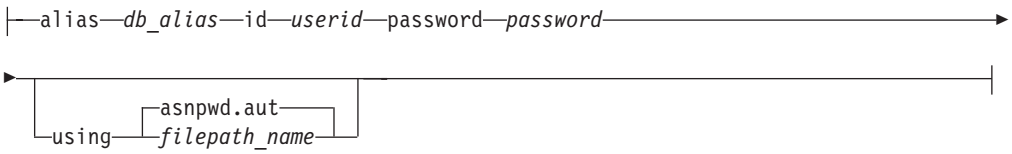


Init パラメーター:

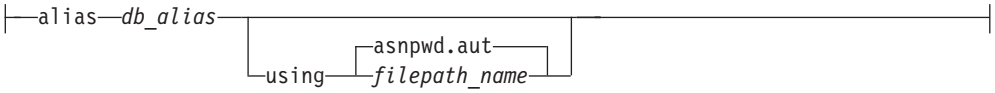
Add パラメーター:



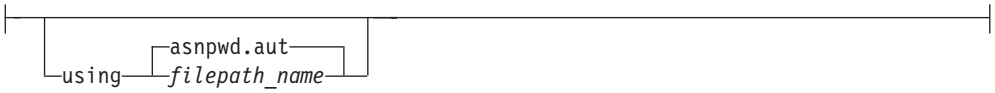
Modify パラメーター:



Delete パラメーター:



List パラメーター:



パラメーター

表 31 は、asnpwd コマンドの呼び出しパラメーターを定義します。

表 31. Linux、UNIX、および Windows オペレーティング・システム用 asnpwd 呼び出しパラメーター定義

パラメーター	定義
init	空のパスワード・ファイルを作成することを指定します。すでに存在するパスワード・ファイルに **init** パラメーターを指定すると、このコマンドは失敗します。
add	パスワード・ファイルに項目を追加することを指定します。パスワード・ファイルにすでに存在する項目に **add** パラメーターを指定すると、このコマンドは失敗します。パスワード・ファイル内の既存の項目を変更するには、 **modify** パラメーターを使用します。
modify	パスワード・ファイル内の項目のパスワードまたはユーザー ID を変更することを指定します。
delete	パスワード・ファイルから項目を削除することを指定します。
list	パスワード・ファイル内の別名およびユーザー ID の項目をリストするよう指定します。このパラメーターは、パスワード・ファイルが **encrypt** パラメーターを使用して作成された場合のみ使用できます。パスワードを **list** コマンドによって表示することはできません。
encrypt	暗号化するファイル内の項目を指定します。 **all (デフォルト)** 指定したファイル内のすべての項目を暗号化し、ファイル内のデータベース別名、ユーザー名、およびパスワードをリストできないようにします。このオプションにより、パスワード・ファイル内の情報漏れを削減できます。 **password** 指定したファイル内のパスワードの項目を暗号化します。このオプションでは、ユーザーは、パスワード・ファイル内に保管されているデータベース別名およびユーザー名をリストできます。パスワードを表示することはできません。

表 31. Linux、UNIX、および Windows オペレーティング・システム用 asnpwd 呼び出しパラメーター定義 (続き)

パラメーター	定義
using *filepath_name*	パスワード・ファイルのパスと名前を指定します。ご使用のオペレーティング・システムのファイル名規則に従ってください。 Windows 上の有効なパスワード・ファイルの例は、C:¥sqllib¥mypwd.aut です。 パスワード・ファイルのパスと名前を指定する場合、そのパスおよびパスワード・ファイルはすでに存在するものでなければなりません。 **init** パラメーターを使用し、パスワード・ファイルのパスと名前を指定する場合、そのパスはすでに存在するものでなければならず、コマンドはユーザーに代わってパスワード・ファイルを作成します。 このパラメーターを指定しない場合、デフォルトのファイル名は asnpwd.aut、デフォルトのファイル・パスは現行ディレクトリーです。
alias *db_alias*	ユーザー ID がアクセス権を持つデータベースの別名を指定します。別名は、どのように入力しても必ず英大文字になります。
id *userid*	データベースへのアクセス権を持つユーザー ID を指定します。
password *password*	指定したユーザー ID のパスワードを指定します。このパスワードには大文字小文字の区別があり、パスワード・ファイル内では暗号化されます。

戻りコード

asnpwd コマンドは、正常終了したときにゼロの戻りコードを戻します。コマンドが失敗する場合、ゼロ以外の戻りコードが戻されます。

asnpwd の例

次の例は、asnpwd コマンドの使用方法を示しています。

例 1

現行ディレクトリーに、デフォルト名 asnpwd.aut を使用してパスワード・ファイルを作成します。

```
asnpwd INIT
```

例 2

c:¥myfiles ディレクトリーに pass1.aut という名前のパスワード・ファイルを作成します。

```
asnpwd INIT USING c:¥myfiles¥pass1.aut
```

例 3

encrypt all パラメーターを使用して mypwd.aut という名前のパスワード・ファイルを作成するには、次のようにします。

```
asnpwd INIT ENCRYPT ALL USING mypwd.aut
```

例 4

encrypt password パラメーターを使用して mypwd.aut という名前のパスワード・ファイルを作成するには、次のようにします。

```
asnpwd INIT ENCRYPT PASSWORD USING mypwd.aut
```

例 5

encrypt password パラメーターを使用してデフォルトのパスワード・ファイルを作成するには、次のようにします。

```
asnpwd INIT ENCRYPT PASSWORD
```

例 6

oneuser というユーザー ID とそのパスワードを、c:¥myfiles ディレクトリー内の pass1.aut という名前のパスワード・ファイルに追加し、このユーザー ID に db1 データベースへのアクセス権を付与します。

```
asnpwd ADD ALIAS db1 ID oneuser PASSWORD mypwd using c:¥myfiles¥pass1.aut
```

例 7

c:¥myfiles ディレクトリー内の pass1.aut という名前のパスワード・ファイル内の項目のユーザー ID またはパスワードを変更します。

```
asnpwd MODIFY AliaS sample ID chglocalid PASSWORD chgmajorpwd
  USING c:¥myfiles¥pass1.aut
```

例 8

c:¥myfiles ディレクトリー内の pass1.aut という名前のパスワード・ファイルから、sample というデータベース別名を削除します。

```
asnpwd delete alias sample USING c:¥myfiles¥pass1.aut
```

例 9

コマンド・ヘルプを表示します。

```
asnpwd
```

例 10

デフォルトのパスワード・ファイル内の項目をリストします。

```
asnpwd LIST
```

例 11

pass1.aut という名前のパスワード・ファイル内の項目をリストします。

```
asnpwd LIST USING pass1.aut
```

このコマンドからの出力は、パスワード・ファイルの初期設定方法によって異なります。

- **encrypt all** パラメーターを使用して初期設定された場合、以下のメッセージが発行されます。

```
ASN1986E "Asnpwd" : "". The password file "pass1.aut" contains
encrypted information that cannot be listed.
```

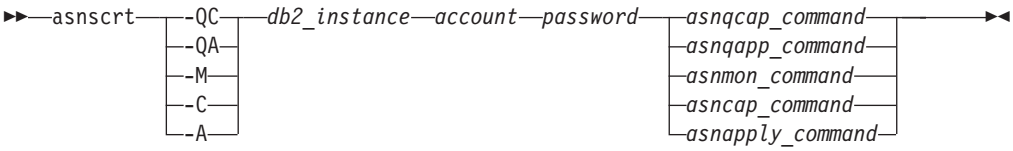
- **encrypt all** パラメーターを使用して初期設定されなかった場合、以下の詳細がリストされます。

```
asnpwd LIST USING pass1.aut
Alias: SAMPLE  ID: chglocalid
Number of Entries: 1
```

asnscrt: レプリケーション・サービスの作成

Windows の Service Control Manager (SCM) にレプリケーション・サービスを作成するには、asnscrt コマンドを使用し、asnqcap、asnqapp、asnmon、asncap、および asnapply コマンドを呼び出します。 asnscrt コマンドは、Windows オペレーティング・システムで実行します。

構文



パラメーター

表 32は、asnscrt コマンドの呼び出しパラメーターを定義します。

表 32. Windows オペレーティング・システム用の asnscrt 呼び出しパラメーター定義

パラメーター	定義
-QC	Q キャプチャー・プログラムの始動を指定します。
-QA	Q アプライ・プログラムの始動を指定します。
-M	レプリケーション・アラート・モニター・プログラムの始動を指定します。
-C	キャプチャー・プログラムの始動を指定します。
-A	アプライ・プログラムの始動を指定します。
db2_instance	固有の DB2 レプリケーション・サービスを識別するために使用される DB2 インスタンスを指定します。DB2 インスタンス名は最大 8 文字です。
account	Windows へのログオンに使用するアカウント名を指定します。アカウントがローカルの場合、ピリオドと円記号 (.¥) で始まる必要があります。それ以外の場合、ドメイン名またはマシン名を指定する必要があります (例、domain_name¥account_name)。

表 32. Windows オペレーティング・システム用の asnscrt 呼び出しパラメーター定義 (続き)

パラメーター	定義
password	このアカウント名で使用するパスワードを指定します。パスワードに特殊文字が含まれる場合は、各特殊文字の前に円記号 (¥) を入力します。
asnqcap_command	Q キャプチャー・プログラムを始動するための完全な asnqcap コマンドを指定します。文書化されている asnqcap コマンド構文と該当の asnqcap パラメーターを使用します。 DB2PATH 環境変数が定義されていない場合は、asnqcap コマンドに **capture_path** パラメーターを含めて、作業ファイルのロケーションを指定する必要があります。 DB2PATH 変数が定義されている状態で **capture_path** を指定すると、**capture_path** パラメーターは DB2PATH 変数をオーバーライドします。 asnscrt コマンドは、ユーザーが入力した asnqcap パラメーターの構文の妥当性を検査しません。
asnqapp_command	Q アプライ・プログラムを始動するための完全な asnqapp コマンドを指定します。文書化されている asnqapp コマンド構文と該当の asnqapp パラメーターを使用します。 DB2PATH 環境変数が定義されていない場合は、asnqapp コマンドに **apply_path** パラメーターを含めて、作業ファイルのロケーションを指定する必要があります。 DB2PATH 変数が定義されている状態で **apply_path** を指定すると、**apply_path** パラメーターは DB2PATH 変数をオーバーライドします。 asnscrt コマンドは、ユーザーが入力した asnqapp パラメーターの構文の妥当性を検査しません。
asnmon_command	レプリケーション・アラート・モニター・プログラムを始動するための完全な asnmon コマンドを指定します。文書化されている asnmon コマンド構文と該当の asnmon パラメーターを使用します。 DB2PATH 環境変数が定義されていない場合は、asnmon コマンドに **monitor_path** パラメーターを含めて、ログ・ファイルのロケーションを指定する必要があります。 DB2PATH 変数が定義されている状態で **monitor_path** を指定すると、**monitor_path** パラメーターは DB2PATH 変数をオーバーライドします。 asnscrt コマンドは、ユーザーが入力した asnmon パラメーターの構文の妥当性を検査しません。

表 32. *Windows* オペレーティング・システム用の *asnscrt* 呼び出しパラメーター定義 *(続き)*

パラメーター	定義
asncap_command	キャプチャー・プログラムを始動するための完全な asncap コマンドを指定します。文書化されている asncap コマンド構文と該当の asncap パラメーターを使用します。 DB2PATH 環境変数が定義されていない場合は、asncap コマンドに **capture_path** パラメーターを含めて、作業ファイルのロケーションを指定する必要があります。 DB2PATH 変数が定義されている状態で **capture_path** を指定すると、**capture_path** パラメーターは DB2PATH 変数をオーバーライドします。 asnscrt コマンドは、ユーザーが入力した asncap パラメーターの構文の妥当性を検査しません。
asnapply_command	アプライ・プログラムを始動するための完全な asnapply コマンドを指定します。文書化されている asnapply コマンド構文と該当の asnapply パラメーターを使用します。 DB2PATH 環境変数が定義されていない場合は、asnapply コマンドに **apply_path** パラメーターを含めて、作業ファイルのロケーションを指定する必要があります。 DB2PATH 変数が定義されている状態で **apply_path** を指定すると、**apply_path** パラメーターは DB2PATH 変数をオーバーライドします。 asnscrt コマンドは、ユーザーが入力した asnapply パラメーターの構文の妥当性を検査しません。

asnscrt の例

次の例は、asnscrt コマンドの使用方法を示しています。

例 1

inst2 という名前の DB2 インスタンスの下で、ログオン・アカウント .¥joesmith およびパスワード my$pwd を使用して、Q アプライ・プログラムを呼び出す DB2 レプリケーション・サービスを作成するには、次のように指定します。

```
asnscrt -QA inst2 .¥joesmith my¥$pwd asnqapp apply_server=mydb2 apply_schema =as2
  apply_path=X:¥sqllib
```

例 2

inst1 という名前の DB2 インスタンスの下でキャプチャー・プログラムを呼び出す DB2 レプリケーション・サービスを作成するには、次のように指定します。

```
asnscrt -C inst1 .¥joesmith password asncap capture_server=sampledb
  capture_schema=ASN capture_path=X:¥logfiles
```

例 3

inst2 という名前の DB2 インスタンスの下で、ログオン・アカウント .¥joesmith およびパスワード my$pwd を使用して、アプライ・プログラムを呼び出す DB2 レプリケーション・サービスを作成するには、次のように指定します。

```
asnscrt -A inst2 .¥joesmith my¥$pwd asnapply control_server=db2 apply_qual=aq2
  apply_path=X:¥sqllib
```

例 4

inst3 という名前の DB2 インスタンスの下でレプリケーション・アラート・モニター・プログラムを呼び出す DB2 レプリケーション・サービスを作成するには、次のように指定します。

```
asnscrt -M inst3 .¥joesmith password asnmon monitor_server=db3 monitor_qual=mq3
  monitor_path=X:¥logfiles
```

例 5

inst4 という名前の DB2 インスタンスの下でキャプチャー・プログラムを呼び出す DB2 レプリケーション・サービスを作成し、デフォルトの作業ファイル・ディレクトリーを完全修飾された **capture_path** でオーバーライドするには、次のように指定します。

```
asnscrt -C inst4 .¥joesmith password X:¥sqllib¥bin¥asncap capture_server=scdb
  capture_schema=ASN capture_path=X:¥logfiles
```

例 6

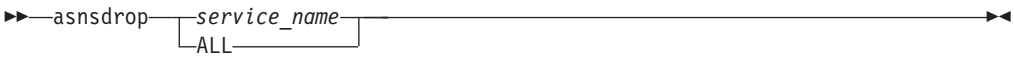
inst1 という名前の DB2 インスタンスの下で Q キャプチャー・プログラムを呼び出す DB2 レプリケーション・サービスを作成するには、次のように指定します。

```
asnscrt -QC inst1 .¥joesmith password asnqcap capture_server=mydb1
  capture_schema=QC1 capture_path=X:¥logfiles
```

asnsdrop: レプリケーション・サービスのドロップ

Windows オペレーティング・システムの Windows Service Control Manager (SCM) から、レプリケーション・サービスをドロップするには、asnsdrop コマンドを使用します。 (レプリケーション・サービスの作成には asnscrt コマンドを使用します。)

構文



パラメーター

表 33は、asnsdrop コマンドの呼び出しパラメーターを定義します。

表 33. Windows オペレーティング・システム用の asnsdrop 呼び出しパラメーター定義

パラメーター	定義
service_name	DB2 レプリケーション・サービスの完全修飾名を指定します。DB2 レプリケーション・サービス名を知るには、Windows SCM に入ります。Windows オペレーティング・システムでは、DB2 レプリケーション・サービスの「プロパティー」ウィンドウをオープンするとサービス名がわかります。 DB2 レプリケーション・サービス名にスペースが含まれる場合は、サービス名全体を二重引用符で囲んでください。
ALL	すべての DB2 レプリケーション・サービスをドロップすることを指定します。

asnsdrop の例

次の例は、asnsdrop コマンドの使用方法を示しています。

例 1

DB2 レプリケーション・サービスをドロップするには、次のようにします。

```
asnsdrop DB2.SAMPLEDB.SAMPLEDB.CAP.ASN
```

例 2

A S N (埋め込まれたブランクあり) というスキーマ名の DB2 レプリケーション・サービスをドロップするには、サービス名を二重引用符で囲みます。

```
asnsdrop "DB2.SAMPLEDB.SAMPLEDB.CAP.A S N"
```

例 3

すべてのDB2レプリケーション・サービスをドロップするには、次のようにします。

```
asnsdrop ALL
```

asnslist: レプリケーション・サービスのリスト

Windows の Service Control Manager (SCM) 中のレプリケーション・サービスをリストするには、asnslist コマンドを使用します。オプションで、このコマンドを使用して、個々のサービスに関する詳細情報をリストできます。 asnslist コマンドは、Windowsオペレーティング・システムで実行します。

構文

パラメーター

表 34は、asnslist コマンドの呼び出しパラメーターを定義します。

表 34. Windows オペレーティング・システム用の asnslist 呼び出しパラメーター定義

パラメーター	定義
details	システム上のすべての DB2レプリケーション・サービスに関する詳細データをリストすることを指定します。

asnlist の例

次の例は、asnslistコマンドの使用方法を示しています。

例 1

システム上の DB2 レプリケーション・サービスの名前をリストするには、以下のようにします。

```
asnslist
```

以下にコマンド出力の例を示します。

```
DB2.DB2.SAMPLE.QAPP.ASN
DB2.DB4.SAMPLE.QCAP.ASN
```

例 2

システム上のすべてのサービスに関する詳細情報をリストするには、以下のようにします。

```
asnslist details
```

以下にコマンド出力の例を示します。

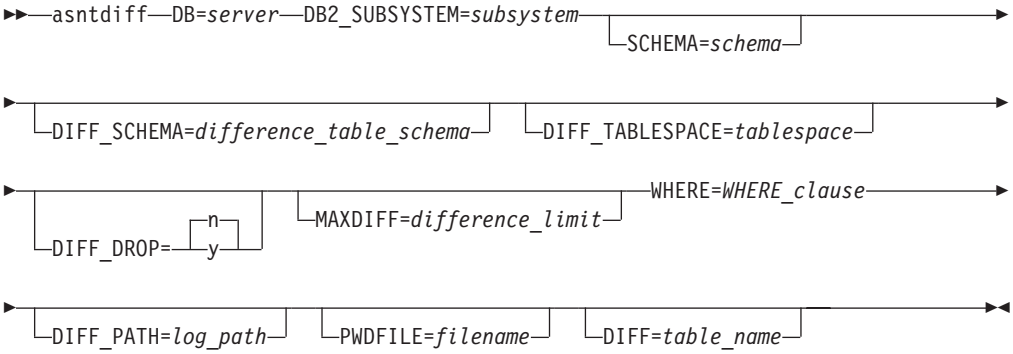
```
DB2.DB2.SAMPLE.QAPP.ASN
Display Name: DB2 DB2 SAMPLE QAPPLY ASN
Image Path:   ASNSERV DB2.DB2.SAMPLE.APP.AQ1 -ASNQAPPLY QAPPLY_SERVER=SAMPLE AP
              PLY_SCHEMA=ASN QAPPLY_PATH=C:¥PROGRA~1¥SQLLIB
Dependency:   DB2-0

DB2.DB4.SAMPLE.QCAP.ASN
Display Name: DB2 DB4 SAMPLE QAPPLY ASN
Image Path:   ASNSERV DB2.DB4.SAMPLE.APP.AQ1 -ASNQCAP QCAPTURE_SERVER=SAMPLE CA
              PTURE_SCHEMA=ASN QCAPTURE_PATH=C:¥PROGRA~1¥SQLLIB
Dependency:   DB4-0
```

asntdiff: ソース表とターゲット表とのデータの比較

ソース表をターゲット表と比較して両者の間の違いのリストを生成するには、asntdiff コマンドを使用します。asntdiff コマンドは、Linux、 UNIX、 Windows、または z/OS 上のオペレーティング・システム上のシステム・プロンプトかシェル・スクリプトで実行してください。

asntdiff コマンドは、Linux、UNIX、Windows、z/OS、および System i オペレーティング・システム上で DB2 表を比較します。

構文



パラメーター

表 35 は、asntdiff コマンドの呼び出しパラメーターを定義します。

表 35. Linux、UNIX、Windows、および z/OS オペレーティング・システム用 asntdiff 呼び出しパラメーター定義

パラメーター	定義
DB=*server*	比較対象のソース表とターゲット表に関する情報を保管するデータベースの DB2 別名を指定します。この値は、 Q レプリケーションか SQL レプリケーションのどちらを使用するかに応じて変わります。 **Q レプリケーション** IBMQREP_SUBS 表を含む Q キャプチャー・サーバーの名前。 z/OS IBMQREP_SUBS 表を含む Q キャプチャー・サーバーのロケーション名。 **SQL レプリケーション** IBMSNAP_SUBS_MEMBR 表を含むアプライ・コントロール・サーバーの名前。 z/OS IBMSNAP_SUBS_MEMBR 表を含むアプライ・コントロール・サーバーのロケーション名。
DB2_SUBSYSTEM=*subsystem*	z/OS asntdiff ユーティリティーを実行するサブシステムの名前を指定します。
SCHEMA=*schema*	Q レプリケーションの場合は Q キャプチャー・コントロール表のスキーマを指定し、SQL レプリケーションの場合はアプライ・コントロール表のスキーマを指定します。デフォルトは ASN です。
DIFF_SCHEMA= *difference_table_schema*	相違検出表を修飾するスキーマを指定します。デフォルトは ASN です。

表 35. *Linux*、*UNIX*、*Windows*、および *z/OS* オペレーティング・システム用 *asntdiff* 呼び出しパラメーター定義 *(続き)*

パラメーター	定義
DIFF_TABLESPACE=*tablespace*	相違検出表を入れる表スペースを指定します。このパラメーターを指定しないと、asntdiff コマンドを実行したデータベースまたはサブシステムのデフォルトの表スペースにこの表が作成されます。 z/OS これは、*dbname.tablespace* という 2 部構成の名前であり、*dbname* は論理データベース名、*tablespace* は表スペース名です。
DIFF_DROP=y/n	既存の相違検出表を使用して相違を記録する前に、この表をドロップして再作成するかどうかを指定します。表が既存でない場合は、asntdiff コマンドにより作成されます。 **n (デフォルト)** 相違検出表が現状のまま使用され、既存の行が削除されます。 **y** 相違検出表がドロップされて再作成されます。
MAXDIFF=*difference_limit*	asntdiff コマンドが停止するまでの間に処理する相違の最大数を指定します。デフォルト値は 10000 です。
WHERE=*WHERE_clause*	比較対象のソース表とターゲット表に関する情報を保管するコントロール表の 1 行をユニークに識別する、 SQL WHERE 文節を指定します。 WHERE 文節は、二重引用符で囲まなければなりません。このパラメーターの値は、 Q レプリケーションか SQL レプリケーションのどちらを使用するかに応じて変わります。 **Q レプリケーション** WHERE 文節は IBMQREP_SUBS 表中の行を指定します。この表は SUBNAME 列を使用して、ソース表とターゲット表を含む Q サブスクリプションを識別します。 **SQL レプリケーション** WHERE 文節は IBMSNAP_SUBS_MEMBR 表中の行を指定します。この表は SET_NAME 列、APPLY_QUAL 列、TARGET_SCHEMA 列と TARGET_TABLE 列を使用して、ソース表とターゲット表を含むサブスクリプション・セットのメンバーを識別します。
DIFF_PATH=*log_path*	asntiff ユーティリティーのログの作成先となるロケーションを指定します。デフォルト値は、このコマンドを実行したディレクトリーです。この値は、絶対パス名でなければなりません。大文字小文字を区別する場合は二重引用符 (″″) を使用してください。

表 35. Linux、UNIX、Windows、および z/OS オペレーティング・システム用 asntdiff 呼び出しパラメーター定義 (続き)

パラメーター	定義
PWDFILE=*filename*	データベースとの接続に使用されるパスワード・ファイルの名前を指定します。パスワード・ファイルを指定しない場合、デフォルト値は asnpwd.aut (asnpwd コマンドによって作成されるパスワード・ファイルの名前) です。asntdiff コマンドは、DIFF_PATH パラメーターで指定されたディレクトリー内でパスワード・ファイルを探します。DIFF_PATH パラメーターを指定しない場合、このコマンドは、コマンドを実行したディレクトリー内でパスワード・ファイルを探します。
DIFF=*table_name*	ソース・データベース中に作成され、ソース表とターゲット表の間の違いを保管する表の名前を指定します。この表には、違いが検出されるごとに 1 行ずつ含まれていきます。このパラメーターを組み込まない場合、相違検出表の名前は ASN.ASNTDIFF になります。

asntdiff の例

次の例は、asntdiff コマンドの使用方法を示しています。

z/OS asntdiff コマンドを実行するサンプル JCL については、asntdiff サンプル・プログラムを参照してください。

例 1

Q レプリケーションにおいて、Q キャプチャー・スキーマが asn の Q キャプチャー・サーバー source_db 上で、my_qsub という Q サブスクリプションに指定されているソース表とターゲット表の間の違いを検出するには、以下のようにします。

```
asntdiff db=source_db schema=asn where="subname = 'my_qsub'"
```

例 2

SQL レプリケーションにおいて、アプライ・スキーマが asn のアプライ・コントロール・サーバー apply_db 上で、my_set というサブスクリプション・セットに指定されているソース表とターゲット表 trg_table の間の違いを検出し、相違検出表の名前を diff_table にするには、以下のようにします。

```
asntdiff DB=apply_db schema=asn where="set_name = 'my_set'
 and target_table = 'trg_table'" diff=diff_table
```

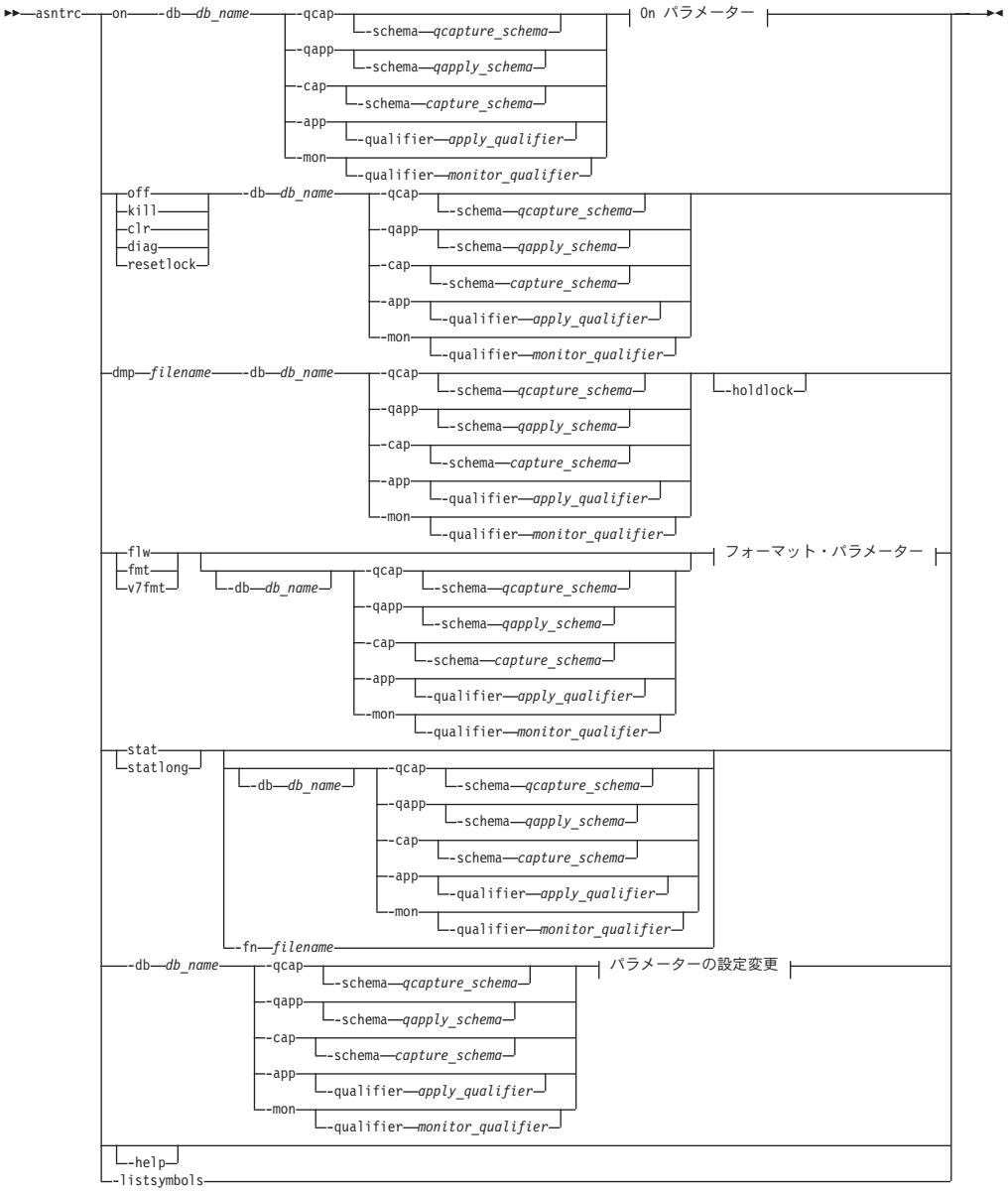
asntrc: レプリケーション・トレース機能の操作

Linux、UNIX、Windows、および z/OS 上の UNIX System Services (USS) でトレース機能を実行するには、asntrc コマンドを使用します。トレース機能は、Q キャプチャー・プログラム、Q アプライ・プログラム、キャプチャー・プログラム、アプライ・プログラムおよびレプリケーション・アラート・モニター・プログラムから、プログラム・フロー情報をログに記録します。このトレース情報を IBM ソフ

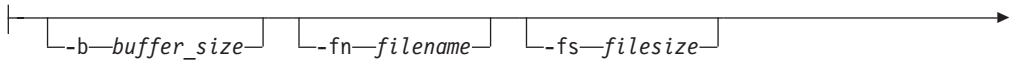
トウェア・サポートに提供して、トラブルシューティングに役立てることができます。このコマンドは、オペレーティング・システムのプロンプトまたはシェル・スクリプトで実行します。

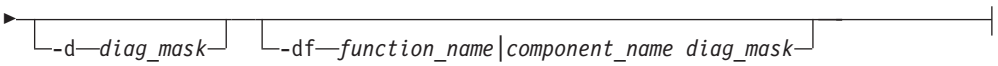
このコマンドは、オペレーティング・システムのプロンプトまたはシェル・スクリプト内で実行します。

構文

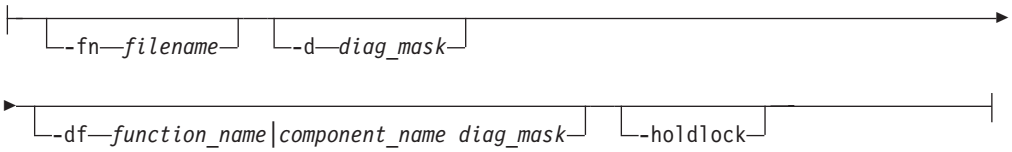


On パラメーター:

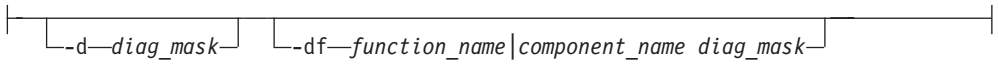




フォーマット・パラメーター:



パラメーターの設定変更:



パラメーター

表 36 は、asntrc コマンドの呼び出しパラメーターを定義します。

表 36. Linux、UNIX、Windows、および z/OS オペレーティング・システム用 asntrc 呼び出しパラメーター定義

パラメーター	定義
on	特定の Q キャプチャー・プログラム、Q アプライ・プログラム、キャプチャー・プログラム、アプライ・プログラム、またはレプリケーション・アラート・モニター・プログラムについて、トレース機能をオンにすることを指定します。トレース機能は、トレース処理中に使用する共有メモリー・セグメントを作成します。
-db *db_name*	トレースするデータベースの名前を指定します。 • トレースする Q キャプチャー・プログラムの Q キャプチャー・サーバーの名前を指定します。 • トレースする Q アプライ・プログラムの Q アプライ・サーバーの名前を指定します。 • トレースするキャプチャー・プログラムのキャプチャー・コントロール・サーバーの名前を指定します。 • トレースするアプライ・プログラムのアプライ・コントロール・サーバーの名前を指定します。 • トレースするレプリケーション・アラート・モニター・プログラム用のモニター・コントロール・サーバーの名前を指定します。
-qcap	Q キャプチャー・プログラムをトレースすることを指定します。 Q キャプチャー・プログラムは **-schema** パラメーターで識別されます。
-schema *qcapture_schema*	トレースする Q キャプチャー・プログラムの名前を指定します。Q キャプチャー・プログラムは指定した Q キャプチャー・スキーマにより識別されます。このパラメーターは **-qcap** パラメーターと一緒に使用します。

表 36. Linux、UNIX、Windows、および z/OS オペレーティング・システム用 asntrc 呼び出しパラメーター定義 (続き)

パラメーター	定義
-qapp	Q アプライ・プログラムをトレースすることを指定します。 Q アプライ・プログラムは **-schema** パラメーターで識別されます。
-schema *qapply_schema*	トレースする Q アプライ・プログラムの名前を指定します。Q アプライ・プログラムは指定した Q アプライ・スキーマにより識別されます。このパラメーターは **-qapp** パラメーターと一緒に使用します。
-cap	キャプチャー・プログラムをトレースすることを指定します。キャプチャー・プログラムは **-schema** パラメーターで識別されます。
-schema *capture_schema*	トレースするキャプチャー・プログラムの名前を指定します。キャプチャー・プログラムは指定したキャプチャー・スキーマにより識別されます。このパラメーターは **-cap** パラメーターと一緒に使用します。
-app	アプライ・プログラムをトレースすることを指定します。アプライ・プログラムは **-qualifier** パラメーターで識別されます。
-qualifier *apply_qualifier*	トレースするアプライ・プログラムの名前を指定します。このアプライ・プログラムは、指定したアプライ修飾子により識別されます。このパラメーターは **-app** パラメーターと一緒に使用します。
-mon	レプリケーション・アラート・モニター・プログラムをトレースすることを指定します。レプリケーション・アラート・モニター・プログラムは **-qualifier** パラメーターで識別されます。
-qualifier *monitor_qualifier*	トレースするレプリケーション・アラート・モニター・プログラムの名前を指定します。このレプリケーション・アラート・モニター・プログラムは、指定したモニター修飾子により識別されます。このパラメーターは **-mon** パラメーターと一緒に使用します。
off	特定の Q キャプチャー・プログラム、Q アプライ・プログラム、キャプチャー・プログラム、アプライ・プログラム、またはレプリケーション・アラート・モニター・プログラムについて、トレース機能をオフにし、使用中の共有メモリー・セグメントを解放することを指定します。
kill	トレース機能を強制的に異常終了させることを指定します。 このパラメーターは、何らかの問題により、トレース機能を **off** パラメーターでオフにできない場合のみ使用してください。
clr	トレース・バッファーをクリアすることを指定します。このパラメーターは、トレース・バッファーの内容を消去しますが、バッファーはアクティブのままにします。

表 36. Linux、UNIX、Windows、および z/OS オペレーティング・システム用 asntrc 呼び出しパラメーター定義 (続き)

パラメーター	定義
diag	トレース機能の実行中に、フィルター設定を表示することを指定します。
resetlock	トレース機能のバッファー・ラッチを解放することを指定します。このパラメーターは、エラー状態が起こり、トレース・プログラムがバッファー・ラッチを保留したまま終了した場合に、バッファー・ラッチをエラー状態からリカバリーできるようにします。
dmp *filename*	トレース・バッファーの現在の内容をファイルに書き込むことを指定します。
-holdlock	トレース機能がバッファーをコピーするためのメモリーが不足している場合でも、ロックを保留している間に、トレース機能がファイルのダンプまたはコマンドの出力を完了できることを指定します。
flw	トレース機能が作成し、共有メモリーまたはファイルに保管したサマリー情報を表示することを指定します。この情報には、プログラム・フローが含まれ、それぞれの処理およびスレッドごとに、関数と呼び出しのスタック構造がわかるように字下げして表示されます。
fmt	トレース機能が作成し、共有メモリーまたはファイルに保管した詳細情報を表示することを指定します。このパラメーターは、トレースしたデータ構造の内容全体を発生順に表示します。
v7fmt	トレース機能が作成し、共有メモリーまたはファイルに保管した情報を表示することを指定します。このトレース情報はバージョン 7 のフォーマットで表示されます。
stat	トレース機能の状況を表示することを指定します。この状況情報には、トレース・バージョン、アプリケーション・バージョン、項目数、バッファー・サイズ、使用中のバッファー量、状況コード、およびプログラム・タイム・スタンプが含まれます。
statlong	トレース機能の状況に z/OS バージョン・レベル情報を追加して表示することを指定します。この追加情報には、アプリケーション内の各モジュールのサービス・レベルが含まれ、長ストリングのテキストとして表示されます。
-fn *filename*	ミラーリングされたトレース情報を含むファイル名を指定します。ここには、トレース機能からのすべての出力が含まれます。
-help	有効なコマンド・パラメーターを記述と一緒に表示します。
-listsymbols	**-df** パラメーターで使用できる有効な関数およびコンポーネント ID を表示します。

表 36. Linux、UNIX、Windows、および z/OS オペレーティング・システム用 asntrc 呼び出しパラメーター定義 (続き)

パラメーター	定義
-b *buffer_size*	トレース・バッファーのサイズを指定します (バイト単位)。数値の後に、キロバイトなら K を、メガバイトなら M を指定できます。これらの文字には大文字小文字の区別はありません。
-fs *filesize*	ミラーリングされたトレース情報ファイルのサイズ制限をバイト単位で指定します。
-d *diag_mask*	トレース機能により記録されるトレース・レコードのタイプを指定します。トレース・レコードは、以下の診断マスク番号により分類されます。 **1** フロー・データ。関数の入口点と出口点が含まれます。 **2** 基本データ。トレース機能が検出したすべての主要なイベントが含まれます。 **3** 詳細データ。主要なイベントとその記述が含まれます。 **4** パフォーマンス・データ。 **重要:** 診断マスク番号の大きいものは、診断マスク番号の小さいものを包含していません。 これらの番号を 1 つまたは複数入力し、必要なトレース・レコードだけを含む診断マスクを作成することができます。例えば、**-d 4** を指定すると、パフォーマンス・データだけが記録されます。フローとパフォーマンスのデータだけを記録するには **-d 1,4** と指定し、すべてのトレース・レコードを記録するには **-d 1,2,3,4** (デフォルト) と指定します。番号はコンマで区切ります。 トレース機能がグローバル・トレース・レコードを記録しないようにするには、診断マスク番号 0 (ゼロ) を入力します。トレース機能に新しい診断マスク番号を指定する前に、診断レベルをリセットするには、**-d 0** を入力します。
-df *function_name|component_name diag_mask*	特定の関数またはコンポーネント ID をトレースすることを指定します。 関数またはコンポーネント ID 名の後に診断マスク番号 (1、2、3、4) を入力します。1 つ、または複数のこれらの番号を入力できます。番号はコンマで区切ります。

asntrc の例

次の例は、asntrc コマンドの使用方法を示しています。これらの例は、Linux、UNIX、Windows、z/OSの各オペレーティング・システムで実行できます。

例 1

キャプチャー・プログラムの実行をトレースするには、次のように入力します。

1. 最大のバッファー・サイズとファイル・サイズを持つトレース・ファイル名を指定して、トレース機能を開始します。

   ```
   asntrc on -db mydb -cap -schema myschema -b 256k -fn myfile.trc -fs 500m
   ```

2. キャプチャー・プログラムを始動し、適切な長さの時間実行させます。
3. トレース機能がオンになっている間に、共有メモリーからデータを直接表示します。

   トレース機能から処理とスレッドのサマリー情報を表示するには、次のように指定します。

   ```
   asntrc flw -db mydb -cap -schema myschema
   ```

   キャプチャー・ログ・リーダーからのみ、フロー、基本、詳細、およびパフォーマンスのデータ・レコードを表示するには、次のように指定します。

   ```
   asntrc fmt -db mydb -cap -schema myschema -d 0
     -df "Capture Log Read" 1,2,3,4
   ```

4. トレース機能を停止します。

   ```
   asntrc off -db mydb -cap -schema myschema
   ```

   トレース・ファイルには、キャプチャー・プログラムの始動時点から、トレース機能をオフにした時点までに生成された、すべてのキャプチャー・プログラム・トレース・データが含まれます。
5. トレース機能を停止した後、生成されたバイナリー・ファイルからのデータをフォーマットします。

   ```
   asntrc flw -fn myfile.trc
   ```

   および

   ```
   asntrc fmt -fn myfile.trc -d 0 -df "Capture Log Read" 1,2,3,4
   ```

例 2

レプリケーション・アラート・モニター・プログラムのトレース機能を開始するには、次のように指定します。

```
asntrc on -db mydb -mon -qualifier monq
```

例 3

アプライ・プログラムのパフォーマンス・データだけをトレースするには、次のように指定します。

```
asntrc on -db mydb -app -qualifier aq1 -b 256k -fn myfile.trc -d 4
```

例 4

キャプチャー・プログラムのすべてのフローおよびパフォーマンス・データをトレースするには、次のように指定します。

```
asntrc on dbserv1 -cap -schema myschema -b 256k
  -fn myfile.trc -d 1,4
```

例 5

すべてのグローバルなパフォーマンス・データおよび、キャプチャー・プログラムの、特定のキャプチャー・ログ・リーダーのフロー・データをトレースするには、次のように指定します。

```
asntrc on -db mydb -cap -schema myschema -b 256k -fn myfile.trc -d 4
  -df "Capture Log Read" 1
```

例 6

キャプチャー・プログラムをトレースし、トレース機能の特定の時点のイメージを表示して保管するには、次のようにします。

1. 最新のレコードを保留するために十分なバッファー・サイズを指定して、トレース・コマンドを始動します。

   ```
   asntrc on -db mydb -cap -schema myschema -b 4m
   ```

2. キャプチャー・プログラムを始動し、適切な長さの時間実行させます。
3. 共有メモリーに保管された、特定の時点の詳細なトレース情報を表示します。

   ```
   asntrc fmt -db mydb -cap -schema myschema
   ```

4. 特定の時点のトレース情報をファイルに保管します。

   ```
   asntrc dmp myfile.trc -db mydb -cap -schema myschema
   ```

5. トレース機能を停止します。

   ```
   asntrc off -db mydb -cap -schema myschema
   ```

共有セグメントを使用した asntrc の例

スタンドアロン・トレース機能 asntrc は、共有セグメントを使用して、トレース対象の Q キャプチャー・プログラム、 Q アプライ・プログラム、キャプチャー・プログラム、アラート・プログラム、またはレプリケーション・アラート・モニター・プログラムそれぞれと通信します。ファイルが指定されていない場合、トレース項目を保持するためにも共有セグメントが使用されます。それ以外の場合、正しい共有セグメントを突き合わせてトレースを制御するために、 asntrc コマンドおよびトレース対象の各プログラムの両方に、マッチング・オプションを指定しなければなりません。以下の例は、 Q キャプチャー・プログラム、Q アプライ・プログラム、キャプチャー・プログラム、アラート・プログラム、またはアラート・モニター・プログラムとの関連でトレース機能を使用する場合に、指定する必要のあるオプションを示しています。

Q キャプチャー・プログラムについては、 asntrc コマンドに **-db** パラメーターで指定したデータベースが、 asntrc コマンドに **capture_server** パラメーターで指定したデータベースと一致している必要があります。

```
asntrc -db ASN6 -schema EMI -qcap
asnqcap capture_server=ASN6 capture_schema=EMI
```

Q アプライ・プログラムについては、 asntrc コマンドに **-db** パラメーターで指定したデータベースが、 asnqapp コマンドに **apply_server** パラメーターで指定したデータベースと一致している必要があります。

```
asntrc -db TSN3 -schema ELB -qapp
asnqapp apply_server=TSN3 apply_schema=ELB
```

キャプチャー・プログラムについては、 asntrc コマンドに **-db** パラメーターで指定したデータベースが、 asncap コマンドに **capture_server** パラメーターで指定したデータベースと一致している必要があります。

```
asntrc -db DSN6 -schema JAY -cap
asncap capture_server=DSN6 capture_schema=JAY
```

アプライ・プログラムについては、 asntrc コマンドに **-db** パラメーターで指定したデータベースが、 asnapply コマンドに **control_server** パラメーターで指定したデータベースと一致している必要があります。

```
asntrc -db SVL_LAB_DSN6 -qualifier MYQUAL -app
asnapply control_server=SVL_LAB_DSN6 apply_qual=MYQUAL
```

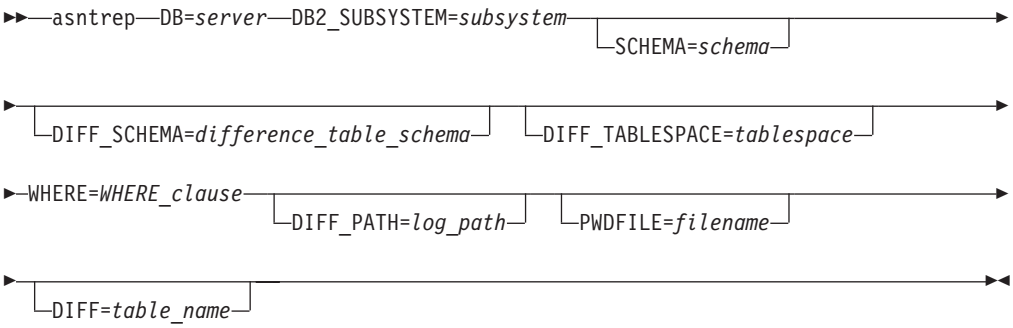
レプリケーション・アラート・モニター・プログラムについては、 asntrc コマンドに **-db** パラメーターで指定したデータベースが、 asnmon コマンドに **monitor_server** パラメーターで指定したデータベースと一致している必要があります。

```
asntrc -db DSN6 -qualifier MONQUAL -mon
asnmon monitor_server=DSN6 monitor_qual=MONQUAL
```

asntrep: ソース表とターゲット表の間の違いの修復

ソース表とターゲット表の間の違いを修復して 2 つの表を同期化するには、asntrep コマンドを使用します。asntrep コマンドは、Linux、 UNIX、 Windows、または z/OS 上のオペレーティング・システム上のシステム・プロンプトかシェル・スクリプトで実行してください。

構文



パラメーター

表 37 は、asntrep コマンドの呼び出しパラメーターを定義します。

表 37. Linux、UNIX、Windows、および z/OS オペレーティング・システム用 asntrep 呼び出しパラメーター定義

パラメーター	定義
DB=*server*	同期化するソース表とターゲット表に関する情報を保管するデータベースの DB2 別名を指定します。この値は、 Q レプリケーションか SQL レプリケーションのどちらを使用するかに応じて変わります。 **Q レプリケーション** この値は、 IBMQREP_SUBS 表を含む Q キャプチャー・サーバーの名前になります。 **SQL レプリケーション** この値は、 IBMSNAP_SUBS_MEMBR 表を含むアプライ・コントロール・サーバーの名前になります。 z/OS このパラメーターの値は、ロケーション名になります。
DB2_SUBSYSTEM=*subsystem*	z/OS asntrep ユーティリティーを実行するサブシステムの名前を指定します。
SCHEMA=*schema*	Q レプリケーションの場合は Q キャプチャー・コントロール表のスキーマを指定し、SQL レプリケーションの場合はアプライ・コントロール表のスキーマを指定します。
DIFF_SCHEMA= *difference_table_schema*	相違検出表を修飾するスキーマを指定します。デフォルトは ASN です。
DIFF_TABLESPACE=*tablespace*	ターゲット・データベースまたはサブシステム中の、相違検出のコピーを入れる表スペースを指定します。その後このコピーは、ターゲット表の修復に使用されます。このパラメーターを指定しないと、asntrep コマンドを実行したデータベースまたはサブシステム中のデフォルトの表スペースにこの表が作成されます。

表 37. Linux、UNIX、Windows、および zOS オペレーティング・システム用 asntrep 呼び出しパラメーター定義 (続き)

パラメーター	定義
WHERE=*WHERE_clause*	同期化対象のソース表とターゲット表に関する情報を保管するコントロール表の 1 行をユニークに識別する、 SQL WHERE 文節を指定します。 WHERE 文節は、二重引用符で囲まなければなりません。このパラメーターの値は、 Q レプリケーションか SQL レプリケーションのどちらを使用するかに応じて変わります。 **Q レプリケーション** WHERE 文節は IBMQREP_SUBS 表中の行を指定します。この表は SUBNAME 列を使用して、ソース表とターゲット表を含む Q サブスクリプションを識別します。 **SQL レプリケーション** WHERE 文節は IBMSNAP_SUBS_MEMBR 表中の行を指定します。この表は SET_NAME 列、APPLY_QUAL 列、TARGET_SCHEMA 列と TARGET_TABLE 列を使用して、ソース表とターゲット表を含むサブスクリプション・セットのメンバーを識別します。
DIFF_PATH=*log_path*	asntrep ユーティリティーのログの作成先となるロケーションを指定します。デフォルト値は、このコマンドを実行したディレクトリーです。この値は、絶対パス名でなければなりません。大文字小文字を区別する場合は二重引用符 (″″) を使用してください。
PWDFILE=*filename*	データベースとの接続に使用されるパスワード・ファイルの名前を指定します。パスワード・ファイルを指定しない場合、デフォルト値は asnpwd.aut (asnpwd コマンドによって作成されるパスワード・ファイルの名前) です。asntrep ユーティリティーは、DIFF_PATH パラメーターで指定されたディレクトリー内でパスワード・ファイルを探します。DIFF_PATH パラメーターを指定しない場合、このコマンドは、コマンドを実行したディレクトリー内でパスワード・ファイルを探します。
DIFF=*table_name*	asntdiff コマンドを使用してソース・データベース中に作成され、ソース表とターゲット表の間の違いを保管する表の名前を指定します。この表に保管される情報は、ソース表とターゲット表の同期化に使用されます。

asntrep の例

次の例は、asntrep コマンドの使用方法を示しています。

例 1

Q レプリケーションにおいて、Q キャプチャー・スキーマが asn の Q キャプチャー・サーバー source_db 上で、my_qsub という Q サブスクリプションに指定されているソース表とターゲット表 (両者の違いは表 q_diff_table に保管される) を同期化するには、以下のようにします。

```
asntrep db=source_db schema=asn where="subname = 'my_qsub'" diff=q_diff_table
```

例 2

SQL レプリケーションにおいて、アプライ・スキーマが asn のアプライ・コントロール・サーバー apply_db 上で、my_set というサブスクリプション・セットに指定されているソース表とターゲット表 trg_table (両者の違いは表 sql_diff_table に保管される) を同期化するには、以下のようにします。

```
asntrep DB=apply_db SCHEMA=asn WHERE="set_name = 'my_set'
 and target_table = 'trg_table'" diff=sql_diff_table
```

第 23 章 SQL レプリケーション用のシステム・コマンド (System i)

System i サーバーの場合は、System i オペレーティング・システム固有のレプリケーション・コマンドもあります。これらのコマンドは、オペレーティング・システムのコマンド・プロンプトか、またはコマンド行プログラムを使用して入力することができます。

以下のトピックでは、これらのコマンドについて説明しています。

ADDDPRREG: DPR 登録の追加 (System i)

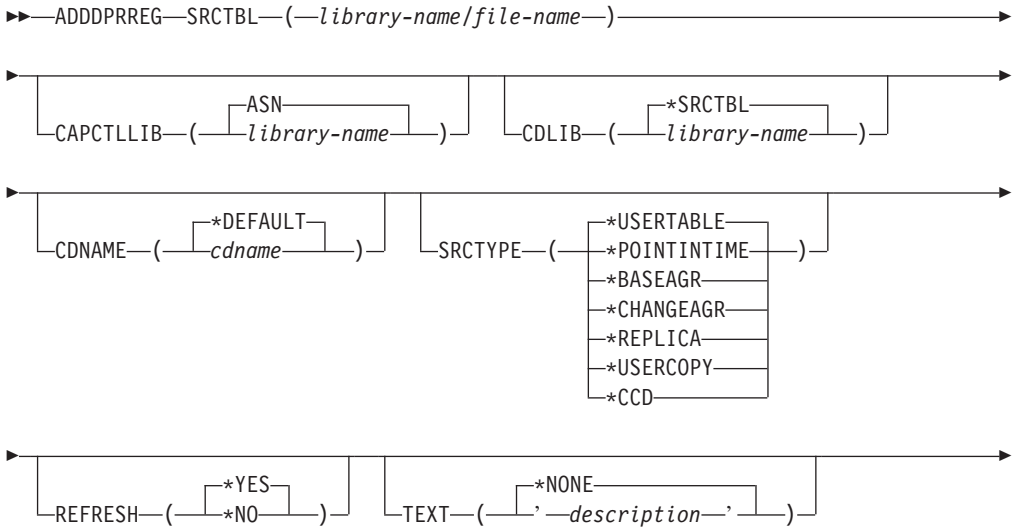
表を DB2 DataPropagator for iSeriesのソース表として登録するには、 DPR 登録の追加 (ADDDPRREG) コマンドを使用します。

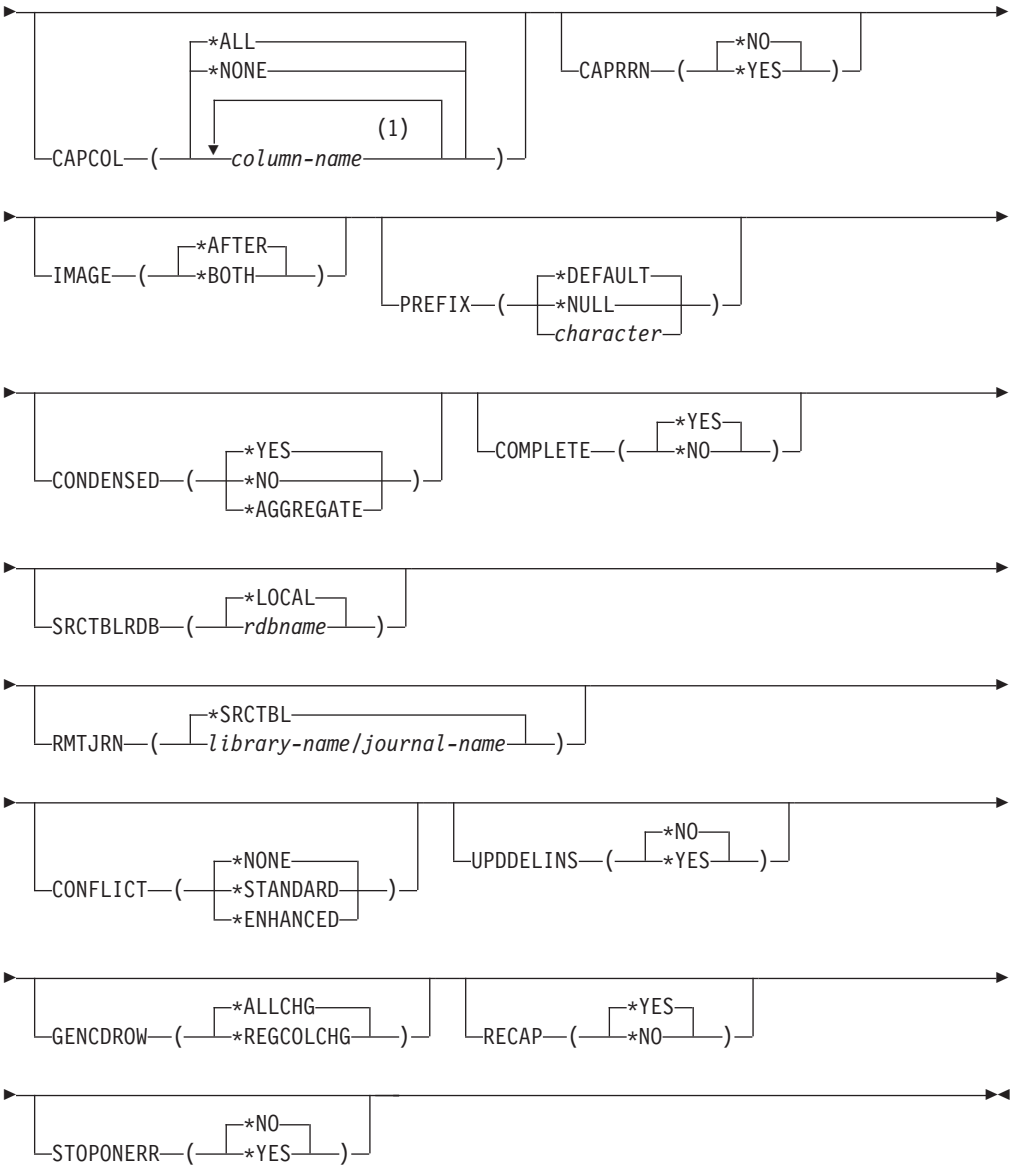
制約事項: 表を登録できるのは、 ASN (キャプチャー・スキーマ) ライブラリーが ASN ライブラリーが存在するのと同じ補助プール (基本または独立 ASP のいずれか) にある場合だけです。

コマンド行にコマンド名を入力してから、F4 キーを押してコマンド構文を表示することができます。

このコマンド、およびこのコマンドのすべてのパラメーターの詳しい記述を表示するには、画面の一番上のコマンドにカーソルを移動し、F1 キーを押します。特定のパラメーターの記述を表示するには、そのパラメーター上にカーソルを移動し、F1 キーを押します。

構文





注:

1 列名は 300 個まで指定できます。

表 38 は、呼び出しパラメーターをリストしています。

表 38. ADDDPRREG コマンド・パラメーター定義 (System i 版)

パラメーター	定義およびプロンプト
SRCTBL	ソース表として登録する表を指定します。キャプチャー・プログラムは、System i ライブラリー内の任意の物理ファイルまたは、外部で定義され、1 つのフォーマットを持つ集合をサポートします。このパラメーターは必須です。 *library-name/file-name* 登録する表の修飾名を表します。

表 38. ADDDPRREG コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
CAPCTLLIB	キャプチャー・スキーマ (キャプチャー・コントロール表が存在するライブラリーの名前) を指定します。 **ASN (デフォルト)** キャプチャー・コントロール表は ASN ライブラリー内にあります。 *library-name* キャプチャー・コントロール表を含むライブラリーの名前。CRTDPRTBL コマンドに **CAPCTLLIB** パラメーターを指定して、このライブラリーを作成することができます。
CDLIB	この登録されたソースの変更データ (CD) 表を作成するライブラリーを指定します。 ***SRCTBL (デフォルト)** ソース表が存在するライブラリー内に CD 表を作成します。 *library-name* 指定したライブラリー名に CD 表を作成します。
CDNAME	変更データ (CD) 表の名前を指定します。 ***DEFAULT (デフォルト)** デフォルト名 (現行タイム・スタンプを基にしたもの) を使用して CD 表を作成します。例えば、現行のタイム・スタンプが 2002 年 1 月 23 日 9 時 58 分 26 秒の場合、デフォルト名は ASN020123095826CD となります。 *cdname* ここに指定した名前で CD 表を作成します。

表 38. ADDDPRREG コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
SRCTYPE	登録するソース表のタイプを指定します。ご使用のレプリケーション構成に基づいてソース・タイプを選択します。 • 基本データ分散またはデータ統合構成の場合は、デフォルトの USERTABLE を使用します。 • Update-anywhere 構成の場合は REPLICA を使用します。 • 複数の階層からなる構成を持ち、ターゲット表を、レプリケーション構成内の下位の層のためのソースにする場合は、 POINTINTIME、BASEAGR、CHANGEAGR、USERCOPY、または CCD を使用します。 既存のターゲット表をソースとして登録する場合、指定されたソース・タイプに示された IBMSNAP 表の列がターゲット表に含まれていないと、登録は失敗します。 ***USERTABLE (デフォルト)** ユーザー・データベース表。登録された表の最もよくあるタイプです。表には、DB2 DataPropagator for System i の列 ID である、IBMSNAP または IBMQSQ で始まる列を含めることはできません。 ***POINTINTIME** ポイント・イン・タイム・コピー表。これは、ソース表の内容の一部またはすべてと一致する内容を含み、さらに、DB2 DataPropagator for System i システム列を保持します。このシステム列は、特定の行がソース・システムで最後に挿入または更新されたポイント・イン・タイムを示します。表には IBMSNAP_LOGMARKER タイム・スタンプ列を含める必要があり、オプションとして IBMQSQ_RRN と呼ばれる INTEGER 列を含めることができます。 ***BASEAGR** 基本集約コピー。ユーザー表またはポイント・イン・タイム表からインターバルごとに集約されたデータが入ります。基礎集約表には、IBMSNAP_HLOGMARKER および IBMSNAP_LLOGMARKER タイム・スタンプ列を含める必要があります。 ***CHANGEAGR** 変更を集約したコピー表。ソース表に記録された変更に基づくデータの集約が入ります。この表には、IBMSNAP_HLOGMARKER および IBMSNAP_LLOGMARKER タイム・スタンプ列を含める必要があります。 ***REPLICA** レプリカ・サブスクリプションのターゲット表。ターゲット表からの変更が複製されて、元のソース表に戻せるようにするには、このタイプの表を登録します。この表には、DB2 DataPropagator for System i のシステム列または、DB2 DataPropagator for System i の列 ID である、 IBMSNAP または IBMQSQ で始まる列を含めることはできません。この表は、元のソース表からのすべての列を保持します。 ***USERCOPY** ソース表の内容のすべてまたは一部と一致する内容を持つ、ターゲット表です。ユーザー・コピー表はユーザー・データ列のみ保持します。

表 38. ADDDPRREG コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
SRCTYPE (続き)	***CCD** 整合変更データ (CCD) 表。ソース表からのトランザクション整合性のあるデータが入ります。表には、以下のように定義された列を含める必要があります。 • IBMSNAP_INTENTSEQ CHAR(10) FOR BIT DATA NOT NULL • IBMSNAP_OPERATION CHAR(1) NOT NULL • IBMSNAP_COMMITSEQ CHAR(10) FOR BIT DATA NOT NULL • IBMSNAP_LOGMARKER TIMESTAMP NOT NULL
REFRESH	フル・リフレッシュ機能を使用可能にするかどうかを指定します。この値を使用して、ソース・データベースからのフル・リフレッシュを実行するアプライ・プログラムの機能をオフにすることができます。 ***YES (デフォルト)** フル・リフレッシュを使用可能にします。 ***NO** フル・リフレッシュは使用不可です。 ターゲット表が基礎集約表または変更集約表の場合、このパラメーターは *No にする必要があります。
TEXT	この登録に関する記述テキストを指定します。 ***NONE (デフォルト)** 項目には記述がありません。 *description* この登録を記述するテキスト。最大 50 文字を入力でき、テキストは単一引用符で囲む必要があります。
CAPCOL	この登録済み表について、どの列の変更をキャプチャーするかを指定します。 ***ALL (デフォルト)** すべての列の変更をキャプチャーします。 ***NONE** この表の変更をキャプチャーしません。この値は、この表をフル・リフレッシュ用としてのみ登録する場合に指定します。登録されたこの表については変更データ (CD) 表は作成されず、キャプチャー・プログラムは表の変更をキャプチャーしません。 *column-name* その列の変更をキャプチャーする列名。列名は 300 個まで指定できます。列名はスペースで区切ります。

表 38. ADDDPRREG コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
CAPRRN	変更されたレコードの相対レコード番号 (RRN) をキャプチャーするかどうかを指定します。 ***NO(デフォルト)** 相対レコード番号をキャプチャーしません。 ***YES** 相対レコード番号をキャプチャーします。 変更データ (CD) 表に IBMQSQ_RRN と呼ばれる追加の列が作成されます。 ソース表にユニーク・キーがない場合のみ、このパラメーターを *YES にします。
IMAGE	変更データ (CD) 表に、ソース表の変更前イメージと変更後イメージの両方を含めるかどうかを指定します。この値は、キャプチャー列パラメーター (**CAPCOL**) に指定したすべての列にグローバルに適用されます。 この **IMAGE** パラメーターは、**CAPCOL** パラメーターが *NONE の場合は無効です。 このパラメーターに *AFTER を指定した場合でも、ソース表は *BOTH イメージでジャーナルに記録する必要があります。 ***AFTER (デフォルト)** キャプチャー・プログラムはソース表の変更後イメージのみ CD 表に記録します。 ***BOTH** キャプチャー・プログラムはソース表の変更前イメージと変更後イメージの両方を CD 表に記録します。
PREFIX	変更データ (CD) 表の変更前イメージの列を識別する接頭部文字を指定します。ソース表の登録された列名に、この接頭部文字で始まる列名がないことを確認してください。 ***DEFAULT (デフォルト)** デフォルトの接頭部 (@) が使用されます。 ***NULL** 変更前イメージはキャプチャーされません。この値は、**IMAGE** パラメーターを *BOTH にした場合は無効です。 *character* オブジェクト名に使用できる任意の 1 つの英字。

表 38. ADDDPRREG コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
CONDENSED	ソース表が圧縮されているかどうかを指定します。コンデンス表には、現行データが含まれ、 1 つの主キーの値に対して複数の行が含まれることはありません。 ***YES (デフォルト)** ソース表は圧縮されています。 ***NO** ソース表は圧縮されていません。 ***AGGREGATE** ソース表のタイプは *BASEAGR (基本集約) または *CHANGEAGR (変更集約) のいずれかです。この値を使用する場合、**COMPLETE** パラメーターを *No にする必要があります。
COMPLETE	ソース表が完全であるかどうかを指定します。これは、表に、対象となる主キー値ごとに 1 行が含まれていることを意味します。 ***YES (デフォルト)** ソース表は完全です。 ***NO** ソース表は完全ではありません。
SRCTBLRDB	リモート・ジャーナリングを使用するかどうかを指定します。リモート・ジャーナリングでは、ソース表とリモート・ジャーナルは別のシステムにあります。このパラメーターを使用して、ソース表のロケーションを指定します。 ***LOCAL (デフォルト)** ソース表はローカルにあります (ADDDPRREG コマンドを実行するマシン上に)。 *rdbname* ソース表が存在するリレーショナル・データベースの名前。 RDB ディレクトリー項目の作業 (WRKRDBDIRE) コマンドを使用して、このリレーショナル・データベース名を見つけることができます。
RMTJRN	このジャーナルの名前とソース・システム上のジャーナルの名前が異なる場合に、リモート・ジャーナルの名前を指定します。このコマンドは、リモート・ジャーナルが存在するシステムから発行する必要があります。 ***SRCTBL (デフォルト)** リモート・ジャーナル名は、ソース表のジャーナル名と同じです。 *library-name/journal-name* このシステムにあり、リモート・ソース表のジャーナリングに使用される、修飾されたライブラリー名とジャーナル名。 リモート・ジャーナル名は、**SRCTBLRDB** パラメーターでリモート・ソース表のロケーションを指定した場合のみ指定できます。

表 38. ADDDPRREG コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
CONFLICT	レプリカ・サブスクリプション内の競合を検出するときにアプライ・プログラムが使用する、競合レベルを指定します。 ***NONE (デフォルト)** 競合検出を行いません。 ***STANDARD** 適度な競合検出。アプライ・プログラムは、レプリカ変更データ (CD) 表内のすでにキャプチャーされた行について、競合を探します。 ***ENHANCED** 拡張競合検出を行います。このオプションは、すべてのレプリカとソース表の間に最良のデータ保全性を提供します。
UPDDELINS	キャプチャー・プログラムが、更新されたソース・データを、変更データ (CD) 表にどのように保管するかを決めます。 ***NO (デフォルト)** キャプチャー・プログラムは、それぞれのソース変更を CD 表内の 1 つの行に保管します。 ***YES** キャプチャー・プログラムは、それぞれのソース変更を CD 表内に 2 行を使用して保管し、その 1 行は削除用、もう 1 行は挿入用です。アプライ・プログラムは最初に削除用の行を処理し、2 番目に挿入用の行を処理します。
GENCDROW	キャプチャー・プログラムがソース表のすべての行から変更をキャプチャーするかどうかを指定します。 ***ALLCHG (デフォルト)** キャプチャー・プログラムはソース表のすべての行から変更をキャプチャーし (登録されていない列の変更も含む)、これらの変更を変更データ (CD) 表に追加します。 ***REGCOLCHG** キャプチャー・プログラムは、登録された列に変更があった場合のみ変更をキャプチャーします。その後、キャプチャー・プログラムはこれらの行を CD 表に追加します。 *REGCOLCHG は、**CAPCOL** パラメーターが *ALL または *NONE の場合には指定できません。
RECAP	アプライ・プログラムが行った変更を、キャプチャー・プログラムが再度キャプチャーするかどうかを指定します。 ***YES(デフォルト)** アプライ・プログラムがソース表を変更した場合に、その変更をキャプチャーし、変更データ (CD) 表に入れます。 ***NO** アプライ・プログラムがソース表を変更した場合に、その変更はキャプチャーせず、したがって、変更データ (CD) 表には入れません。このオプションは、REPLICA タイプの表を登録する場合に使用してください。

表 38. ADDDPRREG コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
STOPONERR	キャプチャー・プログラムがエラーを検出した場合に、キャプチャー・プログラムを停止するかどうかを指定します。[1] ***NO(デフォルト)** キャプチャー・プログラムはエラーを検出しても停止しません。キャプチャー・プログラムはメッセージを出し、エラーの原因となった登録を非活動化してから、処理を継続します。 ***YES** キャプチャー・プログラムはエラーを検出した場合に、メッセージを出してから停止します。

注:

1. このパラメーターを Yes (Y) に設定すると、他のジャーナル・ジョブが実行を継続している間、キャプチャー・ジャーナル・ジョブは停止します。このパラメーターを No (N) に設定すると、キャプチャー・プログラムはエラーを含む登録ファイルを停止します。

   さらに、このパラメーターは登録表の行に列を設定します。 STATE 列が 'S' に設定され、STATE_INFO 列は 200A 200A*xxxx* に設定されます (*xxxx* は理由コード)。登録の設定をアクション ('A') 状態に戻すには、以下のステップを実行します。

   - ASN200A メッセージを訂正します。修正されたアクションについては、該当する System i 資料を参照してください。
   - レプリケーション・センターまたは System i コマンド STRSQL を使用して、IBMSNAP_REGISTER 表の行に列を設定します。STATE 列を 'A' に設定し、STATE_INFO 列を NULL に設定します。
   - キャプチャーを実行中の場合は、 INZDPRCAP コマンドを出して、そのジャーナルのデータ・レプリケーションを再初期化します。

ADDDPRREG の例

以下の例は、ADDDPRREG コマンドの使用法を示しています。

例 1:

デフォルトのキャプチャー・スキーマの下に、HR ライブラリーから EMPLOYEE という名前のソース表を登録します。

```
ADDDPRREG SRCTBL(HR/EMPLOYEE)
```

例 2

BSN キャプチャー・スキーマの下に、HR ライブラリーから EMPLOYEE という名前のソース表を登録し、 HRCDLIB ライブラリーの下に CDEMPLOYEE という名前の CD 表を作成します。

```
ADDDPRREG SRCTBL(HR/EMPLOYEE) CAPCTLLIB(BSN) CDLIB(HRCDLIB) CDNAME(CDEMPLOYEE)
```

例 3

BSN キャプチャー・スキーマの下に、DEPT ライブラリーから SALES という名前のポイント・イン・タイムのソース・タイプ付きソース表を登録します。

```
ADDDPRREG SRCTBL(DEPT/SALES) CAPCTLLIB(BSN) SRCTYPE(*POINTINTIME)
```

例 4

DEPT ライブラリーから SALES という名前のソース表を登録し、CD 表には、ソース表の変更について変更前イメージと変更後イメージの両方を含めます。

```
ADDDPRREG SRCTBL(DEPT/SALES) IMAGE(*BOTH)
```

例 5

DEPT ライブラリーから SALES という名前のソース表 (RMTRDB1 という名前のリレーショナル・データベースで、リモート・ジャーナルを使用する) を登録します。

```
ADDDPRREG SRCTBL(DEPT/SALES) SRCTBLRDB(RMTRDB1) RMTJRN(RMTJRNLIB/RMTJRN)
```

例 6

HR ライブラリーから EMPLOYEE ソース表を登録し、EMPNO、NAME、 DEPT、および NETPAY の列についてのみ、変更をキャプチャーします。

```
ADDDPRREG SRCTBL(HR/EMPLOYEE) CAPCOL(EMPNO NAME DEPT NETPAY)
```

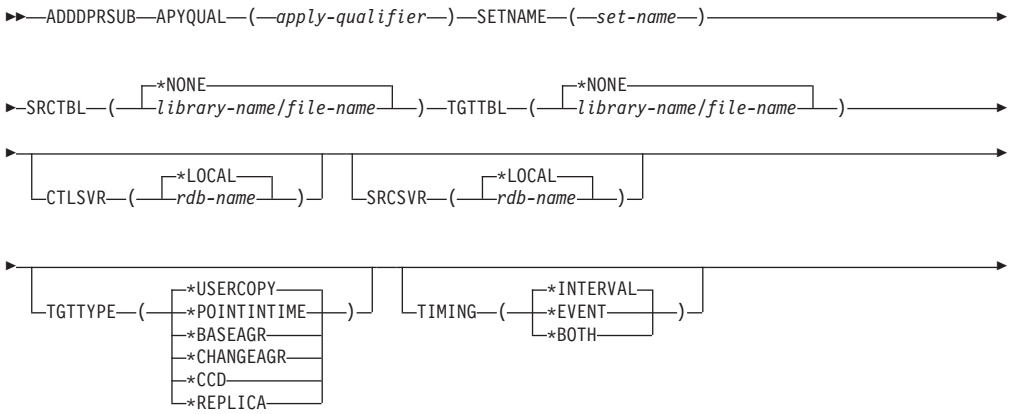
ADDDPRSUB: DPR サブスクリプション・セットの追加 (System i)

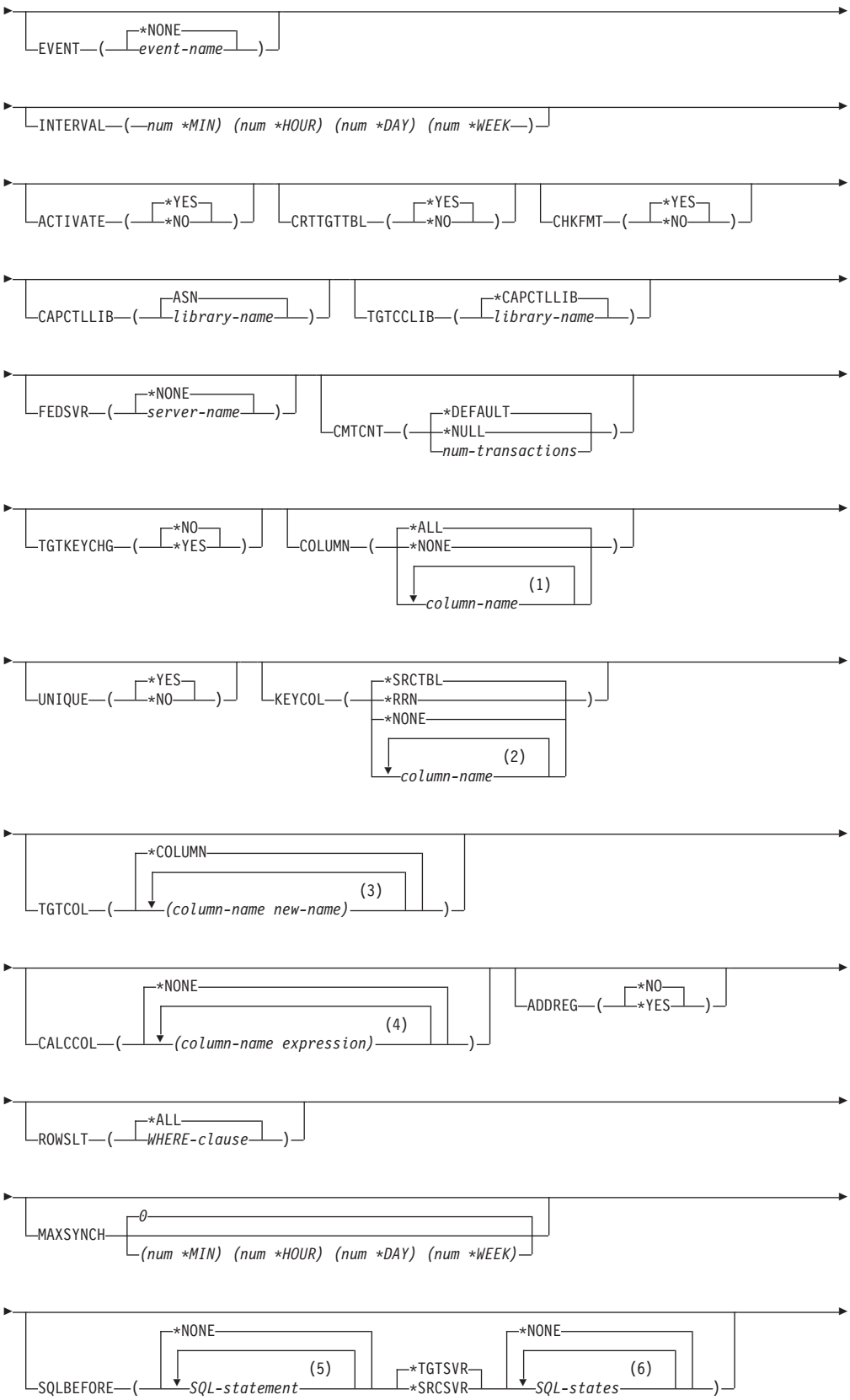
1 つのメンバーを持つ、またはメンバーを持たないサブスクリプション・セットを作成するには、DPR サブスクリプション・セットの追加 (ADDDPRSUB) コマンドを使用します。

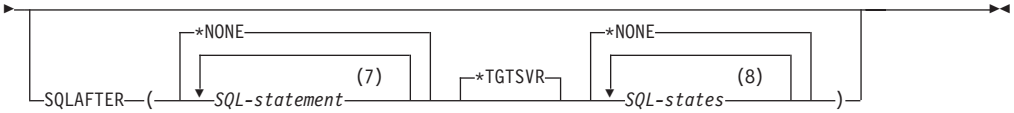
コマンド行にコマンド名を入力してから、F4 キーを押してコマンド構文を表示することができます。

このコマンド、およびこのコマンドのすべてのパラメーターの詳しい記述を表示するには、画面の一番上のコマンドにカーソルを移動し、F1 キーを押します。特定のパラメーターの記述を表示するには、そのパラメーター上にカーソルを移動し、F1 キーを押します。

構文




EVENT
*NONE
event-name
INTERVAL
num *MIN) (num *HOUR) (num *DAY) (num *WEEK
ACTIVATE
*YES
*NO
CRTTGTTBL
*YES
*NO
CHKFMT
*YES
*NO
CAPCTLLIB
ASN
library-name
TGTCCLIB
*CAPCTLLIB
library-name
FEDSVR
*NONE
server-name
CMTCNT
*DEFAULT
*NULL
num-transactions
TGTKEYCHG
*NO
*YES
COLUMN
*ALL
*NONE
(1)
column-name
UNIQUE
*YES
*NO
KEYCOL
*SRCTBL
*RRN
*NONE
(2)
column-name
TGTCOL
*COLUMN
(3)
(column-name new-name)
CALCCOL
*NONE
(4)
(column-name expression)
ADDREG
*NO
*YES
ROWSLT
*ALL
WHERE-clause
MAXSYNCH
0
(num *MIN) (num *HOUR) (num *DAY) (num *WEEK)
SQLBEFORE
*NONE
(5)
SQL-statement
*TGTSVR
*SRCSVR
*NONE
(6)
SQL-states



注:

1 列名は 300 個まで指定できます。

2 列名は 120 個まで指定できます。

3 列名は 300 個まで指定できます。

4 100 個までの列名および式を指定できます。

5 SQLステートメントは 3 個まで指定できます。

6 SQLSTATES は 10 個まで指定できます。

7 SQLステートメントは 3 個まで指定できます。

8 SQLSTATES は 10 個まで指定できます。

表 39 は、呼び出しパラメーターをリストしています。

表 39. ADDDPRSUB コマンド・パラメーター定義 (System i 版)

パラメーター	定義およびプロンプト
APYQUAL	どのアプライ・プログラムがこのサブスクリプション・セットを処理するかを示すアプライ修飾子を指定します。 アプライ修飾子の下のサブスクリプション・セットは別のジョブで実行されます。このパラメーターは必須です。 *apply-qualifier* アプライ修飾子の名前。
SETNAME	サブスクリプション・セットの名前を指定します。このパラメーターは必須です。 *set-name* サブスクリプション・セットの名前。入力するサブスクリプション・セット名は、指定されたアプライ修飾子についてユニークでなければなりません。ユニークでない場合、ADDDPRSUB コマンドはエラーになります。アプライ・プログラムはターゲット表のセットをグループとして扱うので、何らかの理由で 1 つのターゲット表が失敗すると、サブスクリプション・セット全体が失敗します。
SRCTBL	情報をサブスクリプション・セットにコピーするために使用する、ソース表の名前を指定します。この表をサブスクリプション・セットのメンバーにするには、この表をキャプチャー・コントロール・サーバーにあらかじめ登録しておく必要があります。このパラメーターは必須です。 ***NONE (デフォルト)** このサブスクリプション・セットはソース・メンバーを持ちません。メンバーのないサブスクリプション・セットを作成する場合に使用します。 *library-name/file-name* ソース表の修飾名。1 つのメンバーを持つサブスクリプション・セットを作成する場合に使用します。

表 39. ADDDPRSUB コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
TGTTBL	ターゲット表の名前を指定します。 **CRTTGTTBL** パラメーターを *YES に設定し、ターゲット表が存在しない場合、ターゲット表は自動的に作成されます。このパラメーターは必須です。 ***NONE (デフォルト)** このサブスクリプション・セットはターゲット・メンバーを持ちません。メンバーのないサブスクリプション・セットを作成する場合に使用します。 *library-name/file-name* ターゲット表の修飾名。1 つのメンバーを持つサブスクリプション・セットを作成する場合に使用します。
CTLSVR	アプライ・コントロール表を含むシステムのリレーショナル・データベース名を指定します。 ***LOCAL (デフォルト)** アプライ・コントロール表はローカルに存在します (ADDDPRSUB コマンドを実行するマシン上)。 *rdb-name* アプライ・コントロール表が置かれているリレーショナル・データベースの名前。 RDB ディレクトリー項目の作業 (WRKRDBDIRE) コマンドを使用して、この名前を検出することができます。
SRCSVR	キャプチャー・コントロール表を含むシステムのリレーショナル・データベース名を指定します。 ***LOCAL (デフォルト)** ソース表はローカル・マシンに登録されています (ADDDPRSUB コマンドを実行するマシン)。 *rdb-name* キャプチャー・コントロール表が存在するリレーショナル・データベースの名前。 RDB ディレクトリー項目の作業 (WRKRDBDIRE) コマンドを使用して、この名前を検出することができます。

表 39. ADDDPRSUB コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
TGTTYPE	ターゲット表のタイプを指定します。これらのいずれかのタイプとしてターゲット表を作成した後、 DPR 登録の追加 (ADDDPRREG) コマンドの **SRCTBL** パラメーターにこのパラメーター値を使用し、このターゲット表を multi-tier レプリケーションのソース表として登録することができます。 ***USERCOPY (デフォルト)** ターゲット表はユーザー・コピーであり、これはソース表の内容のすべてまたは一部と一致する内容を持つ、ターゲット表です。ユーザー・コピーはポイント・イン・タイムのコピーのように扱われますが、ポイント・イン・タイムのターゲット表に存在する、DB2 DataPropagator for System i システム列は 1 つも含まれていません。 この値は、**KEYCOL** パラメーターに *RRN の値が指定されている場合は無効です。 **SRCTBL** パラメーターで指定した表は、ユーザー・データベース、ポイント・イン・タイム・コピー、または整合変更データ (CCD) のいずれかでなければなりません。 **重要:** ターゲット表がすでに存在する場合、DB2 DataPropagator for System i は、これに対する変更を自動的にジャーナルに記録しません。ジャーナリングは、DB2 DataPropagator for System i の外側で開始する必要があります。 ***POINTINTIME** ターゲット表はポイント・イン・タイム・コピーです。ポイント・イン・タイム・コピーは、ソース表の内容の一部またはすべてと一致する内容を持つターゲット表であり、DB2 DataPropagator for System i システム列 (IBMSNAP_LOGMARKER) を保持します。この列は、特定の行がキャプチャー・コントロール・サーバーでいつ挿入または変更されたかを示します。 ***BASEAGR** ターゲット表は基本集約コピーであり、これはソース表から集約された (算出された) データを含む、ターゲット表です。基本集約ターゲットのソース表は、ユーザー表またはポイント・イン・タイム表のどちらかでなければなりません。このターゲット表には、システム・タイム・スタンプ列の IBMSNAP_HLOGMARKER および IBMSNAP_LLOGMARKER が含まれます。 ***CHANGEAGR** 表は変更集約コピーであり、これは、変更データ (CD) 表の内容を基に集約された (算出された) データを含むターゲット表です。このターゲット表は、システム・タイム・スタンプ列 IBMSNAP_HLOGMARKER および IBMSNAP_LLOGMARKER を使用して作成されます。

表 39. ADDDPRSUB コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
TGTTYPE (続き)	***CCD** 表は整合変更データ (CCD) 表であり、これは、変更データ (CD) 表と作業単位 (UOW) 表内のデータを結合したものから作成されたターゲット表です。 CCD 表は、アプライ・プログラムにトランザクション整合性のあるデータを提供し、次の列を含む必要があります。 • IBMSNAP_INTENTSEQ • IBMSNAP_OPERATION • IBMSNAP_COMMITSEQ • IBMSNAP_LOGMARKER ***REPLICA** ターゲット表はレプリカ表であり、これは Update-anywhere レプリケーションにのみ使用されます。レプリカ・ターゲット表はマスター・ソース表から変更を受信し、またレプリカ・ターゲット表への変更は、マスター・ソース表に戻して伝搬されます。レプリカ表はソース表として自動的に登録されます。
TIMING	アプライ・プログラムがサブスクリプション・セットの処理に使用するタイミング (スケジューリング) のタイプを指定します。 ***INTERVAL (デフォルト)** アプライ・プログラムは、サブスクリプション・セットを特定の時間間隔 (例えば、1 日に一度) で処理します。 ***EVENT** アプライ・プログラムは、特定のイベントが起こった時にサブスクリプション・セットを処理します。 ***BOTH** アプライ・プログラムは、特定の時間インターバル、またはイベントが起こった時のどちらでも、最初に起こった時にサブスクリプション・セットを処理します。
EVENT	イベントを指定します。入力するイベントは、IBMSNAP_SUBS_EVENT 表のイベント名と一致する必要があります。 ***NONE (デフォルト)** イベントは使用しません。 *event-name* IBMSNAP_SUBS_EVENT 表に記述されているイベントを表す、ユニークな文字ストリング。

表 39. ADDDPRSUB コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
INTERVAL	ターゲット・コピーのリフレッシュを行う時間間隔を指定します。これは開始時刻から開始時刻までの時間間隔であり、週、日、時間、および分で指定します。これは 2 つの部分からなる値です。最初の部分は数値、2 番目の部分は時間の単位です。 ***MIN** 分 ***HOUR** 時間 ***DAY** 日 ***WEEK** 週 時間の単位と一緒に、数値の組み合わせを指定することができます。例えば、((2 *WEEK) (3 *DAY) (35 *MIN)) は、2 週、3 日、および 35 分の時間間隔を指定します。同じ時間単位で複数を指定すると、最後の指定が使用されます。
ACTIVATE	サブスクリプション・セットがアクティブかどうかを指定します。アプライ・プログラムは、このパラメーターが *YES でない場合、このサブスクリプション・セットを処理しません。 ***YES (デフォルト)** サブスクリプション・セットはアクティブです。 ***NO** サブスクリプション・セットはアクティブではありません。

表 39. ADDDPRSUB コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
CRTTGTTBL	ターゲット表 (またはビュー) を作成するかどうかを指定します。 ***YES (デフォルト)** ターゲット表 (またはビュー) が存在しなければ、作成します。存在する場合は、既存の表またはビューがターゲットになり、この既存の表またはビューのフォーマットの**CHKFMT** パラメーターが *YES に設定されているかどうかをチェックします。**UNIQUE** および **KEYCOL** パラメーターに指定された値を使用して、 (そのような索引がまだ存在しなければ) ターゲット表に追加の索引が作成されます。既存のターゲット表に、追加索引の条件に違反するような行が含まれていると、コマンドは失敗します。 ***NO** ターゲット表またはビューを作成しません。アプライ・プログラムを始動する前に、正しい属性を使用して表またはビューを作成する必要があります。 表またはビューが存在する場合に **CHKFMT** を *YES にすると、ADDDPRSUB コマンドは、既存の表のフォーマットが、設定されたサブスクリプション・セット定義と一致することを確認します。**CHKFMT** を *NO にする場合は、ユーザーは既存の表のフォーマットがサブスクリプション・セット定義と一致することを確認しておく必要があります。 **重要:** 表またはビューがすでに存在する場合、DB2 DataPropagator for System i は、既存のオブジェクトへの変更を自動的にジャーナルに記録しません。ジャーナリングは、DB2 DataPropagator for System i の外側で開始する必要があります。
CHKFMT	DB2 DataPropagator for System i が、サブスクリプション・セットとターゲット表をチェックし、列が一致することを確認するかどうかを指定します。このパラメーターは、**CRTTGTTBL** パラメーターが *YES の場合は無視され、また **CRTTGTTBL** パラメーターが *NO でターゲット表が存在しない場合も無視されます。 ***YES (デフォルト)** DB2 DataPropagator for System i は、このサブスクリプション・セットに定義された列がターゲット表内の列と一致するか検証します。両者が一致しない場合、このコマンドは失敗します。 ***NO** DB2 DataPropagator for System i は、サブスクリプション・セットと既存のターゲット表間の相違を無視します。ユーザーはターゲット表がサブスクリプション・セットと互換性があることを確認する必要があります。

表 39. ADDDPRSUB コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
CAPCTLLIB	キャプチャー・スキーマ (キャプチャー・コントロール表が存在するライブラリーの名前) を指定します。これらのキャプチャー・コントロール表は、このサブスクリプション・セットのソースを処理します。 **ASN (デフォルト)** キャプチャー・コントロール表は ASN ライブラリー内にあります。 *library-name* キャプチャー・コントロール表を含むライブラリーの名前。これは、ソース表が登録されたライブラリーです。
TGTCCLIB	ターゲット・コントロール・ライブラリーを指定します。 ***CAPCTLLIB (デフォルト)** ターゲット・コントロール・ライブラリーは、キャプチャー・コントロール表が存在するライブラリーと同じです。 *library-name* ターゲット・コントロール表を含むライブラリーの名前。 ターゲット表を別のサブスクリプション・セット (例えば外部 CCD 表など) のソースとして使用する場合、このパラメーター値は、この表がソースとして使用される時のキャプチャー・スキーマです。
FEDSVR	このサブスクリプション・セットのソースがフェデレーテッド・データベース・システムであるかどうかを指定します。 ***NONE (デフォルト)** ソース・サーバーはフェデレーテッド・データベース・システムではありません。 *server-name* このサブスクリプション・セットのフェデレーテッド・データベース・システムの名前 (DB2 以外のリレーショナル・ソースの場合)。
CMTCNT	コミットメント・カウントを指定します。これは、アプライ・プログラムがトランザクションをいくつ処理したらコミットするかを示す数です。 ***DEFAULT (デフォルト)** 使用する値をコマンドが決めます。**TGTTYPE** が *REPLICA の場合、**CMTCNT** はゼロ (0) です。**TGTTYPE** が *REPLICA 以外の場合、**CMTCNT** は NULL です。 ***NULL** サブスクリプション・セットは読み取り専用です。アプライ・プログラムは、サブスクリプション・セット・メンバーの応答セットを一度に 1 メンバーずつフェッチし、すべてのデータの処理を終了した後、サブスクリプション・セット全体について 1 つのコミットを出します。 *num-transactions* いくつのトランザクション処理を処理したらアプライ・プログラムが変更をコミットするかを示す数を指定します。このパラメーターは、**TGTTYPE** パラメーターが *REPLICA の場合のみ有効です。

表 39. ADDDPRSUB コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
TGTKEYCHG	ターゲット表のターゲット・キー列の一部であるソース列に変更があった場合、アプライ・プログラムがその更新をどのように扱うかを指定します。このパラメーターは、ADDDPRREG コマンドの **USEDELINS** パラメーターと組み合わせて働きます。 • **USEDELINS** が YES で **TGTKEYCHG** が YES の場合、更新はできません。 • **USEDELINS** が YES で **TGTKEYCHG** が NO の場合、更新は削除と挿入の対になります。 • **USEDELINS** が NO で **TGTKEYCHG** が YES の場合、アプライ・プログラムは特別な論理を使用してこの条件を扱います。 • **USEDELINS** が NO で **TGTKEYCHG** が NO の場合、アプライ・プログラムは変更を通常の更新として処理します。 ***NO (デフォルト)** ソース表に対する更新は、キャプチャー・プログラムによりステージ化され、アプライ・プログラムによりターゲット表に処理が行われます。 ***YES** アプライ・プログラムは、ターゲット・キー列の変更前イメージに基づいてターゲット表を更新します。つまり、アプライ・プログラムは述部を新しい値ではなく、古い値に変更します。
COLUMN	ターゲット表に含める列を指定します。 列名は修飾できません。列名は、ソース表の登録時に **CAPCOL** パラメーターに指定した列名のリストから選択してください。 この表の登録時に **IMAGE** パラメーターを *BOTH に設定した場合は、変更前イメージ列名を指定することができます。変更前イメージ列名は、接頭部を持つオリジナルの列名です。この接頭部は、ADDDPRREG コマンドの **PREFIX** パラメーターに指定した文字です。 ***ALL (デフォルト)** ソースに登録した列のすべてがターゲット表に含まれます。 ***NONE** ソース表からの列は 1 つもターゲット表に含まれません。 *NONE は、算出列だけをターゲット表に含めたい場合に使用します。この値は、**CALCCOL** パラメーターに合計関数が含まれているが、GROUP BY が実行されない、という場合に必要です。 *column-name* ターゲット表に含めたいソース列の名前を 300 個まで指定できます。列名はスペースで区切ります。

表 39. ADDDPRSUB コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
UNIQUE	ターゲット表が**KEYCOL** パラメーターで示されたユニーク・キーを持つかどうかを指定します。 ***YES (デフォルト)** ターゲット表はキーごとに正味 1 つの変更をサポートします。つまり、キーに対していかに多くの変更がなされたとしても、そのキーについてはターゲット表に 1 つの行しか存在しないということです。 この値は、表がデータの変更の履歴ではなく、現行のデータを含むことを指定します。コンデンス表には、1 つの主キー値に対して複数の行が含まれることはなく、リフレッシュ用の最新情報を提供するために使用できます。 ***NO** ターゲット表はキーごとに複数の変更をサポートします。変更はターゲット表に付加されます。 この値は、表が現行のデータではなく、変更の履歴を含むことを指定します。非コンデンス表には、それぞれのキー値に対して複数の行が含まれ、データの変更履歴を提供するために使用できます。ただし、非コンデンス表はリフレッシュ用の最新データは提供できません。

表 39. ADDDPRSUB コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
KEYCOL	ターゲット表のキーを記述する列を指定します。列名は修飾できません。*POINTINTIME、*REPLICA、および *USERCOPY ターゲット表 (**TGTTYPE** パラメーターで指定されている) の場合、ターゲット表に 1 つまたは複数の列をターゲット・キーとして指定する必要があります。アプライ・プログラムはこのターゲット・キーを使用して、変更キャプチャー・レプリケーション中に、変更された個々のユニークな行を識別します。 ***SRCTBL (デフォルト)** ターゲット表のキー列は、ソース表のキー列と同じです。ADDDPRREG コマンドは、ソース表がキー付きの場合、ソース表に指定されたキーを使用します。以下のキー列が使用されます。 • 物理ファイル作成コマンド (CRTPF) を使用して表を作成した時に、DDS を使用して定義したキー列 • CREATE TABLE および ALTER TABLE SQL ステートメントを使用して定義した、主キーおよびユニーク・キー • CREATE INDEX SQL ステートメントを使用して定義したユニーク・キー 1 つの列を、キーとして、異なる順序付けで複数回使用すると、ターゲット表のキーは昇順で定義されます。 ***RRN** ターゲット表のキー列は IBMQSQ_RRN 列です。ターゲット表は IBMQSQ_RRN 列を使用して作成され、この列がキーとして使用されます。アプライ・プログラムの実行時に、ソース表がユーザー表であり、ターゲット表がポイント・イン・タイムまたはユーザー・コピーの場合、ターゲット表の IBMQSQ_RRN 列が、ソース表内の関連するレコードの RRN (相対レコード番号) で更新されます。それ以外では、ターゲット表の IBMQSQ_RRN 列は、ソース表内の IBMQSQ_RRN 列の値で更新されます。 ***NONE** ターゲット・コピーはターゲット・キーを含みません。ターゲット表のタイプが *POINTINTIME、*REPLICA、または *USERCOPY の場合、 *NONE は指定できません。 *column-name* ターゲット・キー列として使用するターゲット列の名前。列名は 120 個まで指定できます。列名はスペースで区切ります。

表 39. ADDDPRSUB コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
TGTCOL	アプライ・プログラムがターゲット表内で更新するすべての列の新しい名前を指定します。これらの名前は、ソース表から取られた列名をオーバーライドします。列名は修飾できません。**COLUMN** パラメーターに *NONE を指定した場合はこのパラメーターを使用しないでください。 このパラメーターを使用して、ターゲット表の列により分かりやすい名前を付けることができます。 それぞれのソース列の名前と、ターゲット表の対応する列の名前を指定します。 ***COLUMN (デフォルト)** ターゲット列は、**COLUMN** パラメーターに指定した列と同じです。 *column-name* ターゲットで変更するソース表の列名。列名は 300 個まで指定できます。 *new-name* ターゲット列の新しい名前。新しい列名を 300 個まで指定できます。このパラメーターを使用しない場合、ターゲット表の列名はソースの列名と同じになります。
CALCCOL	ターゲット表のユーザー定義の列または算出された列のリストを指定します。列名は修飾できません。それぞれの列名と式の対を括弧で囲みます。 各 SQL 式には列名を指定する必要があります。 GROUP BY ステートメントのない SQL 式として列を定義する場合は、**COLUMN** パラメーターを *NONE にする必要があります。 ***NONE (デフォルト)** ターゲット表はユーザー定義の列または算出された列を含みません。 *column-name* ターゲット表のユーザー定義の列または算出された列の列名。列名は 100 個まで指定できます。 *expression* ターゲット表のユーザー定義の列または算出された列の式。 SQL 列の式は 100 個まで指定できます。

表 39. ADDDPRSUB コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
ADDREG	ターゲット表をソース表として自動的に登録するかどうかを指定します。このパラメーターは CCD ターゲット・タイプの表を登録する場合に使用します。 ***NO (デフォルト)** ターゲット表はソース表として登録されません。 DB2 DataPropagator for System i は、ターゲット・タイプが *REPLICA の場合、このパラメーター値を無視します。レプリカ・ターゲット表は必ず、ソース表として自動的に登録されます。 ***YES** ターゲット表はソース表として登録されます。ターゲット表をすでにユーザーが登録していると、このコマンドは失敗します。 ターゲット表のタイプが *USERCOPY、*POINTINTIME、*BASEAGR、または *CHANGEAGR の場合、このパラメーターを *YES にしないでください。 **CRTTGTTBL** パラメーターを *NO にした場合、これをソースとして登録する前に、ターゲット表を作成する必要があります。
ROWSLT	SQL の WHERE 文節に入れる述部を指定します。アプライ・プログラムはこれらの述部を使用して、ソースの変更データ (CD) 表のどの行をターゲット表に適用するかを決めます。ソースの変更のサブセットだけをターゲット表に複製する場合に、このパラメーターを使用します。 ***ALL (デフォルト)** アプライ・プログラムは、CD 表内のすべての変更をターゲット表に適用します。 *WHERE-clause* アプライ・プログラムが CD 表からどの行をターゲット表に適用するかを指定する SQL の WHERE 文節です。 WHERE キーワードはこのパラメーターで暗黙に想定されているので、 WHERE キーワードを含めないでください。この WHERE 文節は、この文節を実行するデータ・サーバー上で有効なものでなければなりません。 **注:**このパラメーターの WHERE 文節は、 SQLBEFORE または SQLAFTER パラメーターに指定された WHERE 文節とはまったく関係ありません。

表 39. ADDDPRSUB コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
MAXSYNCH	最大の同期化の分数を指定します。このパラメーターは、時間しきい値の制限を指定し、サブスクリプションのサイクル中に、キャプチャー・プログラムおよびアプライ・プログラムが処理する変更データの量を規制するために使用されます。時間しきい値の制限は 2 つの部分からなる値で指定します。最初の部分は数値、2 番目の部分は時間の単位です。 ***MIN** 分 ***HOUR** 時間 ***DAY** 日 ***WEEK** 週 時間の単位と一緒に、数値の組み合わせを指定することができます。例えば、((1 *WEEK) (2 *DAY) (35 *MIN)) は、1 週、2 日、および 35 分の時間間隔を指定します。同じ時間単位で複数を指定すると、最後の指定が使用されます。 デフォルトはゼロ (0) であり、変更データのすべてを適用することを示します。

表 39. ADDDPRSUB コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
SQLBEFORE	アプライ・プログラムがターゲット表をリフレッシュする前に実行する、 SQL ステートメントを指定します。このパラメーターは以下の 3 つのエレメントからなります。 エレメント 1: SQL コード ***NONE (デフォルト)** SQL ステートメントを指定しません。 *SQL-statement* 実行する SQL ステートメント。 SQL ステートメントの構文が正しいことを確認してください。 DB2 DataPropagator for System i は構文の妥当性を検査しません。また、適切な SQL 命名規則を使用する必要があります。 SQL ファイル参照は、システムの命名規則 (LIBRARY/FILE) ではなく、 LIBRARY.FILE の形式でなければなりません。 3 つまでの SQL ステートメントを指定できます。 エレメント 2: 実行するサーバー ***TGTSVR (デフォルト)** SQL ステートメントは、ターゲット表が存在するターゲット・サーバーで実行されます。 ***SRCSVR** SQL ステートメントは、ソース表が存在するキャプチャー・コントロール・サーバーで実行されます。 エレメント 3: 許される SQLSTATE 値 ***NONE (デフォルト)** SQLSTATE 値 00000 のみが正常と見なされます。 *SQL-states* 1 から 10 個までの許容される SQLSTATE 値のリスト。 SQLSTATE 値はスペースで区切ります。 SQLSTATE 値は、00000 から FFFFF の範囲の 5 桁の 16 進数です。 SQL ステートメントは、SQLSTATE 値 00000 または、リストされた許容可能な SQLSTATE 値の 1 つで完了した場合に、正常終了となります。

表 39. ADDDPRSUB コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
SQLAFTER	アプライ・プログラムがターゲット表をリフレッシュした後に実行する、 SQL ステートメントを指定します。このパラメーターは以下の 3 つのエレメントからなります。 エレメント 1: SQL コード ***NONE (デフォルト)** SQL ステートメントを指定しません。 *SQL-statement* 実行する SQL ステートメント。 SQL ステートメントの構文が正しいことを確認してください。 DB2 DataPropagator for System i は構文の妥当性を検査しません。また、適切な SQL 命名規則を使用する必要があります。 SQL ファイル参照は、システムの命名規則 (LIBRARY/FILE) ではなく、 LIBRARY.FILE の形式でなければなりません。 3 つまでの SQL ステートメントを指定できます。 エレメント 2: 実行するサーバー ***TGTSVR (デフォルト)** SQL ステートメントは、ターゲット表が存在するターゲット・サーバーで実行されます。 エレメント 3: 許される SQLSTATE 値 ***NONE (デフォルト)** SQLSTATE 値 00000 のみが正常と見なされます。 *SQL-states* 1 から 10 個までの許容される SQLSTATE 値のリスト。 SQLSTATE 値はスペースで区切ります。 SQLSTATE 値は、00000 から FFFFF の範囲の 5 桁の 16 進数です。 SQL ステートメントは、SQLSTATE 値 00000 または、リストされた許容可能な SQLSTATE 値の 1 つで完了した場合に、正常終了となります。

ADDDPRSUB の例

以下の例は、ADDDPRSUB コマンドの使用法を示しています。

例 1:

AQHR アプライ修飾子の下に SETHR という名前のサブスクリプション・セットを作成します。

```
ADDDPRSUB APYQUAL(AQHR) SETNAME(SETHR) SRCTBL(HR/EMPLOYEE)
  TGTTBL(TGTLIB/TGTEMPL)
```

このサブスクリプション・セットには 1 つのサブスクリプション・セット・メンバーが含まれ、 HR ライブラリー下の EMPLOYEE という名前の登録済みソース表から、 TGTLIB ライブラリー下の TGTEMPL という名前のターゲット表にデータを複製します。

例 2

2 つだけの列 (EMPNO (キー) および NAME) を持つ、SETHR という名前のサブスクリプション・セットを、EMPLOYEE という名前の登録済みソース表から作成し、これらの列を TGTEMPL という名前の既存のターゲット表に複製します。

```
ADDDPRSUB APYQUAL(AQHR) SETNAME(SETHR) SRCTBL(HR/EMPLOYEE)
  TGTTBL(TGTLIB/TGTEMPL) CRTTGTTBL(*NO) COLUMN(EMPNO NAME) KEYCOL(EMPNO)
```

例 3

SETHR という名前のサブスクリプション・セットを、 EMPLOYEE という名前の登録済みソース表からのデータを使用して作成し、このデータを TGTREPL という名前のレプリカ・タイプのターゲット表に複製します。

```
ADDDPRSUB APYQUAL(AQHR) SETNAME(SETHR) SRCTBL(HR/EMPLOYEE)
  TGTTBL(TGTLIB/TGTREPL) TGTTYPE(*REPLICA)
```

例 4

NOMEM という名前のサブスクリプション・セットを、サブスクリプション・セット・メンバーなしで作成します。

```
ADDDPRSUB APYQUAL(AQHR) SETNAME(NOMEM) SRCTBL(*NONE) TGTTBL(*NONE)
```

ADDDPRSUBM: DPR サブスクリプション・セット・メンバーの追加 (System i)

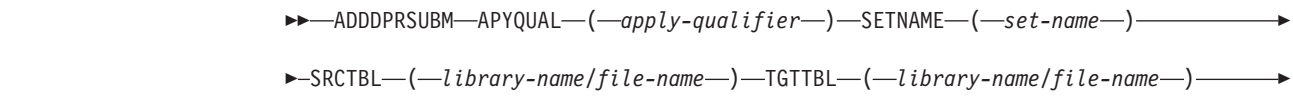
既存のサブスクリプション・セットにメンバーを追加するには、 DPR サブスクリプション・セット・メンバーの追加 (ADDDPRSUBM) コマンドを使用します。

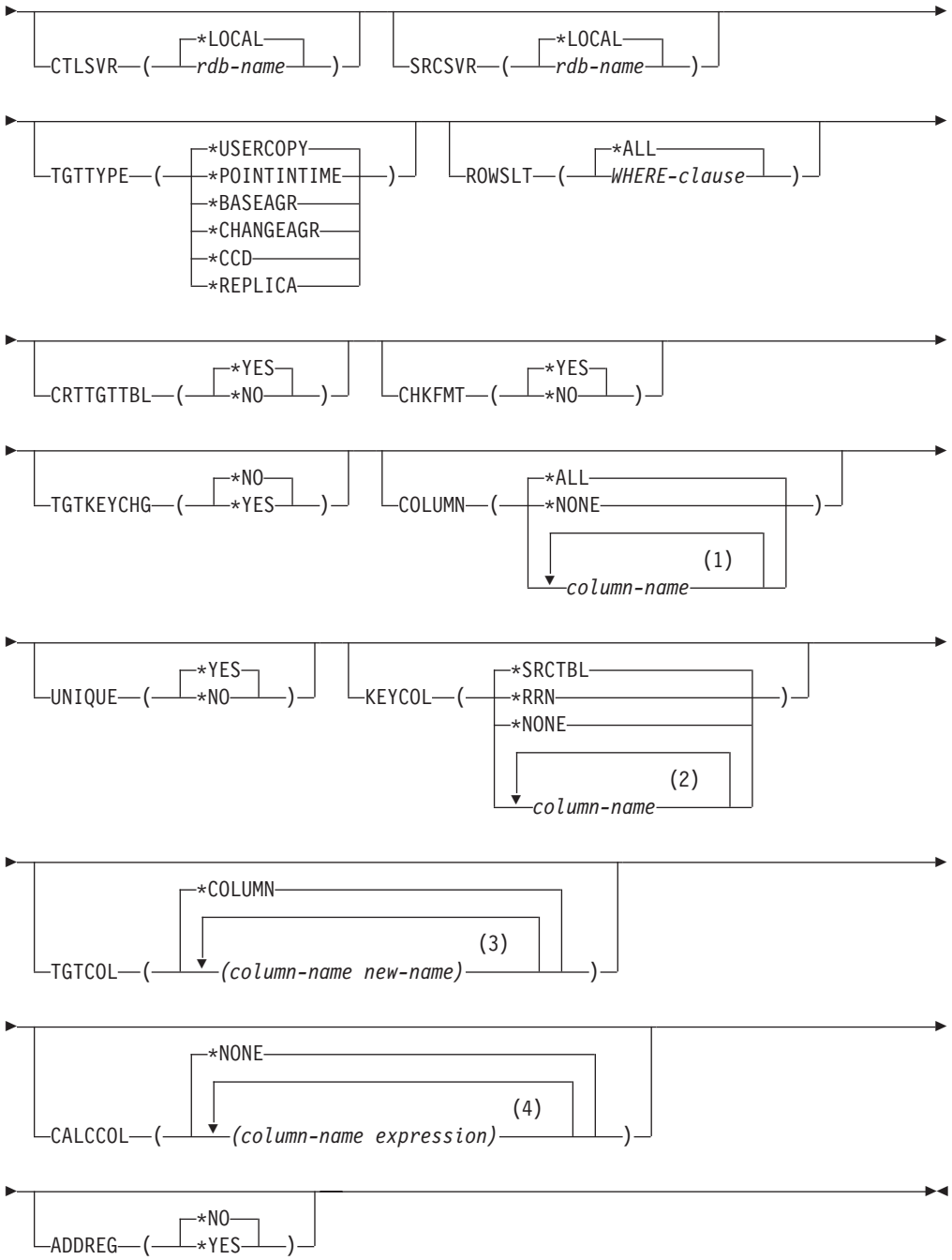
サブスクリプション・セットは、 ADDDPRSUB コマンドを使用して、UNIX、Windows、または z/OS上のシステム・コマンドを使用して、またはレプリケーション・センターから作成することができます。サブスクリプション・セット内のすべてのソース表は、すでにジャーナルに記録済みであり、登録済みでなければならず、その後でないとこのコマンドは使用できません。

コマンド行にコマンド名を入力してから、F4 キーを押してコマンド構文を表示することができます。

このコマンド、およびこのコマンドのすべてのパラメーターの詳しい記述を表示するには、画面の一番上のコマンドにカーソルを移動し、F1 キーを押します。特定のパラメーターの記述を表示するには、そのパラメーター上にカーソルを移動し、F1 キーを押します。

構文





注:

1 列名は 300 個まで指定できます。

2 列名は 120 個まで指定できます。

3 列名は 300 個まで指定できます。

4 100 個までの列名および式を指定できます。

表 40 は、呼び出しパラメーターをリストしています。

表 40. ADDDPRSUBM コマンド・パラメーター定義 (System i 版)

パラメーター	定義およびプロンプト
APYQUAL	どのアプライ・プログラムがこのサブスクリプション・セットを処理するかを示すアプライ修飾子を指定します。 アプライ修飾子の下のサブスクリプション・セットは別のジョブで実行されます。このパラメーターは必須です。 *apply-qualifier* アプライ修飾子の名前。
SETNAME	サブスクリプション・セットの名前を指定します。このパラメーターは必須です。 *set-name* サブスクリプション・セットの名前。入力するサブスクリプション・セット名は、指定されたアプライ修飾子についてユニークでなければなりません。ユニークでない場合、ADDDPRSUBM コマンドはエラーになります。アプライ・プログラムはターゲット表のセットをグループとして扱うので、何らかの理由で 1 つのターゲット表が失敗すると、そのセット全体が失敗します。
SRCTBL	このサブスクリプション・セット・メンバーのソースである表の名前を指定します。この表をサブスクリプション・セットのメンバーにするには、この表をキャプチャー・コントロール・サーバーにあらかじめ登録しておく必要があります。このパラメーターは必須です。 *library-name/file-name* ソース表の修飾名。
TGTTBL	このサブスクリプション・セット・メンバーのターゲット表の名前を指定します。 **CRTTGTTBL** パラメーターを *YES に設定し、ターゲット表が存在しない場合、ターゲット表は自動的に作成されます。このパラメーターは必須です。 *library-name/file-name* ターゲット表の修飾名。
CTLSVR	アプライ・コントロール表を含むシステムのリレーショナル・データベース名を指定します。 ***LOCAL (デフォルト)** アプライ・コントロール表はローカルに存在します (ADDDPRSUBM コマンドを実行するマシン上)。 *rdb-name* アプライ・コントロール表が置かれているリレーショナル・データベースの名前。 RDB ディレクトリー項目の作業 (WRKRDBDIRE) コマンドを使用して、この名前を検出することができます。

表 40. ADDDPRSUBM コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
SRCSVR	キャプチャー・コントロール表を含むシステムのリレーショナル・データベース名を指定します。 ***LOCAL (デフォルト)** ソース表はローカル・マシンに登録されています (ADDDPRSUBM コマンドを実行するマシン)。 *rdb-name* キャプチャー・コントロール表が存在するリレーショナル・データベースの名前。 RDB ディレクトリー項目の作業 (WRKRDBDIRE) コマンドを使用して、この名前を検出することができます。

表 40. ADDDPRSUBM コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
TGTTYPE	ターゲット表のタイプを指定します。これらは、ターゲット表の内容を記述する SQL レプリケーション用語です。これらのいずれかのタイプとしてターゲット表を作成した後、 DPR 登録の追加 (ADDDPRREG) コマンドの **SRCTBL** パラメーターにこのパラメーター値を使用し、このターゲット表をソース表として登録することができます。 ***USERCOPY (デフォルト)** ターゲット表はユーザー・コピーであり、これはソース表の内容のすべてまたは一部と一致する内容を持つ、ターゲット表です。ユーザー・コピーはポイント・イン・タイムの表のように扱われますが、ポイント・イン・タイムのターゲット表に存在する、DB2 DataPropagator for System i システム列は 1 つも含まれていません。 この値は、**KEYCOL** パラメーターに *RRN の値が指定されている場合は無効です。 **SRCTBL** パラメーターで指定した表は、ユーザー・データベース、ポイント・イン・タイム表、または整合変更データ (CCD) のいずれかでなければなりません。 **重要:** ターゲット表がすでに存在する場合、DB2 DataPropagator for System i は、これに対する変更を自動的にジャーナルに記録しません。ジャーナリングは、DB2 DataPropagator for System i の外側で開始する必要があります。 ***POINTINTIME** ターゲット表はポイント・イン・タイム表です。ポイント・イン・タイム表は、ソース表の内容の一部またはすべてと一致する内容を持つターゲット表であり、DB2 DataPropagator for System i システム列 (IBMSNAP_LOGMARKER) を保持します。この列は、特定の行がキャプチャー・コントロール・サーバーでいつ挿入または変更されたかを示します。 ***BASEAGR** ターゲット表は基本集約表であり、これはソース表から集約された (算出された) データを含む、ターゲット表です。基本集約ターゲットのソース表は、ユーザー表またはポイント・イン・タイム表のどちらかでなければなりません。このターゲット表には、システム・タイム・スタンプ列の IBMSNAP_HLOGMARKER および IBMSNAP_LLOGMARKER が含まれます。 ***CHANGEAGR** 表は変更集約表であり、これは、変更データ (CD) 表の内容を基に集約された (算出された) データを含むターゲット表です。このターゲット表は、システム・タイム・スタンプ列 IBMSNAP_HLOGMARKER および IBMSNAP_LLOGMARKER を使用して作成されます。

表 40. ADDDPRSUBM コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
TGTTYPE (続き)	***CCD** 表は整合変更データ (CCD) 表であり、これは、変更データ (CD) 表と作業単位 (UOW) 表内のデータを結合したものから作成されたターゲット表です。 CCD 表は、アプライ・プログラムにトランザクション整合性のあるデータを提供し、次の列を含む必要があります。 • IBMSNAP_INTENTSEQ • IBMSNAP_OPERATION • IBMSNAP_COMMITSEQ • IBMSNAP_LOGMARKER ***REPLICA** ターゲット表はレプリカ表であり、これは Update-anywhere レプリケーションにのみ使用されます。レプリカ・ターゲット表はマスター・ソース表から変更を受信し、またレプリカ・ターゲット表への変更は、マスター・ソース表に戻して伝搬されます。レプリカ表はソース表として自動的に登録されます。
ROWSLT	SQL の WHERE 文節に入れる述部を指定します。アプライ・プログラムはこれらの述部を使用して、ソースの変更データ (CD) 表のどの行をターゲット表に適用するかを決めます。ソースの変更のサブセットだけをターゲット表に複製する場合に、このパラメーターを使用します。 ***ALL (デフォルト)** アプライ・プログラムは、CD 表内のすべての変更をターゲット表に適用します。 *WHERE-clause* アプライ・プログラムが CD 表からどの行をターゲット表に適用するかを指定する SQL の WHERE 文節です。 WHERE キーワードはこのパラメーターで暗黙に想定されているので、 WHERE キーワードを含めないでください。この WHERE 文節は、この文節を実行するデータ・サーバー上で有効なものでなければなりません。 **注:**このパラメーターの WHERE 文節は、 SQLBEFORE または SQLAFTER パラメーターに指定された WHERE 文節とはまったく関係ありません。

表 40. ADDDPRSUBM コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
CRTTGTTBL	ターゲット表 (またはビュー) を作成するかどうかを指定します。 ***YES (デフォルト)** ターゲット表 (またはビュー) が存在しなければ、作成します。存在する場合は、既存の表またはビューがターゲットになり、この既存の表またはビューのフォーマットの**CHKFMT** パラメーターが *YES に設定されているかどうかをチェックします。**UNIQUE** および **KEYCOL** パラメーターに指定された値を使用して、 (そのような索引がまだ存在しなければ) ターゲット表に追加の索引が作成されます。既存のターゲット表に、追加索引の条件に違反するような行が含まれていると、コマンドは失敗します。 ***NO** ターゲット表またはビューを作成しません。アプライ・プログラムを始動する前に、正しい属性を使用して表またはビューを作成する必要があります。 表またはビューが存在する場合に **CHKFMT** を *YES に設定すると、ADDDPRSUBM コマンドは、既存の表のフォーマットが、設定されたサブスクリプション・セット定義と一致することを確認します。**CHKFMT** を *NO にする場合は、ユーザーは既存の表のフォーマットがサブスクリプション・セット定義と一致することを確認しておく必要があります。 **重要:** 表またはビューがすでに存在する場合、DB2 DataPropagator for System i は、既存のオブジェクトへの変更を自動的にジャーナルに記録しません。ジャーナリングは、DB2 DataPropagator for System i の外側で開始する必要があります。
CHKFMT	DB2 DataPropagator for System i が、サブスクリプション・セット・メンバーの定義を既存のターゲット表に照らし合わせ、列が一致することをチェックするかどうかを指定します。このパラメーターは、**CRTTGTTBL** パラメーターが *YES の場合は無視され、また **CRTTGTTBL** パラメーターが *NO でターゲット表が存在しない場合も無視されます。 ***YES (デフォルト)** DB2 DataPropagator for System i は、このサブスクリプション・セット・メンバーに定義された列が、ターゲット表内の列と一致するか検証します。両者が一致しない場合、このコマンドは失敗します。 ***NO** DB2 DataPropagator for System i は、サブスクリプション・セット・メンバーと既存のターゲット表間の相違を無視します。ユーザーはターゲット表がサブスクリプション・セット・メンバーと互換性があることを確認する必要があります。

表 40. ADDDPRSUBM コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
TGTKEYCHG	ターゲット表のターゲット・キー列の一部であるソース列に変更があった場合、アプライ・プログラムがその更新をどのように扱うかを指定します。このパラメーターは、ADDDPRREG コマンドの **USEDELINS** パラメーターと組み合わせて働きます。 • **USEDELINS** が YES で **TGTKEYCHG** が YES の場合、更新はできません。 • **USEDELINS** が YES で **TGTKEYCHG** が NO の場合、更新は削除と挿入の対になります。 • **USEDELINS** が NO で **TGTKEYCHG** が YES の場合、アプライ・プログラムは特別な論理を使用してこの条件を扱います。 • **USEDELINS** が NO で **TGTKEYCHG** が NO の場合、アプライ・プログラムは変更を通常の更新として処理します。 ***NO (デフォルト)** ソース表に対する更新は、キャプチャー・プログラムによりステージ化され、アプライ・プログラムによりターゲット表に処理が行われます。 ***YES** アプライ・プログラムは、ターゲット・キー列の変更前イメージに基づいてターゲット表を更新します。つまり、アプライ・プログラムは述部を新しい値ではなく、古い値に変更します。
COLUMN	ターゲット表に含める列を指定します。 列名は修飾できません。列名は、ソース表の登録時に **CAPCOL** パラメーターに指定した列名のリストから選択してください。 この表の登録時に **IMAGE** パラメーターを *BOTH に設定した場合は、変更前イメージ列名を指定することができます。変更前イメージ列名は、接頭部を持つオリジナルの列名です。この接頭部は、ADDDPRREG コマンドの **PREFIX** パラメーターに指定した文字です。 ***ALL (デフォルト)** ソースに登録した列のすべてがターゲット表に含まれます。 ***NONE** ソース表からの列は 1 つもターゲット表に含まれません。 *NONE は、算出列だけをターゲット表に含めたい場合に使用します。この値は、**CALCCOL** パラメーターに合計関数が含まれているが、グループ化が実行されない、という場合に必要です。 *column-name* ターゲット表に含めたいソース列の名前を 300 個まで指定できます。列名はスペースで区切ります。

表 40. ADDDPRSUBM コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
UNIQUE	ターゲット表が**KEYCOL** パラメーターで示されたユニーク・キーを持つかどうかを指定します。 ***YES (デフォルト)** ターゲット表はキーごとに正味 1 つの変更をサポートします。つまり、キーに対していかに多くの変更がなされたとしても、そのキーについてはターゲット表に 1 つの行しか存在しないということです。 この値は、表がデータの変更の履歴ではなく、現行のデータを含むことを指定します。コンデンス表には、1 つの主キー値に対して複数の行が含まれることはなく、リフレッシュ用の最新情報を提供するために使用できます。 ***NO** ターゲット表はキーごとに複数の変更をサポートします。変更はターゲット表に付加されます。 この値は、表が現行のデータではなく、変更の履歴を含むことを指定します。非コンデンス表には、それぞれのキー値に対して複数の行が含まれ、データの変更履歴を提供するために使用できます。ただし、非コンデンス表はリフレッシュ用の最新データは提供できません。

表 40. ADDDPRSUBM コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
KEYCOL	ターゲット表のキーを記述する列を指定します。列名は修飾できません。*POINTINTIME、*REPLICA、および *USERCOPY ターゲット表 (**TGTTYPE** パラメーターで指定されている) の場合、ターゲット表に 1 つまたは複数の列をターゲット・キーとして指定する必要があります。アプライ・プログラムはこのターゲット・キーを使用して、変更キャプチャー・レプリケーション中に、変更された個々のユニークな行を識別します。 ***SRCTBL (デフォルト)** ターゲット表のキー列は、ソース表のキー列と同じです。ADDDPRREG コマンドは、ソース表にキーがある場合、ソース表に指定されたキーを使用します。以下のキー列が使用されます。 • 物理ファイル作成コマンド (CRTPF) を使用して表を作成した時に、DDS を使用して定義したキー列 • CREATE TABLE および ALTER TABLE SQL ステートメントを使用して定義した、主キーおよびユニーク・キー • CREATE INDEX SQL ステートメントを使用して定義したユニーク・キー 1 つの列を、キーとして、異なる順序付けで複数回使用すると、ターゲット表のキーは昇順で定義されます。 ***RRN** ターゲット表のキー列は IBMQSQ_RRN 列です。ターゲット表は IBMQSQ_RRN 列を使用して作成され、この列がキーとして使用されます。アプライ・プログラムの実行時に、ソース表がユーザー表であり、ターゲット表がポイント・イン・タイム表またはユーザー・コピーの場合、ターゲット表の IBMQSQ_RRN 列が、ソース表内の関連するレコードの RRN (相対レコード番号) で更新されます。それ以外では、ターゲット表の IBMQSQ_RRN 列は、ソース表内の IBMQSQ_RRN 列の値で更新されます。 ***NONE** ターゲット・コピーはターゲット・キーを含みません。ターゲット表のタイプが *POINTINTIME、*REPLICA、または *USERCOPY の場合、 *NONE は指定できません。 *column-name* ターゲット・キー列として使用するターゲット列の名前。列名は 120 個まで指定できます。列名はスペースで区切ります。

表 40. ADDDPRSUBM コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
TGTCOL	アプライ・プログラムがターゲット表内で更新するすべての列の新しい名前を指定します。これらの名前は、ソース表から取られた列名をオーバーライドします。列名は修飾できません。**COLUMN** パラメーターに *NONE を指定した場合は、**TGTCOL** パラメーターを使用しないでください。 このパラメーターを使用して、ターゲット表の列により分かりやすい名前を付けることができます。 それぞれのソース列の名前と、ターゲット表の対応する列の名前を指定します。 ***COLUMN (デフォルト)** ターゲット列は、**COLUMN** パラメーターに指定した列と同じです。 *column-name* ターゲットで変更するソース表の列名。列名は 300 個まで指定できます。 *new-name* ターゲット列の新しい名前。新しい列名を 300 個まで指定できます。このパラメーターを使用しない場合、ターゲット表の列名はソースの列名と同じになります。
CALCCOL	ターゲット表のユーザー定義の列または算出された列のリストを指定します。列名は修飾できません。それぞれの列名と式の対を括弧で囲みます。 各 SQL 式には列名を指定する必要があります。 GROUP BY 文節のない SQL 式として列を定義する場合は、**COLUMN** パラメーターを *NONE にする必要があります。 ***NONE (デフォルト)** ターゲット表はユーザー定義の列または算出された列を含みません。 *column-name* ターゲット表のユーザー定義の列または算出された列の列名。列名は 100 個まで指定できます。 *expression* ターゲット表のユーザー定義の列または算出された列の式。 SQL 列の式は 100 個まで指定できます。

表 40. ADDDPRSUBM コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
ADDREG	ターゲット表をソース表として自動的に登録するかどうかを指定します。このパラメーターは CCD ターゲット・タイプの表を登録する場合に使用します。 ***NO (デフォルト)** ターゲット表はソース表として登録されません。 DB2 DataPropagator for System i は、ターゲット・タイプが *REPLICA の場合、このパラメーター値を無視します。レプリカ・ターゲット表は必ず、ソース表として自動的に登録されます。 ***YES** ターゲット表はソース表として登録されます。ターゲット表をすでにユーザーが登録していると、このコマンドは失敗します。 ターゲット表のタイプが *USERCOPY、*POINTINTIME、*BASEAGR、または *CHANGEAGR の場合、このパラメーターを *YES にしないでください。 **CRTTGTTBL** パラメーターを *NO にした場合、これをソースとして登録する前に、ターゲット表を作成する必要があります。

ADDDPRSUBM の例

以下の例は、ADDDPRSUBM コマンドの使用法を示しています。

例 1:

サブスクリプション・セット・メンバーを AQHR アプライ修飾子の下の SETHR という名前のサブスクリプション・セットに追加します。

```
ADDDPRSUBM APYQUAL(AQHR) SETNAME(SETHR) SRCTBL(HR/YTDTAX) TGTTBL(TGTHR/TGTTAX)
```

例 2

2 つだけの列 (AMOUNT および NAME) を持つサブスクリプション・セット・メンバーを、YTDTAX という名前の登録済みソース表から追加し、これらの列を TGTTAX という名前の既存のターゲット表に複製します。

```
ADDDPRSUBM APYQUAL(AQHR) SETNAME(SETHR) SRCTBL(HR/YTDTAX) TGTTBL(TGTLIB/TGTTAX)
  CRTTGTTBL(*NO) COLUMN(AMOUNT NAME) CHKFMT(*YES)
```

このコマンドは、このサブスクリプション・セット・メンバーに定義された AMOUNT 列と NAME 列が、ターゲット表内の列と一致するか検証します。

例 3

サブスクリプション・セット・メンバーを SETHR という名前のサブスクリプション・セットに追加し、このデータを TGTYTD という名前の整合変更データ・ターゲット表に複製します。

```
ADDDPRSUBM APYQUAL(AQHR) SETNAME(SETHR) SRCTBL(HR/YTDTAX) TGTTBL(TGTLIB/TGTYTD)
  TGTTYPE(*CCD) ADDREG (*YES)
```

このコマンドは、ターゲット表を DB2 DataPropagator for System i のソース表として登録します。

ANZDPR: アナライザーの操作 (System i)

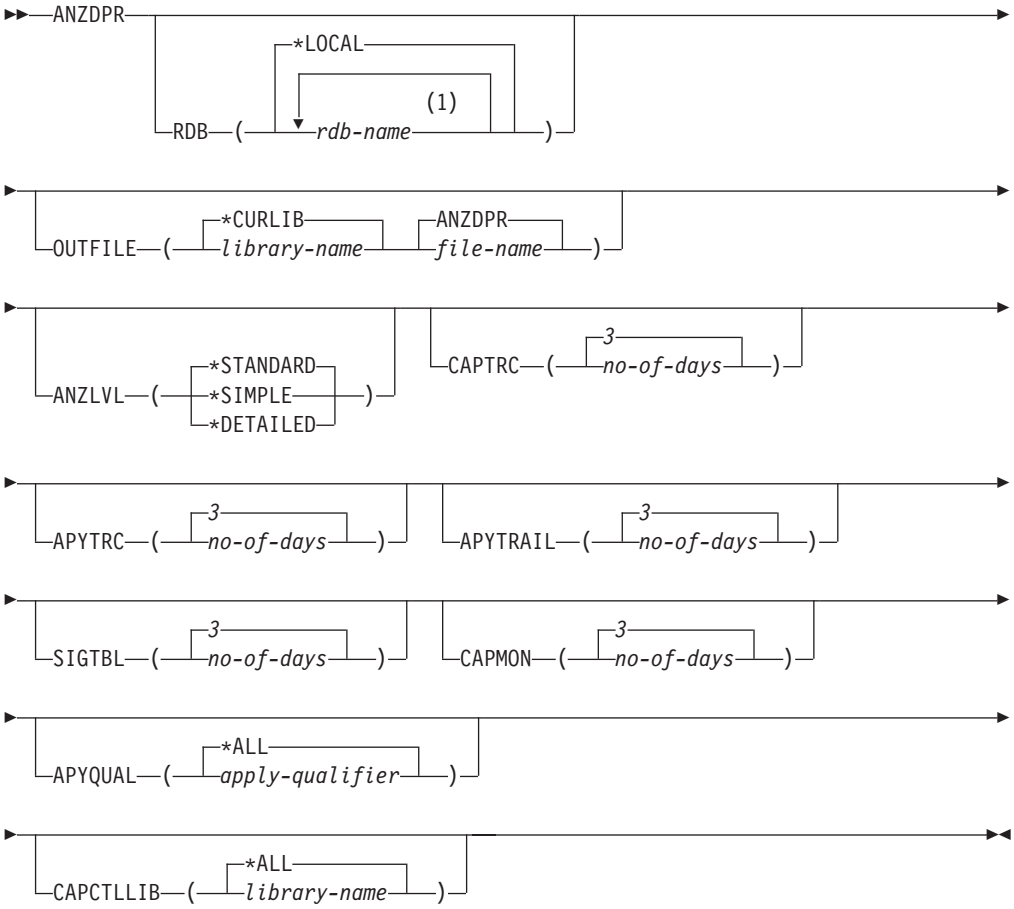
キャプチャー・プログラムまたはアプライ・プログラムの障害を分析したり、レプリケーション構成のセットアップを検証したり、問題診断およびパフォーマンス・チューニングの情報を入手するには、 DPR の分析 (ANZDPR) コマンドを使用します。

このコマンドは、レプリケーション構成をセットアップしてから実行してください。

コマンド行にコマンド名を入力してから、F4 キーを押してコマンド構文を表示することができます。

このコマンド、およびこのコマンドのすべてのパラメーターの詳しい記述を表示するには、画面の一番上のコマンドにカーソルを移動し、F1 キーを押します。特定のパラメーターの記述を表示するには、そのパラメーター上にカーソルを移動し、F1 キーを押します。

構文



注:

1 最大 10 のデータベースを指定できます。

表 41 は、呼び出しパラメーターをリストしています。

表 41. ANZDPR コマンド・パラメーター定義 (System i 版)

パラメーター	定義およびプロンプト
RDB	分析対象のデータベースを指定します。 ***LOCAL (デフォルト)** ローカル・システム上のデータベース。 *rdb-name* データベースを示す、RDB ディレクトリー項目名。 最大 10 のデータベースを入力できます。ローカル・システム上のデータベースを含めて複数のデータベースを分析する場合は、リストの最初の項目を *LOCAL にしてください。また、現行システムからこれらすべてのデータベースに接続できることを確認してください。
OUTFILE	アナライザー出力の保管に使用されるライブラリーとファイル名を指定します。このコマンドは、出力を HTML ファイルに書き込みます。 ***CURLIB (デフォルト)** 現行ライブラリー。 *library-name* ライブラリーの名前。 **ANZDPR (デフォルト)** 出力は、ANZDPR という名前の HTML ファイルに書き込まれます。 *file-name* HTML 出力ファイルの名前。 ファイル名がすでに存在する場合、ファイルは上書きされます。ファイル名が存在しない場合は、RCDLEN(512) および SIZE(*NOMAX) という属性のファイルがコマンドにより作成されます。
ANZLVL	報告される分析のレベルを指定します。分析のレベルは次のとおりです。 ***STANDARD (デフォルト)** コントロール表の内容と、キャプチャー・プログラムおよびアプライ・プログラムの状況情報を含むレポートが生成されます。 ***SIMPLE** 標準レポートで情報を生成しますが、サブカラムの詳細は含まれません。システム・リソースの使用量を抑えて小さなレポートを生成する場合は、このオプションを使用してください。 ***DETAILED** 最大限に詳細な分析を含むレポートを生成します。詳細レポートには、サブスクリプション・セット情報に加えて、標準レポートの情報が含まれます。

表 41. ANZDPR コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
CAPTRC	IBMSNAP_CAPTRACE 表から報告される項目の日付範囲 (0 から 30 日) を指定します。デフォルトは 3 です。 *no-of-days* 報告される日数。
APYTRC	IBMSNAP_APPLYTRACE 表から報告される項目の日付範囲 (0 から 30 日) を指定します。デフォルトは 3 です。 *no-of-days* 報告される日数。
APYTRAIL	IBMSNAP_APPLYTRAIL 表から報告される項目の日付範囲 (0 から 30 日) を指定します。デフォルトは 3 です。 *no-of-days* 報告される日数。
SIGTBL	IBMSNAP_SIGNAL 表から報告される項目の日付範囲 (0 から 30 日) を指定します。デフォルトは 3 です。 *no-of-days* 報告される日数。
CAPMON	IBMSNAP_CAPMON 表から報告される項目の日付範囲 (0 から 30 日) を指定します。デフォルトは 3 です。 *no-of-days* 報告される日数。
APYQUAL	分析対象のアプライ修飾子を指定します。 ***ALL (デフォルト)** すべてのアプライ修飾子が分析されます。 *apply-qualifier* 分析対象のアプライ修飾子の名前。最大 10 のアプライ修飾子を入力できます。
CAPCTLLIB	分析対象のキャプチャー・コントロール・ライブラリーの名前である、キャプチャー・スキーマを指定します。特定のキャプチャー・コントロール・ライブラリーを分析するか、デフォルトの *ALL を選択してすべてのキャプチャー・コントロール・ライブラリーを分析できます。 ***ALL (デフォルト)** すべてのキャプチャー・コントロール・ライブラリーが分析されます。 *library-name* 分析対象の、特定のキャプチャー・コントロール・ライブラリーの名前。

ANZDPR の例

以下の例は、ANZDPR コマンドの使用法を示しています。

例 1:

標準レベルの分析を使用して、ローカル・データベースと、 RMTRDB1 という名前のリモート・データベースの両方に対してアナライザーを実行するには、次のようにします。

```
ANZDPR RDB(*LOCAL RMTRDB1) OUTFILE(MYLIB/ANZDPR) ANZLVL(*STANDARD) CAPTRC(1)
  APYTRC(1) APYTRAIL(1) SIGTBL(1) CAPMON(1) APYQUAL(*ALL)
```

この例では、すべてのアプライ修飾子に関して、IBMSNAP_CAPTRACE、IBMSNAP_APPLYTRACE、IBMSNAP_APPLYTRAIL、IBMSNAP_SIGNAL、および IBMSNAP_CAPMON 表から 1 日分の項目が生成され、MYLIB という名前のライブラリー内の ANZDPR という名前の HTML ファイルに出力が書き込まれます。

例 2

すべてのデフォルト値を使用してアナライザーを実行するには、次のようにします。

```
ANZDPR
```

CHGDPRCAPA: DPR キャプチャー属性の変更 (System i)

DPR キャプチャー属性の変更 (CHGDPRCAPA) コマンドは、キャプチャー・プログラムにより使用され、IBMSNAP_CAPPARMS 表に保管されるグローバル操作パラメーターを変更するために使用されます。

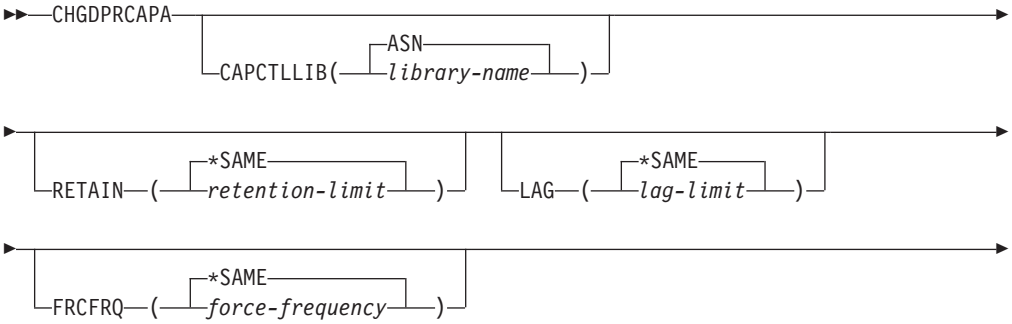
これらのパラメーターに対して行われた変更は、次のアクションのいずれかが実行されるまでは有効になりません。

- INZDPRCAP コマンドの発行。
- キャプチャー・プログラムの終了と再始動。

コマンド行にコマンド名を入力してから、F4 キーを押してコマンド構文を表示することができます。

このコマンド、およびこのコマンドのすべてのパラメーターの詳しい記述を表示するには、画面の一番上のコマンドにカーソルを移動し、F1 キーを押します。特定のパラメーターの記述を表示するには、そのパラメーター上にカーソルを移動し、F1 キーを押します。

構文



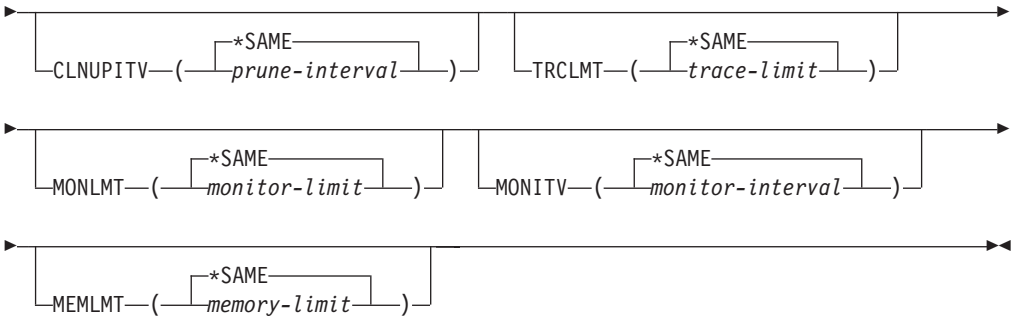


表 42 は、呼び出しパラメーターをリストしています。

表 42. CHGDPRCAPA コマンド・パラメーター定義 (System i 版)

パラメーター	定義およびプロンプト
CAPCTLLIB	キャプチャー・スキーマ (キャプチャー・コントロール表が存在するライブラリーの名前) を指定します。 **ASN (デフォルト)** キャプチャー・コントロール表は ASN ライブラリー内にあります。 *library-name* キャプチャー・コントロール表を含むライブラリーの名前。
RETAIN	データが除去されずに、変更データ (CD) 表、作業単位 (UOW) 表、IBMSNAP_SIGNAL 表、および IBMSNAP_AUTHTKN 表に保存される分数である、新規の保持制限を指定します。この値は、IBMSNAP_CAPPARMS 表の RETENTION_LIMIT 列に保管されます。 この値は、**CLNUPITV** パラメーターの値と共同で処理を行います。**CLNUPITV** の値に達すると、 CD、UOW、IBMSNAP_SIGNAL、および IBMSNAP_AUTHTKN データのうちで、保持制限よりも古いデータが削除されます。 表内のデータに矛盾が生じないように、データがこの **RETAIN** パラメーター値に達する前に変更情報がコピーされるようにアプライ・インターバルを設定してください。データの矛盾が発生した場合、アプライ・プログラムはフル・リフレッシュを実行します。 デフォルトは 10,080 分 (7 日) です。最大値は、35000000 分です。 ***SAME (デフォルト)** この値は変更されません。 *retention-limit* 新しい保持制限値。

表 42. CHGDPRCAPA コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
LAG	キャプチャー・プログラムの処理が遅れても、その間は再始動が行われない分数である、新規の遅延限度を指定します。この値は、IBMSNAP_CAPPARMS 表の LAG_LIMIT 列に保管されます。 遅延限度に達すると (つまり、ジャーナル項目のタイム・スタンプが現在の時間から遅延限度を引いたものより古くなると)、キャプチャー・プログラムは、そのジャーナルで処理中の表のコールド・スタートを開始します。アプライ・プログラムはその後フル・リフレッシュを実行し、キャプチャー・プログラムに新しい開始点を提供します。 デフォルトは 10,080 分 (7 日) です。最大値は、35000000 分です。 ***SAME (デフォルト)** この値は変更されません。 *lag-limit* 新しい遅延限度値。
FRCFRQ	キャプチャー・プログラムが変更データ (CD) 表および UOW 表に変更を書き込む頻度 (30 秒から 600 秒) を指定します。この値は、IBMSNAP_CAPPARMS 表の COMMIT_INTERVAL 列に保管されます。 キャプチャー・プログラムは、バッファーがフルになるか、**FRCFRQ** 時間制限が満了するか、いずれか先に発生した時点で、これらの変更をアプライ・プログラムから使用可能にします。 このパラメーターは、ソース表の変更率が少ないサーバー上で、アプライ・プログラムが変更をより早期に使用できるようにするために使用してください。**FRCFRQ** パラメーター値はグローバル値であり、すべての定義済みソース表で使用されます。**FRCFRQ** 値を低い数値に設定すると、システム・パフォーマンスが影響を受ける可能性があります。 デフォルトは 30 秒です。 ***SAME (デフォルト)** この値は変更されません。 *force-frequency* キャプチャー・プログラムが、アプライ・プログラムで変更を使用できるようにする前に、 CD 表および UOW 表の変更をバッファー・スペース内に保持する秒数である、新規のコミット・インターバル値です。

表 42. CHGDPRCAPA コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
CLNUPITV	キャプチャー・プログラムが変更データ (CD) 表、UOW 表、IBMSNAP_SIGNAL 表、 IBMSNAP_CAPMON 表、IBMSNAP_CAPTRACE 表、および IBMSNAP_AUTHTKN 表から古いレコードを除去するまでの最大時間 (時間単位) を指定します。 このパラメーターは、**RETAIN** パラメーターと組み合わせて、CD、UOW、IBMSNAP_SIGNAL、および IBMSNAP_AUTHTKN 表の整理、**MONLMT** パラメーターと組み合わせて、IBMSNAP_CAPMON 表の整理、**TRCLMT** パラメーターと組み合わせて、IBMSNAP_CAPTRACE 表の整理をコントロールします。(キャプチャー・プログラムの **RETAIN**、**MONLMT**、および **TRCLMT** パラメーターを設定するには、STRDPRCAP コマンドを使用します。) このパラメーターの値は、自動的に時間から秒に変換され、IBMSNAP_CAPPARMS 表の PRUNE_INTERVAL 列に保管されます。PRUNE_INTERVAL 列が (CHGDPRCAPA コマンドを使用せずに) 手動で変更された場合、F4 キーを使用してプロンプトを出すと、丸めによる変更が行われている場合があります。 ***SAME (デフォルト)** このキャプチャー属性値は変更されません。 *prune-interval* 時間数 (1 から 100) で指定された、整理インターバル。
TRCLMT	トレース限度を指定します (分単位)。この値は、IBMSNAP_CAPPARMS 表の TRACE_LIMIT 列に保管されます。 キャプチャー・プログラムは、トレース限度よりも古い IBMSNAP_CAPTRACE 行を削除します。デフォルトは 10,080 分 (トレース項目が 7 日間) です。 ***SAME (デフォルト)** この値は変更されません。 *trace-limit* 整理後に、トレース・データが IBMSNAP_CAPTRACE 表に保持される分数。
MONLMT	モニター限度を指定します (分単位)。この値は、IBMSNAP_CAPPARMS 表の MONITOR_LIMIT 列に保管されます。 キャプチャー・プログラムは、モニター限度よりも古い IBMSNAP_CAPMON 行を削除します。 デフォルトは 10,080 分 (モニター項目が 7 日間) です。 ***SAME (デフォルト)** この値は変更されません。 *monitor-limit* 整理後に、モニター・データが IBMSNAP_CAPMON 表に保持される分数。

表 42. CHGDPRCAPA コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
MONITV	キャプチャー・プログラムが、 IBMSNAP_CAPMON 表に行を挿入する頻度 (秒単位) を指定します。この値は、IBMSNAP_CAPPARMS 表の MONITOR_INTERVAL 列に保管されます。 デフォルトは 300 秒 (5 分) です。 ***SAME (デフォルト)** この値は変更されません。 *monitor-interval* IBMSNAP_CAPMON 表への行挿入の間隔の秒数。モニター・インターバルは、少なくとも 120 秒 (2 分) あける必要があります。120 よりも小さい数値がユーザーから指定された場合、このコマンドは自動的にパラメーター値を 120 に設定します。
MEMLMT	キャプチャー・ジャーナル・ジョブが使用できるメモリーの最大サイズ (MB 単位) を指定します。この値は、IBMSNAP_CAPPARMS 表の MEMORY_LIMIT 列に保管されます。 デフォルトは 32 MB です。 ***SAME (デフォルト)** この値は変更されません。 *memory-limit* メモリーの最大値の MB 数。

CHGDPRCAPA の例

以下の例は、CHGDPRCAPA コマンドの使用法を示しています。

例 1:

キャプチャー・プログラムが IBMSNAP_CAPMON 表に行う行挿入の頻度を 6,000 秒 (100 分) 間隔に変更するには、次のようにします。

```
CHGDPRCAPA CAPCTLLIB(ASN) MONITV(6000)
```

この頻度の値は、デフォルト ASN ライブラリー内の IBMSNAP_CAPPARMS 表に保管されます。

例 2

LIB1 と呼ばれるキャプチャー・コントロール・ライブラリー内の IBMSNAP_CAPPARMS 表で保持制限、遅延限度、トレース限度、およびモニター限度を変更するには、次のようにします。

```
CHGDPRCAPA CAPCTLLIB(LIB1) RETAIN(6000) LAG(3000) TRCLMT(3000) MONLMT(6000)
```

例 3

キャプチャー・プログラムが CD 表および UOW 表に変更を書き込む頻度を示すコミット・インターバルを変更するには、次のようにします。

```
CHGDPRCAPA CAPCTLLIB(ASN) FRCFRQ(360)
```

CRTDPRTBL: レプリケーション・コントロール表の作成 (System i)

DPR 表の作成 (CRTDPRTBL) コマンドを使用して、誤って削除されたり破壊されたりしたレプリケーション・コントロール表を作成します。

重要: CRTDPRTBL コマンドは、System i のコントロール表を作成するために使用できる唯一のコマンドです。レプリケーション・センターまたは ASNCLP コマンド行プログラムを使用して、コントロール表を作成しないでください。

制約事項: 代替キャプチャー・スキーマを作成する場合は、 ASN ライブラリーが存在するのと同じ補助記憶域プール (基本または独立のいずれか) 内に作成しなければなりません。

コマンド行にコマンド名を入力してから、F4 キーを押してコマンド構文を表示することができます。

このコマンド、およびこのコマンドのすべてのパラメーターの詳しい記述を表示するには、画面の一番上のコマンドにカーソルを移動し、F1 キーを押します。特定のパラメーターの記述を表示するには、そのパラメーター上にカーソルを移動し、F1 キーを押します。

構文

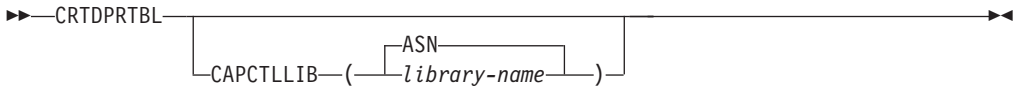


表 43 は、呼び出しパラメーターをリストしています。

表 43. CRTDPRTBL コマンド・パラメーター定義 (System i 版)

パラメーター	定義およびプロンプト
CAPCTLLIB	新しく作成されたキャプチャー・コントロール表が置かれるライブラリーの名前である、キャプチャー・スキーマを指定します。 **ASN (デフォルト)** キャプチャー・コントロール表は ASN ライブラリー内に置かれます。 *library-name* キャプチャー・コントロール表が置かれるライブラリーの名前。

CRTDPRTBL の例

以下の例は、CRTDPRTBL コマンドの使用法を示しています。

例 1:

新しいレプリケーション・コントロール表をデフォルト ASN ライブラリー内に作成するには、次のようにします。

```
CRTDPRTBL CAPCTLLIB(ASN)
```

例 2

DPRSALES という名前のキャプチャー・スキーマの新しいレプリケーション・コントロール表を作成するには、次のようにします。

```
CRTDPRTBL CAPCTLLIB(DPRSALES)
```

ENDDPRAPY: アプライの停止 (System i)

DPR アプライ・プログラムの終了 (ENDDPRAPY) コマンドは、ローカル・システム上のアプライ・プログラムを停止するために使用されます。

計画されているシステムのダウン時間より前に、アプライ・プログラムを停止する必要があります。また、システムの使用がピークになる間、アプライ・プログラムを終了することもできます。

コマンド行にコマンド名を入力してから、F4 キーを押してコマンド構文を表示することができます。

このコマンド、およびこのコマンドのすべてのパラメーターの詳しい記述を表示するには、画面の一番上のコマンドにカーソルを移動し、F1 キーを押します。特定のパラメーターの記述を表示するには、そのパラメーター上にカーソルを移動し、F1 キーを押します。

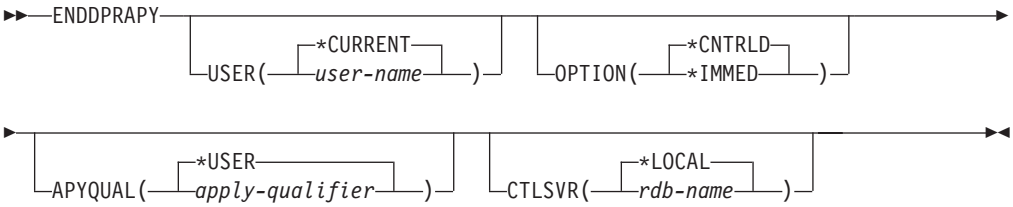


表 44 は、呼び出しパラメーターをリストしています。

表 44. ENDDPRAPY コマンド・パラメーター定義 (System i 版)

パラメーター	定義およびプロンプト
USER	このパラメーターは、**APYQUAL** パラメーターの値が *USER である場合はアプライ・プログラムに関連付けられたアプライ修飾子を指定しますが、それ以外の場合は無視されます。 ***CURRENT (デフォルト)** 現行ジョブに関連付けられたユーザーのアプライ・プログラムです。 *user-name* 指定したユーザーのアプライ・プログラム。 ENDDPRAPY コマンドでプロンプトを出すと、 F4 キーを押して、サブスクリプションを定義したユーザーのリストを表示することができます。

表 44. ENDDPRAPY コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
OPTION	アプライ・プログラムを停止する方法を指定します。 ***CNTRLD (デフォルト)** アプライ・プログラムは、停止前にすべてのタスクを完了します。アプライ・プログラムがサブスクリプション・セットを完了している場合、これらのタスクを終わらせるまでに、かなりの時間がかかる場合があります。 ***IMMED** アプライ・プログラムは、ENDJOB OPTION(*IMMED) コマンドですべてのタスクを完了します。終結処理を行うことなく、このタスクはすぐに終了します。望ましくない結果を引き起こすことがあるため、コントロールされた終了が正常に行われなかった場合にのみ、このオプションを使用してください。 (ENDDPRAPY コマンドの発行時に、アプライ・プログラムがスリープ状態でなければ、ターゲット表の目次を検査する必要があります。) アプライ・プログラムがターゲット表へのフル・リフレッシュを実行していた場合、その表がソース表の目次を使ってリフレッシュされる前にアプライ・プログラムが終了しているため、ターゲット表が空である可能性があります。ターゲット表が空の場合、このレプリケーション・ターゲットにフル・リフレッシュを行う必要があります。 サブスクリプション・セットが使用中 (IBMSNAP_SUBS_SET にある STATUS 列の値が 1 である) と見なされている場合があります。この場合、この値を 0 または -1 に設定し直してください。こうすると、アプライ・プログラムはサブスクリプション・セットを再度実行することができます。
APYQUAL	アプライ・プログラムによって使われるアプライ修飾子を指定します。 ***USER (デフォルト)** **USER** パラメーターで指定されたユーザー名がアプライ修飾子。 *apply-qualifier* このアプライ・プログラムが実行するサブスクリプション・セットをグループ化するのに使われる名前。アプライ修飾子名として最大 18 文字を指定できます。この名前は、リレーショナル・データベース名と同じ命名規則に準拠します。実行中のサブスクリプションは、APPLY_QUAL 列にあるこの値を使用して、IBMSNAP_SUBS_SET 表にあるレコードが識別します。 ENDDPRAPY コマンドでプロンプトを出すと、F4 キーを押して、既存のサブスクリプションを使用したアプライ修飾子のリストを表示することができます。

表 44. ENDDPRAPY コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
CTLSVR	アプライ・コントロール表を含むシステムのリレーショナル・データベース名を指定します。 ***LOCAL (デフォルト)** アプライ・コントロール表はローカルに (ENDDPRAPY コマンドが実行されたマシンに) 存在します。 *rdb-name* アプライ・コントロール表が置かれているリレーショナル・データベースの名前。 RDB ディレクトリー項目の作業 (WRKRDBDIRE) コマンドを使用して、この名前を検出することができます。 ENDDPRAPY コマンドでプロンプトを出すと、 F4 キーを押して、RDB ディレクトリーにあるデータベースのリストから選択することができます。

使用上の注意

ENDDPRAPY コマンドは、**APYQUAL** および **CTLSVR** パラメーターの値を使用して、参照済みのアプライ・プログラムのジョブ名、ジョブ番号、およびジョブ・ユーザーに応じた IBMSNAP_APPLY_JOB 表を検索し、そのジョブを終了します。

以下のいずれかの状態が生じると、ENDDPRAPY コマンドはエラー・メッセージを発行します。

- IBMSNAP_APPLY_JOB 表が存在していないか、または破壊された場合。
- アプライ修飾子およびコントロール・サーバー名の IBMSNAP_APPLY_JOB 表にレコードがない場合。
- アプライ・ジョブがすでに終了している場合。
- コマンドを実行しているユーザー ID が、アプライ・ジョブを終了する権限を所有していない場合。

ENDDPRAPY の例

以下の例は、ENDDPRAPY コマンドの使用法を示しています。

例 1:

AQHR アプライ修飾子を使用するアプライ・プログラムを終了するには、次のようにします。

```
ENDDPRAPY OPTION(*CNTRLD) APYQUAL(AQHR)
```

アプライ・プログラムは、すべてのタスクが完了すると終了します。

例 2

アプライ・プログラムを即時に終了するには、次のようにします。

```
ENDDPRAPY OPTION(*IMMED) APYQUAL(AQHR)
```

終結処理を行うことなく、アプライ・プログラムのタスクはすぐに終了します。

例 3

DB1X という名前のリレーショナル・データベース上にあるアプライ・コントロール表を使用するアプライ・プログラムを終了するには、次のようにします。

```
ENDDPRAPY OPTION(*CNTRLD) APYQUAL(AQHR) CTLSVR(DB1X)
```

ENDDPRCAP キャプチャーの停止 (System i)

DPR キャプチャー・プログラムの終了 (ENDDPRCAP) コマンドを使用して、キャプチャー・プログラムを停止します。

このコマンドを使用して、システムをシャットダウンする前にキャプチャー・プログラムを停止します。また、システムで実行している他のプログラムのパフォーマンスを向上させるために、システムの使用がピークになる間、プログラムを停止することもできます。

コマンド行にコマンド名を入力してから、F4 キーを押してコマンド構文を表示することができます。

このコマンド、およびこのコマンドのすべてのパラメーターの詳しい記述を表示するには、画面の一番上のコマンドにカーソルを移動し、F1 キーを押します。特定のパラメーターの記述を表示するには、そのパラメーター上にカーソルを移動し、F1 キーを押します。

構文

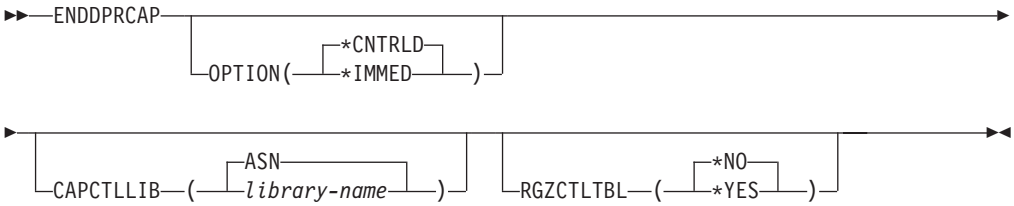


表 45 は、呼び出しパラメーターをリストしています。

表 45. ENDDPRCAP コマンド・パラメーター定義 (System i 版)

パラメーター	定義およびプロンプト
OPTION	キャプチャー・プログラムを停止する方法を指定します。 ***CNTRLD (デフォルト)** キャプチャー・プログラムは、すべてのタスクを完了した後、通常終了します。 ENDDPRCAP コマンドは、***CNTRLD** オプションを指定すると、処理が終了するまで長く時間がかかる場合があります。これは、停止する前に、キャプチャー・プログラムはそれに従属する処理をすべて完了させるからです。 ***IMMED** キャプチャー・プログラムは、ENDJOB OPTION(*IMMED) コマンドですべてのタスクを完了してから通常終了します。

表 45. ENDDPRCAP コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
CAPCTLLIB	キャプチャー・コントロール表が置かれるライブラリーの名前である、キャプチャー・スキーマを指定します。このライブラリーには、ソース表の登録情報を保管する IBMSNAP_REGISTER 表が入っています。 **ASN (デフォルト)** キャプチャー・コントロール表は ASN ライブラリー内にあります。ASN ライブラリーはデフォルト・ライブラリーです。 *library-name* キャプチャー・コントロール表を含むライブラリーの名前。
RGZCTLTBL	キャプチャー・プログラムの終了時に、コントロール表 (変更データ (CD) および作業単位 (UOW) 表を含む) に対して物理ファイル・メンバーの再編成 (RGZPFM) コマンドが実行されるかどうかを指定します。表に対して RGZPFM コマンド処理が実行されないかぎり、システムはディスク・スペースを回復しません。コントロール表がアプライ・プログラム、またはその他のアプリケーション・プログラムからアクセスされているときには、RGZPFM コマンドは実行されません。 ***NO (デフォルト)** RGZPFM コマンドは実行されません。 ***YES** RGZPFM コマンドは実行されます。

使用上の注意

ENDJOB コマンドを使用すると、一時オブジェクトが QDP4 ライブラリーに残される場合があります。 これらのオブジェクトのタイプは *DTAQ および *USRSPC で、QDP4*nnnnnn* という名前です。ここで、*nnnnnn* は、オブジェクトを使用したジョブのジョブ番号です。 これらのオブジェクトを使用したジョブ (オブジェクト名にあるジョブ番号で識別) がアクティブでないとき、オブジェクトを削除することができます。

このコマンドを発行してもキャプチャー・コントロール・ライブラリー下のジョブが終了しない場合は、*IMMED オプションを指定して ENDJOB コマンドを使用して、このジョブ、および DB2 DataPropagator for System i サブシステムで実行中のすべてのジャーナル・ジョブを終了させます。 キャプチャー・プログラムだけを終了させたい場合には、同じサブシステムで実行しているアプライ・ジョブを終了させないでください。

まれなことですが、キャプチャー・コントロール・ジョブが異常終了した場合、キャプチャー・コントロール・ジョブにより作成された (**CAPCTLLIB** パラメーターに従って命名される) ジャーナル・ジョブが実行中のまま残されることがあります。こうしたジョブを終了させる唯一の方法は、*IMMED または *CNTRLD オプションのいずれかを指定した ENDJOB コマンドを使用することです。

ENDDPRCAP の例

以下の例は、ENDDPRCAP コマンドの使用法を示しています。

例 1:

すべての処理タスクが完了した後、ASN ライブラリー内のキャプチャー・コントロール表を使用するキャプチャー・プログラムを終了するには、次のようにします。

```
ENDDPRCAP OPTION(*CNTRLD) CAPCTLLIB(ASN) RGZCTLTBL(*NO)
```

例 2

キャプチャー・スキーマ BSN のキャプチャー・プログラムを即時に終了するには、次のようにします。

```
ENDDPRCAP OPTION(*IMMED) CAPCTLLIB(BSN) RGZCTLTBL(*NO)
```

例 3

すべての処理タスクが完了した後、キャプチャー・プログラムを終了し、キャプチャー・コントロール表を再編成するには、次のようにします。

```
ENDDPRCAP OPTION(*CNTRLD) CAPCTLLIB(ASN) RGZCTLTBL(*YES)
```

GRTDPRAUT: ユーザーの許可 (System i)

DPR 権限付与 (GRTDPRAUT) コマンドを使用して、キャプチャー・プログラムとアプライ・プログラムを実行するためにレプリケーション・コントロール表にアクセスする許可をユーザーのリストに与えます。

例えば、キャプチャー・プログラムおよびアプライ・プログラムを実行しているユーザーの許可要件が、レプリケーション・ソースおよびターゲットを定義するユーザーの許可要件とは異なる場合があります。

権限を付与するための *ALLOBJ 権限を所有している必要があります。

コマンド行にコマンド名を入力してから、F4 キーを押してコマンド構文を表示することができます。

このコマンド、およびこのコマンドのすべてのパラメーターの詳しい記述を表示するには、画面の一番上のコマンドにカーソルを移動し、F1 キーを押します。特定のパラメーターの記述を表示するには、そのパラメーター上にカーソルを移動し、F1 キーを押します。

構文

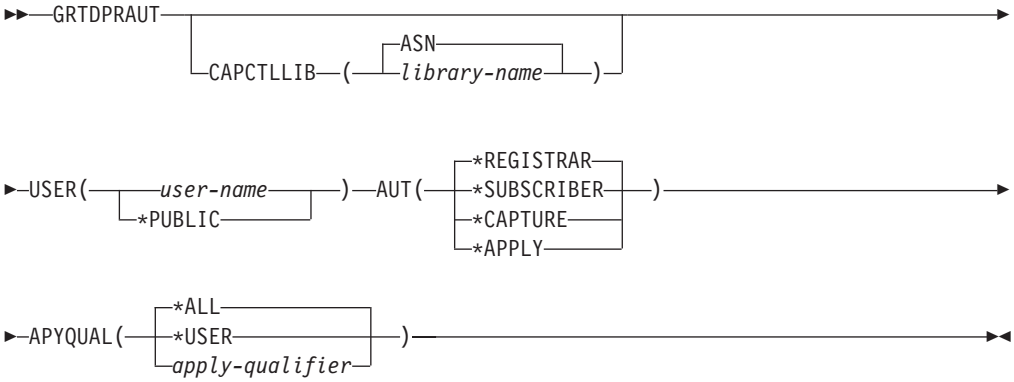


表 46 は、呼び出しパラメーターをリストしています。

表 46. GRTDPRAUT コマンド・パラメーター定義 (System i 版)

パラメーター	定義およびプロンプト
CAPCTLLIB	ユーザーが権限を付与されているレプリケーション・コントロール表を含むライブラリーである、キャプチャー・スキーマを指定します。 **ASN (デフォルト)** キャプチャー・コントロール表は ASN ライブラリー内にあります。 *library-name* レプリケーション・コントロール表を含むライブラリーの名前。
USER	権限を所有するユーザーを指定します。 *user-name* 権限を所有するユーザーの名前を最大 50 まで指定します。 ***PUBLIC** ファイルに対して *PUBLIC 許可を認可するよう指定しますが、 (その許可がタスクを行う上で不十分な場合には) 特定の許可を所有していないユーザー、ファイルに関連付けられた許可リストにないユーザー、およびグループ・プロファイルに許可がないユーザーに関してのみ使用されます。

表 46. GRTDPRAUT コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
AUT	付与されている権限のタイプを指定します。 ***REGISTRAR (デフォルト)** ユーザーは、登録を定義、変更、および消去するための権限を付与されます。 AUT(*REGISTRAR) 付き権限の完全なリストについては、 386 ページの表 47 を参照してください。 ***SUBSCRIBER** ユーザーは、サブスクリプション・セットを定義、変更、および消去するための権限を付与されます。 AUT(*SUBSCRIBER) 付き権限の完全なリストについては、 387 ページの表 48 を参照してください。 ***CAPTURE** ユーザーは、キャプチャー・プログラムを実行するための権限を付与されます。 AUT(*CAPTURE) を付与された権限の完全なリストについては、 388 ページの表 49 を参照してください。 ***APPLY** ユーザーは、アプライ・プログラムを実行するための権限を付与されます。 このコマンドは、アプライ・プログラムがアクセスする他のデータベースにあるオブジェクトに権限を付与することはありません。 アプライ・プログラムが起動されるとき、DRDA アプリケーション・サーバーのジョブに関連するユーザーは、 *APPLY 権限も付与されている必要があります。ソースが System i サーバーの場合は、**USER** パラメーターで指定されたアプリケーション・サーバー・ジョブ・ユーザーと、**APYQUAL** パラメーターで指定されたアプライ修飾子を指定して、ソース・サーバー・システム上で GRTDPRAUT コマンドを実行する必要があります。 ターゲット・サーバーがコントロール・サーバーと同じで、その両方がコマンドが実行されるシステムに常駐していない場合、権限がターゲット表に付与されることはありません。 AUT(*APPLY) を付与された権限の完全なリストについては、 390 ページの表 50 を参照してください。

表 46. GRTDPRAUT コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
APYQUAL	**USER** パラメーターで指定された、ユーザーにより使用されるアプライ修飾子を指定します。このパラメーターは、AUT(*APPLY) または AUT(*SUBSCRIBER) が指定されるときだけ使われます。 ***ALL (デフォルト)** ユーザーは、すべてのアプライ修飾子に関して、アプライ・プログラムを実行する、またはサブスクリプション・セットを定義または除去する権限を付与されます。 ***USER** **USER** パラメーターで指定されたユーザーは、ユーザー名と同じアプライ修飾子を持つサブスクリプション・セットに対する権限を付与されます。 *apply-qualifier* ユーザーは、アプライ・プログラムの実行、または、このアプライ修飾子に関連するアプライ修飾子のサブスクリプション・セットの定義および除去の権限を付与されます。 • ユーザーは、**APYQUAL** パラメーターを指定して入力された値と一致する APPLY_QUAL 列の中に値がある IBMSNAP_PRUNCNTL 表のレコードに関連付けられているすべてのレプリケーション・ソース、変更データ (CD) 表、および整合変更データ (CCD) 表に対する権限を付与されます。 • ユーザーは、このシステム上にある IBMSNAP_SUBS_MEMBR 表に記載されたサブスクリプション・セットに対する権限を付与されます。

使用上の注意

使用中のファイルでは許可を変更できないため、キャプチャー・プログラムまたはアプライ・プログラムの実行中や、ソース表を使用するアプリケーションがアクティブのときは、GRTDPRAUT コマンドを使用することはできません。

以下の表では、

- AUT(*REGISTRAR)
- AUT(*SUBSCRIBER)
- AUT(*CAPTURE)
- AUT(*APPLY)

を、GRTDPRAUT コマンドに対して指定すると付与される権限をリストします。

以下の表では、GRTDPRAUT コマンドに対して AUT(*REGISTRAR) パラメーターを指定すると付与される権限をリストします。

表 47. GRTDPRAUT AUT(*REGISTRAR) を指定して付与される権限

ライブラリー	オブジェクト	種類	許可
QSYS	capctllib	*LIB	*USE、 *ADD

*表 47. GRTDPRAUT AUT(*REGISTRAR) を指定して付与される権限 (続き)*

ライブラリー	オブジェクト	種類	許可
capctllib[1]	QSQJRN	*JRN	*OBJOPR、 *OBJMGT
capctllib[1]	QZS8CTLBLK	*USRSPC	*CHANGE
capctllib[1]	IBMSNAP_REGISTER	*FILE	*OBJOPR、 *READ、 *ADD、 *UPDT、 *DLT
capctllib[1]	IBMSNAP_REGISTERX	*FILE	*OBJOPR、 *READ、 *ADD、 *UPDT、 *DLT
capctllib[1]	IBMSNAP_REGISTERX1	*FILE	*OBJOPR、 *READ、 *ADD、 *UPDT、 *DLT
capctllib[1]	IBMSNAP_REGISTERX2	*FILE	*OBJOPR、 *READ、 *ADD、 *UPDT、 *DLT
capctllib[1]	IBMSNAP_REG_EXT	*FILE	*OBJOPR、 *READ、 *ADD、 *UPDT、 *DLT
capctllib[1]	IBMSNAP_REG_EXTX	*FILE	*OBJOPR、 *READ、 *ADD、 *UPDT、 *DLT
capctllib[1]	IBMSNAP_PRUNCNTL	*FILE	*OBJOPR、 *READ
capctllib[1]	IBMSNAP_PRUNCNTLX	*FILE	*OBJOPR、 *READ
capctllib[1]	IBMSNAP_PRUNCNTLX1	*FILE	*OBJOPR、 *READ
capctllib[1]	IBMSNAP_PRUNCNTLX2	*FILE	*OBJOPR、 *READ
capctllib[1]	IBMSNAP_PRUNCNTLX3	*FILE	*OBJOPR、 *READ
ASN	ASN4B*	*SQLPKG	*USE
ASN	ASN4C*	*SQLPKG	*USE

注:

1. ライブラリー列の *capctllib* 項目は、GRTDPRAUT コマンドの **CAPCTLLIB** パラメーターに渡される値を表します。このコマンドは、同時には 1 つのキャプチャー・コントロール・ライブラリーに対する権限しか更新できません。

以下の表では、GRTDPRAUT コマンドに対して AUT(*SUBSCRIBER) パラメーターを指定すると付与される権限をリストします。

*表 48. GRTDPRAUT AUT(*SUBSCRIBER) を指定して付与される権限*

ライブラリー	オブジェクト	種類	許可
QSYS	ASN	*LIB	*OBJOPR、 *READ、 *ADD、 *EXECUTE
QSYS	capctllib	*LIB	*OBJOPR、 *READ、 *ADD、 *EXECUTE

*表 48. GRTDPRAUT AUT(*SUBSCRIBER) を指定して付与される権限 (続き)*

ライブラリー	オブジェクト	種類	許可
ASN	IBMSNAP_SUBS_SET	*FILE	*CHANGE
ASN	IBMSNAP_SUBS_COLS	*FILE	*CHANGE
ASN	IBMSNAP_SUBS_EVENT	*FILE	*CHANGE
ASN	IBMSNAP_SUBS_STMTS	*FILE	*CHANGE
ASN	IBMSNAP_SUBS_MEMBR	*FILE	*CHANGE
capctllib[1]	IBMSNAP_REGISTER	*FILE	*OBJOPR、 *READ、 *UPD、 *EXECUTE
capctllib[1]	IBMSNAP_REG_EXT	*FILE	*OBJOPR、 *READ、 *UPD、 *EXECUTE
capctllib[1]	IBMSNAP_PRUNCNTL	*FILE	*OBJOPR、 *READ、 *DLT、 *ADD、 *EXECUTE
capctllib[1]	IBMSNAP_PRUNCNTLX	*FILE	*USE
ASN	ASN4A*	*SQLPKG	*USE
ASN	ASN4U*	*SQLPKG	*USE

注:

1. ライブラリー列の *capctllib* 項目は、GRTDPRAUT コマンドの **CAPCTLLIB** パラメーターに渡される値を表します。このコマンドは、同時には 1 つのキャプチャー・コントロール・ライブラリーに対する権限しか更新できません。

以下の表では、GRTDPRAUT コマンドに対して AUT(*CAPTURE) パラメーターを指定すると付与される権限をリストします。

*表 49. GRTDPRAUT AUT(*CAPTURE) を指定して付与される権限*

ライブラリー	オブジェクト	種類	許可
QSYS	capctllib	*LIB	*OBJOPR、 *OBJMGT、 *READ、 *EXECUTE
QSYS	QDP4	*LIB	*OBJOPR、 *ADD、 *READ、 *EXECUTE
capctllib[1]	QZSN	*MSGQ	*CHANGE
capctllib[1]	IBMSNAP_REGISTER	*FILE	*OBJOPR、 *OBJMGT、 *READ、 *ADD、 *UPD、 *EXECUTE
capctllib[1]	IBMSNAP_REGISTERX	*FILE	*OBJOPR、 *OBJMGT、 *READ、 *ADD、 *UPD、 *EXECUTE

*表 49. GRTDPRAUT AUT(*CAPTURE) を指定して付与される権限 (続き)*

ライブラリー	オブジェクト	種類	許可
capctllib[1]	IBMSNAP_REGISTERX1	*FILE	*OBJOPR、 *OBJMGT、 *READ、 *ADD、 *UPD、 *EXECUTE
capctllib[1]	IBMSNAP_REGISTERX2	*FILE	*OBJOPR、 *OBJMGT、 *READ、 *ADD、 *UPD、 *EXECUTE
capctllib[1]	IBMSNAP_REG_EXT	*FILE	*OBJOPR、 *OBJMGT、 *READ、 *ADD、 *UPD、 *EXECUTE
capctllib[1]	IBMSNAP_REG_EXTX	*FILE	*OBJOPR、 *OBJMGT、 *READ、 *ADD、 *UPD、 *EXECUTE
capctllib[1]	IBMSNAP_PRUNCNTL	*FILE	*OBJOPR、 *OBJMGT、 *READ、 *UPD、 *EXECUTE
capctllib[1]	IBMSNAP_PRUNCNTLX	*FILE	*OBJOPR、 *OBJMGT、 *READ、 *UPD、 *EXECUTE
capctllib[1]	IBMSNAP_PRUNCNTLX1	*FILE	*OBJOPR、 *OBJMGT、 *READ、 *UPD、 *EXECUTE
capctllib[1]	IBMSNAP_PRUNCNTLX2	*FILE	*OBJOPR、 *OBJMGT、 *READ、 *UPD、 *EXECUTE
capctllib[1]	IBMSNAP_PRUNCNTLX3	*FILE	*OBJOPR、 *OBJMGT、 *READ、 *UPD、 *EXECUTE
capctllib[1]	IBMSNAP_CAPTRACE	*FILE	*CHANGE
capctllib[1]	IBMSNAP_CAPTRACEX	*FILE	*CHANGE
capctllib[1]	IBMSNAP_RESTART	*FILE	*CHANGE
capctllib[1]	IBMSNAP_RESTARTX	*FILE	*CHANGE
capctllib[1]	IBMSNAP_AUTHTKN	*FILE	*CHANGE
capctllib[1]	IBMSNAP_AUTHTKNX	*FILE	*CHANGE

*表 49. GRTDPRAUT AUT(*CAPTURE) を指定して付与される権限 (続き)*

ライブラリー	オブジェクト	種類	許可
capctllib[1]	IBMSNAP_UOW	*FILE	*OBJOPR、 *OBJMGT、 *READ、 *UPD、 *DLT、 *ADD、 *EXECUTE
capctllib[1]	IBMSNAP_UOW_IDX	*FILE	*CHANGE
capctllib[1]	IBMSNAP_PRUNE_SET	*FILE	*CHANGE
capctllib[1]	IBMSNAP_PRUNE_SETX	*FILE	*CHANGE
capctllib[1]	IBMSNAP_CAPPARMS	*FILE	*READ、 *EXECUTE
capctllib[1]	IBMSNAP_SIGNAL	*FILE	*CHANGE
capctllib[1]	IBMSNAP_SIGNALX	*FILE	*CHANGE
capctllib[1]	IBMSNAP_CAPMON	*FILE	*CHANGE
capctllib[1]	IBMSNAP_CAPMONX	*FILE	*CHANGE
capctllib[1]	IBMSNAP_PRUNE_LOCK	*FILE	*CHANGE
ASN	ASN4B*	*SQLPKG	*USE
ASN	ASN4C*	*SQLPKG	*USE
ASN	QZS8CTLBLK	*USRSPC	*CHANGE

注:

1. ライブラリー列の *capctllib* 項目は、GRTDPRAUT コマンドの **CAPCTLLIB** パラメーターに渡される値を表します。このコマンドは、同時には 1 つのキャプチャー・コントロール・ライブラリーに対する権限しか更新できません。

以下の表では、GRTDPRAUT コマンドに対して AUT(*APPLY) パラメーターを指定すると付与される権限をリストします。

*表 50. GRTDPRAUT AUT(*APPLY) を指定して付与される権限*

ライブラリー	オブジェクト	種類	許可
QSYS	ASN	*LIB	*OBJOPR、 *READ、 *EXECUTE
QSYS	capctllib	*LIB	*OBJOPR、 *READ、 *EXECUTE
QDP4	QZSNAPV2	*PGM	*OBJOPR、 *READ、 *OBMGT、 *OBJALTER、 *EXECUTE
capctllib[1]	IBMSNAP_REGISTER	*FILE	*OBJOPR、 *READ、 *UPD、 *EXECUTE

*表 50. GRTDPRAUT AUT(*APPLY) を指定して付与される権限 (続き)*

ライブラリー	オブジェクト	種類	許可
capctllib[1]	IBMSNAP_REGISTERX	*FILE	*OBJOPR、 *READ、 *UPD、 *EXECUTE
capctllib[1]	IBMSNAP_REGISTERX1	*FILE	*OBJOPR、 *READ、 *UPD、 *EXECUTE
capctllib[1]	IBMSNAP_REGISTERX2	*FILE	*OBJOPR、 *READ、 *UPD、 *EXECUTE
capctllib[1]	IBMSNAP_REGISTER_EXT	*FILE	*OBJOPR、 *READ、 *UPD、 *EXECUTE
capctllib[1]	IBMSNAP_REGISTER_EXTX	*FILE	*OBJOPR、 *READ、 *UPD、 *EXECUTE
capctllib[1]	IBMSNAP_SIGNAL	*FILE	*OBJOPR、 *READ、 *UPD、 *ADD、 *EXECUTE
capctllib[1]	IBMSNAP_SIGNALX	*FILE	*OBJOPR、 *READ、 *UPD、 *ADD、 *EXECUTE
capctllib[1]	IBMSNAP_PRUNE_LOCK	*FILE	*CHANGE
capctllib[1]	IBMSNAP_UOW	*FILE	*OBJOPR、 *READ、 *UPD、 *ADD、 *EXECUTE
capctllib[1]	IBMSNAP_PRUNCNTL	*FILE	*OBJOPR、 *READ、 *UPD、 *ADD、 *EXECUTE
capctllib[1]	IBMSNAP_AUTHTKN	*FILE	*OBJOPR、 *READ、 *UPD、 *ADD、 *EXECUTE
capctllib[1]	IBMSNAP_AUTHTKNX	*FILE	*OBJOPR、 *READ、 *UPD、 *ADD、 *EXECUTE
ASN	IBMSNAP_SUBS_SET	*FILE	*OBJOPR、 *READ、 *UPD、 *EXECUTE
ASN	IBMSNAP_SUBS_SETX	*FILE	*OBJOPR、 *READ、 *UPD、 *EXECUTE

*表 50. GRTDPRAUT AUT(*APPLY) を指定して付与される権限 (続き)*

ライブラリー	オブジェクト	種類	許可
ASN	IBMSNAP_APPLYTRAIL	*FILE	*OBJOPR、 *READ、 *UPD、 *ADD、 *EXECUTE
ASN	IBMSNAP_APPLYTRACE	*FILE	*OBJOPR、 *READ、 *UPD、 *EXECUTE
ASN	IBMSNAP_APPLYTRACX	*FILE	*OBJOPR、 *READ、 *UPD、 *EXECUTE
ASN	IBMSNAP_SUBS_COLS	*FILE	*USE
ASN	IBMSNAP_SUBS_EVENT	*FILE	*USE
ASN	IBMSNAP_SUBS_STMTS	*FILE	*USE
ASN	IBMSNAP_SUBS_MEMBR	*FILE	*USE
ASN	ASN4A*	*SQLPKG	*USE
ASN	ASN4U*	*SQLPKG	*USE
ASN	IBMSNAP_APPLY_JOB	*FILE	*OBJOPR、 *READ、 *UPD、 *ADD、 *EXECUTE

注:

1. ライブラリー列の *capctllib* 項目は、GRTDPRAUT コマンドの **CAPCTLLIB** パラメーターに渡される値を表します。このコマンドは、同時には 1 つのキャプチャー・コントロール・ライブラリーに対する権限しか更新できません。

GRTDPRAUT の例

以下の例は、GRTDPRAUT コマンドの使用法を示しています。

例 1:

登録の定義および変更を行う権限を USER1 という名前のユーザーに許可するには、次のようにします。

```
GRTDPRAUT CAPCTLLIB(ASN) USER(USER1) AUT(*REGISTRAR)
```

例 2

サブスクリプション・セットの定義および変更を行う権限を USER1 という名前のユーザーに許可するには、次のようにします。

```
GRTDPRAUT CAPCTLLIB(ASN) USER(USER1) AUT(*SUBSCRIBER)
```

例 3

キャプチャー・プログラムを実行する権限を USER1 という名前のユーザーに許可するには、次のようにします。

```
GRTDPRAUT CAPCTLLIB(ASN) USER(USER1) AUT(*CAPTURE)
```

例 4

アプライ修飾子 A1 に関連付けられた既存のサブスクリプション・セットの定義および変更を行う権限を USER1 という名前のユーザーに許可するには、次のようにします。

```
GRTDPRAUT CAPCTLLIB(ASN) USER(USER1) AUT(*SUBSCRIBER) APYQUAL(A1)
```

例 5

ターゲット・サーバーとコントロール・サーバーが同じであるという条件で、アプライ修飾子 A1 に関連付けられたすべてのサブスクリプション・セットのコントロール・サーバー・システム上にあるアプライ・プログラムを実行するための権限をユーザーに与えるには、以下の操作を行ってください。

1. アプライ・プログラムが稼働するシステムで、以下のコマンドを実行します。

   ```
   GRTDPRAUT CAPCTLLIB(ASN) USER(USER1) AUT(*APPLY) APYQUAL(A1)
   ```

2. ソース・サーバー・システム上で適切な GRTDPRAUT コマンドを実行します。
   - アプライ・プログラムによって使われるソース・サーバーのアプリケーション・サーバー・ジョブが、ユーザー・プロファイル USER1 の下で実行される場合、ソース・サーバー・システムで以下のコマンドを実行します。

     ```
     GRTDPRAUT CAPCTLLIB(ASN) USER(USER1) AUT(*APPLY) APYQUAL(A1)
     ```

   - アプライ・プログラムによって使われるソース・サーバーのアプリケーション・サーバー・ジョブが、別のユーザー・プロファイル (例えば、QUSER) の下で実行される場合には、ソース・サーバー・システムで以下のコマンドを実行します。

     ```
     GRTDPRAUT CAPCTLLIB(ASN) USER(QUSER) AUT(*APPLY) APYQUAL(A1)
     ```

INZDPRCAP: DPR キャプチャーの再初期化 (System i)

DPR キャプチャー・プログラムの初期化 (INZDPRCAP) コマンドを使用して、ソース表の更新済みリストを使用するようにキャプチャー・プログラムに指示し、キャプチャー・プログラムを初期化します。

キャプチャー・プログラムのコントロール下にあるソース表は、キャプチャー・プログラムの実行中に変更することができます。INZDPRCAP コマンドを使用して、キャプチャー・プログラムが必ず最新のレプリケーション・ソースを処理するようにします。

このコマンドを実行する前に、キャプチャー・プログラムを実行しておく必要があります。

コマンド行にコマンド名を入力してから、F4 キーを押してコマンド構文を表示することができます。

このコマンド、およびこのコマンドのすべてのパラメーターの詳しい記述を表示するには、画面の一番上のコマンドにカーソルを移動し、F1 キーを押します。特定のパラメーターの記述を表示するには、そのパラメーター上にカーソルを移動し、F1 キーを押します。

構文

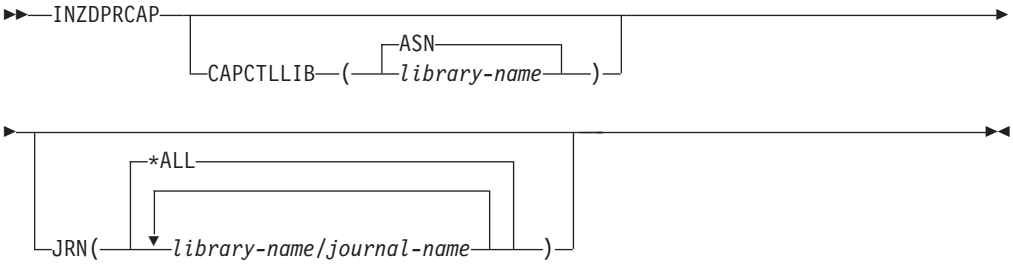


表 51 は、呼び出しパラメーターをリストしています。

表 51. INZDPRCAP コマンド・パラメーター定義 (System i 版)

パラメーター	定義およびプロンプト
CAPCTLLIB	キャプチャー・スキーマ (キャプチャー・コントロール表が存在するライブラリーの名前) を指定します。 **ASN (デフォルト)** キャプチャー・コントロール表は ASN ライブラリー内にあります。ASN ライブラリーはデフォルト・ライブラリーです。 *library-name* キャプチャー・コントロール表を含むライブラリーの名前。
JRN	最大 50 のジャーナルのサブセットをキャプチャー・プログラムが使用して作動するように指定します。キャプチャー・プログラムは、現在このジャーナルに記録されているすべてのソース表の処理を開始します。 ***ALL (デフォルト)** キャプチャー・プログラムは、すべてのジャーナルを使って動作します。 *library-name/journal-name* キャプチャー・プログラムが使用して作動するジャーナルの修飾名。

INZDPRCAP の例

以下の例は、INZDPRCAP コマンドの使用法を示しています。

例 1:

TRAINING という名前のライブラリーにある QSQJRN ジャーナルを使用して、キャプチャー・プログラムの初期化を行うには、次のようにします。

```
INZDPRCAP CAPCTLLIB(ASN) JRN(TRAINING/QSQJRN)
```

キャプチャー・コントロール表はデフォルト ASN スキーマ内にあります。

例 2

すべてのジャーナルを扱うキャプチャー・プログラムの初期化を行うには、次のようにします。

```
INZDPRCAP CAPCTLLIB(BSN) JRN(*ALL)
```

キャプチャー・コントロール表は BSN という名前のスキーマ内にあります。

OVRDPRCAPA: DPR キャプチャー属性のオーバーライド (System i)

実行中のキャプチャー・プログラムの動作を変更するには、 DPR キャプチャー属性のオーバーライド (OVRDPRCAPA) コマンドを使用します。

このコマンドは、キャプチャー・プログラムの始動時に IBMSNAP_CAPPARMS 表、または STRDPRCAP コマンドからキャプチャー・プログラムに渡された値をオーバーライドすることにより、プログラムの動作を変更します。

コマンド行にコマンド名を入力してから、F4 キーを押してコマンド構文を表示することができます。

このコマンド、およびこのコマンドのすべてのパラメーターの詳しい記述を表示するには、画面の一番上のコマンドにカーソルを移動し、F1 キーを押します。特定のパラメーターの記述を表示するには、そのパラメーター上にカーソルを移動し、F1 キーを押します。

構文

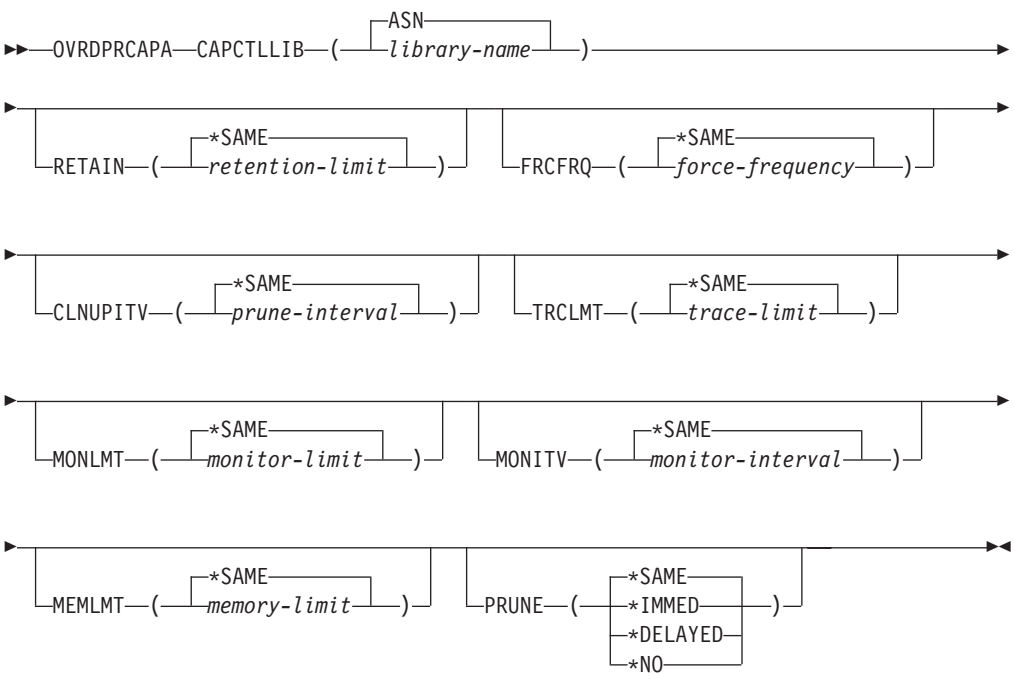


表 52 は、呼び出しパラメーターをリストしています。

表 52. OVRDPRCAPA コマンド・パラメーター定義 (System i 版)

パラメーター	定義およびプロンプト
CAPCTLLIB	キャプチャー・スキーマ (キャプチャー・コントロール表が存在するライブラリーの名前) を指定します。このライブラリーには、ソース表の登録情報を保管する IBMSNAP_REGISTER 表が入っています。このパラメーターは必須です。 **ASN(デフォルト)** キャプチャー・コントロール表は ASN ライブラリー内にあります。 *library-name* キャプチャー・コントロール表を含むライブラリーの名前。CRTDPRTBL コマンドに **CAPCTLLIB** パラメーターを指定して、このライブラリーを作成することができます。
RETAIN	データが除去されずに、変更データ (CD)、UOW、IBMSNAP_SIGNAL、および IBMSNAP_AUTHTKN 表に保存される分数を指定します。 この値は、DPR キャプチャー・プログラムの始動 (STRDPRCAP) コマンドの **CLNUPITV** パラメーターと共同で処理を行います。最初にキャプチャー・プログラムは、現在実行中のアプライ・プログラムの最も古いものよりも古い、CD、UOW、 IBMSNAP_SIGNAL、または IBMSNAP_AUTHTKN 行を削除します。その後、CD、UOW、IBMSNAP_SIGNAL、または IBMSNAP_AUTHTKN 表の新規の行、または残りの行は、経過時間が **RETAIN** パラメーターの値に達したときに削除されます。 表内のデータに矛盾が生じないように、データがこの **RETAIN** パラメーター値に達する前に変更情報がコピーされるようにアプライ・インターバルを設定してください。データの矛盾が発生した場合、アプライ・プログラムはフル・リフレッシュを実行します。 デフォルトは 10,080 分 (7 日) です。最大値は、35000000 分です。 ***SAME (デフォルト)** この値は変更されません。 *retention-limit* 新しい保持制限値。

表 52. OVRDPRCAPA コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
FRCFRQ	キャプチャー・プログラムが変更データ (CD) 表および作業単位 (UOW) 表に変更を書き込む頻度 (30 秒から 600 秒) を指定します。 キャプチャー・プログラムは、バッファーがフルになるか、**FRCFRQ** 時間制限が満了するか、いずれか先に発生した時点で、これらの変更をアプライ・プログラムから使用可能にします。このパラメーター値は、キャプチャー・プログラムが DPR キャプチャー・プログラムの初期化 (INZDPRCAP) コマンドからの変更に応答するためにかかる時間の長さに影響します。 このパラメーターは、ソース表の変更率が少ないサーバー上で、アプライ・プログラムが変更をより早期に使用できるようにするために使用してください。**FRCFRQ** パラメーター値はグローバル値であり、すべての登録済みソース表で使用されます。**FRCFRQ** 値を低い数値に設定すると、システム・パフォーマンスが影響を受ける可能性があります。 デフォルトは 30 秒です。 ***SAME (デフォルト)** この値は変更されません。 *force-frequency* キャプチャー・プログラムが、アプライ・プログラムで変更を使用できるようにする前に、 CD 表および UOW 表の変更をバッファー・スペース内に保持する秒数の新規の値。
CLNUPITV	キャプチャー・プログラムが変更データ (CD) 表、作業単位 (UOW) 表、IBMSNAP_SIGNAL 表、 IBMSNAP_CAPMON 表、IBMSNAP_CAPTRACE 表、および IBMSNAP_AUTHTKN 表から古いレコードを除去するまでの最大時間 (時間単位) を指定します。 このパラメーターは、**RETAIN** パラメーターと組み合わせて、CD、UOW、IBMSNAP_SIGNAL、および IBMSNAP_AUTHTKN 表の整理、**MONLMT** パラメーターと組み合わせて、IBMSNAP_CAPMON 表の整理、**TRCLMT** パラメーターと組み合わせて、IBMSNAP_CAPTRACE 表の整理をコントロールします。 (キャプチャー・プログラムの **RETAIN**、**MONLMT**、および **TRCLMT** パラメーターを設定するには、STRDPRCAP コマンドを使用します。) **CLNUPITV** パラメーターの値は、自動的に時間から秒に変換され、IBMSNAP_CAPPARMS 表の PRUNE_INTERVAL 列に保管されます。 ***SAME (デフォルト)** このキャプチャー属性値は変更されません。 *prune-interval* 時間数 (1 から 100) で指定された、整理インターバル。

表 52. *OVRDPRCAPA* コマンド・パラメーター定義 *(System i* 版*) (*続き*)*

パラメーター	定義およびプロンプト
TRCLMT	IBMSNAP_CAPTRACE 表の整理の頻度を示す、トレース限度を指定します。 ***SAME (デフォルト)** キャプチャー・プログラムは、現行のトレース限度値を継続して使用します。 *trace-limit* IBMSNAP_CAPTRACE 表の各整理操作の間隔の分数。
MONLMT	IBMSNAP_CAPMON 表の整理の頻度を示す、モニター限度を指定します。 ***SAME (デフォルト)** キャプチャー・プログラムは、現行のモニター限度値を継続して使用します。 *monitor-limit* IBMSNAP_CAPMON 表の各整理操作の間隔の分数。
MONITV	キャプチャー・プログラムが IBMSNAP_CAPMON 表に行を挿入する頻度を示すモニター・インターバル (秒単位) を指定します。 ***SAME (デフォルト)** キャプチャー・プログラムは、現行のモニター・インターバル値の使用を継続します。 *monitor-interval* IBMSNAP_CAPMON 表への行挿入の間隔の秒数。モニター・インターバルは、少なくとも 120 秒 (2 分) あける必要があります。120 よりも小さい数値がユーザーから入力された場合、このコマンドは自動的にパラメーター値を 120 に設定します。
MEMLMT	キャプチャー・ジャーナル・ジョブが使用できるメモリーの最大サイズ (MB 単位) を指定します。 ***SAME (デフォルト)** キャプチャー・プログラムは、現行のメモリー限度値を継続して使用します。 *memory-limit* メモリーの最大値の MB 数。

表 52. OVRDPRCAPA コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
PRUNE	キャプチャー・プログラムが変更データ (CD) 表、作業単位 (UOW) 表、IBMSNAP_SIGNAL 表、 IBMSNAP_CAPMON 表、IBMSNAP_CAPTRACE 表、および IBMSNAP_AUTHTKN 表から行を削除する方法を変更するには、このパラメーターを使用します。 ***SAME (デフォルト)** キャプチャー・プログラムは、STRDPRCAP コマンドの始動時にユーザーから指定された整理パラメーターを継続して使用します。 ***IMMED** キャプチャー・プログラムは、STRDPRCAP コマンドの始動時にユーザーから指定された **CLNUPITV** パラメーターの値に関係なく、表の整理を即時に開始します。 ***DELAYED** キャプチャー・プログラムは、指定された整理インターバルの最後に古い行を削除します。 PRUNE(*DELAYED) は、STRDPRCAP コマンドの **CLNUPITV** パラメーターの 2 番目の部分が *IMMED または *DELAYED に設定された場合は、整理の頻度に影響を与えることはありません。しかし PRUNE(*DELAYED) は、STRDPRCAP コマンドの始動時に **CLNUPITV** パラメーターの 2 番目の部分が *NO に設定された場合は、整理を開始します。 ***NO** キャプチャー・プログラムは整理を開始しません。この値は、STRDPRCAP コマンドの **CLNUPITV** パラメーターの設定値をオーバーライドします。

OVRDPRCAPA の例

以下の例は、OVRDPRCAPA コマンドの使用法を示しています。

例 1:

CD、UOW、IBMSNAP_SIGNAL、IBMSNAP_CAPMON、IBMSNAP_CAPTRACE、および IBMSNAP_AUTHTKN 表 (デフォルト ASN ライブラリー内にある) の整理パラメーターを変更し、実行中のキャプチャー・プログラムのキャプチャー・ジャーナル・ジョブの IBMSNAP_CAPMON モニター・インターバルおよびメモリー限度を変更するには、次のようにします。

```
OVRDPRCAPA CAPCTLLIB(ASN) CLNUPITV(12) MONITV(600) MEMLMT(64)
```

例 2

BSN ライブラリー内にある CD、 UOW、 IBMSNAP_SIGNAL、 IBMSNAP_CAPMON、IBMSNAP_CAPTRACE、および IBMSNAP_AUTHTKN 表の整理を開始するには、次のようにします。

```
OVRDPRCAPA CAPCTLLIB(BSN) PRUNE(*IMMED)
```

RMVDPRREG: DPR 登録の除去 (System i)

ソース表がレプリケーションの目的で使用されないようにするために、IBMSNAP_REGISTER 表から 1 つのソース表を除去するには、 DPR 登録の除去 (RMVDPRREG) コマンドを使用します。

コマンド行にコマンド名を入力してから、F4 キーを押してコマンド構文を表示することができます。

このコマンド、およびこのコマンドのすべてのパラメーターの詳しい記述を表示するには、画面の一番上のコマンドにカーソルを移動し、F1 キーを押します。特定のパラメーターの記述を表示するには、そのパラメーター上にカーソルを移動し、F1 キーを押します。

構文

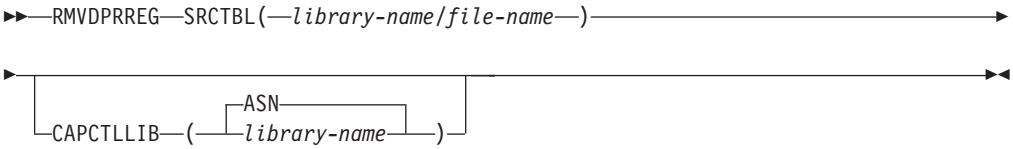


表 53 は、呼び出しパラメーターをリストしています。

表 53. RMVDPRREG コマンド・パラメーター定義 (System i 版)

パラメーター	定義およびプロンプト
SRCTBL	除去する登録を識別します。このパラメーターは必須です。 *library-name/file-name* 登録済みファイルの修飾名。
CAPCTLLIB	キャプチャー・スキーマ (キャプチャー・コントロール表が存在するライブラリーの名前) を指定します。 **ASN (デフォルト)** キャプチャー・コントロール表は ASN ライブラリー内にあります。 *library-name* キャプチャー・コントロール表を含むライブラリーの名前。

RMVDPRREG の例

以下の例は、RMVDPRREG コマンドの使用法を示しています。

例 1:

デフォルト ASN キャプチャー・スキーマ内の HR ライブラリーの、EMPLOYEE という名前のソース表の登録を除去するには、次のようにします。

```
RMVDPRREG SRCTBL(HR/EMPLOYEE)
```

例 2

BSN という名前のキャプチャー・スキーマ内の DEPT ライブラリーの、SALES という名前のソース表の登録を除去するには、次のようにします。

```
RMVDPRREG SRCTBL(DEPT/SALES) CAPCTLLIB(BSN)
```

RMVDPRSUB: DPR サブスクリプション・セットの除去 (System i)

サブスクリプション・セットを除去するには、 DPR サブスクリプション・セットの除去 (RMVDPRSUB) コマンドを使用します。**RMVMBRS** パラメーターが *YES に設定された場合、このコマンドはサブスクリプション・セットとそのすべてのメンバーを除去します。

コマンド行にコマンド名を入力してから、F4 キーを押してコマンド構文を表示することができます。

このコマンド、およびこのコマンドのすべてのパラメーターの詳しい記述を表示するには、画面の一番上のコマンドにカーソルを移動し、F1 キーを押します。特定のパラメーターの記述を表示するには、そのパラメーター上にカーソルを移動し、F1 キーを押します。

構文

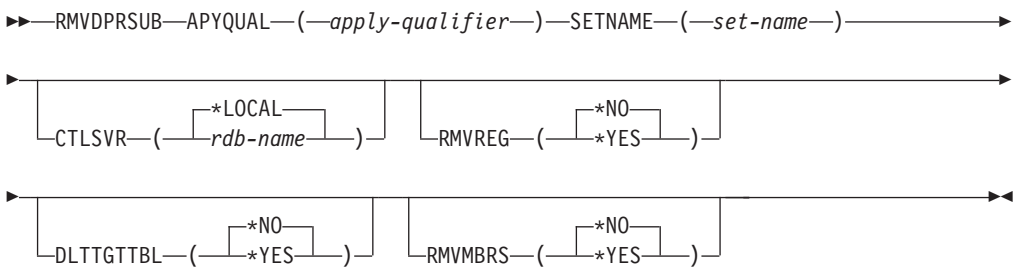


表 54 は、呼び出しパラメーターをリストしています。

表 54. RMVDPRSUB コマンド・パラメーター定義 (System i 版)

パラメーター	定義およびプロンプト
APYQUAL	アプライ・プログラムでサブスクリプション・セットの識別に使用されるアプライ修飾子を指定します。このパラメーターは必須です。 *apply-qualifier* アプライ修飾子の名前。
SETNAME	サブスクリプション・セットの名前を指定します。このパラメーターは必須です。 *set-name* サブスクリプション・セットの名前。指定されたアプライ修飾子に対して存在していないサブスクリプション・セット名が指定されると、エラー・メッセージが戻されます。

表 54. RMVDPRSUB コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
CTLSVR	アプライ・コントロール表を含むシステムのリレーショナル・データベース名を指定します。 ***LOCAL (デフォルト)** アプライ・コントロール表はローカルに存在します (RMVDPRSUB コマンドを実行するマシン上)。 *rdb-name* アプライ・コントロール表が置かれているリレーショナル・データベースの名前。 RDB ディレクトリー項目の作業 (WRKRDBDIRE) コマンドを使用して、この名前を検出することができます。
RMVREG	このコマンドが、サブスクリプション・セット内のすべてのサブスクリプション・セット・メンバーのターゲット表に関連付けられた登録を除去するかどうかを指定します。このパラメーターは、**RMVMBRS** パラメーターが *YES に設定されている場合にのみ使用してください。 ***NO (デフォルト)** 登録を除去しません。 ***YES** 登録を除去します。
DLTTGTTBL	このコマンドが、サブスクリプション・セットのドロップ後に、サブスクリプション・セット・メンバーのターゲット表をドロップするかどうかを指定します。このパラメーターは、**RMVMBRS** パラメーターが *YES に設定されている場合にのみ使用してください。 ***NO (デフォルト)** ターゲット表はドロップされません。 ***YES** ターゲット表はドロップされます。
RMVMBRS	このコマンドが、サブスクリプション・セットと、そのサブスクリプション・セット内のすべてのメンバーを除去するかどうかを指定します。 ***NO (デフォルト)** サブスクリプション・セット内に既存のメンバーがある場合には、サブスクリプション・セットは除去されません。 ***YES** サブスクリプション・セットと、そのサブスクリプション・セット・メンバーがすべて除去されます。

RMVDPRSUB の例

次の例は、**RMVDPRSUB** コマンドの使用方法を示しています。

例 1:

サブスクリプション・セット・メンバーを含まない、SETHR という名前のサブスクリプション・セットを除去するには、次のようにします。

```
RMVDPRSUB APYQUAL(AQHR) SETNAME(SETHR)
```

例 2

SETHR という名前のサブスクリプション・セットと、そのサブスクリプション・セット・メンバーをすべて除去するには、次のようにします。

```
RMVDPRSUB APYQUAL(AQHR) SETNAME(SETHR) RMVMBRS(*YES)
```

例 3

SETHR という名前のサブスクリプション・セットと、そのサブスクリプション・セット・メンバーのすべて、および関連する登録を除去するには、次のようにします。

```
RMVDPRSUB APYQUAL(AQHR) SETNAME(SETHR) RMVREG(*YES) RMVMBRS(*YES)
```

RMVDPRSUBM: DPR サブスクリプション・セット・メンバーの除去 (System i)

サブスクリプション・セットから 1 つのサブスクリプション・セット・メンバーを除去するには、 DPR サブスクリプション・セット・メンバーの除去 (RMVDPRSUBM) コマンドを使用します。

コマンド行にコマンド名を入力してから、F4 キーを押してコマンド構文を表示することができます。

このコマンド、およびこのコマンドのすべてのパラメーターの詳しい記述を表示するには、画面の一番上のコマンドにカーソルを移動し、F1 キーを押します。特定のパラメーターの記述を表示するには、そのパラメーター上にカーソルを移動し、F1 キーを押します。

構文

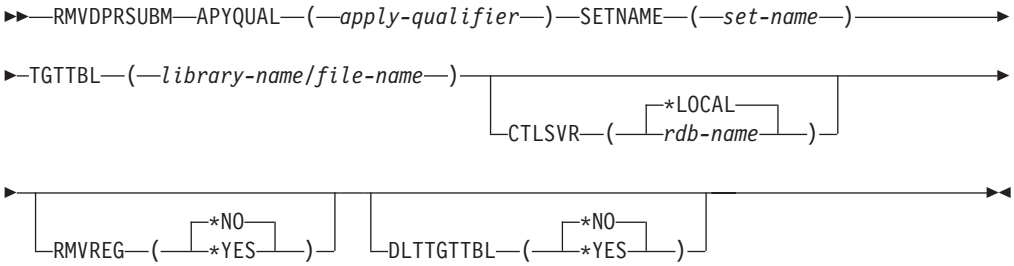


表 55 は、呼び出しパラメーターをリストしています。

表 55. RMVDPRSUBM コマンド・パラメーター定義 (System i 版)

パラメーター	定義およびプロンプト
APYQUAL	アプライ・プログラムでサブスクリプション・セットの識別に使用されるアプライ修飾子を指定します。このパラメーターは必須です。 *apply-qualifier* アプライ修飾子の名前。

表 55. RMVDPRSUBM コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
SETNAME	サブスクリプション・セットの名前を指定します。このパラメーターは必須です。 *set-name* サブスクリプション・セットの名前。指定されたアプライ修飾子に対して存在していないサブスクリプション・セット名が指定されると、エラー・メッセージが戻されます。
TGTTBL	サブスクリプション・セット・メンバーに関して登録されているターゲット表を指定します。このパラメーターは必須です。 *library-name/file-name* ターゲット表の修飾名。
CTLSVR	アプライ・コントロール表を含むシステムのリレーショナル・データベース名を指定します。 ***LOCAL (デフォルト)** アプライ・コントロール表はローカルに存在します (RMVDPRSUBM コマンドを実行するマシン上)。 *rdb-name* アプライ・コントロール表が置かれているリレーショナル・データベースの名前。 RDB ディレクトリー項目の作業 (WRKRDBDIRE) コマンドを使用して、この名前を検出することができます。
RMVREG	このコマンドが、サブスクリプション・セット・メンバーのターゲット表に関連付けられた登録を除去するかどうかを指定します。 ***NO (デフォルト)** 登録を除去しません。 ***YES** 登録を除去します。
DLTTGTTBL	このコマンドが、サブスクリプション・セット・メンバーのドロップ後に、サブスクリプション・セット・メンバーのターゲット表をドロップするかどうかを指定します。 ***NO (デフォルト)** ターゲット表はドロップされません。 ***YES** ターゲット表はドロップされます。

RMVDPRSUBM の例

以下の例は、RMVDPRSUBM コマンドの使用法を示しています。

例 1:

EMP という名前のターゲット表を使用するサブスクリプション・セット・メンバーを、 RMTRDB1 という名前のリレーショナル・データベース上の SETEMP サブスクリプション・セットから除去するには、次のようにします。

```
RMVDPRSUBM APYQUAL(AQHR) SETNAME(SETEMP) TGTTBL(TGTEMP/EMP) CTLSVR(RMTRDB1)
```

例 2

SETHR サブスクリプション・セットからサブスクリプション・セット・メンバーをドロップし、登録をドロップしてから表をドロップするには、次のようにします。

```
RMVDPRSUBM APYQUAL(AQHR) SETNAME(SETHR) TGTTBL(TGTHR/YTDTAX) RMVREG(*YES)
  DLTTGTTBL(*YES)
```

RVKDPRAUT: 権限の取り消し (System i)

DPR 権限の取り消し (RVKDPRAUT) コマンドは、ユーザーがレプリケーション・ソースおよびサブスクリプション・セットの定義または変更を行うことができないようにするため、レプリケーション・コントロール表に対する権限を取り消します。

コマンド行にコマンド名を入力してから、F4 キーを押してコマンド構文を表示することができます。

このコマンド、およびこのコマンドのすべてのパラメーターの詳しい記述を表示するには、画面の一番上のコマンドにカーソルを移動し、F1 キーを押します。特定のパラメーターの記述を表示するには、そのパラメーター上にカーソルを移動し、F1 キーを押します。

構文

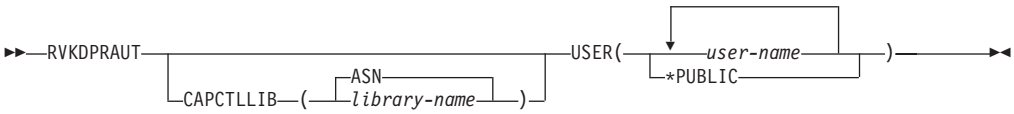


表 56 は、呼び出しパラメーターをリストしています。

表 56. RVKDPRAUT コマンド・パラメーター定義 (System i 版)

パラメーター	定義およびプロンプト
CAPCTLLIB	ユーザー権限が取り消されるライブラリーの名前である、キャプチャー・スキーマを指定します。 **ASN (デフォルト)** キャプチャー・コントロール表は ASN ライブラリー内にあります。 *library-name* レプリケーション・コントロール表を含むライブラリーの名前。
USER	権限が取り消されるユーザーを指定します。このパラメーターは必須です。 *user-name* 権限が取り消されるユーザーの名前を最大 50 まで指定します。 ***PUBLIC** 特定の許可を所有していないすべてのユーザー、許可リストにないすべてのユーザー、およびグループ・プロファイルが許可を所有していないすべてのユーザーから、許可を取り消すように指定します。

使用上の注意

以下のいずれかの状態が生じると、このコマンドはエラー・メッセージを戻します。

- 指定したユーザーが存在しない。
- コマンドを実行しているユーザーが、指定したユーザー・プロファイルへの権限を所有していない。
- このコマンドを実行しているユーザーには、DB2 DataPropagator for System i のコントロール表に対する権限を取り消す許可がない。
- DB2 DataPropagator for System i のコントロール表が存在しない。
- キャプチャー・プログラムまたはアプライ・プログラムが実行中。

RVKDPRAUT の例

以下の例は、RVKDPRAUT コマンドの使用法を示しています。

例 1:

ASN ライブラリー下のコントロール表に対する権限を、HJONES という名前のユーザーから取り消すには、次のようにします。

```
RVKDPRAUT CAPCTLLIB(ASN) USER(HJONES)
```

例 2

GRTDPRAUT コマンドに指定されていないすべてのユーザーの権限を取り消し、ASN ライブラリー内のコントロール表にアクセスできないようにするには、以下のようにします。

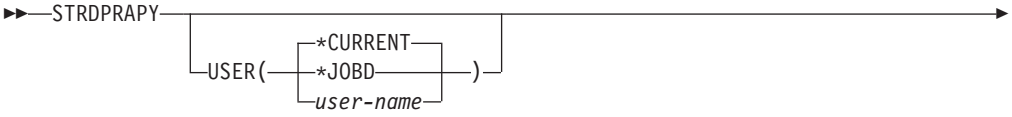
```
RVKDPRAUT CAPCTLLIB(ASN) USER(*PUBLIC)
```

STRDPRAPY: アプライの始動 (System i)

DPR アプライ・プログラムの始動 (STRDPRAPY) コマンドは、ローカル・システム上でアプライ・プログラムを始動するために使用されます。アプライ・プログラムの実行は、ユーザーがプログラムを停止するまで、またはリカバリー不能エラーが検出されるまで続けられます。

コマンド行にコマンド名を入力してから、F4 キーを押してコマンド構文を表示することができます。

このコマンド、およびこのコマンドのすべてのパラメーターの詳しい記述を表示するには、画面の一番上のコマンドにカーソルを移動し、F1 キーを押します。特定のパラメーターの記述を表示するには、そのパラメーター上にカーソルを移動し、F1 キーを押します。



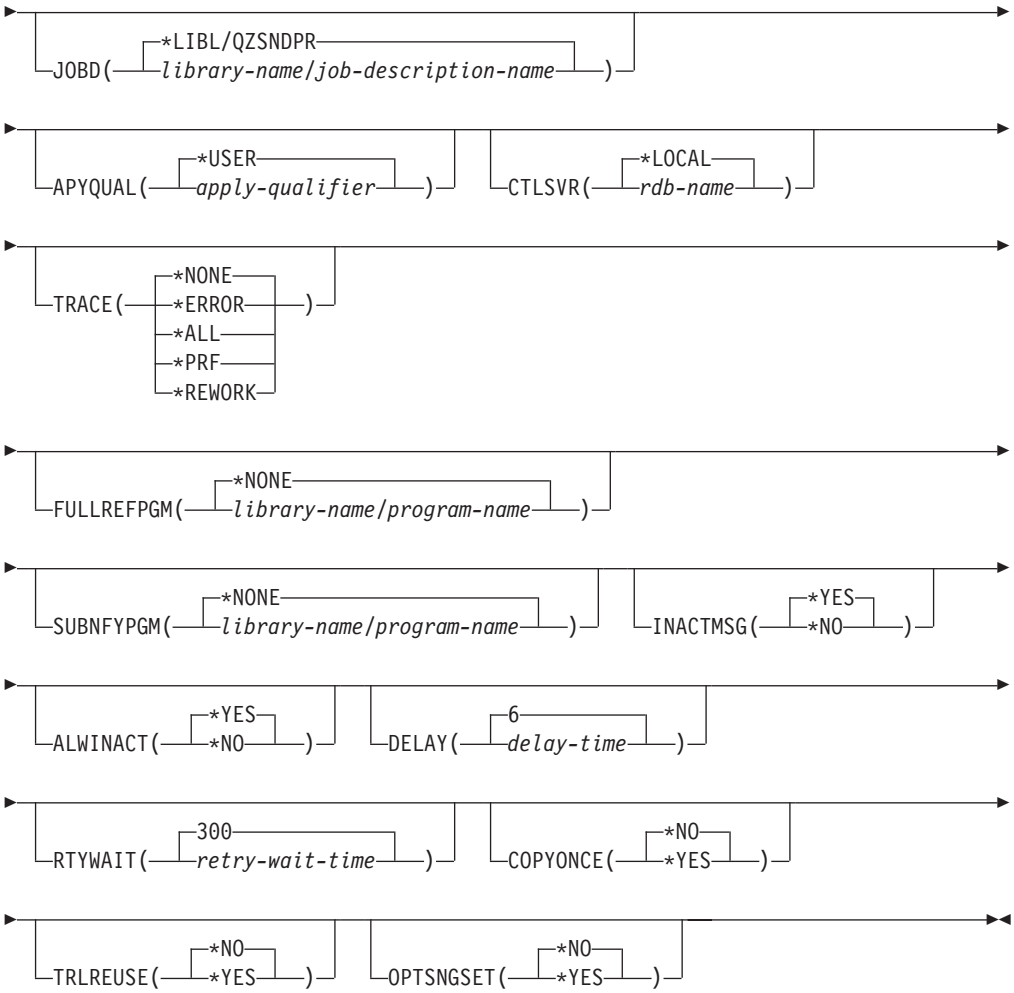

JOBD(*LIBL/QZSNDPR library-name/job-description-name)
APYQUAL(*USER apply-qualifier)
CTLSVR(*LOCAL rdb-name)
TRACE(*NONE *ERROR *ALL *PRF *REWORK)
FULLREFPGM(*NONE library-name/program-name)
SUBNFYPGM(*NONE library-name/program-name)
INACTMSG(*YES *NO)
ALWINACT(*YES *NO)
DELAY(6 delay-time)
RTYWAIT(300 retry-wait-time)
COPYONCE(*NO *YES)
TRLREUSE(*NO *YES)
OPTSNGSET(*NO *YES)

表 57 は、呼び出しパラメーターをリストしています。

表 57. STRDPRAPY コマンド・パラメーター定義 (System i 版)

パラメーター	定義およびプロンプト
USER	アプライ・プログラムが開始するユーザー ID の名前を指定します。このコマンドを実行するには、指定されたユーザー・プロファイルに対する許可が (*USE 権限を持つ) 必要です。アプライ・プログラムは、この指定されたユーザー・プロファイルの下で実行されます。 コントロール表は、**CTLSVR** パラメーターで指定されたリレーショナル・データベース上にあります。同じコントロール表は、**USER** パラメーターに指定した値を無視して使われます。 ***CURRENT (デフォルト)** 現行ジョブに関連付けられたユーザー ID は、このアプライ・プログラムと関連付けられたユーザー ID と同じです。 ***JOBD** このアプライ・プログラムに関連付けられたジョブ記述の中で指定されたユーザー ID。ジョブ記述は USER(*RQD) を指定することはできません。 *user-name* このアプライ・プログラムに関連付けられたユーザー ID。以下の IBM 提供のオブジェクトは、このパラメーターに対しては有効ではありません。 QDBSHR、 QDFTOWN、 QDOC、 QLPAUTO、 QLPINSTALL、 QRJE、 QSECOFR、 QSPL、 QSYS、または QTSTRQS。 STRDPRAPY コマンドでプロンプトを出すと、 F4 キーを押して、サブスクリプション・セットを定義したユーザーのリストを表示することができます。
JOBD	アプライ・プログラムをサブミットするときに使用するジョブ記述の名前を指定します。 ***LIBL/QZSNDPR (デフォルト)** DB2 DataPropagator for System i から提供されるデフォルトのジョブ記述。 *library-name/job-description-name* アプライ・プログラムで使用するジョブ記述の名前。

表 57. *STRDPRAPY* コマンド・パラメーター定義 *(System i 版) (続き)*

パラメーター	定義およびプロンプト
APYQUAL	アプライ・プログラムによって使われるアプライ修飾子を指定します。このアプライ修飾子によりグループ化されるすべてのサブスクリプション・セットが、アプライ・プログラムで実行されます。 ***USER (デフォルト)** ユーザーが入力する **USER** パラメーターの値が、アプライ修飾子の名前として使用されます。 *apply-qualifier* このアプライ・プログラムによって実行されるサブスクリプション・セットをグループ化するために使用される名前。アプライ修飾子名として最大 18 文字を指定できます。この名前は、リレーショナル・データベース名と同じ命名規則に準拠します。 STRDPRAPY コマンドでプロンプトを出すと、F4 キーを押して、既存のサブスクリプション・セットを使用したアプライ修飾子のリストを表示することができます。
CTLSVR	アプライ・コントロール表を含むシステムのリレーショナル・データベース名を指定します。 ***LOCAL (デフォルト)** アプライ・コントロール表はローカルに存在します (STRDPRAPY コマンドを実行するマシン上)。 *rdb-name* アプライ・コントロール表が置かれているリレーショナル・データベースの名前。 RDB ディレクトリー項目の作業 (WRKRDBDIRE) コマンドを使用して、この名前を検出することができます。 STRDPRAPY コマンドでプロンプトを出すと、F4 キーを押して、使用可能な RDB 名を表示することができます。
TRACE	アプライ・プログラムがトレースを生成するかどうかを指定します。アプライ・プログラムは、QPZSNATRC と呼ばれるスプール・ファイルにトレース・データを書き込みます。 ***NONE (デフォルト)** トレースが生成されません。 ***ERROR** トレースはエラー情報だけを保持します。 ***ALL** トレースはエラーおよび実行フロー情報を保持します。 ***PRF** トレースは、アプライ・プログラムの実行のさまざまな段階でパフォーマンスの分析に使用できる情報を保持します。 ***REWORK** トレースは、アプライ・プログラムによって再処理された行についての情報を保持します。

表 57. STRDPRAPY コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
FULLREFPGM	アプライ・プログラムが、出口ルーチンを呼び出して、ターゲット表を初期化するかどうかを指定します。 ターゲット表のフル・リフレッシュの実行準備が整うと、アプライ・プログラムはフル・リフレッシュを実行せずに、指定された出口ルーチンを呼び出します。 フル・リフレッシュ出口ルーチンがアプライ・プログラムで使用される場合、 IBMSNAP_APPLYTRAIL 表にある ASNLOAD 列の値は Y です。 ***NONE (デフォルト)** フル・リフレッシュ出口ルーチンが使用されません。 *library-name/program-name* ターゲット表のフル・リフレッシュを実行するアプライ・プログラムから呼び出されるプログラムの修飾名。例えば、ライブラリー DATAPROP にあるプログラム ASNLOAD を呼び出すための修飾名は DATAPROP/ASNLOAD です。
SUBNFYPGM	サブスクリプション・セットの処理が終了したときに、アプライ・プログラムから出口ルーチンを呼び出すかどうかを指定します。出口ルーチンへの入力には、サブスクリプション・セット名、アプライ修飾子、完了状況、およびリジェクト回数を含む統計が含まれます。 通知プログラムを使用すると、作業単位 (UOW) 表を検査して、リジェクトされたトランザクションを判別したり、メッセージの発行やイベントの生成などのアクションをさらに続けるかを判別できます。 ***NONE (デフォルト)** 出口ルーチンが使用されません。 *library-name/program-name* サブスクリプション・セットの処理時に、アプライ・プログラムから呼び出される出口ルーチン・プログラムの修飾名。例えば、ライブラリー DATAPROP にあるプログラム APPLYDONE を呼び出すための修飾名は DATAPROP/APPLYDONE です。
INACTMSG	アプライ・プログラムが作業を完了し、一定の期間非アクティブになったときに、メッセージを生成するかどうかを指定します。 ***YES (デフォルト)** アプライ・プログラムがアクティブではなくなる前に、メッセージ ASN1044 を生成します。メッセージ ASN1044 は、アプライ・プログラムが非アクティブであった時間の長さを示します。 ***NO** メッセージが生成されません。

表 57. STRDPRAPY コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
ALWINACT	アプライ・プログラムを非アクティブ (スリープ) 状態で実行できるかどうかを指定します。 ***YES (デフォルト)** 処理するものが何もない場合、アプライ・プログラムはスリープします。 ***NO** アプライ・プログラムで処理するものが何もない場合、アプライ・プログラムをサブミットして開始したジョブは終了します。
DELAY	連続レプリケーションが使用される場合に、それぞれのアプライ・プログラムのサイクルの終わりにおける遅延時間 (秒単位) を指定します。 **6 (デフォルト)** 遅延時間は 6 秒です。 *delay-time* 0 から 6 までの数値で入力された遅延時間。
RTYWAIT	エラーが生じた場合に、失敗した操作を再試行するまでアプライ・プログラムが待機する時間 (秒単位) を指定します。 **300 (デフォルト)** 再試行待ち時間は 300 秒 (5 分) です。 *retry-wait-time* 0 から 35000000 の数値で入力された待ち時間。この時間が過ぎると、アプライ・プログラムは失敗した操作を再試行します。

表 57. STRDPRAPY コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
COPYONCE	アプライ・プログラムが呼びだされた時点で適格と見なされたサブスクリプション・セットごとに、アプライ・プログラムがコピー・サイクルを 1 回実行するかどうかを指定します。その後、アプライ・プログラムは終了します。適格と見なされるサブスクリプション・セットとは、以下の基準を満たすものです。 • IBMSNAP_SUBS_SET 表で (ACTIVATE > 0)。ACTIVATE 列の値がゼロより大きい場合、そのサブスクリプション・セットは無期限にアクティブであるか、または 1 回のみのサブスクリプション処理に使用されています。 • (REFRESH_TYPE = R または B) または (REFRESH_TYPE = E であり、指定されたイベントが発生)。 REFRESH_TYPE 列の値は IBMSNAP_SUBS_SET 表に保管されます。 IBMSNAP_SUBS_SET 表で MAX_SYNCH_MINUTES 限度と、IBMSNAP_SUBS_EVENT 表で END_OF_PERIOD タイム・スタンプが指定されている場合、これらの設定が使用されます。 ***NO(デフォルト)** アプライ・プログラムは、適格なサブスクリプション・セットごとにコピー・サイクルを 1 回実行しません。 ***YES** アプライ・プログラムは、適格なサブスクリプション・セットごとにコピー・サイクルを 1 回実行した後、終了します。
TRLREUSE	アプライ・プログラムの始動時に、アプライ・プログラムが IBMSNAP_APPLYTRAIL 表を空にするかどうかを指定します。 ***NO(デフォルト)** アプライ・プログラムはプログラム始動時に IBMSNAP_APPLYTRAIL 表を空にしません。 ***YES** アプライ・プログラムはプログラム始動時に IBMSNAP_APPLYTRAIL 表を空にします。

表 57. *STRDPRAPY* コマンド・パラメーター定義 *(System i 版) (続き)*

パラメーター	定義およびプロンプト
OPTSNGSET	アプライ・プログラムのパフォーマンスを、処理するサブスクリプション・セットが 1 つだけである状況に対して最適化するかどうかを指定します。このパラメーターは、レプリカ・ターゲット表には影響しません。 このパラメーターが *YES に設定された場合、アプライ・プログラムはサブスクリプション・セットのメンバーおよび列を一度しかフェッチせず、複数の連続する処理サイクルで同じサブスクリプション・セットを処理する場合は、このフェッチした情報を再利用します。 ***NO(デフォルト)** 処理するサブスクリプション・セットが 1 つだけである場合、アプライ・プログラムのパフォーマンスは最適化されません。 ***YES** 処理するサブスクリプション・セットが 1 つだけである場合、アプライ・プログラムのパフォーマンスは最適化されます。アプライ・プログラムはその後の処理サイクルでサブスクリプション・セット情報を再利用するため、CPU リソースの所要数量が減少し、スループット率が向上します。

使用上の注意

システムの QSTRUPPGM 値で参照されるコマンドを追加することにより、サブシステムを自動的に開始するようシステムを設定することができます。 QDP4/QZSNDPR サブシステムを使用する場合、それは STRDPRAPY コマンド処理の一部として開始されます。

CTLSVR パラメーターによって指定されたリレーショナル・データベース (RDB) が DB2 for i5/OS データベースである場合、サーバーの表は ASN ライブラリーにあります。RDB が DB2 fori5/OS データベースでない場合、ASN を修飾子として使って、表にアクセスすることができます。

アプライ・プログラムを始動するときのエラー条件

以下のいずれかの状態が生じると、STRDPRAPY コマンドはエラー・メッセージを戻します。

- ユーザーが存在していない場合。
- コマンドを実行しているユーザーが、そのコマンドまたはジョブ記述で指定した、ユーザー・プロファイルへの権限を所有していない場合。
- アプライ・プログラムのインスタンスが、このアプライ修飾子とコントロール・サーバーの組み合わせのローカル・システムですでにアクティブの場合。
- **CTLSVR** パラメーターによって指定された RDB 名がリレーショナル・データベース・ディレクトリーの中にない場合。
- **CTLSVR** パラメーターによって指定された RDB 上にコントロール表が存在しない場合。
- **APYQUAL** パラメーターによって指定されたアプライ修飾子に対して定義されたサブスクリプション・セットがない場合。

すべての IBMSNAP_SUBS_SET 表の中のユニークなアプライ修飾子ごとに、 1 つのアプライ・プログラムを始動する必要があります。STRDPRAPY コマンドを発行するたびに異なるアプライ修飾子を指定する

ことにより、複数のアプライ・プログラムを始動できます。これらのアプライ・プログラムは、同じユーザー・プロファイルの下で実行します。

アプライ・プログラムのジョブの識別

各アプライ・プログラムは、アプライ修飾子とコントロール・サーバー名の両方を使って識別されます。実行時には、アプライ・プログラムに応じて始まるジョブに十分な外部属性がないため、特定のアプライ修飾子とコントロール・サーバーとの組み合わせに関連付けられているアプライ・プログラムを正確に識別することはできません。そのため、以下に示す方法でジョブが識別されます。

- **USER** パラメーターと関連付けられたユーザー・プロファイルの下で、ジョブが開始します。
- アプライ修飾子の最初の 10 文字は切り捨てられて、ジョブ名になります。
- DB2 DataPropagator for System i は、ローカル・システム上の ASN ライブラリー内に名前のあるアプライ・ジョブ (IBMSNAP_APPLY_JOB) 表を保持しています。この表が、アプライ修飾子およびコントロール・サーバーの値を、正しいアプライ・プログラムのジョブにマップします。
- ジョブ・ログを表示することができます。アプライ修飾子とコントロール・サーバー名が、アプライ・プログラムに対する呼び出しで使用されています。

通常、以下に示す 2 つの条件が当てはまれば、QZSNDPR サブシステムで実行中のジョブのリストを表示することにより、正しいアプライ・プログラムを識別することができます。

- アプライ修飾子の最初の 10 文字がユニークである。
- アプライ・プログラムが、ローカル・コントロール・サーバーでのみ開始される。

STRDPRAPY の例

以下の例は、STRDPRAPY コマンドの使用法を示しています。

例 1:

AQHR アプライ修飾子、およびローカルに常駐しているコントロール表を使用して、エラーおよび実行フローの情報を含むトレース・ファイルを生成するアプライ・プログラムを始動するには、次のようにします。

```
STRDPRAPY APYQUAL(AQHR) CTLSVR(*LOCAL) TRACE(*ALL)
```

例 2

ローカルに常駐するアプライ・コントロール表を使用してアプライ・プログラムを始動し、アプライ・プログラムで処理するものが何もない場合は、このアプライ・プログラムを始動したジョブが自動的に終了するように指定するには、次のようにします。

```
STRDPRAPY APYQUAL(AQHR) CTLSVR(*LOCAL) ALWINACT(*NO)
```

例 3

プログラム始動時に IBMSNAP_APPLYTRAIL 表を空にするアプライ・プログラムを始動するには、次のようにします。

```
STRDPRAPY APYQUAL(AQHR) CTLSVR(*LOCAL) TRLREUSE(*YES)
```

例 4

すべてのデフォルト値を使用してアプライ・プログラムを始動するには、次のようにします。

```
STRDPRAPY
```

STRDPRCAP: キャプチャーの始動 (System i)

System i サーバー上の System i データベース表への変更のキャプチャーを開始するには、 DPR キャプチャーの始動 (STRDPRCAP) コマンドを使用します。

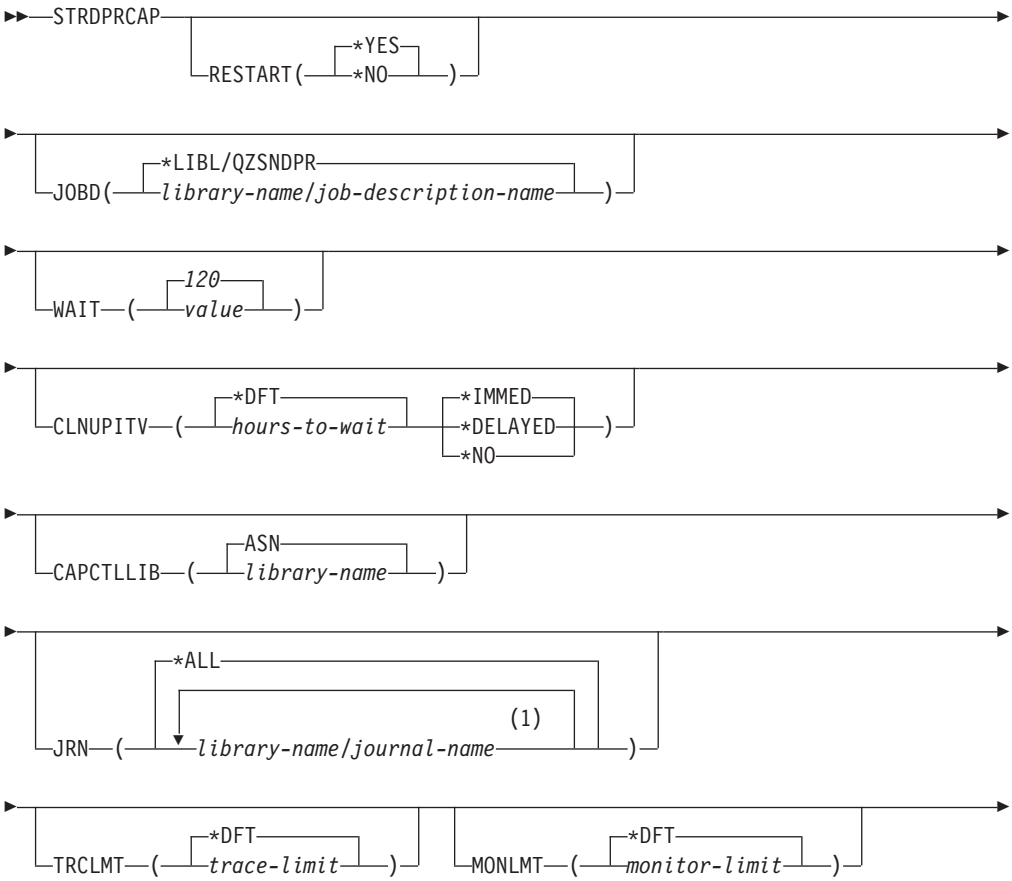
このコマンドは IBMSNAP_REGISTER 表内のすべてのレプリケーション・ソースを処理するため、適切な権限に基づいてこのコマンドを実行していることを確認してください。

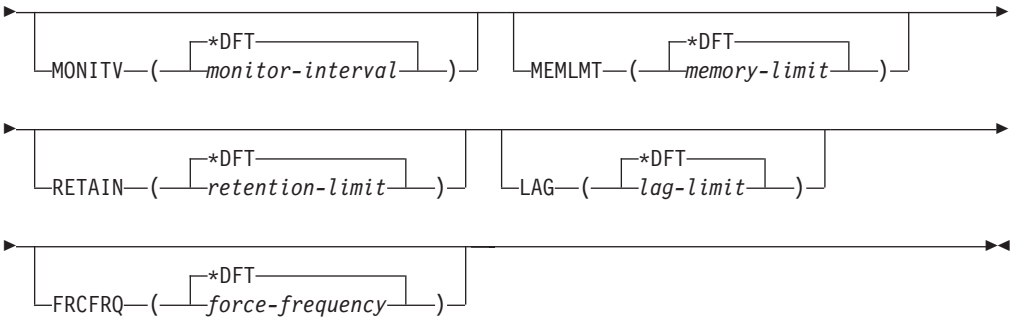
キャプチャー・プログラムを始動すると、停止されるかまたはリカバリー不能エラーが検出されるまで実行を続けます。

コマンド行にコマンド名を入力してから、F4 キーを押してコマンド構文を表示することができます。

このコマンド、およびこのコマンドのすべてのパラメーターの詳しい記述を表示するには、画面の一番上のコマンドにカーソルを移動し、F1 キーを押します。特定のパラメーターの記述を表示するには、そのパラメーター上にカーソルを移動し、F1 キーを押します。

構文





注:

1 最大 50 のジャーナルを指定できます。

表 58 は、呼び出しパラメーターをリストしています。

表 58. STRDPRCAP コマンド・パラメーター定義 (System i 版)

パラメーター	定義およびプロンプト
RESTART	キャプチャー・プログラムがウォーム・スタートとコールド・スタートを処理する方法を指定します。 ***YES (デフォルト)** キャプチャー・プログラムは、以前に終了したときに停止したポイントから変更の処理を継続します。これは、ウォーム・スタートとしても知られており、操作の通常モードです。 ***NO** キャプチャー・プログラムは、変更データ (CD) 表からすべての情報を消去します。JRN(*ALL) を指定すると、キャプチャー・プログラムは作業単位 (UOW) 表からもすべての情報を消去します。 変更の取り込みが再開される前に、影響を受けたソース表のすべてのサブスクリプションはフル・リフレッシュされます。この処理は、コールド・スタート としても知られています。 RESTART(*NO) および JRN(*library-name/journal-name*) を指定すると、指定したジャーナルのキャプチャー・プログラムのコールド・スタートを行うことができます。
JOBD	キャプチャー・プログラムをサブミットするときに使用するジョブ記述の名前を指定します。 ***LIBL/QZSNDPR (デフォルト)** DB2 DataPropagator for System i に用意されているデフォルトのジョブ記述を指定します。 *library-name/job-description-name* キャプチャー・プログラムで使用するジョブ記述の名前。

表 58. STRDPRCAP コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
WAIT	キャプチャー・プログラムが状況をチェックするまで待つ最大時間を秒単位 (60 から 6,000) で指定します。この値を使用して、キャプチャー・プログラムの応答時間を調整することができます。 この値を低くすると、キャプチャー・プログラムが終了または初期化するまでにかかる時間は減りますが、システム・パフォーマンスによくない影響を及ぼす場合があります。この値を高くすると、キャプチャー・プログラムが終了または初期化するまでにかかる時間は増えますが、システム・パフォーマンスを向上させることができます。この値をあまり高くしすぎると、キャプチャー・プログラムは定期的に処理を行っていても、応答時間が悪くなることがあります。応答時間がどの程度下がるかは、ソース表に加える変更の量やシステムで発生する他の作業の量によって異なります。 **120 (デフォルト)** キャプチャー・プログラムは 120 秒待機します。 *value* キャプチャー・プログラムが待機する最大時間数 (秒)。

表 58. STRDPRCAP コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
CLNUPITV	キャプチャー・プログラムが変更データ (CD) 表、作業単位 (UOW) 表、IBMSNAP_SIGNAL 表、 IBMSNAP_CAPMON 表、IBMSNAP_CAPTRACE 表、および IBMSNAP_AUTHTKN 表から古いレコードを除去するまでの最大時間 (時間単位) を指定します。 このパラメーターは、**RETAIN** パラメーターと組み合わせて、CD、UOW、IBMSNAP_SIGNAL、および IBMSNAP_AUTHTKN 表の整理、**MONLMT** パラメーターと組み合わせて、IBMSNAP_CAPMON 表の整理、**TRCLMT** パラメーターと組み合わせて、IBMSNAP_CAPTRACE 表の整理をコントロールします。 (キャプチャー・プログラムの **RETAIN**、**MONLMT**、および **TRCLMT** パラメーターを設定するには、**STRDPRCAP** コマンドを使用します。) これらのパラメーターの設定値を変更するには、CHGDPRCAPA または OVRDPRCAPA コマンドを使用します。) **CLNUPITV** パラメーターには 2 つの部分があります。 ***DFT (デフォルト)** キャプチャー・プログラムは、IBMSNAP_CAPPARMS 表の PRUNE_INTERVAL 列の値を使用します。 *hours-to-wait* 時間数 (1 から 100) で指定された、整理インターバル。 ***IMMED (デフォルト)** キャプチャー・プログラムは、指定されたインターバルの開始時に (つまり即時に)、そしてその後は各インターバルで、古いレコードを除去します。 ***DELAYED** キャプチャー・プログラムは、指定されたインターバルの終了時に、そしてその後は各インターバルで、古いレコードを除去します。 ***NO** キャプチャー・プログラムはレコードを除去しません。
CAPCTLLIB	キャプチャー・スキーマ (キャプチャー・コントロール表が存在するライブラリーの名前) を指定します。 **ASN (デフォルト)** キャプチャー・コントロール表が常駐するデフォルト・ライブラリーです。 *library-name* キャプチャー・コントロール表が常駐するライブラリーの名前。

表 58. STRDPRCAP コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
JRN	最大 50 のジャーナルのサブセットをキャプチャー・プログラムが使用して作動するように指定します。キャプチャー・プログラムは、現在このジャーナルに記録されているすべてのソース表の処理を開始します。 ***ALL (デフォルト)** キャプチャー・プログラムは、ソース表が記録されているジャーナルをすべて使用して作業を開始します。 *library-name/journal-name* キャプチャー・プログラムが使用して作動するジャーナルの修飾名。複数のジャーナルを入力するときには、スペースを使用してジャーナルを区切ってください。
TRCLMT	トレース限度を指定します (分単位)。キャプチャー・プログラムは、トレース限度よりも古い IBMSNAP_CAPTRACE 表の行をすべて除去します。デフォルトは 10,080 分 (トレース項目が 7 日間) です。 ***DFT (デフォルト)** キャプチャー・プログラムは、IBMSNAP_CAPPARMS 表の TRACE_LIMIT 列の値を使用します。 *trace-limit* 整理後に、トレース・データが IBMSNAP_CAPTRACE 表に保持される分数。
MONLMT	モニター限度を指定します (分単位)。キャプチャー・プログラムは、モニター限度よりも古い IBMSNAP_CAPMON 表の行を整理します。デフォルトは 10,080 分 (モニター項目が 7 日間) です。 ***DFT (デフォルト)** キャプチャー・プログラムは、IBMSNAP_CAPPARMS 表の MONITOR_LIMIT 列の値を使用します。 *monitor-limit* 整理後に、モニター・データが IBMSNAP_CAPMON 表に保持される分数。
MONITV	キャプチャー・プログラムが、 IBMSNAP_CAPMON 表に行を挿入する頻度 (秒単位) を指定します。デフォルトは 300 秒 (5 分) です。 ***DFT (デフォルト)** キャプチャー・プログラムは、IBMSNAP_CAPPARMS 表の MONITOR_INTERVAL 列の値を使用します。 *monitor-interval* IBMSNAP_CAPMON 表への行挿入の間隔の秒数。モニター・インターバルは、少なくとも 120 秒 (2 分) あける必要があります。120 よりも小さい数値がユーザーから入力された場合、パラメーター値は 120 に設定されます。

表 58. STRDPRCAP コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
MEMLMT	キャプチャー・ジャーナル・ジョブが使用できるメモリーの最大サイズ (MB 単位) を指定します。デフォルトは 32 MB です。 ***DFT (デフォルト)** キャプチャー・プログラムは、IBMSNAP_CAPPARMS 表の MEMORY_LIMIT 列の値を使用します。 *memory-limit* メモリーの最大値の MB 数。
RETAIN	データが除去されずに、変更データ (CD) 表、作業単位 (UOW) 表、IBMSNAP_SIGNAL 表、および IBMSNAP_AUTHTKN 表に保存される分数である、新規の保持制限を指定します。この値は、**CLNUPITV** パラメーターの値と共同で処理を行います。**CLNUPITV** の値に達すると、 CD、UOW、IBMSNAP_SIGNAL、および IBMSNAP_AUTHTKN データのうちで、保持制限よりも古いデータが削除されます。 表内のデータに矛盾が生じないように、データがこの **RETAIN** パラメーター値に達する前に変更情報がコピーされるようにアプライ・インターバルを設定してください。データの矛盾が発生した場合、アプライ・プログラムはフル・リフレッシュを実行します。 デフォルトは 10,080 分 (7 日) です。最大値は、35000000 分です。 ***DFT (デフォルト)** キャプチャー・プログラムは、IBMSNAP_CAPPARMS 表の RETENTION_LIMIT 列の値を使用します。 *retention-limit* CD、UOW、IBMSNAP_SIGNAL、および IBMSNAP_AUTHTKN データが保存される分数。
LAG	キャプチャー・プログラムの処理が遅れても、その間は再始動が行われない分数である、新規の遅延限度を指定します。 遅延限度に達すると (つまり、ジャーナル項目のタイム・スタンプが現在の時間から遅延限度を引いたものより古くなると)、キャプチャー・プログラムは、そのジャーナルで処理中の表のコールド・スタートを開始します。アプライ・プログラムはその後フル・リフレッシュを実行し、キャプチャー・プログラムに新しい開始点を提供します。 デフォルトは 10,080 分 (7 日) です。最大値は、35000000 分です。 ***DFT (デフォルト)** キャプチャー・プログラムは、IBMSNAP_CAPPARMS 表の LAG_LIMIT 列の値を使用します。 *lag-limit* キャプチャー・プログラムの遅れとして許容される分数。

表 58. STRDPRCAP コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義およびプロンプト
FRCFRQ	キャプチャー・プログラムが変更データ (CD) 表および作業単位 (UOW) 表に変更を書き込む頻度 (30 秒から 600 秒) を指定します。キャプチャー・プログラムは、バッファーがフルになるか、 FRCFRQ 時間制限が満了するか、いずれか先に発生した時点で、これらの変更をアプライ・プログラムから使用可能にします。 このパラメーターは、ソース表の変更率が少ないサーバー上で、アプライ・プログラムが変更をより早期に使用できるようにするために使用してください。**FRCFRQ** パラメーター値はグローバル値であり、すべての定義済みソース表で使用されます。**FRCFRQ** 値を低い数値に設定すると、システム・パフォーマンスが影響を受ける可能性があります。 デフォルトは 30 秒です。 ***DFT (デフォルト)** キャプチャー・プログラムは、IBMSNAP_CAPPARMS 表の COMMIT_INTERVAL 列の値を使用します。 *force-frequency* キャプチャー・プログラムが、アプライ・プログラムで変更を使用できるようにする前に、 CD 表および UOW 表の変更をバッファー・スペース内に保持する秒数。

使用上の注意

STRDPRCAP コマンドの **CLNUPITV** パラメーターは、キャプチャー・プログラムが変更データ (CD) 表、作業単位 (UOW) 表、IBMSNAP_SIGNAL 表、IBMSNAP_CAPMON 表、IBMSNAP_CAPTRACE 表、および IBMSNAP_AUTHTKN 表から古いレコードを除去するまでに待つ最大時間数を指定します。

STRDPRCAP コマンドは手動で実行することができます。また、このコマンドは、初期プログラム・ロード (IPL 始動プログラム) の一部として、自動的に実行することができます。

JOBD パラメーターで指定されたジョブ記述がジョブ・キュー QDP4/QZSNDPR を使用する場合に、DB2 DataPropagator for System i サブシステムがアクティブでない場合は、STRDPRCAP コマンドはサブシステムを始動します。ジョブ記述が別のジョブ・キューおよびサブシステムを使用するよう定義されている場合には、STRDPRCAP コマンドの実行前または後のいずれかに、サブシステムの始動 (STRSBS) コマンドを使用して、このサブシステムを手動で始動する必要があります。

```
STRSBS QDP4/QZSNDPR
```

システムの QSTRUPPGM システム値で参照されるプログラムに STRSBS コマンドを追加することにより、サブシステムを自動的に開始するようシステムを設定することができます。

ウォーム・スタートまたはコールド・スタートを使用したキャプチャー・プログラムの再始動

STRDPRCAP コマンドに対する **RESTART** パラメーターの値が、キャプチャー・プログラムがウォーム・スタートとコールド・スタートを処理する方法をコントロールします。

ウォーム・スタート処理
ほとんどの場合、ウォーム・スタート情報は保管されます。時々、ウォーム・スタート情報が保管されないことがあります。そのような場合、キャプチャー・プログラムは、CD 表、UOW 表、または IBMSNAP_PRUNCNTL 表を使用して、それが停止した時刻に再同期します。

自動コールド・スタート
場合によっては、ウォーム・スタートが指定されていても、キャプチャー・プログラムは自動的にコールド・スタートに切り替えます。 System i システムでは、コールド・スタートはジャーナルによるジャーナル方式で動作します。例えば、ジャーナルがラグの制限を超える場合、そのジャーナルを使用するすべてのレプリケーション・ソースは、別のジャーナルを使用するレプリケーション・ソースがコールド・モードで始動されていなくても、コールド・モードで始動されます。

STRDPRCAP の例

以下の例は、STRDPRCAP コマンドの使用法を示しています。

例 1:

2 つの異なるジャーナルに関して、キャプチャー・プログラムのウォーム・スタートを開始するには、次のようにします。

```
STRDPRCAP RESTART(*YES) JRN(HR/QSQJRN ACCTS/QSQJRN)
```

例 2

1 つの指定されたジャーナルに関してキャプチャー・プログラムを始動するには、次のようにします。

```
STRDPRCAP CAPCTLLIB(BSN) JRN(MARKETING/QSQJRN)
```

キャプチャー・コントロール表は BSN という名前のライブラリーにあります。

例 3

2 つのジャーナルの整理を行わないでキャプチャー・プログラムを始動するには、次のようにします。

```
STRDPRCAP RESTART(*YES) CLNUPITV(*DFT *NO) JRN(HR/QSQJRN ACCTS/QSQJRN)
```

例 4

デフォルトのキャプチャー・コントロール・ライブラリーにある、1 つの指定されたジャーナルについてキャプチャー・プログラムを始動し、トレース限度整理、モニター限度整理、 IBMSNAP_CAPMON 表挿入、およびメモリー限度のデフォルト・パラメーターを変更するには、次のようにします。

```
STRDPRCAP CAPCTLLIB(ASN) JRN(SALES/QSQJRN) TRCLMT(1440) MONLMT(1440)
  MONITV(3600) MEMLMT(64)
```

例 5

キャプチャー・プログラムのコールド・スタートを開始するには、次のようにします。

```
STRDPRCAP RESTART(*NO)
```

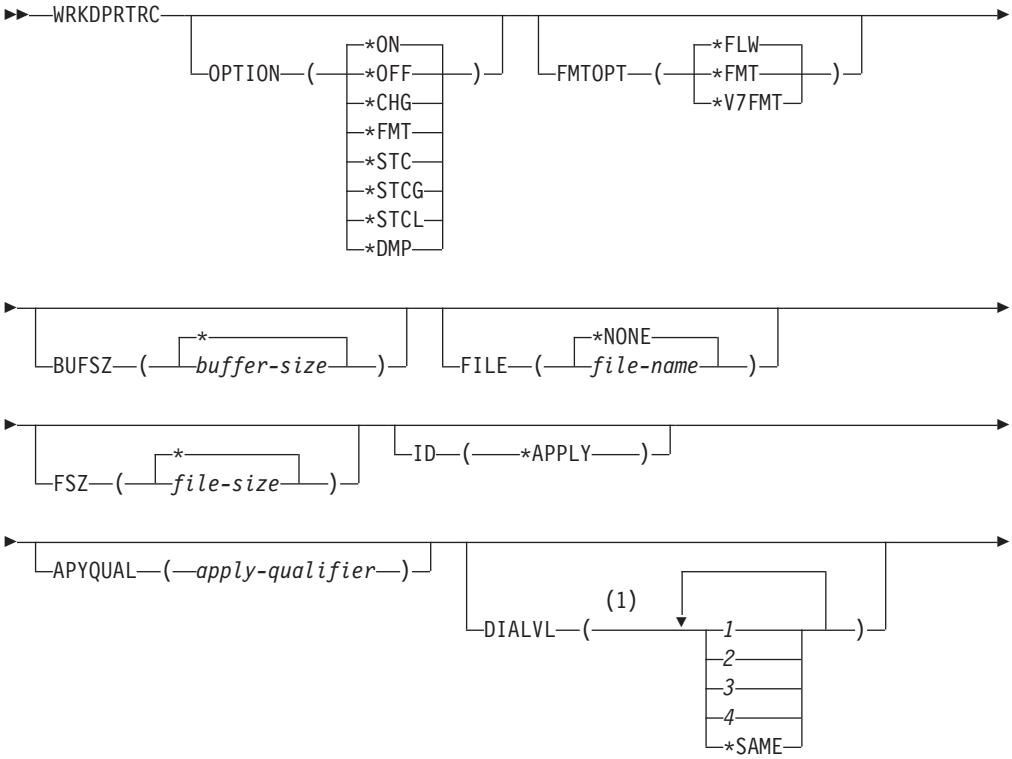
WRKDPRTRC: DPR トレース機能の使用方法 (System i)

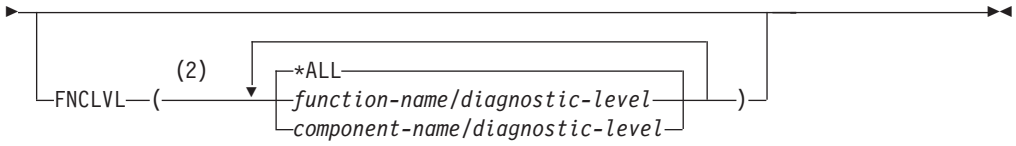
DPR トレース (WRKDPRTRC) コマンドは、IBM ソフトウェア・サポートによって使用を指示された場合にのみ使用してください。このコマンドは、トレース機能を実行し、指定されたアプライ・プログラムのプログラム・フロー情報を記録します。

コマンド行にコマンド名を入力してから、F4 キーを押してコマンド構文を表示することができます。

このコマンド、およびこのコマンドのすべてのパラメーターの詳しい記述を表示するには、画面の一番上のコマンドにカーソルを移動し、F1 キーを押します。特定のパラメーターの記述を表示するには、そのパラメーター上にカーソルを移動し、F1 キーを押します。

構文





注:

1 複数の値を指定できます。

2 最大 20 の関数またはコンポーネントを指定できます。

表 59 は、呼び出しパラメーターをリストしています。

表 59. WRKDPRTRC コマンド・パラメーター定義 (System i 版)

パラメーター	定義
OPTION	トレース機能を 1 つ指定します。 ***ON (デフォルト)** トレース機能をオンにします。このオプションは、トレース用の共有メモリー・セグメントを自動的に作成します。 ***OFF** トレース機能をオフにします。 ***CHG** トレース機能パラメーターの値を変更します。 ***FMT** 共有メモリーからのトレース機能の出力をフォーマットします。 ***STC** トレース機能の状況を表示します。この状況情報には、トレース・バージョン、アプリケーション・バージョン、項目数、バッファー・サイズ、使用中のバッファー量、状況コード、およびプログラム・タイム・スタンプが含まれます。 このパラメーター・オプションは、UNIX、Windows、および z/OS オペレーティング・システムで使用される asntrc コマンドの stat オプションと同等です。 ***STCG** レプリケーション・センターで読み取れるフォーマットでトレース機能の状況を表示します。 ***STCL** 追加のバージョン・レベル情報を含めてトレース機能の状況を表示します。この追加情報には、アプリケーション内の各モジュールのサービス・レベルが含まれ、長ストリングのテキストとして表示されます。 このパラメーター・オプションは、UNIX、Windows、および z/OS オペレーティング・システムで使用される asntrc コマンドの statlong オプションと同等です。 ***DMP** トレース・バッファーの現在の内容をファイルに書き込みます。 WRKDPRTRC コマンドのプロンプトから F4 キーを押すと、トレース・オプションのリストを表示できます。

表 59. WRKDPRTRC コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義
FMTOPT	フォーマット ID のオプションを指定します。 **OPTION(*FMT)** パラメーターと一緒に使用されます。 ***FLW (デフォルト)** 関数呼び出しのフローを表示します。 ***FMT** トレース・バッファーまたはトレース・ファイルのフォーマットを表示します。すべての詳細データが表示されます。 ***V7FMT** トレース・バッファーまたはトレース・ファイル情報をバージョン 7 の形式にフォーマットします。 WRKDPRTRC コマンドのプロンプトから F4 キーを押すと、フォーマット・オプションのリストを表示できます。
BUFSZ	トレース・バッファーのサイズを指定します (バイト単位)。数字の後ろに M、K、または G を入力すれば、メガバイト、キロバイト、またはギガバイトをそれぞれ指定できます。 デフォルトは 2 メガバイトです。 *** (デフォルト)** 2 MB のデフォルト・サイズが使用されます。 *buffer-size* バッファー・サイズ (バイト単位)。
FILE	トレース出力をファイルに書き込むかどうかを指定します。 ***NONE(デフォルト)** トレース出力は共有メモリーにのみ入れられます。 *file-name* 出力ファイルの名前。 **OPTION(*DMP)** パラメーターが使用された場合、このファイル名はダンプ・ファイルの名前を表します。
FSZ	トレース・データが保管されるファイルのサイズを指定します (バイト単位)。数字の後ろに M、K、または G を入力すれば、メガバイト、キロバイト、またはギガバイトをそれぞれ指定できます。 デフォルトは 2 GB です。 *** (デフォルト)** 2 GB のデフォルト・サイズが使用されます。 *file-size* ファイル・サイズ (バイト単位)。

表 59. WRKDPRTRC コマンド・パラメーター定義 (System i 版) (続き)

パラメーター	定義
ID	トレース対象のプログラムのタイプを指定します。 ***APPLY (デフォルト)** アプライ・プログラムのトレース。
APYQUAL	トレースするアプライ・プログラムの名前を指定します。 *apply-qualifier* アプライ修飾子の名前。
DIALVL	トレース機能により記録されるトレース・レコードのタイプを指定します。トレース・レコードは、以下の診断マスク番号により分類されます。 **1** フロー・データ。関数の入口点と出口点が含まれます。 **2** 基本データ。トレース機能が検出したすべての主要なイベントが含まれます。 **3** 詳細データ。主要なイベントとその記述が含まれます。 **4** パフォーマンス・データ。 ***SAME** このコマンドは、直前のトレース機能で使用された診断レベル設定値を使用します。 1 つ、または複数の診断マスク番号を入力できます。番号は昇順に入力する必要があります。数字間にスペースを入れないでください。 **重要:** 番号レベルは包括的ではありません。 トレース機能の開始時のデフォルトは DIALVL(1234) です。その後にトレース機能を呼び出した場合のデフォルトは *SAME です。 WRKDPRTRC コマンドのプロンプトから F4 キーを押すと、使用可能な診断レベルのリストを表示できます。
FNCLVL	特定の関数またはコンポーネント ID をトレースするかどうかを指定します。 ***ALL (デフォルト)** すべての関数およびコンポーネントがトレース機能に含まれます。 *function-name/diagnostic-level* トレースする関数の名前と、対応する診断マスク番号。 *component-name/diagnostic-level* トレースするコンポーネントの名前と、対応する診断マスク番号。 最大 20 の関数またはコンポーネント名を入力できます。

WRKDPRTRC の例

以下の例は、WRKDPRTRC コマンドの使用法を示しています。

例 1:

すべての関数およびコンポーネントについて、アプライ修飾子 AQ1 のアプライ・トレースを開始し、出力を TRCFILE という名前のファイルに書き込むには、次のようにします。

```
WRKDPRTRC OPTION(*ON) FILE(TRCFILE) ID(*APPLY) APYQUAL(AQ1)
```

例 2

アプライ修飾子 AQ1 のアプライ・トレースを終了するには、次のようにします。

```
WRKDPRTRC OPTION(*OFF) ID(*APPLY) APYQUAL(AQ1)
```

例 3

すべての関数およびコンポーネントについて、アプライ修飾子 AQ1 のアプライ・トレースを診断レベル 3 および 4 (詳細データおよびパフォーマンス・データ) に変更するには、次のようにします。

```
WRKDPRTRC OPTION(*CHG) ID(*APPLY) APYQUAL(AQ1) DIALVL(34)
```

例 4

アプライ修飾子 AQ1 のアプライ・トレースの状況を表示するには、次のようにします。

```
WRKDPRTRC OPTION(*STC) ID(*APPLY) APYQUAL(AQ1)
```

例 5

アプライ修飾子 AQ1 の関数呼び出しを診断レベル 3 および 4 で表示するには、次のようにします。

```
WRKDPRTRC OPTION(*FMT) FMTOPT(*FLW) ID(*APPLY) APYQUAL(AQ1) DIALVL (34)
```

例 6

アプライ修飾子 AQ1 のアプライ・トレース情報を DMPFILE という名前のダンプ・ファイルに書き込むには、次のようにします。

```
WRKDPRTRC OPTION(*DMP) FILE(DMPFILE) ID(*APPLY) APYQUAL(AQ1)
```

第 24 章 SQL レプリケーション表の構造

リレーショナル・データベース表は、各サーバー (キャプチャー・コントロール・サーバー、アプライ・コントロール・サーバー、モニター・コントロール・サーバー、およびターゲット・サーバー) のレプリケーション・プログラムに関する情報の保管に使用されます。これらの表のことを、*コントロール表* と呼びます。

以下のトピックでは、コントロール表の構造と、SQL レプリケーションに特有の他の表の構造について説明しています。

キャプチャー・コントロール・サーバーの表

キャプチャー・コントロール・サーバーに保管される表には、ユーザーの登録済みソースに関する情報と、キャプチャー・プログラムまたはトリガーがソースを処理する方法に関する情報が入っています。

Linux、UNIX、Windows、および z/OS の場合は、ASNCLP コマンド行プログラムまたはレプリケーション・センターを使用して、ユーザーの指定に合わせてこれらのコントロール表を作成します。 System i の場合は、 DataPropagator for System i のインストール時に、ASN ライブラリーの中にこれらのコントロール表が自動的に作成されます。System i コマンドを使用して、代替のキャプチャー・スキーマ内にキャプチャー・コントロール表を作成できます。

表 60 は、キャプチャー・サーバーのコントロール表を説明しています。

表 60. キャプチャー・コントロール・サーバーで使用される表のクイック・リファレンス

表名	説明
437 ページの『IBMSNAP_CAPSCHEMAS 表』	すべてのキャプチャー・スキーマの名前を保持します。
IBMSNAP_AUTHTKN表(System i)	Update-anywhere レプリケーションをサポートするための情報が入っています。
z/OS Linux UNIX Windows 431 ページの『IBMSNAP_CAPENQ 表 (z/OS、Linux、 UNIX、 Windows)』	この表は、各キャプチャー・スキーマについて、次のことを保証するために使用されます。 • Linux UNIX Windows DB2 for Linux, UNIX and Windows の場合は、1 つのデータベースに対して 1 つのキャプチャー・プログラムのみが実行される。 • z/OS データを共有しない DB2 for z/OS の場合は、1 つのサブシステムに対して 1 つのキャプチャー・プログラムのみが実行される。 • z/OS データを共有する DB2 for z/OS の場合は、1 つのデータ共有グループに対して 1 つのキャプチャー・プログラムのみが実行される。

表 60. キャプチャー・コントロール・サーバーで使用される表のクイック・リファレンス (続き)

表名	説明
441 ページの『CD 表』	ソースに発生する変更に関する情報を保持します。この表は、レプリケーション・ソースが登録されるまでは作成されません。
439 ページの『CCD 表 (DB2 以外)』	ソースに発生する変更に関する情報と、これらの変更の順序を識別するための追加の列を保持します。
432 ページの『IBMSNAP_CAPMON 表』	キャプチャー・プログラムの進行状況のモニターに役立つ操作統計を保持します。
433 ページの『IBMSNAP_CAPPARMS 表』	キャプチャー・プログラムの操作をコントロールするためにユーザーが指定できるパラメーターを保持します。
438 ページの『IBMSNAP_CAPTRACE 表』	キャプチャー・プログラムからのメッセージを保持します。
442 ページの『IBMQREP_IGNTRAN 表』	DB2 リカバリー・ログからキャプチャーしないトランザクションについてキャプチャー・プログラムに通知するために使用できます。
442 ページの『IBMQREP_IGNTRANTRC 表』	無視するように指定されたトランザクションについての情報を記録します。
443 ページの『IBMSNAP_PARTITIONINFO 表』	必要とされる中で一番古いログ・シーケンス番号からキャプチャー・プログラムを再始動できるようにするための情報が含まれます。
446 ページの『IBMSNAP_PRUNE_LOCK 表』	コールド・スタート、または保存限度整理 (保存限度に達したか、超えたときの整理) 時に、キャプチャー・プログラムの CD 表へのアクセスをシリアライズするために使用されます。
447 ページの『IBMSNAP_PRUNE_SET 表』	CD 表の整理を調整します。
444 ページの『IBMSNAP_PRUNCNTL 表』	キャプチャー・プログラムとアプライ・プログラムの間で同期点更新を調整します。
System i IBMSNAP_REG_EXT (System i)	登録表を拡張したものです。ジャーナル名やリモート・ソース表のデータベース項目名など、レプリケーション・ソースに関する追加情報が含まれます。
449 ページの『IBMSNAP_REGISTER 表』	レプリケーション・ソース表の名前、その属性、および対応する CD 表および CCD 表の名前など、レプリケーション・ソースに関する情報が入ります。
457 ページの『IBMSNAP_REG_SYNCH 表 (DB2 以外のリレーショナル)』	DB2 以外のリレーショナル・データ・ソースから複製するときに使用されます。この表の更新トリガーは、アプライ・プログラムが登録表から情報を読み込む前に、登録表のすべての行で SYNCHPOINT 値の更新を始めることにより、キャプチャー・プログラムをシミュレートします。

表 60. キャプチャー・コントロール・サーバーで使用される表のクイック・リファレンス (続き)

表名	説明
457 ページの『IBMSNAP_RESTART 表』	キャプチャー・プログラムがログまたはジャーナル内の正しい時点からキャプチャーを再開できるようにするための情報が入っています。 System i 環境では、この表は RCVJRNE (ジャーナル項目の受信) コマンドの開始時刻を判別するためにも使用されます。
459 ページの『IBMSNAP_SEQTABLE 表 (Informix)』	Informix 表のログ・シーケンス番号と同等のものとして SQL レプリケーションが使用される一連のユニーク番号を保持します。
460 ページの『IBMSNAP_SIGNAL 表』	キャプチャー・プログラムにプロンプトを出すために使用されるすべてのシグナルを保持します。これらのシグナルは手動で、またはアプライ・プログラムから送信できます。
463 ページの『IBMSNAP_UOW 表』	ソース表にコミットされたトランザクションに関する追加情報を提供します。

IBMSNAP_CAPENQ 表 (z/OS、 Linux、 UNIX、 Windows)

単一キャプチャー・スキーマの場合、IBMSNAP_CAPENQ 表は、データベース、サブシステム、またはデータ共用グループ当たり 1 つのキャプチャー・プログラムだけが実行されることを保証するものです。

サーバー: キャプチャー・コントロール・サーバー

デフォルト・スキーマ: ASN

索引: なし

重要: SQL を使用してこの表を更新するときには、注意してください。この表の変更の方法が不適切であると、予期せぬ結果が生じたり、データが失われたりします。

IBMSNAP_CAPENQ 表は、DB2 以外のリレーショナル・サーバー、または System i サーバーでは使用されません。

キャプチャー・プログラムは実行中、この表を排他的にロックします。

表 61 では、IBMSNAP_CAPENQ 表の列の要旨を示します。

表 61. IBMSNAP_CAPENQ 表の列

列名	説明
LOCKNAME	**データ・タイプ**: CHAR(9)。**NULL 可能**: 可 この列にはデータは含まれません。

IBMSNAP_CAPMON 表

キャプチャー・プログラムは、インターバルが終了するたびに IBMSNAP_CAPMON 表に行を挿入して、操作統計を提供します。レプリケーション・センターはこの表 (およびその他の表) の情報を使用するため、ユーザーはキャプチャー・プログラムの状況をモニターできます。

サーバー: キャプチャー・コントロール・サーバー

デフォルト・スキーマ: ASN

索引: MONITOR_TIME

IBMSNAP_CAPPARMS 表で、ユーザーが MONITOR_INTERVAL に対して指定した値は、キャプチャー・プログラムがキャプチャー・モニター表に挿入を行う頻度を示し、ユーザーが MONITOR_LIMIT に対して指定した値は、表の行が整理の対象となるまでに、表内に留まる分数を示します。

表 62 では、IBMSNAP_CAPMON 表の列の要旨を示します。

表 62. IBMSNAP_CAPMON 表の列

列名	説明
MONITOR_TIME	**データ・タイプ**: TIMESTAMP。**NULL 可能**: 不可 この表に行が挿入されたときの (キャプチャー・コントロール・サーバーでの) タイム・スタンプ。
RESTART_TIME	**データ・タイプ**: TIMESTAMP。**NULL 可能**: 不可 キャプチャー・プログラムの現在の呼び出しが再始動されたときのタイム・スタンプ。
CURRENT_MEMORY	**データ・タイプ**: INT。**NULL 可能**: 不可 キャプチャー・プログラムが使用したメモリーの量 (バイト単位)。
CD_ROWS_INSERTED	**データ・タイプ**: INT。**NULL 可能**: 不可 すべてのソース表について、キャプチャー・プログラムが CD 表に挿入した行数。
RECAP_ROWS_SKIPPED	**データ・タイプ**: INT。**NULL 可能**: 不可 Update-anywhere レプリケーションの場合、キャプチャー・プログラムが処理したが、 CD 表に挿入していない行数。キャプチャー・プログラムの登録では、この表に複製された変更でも、このソース・サーバーから発生したものではない変更は、再キャプチャーしないように定義されているため、これらの行はスキップされます。
TRIGR_ROWS_SKIPPED	**データ・タイプ**: INT。**NULL 可能**: 不可 キャプチャー・プログラムが処理したが、 CD 表に挿入していない行数。キャプチャー・プログラムの登録で、特定の行を抑制するようにトリガーが定義されているため、これらの行はスキップされます。

表 62. IBMSNAP_CAPMON 表の列 (続き)

列名	説明
CHG_ROWS_SKIPPED	**データ・タイプ**: INT。**NULL 可能**: 不可 キャプチャー・プログラムが処理したが、 CD 表に挿入していない行数。キャプチャー・プログラムの登録では、登録済みの列で発生した変更のみをキャプチャーするように定義されているため、これらの行はスキップされます。
TRANS_PROCESSED	**データ・タイプ**: INT。**NULL 可能**: 不可 キャプチャー・プログラムが処理した、ソース・システム上のトランザクションの数。
TRANS_SPILLED	**データ・タイプ**: INT。**NULL 可能**: 不可 メモリー制限のためにキャプチャー・プログラムがディスクに書き出した、ソース・システム上のトランザクションの数。
MAX_TRAN_SIZE	**データ・タイプ**: INT。**NULL 可能**: 不可 ソース・システムで発生した最大のトランザクション。トランザクション・サイズを知ることによって、メモリー・パラメーターの変更を検討することになります。
LOCKING_RETRIES	**データ・タイプ**: INT。**NULL 可能**: 不可 デッドロックにより再処理が必要となった回数。
JRN_LIB (System i)	**データ・タイプ**: CHAR(10)。**NULL 可能**: 可 System i キャプチャー・プログラムが処理していたジャーナルのライブラリー名。
JRN_NAME (System i)	**データ・タイプ**: CHAR(10)。**NULL 可能**: 可 キャプチャー・プログラムが処理していたジャーナルの名前。
LOGREADLIMIT	**データ・タイプ**: INT。**NULL 可能**: 不可 1000 のレコードが読み取られたが、それら 1000 のレコードの中に完了済みのトランザクションが見つからなかったために、キャプチャー・プログラムがログ・レコード読み取りを一時停止した回数。
CAPTURE_IDLE	**データ・タイプ**: INT。**NULL 可能**: 不可 処理するものがないため、キャプチャー・プログラムがスリープした回数。
SYNCHTIME	**データ・タイプ**: TIMESTAMP。**NULL 可能**: 不可 この表にモニター・レコードが挿入されたときに、登録表のグローバル行から読み取られた SYNCHTIME の現行値。

IBMSNAP_CAPPARMS 表

IBMSNAP_CAPPARMS 表には、キャプチャー・プログラムの動作を制御するために変更できるパラメーターが含まれています。これらのパラメーターを定義して、キャプチャー・プログラムが整理を行う前にデータを CD 表および UOW 表の中に保持する時間の長さや、ログ・レコードの処理においてキャプチャー・プログラムに許される遅延時間などの値を設定できます。ユーザーがこの表のパラメーターを変更しても、キャプチャー・プログラムは始動時にしか変更を読み取りません。

サーバー: キャプチャー・コントロール・サーバー

デフォルト・スキーマ: ASN

索引: なし

この表の情報は、SQL を使って更新できます。

表 63 では、IBMSNAP_CAPPARMS 表の列の要旨を示します。

表 63. IBMSNAP_CAPPARMS 表の列

列名	説明
RETENTION_LIMIT	**データ・タイプ**: INT。**NULL 可能**: 可 通常の基準に基づいて整理が行われていないときに、 CD 表、UOW 表、およびシグナル表の中の行が整理の対象となるまで、表の中に留まる時間の長さ。一般的に CD 行および UOW 行は、すべてのターゲットに適用された後で除去され、シグナル行は、サイクルが完了 (SIGNAL_STATE = C) したときに除去されます。
LAG_LIMIT	**データ・タイプ**: INT。**NULL 可能**: 可 ログ・レコードの処理時にキャプチャー・プログラムがシャットダウンせずに処理を遅らせることができる分数。更新頻度の高い期間では、更新よりもフル・リフレッシュの方が経済的です。
COMMIT_INTERVAL	**データ・タイプ**: INT。**NULL 可能**: 可 UOW 表および CD 表を含めて、キャプチャー・プログラムがキャプチャー・コントロール表にデータをコミットする頻度 (秒単位)。キャプチャー・スレッドと整理スレッドの競合を避けるために、この値は DB2 ロックアウトの値よりも小さい値にする必要があります。
PRUNE_INTERVAL	**データ・タイプ**: INT。**NULL 可能**: 可 キャプチャー・プログラムが、不必要になった CD 表、UOW 表、シグナル表、トレース表、およびキャプチャー・モニター表の行を自動的に除去する (AUTOPRUNE = Y) 頻度 (秒単位)。除去するインターバルを短くするとスペースは節約できますが、処理コストが増加します。除去するインターバルを長くすると CD 表および UOW 表のスペースはより多く必要になりますが、処理コストは減少します。
TRACE_LIMIT	**データ・タイプ**: INT。**NULL 可能**: 可 整理の対象となるまでに、行が IBMSNAP_CAPTRACE 表の中に留まる分数。整理プロセス時に、分数 (現在のタイム・スタンプからキャプチャー・トレース表に行が挿入された時刻を引いたもの) が TRACE_LIMIT の値を超えると、キャプチャー・トレース表の中の行を削除します。
MONITOR_LIMIT	**データ・タイプ**: INT。**NULL 可能**: 可 整理の対象となるまでに、行が IBMSNAP_CAPMON 表の中に留まる分数。整理プロセス時に、分数の値 (現在のタイム・スタンプから MONITOR_TIME を引いたもの) が MONITOR_LIMIT の値を超えると、キャプチャー・モニター表の中の行を削除します。

表 63. IBMSNAP_CAPPARMS 表の列 (続き)

列名	説明
MONITOR_INTERVAL	**データ・タイプ**: INT。**NULL 可能**: 可 モニター・スレッドがキャプチャー・モニター IBMSNAP_CAPMON 表に行を追加する頻度 (秒単位)。System i の場合は、120 よりも長いインターバルを入力してください。
MEMORY_LIMIT	**データ・タイプ**: SMALLINT。**NULL 可能**: 可 キャプチャー・プログラムが使用することを許されているメモリーの量 (MB 単位)。この割り振りがすべて使用されてしまうと、メモリー・トランザクションはファイルに書き出されます。
REMOTE_SRC_SERVER	**データ・タイプ**: CHAR(18)。**NULL 可能**: 可 SQL レプリケーションの将来のオプション用に予約済み。この列には現在、デフォルト値の NULL が入っています。
AUTOPRUNE	**データ・タイプ**: CHAR(1)。**NULL 可能**: 可 キャプチャー・プログラムが、不必要になった行を CD 表、UOW 表、シグナル表、トレース表、およびキャプチャー・モニター表から自動的に除去するかどうかを示すフラグ。 **Y** 自動整理はオン。 **N** 自動整理はオフ。
TERM	**データ・タイプ**: CHAR(1)。**NULL 可能**: 可 DB2 が静止または停止するとき、キャプチャー・プログラムが停止するかどうかを示すフラグ。 **Y** DB2 が静止または停止した場合、キャプチャー・プログラムは終了します。 **N** キャプチャー・プログラムはアクティブのまま、DB2 が再始動または静止解除されるのを待ちます。
AUTOSTOP	**データ・タイプ**: CHAR(1)。**NULL 可能**: 可 アクティブ・ログの最後に達したときにキャプチャー・プログラムが変更のキャプチャーを終了するかどうかを示すフラグ。 **Y** アクティブ・ログの最後に達するとキャプチャー・プログラムは変更のキャプチャーをすぐに終了します。 **N** アクティブ・ログの最後に達してもキャプチャー・プログラムは実行を続けます。
LOGREUSE	**データ・タイプ**: CHAR(1)。**NULL 可能**: 可 キャプチャー・プログラムがキャプチャー・ログ・ファイルに上書きするか、ファイルに追加するかを示すフラグ。 **Y** キャプチャー・プログラムは再始動時に、最初にログ・ファイルを削除してから再作成してログ・ファイルを再利用します。 **N** キャプチャー・プログラムは新しい情報をキャプチャー・ログ・ファイルに追加します。

表 63. IBMSNAP_CAPPARMS 表の列 (続き)

列名	説明
LOGSTDOUT	**データ・タイプ**: CHAR(1)。**NULL 可能**: 可 キャプチャー・プログラムがログ・ファイル・メッセージを送る場所を示すフラグ。 **Y** キャプチャー・プログラムは、標準出力 (STDOUT) とログ・ファイルの両方にログ・ファイル・メッセージを送ります。 **N** キャプチャー・プログラムは、ほとんどのログ・ファイル・メッセージをログ・ファイルにのみ送ります。初期化メッセージは、標準出力 (STDOUT) とログ・ファイルの両方に送られます。
z/OS Linux UNIX Windows SLEEP_INTERVAL (z/OS、Linux、UNIX、Windows)	**データ・タイプ**: SMALLINT。**NULL 可能**: 可 アクティブ・ログの最後に達したとき (Linux、UNIX および Windows、または z/OS のデータを共有しない環境の場合)、または戻されたデータの量が不十分なとき (z/OS のデータ共有環境の場合) にキャプチャー・プログラムがスリープする秒数。
CAPTURE_PATH	**データ・タイプ**: VARCHAR(1040)。**NULL 可能**: 可 キャプチャー・プログラムからの出力が送信されるパス。
STARTMODE	**データ・タイプ**: VARCHAR(10)。**NULL 可能**: 可 キャプチャー・プログラムの始動時に使用される処理プロシージャー。 **cold** キャプチャー・プログラムは、初期化時に CD 表と UOW 表の中のすべての行を削除します。これらのレプリケーション・ソースに対するすべてのサブスクリプションは、次のアプライ処理サイクル時にフル・リフレッシュされます (つまり、すべてのデータがソース表からターゲット表にコピーされます)。キャプチャー・プログラムがコールド・スタートを試行したときに、フル・リフレッシュが使用不可な場合には、キャプチャー・プログラムは開始されますが、アプライ・プログラムは失敗し、エラー・メッセージが発行されます。 **warmsi** キャプチャー・プログラムはウォーム・スタートします。ただし、キャプチャー・プログラムを初めて開始する場合には、コールド・スタートに切り替えられます。warmsi 開始モードは、キャプチャー・プログラムを初めて開始する場合にのみコールド・スタートが行われるようにします。 **warmns** キャプチャー・プログラムはウォーム・スタートします。ウォーム・スタートできない場合、コールド・スタートに切り替わりません。 warmns 開始モードは、予期せずにコールド・スタートが発生することを防止します。これは、修復が必要な、ウォーム・スタートの進行を阻害している問題 (データベースまたは表スペースが使用できないなど) が発生した場合に便利です。ウォーム・スタートした場合、キャプチャー・プログラムは終了したところから処理を再開します。キャプチャー・プログラムの始動後にエラーが発生した場合は、キャプチャー・プログラムは終了し、すべての表はそのまま残されます。

IBMSNAP_CAPSCHEMAS 表

IBMSNAP_CAPSCHEMAS 表は、すべてのキャプチャー・スキーマの名前を保留します。管理ツールやその他のユーティリティーはこの表を使用して、特定のキャプチャー・コントロール・サーバーのすべての表を迅速に検出できます。ユーザーが新しいキャプチャー・スキーマを作成するたびに、行が 1 つ自動的に挿入されます。

サーバー: キャプチャー・コントロール・サーバー

索引: CAP_SCHEMA_NAME

重要: SQL を使用してこの表を更新するときには、注意してください。この表を不適切に変更すると、管理ツールの使用中に予期しない結果が生じることがあります。

以下の 2 つの表は、オペレーティング・システム別の IBMSNAP_CAPSCHEMAS 表のレイアウトを示しています。

表 64. System i 以外のオペレーティング・システムの場合の IBMSNAP_CAPSCHEMAS 表の列

列名	説明
CAP_SCHEMA_NAME	**データ・タイプ**: VARCHAR(30)、DB2 UDB for z/OS バージョン 8 新機能モードの場合 VARCHAR(128)。**NULL 可能**: 可 キャプチャー・スキーマの名前。キャプチャー・スキーマごとに 1 つの行があります。

表 65. System i の場合のキャプチャー・スキーマ表の列

列名	説明
CAP_SCHEMA_NAME	**データ・タイプ**: VARCHAR(30)。**NULL 可能**: 可 キャプチャー・スキーマの名前。キャプチャー・スキーマごとに 1 つの行があります。
STATUS	**データ・タイプ**: CHAR(1)。**NULL 可能**: 可 このキャプチャー・スキーマによって識別されるキャプチャー・プログラムが実行中かどうかを示すフラグ。 **Y** キャプチャー・プログラムは実行中です。 **N** キャプチャー・プログラムは実行されていません。

IBMSNAP_AUTHTKN 表 (System i)

IBMSNAP_AUTHTKN 表は、System i 環境でのみ使用されます。この表は、Update-anywhere レプリケーション時に、特定のアプライ・プログラムによって実行されたトランザクションをトラッキングするために使用されます。キャプチャー・プログラムは、ユーザーが設定した保存限度に基づいて、この表の整理を行います。

サーバー: キャプチャー・コントロール・サーバー

デフォルト・スキーマ: ASN

索引: JRN_LIB、JRN_NAME

重要: SQL を使用してこの表を更新するときには、注意してください。この表の変更の方法が不適切であると、予期せぬ結果が生じたり、データが失われたりします。

表 66 では、IBMSNAP_AUTHTKN 表の列の要旨を示します。

表 66. IBMSNAP_AUTHTKN 表の列

列名	説明
APPLY_QUAL	**データ・タイプ**: CHAR(18)。**NULL 可能**: 不可 トランザクションを処理したアプライ・プログラムを識別するためのアプライ修飾子。この修飾子は、Update-anywhere レプリケーションにおいて、アプライ・プログラムが同じ変更を繰り返し複製しないようにするために使用されます。
IBMSNAP_AUTHTKN	**データ・タイプ**: CHAR(26)。**NULL 可能**: 不可 トランザクションに関連付けられたジョブ名。Capture for System i は、この列の名前と、トランザクションを発行したジョブの名前を突き合わせて、トランザクションがアプライ・プログラムから発行されたものか、ユーザー・アプリケーションから発行されたものかを判別します。ジョブ名が一致すると、Capture for System i は、この表の APPLY_QUAL 列にあるアプライ修飾子を、UOW 表の対応する行の APPLY_QUAL 列にコピーします。名前が一致しない場合、Capture for System i は、UOW 行の APPLY_QUAL 列を NULL に設定します。この列は自動的には他の表にコピーされません。ユーザー・データ列として選択してコピーする必要があります。
JRN_LIB	**データ・タイプ**: CHAR(10)。**NULL 可能**: 不可 トランザクションの発行元のジャーナルのライブラリー名。
JRN_NAME	**データ・タイプ**: CHAR(10)。**NULL 可能**: 不可 トランザクションの発行元のジャーナルの名前。
IBMSNAP_LOGMARKER	**データ・タイプ**: TIMESTAMP。**NULL 可能**: 不可 トランザクションがキャプチャー・コントロール・サーバーでコミットされたおおよその時間。

IBMSNAP_CAPTRACE 表

キャプチャー・トレース表は、キャプチャー・プログラムからメッセージを保持します。

サーバー: キャプチャー・コントロール・サーバー

デフォルト・スキーマ: ASN

索引: TRACE_TIME

以下の 2 つの表は、オペレーティング・システム別の IBMSNAP_CAPTRACE 表のレイアウトを示しています。 z/OS Linux UNIX Windows

表 67. Linux、UNIX、Windows、および z/OS の IBMSNAP_CAPTRACE 表の列

列名	説明
OPERATION	**データ・タイプ**: CHAR(8)。**NULL 可能**: 不可 キャプチャー・プログラムの操作のタイプ (例えば、初期化、キャプチャー、またはエラー条件)。
TRACE_TIME	**データ・タイプ**: TIMESTAMP。**NULL 可能**: 不可 キャプチャー・トレース表に行が挿入されたときの、キャプチャー・コントロール・サーバーにおける時刻。
DESCRIPTION	**データ・タイプ**: VARCHAR(1024)。**NULL 可能**: 不可 メッセージ ID とメッセージ・テキスト。エラー・メッセージ、警告メッセージ、または情報メッセージです。この列に入れられるテキストは英語のみです。

System i

表 68. System i のキャプチャー・トレース表の列

列名	説明
OPERATION	**データ・タイプ**: CHAR(8)。**NULL 可能**: 不可 初期化、キャプチャー、またはエラー条件など、キャプチャー・プログラムで実行された操作のタイプ。
TRACE_TIME	**データ・タイプ**: TIMESTAMP。**NULL 可能**: 不可 キャプチャー・トレース表に行が挿入された時刻。トレース限度整理の対象となる TRACE_TIME 行は、キャプチャー・プログラムが CD 表および UOW 表の整理を行うときに削除されます。
JOB_NAME	**データ・タイプ**: CHAR(26)。**NULL 可能**: 不可 このトレース項目を書き込んだジョブの完全修飾名。 **位置　説明** **1–10**　キャプチャー・スキーマ名またはジャーナル・ジョブ名 **11–20**　キャプチャー・プログラムを始動したユーザーの ID **21–26**　ジョブ番号
JOB_STR_TIME	**データ・タイプ**: TIMESTAMP。**NULL 可能**: 不可 JOB_NAME 列で指定されたジョブの開始時刻。
DESCRIPTION	**データ・タイプ**: VARCHAR(298)。**NULL 可能**: 不可 メッセージ ID とメッセージ・テキスト。メッセージ ID は DESCRIPTION 列の最初の 7 文字です。 メッセージ・テキストは、DESCRIPTION 列の位置 9 から始まります。

CCD 表 (DB2 以外)

キャプチャー・コントロール・サーバー上の整合変更データ (CCD) 表は、 DB2 以外のソースで発生した変更に関する情報と、これらの変更の順序を識別するための

追加の列を含む表です。キャプチャー・コントロール・サーバー上の CCD 表は、アプライ・プログラム以外のプログラムによってデータがキャプチャーされる表です。

サーバー: キャプチャー・コントロール・サーバー

重要: SQL を使用してこの表を更新するときには、注意してください。この表を不適切に変更すると、データの損失が生じることがあります。

キャプチャー・コントロール・サーバーは、次のいずれかになります。

- DB2 以外のリレーショナル・ソースの内部 CCD 表。

  変更キャプチャー・レプリケーションの場合、DB2 以外のリレーショナル・ソースで更新が行われると、キャプチャー・プログラムはこの表で挿入変更を起動します。このタイプの CCD 表の名前は、IBMSNAP_REGISTER 表の中で、変更元のレプリケーション・ソースと同じ行に保管されます。この表は、 DB2 以外のリレーショナル・ソースを登録したときに作成される整理トリガーによって自動的に整理が行われます。
- 非リレーショナルおよびマルチベンダーのデータ用の外部 CCD 表。

  外部プログラムは、SQL レプリケーションにより、レプリケーション・ソースとして使用される CCD 表を作成できます。これらの外部プログラムは、IMS データのコピーをリレーショナル・データベース内に再作成できるように、CCD 表の中の IMS 変更をキャプチャーします。外部プログラムは、コントロール列の正しい値を初期化、保守および提供する必要があります。IMS DataPropagator または DataRefresher などのプログラムによって保守されていない、外部で取り込まれる CCD 表がある場合には、アプライ・プログラムがソースとして CCD 表を読み取り、正しく機能することができるように、ユーザー自身でこれらの表を保守する必要があります。

表 69 では、CCD 表の列の要旨を示します。

表 69. CCD 表の列

列名	説明
IBMSNAP_INTENTSEQ	変更を一意的に識別するシーケンス番号。この値はグローバルに昇順です。
IBMSNAP_OPERATION	レコードの操作のタイプを示すフラグ。 **I** 挿入 **U** 更新 **D** 削除
IBMSNAP_COMMITSEQ	トランザクションの順序を指定するシーケンス番号。
IBMSNAP_LOGMARKER	データがコミットされた時刻。
user key columns	CCD 表が圧縮されている場合、この列には、ターゲット・キーを構成する列が含まれます。
user non-key columns	ソース表からの非キー・データ列。ソース表の中の列名とこれらの列名が一致している必要はありませんが、データ・タイプは互換性がなければなりません。
user computed columns	SQL 式から派生したユーザー定義の列。ソース・データ・タイプを別のターゲット・データ・タイプに変換するために、SQL 関数で算出列を使用することができます。

CD 表

変更データ (CD) 表は、レプリケーション・ソースに対して行われたすべてのコミット済み変更を記録します。CD 表の整理は、IBMSNAP_PRUNE_SET 表によって調整されます。CD 表はキャプチャー・コントロール表とは異なり、ユーザーがレプリケーション・ソースを定義したときに作成されます。キャプチャー・コントロール・サーバー用のコントロール表を作成するときに自動的に作成されるのではありません。

サーバー: キャプチャー・コントロール・サーバー

重要: SQL を使用してこの表を更新するときには、注意してください。この表を不適切に変更すると、データの損失が生じることがあります。

表 70 では、CD 表の列の要旨を示します。

表 70. CD 表の列

列名	説明
IBMSNAP_COMMITSEQ	キャプチャーされたコミット・ステートメントのログ・シーケンス番号。UOW 表にもあるこの列は、アプライ・プログラムが、 UOW 表と CD 表を結合する必要なしに、ユーザー・コピー・ターゲット表を処理できようにするために、CD 表に含められているものです。CD 表と UOW 表の結合が必要な場合は、IBMSNAP_COMMITSEQ 列を使用して結合が行われます。
IBMSNAP_INTENTSEQ	変更 (挿入、更新、または削除) のログ・レコードのログ・シーケンス番号。この値はグローバルに昇順です。更新を削除と挿入のペアとして処理するようにユーザーが選択した場合には、削除行の IBMSNAP_INTENTSEQ 値は、挿入行の対応する値よりも、わずかに小さくなるように作成されます。
IBMSNAP_OPERATION	レコードの操作のタイプを示すフラグ。 **I** 挿入 **U** 更新 **D** 削除
user column after-image	多くの場合、変更後イメージ列には、変更発生後のソース列にある値が含まれます。この列は、ソース列と同じ名前、データ・タイプ、および NULL 属性になります。更新の場合、この列は、更新されたデータの新しい値を反映します。削除の場合、この列は、削除されたデータの値を反映します。挿入の場合、この列は、挿入されたデータの値を反映します。
user column before-image	この列は、ソースが変更前イメージ列の値を含むものとして登録されている場合にのみ CD 表の中に存在します。多くの場合、変更前イメージ列には、変更発生前のソース列にあった値が含まれます。この列は、ソース列と同じ名前になり、IBMSNAP_REGISTER 表の中の BEFORE_IMG_PREFIX 列にある値が接頭部として使用されます。また、データ・タイプもソース列と同じになります。しかし、ソース列が NULL 属性であるかどうかに関係なく、挿入操作ではいつでも NULL 値が許されます。更新の場合、この列は、更新されたデータを反映します。削除の場合、この列は、削除されたデータを反映します。挿入の場合、この列は NULL になります。

IBMQREP_IGNTRAN 表

IBMQREP_IGNTRAN 表は、DB2 リカバリー・ログからキャプチャーしないトランザクションについて Q キャプチャーまたは Q キャプチャー・プログラムに通知するために使用できます。 SQL を使用して、表に行を挿入します。その行は、許可 ID、許可トークン (z/OS のみ)、またはプラン名 (z/OS のみ) に基づいてトランザクションを無視することをプログラムに通知します。

サーバー: Q キャプチャー・サーバー、キャプチャー・コントロール・サーバー

デフォルト・スキーマ: ASN

表 71 では、 IBMQREP_IGNTRAN 表の列の要旨を示します。

表 71. IBMQREP_IGNTRAN 表の列

列名	説明
AUTHID	**データ・タイプ:** CHAR(128)。**NULL 可能:** 可 無視するトランザクションの 1 次許可 ID。
AUTHTOKEN	**データ・タイプ:** CHAR(30)。**NULL 可能:** 可 z/OS 無視するトランザクションの許可トークン (ジョブ名)。
PLANNAME	**データ・タイプ:** CHAR(8)。**NULL 可能:** 可 z/OS 無視するトランザクションのプラン名。
IGNTRANTRC	**データ・タイプ:** CHAR(1)。**NULL 可能:** デフォルトでは不可 IBMQREP_IGNTRAN 表で指定された AUTHID、AUTHTOKEN、または PLANNAME 値に基づいて無視されたトランザクションをトレースするかどうかを、Q キャプチャーまたはキャプチャー・プログラムに指示するフラグ。 **Y (デフォルト)** トレースは有効です。トランザクションが無視されるたびに、IBMQREP_IGNTRANTRC 表に 1 行が挿入され、メッセージが発行されます。 **N** トレースは無効です。

IBMQREP_IGNTRANTRC 表

IBMQREP_IGNTRANTRC 表は、無視するように指定されたトランザクションに関する情報を記録します。

サーバー: Q キャプチャー・サーバー、キャプチャー・コントロール・サーバー

デフォルト・スキーマ: ASN

重要: SQL を使用してこの表を変更しないでください。この表の変更の方法が不適切であると、予期せぬ結果が生じたり、データが失われたりします。

DB2 リカバリー・ログ内のトランザクションが無視されると、IBMQREP_IGNTRANTRC 表に行が挿入されます。この表は Q キャプチャーまたはキャプチャー・プログラムの **trace_limit** パラメーターに従って整理されます。

表 72 では、IBMQREP_IGNTRANTRC 表の列の要旨を示します。

表 72. IBMQREP_IGNTRANTRC 表の列

列名	説明
IGNTRAN_TIME	**データ・タイプ:** TIMESTAMP。**NULL 可能:** デフォルトでは不可 トランザクションが無視された時刻。デフォルト: 現在のタイム・スタンプ
AUTHID	**データ・タイプ:** CHAR(128)。**NULL 可能:** 可 無視されたトランザクションの 1 次許可 ID。
AUTHTOKEN	**データ・タイプ:** CHAR(30)。**NULL 可能:** 可 z/OS 無視されたトランザクションの許可トークン (ジョブ名)。
PLANNAME	**データ・タイプ:** CHAR(8)。**NULL 可能:** 可 z/OS 無視されたトランザクションのプラン名。
TRANSID	**データ・タイプ:** CHAR(10) FOR BIT DATA。**NULL 可能:** 不可 無視されたトランザクションのトランザクション ID。
COMMITLSN	**データ・タイプ:** CHAR(10) FOR BIT DATA。**NULL 可能:** 不可 無視されたトランザクションのコミット・ログ・シーケンス番号または時刻シーケンス。

IBMSNAP_PARTITIONINFO 表

IBMSNAP_PARTITIONINFO 表は、複数のパーティション化された環境において IBMSNAP_RESTART 表を補強し、各パーティションのログ・ファイルのセットのうちの、必要とされる最も古いログ・シーケンスからキャプチャー・プログラムを再始動するための情報を含んでいます。

サーバー: キャプチャー・コントロール・サーバー

デフォルト・スキーマ: ASN

索引: PARTITIONID, USAGE

重要: SQL を使用してこの表を更新するときには、注意してください。この表の変更の方法が不適切であると、予期せぬ結果が生じたり、データが失われたりします。ユーザーがこの表から行を削除した場合、キャプチャー・プログラムはコールド・スタートせざるをえません。

複数パーティション環境では、IBMSNAP_PARTITIONINFO 表と IBMSNAP_RESTART 表が、バージョン 7 以前の SQL レプリケーションの IBMSNAP_WARM_START 表の代わりに使用されます。パーティションが追加されるごとに、この表に行が挿入されます。キャプチャー・プログラムは、最初のデータベース CONNECT の発行後に DB2 が使用した最初のログ・シーケンス番号から、新規パーティションのログ・ファイルの読み取りを開始します。

キャプチャー・プログラムをまだ開始したことがない場合は、この表は空であるため、キャプチャー・プログラムはコールド・スタートを実行する必要があります。

表 73 では、IBMSNAP_PARTITIONINFO 表の列の要旨を示します。

表 73. IBMSNAP_PARTITIONINFO 表の列

列名	説明
PARTITIONID	**データ・タイプ**: INT。**NULL 可能**: 不可 各有効なパーティションのパーティション ID。
USAGE	**データ・タイプ**: CHAR(1)。**NULL 可能**: 不可 ログ・シーケンス番号 (LSN) の使用状況。この列の「R」は、LSN が再始動していることを示します。
SEQUENCE	**データ・タイプ**: CHAR(10) FOR BIT DATA。**NULL 可能**: 不可 パーティション ID を持つノードの再始動 LSN。
STATUS	**データ・タイプ**: CHAR(1)。**NULL 可能**: 可 パーティションの状況。この列の A は、パーティションがアクティブであることを示します。この列は将来の利用のために予約されています。
LAST_UPDATE	**データ・タイプ**: TIMESTAMP。**NULL 可能**: 可 パーティション ID を持つノードの再始動 LSN が最後に更新されたときのタイム・スタンプ。

IBMSNAP_PRUNCNTL 表

整理コントロール表には、このキャプチャー・スキーマに対して定義されている、すべてのサブスクリプション・セット・メンバーに関する詳細情報が入っています。この表は、整理時に IBMSNAP_PRUNE_SET 表と一緒に使用されます。また、アプライ・プログラムとキャプチャー・プログラムの間で初期化ハンドシェーク・プロセス時にも使用されます。

サーバー: キャプチャー・コントロール・サーバー

デフォルト・スキーマ: ASN

索引:
SOURCE_OWNER、SOURCE_TABLE、SOURCE_VIEW_QUAL、APPLY_QUAL、SET_NAME、TARGET_SERVER、TARGET_TABLE、TARGET_OWNER

重要: SQL を使用してこの表を更新するときには、注意してください。この表の変更の方法が不適切であると、予期せぬ結果が生じたり、データが失われたりします。

DB2 ソースの場合は、prune コマンドを発行して整理を呼び出すこともできますし、自動的に起動することもできます。DB2 以外のリレーショナル・ソースの場合の整理は、ユーザーがソースを登録したときに作成された整理トリガーを使用して行われます。

445 ページの表 74 では、IBMSNAP_PRUNCNTL 表の列の要旨を示します。

表 74. IBMSNAP_PRUNCNTL 表の列

列名	説明
TARGET_SERVER	**データ・タイプ**: CHAR(18)。**NULL 可能**: 不可 このメンバーのターゲット表またはビューが置かれているサーバー名。
TARGET_OWNER	**データ・タイプ**: VARCHAR(30)、DB2 UDB for z/OS バージョン 8 新機能モードの場合 VARCHAR(128)。**NULL 可能**: 不可 このメンバーのターゲット表またはビューの上位修飾子。
TARGET_TABLE	**データ・タイプ**: VARCHAR(128)、DB2 UDB for z/OS バージョン 8 互換モード・サブシステムかそれ以前の場合 VARCHAR(18)。 **NULL 可能**: 不可。 このメンバーのターゲット表またはビューの名前。
SYNCHTIME	**データ・タイプ**: TIMESTAMP。**NULL 可能**: 可 キャプチャー・プログラムは、アプライ・プログラムとの初期化ハンドシェーク・プロセス時にこのタイム・スタンプを設定します。この値は、CAPSTART シグナル挿入のトランザクションと関連付けられたコミット・ログ・レコードのタイム・スタンプによって決まります。この値は、その後に初期化処理が行われないかぎり、再度更新されることはありません。
SYNCHPOINT	**データ・タイプ**: CHAR(10) FOR BIT DATA。**NULL 可能**: 可 キャプチャー・プログラムは、アプライ・プログラムとの初期化ハンドシェーク・プロセス時にこの値を設定します。この値は、CAPSTART シグナル挿入のトランザクションと関連付けられたコミット・ログ・レコードのログ・シーケンス番号によって決まります。この値は、その後に初期化処理が行われないかぎり、再度更新されることはありません。
SOURCE_OWNER	**データ・タイプ**: VARCHAR(30)、DB2 UDB for z/OS バージョン 8 新機能モードの場合 VARCHAR(128)。**NULL 可能**: 不可 このメンバーのソース表またはビューの上位修飾子。
SOURCE_TABLE	**データ・タイプ**: VARCHAR(128)、DB2 UDB for z/OS バージョン 8 互換モード・サブシステムかそれ以前の場合 VARCHAR(18)。 **NULL 可能**: 不可。 このメンバーのソース表またはビューの名前。
SOURCE_VIEW_QUAL	**データ・タイプ**: SMALLINT。**NULL 可能**: 不可 この列は、SOURCE_OWNER および SOURCE_TABLE 列に同じ値を持つ、複数の異なるソース・ビューがある場合の複数登録をサポートするために使用されます。この値は、ソースとして定義されている物理表の場合は 0 に、ソースとして定義されているビューの場合は 0 より大きい値に設定されます。
APPLY_QUAL	**データ・タイプ**: CHAR(18)。**NULL 可能**: 不可 このメンバーを処理しているアプライ・プログラムを識別するためのアプライ修飾子。
SET_NAME	**データ・タイプ**: CHAR(18)。**NULL 可能**: 不可 このサブスクリプション・セット・メンバーが所属するサブスクリプション・セットの名前。

表 74. IBMSNAP_PRUNCNTL 表の列 (続き)

列名	説明
CNTL_SERVER	**データ・タイプ**: CHAR(18)。**NULL 可能**: 不可 APPLY_QUAL により識別される、このアプライ・プログラムのアプライ・コントロール表が置かれているサーバーの名前。
TARGET_STRUCTURE	**データ・タイプ**: SMALLINT。**NULL 可能**: 不可 ターゲット表またはビューのタイプを示す値。 **1** ソース表 **3** CCD 表 **4** ポイント・イン・タイム表 **5** 基礎集約表 **6** 変更集約表 **7** レプリカ表 **8** ユーザー・コピー表 **9** IBMSNAP_UOW と CD 表の結合のない CCD 表
CNTL_ALIAS	**データ・タイプ**: CHAR(8)。**NULL 可能**: 可 CNTL_SERVER 列で指定されているアプライ・コントロール・サーバーに対応する DB2 別名。
PHYS_CHANGE_OWNER	**データ・タイプ**: VARCHAR(30)、DB2 UDB for z/OS バージョン 8 新機能モードの場合 VARCHAR(128)。**NULL 可能**: 可 この特定のサブスクリプション・セット・メンバーのソースに関連付けられた IBMSNAP_REGISTER 表の PHYS_CHANGE_OWNER 列の中の値。
PHYS_CHANGE_TABLE	**データ・タイプ**: VARCHAR(128)、DB2 UDB for z/OS バージョン 8 互換モード・サブシステムかそれ以前の場合 VARCHAR(18)。**NULL 可能**: 可。 この特定のサブスクリプション・セット・メンバーのソースに関連付けられた IBMSNAP_REGISTER 表の PHYS_CHANGE_TABLE 列の中の値。
MAP_ID	**データ・タイプ**: VARCHAR(10)。**NULL 可能**: 不可 この表で、より短く、簡単に使用できる索引を提供するユニークな因子。シグナル表への CAPSTART 挿入を、整理コントロール表の中の適切な行に関連付けるためにも使用されます。

IBMSNAP_PRUNE_LOCK 表

IBMSNAP_PRUNE_LOCK 表は、コールド・スタート、または保存限度整理時に、CD 表へのアクセスをシリアライズするために使用されます。この表は、これらの重要なフェーズでアプライ・プログラムが CD 表にアクセスしないようにするものです。この表には行がありません。

サーバー: キャプチャー・コントロール・サーバー

デフォルト・スキーマ: ASN

索引: なし

重要: SQL を使用してこの表を更新するときには、注意してください。この表の変更の方法が不適切であると、予期せぬ結果が生じたり、データが失われたりします。

IBMSNAP_PRUNE_SET 表

IBMSNAP_PRUNE_SET 表は、各サブスクリプション・セットについてキャプチャー・プログラムおよびアプライ・プログラムの進行をトラッキングすることにより、CD 表および UOW 表の整理の調整を行います。1 つのソースからターゲットへのマッピングに対して 1 つの行がある IBMSNAP_PRUNCNTL 表とは異なり、IBMSNAP_PRUNE_SET 表には、1 つのサブスクリプション・セットに対して 1 つの行があります。

サーバー: キャプチャー・コントロール・サーバー

デフォルト・スキーマ: ASN

索引: TARGET_SERVER、APPLY_QUAL、SET_NAME

重要: SQL を使用してこの表を更新するときには、注意してください。この表の変更の方法が不適切であると、予期せぬ結果が生じたり、データが失われたりします。

表 75 では、IBMSNAP_PRUNE_SET 表の列の要旨を示します。

表 75. IBMSNAP_PRUNE_SET 表の列

列名	説明
TARGET_SERVER	**データ・タイプ**: CHAR(18)。**NULL 可能**: 不可 このセットのターゲット表またはビューが置かれているサーバー名。
APPLY_QUAL	**データ・タイプ**: CHAR(18)。**NULL 可能**: 不可 このセットを処理しているアプライ・プログラムを識別するためのアプライ修飾子。
SET_NAME	**データ・タイプ**: CHAR(18)。**NULL 可能**: 不可 サブスクリプション・セットの名前。
SYNCHTIME	**データ・タイプ**: TIMESTAMP。**NULL 可能**: 可 アプライ・プログラムはこの列を使用して進行状況を記録します。サブスクリプション・セットのデータの処理が、このタイム・スタンプまで終了していることを示します。
SYNCHPOINT	**データ・タイプ**: CHAR(10) FOR BIT DATA。**NULL 可能**: 不可 アプライ・プログラムはこの列を使用して進行状況を記録します。サブスクリプション・セットのデータの処理が、この同期点の値まで終了していることを示します。

IBMSNAP_REG_EXT (System i)

IBMSNAP_REG_EXT 表は、IBMSNAP_REGISTER 表の補足情報を提供する System i 固有の表です。IBMSNAP_REG_EXT 表には、IBMSNAP_REGISTER 表のすべての行に対して対応する行があり、さらにこの表には System i 固有の列がいくつか追加で含まれています。

サーバー: キャプチャー・コントロール・サーバー

デフォルト・スキーマ: ASN

索引: VERSION、SOURCE_OWNER、SOURCE_TABLE、SOURCE_VIEW_QUAL

重要: SQL を使用してこの表を更新するときには、注意してください。この表の変更の方法が不適切であると、予期せぬ結果が生じたり、データが失われたりします。

この表は、IBMSNAP_REGISTER 表のトリガー・プログラム (ライブラリー QDP4 の QZSNJLV8) によって保守されます。トリガーは、IBMSNAP_REGISTER 表が作成されるときに定義されます。

この表の情報は、System i サーバー上でレプリケーション・ソースがどこでどのように定義されているかをトラッキングするために使用されます。

表 76 では、IBMSNAP_REG_EXT 表の列の要旨を示します。

表 76. IBMSNAP_REG_EXT 表の列

列名	説明
VERSION	**データ・タイプ**: INT。**NULL 可能**: 不可 ソースの登録に使用された DB2 DataPropagator for System i のバージョン。
SOURCE_OWNER	**データ・タイプ**: VARCHAR(30)。**NULL 可能**: 不可 ユーザーにより登録されたソース表またはビューの上位修飾子。
SOURCE_TABLE	**データ・タイプ**: VARCHAR(128)。**NULL 可能**: 不可 ユーザーにより登録されたソース表またはビューの名前。
SOURCE_NAME	**データ・タイプ**: CHAR(10)。**NULL 可能**: 可 コマンドの発行に使用されたソース表またはビューの 10 文字のシステム名。
SOURCE_MBR	**データ・タイプ**: CHAR(10)。**NULL 可能**: 可 ジャーナル項目の受信 (RCVJRNE) コマンドの発行および ALIAS サポートに使用されるソース表メンバーの名前。
SOURCE_TABLE_RDB	**データ・タイプ**: CHAR(18)。**NULL 可能**: 可 リモート・ジャーナルを使用する場合、この列には、ソース表が実際に置かれているシステムのデータベース名が含まれます。 ローカル・ジャーナルの場合、この列は NULL になります。
JRN_LIB	**データ・タイプ**: CHAR(10)。**NULL 可能**: 可 ソース表が使用するジャーナルのライブラリー名。

表 76. IBMSNAP_REG_EXT 表の列 (続き)

列名	説明
JRN_NAME	**データ・タイプ**: CHAR(10)。**NULL 可能**: 可 ソース表が使用するジャーナルの名前。この列で、アスタリスクの後ろに 9 つのブランクが続くときには、ソース表が現在ジャーナルの中にないことを意味します。この場合キャプチャー・プログラムは、このソースのデータをキャプチャーすることはできません。
FR_START_TIME	**データ・タイプ**: TIMESTAMP。**NULL 可能**: 可 アプライ・プログラムがフル・リフレッシュの実行を開始した時刻。
SOURCE_VIEW_QUAL	**データ・タイプ**: SMALLINT。**NULL 可能**: 不可 登録表内の類似した列を突き合わせることによって、サブスクリプションのビューをサポートします。この値は、ソースとして定義されている物理表の場合は 0 に、ソースとして定義されているビューの場合は 0 より大きい値に設定されます。 SOURCE_OWNER および SOURCE_TABLE 列の値が同じである、複数の異なるソース・ビューについて複数のサブスクリプションをサポートするには、この列が必要です。
CMT_BEHAVIOR_CASE	**データ・タイプ**: SMALLINT。**NULL 可能**: 不可。デフォルトあり。**デフォルト**: 0 ソース表を更新するアプリケーション・プログラムがコミットメント・コントロールを使用する方法を表す整数。キャプチャー・プログラムはこの値を使用して、構成済みであるが、CD 表にまだ書き込んでいない CD 行に使用するメモリーを管理します。 **-1** アプリケーションのコミットメント・コントロール・パターンはまだ設定されていません。これは、この列の初期値です。 **0** ソースを更新するアプリケーションではコミットメント・コントロールが使用されていません。 **1** ソースを更新するアプリケーションはすべて、コミットメント・コントロールを使用します。このため、コミットメント・コントロール下の同じソース表が、2 つの異なるアプリケーションから同時に更新されることはありません。 **2** ソースを更新する同時アプリケーションの中にはコミットメント・コントロールを使用するものも使用しないものもあります。2 つのアプリケーションが同時にコミットメント・コントロールを使用してソース表を更新する可能性があります。
MAX_ROWS_BTWN_CMTS	**データ・タイプ**: SMALLINT。**NULL 可能**: 不可。デフォルトあり。**デフォルト**: 0 キャプチャー・プログラムがデータを CD 表にコミットする前に処理できる行の最大数。

IBMSNAP_REGISTER 表

IBMSNAP_REGISTER 表は、レプリケーション・ソース表の名前、属性、および、これらに関連付けられた CD 表および CCD 表の名前など、レプリケーション・ソ

ースに関する情報を保持します。キャプチャー・プログラムで処理されるように、新しいレプリケーション・ソース表またはビューが定義されるたびに、この表に行が 1 つ自動的に挿入されます。

サーバー: キャプチャー・コントロール・サーバー

デフォルト・スキーマ: ASN

索引: SOURCE_OWNER、SOURCE_TABLE、SOURCE_VIEW_QUAL

重要: SQL を使用してこの表を更新するときには、注意してください。この表の変更の方法が不適切であると、予期せぬ結果が生じたり、データが失われたりします。

レプリケーション・ソースの定義を調べる必要があるときには、登録表を使用します。

表 77 では、IBMSNAP_REGISTER 表の列の要旨を示します。

表 77. IBMSNAP_REGISTER 表の列

列名	説明
SOURCE_OWNER	**データ・タイプ**: VARCHAR(30)、DB2 UDB for z/OS バージョン 8 新機能モード・サブシステムの場合 VARCHAR(128)。**NULL 可能**: 不可 ユーザーにより登録されたソース表またはビューの上位修飾子。
SOURCE_TABLE	**データ・タイプ**: VARCHAR(128)、DB2 UDB for z/OS バージョン 8 互換モード・サブシステムかそれ以前の場合 VARCHAR(18)。**NULL 可能**: 不可。 ユーザーにより登録されたソース表またはビューの名前。
SOURCE_VIEW_QUAL	**データ・タイプ**: SMALLINT。**NULL 可能**: 不可 この列は、SOURCE_OWNER および SOURCE_TABLE 列に同じ値を持つ、複数の異なるソース・ビューがある場合の複数登録をサポートするために使用されます。この値は、ソースとして定義されている物理表の場合は 0 に、ソースとして定義されているビューの場合は 0 より大きい値に設定されます。
GLOBAL_RECORD	**データ・タイプ**: CHAR(1)。**NULL 可能**: 不可
SOURCE_STRUCTURE	**データ・タイプ**: SMALLINT。**NULL 可能**: 不可 ソース表またはビューの構造を示す値 **1** ユーザー表 **3** CCD 表 **4** ポイント・イン・タイム表 **5** 基礎集約表 **6** 変更集約表 **7** レプリカ表 **8** ユーザー・コピー表 **9** IBMSNAP_UOW と CD 表の結合のない CCD 表

表 77. IBMSNAP_REGISTER 表の列 (続き)

列名	説明
SOURCE_CONDENSED	**データ・タイプ**: CHAR(1)。**NULL 可能**: 不可 ソース表がコンデンス表であるかどうかを示すフラグ。圧縮されている場合は、同じキーを持つすべての行が、1 つの行に圧縮されます。 **Y** ソースは圧縮されています。 **N** ソースは圧縮されていません。 **A** ソースは基礎集約表または変更集約表です。
SOURCE_COMPLETE	**データ・タイプ**: CHAR(1)。**NULL 可能**: 不可 主キー値を持つ行がソース表にどのように保管されるかを示すフラグ。 **Y** ソース表は、関係するそれぞれの主キー値につき 1 行を保持します。 **N** ソース表は、主キー値の行のサブセットを保持します。
CD_OWNER	**データ・タイプ**: DB2 UDB for z/OS バージョン 8 新機能モード・サブシステムの場合 VARCHAR(30)。**NULL 可能**: 可 ソースの CD 表の上位修飾子。 **ソースが表の場合** 外部 CCD 表ではないすべての登録済みソース表の場合、この列は、このソース表に関連付けられた CD 表の上位修飾子を保持します。 **ソースがビューの場合** この列は、CD ビューの上位修飾子を保持します。 **ソースが外部 CCD 表の場合** この列は NULL です。
CD_TABLE	**データ・タイプ**: VARCHAR(128)、DB2 UDB for z/OS バージョン 8 互換モード・サブシステムかそれ以前の場合 VARCHAR(18)。**NULL 可能**: 可。 ソースの CD 表の名前。 **ソースが表の場合** 外部 CCD 表ではないすべての登録済みソース表の場合、この列には、このソース表のキャプチャーされた更新を保留する CD 表の名前が入ります。 **ソースがビューの場合** この列は、CD ビューの名前を保持します。 **ソースが外部 CCD 表の場合** この列は NULL です。

表 77. IBMSNAP_REGISTER 表の列 (続き)

列名	説明
PHYS_CHANGE_OWNER	**データ・タイプ**: VARCHAR(30)、DB2 UDB for z/OS バージョン 8 新機能モード・サブシステムの場合 VARCHAR(128)。**NULL 可能**: 可 アプライ・プログラムが変更キャプチャー・レプリケーションに使用する、表またはビューの上位修飾子。 **ソースが表の場合** 外部 CCD 表ではないすべての登録済みソース表の場合、この列は、このソース表に関連付けられた物理 CD 表の上位修飾子を保持します。 **ソースがビューの場合** この列は、このソース・ビューに関連付けられた物理 CD 表の上位修飾子を保持します。 **ソースが外部 CCD 表の場合** この列は、外部 CCD 表の上位修飾子を保持します。
PHYS_CHANGE_TABLE	**データ・タイプ**: VARCHAR(128)、DB2 UDB for z/OS バージョン 8 互換モード・サブシステムかそれ以前の場合 VARCHAR(18)。**NULL 可能**: 可。 アプライ・プログラムが変更キャプチャー・レプリケーションに使用する、表またはビューの名前。 **ソースが表の場合** 外部 CCD 表ではないすべての登録済みソース表の場合、この列は、このソース表に関連付けられた物理 CD 表の名前を保持します。 **ソースがビューの場合** この列は、このソース・ビューに関連付けられた物理 CD 表の名前を保持します。 **ソースが外部 CCD の場合** この列は、外部 CCD 表の名前を保持します。
CD_OLD_SYNCHPOINT	**データ・タイプ**: CHAR(10) FOR BIT DATA。**NULL 可能**: 可 この列は、アプライ・プログラムとキャプチャー・プログラムの間の初期ハンドシェークに使用されます。キャプチャー・プログラムはその後、ソース・ログのこのログ・シーケンス番号からデータのキャプチャーを開始します。この列は、CD 表で保存限度整理が行われたことを示すためにも使用されます。この値が NULL の場合、登録は非アクティブです。
CD_NEW_SYNCHPOINT	**データ・タイプ**: CHAR(10) FOR BIT DATA。**NULL 可能**: 可 この列は、キャプチャー・プログラムが CD 表に新しい行を挿入するにつれて進められます。アプライ・プログラムはこの列を使用して、複製する新しい変更があるかどうかを確認します。
DISABLE_REFRESH	**データ・タイプ**: SMALLINT。**NULL 可能**: 可 フル・リフレッシュが使用可能かどうかを示すフラグ。 **0** フル・リフレッシュを使用可能にします。 **1** フル・リフレッシュを使用不可にします。

表 77. IBMSNAP_REGISTER 表の列 (続き)

列名	説明
CCD_OWNER	**データ・タイプ**: VARCHAR(30)、DB2 for z/OS バージョン 8 新機能モード・サブシステムの場合 VARCHAR(128)。**NULL 可能**: 可 ソースに内部 CCD 表が関連付けられている場合、この列は、内部 CCD の上位修飾子を保持します。外部 CCD 表の場合、この列は NULL になります。
CCD_TABLE	**データ・タイプ**: VARCHAR(128)、DB2 for z/OS バージョン 8 互換モード・サブシステムかそれ以前の場合 VARCHAR(18)。**NULL 可能**: 可。 ソースに内部 CCD 表が関連付けられている場合、この列は、内部 CCD の名前を保持します。外部 CCD 表の場合、この列は NULL になります。
CCD_OLD_SYNCHPOINT	**データ・タイプ**: CHAR(10) FOR BIT DATA。**NULL 可能**: 可 CCD 表が再初期化されたときのログ・シーケンス番号。この列は、CCD のフル・リフレッシュ処理に関連しています。この列の値を変更する必要があるのは、CCD 表が初期に、またはその後に、フル・リフレッシュされた場合だけです。この値は、CCD 表に残っているどの行よりも古いものである可能性があります。この列が保守されないと、 CCD 表をレプリケーション・ソースとして使用するアプライ・プログラムは、 CCD 表が再初期化されたかどうかが分からないので、 CCD ソースの完全なコピーを再初期化できません。
SYNCHPOINT	**データ・タイプ**: CHAR(10) FOR BIT DATA。**NULL 可能**: 可 グローバル行の場合 (GLOBAL_RECORD = Y)、同期点は、キャプチャー・プログラムによって処理された最後のログまたはジャーナル・レコードのログ・シーケンス番号を表します。CCD 表 (内部または外部) に関する登録情報を含む IBMSNAP_REGISTER 表の中の行の場合、CCD 表の中に使用可能な新しいデータがあることを示すために、同期点の値は、CCD 表を保守するプログラムによって進められます。
SYNCHTIME	**データ・タイプ**: TIMESTAMP。**NULL 可能**: 可 グローバル行の場合 (GLOBAL_RECORD = Y)、同期時刻は、キャプチャー・プログラムによって処理された最後のログまたはジャーナル・レコードのタイム・スタンプを表します。 キャプチャー・プログラムが DB2 ログの最後に達すると、同期時刻は現在の DB2 タイム・スタンプまで進められます。CCD 表 (内部または外部) に関する登録情報を含む IBMSNAP_REGISTER 表の中の行の場合、CCD 表の中の使用可能データの現行性 (currency) を示すために、同期時刻の値は、CCD 表を保守するプログラムによって進められます。
CCD_CONDENSED	**データ・タイプ**: CHAR(1)。**NULL 可能**: 可 このソースに関連付けられた内部 CCD が圧縮されているかどうかを示すフラグ。圧縮されている場合は、同じキーを持つすべての行が、1 つの行に圧縮されます。 **Y** 内部 CCD は圧縮されています。 **N** 内部 CCD は圧縮されていません。 **NULL** このソースに対して内部 CCD 表が定義されていません。

表 77. IBMSNAP_REGISTER 表の列 (続き)

列名	説明
CCD_COMPLETE	**データ・タイプ**: CHAR(1)。**NULL 可能**: 可 このソースに関連付けられた内部 CCD 表が完成しているかどうかを示すフラグ。完成しているとは、ソース表からのすべての行が初期に含まれていることを意味します。 **N** 内部 CCD は完成していません。 **NULL** このソースに対して内部 CCD 表が定義されていません。
ARCH_LEVEL	**データ・タイプ**: CHAR(4)。**NULL 可能**: 不可 レプリケーション・コントロール表の構造レベルは以下のとおりです。 **0801** バージョン 8 SQL レプリケーション **0803** バージョン 8 SQL レプリケーション (Oracle ソースの拡張サポート付き) **0805** バージョン 8 SQL レプリケーション (DB2 for z/OS 新機能モードのサポート付き)
DESCRIPTION	**データ・タイプ**: CHAR(254)。**NULL 可能**: 可 レプリケーション・ソースの記述。
BEFORE_IMG_PREFIX	**データ・タイプ**: VARCHAR(4)。**NULL 可能**: 可 CD 表の中の変更前イメージ列の名前を識別する 1 文字の接頭部。変更前イメージの接頭部と CD 列名の組み合わせは判別可能なものにする必要があります。つまり、接頭部の付いた CD 列名は、現在の列名、そして想定される変更後イメージ列の名前と同じであってはなりません。BEFORE_IMG_PREFIX の長さ (バイト単位) は次のとおりです。 **1** ASCII または EBCDIC の 1 バイトの接頭文字の場合。 **2** ASCII の 2 バイトの接頭文字の場合。 **4** EBCDIC DBCS 接頭文字の場合。この長さは、シフトインおよびシフトアウト文字を見込んでいます。

表 77. IBMSNAP_REGISTER 表の列 (続き)

列名	説明
CONFLICT_LEVEL	**データ・タイプ**: CHAR(1)。**NULL 可能**: 可 このソースの競合検出のレベルを示すフラグ。 **0** アプライ・プログラムは競合をチェックしません。更新の競合を防止するために、アプリケーション側でデータ整合性を適用する必要があります。 **1** カスケード・トランザクション・リジェクトの標準検出。アプライ・プログラムは、これまでにキャプチャーされた変更に基づいて競合をチェックします。アプライ・プログラムは、レプリカに競合するトランザクションがあればそれを元に戻し、競合するトランザクションに対して従属関係を持つトランザクションがあれば、それも元に戻します。アプライ・プログラムが競合検出を開始した後でキャプチャーされた変更は、このアプライ・サイクルではチェックされません。 **2** カスケード・トランザクション・リジェクトの拡張検出。アプライ・プログラムは、キャプチャー・プログラムがログまたはジャーナルからすべての変更をキャプチャーするまで待ち (SYNCHTIME 列の記述を参照)、1 に設定された場合と同じように、標準競合検出を行います。アプライ・プログラムはエンキュー中、競合検出プロセス中に変更が行われないように、ソース表にロックをかけます。
CHG_UPD_TO_DEL_INS	**データ・タイプ**: CHAR(1)。**NULL 可能**: 可 キャプチャー・プログラムが CD 表に更新を保管する方法を示すフラグ。 **Y** キャプチャー・プログラムは、1 つは削除用、1 つは挿入用の 2 つの行を CD 表で使用して更新を保管します。アプライ・プログラムは最初に削除を処理し、次に挿入を処理します。このフラグを Y に設定すると、レプリケーション・ソースに対するすべての更新は、 2 つの行を使用して CD 表に保管されます。このフラグは、パーティション化された列、またはサブスクリプション・セット述部から参照される列の更新が正しく処理されるようにします。 **N** ソース表に対する更新はそれぞれ、CD 表の中の 1 つの行に保管されます。
CHGONLY	**データ・タイプ**: CHAR(1)。**NULL 可能**: 可 キャプチャー・プログラムがソースで発生したすべての変更をキャプチャーするか、登録済み列で発生した変更のみをキャプチャーするかを示すフラグ。一般的には、キャプチャー・プログラムによって CD 表に挿入される行数を最小化するために、このオプションは Y に設定されます。しかし、ソース表の中のどの行が更新されたかを正確にトラッキングするために、このオプションを N に設定することも考えられます。例えば、ソース表の中のどの行が更新されたかを監査するために、主キー列の値だけをキャプチャーできます。 **Y** キャプチャー・プログラムは、ソース表の登録済み列で発生した変更のみをキャプチャーします。 **N** キャプチャー・プログラムは、ソース表すべての列で発生した変更をキャプチャーします。

表 77. IBMSNAP_REGISTER 表の列 (続き)

列名	説明
RECAPTURE	**データ・タイプ**: CHAR(1)。**NULL 可能**: 可 この列は Update-anywhere レプリケーションに使用されます。表またはビューで行われた変更の再キャプチャーを行い、他の表またはビューに転送するかどうかを示すフラグを保持します。 マスター側の表の場合: **N** レプリカから適用されたマスターに対する更新の再キャプチャーは行われず、他のレプリカに複製されません。 **Y** レプリカから適用されたマスターに対する更新は他のレプリカに複製されます。 レプリカ側の表の場合: **Y** マスターから適用されたレプリカに対する更新の再キャプチャーが行われ、このレプリカをソースとして使用する他の表に複製するために使用できます。 **N** マスターから適用されたレプリカに対する更新の再キャプチャーは行われません。
OPTION_FLAGS	**データ・タイプ**: CHAR(4)。**NULL 可能**: 不可 SQL レプリケーションの将来のオプション用に予約済み。この列には現在、デフォルト値の NNNN が入っています。
STOP_ON_ERROR	**データ・タイプ**: CHAR(1)。**NULL 可能**: 可。デフォルトあり。**デフォルト:** Y。 キャプチャー・プログラムが、始動、開始、再開始、または CD 表への行の挿入時にエラーを検出したときに、終了するか、登録の処理を停止するだけかを示すフラグ。 **Y** キャプチャー・プログラムは、始動、開始、再開始、または CD 表への行の挿入時にエラーを検出すると終了します。 **N** キャプチャー・プログラムは、始動、再初期化、または CD 表への行の挿入時にエラーを検出すると、登録を停止しますが、終了はしません。プログラムはその他の登録処理を続けます。
STATE	**データ・タイプ**: CHAR(1)。**NULL 可能**: 可。デフォルトあり。**デフォルト:** I。 登録の状態を示すフラグ。 **S** キャプチャー・プログラムはこの登録の処理を停止しました。アプライ・プログラムは、登録が修復され、I (非アクティブ) 状態になるまで、この登録を処理しません。 **A** 登録はアクティブです。 **I** 登録は非アクティブです。
STATE_INFO	**データ・タイプ**: CHAR(8)。**NULL 可能**: 可。 キャプチャー・プログラムが登録の処理を停止した場合、この列は、この障害に関して発行されたエラー・メッセージを保持します。

IBMSNAP_REG_SYNCH 表 (DB2 以外のリレーショナル)

IBMSNAP_REG_SYNCH 表は、アプライ・プログラムが DB2 以外のリレーショナル・データ・ソースからデータをフェッチする準備をするときに、IBMSNAP_REGISTER 表の中のすべての行の SYNCHPOINT 値の更新を開始するために、更新トリガーを使用します。

サーバー: キャプチャー・コントロール・サーバー

デフォルト・スキーマ: ASN

索引: TRIGGER_ME

重要: SQL を使用してこの表を更新するときには、注意してください。この表の変更の方法が不適切であると、予期せぬ結果が生じたり、データが失われたりします。

表 78 では、IBMSNAP_REG_SYNCH 表の列の要旨を示します。

表 78. IBMSNAP_REG_SYNCH 表の列

列名	説明
TRIGGER_ME	**データ・タイプ**: CHAR(1)。**NULL 可能**: 不可 登録表の中のすべての行の SYNCHPOINT 値を更新するためにトリガーが開始されたかどうかを示す Y というフラグ。
TIMESTAMP	Microsoft SQL Server および Sybase のソースの場合、この列は、表のタイム・スタンプ列で更新が発生したときにシステムから生成されるユニークな番号を保持します。この値は、 IBMSNAP_REGISTER 表に記録される SYNCHPOINT 値を生成するために使用されます。

IBMSNAP_RESTART 表

IBMSNAP_RESTART 表には、必要とされる中で一番古いログまたはジャーナル・レコードからキャプチャー・プログラムを再始動できるようにするための情報が含まれます。この表は、SQL レプリケーションのバージョン 7 およびそれ以前のバージョンの IBMSNAP_WARM_START 表を置き換えるものです。この表は、コミット・ポイントのたびに更新される行を保持します。このため、キャプチャー・プログラムは、すでに処理済みで、CD 表および UOW 表に挿入済みの情報の再キャプチャーを行う必要がなく、常に正しい場所から正確に再始動できます。

サーバー: キャプチャー・コントロール・サーバー

デフォルト・スキーマ: ASN

索引: なし

重要: SQL を使用してこの表を更新するときには、注意してください。この表の変更の方法が不適切であると、予期せぬ結果が生じたり、データが失われたりします。ユーザーがこの表から行を削除した場合、キャプチャー・プログラムはコールド・スタートせざるをえません。

キャプチャー・プログラムをまだ開始したことがない場合は、この表は空であるため、キャプチャー・プログラムはコールド・スタートを実行する必要があります。

以下の 2 つのセクションは、オペレーティング・システム別の IBMSNAP_RESTART 表のレイアウトを示しています。

z/OS、Linux、UNIX、Windows z/OS Linux UNIX Windows

表 79. z/OS、Linux、UNIX、および Windows の場合の IBMSNAP_RESTART 表の列

列名	説明
MAX_COMMITSEQ	**データ・タイプ**: CHAR(10) FOR BIT DATA。**NULL 可能**: 不可 キャプチャー・プログラムが CD 表および UOW 表にコミット済みの論理ログ・シーケンス番号 (IBMSNAP_COMMITSEQ) の最大値。
MAX_COMMIT_TIME	**データ・タイプ**: TIMESTAMP。**NULL 可能**: 不可 MAX_COMMITSEQ 列のログ・シーケンス番号に関連付けられたタイム・スタンプ。
MIN_INFLIGHTSEQ	**データ・タイプ**: CHAR(10) FOR BIT DATA。**NULL 可能**: 不可 キャプチャー・プログラムがウォーム・リスタート時に開始する時点の論理ログ・シーケンス番号。 この値は、キャプチャー・プログラムが見つけた、コミットまたはアボート・レコードがまだ検出されていない、一番若いログ・シーケンス番号です。
CURR_COMMIT_TIME	**データ・タイプ**: TIMESTAMP。**NULL 可能**: 不可 この表がキャプチャー・プログラムによって更新されたときの、ローカルの現行タイム・スタンプ。
CAPTURE_FIRST_SEQ	**データ・タイプ**: CHAR(10) FOR BIT DATA。**NULL 可能**: 不可 最後にコールド・スタートを実行したときにキャプチャー・プログラムの開始時点であったリカバリー・ログに関連付けられた論理ログ・シーケンス番号。 この値は、キャプチャー・プログラムでコールド・スタートを実行せざるをえなくなるようなデータベース RESTORE が発生したかどうかを検出するために使用されます。このような事態は、データベース・ログ・マネージャーが特定の RESTORE 操作時にログ・シーケンス番号を再利用することがあるために発生します。

System i System i

System i の場合、IBMSNAP_RESTART 表は RCVJRNE (ジャーナル項目の受信) コマンドの開始時刻を判別するために使用されます。1 つのレプリケーション・ソース、またはレプリケーション・ソースのグループで使用されるジャーナルごとに、再始動表に行が 1 つ挿入されます。

索引: JRN_LIB、 JRN_NAME

表 80. System i の場合の IBMSNAP_RESTART 表の列

列名	説明
MAX_COMMITSEQ	**データ・タイプ**: CHAR(10) FOR BIT DATA。**NULL 可能**: 不可 UOW 表からの最新のコミットのジャーナル・レコード番号。
MAX_COMMIT_TIME	**データ・タイプ**: TIMESTAMP。**NULL 可能**: 不可 MAX_COMMITSEQ 列のジャーナル・レコード番号に関連付けられたタイム・スタンプ、または、キャプチャー・プログラムがログの処理を終了し、実行する処理がない場合は、現在のタイム・スタンプ。
MIN_INFLIGHTSEQ	**データ・タイプ**: CHAR(10) FOR BIT DATA。**NULL 可能**: 不可 キャプチャー・プログラムがウォーム・リスタート時に開始する時点の論理ログ・シーケンス番号。
CURR_COMMIT_TIME	**データ・タイプ**: TIMESTAMP。**NULL 可能**: 不可 この表が更新された時点の現行タイム・スタンプ。
CAPTURE_FIRST_SEQ	**データ・タイプ**: CHAR(10) FOR BIT DATA。**NULL 可能**: 不可 キャプチャー・プログラムがコールド・スタート後に開始する時点のジャーナル・レコード番号。
UID	**データ・タイプ**: INTEGER。**NULL 可能**: 不可 UOW 表の IBMSNAP_UOWID 列の内容で接頭部として使用されるユニーク番号。
SEQNBR	**データ・タイプ**: BIGINT。**NULL 可能**: 不可 キャプチャー・プログラムが処理した最後のジャーナル項目のシーケンス番号。
JRN_LIB	**データ・タイプ**: CHAR(10)。**NULL 可能**: 不可 キャプチャー・プログラムが処理しているジャーナルのライブラリー名。
JRN_NAME	**データ・タイプ**: CHAR(10)。**NULL 可能**: 不可 キャプチャー・プログラムが処理しているジャーナルの名前。
STATUS	**データ・タイプ**: CHAR(1)。**NULL 可能**: 可 キャプチャー・プログラムが特定のジャーナル・ジョブを処理しているかどうかを示すフラグ。 **Y** キャプチャー・プログラムはジャーナル・ジョブを処理しています。 **N** キャプチャー・プログラムはジャーナル・ジョブを処理していません。

IBMSNAP_SEQTABLE 表 (Informix)

IBMSNAP_SEQTABLE 表は、Informix 表のログ・シーケンス番号と同等のものとして SQL レプリケーションが使用する一連のユニーク番号を保持します。これらのユニーク ID は、キャプチャー・プログラム、アプライ・プログラム、およびレプリケーション・アラート・モニターが最後のサイクル時に終了時点を連絡し合えるように、IBMSNAP_REGISTER 表で同期点値の代わりに使用されます。

サーバー: キャプチャー・コントロール・サーバー

デフォルト・スキーマ: ASN

ユニーク索引: SEQ

重要: SQL を使用してこの表を更新するときには、注意してください。この表の変更の方法が不適切であると、予期せぬ結果が生じたり、データが失われたりします。

表 81 では、IBMSNAP_SEQTABLE 表の列の要旨を示します。

表 81. IBMSNAP_SEQTABLE 表の列

列名	説明
SEQ	**データ・タイプ**: INTEGER。**NULL 可能**: 不可 Informix 表のログまたはジャーナル ID (同期点) として使用されるユニークな番号。

IBMSNAP_SIGNAL 表

シグナル表には、キャプチャー・プログラムに特定のアクションを実行するように促すシグナルが保管されます。シグナルは、ユーザーまたはアプライ・プログラムから入力されます。

サーバー: キャプチャー・コントロール・サーバー

デフォルト・スキーマ: ASN

この表の情報は、SQL を使って更新できます。

IBMSNAP_SIGNAL 表は DATA CAPTURE CHANGES 属性を指定して作成されます。つまり、この表に対して実行されたすべての挿入、更新、および削除操作は、DB2 リカバリー・ログから読み取られたログ・レコードのように、キャプチャー・プログラムから見ることができます。キャプチャー・プログラムは、IBMSNAP_SIGNAL 表の更新および削除ログ・レコードはすべて無視しますが、シグナル挿入の、有効に作成およびコミットされたログ・レコードはすべて、注意が必要な「シグナル」として認識します。シグナル挿入によるログ・レコードに対してキャプチャー・プログラムが実行するアクションは、その挿入に関して IBMSNAP_SIGNAL 表がどのように指定されているかによって異なります。IBMSNAP_SIGNAL 表の中の値は、取るべきアクションをキャプチャー・プログラムに指示します。

この表の中で、コンプリートを表す、SIGNAL_STATE 値が C のレコード、または保存限度整理の対象となるタイム・スタンプを持つレコードは、キャプチャー・プログラムによる整理が行われると削除されます。

461 ページの表 82 では、IBMSNAP_SIGNAL 表の列の要旨を示します。

表 82. IBMSNAP_SIGNAL 表の列

列名	説明
SIGNAL_TIME	**データ・タイプ**: TIMESTAMP。**NULL 可能**: デフォルトでは不可。**デフォルト**: 現行タイム・スタンプ。 行を一意的に識別するために使用されるタイム・スタンプ。キャプチャー・プログラムはこのユニーク値を使用して、キャプチャー・シグナルの処理が終了した時刻を示す、シグナル表の中の正しい行を検出します。タイム・スタンプ列は NOT NULL WITH DEFAULT として作成されるため、キャプチャー・シグナルは一般的に、DB2 で現行タイム・スタンプが SIGNAL_TIME 値として提供されるのと同じ方法で挿入できます。
SIGNAL_TYPE	**データ・タイプ**: VARCHAR(30)。**NULL 可能**: 不可 通知されたシグナルのタイプを示すフラグ。 **CMD** ユーザー、アプライ・プログラム、または別のアプリケーション (よく知られたシステム・コマンドやシグナル) から通知されたシグナル。使用可能なシグナルのサブタイプのリストについては、この表の SIGNAL_SUBTYPE 列を参照してください。 **USER** ユーザーから通知されたシグナル。キャプチャー・プログラムは、SIGNAL_LSN 列の値を、シグナルが挿入されたときのログの LSN で更新し、 SIGNAL_STATE 列の値を、P (ペンディング) から R (受信) に更新します。

表 82. IBMSNAP_SIGNAL 表の列 (続き)

列名	説明
SIGNAL_SUBTYPE	**データ・タイプ**: VARCHAR(30)。**NULL 可能**: 可 システム・コマンドからのシグナルが発生したときに (SIGNAL_TYPE = CMD)、キャプチャー・プログラムが実行するアクション。 **CAPSTART** キャプチャー・プログラムは、SIGNAL_INPUT_IN 列の中の MAP_ID (IBMSNAP_PRUNCNTL 表から) で識別される、特定のサブスクリプション・セット・メンバーの登録済みソースでの変更のキャプチャーを開始します。 例えば、アプライ・プログラムはセット内のすべてのターゲット表に対してフル・リフレッシュを実行する前にこのシグナルを発行して、このセットで変更キャプチャー・レプリケーションを行う準備ができていることをキャプチャー・プログラムに知らせます。アプライ・プログラムがこのシグナルを通知します。 **STOP** キャプチャー・プログラムは変更のキャプチャーを停止して終了します。このコマンドは、ユーザーからのみ発行できます。アプライ・プログラムからは発行できません。 **CAPSTOP** キャプチャー・プログラムは、SIGNAL_INPUT_IN 列の中の *source_owner.source_table* で識別される特定の登録済みソースの変更のキャプチャーを停止します。このコマンドは、ユーザーからのみ発行できます。アプライ・プログラムからは発行できません。 **UPDANY** アプライ・プログラム (SIGNAL_INPUT_IN 列の中のアプライ修飾子で識別される) は、Update-anywhere 構成で、2 つのキャプチャー・プログラムを使用していることを、キャプチャー・プログラムに知らせます。アプライ・プログラムがこのシグナルを通知します。 シグナル・タイプが USER の場合、シグナル・サブタイプは使用されないか、キャプチャー・プログラムから認識されないため、これは必要フィールドではありません。どのような値に設定してもかまいません。
SIGNAL_INPUT_IN	**データ・タイプ**: VARCHAR(500)。**NULL 可能**: 可 SIGNAL_TYPE = USER の場合、この列は、ユーザー定義の入力を保持します。SIGNAL_TYPE = CMD の場合は、この値の意味は、このシグナルの SIGNAL_SUBTYPE によって異なります。 **CMD + CAPSTART** マッピング ID。DB2 以外のリレーショナル・ソースは、キャプチャー・プログラムではなく、キャプチャー・トリガーによって処理されるため、シーケンス内の次の値で IBMSNAP_PRUNCNTL 表を更新する、IBMSNAP_SIGNAL 表の更新後に起動される、SIGNAL_TRIGGER というトリガーがあります。 **CMD + UPDANY** Update-anywhere 構成でアプライ・プログラムを識別するためのアプライ修飾子。 **CMD + CAPSTOP** キャプチャー・プログラムによる変更キャプチャーを停止する必要のある、ソース所有者およびソース表の名前 (*source_owner.source_table*)。

表 82. *IBMSNAP_SIGNAL 表の列 (続き)*

列名	説明
SIGNAL_STATE	**データ・タイプ**: CHAR(1)。**NULL 可能**: 不可 シグナルの状況を示すフラグ。 **P** シグナルはペンディング。キャプチャー・プログラムはまだシグナルを受け取っていません。ユーザーがシグナルを通知するときには、SIGNAL_STATE を P に設定してください。 **R** キャプチャー・プログラムはシグナルを受け取りました。キャプチャー・プログラムは、 SIGNAL_TYPE = USER であるか、SIGNAL_TYPE = CMD および SIGNAL_SUBTYPE = STOP であるシグナルを受け取ると、SIGNAL_STATE を R に設定します (コンプリートを示す C に変更するのではなく)。 **C** キャプチャー・プログラムはシグナルの処理を完了しました。キャプチャー・プログラムは、 SIGNAL_TYPE = CMD の場合、STOP を除くすべての SIGNAL_SUBTYPE について、この値を C に設定します。
SIGNAL_LSN	**データ・タイプ**: CHAR(10) FOR BIT DATA。**NULL 可能**: 可 コミット・レコードのログ・シーケンス番号。この値は、キャプチャー・プログラムからのみ設定されます。

System i

System i では、シグナル表は、ソース表で使用される各ジャーナルに関連しています。これらの表はジャーナル・シグナル表と呼ばれ、グローバルな IBMSNAP_SIGNAL 表と同じ構造です。ジャーナル・シグナル表の名前は *schema*.IBMSNAP_SIGNAL_*xxxx_yyyy* (*xxxx* はジャーナル・ライブラリーで、*yyyy* はジャーナル名) です。この表は自動的に作成されて、ソース・サーバー上のソース・ジャーナルに記録されます。

IBMSNAP_UOW 表

IBMSNAP_UOW 表は、ソース表にコミットされたトランザクションに関する追加情報を提供します。ユーザー・コピーおよびタイプ 9 CCD 以外のすべてのターゲット表タイプの場合、アプライ・プログラムはターゲット表に変更を適用するときに、IBMSNAP_COMMITSEQ 値を突き合わせることにより、IBMSNAP_UOW 表と変更データ (CD) 表を結合します。キャプチャー・プログラムをコールド・スタートすると、この表のすべての項目は削除されます。

サーバー: キャプチャー・コントロール・サーバー

デフォルト・スキーマ: ASN

索引: IBMSNAP_COMMITSEQ、IBMSNAP_LOGMARKER

重要: SQL を使用してこの表を更新するときには、注意してください。この表の変更の方法が不適切であると、予期せぬ結果が生じたり、データが失われたりします。

System i

- Capture for System i は、レプリケーション・ソースのサブセットのデータのキャプチャーを開始することがあるため、部分的コールド・スタートを行った場合は、IBMSNAP_UOW 表の中のすべての行が削除されることはありません。
- 一部のユーザー・プログラムはコミットメント・コントロールを使用しません。このような場合、Capture for System i は、CD 表に複数の行が書き込まれた後で、任意に新しい UOW 行を挿入します。このように見せかけのコミットメント境界を設けることにより、 UOW 表のサイズを削減できます。
- UOW 表は、IBMSNAP_PRUNE_SET 表の情報によってではなく、保存限度に従って整理されます。

キャプチャー・プログラムでは、キャプチャー・スキーマごとに 1 つの IBMSNAP_UOW 表が必要になります。キャプチャー・プログラムは、レプリケーション・ソースでコミットされたログまたはジャーナル・レコードごとに、この表に新しい行を 1 つ挿入します。

キャプチャー・プログラムは、アプライ・プログラムが IBMSNAP_PRUNE_SET 表に挿入した情報に基づいて、UOW 表の整理も行います。

表 83 では、IBMSNAP_UOW 表の列の要旨を示します。

表 83. IBMSNAP_UOW 表の列

列名	説明
IBMSNAP_UOWID	**データ・タイプ**: CHAR(10) FOR BIT DATA。**NULL 可能**: 不可 この作業単位についての、ログ・レコード・ヘッダーからの作業単位 ID。この列を、非コンプリート CCD ターゲット表の一部とすることができます。
IBMSNAP_COMMITSEQ	**データ・タイプ**: CHAR(10) FOR BIT DATA。**NULL 可能**: 不可 キャプチャーされたコミット・ステートメントのログ・レコード・シーケンス番号。ユーザー・コピー以外のすべてのターゲット表タイプの場合、アプライ・プログラムはターゲット表に変更を適用するときに、この列の値に基づいて、UOW 表と CD 表を結合します。
IBMSNAP_LOGMARKER	**データ・タイプ**: TIMESTAMP。**NULL 可能**: 不可 データがコミットされたときの時刻 (キャプチャー・コントロール・サーバーの)。
IBMSNAP_AUTHTKN	**データ・タイプ**: VARCHAR(30)。**NULL 可能**: 不可 トランザクションに関連付けられた許可トークン。この ID は、データベースの監査に役立ちます。 DB2 for z/OS の場合、この列は相関 ID です。DB2 for i5/OS の場合、この列は、トランザクションを発生させたジョブのジョブ名です。この列は自動的には他の表にコピーされません。ユーザー・データ列として選択してコピーする必要があります。この列を、非コンプリート CCD ターゲット表の一部とすることができます。

表 83. IBMSNAP_UOW 表の列 (続き)

列名	説明
IBMSNAP_AUTHID	**データ・タイプ**: VARCHAR(30)、DB2 UDB for z/OS バージョン 8 新機能モード・サブシステムの場合 VARCHAR(128)。**NULL 可能**: 不可 トランザクションに関連付けられた許可 ID。これはデータベースの監査に役立ちます。 DB2 for z/OS の場合、この列は 1 次許可 ID です。DB2 for i5/OS の場合、この列は、トランザクションを発生させたアプリケーションを実行しているユーザー・プロファイル ID の名前になります。この列にはブランクが埋め込まれた 10 文字の ID が入ります。この列は自動的には他の表にコピーされません。ユーザー・データ列として選択してコピーする必要があります。この列を、非コンプリート CCD ターゲット表の一部とすることができます。
IBMSNAP_REJ_CODE	**データ・タイプ**: CHAR(1)。**NULL 可能**: 不可。デフォルトあり。**デフォルト**: 0。 リジェクトまたはロールバックされた行があるかどうかを示すフラグ。この値は、レプリケーション・ソースの定義時に競合検出が標準または拡張と指定された場合に Update-anywhere レプリケーションでのみ使用されます。この列を、非コンプリート CCD ターゲット表の一部とすることができます。 **0** トランザクションで競合の発生は報告されていません。 **1** マスターとレプリカの同じ行が更新されたため、競合が発生しています。レプリカ側のトランザクションはリジェクトされ、ロールバックされます。 **2** このトランザクションは、以前にリジェクトされたトランザクションに従属するため、リジェクトされてロールバックされました。前のトランザクションは、マスターとレプリカの中で同じ行が更新されたためにリジェクトされたものであり、レプリカ側のトランザクションが、リジェクトされてロールバックされています。 **3** 参照整合性制約違反が少なくとも 1 つ含まれているため、このトランザクションはリジェクトされ、ロールバックされました。このトランザクションはソース表で定義された参照制約に違反しているため、アプライ・プログラムはこのサブスクリプション・セットに失敗というマークを付けます。参照整合性定義が訂正されるまで、更新はコピーできません。 **4** このトランザクションは、以前にリジェクトされたトランザクションに従属するため、リジェクトされてロールバックされました。前のトランザクションは、参照整合性制約違反が少なくとも 1 つ含まれているためにリジェクトされました。
IBMSNAP_APPLY_QUAL	**データ・タイプ**: CHAR(18)。**NULL 可能**: 不可。デフォルトあり。**デフォルト**: 現行ユーザー名。 変更を適用したアプライ・プログラムを識別するためのアプライ修飾子。この列を、非コンプリート CCD ターゲット表の一部とすることができます。

アプライ・コントロール・サーバーの表

アプライ・コントロール・サーバーに保管される表には、ユーザーのサブスクリプション定義に関する情報が入っています。Linux、UNIX、Windows、および z/OS の場合は、ASNCLP コマンド行プログラムまたはレプリケーション・センターを使用して、ユーザーの指定に合わせてこれらのコントロール表を作成します。 System i の場合は、DataPropagator for System i のインストール時に、これらのコントロール表が自動的に作成されます。

表 84 は、アプライ・サーバーのコントロール表を説明しています。

表 84. アプライ・サーバーのコントロール表

表名	説明
『ASN.IBMSNAP_APPENQ 表』	1 つのアプライ修飾子に対して 1 つのアプライ・プログラムだけが確実に実行されるようにするために使用されます。
System i ASN.IBMSNAP_APPLY_JOB表 (System i)	アプライ・コントロール・サーバーで実行中のアプライ・プログラムのインスタンスごとにユニークなアプライ修飾子があることを確認するのに使用します。
471 ページの『ASN.IBMSNAP_APPLYTRACE 表』	アプライ・プログラムからの重要なメッセージを保持します。
472 ページの『ASN.IBMSNAP_APPLYTRAIL 表』	アプライ・プログラムに関する監査証跡情報を保持します。
468 ページの『ASN.IBMSNAP_APPPARMS 表』	アプライ・プログラムの操作をコントロールするためにユーザーが変更できるパラメーターが含まれています。
478 ページの『ASN.IBMSNAP_SUBS_COLS 表』	ターゲット表またはビュー内の列を、ソース表またはビュー内の対応する列にマップします。
480 ページの『ASN.IBMSNAP_SUBS_EVENT 表』	アプライ・プログラムによるサブスクリプション・セットの処理をコントロールするためにユーザーが定義するイベントを保持します。
481 ページの『ASN.IBMSNAP_SUBS_MEMBR 表』	ソースとターゲット表の対を識別し、その対の処理情報を指定します。
486 ページの『ASN.IBMSNAP_SUBS_SET 表』	アプライ・プログラムによりグループとして処理されるサブスクリプション・セット・メンバーの各セットの処理情報を保持します。
492 ページの『ASN.IBMSNAP_SUBS_STMTS 表』	ユーザーがサブスクリプション・セットに対して定義する、SQL ステートメント、またはストアード・プロシージャー呼び出しを保持します。これらは、アプライ・プログラムによるセットの処理前、または処理後に呼び出されます。

ASN.IBMSNAP_APPENQ 表

アプライ・エンキュー表は、1 つのアプライ修飾子に対して 1 つのアプライ・プログラムだけが確実に実行されるようにするために使用されます。アプライ・プログ

ラムは、アプライ・プログラムがシャットダウンするまで、この表の中の行を排他的にロックします。この表は System i では使用されません。

サーバー: アプライ・コントロール・サーバー

索引: APPLY_QUAL

重要: SQL を使用してこの表を更新するときには、注意してください。この表の変更の方法が不適切であると、予期せぬ結果が生じたり、データが失われたりします。

表 85 では、IBMSNAP_APPENQ 表の列の要旨を示します。

表 85. IBMSNAP_APPENQ 表の列

列名	説明
APPLY_QUAL	**データ・タイプ**: CHAR(18)。**NULL 可能**: 可 同じアプライ・プログラムにより処理されるサブスクリプション・セットのグループを固有に指定します。この値は大文字小文字が区別されます。サブスクリプション・セットの定義時には、この値を指定しなければなりません。

ASN.IBMSNAP_APPLY_JOB (System i)

System i 特有のものである IBMSNAP_APPLY_JOB 表は、アプライ・コントロール・サーバー上で実行中のアプライ・プログラムのすべてのインスタンスの APPLY_QUAL の値がユニークなものであることを保証するために使用されます。アプライ・プログラムの 1 つのインスタンスが開始されるたびに、この表に行が追加されます。アプライ・プログラムの新しいインスタンスを始動するときに、その APPLY_QUAL 値がすでに存在していると、始動コマンドは失敗します。

サーバー: アプライ・コントロール・サーバー

索引: なし

重要: SQL を使用してこの表を更新するときには、注意してください。この表の変更の方法が不適切であると、予期せぬ結果が生じたり、データが失われたりします。

表 86 では、 IBMSNAP_APPLY_JOB 表の列の要旨を示します。

表 86. IBMSNAP_APPLY_JOB 表の列

列名	説明
APPLY_QUAL	**データ・タイプ**: CHAR(18)。**NULL 可能**: 不可 サブスクリプション・セットのグループのユニーク ID。この値は、サブスクリプション・セットの定義時にユーザーによって指定されます。アプライ・プログラムの各インスタンスは、APPLY_QUAL 値によって開始されます。この値は、Update-anywhere レプリケーションで、アプライ・プログラムによる変更の循環レプリケーションを避けるために使用されます。
CONTROL_SERVER	**データ・タイプ**: CHAR(18)。**NULL 可能**: 不可 アプライ・コントロール表およびビューが定義されているデータベースの名前。

表 86. IBMSNAP_APPLY_JOB 表の列 (続き)

列名	説明
JOB_NAME	**データ・タイプ**: CHAR(10)。**NULL 可能**: 不可 このトレース項目を書き込んだジョブの完全修飾名。 **位置 1–10** APPLY_QUAL **位置 11-20** アプライ・プログラムを始動したユーザーの ID **位置 21-26** ジョブ番号
USER_NAME	**データ・タイプ**: CHAR(10)。**NULL 可能**: 不可 アプライ・プログラムの新しいインスタンスを始動したユーザーの名前。
JOB_NUMBER	**データ・タイプ**: CHAR(6)。**NULL 可能**: 不可 特定のジャーナルに対する現行ジョブのジョブ番号。ジャーナルがアクティブでない場合は、この列には、最後に処理されたジョブのジョブ番号が入っています。

ASN.IBMSNAP_APPPARMS 表

IBMSNAP_APPPARMS 表には、アプライ・プログラムの動作を制御するために変更できるパラメーターが含まれています。これらのパラメーターは、サブスクリプション定義とアプライ・プログラム・コントロール表が存在する、アプライ・コントロール・サーバーの名前などの値を設定する場合に定義します。ユーザーがこの表のパラメーターを変更しても、アプライ・プログラムは始動時にしか変更を読み取りません。

サーバー: アプライ・コントロール・サーバー

索引: APPLY_QUAL

この表の情報は、SQL を使って更新できます。

表 87 では、IBMSNAP_APPPARMS 表の列の要旨を示します。

表 87. IBMSNAP_APPPARMS 表の列

列名	説明
APPLY_QUAL	**データ・タイプ**: CHAR(18)。**NULL 可能**: 不可 アプライ修飾子は、パラメーターを、それらパラメーターの適用先のアプライ・プログラムに適合させます。
APPLY_PATH	**データ・タイプ**: VARCHAR(1040)。**NULL 可能**: 可 アプライ・プログラムが使用する作業ファイルのロケーション。デフォルトは、プログラムが開始されたディレクトリーです。

表 87. IBMSNAP_APPPARMS 表の列 (続き)

列名	説明
COPYONCE	**データ・タイプ**: CHAR(1)。**NULL 可能**: 可。デフォルトあり。**デフォルト:** N。 アプライ・プログラムが呼び出された時点で適格と見なされたサブスクリプション・セットごとに、アプライ・プログラムがコピー・サイクルを 1 回実行するかどうかを示すフラグ。 **Y** アプライ・プログラムは、適格なサブスクリプション・セットごとにコピー・サイクルを 1 回実行します。 **N** アプライ・プログラムは、適格なサブスクリプション・セットごとにコピー・サイクルを 1 回実行しません。
DELAY	**データ・タイプ**: INT。**NULL 可能**: 可。デフォルトあり。**デフォルト:** 6。 レプリケーションを連続して実行する場合に、アプライ・サイクルが終わるたびに経過させる遅延時間 (秒単位)。 **copyonce** が指定されている場合、このパラメーターは無視されます。
ERRWAIT	**データ・タイプ**: INT。**NULL 可能**: 可。デフォルトあり。**デフォルト:** 300。 アプライ・プログラムがエラー状態になった後、何秒待ってから再試行するかを示す秒数 (1 〜 300)。 **copyonce** が指定されている場合、このパラメーターは無視されます。
INAMSG	**データ・タイプ**: CHAR(1)。**NULL 可能**: 可。デフォルトあり。**デフォルト:** Y。 アプライ・プログラムを非アクティブにしたとき、アプライ・プログラムからメッセージを出すかどうかを示すフラグ。 **Y** アプライ・プログラムは非アクティブ時にメッセージを出します。 **N** アプライ・プログラムは非アクティブ時にメッセージを出しません。
LOADXIT	**データ・タイプ**: CHAR(1)。**NULL 可能**: 可。デフォルトあり。**デフォルト:** N。 アプライ・プログラムが、エクスポート・ユーティリティーやロード・ユーティリティーを使用してターゲット表をリフレッシュする、 IBM 提供の出口ルーチン (ASNLOAD) を呼び出すかどうかを示すフラグ。 **Y** アプライ・プログラムは ASNLOAD を呼び出します。 **N** アプライ・プログラムは ASNLOAD を呼び出しません。
LOGREUSE	**データ・タイプ**: CHAR(1)。**NULL 可能**: 可。デフォルトあり。**デフォルト:** N。 アプライ・プログラムがアプライ・ログ・ファイルに上書きするか、ファイルに追加するかを示すフラグ。 **Y** アプライ・プログラムは、まずログ・ファイルを削除して、アプライ・プログラムの再始動時にそれを再作成することにより、ログ・ファイルを再利用します。 **N** アプライ・プログラムは新しい情報をアプライ・ログ・ファイルに追加します。

表 87. IBMSNAP_APPPARMS 表の列 (続き)

列名	説明
LOGSTDOUT	**データ・タイプ**: CHAR(1)。**NULL 可能**: 可。デフォルトあり。**デフォルト:** N。 アプライ・プログラムがログ・ファイル・メッセージを送る場所を示すフラグ。 **Y** アプライ・プログラムは、標準出力 (STDOUT) とログ・ファイルの両方にログ・ファイル・メッセージを送信します。 **N** アプライ・プログラムは、ほとんどのログ・ファイル・メッセージをログ・ファイルにのみ送ります。初期化メッセージは、標準出力 (STDOUT) とログ・ファイルの両方に送られます。
NOTIFY	**データ・タイプ**: CHAR(1)。**NULL 可能**: 可。デフォルトあり。**デフォルト:** N。 アプライ・プログラムが、サブスクリプション・セットをコピーした後に、ユーザーにコントロールを戻す出口ルーチン (ASNDONE) を呼び出すかどうかを示すフラグ。 **Y** アプライ・プログラムは ASNDONE を呼び出します。 **N** アプライ・プログラムは ASNDONE を呼び出しません。
OPT4ONE	**データ・タイプ**: CHAR(1)。**NULL 可能**: 可。デフォルトあり。**デフォルト:** N。 アプライ・プログラムに定義されているサブスクリプション・セットが 1 つだけの場合、アプライ・プログラムのパフォーマンスを最適化するかどうかを示すフラグ。 **Y** サブスクリプション・セットが 1 つの場合、アプライ・プログラムのパフォーマンスを最適化します。 **N** サブスクリプション・セットが 1 つの場合、アプライ・プログラムのパフォーマンスを最適化しません。 **copyonce** が指定されている場合、このパラメーターは無視されます。
SLEEP	**データ・タイプ**: CHAR(1)。**NULL 可能**: 可。デフォルトあり。**デフォルト:** Y。 処理の対象として適格となる新しいサブスクリプションがない場合に、アプライ・プログラムがどうするかを示すフラグ。 **Y** アプライ・プログラムはスリープ状態に入ります。 **N** アプライ・プログラムは停止します。 **copyonce** が指定されている場合、このパラメーターは無視されます。
SQLERRCONTINUE	**データ・タイプ**: CHAR(1)。**NULL 可能**: 可。デフォルトあり。**デフォルト:** N。 アプライ・プログラムが、SQLSTATE ファイルのエラーをチェックした後に処理を続行するかどうかを示すフラグ。 **Y** アプライ・プログラムは処理中に SQLSTATE ファイルに SQL エラーがないかチェックします。エラーが見つかった場合、アプライは処理を停止します。 **N** アプライ・プログラムは SQLSTATE ファイルをチェックせず、処理を続行します。

表 87. IBMSNAP_APPPARMS 表の列 (続き)

列名	説明
SPILLFILE	**データ・タイプ**: VARCHAR(10)。**NULL 可能**: 可。デフォルトあり。 フェッチした応答セットをどこに保管するかを示すフラグ。 z/OS 有効な値は以下のとおりです。 **mem (デフォルト)** メモリー・ファイル。 **disk** ディスク・ファイル。 Linux UNIX Windows 有効な値は以下のとおりです。 **disk (デフォルト)** ディスク・ファイル。
TERM	**データ・タイプ**: CHAR(1)。**NULL 可能**: 可。デフォルトあり。**デフォルト:** Y。 DB2 が静止または停止するとき、アプライ・プログラムが停止するかどうかを示すフラグ。 **Y** DB2 が静止または停止するとき、アプライ・プログラムは終了します。 **N** アプライ・プログラムはアクティブのまま、DB2 が再始動または静止解除されるのを待ちます。 **copyonce** が指定されている場合、このパラメーターは無視されます。
TRLREUSE	**データ・タイプ**: CHAR(1)。**NULL 可能**: 可。デフォルトあり。**デフォルト:** N。 アプライ・プログラムが、エクスポート・ユーティリティーやロード・ユーティリティーを使用してターゲット表をリフレッシュする、 IBM 提供の出口ルーチン (ASNLOAD) を呼び出すかどうかを示すフラグ。 **Y** アプライ・プログラムは ASNLOAD を呼び出します。 **y** アプライ・プログラムは ASNLOAD を呼び出しません。

ASN.IBMSNAP_APPLYTRACE 表

IBMSNAP_APPLYTRACE 表は、アプライ・プログラムからのメッセージを保持します。アプライ・プログラムはこの表の整理を自動的に行うことはありませんが、1 つのサブスクリプション・セットの後で実行される SQL ステートメントを追加することにより、整理を自動化できます。

サーバー: アプライ・コントロール・サーバー

索引: APPLY_QUAL、TRACE_TIME

表 88 では、IBMSNAP_APPLYTRACE 表の列の要旨を示します。

表 88. IBMSNAP_APPLYTRACE 表の列

列名	説明
APPLY_QUAL	**データ・タイプ**: CHAR(18)。**NULL 可能**: 不可 メッセージを挿入したアプライ・プログラムを一意的に識別します。

表 88. IBMSNAP_APPLYTRACE 表の列 (続き)

列名	説明
TRACE_TIME	**データ・タイプ**: TIMESTAMP。**NULL 可能**: 不可 この表に行が挿入されたときのアプライ・コントロール・サーバーの時刻。
OPERATION	**データ・タイプ**: CHAR(8)。**NULL 可能**: 不可 アプライ・プログラムの操作のタイプ、例えば、初期化、アプライ、またはエラー条件。
DESCRIPTION	**データ・タイプ**: VARCHAR(1024)。**NULL 可能**: 不可 メッセージ ID とメッセージ・テキスト。メッセージ ID は DESCRIPTION 列の最初の 7 文字です。 メッセージ・テキストは、DESCRIPTION 列の位置 9 から始まります。

ASN.IBMSNAP_APPLYTRAIL 表

IBMSNAP_APPLYTRAIL 表には、アプライ・プログラムによって実行される、すべてのサブスクリプション・セット・サイクルの監査証跡情報が含まれます。アプライ・トレール表には、サブスクリプションに対して実行された更新の履歴が入ります。この表は、診断およびパフォーマンス統計のリポジトリーです。アプライ・プログラムで問題が生じたときにアプライ・トレール表を参照するのは非常に効果的です。アプライ・プログラムはこの表の整理を自動的に行うことはありませんが、1 つのサブスクリプション・セットの後で実行される SQL ステートメントを追加することにより、簡単に整理を自動化できます。

サーバー: アプライ・コントロール・サーバー

索引: LASTRUN、 APPLY_QUAL

表 89 では、IBMSNAP_APPLYTRAIL 表の列の要旨を示します。

表 89. IBMSNAP_APPLYTRAIL 表の列

列名	説明
APPLY_QUAL	**データ・タイプ**: CHAR(18)。**NULL 可能**: 不可 サブスクリプション・セットを処理したアプライ・プログラムを一意的に識別します。
SET_NAME	**データ・タイプ**: CHAR(18)。**NULL 可能**: 不可 アプライ・プログラムが処理したサブスクリプション・セットの名前。
SET_TYPE	**データ・タイプ**: CHAR(1)。**NULL 可能**: 不可 最後のアプライ・サイクルの後で IBMSNAP_SUBS_SET 表の SET_TYPE 列に表示された値。

表 89. IBMSNAP_APPLYTRAIL 表の列 (続き)

列名	説明
WHOS_ON_FIRST	**データ・タイプ**: CHAR(1)。**NULL 可能**: 不可 Update-anywhere レプリケーション・シナリオでは、処理順序をコントロールするために以下の値を使用します。 **F** (first の略) ソース表がレプリカであり、ターゲット表がマスターです。レプリカ表とマスター表の間で更新の矛盾が生じた場合、レプリカの側の矛盾するトランザクションはリジェクトされます。F は読み取り専用のサブスクリプションでは使用されません。Update-anywhere で使用されるものです。 **S** (second の略) ソース表はマスター表またはその他のソースであり、ターゲット表はレプリカまたはその他のコピーです。マスター表とレプリカ表の間で更新の矛盾が生じた場合、レプリカの側の矛盾するトランザクションはリジェクトされます。S は、すべての読み取り専用サブスクリプションについて使用されます。
ASNLOAD	**データ・タイプ**: CHAR(1)。**NULL 可能**: 可 アプライ・プログラムを始動するために使用された値。 **Y** パラメーター **loadxit=y** を指定してアプライ・プログラムを始動したため、サブスクリプション・セットのフル・リフレッシュを実行するために ASNLOAD ユーザー出口ルーチンが呼び出されることを意味します。 **N** フル・リフレッシュが必要ないか、アプライ・プログラムの始動時に **loadxit** パラメーターが指定されていなかったため、ASNLOAD 出口ルーチンが呼び出されないことを意味します。 **NULL** ASNLOAD 出口ルーチンを呼び出すかどうかをアプライ・プログラムが判断する前に、アプライ・プログラム・エラーが生じたことを示します。
FULL_REFRESH	**データ・タイプ**: CHAR(1)。**NULL 可能**: 可 フル・リフレッシュが発生したかどうかを示すフラグ。 **Y** サブスクリプション・セットに対してフル・リフレッシュが実行されたことを示します。 **N** サブスクリプション・セットに対してフル・リフレッシュが実行されなかったことを示します。 **NULL** フル・リフレッシュが必要かどうかをアプライ・プログラムが判断する前に、エラーが生じたことを示します。
EFFECTIVE_MEMBERS	**データ・タイプ**: INT。**NULL 可能**: 可 フル・リフレッシュか、挿入、更新、および削除のレプリケーションのどちらかにより、1 回のアプライ・サイクルで変更されたサブスクリプション・セット・メンバーの数。値の範囲は、0 以上、定義済みのサブスクリプション・セット・メンバーの数以下です。
SET_INSERTED	**データ・タイプ**: INT。**NULL 可能**: 不可 サブスクリプション・サイクルにおいてサブスクリプション・セット・メンバーに挿入された行の合計数。

表 89. IBMSNAP_APPLYTRAIL 表の列 (続き)

列名	説明
SET_DELETED	**データ・タイプ**: INT。**NULL 可能**: 不可 サブスクリプション・サイクルにおいてサブスクリプション・セット・メンバーから削除された行の合計数。
SET_UPDATED	**データ・タイプ**: INT。**NULL 可能**: 不可 サブスクリプション・サイクルにおいてサブスクリプション・セット・メンバーで更新された行の合計数。
SET_REWORKED	**データ・タイプ**: INT。**NULL 可能**: 不可 最後のサイクルでアプライ・プログラムが再処理した合計行数。アプライ・プログラムは、以下の条件下で変更を再試行します。 • 行がターゲット表にすでに存在しているため挿入が失敗した場合、アプライ・プログラムは、挿入操作を既存行の更新操作に変換します。 • 行がターゲット表に存在していないため更新が失敗した場合、アプライ・プログラムは、更新操作を挿入操作に変換します。
SET_REJECTED_TRXS	**データ・タイプ**: INT。**NULL 可能**: 不可 Update-anywhere 競合のためにリジェクトされたトランザクションの合計数。この列は、競合検出が「標準」または「詳細」と定義されている Update-anywhere サブスクリプション・セットに対してのみ使用されます。

表 89. IBMSNAP_APPLYTRAIL 表の列 (続き)

列名	説明
STATUS	**データ・タイプ**: SMALLINT。**NULL 可能**: 不可 特定のサイクル後のアプライ・プログラムの作業状況を表す値。 **-1** レプリケーションは失敗しました。アプライ・プログラムは適用済みの行のセット全体をバックアウトし、データはコミットされません。始動パラメーターが SQLERRCONTINUE = Y の場合、最後のサイクル中にアプライ・プログラムに戻される SQLSTATE は、SQLERRCONTINUE (*apply qualifier*.SQS) の入力ファイルでユーザーが指定した許容エラーの 1 つではありません。 **0** アプライ・プログラムはサブスクリプション・セットを正常に処理しました。始動パラメーターが SQLERRCONTINUE = Y の場合、アプライ・プログラムは、ユーザーから SQLERRCONTINUE 始動パラメーターで (*apply_qualifier*.SQS で) 指定された SQL エラーを検出しておらず、行をリジェクトしていません。 **2** アプライ・プログラムはサブスクリプション・セットを複数のサイクルで処理しています。アプライ・プログラムは、MAX_SYNCH_MINUTES コントロール列に従って分割された 1 つの論理サブスクリプションを正常に処理しました。 **16** アプライ・プログラムはサブスクリプション・セットを正常に処理し、0 という状況を戻しました。しかしアプライ・プログラムは、ユーザーが SQLERRCONTINUE 始動パラメーターで (*apply_qualifier*.SQS で) 指定したいくつかの SQL エラーを検出したため、いくつかの行をリジェクトしました。失敗した行の詳細は、*apply_qualifier*.ERR ファイルで確認してください。 **例**: ユーザーは SQLERRCONTINUE = Y と設定し、SQL の許容される状態を 23502 (SQL コード -407) と指定します。23502 エラーが発生しますが、他のエラーは発生していません。アプライ・プログラムはサブスクリプション・セットの処理を終了し、状態を 16 に設定します。次の実行時に、23502 エラーが発生した後、07006 (SQL コード -301) が発生します。アプライ・プログラムは今回は、サブスクリプション・セットの処理を停止し、適用済みの行のセット全体をバックアウトし、状況を -1 に設定します (データはコミットされていないため)。 **18** アプライ・プログラムは複数のサイクルでサブスクリプション・セットを処理し、 2 という状況を戻しています。これは、MAX_SYNCH_MINUTES コントロール列に従って分割された 1 つの論理サブスクリプションが正常に処理されたことを意味します。しかし、ユーザーが SQLERRCONTINUE 始動パラメーターで (*apply_qualifier*.SQS で) 指定した SQL エラーのいくつかが検出されたため、いくつかの行がリジェクトされています。失敗した行の詳細は、*apply_qualifier*.ERR ファイルで確認してください。
LASTRUN	**データ・タイプ**: TIMESTAMP。**NULL 可能**: 不可 最後のサブスクリプションが開始された概算の時刻。アプライ・プログラムは、サブスクリプション・セットを処理するたびに LASTRUN 値を設定します。これは、アプライ・プログラムがサブスクリプション・セットの処理を開始する、アプライ・コントロール・サーバーにおけるおおよその時刻です。

表 89. IBMSNAP_APPLYTRAIL 表の列 (続き)

列名	説明
LASTSUCCESS	**データ・タイプ**: TIMESTAMP。**NULL 可能**: 可 サブスクリプション・セットが最後に正常に処理されたときの、処理開始時点のアプライ・コントロール・サーバーのタイム・スタンプ。
SYNCHPOINT	**データ・タイプ**: CHAR(10) FOR BIT DATA。**NULL 可能**: 可 アプライ・プログラムはこの列を使用して進行状況を記録します。サブスクリプション・セットのデータの処理が、この同期点の値まで終了していることを示します。
SYNCHTIME	**データ・タイプ**: TIMESTAMP。**NULL 可能**: 可 アプライ・プログラムはこの列を使用して進行状況を記録します。サブスクリプション・セットのデータの処理が、このタイム・スタンプまで終了していることを示します。
SOURCE_SERVER	**データ・タイプ**: CHAR(18)。**NULL 可能**: 不可 ソース表およびビューが定義されている DB2 データベース名。
SOURCE_ALIAS	**データ・タイプ**: CHAR(8)。**NULL 可能**: 可 SOURCE_SERVER 列で指定されているソース・サーバーに対応する DB2 別名。
SOURCE_OWNER	**データ・タイプ**: VARCHAR(30)、DB2 UDB for z/OS バージョン 8 新機能モード・サブシステムの場合 VARCHAR(128)。**NULL 可能**: 可 アプライ・プログラムが処理中であったソース表またはビューの上位修飾子。この値は、アプライ・サイクルが失敗したときにのみ設定されます。
SOURCE_TABLE	**データ・タイプ**: VARCHAR(128)、DB2 UDB for z/OS バージョン 8 互換モード・サブシステムかそれ以前の場合 VARCHAR(18)。**NULL 可能**: 可。 アプライ・プログラムが処理中であったソース表またはビューの名前。この値は、アプライ・サイクルが失敗したときにのみ設定されます。
SOURCE_VIEW_QUAL	**データ・タイプ**: SMALLINT。**NULL 可能**: 可 アプライ・プログラムが処理中であったソース表またはビューのソース・ビュー修飾子の値。この値は、アプライ・サイクルが失敗したときにのみ設定されます。
TARGET_SERVER	**データ・タイプ**: CHAR(18)。**NULL 可能**: 不可 ターゲットの表またはビューが保管されているサーバーのデータベース名。
TARGET_ALIAS	**データ・タイプ**: CHAR(8)。**NULL 可能**: 可 TARGET_SERVER 列で指定されているターゲット・サーバーに対応する DB2 別名。
TARGET_OWNER	**データ・タイプ**: VARCHAR(30)、DB2 UDB for z/OS バージョン 8 新機能モード・サブシステムの場合 VARCHAR(128)。**NULL 可能**: 不可 アプライ・プログラムが処理中であったターゲット表の上位修飾子。この値は、アプライ・サイクルが失敗したときにのみ設定されます。

表 89. IBMSNAP_APPLYTRAIL 表の列 (続き)

列名	説明
TARGET_TABLE	**データ・タイプ**: VARCHAR(128)、DB2 UDB for z/OS バージョン 8 互換モード・サブシステムかそれ以前の場合 VARCHAR(18)。**NULL 可能**: 不可。 アプライ・プログラムが処理中であったターゲット表の名前。この値は、アプライ・サイクルが失敗したときにのみ設定されます。
CAPTURE_SCHEMA	**データ・タイプ**: VARCHAR(30)、DB2 UDB for z/OS バージョン 8 新機能モード・サブシステムの場合 VARCHAR(128)。**NULL 可能**: 不可 このサブスクリプション・セットのキャプチャー・サーバー表のスキーマ名。
TGT_CAPTURE_SCHEMA	**データ・タイプ**: VARCHAR(30)、DB2 UDB for z/OS バージョン 8 新機能モード・サブシステムの場合 VARCHAR(128)。**NULL 可能**: 可 ターゲット表が別のサブスクリプション・セットのソースでもある場合 (multi-tier 構成中の外部の CCD 表、または Update-anywhere 構成中のレプリカ表など) は、この列には、表がソースとして機能するときに使用されるキャプチャー・スキーマが含まれます。
FEDERATED_SRC_SRVR	**データ・タイプ**: VARCHAR(18)。**NULL 可能**: 可 DB2 以外のリレーショナル・ソースの場合にのみアプライされる、サブスクリプション・セットのソースである、フェデレーテッド・リモート・サーバーの名前。
FEDERATED_TGT_SRVR	**データ・タイプ**: VARCHAR(18)。**NULL 可能**: 可 DB2 以外のリレーショナル・ターゲット・サーバーの場合にのみアプライされる、サブスクリプション・セットのターゲットである、フェデレーテッド・リモート・サーバーの名前。
JRN_LIB	**データ・タイプ**: CHAR(10)。**NULL 可能**: 可 System i System i キャプチャー・サーバーにのみアプライされるこの列は、ソース表が使用するジャーナルのライブラリー名です。
JRN_NAME	**データ・タイプ**: CHAR(10)。**NULL 可能**: 可 System i System i キャプチャー・サーバーにのみアプライされるこの列は、ソース表が使用するジャーナルの名前です。この列で、アスタリスクの後ろに 9 つのブランクが続くときには、ソース表が現在ジャーナルの中にないことを意味します。この場合は、このソース表のデータをキャプチャーすることはできません。
COMMIT_COUNT	**データ・タイプ**: SMALLINT。**NULL 可能**: 可 IBMSNAP_SUBS_SET 表の中に記録される、最後のアプライ・サイクルからの COMMIT_COUNT の値。
OPTION_FLAGS	**データ・タイプ**: CHAR(4)。**NULL 可能**: 不可 SQL レプリケーションの将来のオプション用に予約済み。この列には現在、デフォルト値の NNNN が入っています。
EVENT_NAME	**データ・タイプ**: CHAR(18)。**NULL 可能**: 可 セットの処理を起動したイベントを表すために使用されるユニークな文字ストリング。

表 89. IBMSNAP_APPLYTRAIL 表の列 (続き)

列名	説明
ENDTIME	**データ・タイプ**: TIMESTAMP。**NULL 可能**: デフォルトでは不可。**デフォルト**: 現行タイム・スタンプ。 アプライ・プログラムがサブスクリプション・セットの処理を終了したときの、アプライ・コントロール・サーバーにおけるタイム・スタンプ。セットの処理に要した時間を知るには、LASTRUN を ENDTIME から減算します。
SOURCE_CONN_TIME	**データ・タイプ**: TIMESTAMP。**NULL 可能**: 可 アプライ・プログラムが初めてフェッチ・ソース・データに接続したときの、キャプチャー・コントロール・サーバーにおけるタイム・スタンプ。
SQLSTATE	**データ・タイプ**: CHAR(5)。**NULL 可能**: 可 失敗した実行の SQL 状態コード。それ以外の場合は、NULL になります。
SQLCODE	**データ・タイプ**: INT。**NULL 可能**: 可 失敗した実行の SQL エラー・コード。それ以外の場合は、NULL になります。
SQLERRP	**データ・タイプ**: CHAR(8)。**NULL 可能**: 可 実行の失敗原因となった SQL エラーが生じたサーバーのデータベース製品 ID。それ以外の場合は、NULL になります。
SQLERRM	**データ・タイプ**: VARCHAR(70)。**NULL 可能**: 可 失敗した実行の SQL エラー情報。
APPERRM	**データ・タイプ**: VARCHAR(760)。**NULL 可能**: 可 アプライ・プログラムの実行に失敗したときのエラー・メッセージ ID およびテキスト。

ASN.IBMSNAP_SUBS_COLS 表

IBMSNAP_SUBS_COLS 表には、サブスクリプション・セット内にコピーされるサブスクリプション・セット・メンバーの列に関する情報が含まれます。一対のソース表とターゲット表で、 1 つまたは複数の列の情報が変更されると、この表で自動的に行が挿入または削除されます。この表は、サブスクリプション・セット・メンバーの特定の列に関する情報が必要な場合に使用してください。

サーバー: アプライ・コントロール・サーバー

索引: APPLY_QUAL、SET_NAME、WHOS_ON_FIRST、TARGET_OWNER、TARGET_TABLE、TARGET_NAME

重要: SQL を使用してこの表を更新するときには、注意してください。この表の変更の方法が不適切であると、予期せぬ結果が生じたり、データが失われたりします。

479 ページの表 90 では、 IBMSNAP_SUBS_COLS 表の列の要旨を示します。

表 90. IBMSNAP_SUBS_COLS 表の列

列名	説明
APPLY_QUAL	**データ・タイプ**: CHAR(18)。**NULL 可能**: 不可 このサブスクリプション・セット・メンバーを処理するアプライ・プログラムを一意的に識別します。
SET_NAME	**データ・タイプ**: CHAR(18)。**NULL 可能**: 不可 このメンバーが所属するサブスクリプション・セットの名前。
WHOS_ON_FIRST	**データ・タイプ**: CHAR(1)。**NULL 可能**: 不可 Update-anywhere レプリケーション・シナリオでは、処理順序をコントロールするために以下の値を使用します。 **F** (first の略) ソース表がレプリカであり、ターゲット表がマスターです。レプリカ表とマスター表の間で更新の矛盾が生じた場合、レプリカの側の矛盾するトランザクションはリジェクトされます。F は読み取り専用のサブスクリプションでは使用されません。Update-anywhere で使用されるものです。 **S** (second の略) ソース表はマスター表またはその他のソースであり、ターゲット表はレプリカまたはその他のコピーです。マスター表とレプリカ表の間で更新の矛盾が生じた場合、レプリカの側の矛盾するトランザクションはリジェクトされます。S は、すべての読み取り専用サブスクリプションについて使用されます。
TARGET_OWNER	**データ・タイプ**: VARCHAR(30)、DB2 UDB for z/OS バージョン 8 新機能モード・サブシステムの場合 VARCHAR(128)。**NULL 可能**: 不可 ターゲット表またはビューの上位修飾子。
TARGET_TABLE	**データ・タイプ**: VARCHAR(128)、DB2 UDB for z/OS バージョン 8 互換モード・サブシステムかそれ以前の場合 VARCHAR(18)。**NULL 可能**: 不可。 データが適用される表またはビュー。
COL_TYPE	**データ・タイプ**: CHAR(1)。**NULL 可能**: 不可 列のタイプを示すフラグ。 **A** 変更後イメージ列。 **B** 変更前イメージ列。 **C** スカラー関数を使用する算出列または SQL 式。 **F** 列関数を使用する算出列。 **L** LOB 標識値。 **P** 変更前イメージ述部列。 **R** システムから提供され、主キー列として使用される相対レコード番号列。DB2 DataPropagator for System i でのみ使用されます。
TARGET_NAME	**データ・タイプ**: VARCHAR(30)。**NULL 可能**: 不可 ターゲット表またはビューの列の名前。ソース列名と一致する必要はありません。 内部の CCD 列名は変更できません。それらの名前はソース表の列名と一致している必要があります。

表 90. IBMSNAP_SUBS_COLS 表の列 (続き)

列名	説明
IS_KEY	**データ・タイプ**: CHAR(1)。**NULL 可能**: 不可 列がターゲット・キーの一部であるかどうかを示すフラグ。ターゲット・キーは、コンデンス・ターゲット表のユニーク索引、または主キーのいずれかです。 **Y** 列はターゲット・キー全体であるか、ターゲット・キーの一部です。 **N** 列は、ターゲット・キーの一部ではありません。
COLNO	**データ・タイプ**: SMALLINT。**NULL 可能**: 不可 オリジナル・ソースにおける列の数値ロケーション。これは、表示およびサブスクリプションで他のユーザー列に対して相対的に保たれます。
EXPRESSION	**データ・タイプ**: VARCHAR(254)。**NULL 可能**: 不可 ターゲット列の内容を作成するために使用された SQL 式、またはソース列名。

ASN.IBMSNAP_SUBS_EVENT 表

IBMSNAP_SUBS_EVENT 表には、サブスクリプション・セットに関連付けられたイベント・トリガーに関する情報が含まれます。また、イベント名と関連付けられた名前とタイム・スタンプも含まれます。

サーバー: アプライ・コントロール・サーバー

索引: EVENT_NAME、EVENT_TIME

この表の情報は、SQL を使って更新できます。

アプライ・プログラムを始動するために新規イベントを作成するときに、この表に行を挿入してください。

表 91 では、IBMSNAP_SUBS_EVENT 表の列の要旨を示します。

表 91. IBMSNAP_SUBS_EVENT 表の列

列名	説明
EVENT_NAME	**データ・タイプ**: CHAR(18)。**NULL 可能**: 不可 イベントのユニーク ID。この ID は、サブスクリプション・セットのレプリケーションを起動するために使用されます。
EVENT_TIME	**データ・タイプ**: TIMESTAMP。**NULL 可能**: 不可 現在または将来の通知時刻の、アプライ・コントロール・サーバーのタイム・スタンプ。レプリケーション・イベントをシグナルするユーザー・アプリケーションがこの列に値を提供します。

表 91. IBMSNAP_SUBS_EVENT 表の列 (続き)

列名	説明
END_SYNCHPOINT	**データ・タイプ**: CHAR(10) FOR BIT DATA。**NULL 可能**: 可 この時点までにキャプチャーされたデータのみをアプライするようにアプライ・プログラムに指示するログ・シーケンス番号。シグナル表を参照し、タイム・スタンプに関連付けられた正確なログ・シーケンス番号を検索すれば、使用する正確な END_SYNCHPOINT を知ることができます。ログのこの時点以降にコミットされたトランザクションは、その後のイベントが通知されるまでは複製されません。ユーザーが END_SYNCHPOINT および END_OF_PERIOD の値を指定した場合、アプライ・プログラムは END_SYNCHPOINT 値を使用するので、複製する最大ログ・シーケンス番号を知るためにコントロール表から計算を実行する必要はなくなります。
END_OF_PERIOD	**データ・タイプ**: TIMESTAMP。**NULL 可能**: 可 この時点までにログに記録されたデータのみをアプライするためにアプライ・プログラムによって使用されるタイム・スタンプ。ログのこの時点以降にコミットされたトランザクションは、その後のイベントが通知されるまでは複製されません。

ASN.IBMSNAP_SUBS_MEMBR 表

IBMSNAP_SUBS_MEMBR 表には、サブスクリプション・セットに対して定義された、ソース表とターゲット表の個々のペアに関する情報が含まれています。ユーザーがサブスクリプション・セット・メンバーを追加すると、この表に 1 つの行が自動的に挿入されます。この表は、サブスクリプション・セット内で特定のソース表とターゲット表の対を指定するために使用します。

サーバー: アプライ・コントロール・サーバー

索引: APPLY_QUAL、SET_NAME、WHOS_ON_FIRST、SOURCE_OWNER、SOURCE_TARGET、SOURCE_VIEW_QUAL、TARGET_OWNER、TARGET_TABLE

重要: SQL を使用してこの表を更新するときには、注意してください。この表の変更の方法が不適切であると、予期せぬ結果が生じたり、データが失われたりします。

表 92 では、 IBMSNAP_SUBS_MEMBR 表の列の要旨を示します。

表 92. IBMSNAP_SUBS_MEMBR 表の列

列名	説明
APPLY_QUAL	**データ・タイプ**: CHAR(18)。**NULL 可能**: 不可 このサブスクリプション・セット・メンバーを処理するアプライ・プログラムを一意的に識別します。
SET_NAME	**データ・タイプ**: CHAR(18)。**NULL 可能**: 不可 このメンバーが所属するサブスクリプション・セットの名前。

表 92. IBMSNAP_SUBS_MEMBR 表の列 (続き)

列名	説明
WHOS_ON_FIRST	**データ・タイプ**: CHAR(1)。**NULL 可能**: 不可 Update-anywhere レプリケーション・シナリオでは、処理順序をコントロールするために以下の値を使用します。 **F** (first の略) ソース表がレプリカであり、ターゲット表がマスターです。レプリカ表とマスター表の間で更新の矛盾が生じた場合、レプリカの側の矛盾するトランザクションはリジェクトされます。F は読み取り専用のサブスクリプションでは使用されません。Update-anywhere で使用されるものです。 **S** (second の略) ソース表はマスター表またはその他のソースであり、ターゲット表はレプリカまたはその他のコピーです。マスター表とレプリカ表の間で更新の矛盾が生じた場合、レプリカの側の矛盾するトランザクションはリジェクトされます。S は、すべての読み取り専用サブスクリプションについて使用されます。
SOURCE_OWNER	**データ・タイプ**: VARCHAR(30)。DB2 UDB for z/OS バージョン 8 新機能モード・サブシステムの場合 VARCHAR(128)。**NULL 可能**: 不可 このメンバーのソース表またはビューの上位修飾子。
SOURCE_TABLE	**データ・タイプ**: VARCHAR(128)。DB2 UDB for z/OS バージョン 8 互換モード・サブシステムかそれ以前の場合 VARCHAR(18)。**NULL 可能**: 不可 このメンバーのソース表またはビューの名前。
SOURCE_VIEW_QUAL	**データ・タイプ**: SMALLINT。**NULL 可能**: 不可 IBMSNAP_REGISTER 表内の類似した列を突き合わせることによって、物理表のビューをサポートします。この値は、ソースとして定義されている物理表の場合は 0 に、ソースとして定義されているビューの場合は 0 より大きい値に設定されます。同じ SOURCE_OWNER および SOURCE_TABLE 列値を持つ別々のソース・ビューについて複数のサブスクリプションをサポートするためにこの列を使用します。
TARGET_OWNER	**データ・タイプ**: VARCHAR(30)、DB2 UDB for z/OS バージョン 8 新機能モード・サブシステムの場合 VARCHAR(128)。**NULL 可能**: 不可 このメンバーのターゲット表またはビューの上位修飾子。
TARGET_TABLE	**データ・タイプ**: VARCHAR(128)、DB2 UDB for z/OS バージョン 8 互換モード・サブシステムかそれ以前の場合 VARCHAR(18)。**NULL 可能**: 不可 このメンバーのターゲット表またはビューの名前。
TARGET_CONDENSED	**データ・タイプ**: CHAR(1)。**NULL 可能**: 不可 以下のことを示すフラグ。 **Y** 特定の主キー値に関して、ターゲット表に表示される行は 1 つだけです。 **N** 完全な更新履歴を保持して、すべての変更を残す必要があります。 **A** ターゲット表は、基礎集約表または変更集約表です。

表 92. IBMSNAP_SUBS_MEMBR 表の列 (続き)

列名	説明
TARGET_COMPLETE	**データ・タイプ**: CHAR(1)。**NULL 可能**: 不可 以下のことを示すフラグ。 **Y** ターゲット表は、関係するそれぞれの主キー値につき 1 つの行を保持します。 **N** ターゲット表は、主キー値の行のサブセットを保持します。
TARGET_STRUCTURE	**データ・タイプ**: SMALLINT。**NULL 可能**: 不可 ターゲット表の構造。 **1** ユーザー表 **3** CCD 表 **4** ポイント・イン・タイム表 **5** 基礎集約表 **6** 変更集約表 **7** レプリカ **8** ユーザー・コピー **9** IBMSNAP_UOW と CD 表の結合のない CCD 表
PREDICATES	**データ・タイプ**: VARCHAR(1024)。**NULL 可能**: 可 TARGET_TABLE 列内の表用の WHERE 文節に入れられる述部をリストします。この WHERE 文節は、ソース表の行サブセットを作成します。述部が認識されるのは、WHOS_ON_FIRST が S に設定されている場合だけです。アプライ・プログラムは ORDER BY 文節を生成できないため、述部に ORDER BY 文節を含めることはできません。集約表は、ダミー述部とその後に続く GROUP BY 文節を必要としています。 アプライ・プログラムはフル・リフレッシュおよび変更キャプチャー・レプリケーションの両方でこれらの述部を使用するため、この列に、CD 表または UOW 表の列に関係する述部を含めることはできません。CD 表または UOW 表の参照を含む述部は、 UOW_CD_PREDICATES 列に保管されます。

表 92. IBMSNAP_SUBS_MEMBR 表の列 (続き)

列名	説明
MEMBER_STATE	**データ・タイプ**: CHAR(1)。**NULL 可能**: 可 メンバーの状態を示すフラグ。 **N** (New の略) メンバーはこのサブスクリプション・セットの新規メンバーです。最近使用可能になったメンバーがあれば、それらもこの状態で表示されます。 **L** (Loaded の略) このサブスクリプション・セットのメンバーがロードされましたが、変更キャプチャー・サイクルはまだ発生していません。 **S** (Synchronized の略) メンバーは、New (N) 状態から Loaded (L) 状態に進み、 Synchronized 状態の他のすべてのサブスクリプション・セット・メンバーと同期されています。サブスクリプション・セットのすべてのメンバーが Synchronized 状態の場合は、サブスクリプション・セット・レベルで変更レプリケーションを行うことができます。 **D** (Disabled の略) メンバーはこのサブスクリプション・セットでは使用不可です。
TARGET_KEY_CHG	**データ・タイプ**: CHAR(1)。**NULL 可能**: 不可 ユーザーがソース表で、ターゲット表のターゲット・キー列のソース列を変更したときに、アプライ・プログラムが更新をどのように処理するかを示すフラグ。 **Y** アプライ・プログラムは、ターゲット・キー列の変更前イメージに基づいてターゲット表を更新します。つまり、アプライ・プログラムは述部を新しい値ではなく、古い値に変更します。ターゲット・キーの変更前イメージが CD 表の中に存在するように、変更前イメージの各列を登録してください。登録表の中の対応する登録項目について、CHG_UPD_TO_DEL_INS 列の値が N に設定されていることを確認してください。 **N** アプライ・プログラムは、ターゲット・キーを構成する列が更新されていないという仮定に基づくロジックを使用して、更新および削除を処理します。
UOW_CD_PREDICATES	**データ・タイプ**: VARCHAR(1024)。**NULL 可能**: 可 アプライ・プログラムが変更キャプチャー・レプリケーションにのみ必要とし、フル・リフレッシュには必要としない、CD 表または UOW 表からの列を含む述部が入っています。アプライ・プログラムは変更キャプチャー・レプリケーション時には、この列の述部と、PREDICATES 列の述部を処理します。アプライ・プログラムはフル・リフレッシュ時には、 PREDICATES 列の述部のみを処理します。

表 92. IBMSNAP_SUBS_MEMBR 表の列 (続き)

列名	説明
JOIN_UOW_CD	**データ・タイプ**: CHAR(1)。**NULL 可能**: 可 アプライ・プログラムがユーザー・コピー・ターゲット表を処理するときに、CD 表および UOW 表の結合を行うかどうかを示すフラグ。このフラグは、CD 表の中にない、UOW 表の列を使用する述部を持つサブスクリプション・セット・メンバーをユーザーが定義したときに必要になります。ターゲット表のタイプがユーザー・コピー以外のものである場合には、アプライ・プログラムはメンバーの処理時に CD 表および UOW 表の結合を使用するため、メンバーの処理時にはこの列は無視されます。 **Y** アプライ・プログラムはメンバーの処理時に CD 表および UOW 表の結合を使用します。 **N** アプライ・プログラムはメンバーの処理時に CD 表および UOW 表の結合を使用しません。CD 表からのみ変更が読み取られます。 **NULL** アプライ・プログラムはメンバーの処理時にこの列を無視します。ターゲット表がユーザー・コピーであり、この列の値が NULL である場合には、アプライ・プログラムはメンバーの処理時に CD 表および UOW 表を結合しません。
LOADX_TYPE	**データ・タイプ**: SMALLINT。**NULL 可能**: 可 このメンバーのロードのタイプ。この列の値は、デフォルトのオーバーライドに使用されます。 **NULL** z/OS LOAD FROM CURSOR 関数 (DB2 Utilities Suite で使用可能) が、このメンバーに対して使用されます。 Linux UNIX Windows ASNLOAD 出口が、このメンバーに最適なユーティリティーを決定します。(オプション 3、4 または 5) **1** このメンバーには ASNLOAD は使用されません。これにより、ユーザーが始動時に LOADX を指定した場合にでも、特定のサブスクリプション・セット・メンバーについて ASNLOAD オプションがオフになります。 **2** ユーザー定義の、またはユーザーが変更した ASNLOAD 終了コードが使用されます。 **3** このメンバーに対して LOAD FROM CURSOR 関数が使用されます。 Linux UNIX Windows **4** このメンバーには EXPORT および LOAD が使用されます。 Linux UNIX Windows **5** このメンバーには EXPORT および IMPORT が使用されます。 **制約事項:** Linux UNIX Windows LOAD ユーティリティーは、範囲クラスター表の場合はサポートされません。範囲クラスター表のフル・リフレッシュを行うためには、DB2 の IMPORT ユーティリティーを使用するか、アプライ・プログラムを使用して表のフル・リフレッシュを SQL を介して行います。

表 92. IBMSNAP_SUBS_MEMBR 表の列 (続き)

列名	説明
LOADX_SRC_N_OWNER	**データ・タイプ**: VARCHAR(30)。**NULL 可能**: 可 ユーザーが作成したニックネームの所有者。以下のすべての条件が揃っている場合、この値は必須です。 • このメンバーに対して LOAD FROM CURSOR 関数が使用される (LOADX_TYPE は 3)。 • ターゲット・サーバーは Linux、UNIX、または Windows である。 • ソースはニックネームではない。
LOADX_SRC_N_TABLE	**データ・タイプ**: VARCHAR(128)。**NULL 可能**: 可 ユーザーが作成したニックネーム表。以下のすべての条件が揃っている場合、この値は必須です。 • このメンバーに対して LOAD FROM CURSOR 関数が使用される (LOADX_TYPE は 3)。 • ターゲット・サーバーは Linux、UNIX、または Windows である。 • ソースはニックネームではない。

ASN.IBMSNAP_SUBS_SET 表

IBMSNAP_SUBS_SET 表には、アプライ・コントロール・サーバーで定義されたすべてのサブスクリプション・セットが記載され、これらのセットのレプリケーションの進行状況が文書化されます。行は、サブスクリプション・セット定義を作成したときに自動的にこの表に挿入されます。

サーバー: アプライ・コントロール・サーバー

索引: APPLY_QUAL、 SET_NAME、WHOS_ON_FIRST

重要: SQL を使用してこの表を更新するときには、注意してください。この表の変更の方法が不適切であると、予期せぬ結果が生じたり、データが失われたりします。

表 93 では、 IBMSNAP_SUBS_SET 表の列の要旨を示します。

表 93. IBMSNAP_SUBS_SET 表の列

列名	説明
APPLY_QUAL	**データ・タイプ**: CHAR(18)。**NULL 可能**: 不可 このサブスクリプション・セットを処理するアプライ・プログラムを一意的に識別します。
SET_NAME	**データ・タイプ**: CHAR(18)。**NULL 可能**: 不可 サブスクリプション・セットの名前。

表 93. IBMSNAP_SUBS_SET 表の列 (続き)

列名	説明
SET_TYPE	**データ・タイプ**: CHAR(1)。**NULL 可能**: 不可 セットが読み取り専用か読み取り/書き込みかを示すフラグ。 **R** セットは読み取り専用です。 **U** セットは Update-anywhere 構成であるため、読み取り/書き込みです。
WHOS_ON_FIRST	**データ・タイプ**: CHAR(1)。**NULL 可能**: 不可 Update-anywhere レプリケーション・シナリオでは、処理順序をコントロールするために以下の値を使用します。 **F** (first の略) ソース表がレプリカであり、ターゲット表がマスターです。レプリカ表とマスター表の間で更新の矛盾が生じた場合、レプリカの側の矛盾するトランザクションはリジェクトされます。F は読み取り専用のサブスクリプションでは使用されません。Update-anywhere で使用されるものです。 **S** (second の略) ソース表はマスター表またはその他のソースであり、ターゲット表はレプリカまたはその他のコピーです。マスター表とレプリカ表の間で更新の矛盾が生じた場合、レプリカの側の矛盾するトランザクションはリジェクトされます。S は、すべての読み取り専用サブスクリプションについて使用されます。
ACTIVATE	**データ・タイプ**: SMALLINT。**NULL 可能**: 不可 アプライ・プログラムが次のサイクルでこのセットを処理するかどうかを示すフラグ。 **0** サブスクリプション・セットは非活動化されています。アプライ・プログラムはこのセットを処理しません。 **1** サブスクリプション・セットは無期限にアクティブです。アプライ・プログラムは、ユーザーがセットを非活動化するか、アプライ・プログラムが処理を行えなくなるまで、各アプライ・サイクルでセットを処理します。 **2** サブスクリプション・セットは、 1 つのアプライ・サイクルでのみアクティブです。アプライ・プログラムはセットを一度処理した後、セットを非活動化します。
SOURCE_SERVER	**データ・タイプ**: CHAR(18)。**NULL 可能**: 不可 ソース表およびビューが定義されている、キャプチャー・コントロール・サーバーのデータベース名。
SOURCE_ALIAS	**データ・タイプ**: CHAR(8)。**NULL 可能**: 可 SOURCE_SERVER 列で指定されているキャプチャー・コントロール・サーバーに対応する DB2 別名。
TARGET_SERVER	**データ・タイプ**: CHAR(18)。**NULL 可能**: 不可 ターゲットの表またはビューが保管されているサーバーのデータベース名。
TARGET_ALIAS	**データ・タイプ**: CHAR(8)。**NULL 可能**: 可 TARGET_SERVER 列で指定されているターゲット・サーバーに対応する DB2 別名。

表 93. IBMSNAP_SUBS_SET 表の列 (続き)

列名	説明
STATUS	**データ・タイプ**: SMALLINT。**NULL 可能**: 不可 特定のサイクル後のアプライ・プログラムの作業状況を表す値。 **-1** レプリケーションは失敗しました。アプライ・プログラムは適用済みの行のセット全体をバックアウトし、データはコミットされません。始動パラメーターが SQLERRCONTINUE = Y の場合、最後のサイクル中にアプライ・プログラムに戻される SQLSTATE は、SQLERRCONTINUE (*apply qualifier*.SQS) の入力ファイルでユーザーが指定した許容エラーの 1 つではありません。 **0** アプライ・プログラムはサブスクリプション・セットを正常に処理しました。始動パラメーターが SQLERRCONTINUE = Y の場合、アプライ・プログラムは、ユーザーから SQLERRCONTINUE 始動パラメーターで (*apply qualifier*.SQS で) 指定された SQL エラーを検出しておらず、行をリジェクトしていません。 **2** アプライ・プログラムはサブスクリプション・セットを複数のサイクルで処理しています。アプライ・プログラムは、MAX_SYNCH_MINUTES コントロール列に従って分割された 1 つの論理サブスクリプションを正常に処理しました。 **16** アプライ・プログラムはサブスクリプション・セットを正常に処理し、0 という状況を戻しました。しかしアプライ・プログラムは、ユーザーが SQLERRCONTINUE 始動パラメーターで (*apply_qualifier*.SQS で) 指定したいくつかの SQL エラーを検出したため、いくつかの行をリジェクトしました。失敗した行の詳細は、*apply qualifier*.ERR ファイルで確認してください。 **例**: ユーザーは SQLERRCONTINUE = Y と設定し、SQL の許容される状態を 23502 (SQL コード -407) と指定します。23502 エラーが発生しますが、他のエラーは発生していません。アプライ・プログラムはサブスクリプション・セットの処理を終了し、状態を 16 に設定します。次の実行時に、23502 エラーが発生した後、07006 (SQL コード -301) が発生します。アプライ・プログラムは今回は、サブスクリプション・セットの処理を停止し、適用済みの行のセット全体をバックアウトし、状況を -1 に設定します (データはコミットされていないため)。 **18** アプライ・プログラムは複数のサイクルでサブスクリプション・セットを処理し、 2 という状況を戻しています。これは、MAX_SYNCH_MINUTES コントロール列に従って分割された 1 つの論理サブスクリプションが正常に処理されたことを意味します。しかし、ユーザーが SQLERRCONTINUE 始動パラメーターで (*apply_qualifier*.SQS で) 指定した SQL エラーのいくつかが検出されたため、いくつかの行がリジェクトされています。失敗した行の詳細は、*apply_qualifier*.ERR ファイルで確認してください。
LASTRUN	**データ・タイプ**: TIMESTAMP。**NULL 可能**: 不可 サブスクリプション・セットが最後に開始された概算の時刻。アプライ・プログラムは、サブスクリプション・セットを処理するたびに LASTRUN 値を設定します。これは、アプライ・プログラムがサブスクリプション・セットの処理を開始する、アプライ・コントロール・サーバーにおけるおおよその時刻です。

表 93. IBMSNAP_SUBS_SET 表の列 (続き)

列名	説明
REFRESH_TYPE	**データ・タイプ**: CHAR(1)。**NULL 可能**: 不可 アプライ・プログラムにこのサブスクリプション・セットの処理を促すために使用されるスケジューリングのタイプ。 **R** アプライ・プログラムは時間ベースのスケジューリングを使用します。アプライ・プログラムは、 SLEEP_MINUTES の中の値を使用して、サブスクリプション・セットの処理を開始する時間を判断します。 **E** アプライ・プログラムはイベント・ベースのスケジューリングを使用します。アプライ・プログラムは IBMSNAP_SUBS_EVENT 表の中の時刻値を確認して、サブスクリプション・セットの処理を開始する時間を判断します。イベントが生じないと、レプリケーション (変更のキャプチャーまたはフル・リフレッシュ) を何も開始できません。 **B** アプライ・プログラムは時間ベースのスケジューリングとイベント・ベースのスケジューリングの両方を使用します。このため、アプライ・プログラムは時間またはイベントの基準に基づいてサブスクリプション・セットを処理します。
SLEEP_MINUTES	**データ・タイプ**: INT。**NULL 可能**: 可 サブスクリプション・セット処理の間の非活動時間 (分単位) を指定します。 処理時間は、REFRESH_TYPE が R または B の場合にのみ使用されます。SLEEP_MINUTES の値が NULL の場合は、アプライ・プログラムが継続してセットを処理します。アプライ・プログラムは、セットを可能な限り頻繁に処理しますが、また同じアプライ修飾子で、その他すべてのアクティブなサブスクリプション・セットも処理します。
EVENT_NAME	**データ・タイプ**: CHAR(18)。**NULL 可能**: 可 イベントの名前を表すユニークな文字ストリング。この ID は、サブスクリプション・セットに対してレプリケーションを起動したい場合にサブスクリプション・イベント表を更新するのに使用します。イベント名は、 REFRESH_TYPE が E または B の場合にのみ使用されます。
LASTSUCCESS	**データ・タイプ**: TIMESTAMP。**NULL 可能**: 可 サブスクリプション・セットが最後に正常に処理されたときの、処理開始時点のアプライ・コントロール・サーバーのタイム・スタンプ。
SYNCHPOINT	**データ・タイプ**: CHAR(10) FOR BIT DATA。**NULL 可能**: 可 アプライ・プログラムはこの列を使用して進行状況を記録します。サブスクリプション・セットのデータの処理が、この同期点の値まで終了していることを示します。
SYNCHTIME	**データ・タイプ**: TIMESTAMP。**NULL 可能**: 可 アプライ・プログラムはこの列を使用して進行状況を記録します。サブスクリプション・セットのデータの処理が、このタイム・スタンプまで終了していることを示します。

表 93. IBMSNAP_SUBS_SET 表の列 (続き)

列名	説明
CAPTURE_SCHEMA	**データ・タイプ**: VARCHAR(30)、DB2 UDB for z/OS バージョン 8 新機能モード・サブシステムの場合 VARCHAR(128)。**NULL 可能**: 不可 このサブスクリプション・セットのソースを処理するキャプチャー・コントロール表のスキーマ名。
TGT_CAPTURE_SCHEMA	**データ・タイプ**: VARCHAR(30)、DB2 UDB for z/OS バージョン 8 新機能モード・サブシステムの場合 VARCHAR(128)。**NULL 可能**: 可 ターゲット表が別のサブスクリプション・セットのソースでもある場合 (multi-tier 構成中の外部の CCD 表、または Update-anywhere 構成中のレプリカ表など) は、この列には、表がソースとして機能するときに使用されるキャプチャー・スキーマが含まれます。
FEDERATED_SRC_SRVR	**データ・タイプ**: VARCHAR(18)。**NULL 可能**: 可 DB2 以外のリレーショナル・ソースの場合にのみアプライされる、サブスクリプション・セットのソースである、フェデレーテッド・リモート・サーバーの名前。
FEDERATED_TGT_SRVR	**データ・タイプ**: VARCHAR(18)。**NULL 可能**: 可 DB2 以外のリレーショナル・ターゲットの場合にのみアプライされる、サブスクリプション・セットのターゲットである、フェデレーテッド・リモート・サーバーの名前。
JRN_LIB	**データ・タイプ**: CHAR(10)。**NULL 可能**: 可 System i System i キャプチャー・サーバーにのみアプライされるこの列は、ソース表が使用するジャーナルのライブラリー名です。
JRN_NAME	**データ・タイプ**: CHAR(10)。**NULL 可能**: 可 System i System i キャプチャー・サーバーにのみアプライされるこの列は、ソース表が使用するジャーナルの名前です。この列で、アスタリスクの後ろに 9 つのブランクが続くときには、ソース表が現在ジャーナルの中にないことを意味します。この場合は、このソース表のデータをキャプチャーすることはできません。
OPTION_FLAGS	**データ・タイプ**: CHAR(4)。**NULL 可能**: 不可 SQL レプリケーションの将来のオプション用に予約済み。この列には現在、デフォルト値の NNNN が入っています。

表 93. IBMSNAP_SUBS_SET 表の列 (続き)

列名	説明
COMMIT_COUNT	**データ・タイプ**: SMALLINT。**NULL 可能**: 可 アプライ・プログラムがサブスクリプション・セットに対して実行する処理のタイプを示すフラグ。 **NULL** これは、読み取り専用のサブスクリプション・セットの場合のデフォルト設定です。アプライ・プログラムは、すべてのデータが処理されるまで、*n* のサブスクリプション・セット・メンバーのメンバー 1 つずつについて、フェッチした応答セットを処理していき、セット全体のデータ処理が終了すると最後に、1 つのコミットを発行します。この COMMIT_COUNT 設定値を使用する利点は、処理が早く終了する可能性があることです。 *非 NULL の整数* アプライ・プログラムはトランザクションのモードでサブスクリプション・セットを処理します。すべての応答セットがフェッチされると、各トランザクションが IBMSNAP_INTENTSEQ 値の順に並べられ、コミット・シーケンスの順番で予備ファイルの内容が適用されます。このタイプの処理では、すべての予備ファイルを同時に開いて処理できます。この列で指定された数のトランザクションの後に、コミットが発行されます。例えば、1 は各トランザクションの後でコミットすることを意味し、2 は、トランザクション 2 つごとにコミットすることを意味します。0 という整数は、すべてのフェッチ・データがアプライされた後で 1 つのコミットを発行することを意味します。トランザクションのモードの処理を使用する利点は、この処理ではターゲットでの参照整合性制約が可能であり、暫定のコミットを発行できることです。
MAX_SYNCH_MINUTES	**データ・タイプ**: SMALLINT。**NULL 可能**: 可 サブスクリプション・サイクルでフェッチおよびアプライする変更データの量を規制するための時間しきい値限度。アプライ・プログラムはキャプチャー・サーバーの UOW 表または CCD 表の中の IBMSNAP_LOGMARKER 列に基づいてサブスクリプション・セットの処理をミニサイクルに分割し、ミニサイクルが正常に終了するたびに、ターゲット・サーバーで COMMIT を発行します。設定された限度が不適切になるようなリソース制約をアプライ・プログラムが検出した場合、この限度は自動的に再計算されます。1 より小さい MAX_SYNCH_MINUTES 値は、NULL の MAX_SYNCH_MINUTES 値と同じものとして処理されます。
AUX_STMTS	**データ・タイプ**: SMALLINT。**NULL 可能**: 不可 IBMSNAP_SUBS_STMTS 表で定義する SQL ステートメントの数。これらのステートメントは、アプライ・プログラムがサブスクリプション・セットを処理する前または後に実行可能です。

表 93. IBMSNAP_SUBS_SET 表の列 (続き)

列名	説明
ARCH_LEVEL	**データ・タイプ**: CHAR(4)。**NULL 可能**: 不可 レプリケーション・コントロール表の構造レベル。この列は、行を作成する基礎となった規則を識別します。このレベルは、IBM で定義されています。 **0801** バージョン 8 以降の SQL レプリケーション **0803** バージョン 8 SQL レプリケーション (Oracle ソースの拡張サポート付き) **0805** バージョン 8 SQL レプリケーション (DB2 for z/OS 新機能モードのサポート付き)

ASN.IBMSNAP_SUBS_STMTS 表

IBMSNAP_SUBS_STMTS 表には、各サブスクリプション・セット処理サイクルの前または後に実行される、ユーザー定義の SQL ステートメントまたはストアード・プロシージャー呼び出しが入っています。即時実行 (EI) ステートメントまたはストアード・プロシージャーは、ソースまたはターゲット・サーバーでのみ実行できます。この表には、SQL ステートメントまたはストアード・プロシージャー呼び出しを使うサブスクリプション・セットを定義するときに、値が挿入されます。

サーバー: アプライ・コントロール・サーバー

索引: APPLY_QUAL、SET_NAME、WHOS_ON_FIRST、BEFORE_OR_AFTER、STMT_NUMBER

重要: SQL を使用してこの表を更新するときには、注意してください。この表の変更の方法が不適切であると、予期せぬ結果が生じたり、データが失われたりします。サブスクリプションの項目数は、IBMSNAP_SUBS_SET 表の AUX_STMTS 列に反映されていなければなりません。あるサブスクリプション・セットの AUX_STMTS がゼロの場合、アプライ・プログラムは IBMSNAP_SUBS_STMTS 表の対応する項目を無視します。

表 94 では、 IBMSNAP_SUBS_STMTS 表の列の要旨を示します。

表 94. IBMSNAP_SUBS_STMTS 表の列

列名	説明
APPLY_QUAL	**データ・タイプ**: CHAR(18)。**NULL 可能**: 不可 SQL ステートメントまたはストアード・プロシージャーを処理するアプライ・プログラムを一意的に識別します。
SET_NAME	**データ・タイプ**: CHAR(18)。**NULL 可能**: 不可 SQL ステートメントまたはストアード・プロシージャーが関連付けられているサブスクリプション・セットの名前。

表 94. IBMSNAP_SUBS_STMTS 表の列 (続き)

列名	説明
WHOS_ON_FIRST	**データ・タイプ**: CHAR(1)。**NULL 可能**: 不可 Update-anywhere レプリケーション・シナリオでは、処理順序をコントロールするために以下の値を使用します。 **F** (first の略) ターゲット表はユーザー表または親レプリカです。ソース表は従属のレプリカであり、ソース表とターゲット表との間で更新の競合が生じた場合、ソース表の競合するトランザクションはリジェクトされます。F は、読み取り専用サブスクリプションには使用されません。 **S** (second の略) ソース表はユーザー表、親レプリカ、またはその他のソースです。ターゲット表は従属のレプリカまたは他のコピーであり、ソース表とターゲット表との間で更新の競合が生じた場合、ターゲット表の競合するトランザクションはリジェクトされます。 S は、すべての読み取り専用サブスクリプションについて使用されます。
BEFORE_OR_AFTER	**データ・タイプ**: CHAR(1)。**NULL 可能**: 不可 ステートメントの発行時間と場所を示す値。 **A** ステートメントは、すべての応答セット行が適用された後、ターゲット・サーバーで実行されます。 **B** ステートメントは、応答セット行が適用される前に、ターゲット・サーバーで実行されます。 **S** ステートメントは、応答セット・カーソルを開く前に、キャプチャー・コントロール・サーバー上で実行されます。 **G** SQL レプリケーションの使用のために予約済み。 **X** SQL レプリケーションの使用のために予約済み。
STMT_NUMBER	**データ・タイプ**: SMALLINT。**NULL 可能**: 不可 BEFORE_OR_AFTER 列値の有効範囲で、実行の相対順序を定義します。
EI_OR_CALL	**データ・タイプ**: CHAR(1)。**NULL 可能**: 不可 次のどちらであるかを示す値。 **E** SQL ステートメントは、EXEC SQL EXECUTE IMMEDIATE として実行されます。 **C** SQL ステートメントには、EXEC SQL CALL として実行されるストアード・プロシージャー名が入っています。
SQL_STMT	**データ・タイプ**: VARCHAR(1024)。**NULL 可能**: 可 以下のいずれかの値が入ります。 **ステートメント** EI_OR_CALL = E の場合に EXEC SQL EXECUTE IMMEDIATE ステートメントとして実行される SQL ステートメント。 **手順** EI_OR_CALL が C の場合に、EXEC SQL CALL ステートメントとして実行される、パラメーター、または CALL キーワードなしの SQL ストアード・プロシージャーの 8 バイトの名前。

表 94. IBMSNAP_SUBS_STMTS 表の列 (続き)

列名	説明
ACCEPT_SQLSTATES	**データ・タイプ**: VARCHAR(50)。**NULL 可能**: 可 サブスクリプション・セットの定義時に指定した 1〜10 個の 5 バイト SQLSTATE 値。これらの非ゼロ値は、正常実行としてアプライ・プログラムで受け入れられます。それ以外の値は実行が失敗する原因となります。

モニター・コントロール・サーバーのコントロール表

モニター・コントロール・サーバー上のコントロール表には、アラート条件が発生したときに、レプリケーション・アラート・モニターから、いつ、どのように、そしてだれに連絡するかに関する情報が含まれます。Linux、UNIX、Windows、z/OS の場合は、レプリケーション・センターを使用して、ユーザーの指定に合わせてこれらのコントロール表を作成します。System i でのレプリケーションの場合、モニター・コントロール表はありません。

表 95では、モニター・コントロール・サーバーのコントロール表について説明します。

表 95. モニター・コントロール・サーバーのコントロール表

表名	説明
495 ページの『IBMSNAP_ALERTS 表』	レプリケーション・アラート・モニターから発行されるすべてのアラートのレコードを保持します。
496 ページの『IBMSNAP_CONDITIONS 表』	レプリケーション・アラート・モニターから担当者への連絡が必要なアラート条件と、特定の条件が発生したときの連絡先のグループまたは個人の名前が入っています。
503 ページの『IBMSNAP_CONTACTGRP 表』	連絡先グループを構成する個人の連絡先が入っています。
504 ページの『IBMSNAP_CONTACTS 表』	連絡先名に関連付けられたアラート条件が発生したときに、レプリケーション・アラート・モニターから各個人またはグループに通知する方法に関する情報が入っています。
505 ページの『IBMSNAP_GROUPS 表』	各連絡先グループの名前と記述が入っています。
505 ページの『IBMSNAP_MONENQ 表』	1 つのモニター修飾子に対して 1 つのレプリケーション・アラート・モニター・プログラムだけが確実に実行されるようにするために使用されます。
505 ページの『IBMSNAP_MONPARMS 表』	モニター・プログラムの操作をコントロールするためにユーザーが変更できるパラメーターが含まれています。
508 ページの『IBMSNAP_MONSERVERS 表』	サーバーがレプリケーション・アラート・モニター・プログラム (モニター修飾子によって識別される) からモニターされた最後の時刻を保持します。
509 ページの『IBMSNAP_MONTRACE 表』	モニター・プログラムからのメッセージを保持します。

表 95. モニター・コントロール・サーバーのコントロール表 (続き)

表名	説明
510 ページの『IBMSNAP_MONTRAIL 表』	各モニター・サイクルに関する情報を保持します。
512 ページの『IBMSNAP_SUSPENDS 表』	モニター・プログラムの一時中断に関する情報を保持します。
513 ページの『IBMSNAP_TEMPLATES 表』	モニター・プログラムの中断の頻度と長さに関する情報を保持します。

IBMSNAP_ALERTS 表

IBMSNAP_ALERTS 表は、レプリケーション・アラート・モニターから発行されるすべてのアラートのレコードを保持します。この表には、どのようなアラート条件が発生したか、どのサーバーで発生したか、そしていつアラートが検出されたかが記録されます。

サーバー: モニター・コントロール・サーバー

非ユニーク索引: MONITOR_QUAL、 COMPONENT、 SERVER_NAME、 SCHEMA_OR_QUAL、 SET_NAME、 CONDITION_NAME、 ALERT_CODE

表 96では、IBMSNAP_ALERTS 表の列の要旨を示します。

表 96. IBMSNAP_ALERTS 表の列

列名	説明
MONITOR_QUAL	**データ・タイプ:** CHAR(18)。**NULL 可能:** 不可。 アラートを発行したレプリケーション・アラート・モニター・プログラムを示すモニター修飾子。
COMPONENT	**データ・タイプ:** CHAR(1)。**NULL 可能:** 不可。 モニター対象のレプリケーション・コンポーネント。 **C** キャプチャー・プログラム **A** アプライ・プログラム **S** Q キャプチャー・プログラム **R** Q アプライ・プログラム
SERVER_NAME	**データ・タイプ:** CHAR(18)。**NULL 可能:** 不可。 アラート条件が発生した、キャプチャー・コントロール・サーバー、アプライ・コントロール・サーバー、Q キャプチャー・サーバー、または Q アプライ・サーバーの名前。
SERVER_ALIAS	**データ・タイプ:** CHAR(8)。**NULL 可能:** 可。 アラート条件が発生した、キャプチャー・コントロール・サーバー、アプライ・コントロール・サーバー、Q キャプチャー・サーバー、または Q アプライ・サーバーの DB2 別名。

表 96. IBMSNAP_ALERTS 表の列 (続き)

列名	説明
SCHEMA_OR_QUAL	**データ・タイプ:** VARCHAR(128)。**NULL 可能:** 不可。 モニター対象のキャプチャー・スキーマ、アプライ・スキーマ、Q キャプチャー・スキーマ、または Q アプライ・スキーマ。
SET_NAME	**データ・タイプ:** CHAR(18)。**NULL 可能:** 不可。デフォルトあり。**デフォルト:** 現行サブスクリプション・セット。 アプライ・プログラムでアラート条件が設定されている場合、この列は、モニター対象のサブスクリプション・セットの名前を指定します。セット名を指定しないと、アプライ修飾子レベルでモニターが行われます。つまり、特定のアプライ修飾子内のすべてのセットがモニターされます。 Q アプライの受信キュー項目数または予備キュー項目数のアラート条件が設定されている場合、この列は、モニター対象の受信キューまたは予備キューの名前を指定します。
CONDITION_NAME	**データ・タイプ:** CHAR(18)。**NULL 可能:** 不可。 アラートが起動されたときにテストされた条件コード。
OCCURRED_TIME	**データ・タイプ:** TIMESTAMP。**NULL 可能:** 不可。 キャプチャー・コントロール・サーバー、アプライ・コントロール・サーバー、Q キャプチャー・サーバー、または Q アプライ・サーバーでアラート条件が発生した時刻。
ALERT_COUNTER	**データ・タイプ:** SMALLINT。**NULL 可能:** 不可。 連続するモニター・サイクルで、このアラートが以前に検出された回数。
ALERT_CODE	**データ・タイプ:** CHAR(10)。**NULL 可能:** 不可。 アラートの発生時に発行されたメッセージ・コード。
RETURN_CODE	**データ・タイプ:** INT。**NULL 可能:** 不可。 ユーザー条件から戻された整数値。
NOTIFICATION_SENT	**データ・タイプ:** CHAR(1)。**NULL 可能:** 不可。 通知メッセージが送信されたかどうかを示すフラグ。 **Y** 通知メッセージは送信されました。 **E** **email_server** パラメーターが指定されていなかったため、通知が送信されませんでした。 **N** 通知の数が **max_notifications_per_alert** パラメーターで設定された限度にすでに達しているため、通知が送信されませんでした。
ALERT_MESSAGE	**データ・タイプ:** VARCHAR(1024)。**NULL 可能:** 不可。 メッセージ・コードを含む、送信されたメッセージのテキスト。

IBMSNAP_CONDITIONS 表

IBMSNAP_CONDITIONS 表には、レプリケーション・アラート・モニターから担当者への連絡が必要なアラート条件と、特定の条件が発生したときの連絡先のグループまたは個人の名前が含まれます。レプリケーション・アラート・モニターは、キ

ャプチャー・コントロール・サーバー、アプライ・コントロール・サーバー、Q キャプチャー・サーバー、および Q アプライ・サーバー上の複数の条件の組み合わせをモニターできます。

サーバー: モニター・コントロール・サーバー

非ユニーク索引: MONITOR_QUAL、 COMPONENT、 SERVER_NAME、 SCHEMA_OR_QUAL、 SET_NAME、 CONDITION_NAME

表 97 では、IBMSNAP_CONDITIONS 表の列の要旨を示します。

表 97. IBMSNAP_CONDITIONS 表の列

列名	説明
SERVER_NAME	**データ・タイプ**: CHAR(18)。**NULL 可能**: 不可。 この条件がモニターされるキャプチャー・コントロール・サーバー、アプライ・コントロール・サーバー、Q キャプチャー・サーバー、または Q アプライ・サーバーの名前。
COMPONENT	**データ・タイプ**: CHAR(1)。**NULL 可能**: 不可。 モニター対象のレプリケーション・コンポーネント。 **C** キャプチャー・プログラム **A** アプライ・プログラム **S** Q キャプチャー・プログラム **R** Q アプライ・プログラム
SCHEMA_OR_QUAL	**データ・タイプ:** VARCHAR(128)。**NULL 可能**: 不可。 モニター対象のキャプチャー・スキーマ、アプライ・スキーマ、Q キャプチャー・スキーマ、または Q アプライ・スキーマ。
SET_NAME	**データ・タイプ**: CHAR(18)。**NULL 可能**: 不可。**デフォルト**: 現行サブスクリプション・セット。 アプライ・プログラムでアラート条件が設定されている場合、この列は、モニター対象のサブスクリプション・セットの名前を指定します。セット名を指定しないと、アプライ修飾子レベルでモニターが行われます。つまり、特定のアプライ修飾子内のすべてのセットがモニターされます。
MONITOR_QUAL	**データ・タイプ**: CHAR(18)。**NULL 可能**: 不可。 この条件について、キャプチャー・コントロール・サーバー、アプライ・コントロール・サーバー、Q キャプチャー・サーバー、または Q アプライ・サーバーをモニターしているレプリケーション・アラート・モニター・プログラムを識別するモニター修飾子。
SERVER_ALIAS	**データ・タイプ**: CHAR(8)。**NULL 可能**: 可。 この条件がモニターされるキャプチャー・コントロール・サーバー、アプライ・コントロール・サーバー、Q キャプチャー・サーバー、または Q アプライ・サーバーの DB2別名。

表 97. IBMSNAP_CONDITIONS 表の列 (続き)

列名	説明
ENABLED	**データ・タイプ**: CHAR(1)。**NULL 可能**: 不可。 レプリケーション・アラート・モニターが次のモニター・サイクルでこの条件を処理するかどうかを示すフラグ。 **Y** レプリケーション・アラート・モニターは次のモニター・サイクルでこの定義を処理します。 **N** レプリケーション・アラート・モニターは次のモニター・サイクルでこの定義を無視します。
CONDITION_NAME	**データ・タイプ**: CHAR(18)。**NULL 可能**: 不可。 特定のキャプチャー・コントロール・サーバー、アプライ・コントロール・サーバー、Q キャプチャー・サーバー、または Q アプライ・サーバーで、レプリケーション・アラート・モニターがモニターしている条件の名前。キャプチャー・プログラムの条件は CAPTURE で始まります。アプライ・プログラムの条件は APPLY で始まります。Q キャプチャー・プログラムの条件は QCAPTURE で始まります。 Q アプライ・プログラムの条件は QAPPLY で始まります。 **CAPTURE_STATUS** キャプチャー・プログラムの状況。 **CAPTURE_ERRORS** キャプチャー・プログラムがエラー・メッセージを通知したかどうか。 **CAPTURE_WARNINGS** キャプチャー・プログラムが警告メッセージを通知したかどうか。 **CAPTURE_LASTCOMMIT** 最後のモニター・サイクルでキャプチャー・プログラムが最後にデータをコミットした時刻。 **CAPTURE_CLATENCY** キャプチャー・プログラムの現在の待ち時間。 **CAPTURE_HLATENCY** キャプチャー・プログラムの待ち時間が特定の秒数を超えるかどうか。 **CAPTURE_MEMORY** キャプチャー・プログラムが使用しているメモリーの量 (MB)。

表 97. IBMSNAP_CONDITIONS 表の列 (続き)

列名	説明
CONDITION_NAME (続き)	**APPLY_STATUS** アプライ・プログラムの状況。 **APPLY_SUBSFAILING** 失敗したサブスクリプション・セットがあるかどうか。 **APPLY_SUBSINACT** 失敗した、または非アクティブのサブスクリプション・セットがあるかどうか。 **APPLY_ERRORS** アプライ・プログラムがエラー・メッセージを通知するかどうか。 **APPLY_WARNINGS** アプライ・プログラムが警告メッセージを通知するかどうか。 **APPLY_FULLREFRESH** フル・リフレッシュが発生したかどうか。 **APPLY_REJTRANS (update anywhere)** アプライ・プログラムがサブスクリプション・セットでトランザクションをリジェクトするかどうか。 **APPLY_SUBSDELAY** アプライ・プログラムが、ユーザーが PARM_INT パラメーターで指定した時間よりも遅れるかどうか。 **APPLY_REWORKED** アプライ・プログラムが、ターゲット表の行の再処理を行ったかどうか。 **APPLY_LATENCY** アプライ・プログラムのエンドツーエンドの待ち時間がしきい値を超えるかどうか。

表 97. IBMSNAP_CONDITIONS 表の列 (続き)

列名	説明
CONDITION_NAME (続き)	**QCAPTURE_STATUS** Q キャプチャー・プログラムがダウンしているかどうか。 **QCAPTURE_ERRORS** Q キャプチャー・プログラムがエラー・メッセージを通知したかどうか。 **QCAPTURE_WARNINGS** Q キャプチャー・プログラムが警告メッセージを通知したかどうか。 **QCAPTURE_LATENCY** Q キャプチャーの待ち時間 (IBMQREP_CAPMON 表への最後の挿入と、Q キャプチャー・プログラムが DB2 ログ中で読み取る最後のトランザクションのタイム・スタンプとの間の差) が、しきい値を超えるかどうか。 **QCAPTURE_MEMORY** Q キャプチャー・プログラムが使用するメモリーがしきい値を超えるかどうか。 **QCAPTURE_TRANSIZE** トランザクションが、 IBMQREP_CAPMON 表に設定されている MAX_TRANS_SIZE (トランザクションの最大サイズ) を超えるかどうか。 **QCAPTURE_SUBSINACT** Q サブスクリプションが I (非アクティブ) 状態に変更されたかどうか。

表 97. IBMSNAP_CONDITIONS 表の列 (続き)

列名	説明
CONDITION_NAME (続き)	**QAPPLY_STATUS** Q アプライ・プログラムがダウンしているかどうか。 **QAPPLY_ERRORS** Q アプライ・プログラムがエラー・メッセージを通知したかどうか。 **QAPPLY_WARNINGS** Q アプライ・プログラムが警告メッセージを通知したかどうか。 **QAPPLY_LATENCY** キューの待ち時間 (メッセージが送信キューから受信キューに送られるのに要する時間) が、しきい値を超えるかどうか。 **QAPPLY_EELATENCY** エンドツーエンドの待ち時間 (トランザクションがソースからターゲットに複製されるのに要する時間) が、しきい値を超えるかどうか。 **QAPPLY_EXCEPTIONS** SQL エラーか競合のために、Q アプライが行を IBMQREP_EXCEPTIONS 表に挿入したかどうか。 **QAPPLY_MEMORY** Q アプライ・プログラムが特定の受信キューからメッセージを読み取るのに使用したメモリーの量が、しきい値を超えるかどうか。 **QAPPLY_RECVQINACT** 受信キューが I (非アクティブ) 状態に変更されたかどうか。 **QAPPLY_SPILLQDEPTH** 予備キュー上のメッセージの数がしきい値を超えるかどうか。 **QAPPLY_QDEPTH** 受信キュー上のメッセージの数がしきい値を超えるかどうか。

表 97. IBMSNAP_CONDITIONS 表の列 (続き)

列名	説明
PARM_INT	**データ・タイプ**: INT。**NULL 可能**: 可。 条件の整数パラメーター。この列の値は、CONDITION_NAME 列の値によって異なります。 **CAPTURE_LASTCOMMIT** しきい値 (秒単位)。 **CAPTURE_CLATENCY** しきい値 (秒単位)。 **CAPTURE_HLATENCY** しきい値 (秒単位)。 **CAPTURE_MEMORY** しきい値 (MB 単位)。 **APPLY_SUBSDELAY** しきい値 (秒単位)。 **APPLY_REWORKED** 再処理される行のしきい値。 **APPLY_LATENCY** しきい値 (秒単位)。 **QCAPTURE_LATENCY** しきい値 (秒単位) **QCAPTURE_MEMORY** しきい値 (MB 単位) **QCAPTURE_TRANSIZE** しきい値 (MB 単位) **QAPPLY_EELATENCY** しきい値 (秒単位) **QAPPLY_LATENCY** しきい値 (秒単位) **QAPPLY_MEMORY** しきい値 (MB 単位) **QAPPLY_SPILLQDEPTH** メッセージ数のしきい値。 **QAPPLY_QDEPTH** メッセージ数のしきい値。

表 97. IBMSNAP_CONDITIONS 表の列 (続き)

列名	説明
PARM_CHAR	**データ・タイプ**: VARCHAR(128)。**NULL 可能**: 可。 条件の文字パラメーター。この列には、条件が使用する追加のストリングが保持されます。 CAPTURE_STATUS および APPLY_STATUS 条件は、この列の値を使用します。この列の値は、次の 3 つのパラメーターをコンマで区切って連結したストリングになります。 • キャプチャー・サーバーまたはアプライ・コントロール・サーバー。 z/OS これは DB2 サブシステム名です。 • リモート DB2 インスタンス名 (サーバーがリモートの場合のみ)。 • リモート・ホスト名。 値が NULL またはゼロ長ストリングの場合、モニター・プログラムは次のデフォルトを使用します。 • キャプチャーまたはアプライ・コントロール・サーバーの CURRENT SERVER 値。 • リモートDB2 インスタンス名の値: – Linux UNIX この値は、 UNIX サーバーの接続時に使用されたユーザー ID の名前です。 – Windows この値は DB です。 • DB2 ノード・ディレクトリーのホスト名の値。
CONTACT_TYPE	**データ・タイプ**: CHAR(1)。**NULL 可能**: 不可。 この条件が発生したときに個人またはグループに連絡するかどうかを示すフラグ。 **C** 個人連絡先 **G** 連絡先のグループ
CONTACT	**データ・タイプ**: VARCHAR(127)。**NULL 可能**: 不可。 この条件が発生したときに通知する個人連絡先、または連絡先のグループ。

IBMSNAP_CONTACTGRP 表

IBMSNAP_CONTACTGRP 表には、連絡先グループを構成する個人の連絡先が入っています。アラート条件が発生したときに、これらの個人のグループにレプリケーション・アラート・モニターから連絡がいくように指定できます。1 個人は、複数の連絡先グループに所属できます (列はユニークではありません)。

サーバー: モニター・コントロール・サーバー

非ユニーク索引: GROUP_NAME、CONTACT_NAME

504 ページの表 98では、IBMSNAP_CONTACTGRP 表の列の要旨を示します。

表 98. IBMSNAP_CONTACTGRP 表の列

列名	説明
GROUP_NAME	**データ・タイプ**: VARCHAR(127)。**NULL 可能**: 不可。 連絡先グループの名前。
CONTACT_NAME	**データ・タイプ**: VARCHAR(127)。**NULL 可能**: 不可。 グループの一部である連絡先の名前。これらの個人は、モニター連絡先 (IBMSNAP_CONTACTS) 表で指定されます。

IBMSNAP_CONTACTS 表

IBMSNAP_CONTACTS 表は、個人 (またはそのグループ) に関連付けられたアラート条件が発生したときに、レプリケーション・アラート・モニターから個人に通知を行うために必要となる情報が入っています。 1 行に 1 人指定します。

サーバー: モニター・コントロール・サーバー

非ユニーク索引: CONTACT_NAME

表 99では、IBMSNAP_CONTACTS 表の列の要旨を示します。

表 99. IBMSNAP_CONTACTS 表の列

列名	説明
CONTACT_NAME	**データ・タイプ**: VARCHAR(127)。**NULL 可能**: 不可。 連絡先の名前。個人連絡先のみが許可されています。グループ名はサポートされていません。
EMAIL_ADDRESS	**データ・タイプ**: VARCHAR(128)。**NULL 可能**: 不可。 この連絡先の主な E メールまたはページャーのアドレス。
ADDRESS_TYPE	**データ・タイプ**: CHAR(1)。**NULL 可能**: 可。 この連絡先の E メール・アドレスが、E メール・アカウントであるか、ページャー・アドレスであるかを示すフラグ。 **E** E メール・アドレスは E メール・アカウント用です。 **P** E メール・アドレスはページャー用です。
DELEGATE	**データ・タイプ**: VARCHAR(127)。**NULL 可能**: 可。 代行期間中に通知を受ける連絡先の名前。個人連絡先名のみが許可されています。グループ名はサポートされていません。
DELEGATE_START	**データ・タイプ**: DATE。**NULL 可能**: 可。 DELEGATE 列で指定された個人に通知を送信する場合の、代行期間の開始日。
DELEGATE_END	**データ・タイプ**: DATE。**NULL 可能**: 可。 代行期間の終了日。
DESCRIPTION	**データ・タイプ**: VARCHAR(1024)。**NULL 可能**: 可。 連絡先の説明。

IBMSNAP_GROUPS 表

IBMSNAP_GROUPS 表には、各連絡先グループの名前と記述が入っています。 1 つの行に 1 つのグループが指定されます。

サーバー: モニター・コントロール・サーバー

非ユニーク索引: GROUP_NAME

表 100では、IBMSNAP_GROUPS 表の列の要旨を示します。

表 100. IBMSNAP_GROUPS 表の列

列名	説明
GROUP_NAME	**データ・タイプ**: VARCHAR(127)。**NULL 可能**: 可。 連絡先グループの名前。
DESCRIPTION	**データ・タイプ**: VARCHAR(1024)。**NULL 可能**: 可。 連絡先グループの記述。

IBMSNAP_MONENQ 表

IBMSNAP_MONENQ 表は、レプリケーションで将来使用するために予約済みです。

サーバー: モニター・コントロール・サーバー

非ユニーク索引: MONITOR_QUAL

表 101では、IBMSNAP_MONENQ 表の列の要旨を示します。

表 101. IBMSNAP_MONENQ 表の列

列名	説明
MONITOR_QUAL	**データ・タイプ**: CHAR(18)。**NULL 可能**: 不可。 レプリケーションで将来使用するために予約済み。

IBMSNAP_MONPARMS 表

IBMSNAP_MONPARMS 表は、レプリケーション・アラート・モニターの操作をコントロールするためにユーザーが変更できるパラメーターを保持します。

これらのパラメーターを定義して、アラート条件が満たされるときにモニター・プログラムが送信する通知メッセージの数などの値を設定できます。ユーザーがこの表のパラメーターを変更しても、モニター・プログラムは始動時にしか変更を読み取りません。

サーバー: モニター・コントロール・サーバー

索引: MONITOR_QUAL

デフォルト・スキーマ: ASN

この表の情報は、SQL を使って更新できます。

表 102 では、IBMSNAP_MONPARMS 表の列の要旨を示します。

表 102. IBMSNAP_MONPARMS 表の列

列名	説明
MONITOR_QUAL	**データ・タイプ**: CHAR(18)。**NULL 可能**: 不可。 モニター修飾子は、これらのパラメーターの適用対象のレプリケーション・アラート・モニター・プログラムにパラメーターを一致させます。
ALERT_PRUNE_LIMIT	**データ・タイプ**: INT。**NULL 可能**: 不可。デフォルトあり。**デフォルト:** 10080 分 (7 日)。 表から整理されるまでのデータの経過時間を示すフラグ。
AUTOPRUNE	**データ・タイプ**: CHAR(1)。**NULL 可能**: 不可。デフォルトあり。**デフォルト:** Y。 モニター・プログラムが、不必要になった行を IBMSNAP_ALERTS、IBMSNAP_MONTRACE、および IBMSNAP_MONTRAIL の各コントロール表から自動的に整理するかどうかを示すフラグ。 **Y** 自動整理はオン。 **N** 自動整理はオフ。
EMAIL_SERVER	**データ・タイプ**: INT(128)。**NULL 可能**: 可。 SMTP プロトコルを使用する E メール・サーバーのアドレス。
LOGREUSE	**データ・タイプ**: CHAR(1)。**NULL 可能**: 不可。デフォルトあり。**デフォルト:** N。 モニター・プログラムがモニター・ログ・ファイルに上書きするか、ファイルに追加するかを示すフラグ。 **Y** モニター・プログラムは、最初にログ・ファイルを削除し、モニター・プログラムの再始動時にそれを再作成することにより、ログ・ファイルを再利用します。 **N** モニター・プログラムは新しい情報をモニター・ログ・ファイルに追加します。
LOGSTDOUT	**データ・タイプ**: CHAR(1)。**NULL 可能**: 不可。デフォルトあり。**デフォルト:** N。 モニター・プログラムがログ・ファイル・メッセージを送信するかどうかを示すフラグ。 **Y** モニター・プログラムは、標準出力 (STDOUT) とログ・ファイルの両方にログ・ファイル・メッセージを送信します。 **N** モニター・プログラムは、ほとんどのログ・ファイル・メッセージをログ・ファイルにのみ送ります。初期化メッセージは、標準出力 (STDOUT) とログ・ファイルの両方に送られます。
NOTIF_PER_ALERT	**データ・タイプ**: INT。**NULL 可能**: 不可。デフォルトあり。**デフォルト:** 3。 アラート条件が満たされた場合に送信される通知メッセージの数。

表 102. IBMSNAP_MONPARMS 表の列 (続き)

列名	説明
NOTIF_MINUTES	**データ・タイプ**: INT。**NULL 可能**: 不可。デフォルトあり。**デフォルト:** 60。 アラート条件が満たされた場合に通知メッセージを受け取る分数。
MONITOR_ERRORS	**データ・タイプ**: VARCHAR(128)。**NULL 可能**: 可。 レプリケーション・アラート・モニターの操作に関連するエラーが発生したときに通知メッセージを送信する E メール・アドレスを指定します。
MONITOR_INTERVAL	**データ・タイプ**: INT。**NULL 可能**: 不可。デフォルトあり。**デフォルト:** 300000 (5 分)。 レプリケーション・アラート・モニターが、選択されたアラート条件をモニターするために実行する頻度 (ミリ秒単位)。
MONITOR_PATH	**データ・タイプ**: VARCHAR(1040)。**NULL 可能**: 可。 モニター・プログラムからの出力が送信されるパス。
RUNONCE	**データ・タイプ**: CHAR(1)。**NULL 可能**: 不可。デフォルトあり。**デフォルト:** N。 モニター・プログラムが、選択されたアラート条件をチェックするかどうかを示すフラグ。 **Y** モニター・プログラムはアラート条件をチェックします。 **N** モニター・プログラムはアラート条件をチェックしません。 RUNONCE が Y に設定されると、MONITOR_INTERVAL は無視されます。
TERM	**データ・タイプ**: CHAR(1)。**NULL 可能**: 不可。デフォルトあり。**デフォルト:** N。 DB2 が静止モードに置かれるときにモニター・プログラムが終了するかどうかを示すフラグ。 **N** DB2 静止時に、モニター・プログラムはアクティブのまま、DB2 が静止解除されるのを待ちます。 **Y** モニター・プログラムは、DB2 静止時に終了します。 TERM の値に関係なく、モニター・プログラムは、DB2 のシャットダウン時に停止します。 DB2 が再び始動するときに、モニター・プログラムを再始動する必要があります。
TRACE_LIMIT	**データ・タイプ**: INT。**NULL 可能**: 不可。デフォルトあり。**デフォルト:** 10080。 整理の対象となるまでに、行が IBMSNAP_MONTRACE 表の中に留まる分数。整理プロセス時に、分数 (現在のタイム・スタンプから IBMSNAP_MONTRACE 表に行が挿入された時刻を引いたもの) が TRACE_LIMIT の値を超えると、モニター・トレース表の中の行を削除します。

表 102. IBMSNAP_MONPARMS 表の列 (続き)

列名	説明
ARCH_LEVEL	**データ・タイプ**: CHAR(8)。**NULL 可能**: 不可。デフォルトあり。**デフォルト:** 0901。 行に含まれる定義の構造レベル。この列は、行を作成する基礎となった規則を示します。このレベルは IBM によって定義されており、バージョン 9.1 のレベルは 0901 です。 **重要:** IBMSNAP_MONPARMS 表の更新中には、この列の値を変更しないでください。

IBMSNAP_MONSERVERS 表

IBMSNAP_MONSERVERS 表は、レプリケーション・アラート・モニターが、キャプチャー・コントロール・サーバー、アプライ・コントロール・サーバー、Q キャプチャー・サーバー、または Q アプライ・サーバーを最後にモニターしたときの情報を保持します。

サーバー: モニター・コントロール・サーバー

非ユニーク索引: MONITOR_QUAL、SERVER_NAME

表 103では、IBMSNAP_MONSERVERS 表の列の要旨を示します。

表 103. IBMSNAP_MONSERVERS 表の列

列名	説明
MONITOR_QUAL	**データ・タイプ**: CHAR(18)。**NULL 可能**: 不可。 キャプチャー・コントロール・サーバー、アプライ・コントロール・サーバー、Q キャプチャー・サーバー、または Q アプライ・サーバーをモニターしているレプリケーション・アラート・モニターを識別するモニター修飾子。
SERVER_NAME	**データ・タイプ**: CHAR(18)。**NULL 可能**: 不可。 レプリケーション・アラート・モニターによるモニター対象のキャプチャー・コントロール・サーバー、アプライ・コントロール・サーバー、Q キャプチャー・サーバー、または Q アプライ・サーバーの名前。
SERVER_ALIAS	**データ・タイプ**: CHAR(8)。**NULL 可能**: 可。 レプリケーション・アラート・モニターによるモニター対象のキャプチャー・コントロール・サーバー、アプライ・コントロール・サーバー、Q キャプチャー・サーバー、または Q アプライ・サーバーの DB2 別名。
LAST_MONITOR_TIME	**データ・タイプ**: TIMESTAMP。**NULL 可能**: 可。 レプリケーション・アラート・モニターが、キャプチャー・コントロール・サーバー、アプライ・コントロール・サーバー、Q キャプチャー・サーバー、または Q アプライ・サーバーに最後に接続したときの時刻 (このサーバーの)。この値は、コントロール表からメッセージをフェッチする下限値として使用され、最後に成功したモニター・サイクルの START_MONITOR_TIME の値と同じ値になります。

表 103. IBMSNAP_MONSERVERS 表の列 (続き)

列名	説明
START_MONITOR_TIME	**データ・タイプ**: TIMESTAMP。**NULL 可能**: 可。 レプリケーション・アラート・モニターが、キャプチャー・コントロール・サーバー、アプライ・コントロール・サーバー、Q キャプチャー・サーバー、または Q アプライ・サーバーに接続したときの時刻 (キャプチャー・コントロール・サーバー、アプライ・コントロール・サーバー、Q キャプチャー・サーバー、または Q アプライ・サーバーの)。この値は、コントロール表からアラート・メッセージをフェッチする上限値として使用されます。
END_MONITOR_TIME	**データ・タイプ**: TIMESTAMP。**NULL 可能**: 可。 レプリケーション・アラート・モニターが、キャプチャー・コントロール・サーバー、アプライ・コントロール・サーバー、Q キャプチャー・サーバー、または Q アプライ・サーバーのモニターを終了したときの時刻 (このサーバーの)。
LASTRUN	**データ・タイプ**: TIMESTAMP。**NULL 可能**: 不可。 レプリケーション・アラート・モニターが、キャプチャー・コントロール・サーバー、アプライ・コントロール・サーバー、Q キャプチャー・サーバー、または Q アプライ・サーバーの処理を最後に開始したときの時刻 (モニター・コントロール・サーバーの)。
LASTSUCCESS	**データ・タイプ**: TIMESTAMP。**NULL 可能**: 可。 レプリケーション・アラート・モニターが、キャプチャー・コントロール・サーバー、アプライ・コントロール・サーバー、Q キャプチャー・サーバー、または Q アプライ・サーバーの処理を最後に正常に完了した時刻の (モニター・コントロール・サーバーの) LASTRUN 列の値。このサーバーのモニターが繰り返し失敗するときには、この値が同じである可能性があります (この列の履歴は IBMSNAP_MONTRAIL 表にあります)。
STATUS	**データ・タイプ**: SMALLINT。**NULL 可能**: 不可。 モニター・サイクルの状況を示すフラグ。 **-1** レプリケーション・アラート・モニターは、このサーバーを正常に処理できませんでした。 **0** レプリケーション・アラート・モニターは、このサーバーを正常に処理しました。 **1** レプリケーション・アラート・モニターは、このサーバーを現在処理中です。

IBMSNAP_MONTRACE 表

IBMSNAP_MONTRACE 表には、レプリケーション・アラート・モニターの監査証跡情報が含まれます。モニター・プログラムによる処理はすべてこの表に記録されるため、モニター・プログラムの問題が発生した場合は、この表を参照すると便利です。

サーバー: モニター・コントロール・サーバー

非ユニーク索引: MONITOR_QUAL、TRACE_TIME

表 104では、IBMSNAP_MONTRACE 表の列の要旨を示します。

表 104. IBMSNAP_MONTRACE 表の列

列名	説明
MONITOR_QUAL	**データ・タイプ**: CHAR(18)。**NULL 可能**: 不可。 メッセージを発行したレプリケーション・アラート・モニターを示すモニター修飾子。
TRACE_TIME	**データ・タイプ**: TIMESTAMP。**NULL 可能**: 不可。 メッセージがこの表に挿入されたときのタイム・スタンプ。
OPERATION	**データ・タイプ**: CHAR(8)。**NULL 可能**: 不可。 メッセージを分類するために使用される値。 **ERROR** エラー・メッセージ **WARNING** 警告メッセージ **INFO** 情報メッセージ
DESCRIPTION	**データ・タイプ**: VARCHAR(1024)。**NULL 可能**: 不可。 メッセージ・コードおよびテキスト。

IBMSNAP_MONTRAIL 表

IBMSNAP_MONTRAIL 表には、各モニター・サイクルに関する情報が入ります。レプリケーション・アラート・モニターは、モニター対象のキャプチャー・コントロール・サーバー、アプライ・コントロール・サーバー、 Q キャプチャー・サーバー、および Q アプライ・サーバーごとに行を 1 つ挿入します。

サーバー: モニター・コントロール・サーバー

非ユニーク索引: なし

表 105では、IBMSNAP_MONTRAIL 表の列の要旨を示します。

表 105. IBMSNAP_MONTRAIL 表の列

列名	説明
MONITOR_QUAL	**データ・タイプ:** CHAR(18)。**NULL 可能**: 不可。 キャプチャー・コントロール・サーバー、アプライ・コントロール・サーバー、Q キャプチャー・サーバー、または Q アプライ・サーバーをモニターしているレプリケーション・アラート・モニターを識別するモニター修飾子。
SERVER_NAME	**データ・タイプ:** CHAR(18)。**NULL 可能**: 不可。 レプリケーション・アラート・モニターによるモニター対象のキャプチャー・コントロール・サーバー、アプライ・コントロール・サーバー、Q キャプチャー・サーバー、または Q アプライ・サーバーの名前。

表 105. IBMSNAP_MONTRAIL 表の列 (続き)

列名	説明
SERVER_ALIAS	**データ・タイプ:** CHAR(8)。**NULL 可能**: 可。 レプリケーション・アラート・モニターによるモニター対象のキャプチャー・コントロール・サーバー、アプライ・コントロール・サーバー、Q キャプチャー・サーバー、または Q アプライ・サーバーの DB2 別名。
STATUS	**データ・タイプ:** SMALLINT。**NULL 可能**: 不可。 モニター・サイクルの状況を示すフラグ。 **-1** レプリケーション・アラート・モニターは、このサーバーを正常に処理できませんでした。 **0** レプリケーション・アラート・モニターは、このサーバーを正常に処理しました。 **1** レプリケーション・アラート・モニターは、このサーバーを現在処理中です。
LASTRUN	**データ・タイプ:** TIMESTAMP。**NULL 可能**: 不可。 レプリケーション・アラート・モニター・プログラムが、キャプチャー・コントロール・サーバー、アプライ・コントロール・サーバー、Q キャプチャー・サーバー、または Q アプライ・サーバーの処理を最後に開始したときの時刻 (モニター・コントロール・サーバーの)。
LASTSUCCESS	**データ・タイプ:** TIMESTAMP。**NULL 可能**: 可。 レプリケーション・アラート・モニターが、キャプチャー・コントロール・サーバー、アプライ・コントロール・サーバー、Q キャプチャー・サーバー、または Q アプライ・サーバーの処理を最後に正常に完了したときの時刻 (モニター・コントロール・サーバーの)。
ENDTIME	**データ・タイプ:** TIMESTAMP。**NULL 可能:** デフォルトでは不可 この表にこの行が挿入された時刻。デフォルト: 現在のタイム・スタンプ
LAST_MONITOR_TIME	**データ・タイプ:** TIMESTAMP。**NULL 可能**: 可。 レプリケーション・アラート・モニターが、キャプチャー・コントロール・サーバー、アプライ・コントロール・サーバー、Q キャプチャー・サーバー、または Q アプライ・サーバーに最後に接続したときの時刻 (キャプチャー・コントロール・サーバー、アプライ・コントロール・サーバー、Q キャプチャー・サーバー、または Q アプライ・サーバーの)。この値は、コントロール表からメッセージをフェッチする下限値として使用され、直前の成功したモニター・サイクルの START_MONITOR_TIME の値と同じ値になります。
START_MONITOR_TIME	**データ・タイプ:** TIMESTAMP。**NULL 可能**: 可。 レプリケーション・アラート・モニターが、キャプチャー・コントロール・サーバー、アプライ・コントロール・サーバー、Q キャプチャー・サーバー、または Q アプライ・サーバーのモニターを最後に開始した時刻。
END_MONITOR_TIME	**データ・タイプ:** TIMESTAMP。**NULL 可能**: 可。 レプリケーション・アラート・モニターが、キャプチャー・コントロール・サーバー、アプライ・コントロール・サーバー、Q キャプチャー・サーバー、または Q アプライ・サーバーのモニターを最後に終了した時刻。

表 105. IBMSNAP_MONTRAIL 表の列 (続き)

列名	説明
SQLCODE	**データ・タイプ:** INT。**NULL 可能**: 可。 このモニター・サイクル中に発生したエラーの SQLCODE。
SQLSTATE	**データ・タイプ:** CHAR(5)。**NULL 可能**: 可。 このモニター・サイクル中に発生したエラーの SQLSTATE。
NUM_ALERTS	**データ・タイプ:** INT。**NULL 可能**: 不可。 このモニター・サイクル中に発生したアラート条件の数。
NUM_NOTIFICATIONS	**データ・タイプ:** INT。**NULL 可能**: 不可。 このモニター・サイクル中に送信された通知の数。
SUSPENSION_NAME	**データ・タイプ:** VARCHAR(128)。**NULL 可能**: 可。 定義済みの期間、モニターの操作を停止するために使用する中断の名前。

IBMSNAP_SUSPENDS 表

IBMSNAP_SUSPENDS 表は、モニター・プログラムの一時中断に関する情報を格納します。

サーバー: モニター・コントロール・サーバー

デフォルト・スキーマ: ASN

主キー: SUSPENSION_NAME

ユニーク索引: SERVER_NAME、TEMPLATE_NAME、START

表 106では、IBMSNAP_SUSPENDS 表の列の要旨を示します。

表 106. IBMSNAP_SUSPENDS 表の列

列名	説明
SUSPENSION_NAME	**データ・タイプ:** VARCHAR(128)。**NULL 可能:** 不可。 モニター中断の名前。
SERVER_NAME	**データ・タイプ:** CHAR(18)。**NULL 可能:** 不可 モニターを中断する Q キャプチャー・サーバー、Q アプライ・サーバー、キャプチャー・コントロール・サーバー、アプライ・コントロール・サーバーの名前。
SERVER_ALIAS	**データ・タイプ:** CHAR(18)。**NULL 可能:** 可 モニターを中断するサーバーの別名。
TEMPLATE_NAME	**データ・タイプ:** VARCHAR(128)。**NULL 可能:** 可。 モニター中断テンプレートの名前。IBMSNAP_TEMPLATES コントロール表にこの列の値が存在しない場合、モニターは、START タイム・スタンプの時刻から STOP タイム・スタンプの時刻まで 1 回中断します。

表 106. IBMSNAP_SUSPENDS 表の列 (続き)

列名	説明
START	**データ・タイプ:** TIMESTAMP。**NULL 可能:** 不可 テンプレート使用の開始時刻。テンプレートを指定しない場合は、中断の開始時刻になります。
STOP	**データ・タイプ:** TIMESTAMP。**NULL 可能:** 不可 テンプレート使用の終了時刻。テンプレートを指定しない場合は、中断の終了時刻になります。

IBMSNAP_TEMPLATES 表

IBMSNAP_TEMPLATES 表は、モニター・プログラムの中断の頻度と長さに関する情報を格納します。この情報のことをモニター中断テンプレートといいます。

サーバー: モニター・コントロール・サーバー

デフォルト・スキーマ: ASN

ユニーク索引: TEMPLATE_NAME

表 107では、IBMSNAP_TEMPLATES 表の列の要旨を示します。

表 107. IBMSNAP_TEMPLATES 表の列

列名	説明
TEMPLATE_NAME	**データ・タイプ:** VARCHAR(128)。**NULL 可能:** 不可。 モニター中断テンプレートの名前。
START_TIME	**データ・タイプ:** TIME。**NULL 可能:** 不可。 中断を開始する日の時刻。デフォルト: 00:00:00
WDAY	**データ・タイプ:** SMALLINT。**NULL 可能:** 可。 中断を開始する曜日 (日曜日の 0 から始まり、土曜日の 6 まで続きます)。NULL 値の場合は、どの曜日にも中断を開始できるという意味になります。
DURATION	**データ・タイプ:** INTEGER。**NULL 可能:** 不可。 中断の継続時間 (分単位)。

ターゲット・サーバーの表

ターゲット・サーバーには、さまざまなタイプのターゲット表が保管されます。ユーザーがターゲット表として既存の表を使用しない場合、ASNCLP コマンド行プログラムまたはレプリケーション・センターは、ユーザーがサブスクリプション・セット・メンバーをどのように定義しているかに従って、ユーザーの指定に合わせてターゲット表を構築します。

514 ページの表 108 は、ターゲット・サーバーの表を説明しています。

表 108. ターゲット表のクイック・リファレンス

表名	説明
『基礎集約表』	ソース表から集約されたデータを保持します。
『変更集約表』	CD 表から集約されたデータを保持します。
82 ページの『CCD ターゲット』	ソースに発生する変更に関する情報と、これらの変更の順序を識別するための追加の列を保持します。
516 ページの『ポイント・イン・タイム表』	ソース・ログの中でデータがコミットされた特定の時刻を記録する追加の列を持つ、ソース・データのコピー。
516 ページの『レプリカ表』	Update-anywhere レプリケーションで使用されるターゲット表のタイプ。
517 ページの『ユーザー・コピー表』	ソース表のコピー。

基礎集約表

基礎集約表は、ソース表上にあるデータに対して実行された集約関数の結果を含むターゲット表です。

schema.base_aggregate

サーバー: ターゲット・サーバー

重要: SQL を使用してこの表を更新した場合は、アプライ・プログラムによってフル・リフレッシュが実行されたときにユーザーの更新情報が失われる危険性があります。

表 109 では、基礎集約表の列の要旨を示します。

表 109. 基礎集約表の列

列名	説明
user columns	ソース表から算出された集約データ。
IBMSNAP_LLOGMARKER	ソース表のデータの集約が開始されたときのソース・サーバーの現行タイム・スタンプ。
IBMSNAP_HLOGMARKER	ソース表のデータの集約が完了したときのソース・サーバーの現行タイム・スタンプ。

変更集約表

変更集約表は、変更データ (CD) 表にあるデータに対して実行された集約関数の結果を含むターゲット表です。この表は基礎集約表と似ていますが、 CD 表で実行される関数は、特定の時間間隔で発生する変更に対してのみ使用される点が異なります。

schema.change_aggregate

サーバー: ターゲット・サーバー

重要: SQL を使用してこの表を更新した場合は、アプライ・プログラムによってフル・リフレッシュが実行されたときにユーザーの更新情報が失われる危険性があります。

表 110 では、変更集約表の列の要旨を示します。

表 110. 変更集約表の列

列名	説明
user key columns	ターゲット・キーを構成する列。
user nonkey columns	ソース表からの非キー・データ列。このターゲット表の中の列名はソース表の中の列名と一致している必要はありませんが、データ・タイプは一致している必要があります。
user computed columns	SQL 式から派生したユーザー定義の列。ソース・データ・タイプを別のターゲット・データ・タイプに変換するために、SQL 関数で算出列を使用することができます。
IBMSNAP_LLOGMARKER	集約されている (CD+UOW) または CCD 表の行内の最も古い IBMSNAP_LOGMARKER または IBMSNAP_LLOGMARKER 値。
IBMSNAP_HLOGMARKER	集約されている (CD+UOW) または CCD 表の行内の最も新しい IBMSNAP_LOGMARKER または IBMSNAP_HLOGMARKER 値。

CCD ターゲット

ソース・データの監査を行ったり、データの使用状況の履歴を保持したりすることが必要な場合もあります。ターゲット・タイプとして整合変更データ (CCD) 表を使用することで、ソースの変更に関する履歴を追跡管理できます。

例えば、データの変更が発生したときの変更前後の比較や、ソース表への更新を行ったユーザー ID をトラッキングできます。

ソース表の履歴を保持する読み取り専用ターゲット表を定義するには、以下の属性を持つようにターゲット CCD 表を定義します。

非コンデンス

ソースの変更すべてに関するレコードを保持するには、CCD 表を非コンデンスに定義し、発生した変更ごとに 1 行が保管されるようにします。非コンデンスの表には同じキー値を持つ複数の行が含まれるため、ユニーク索引は定義しないでください。非コンデンス CCD 表は、 UPDATE、INSERT、または DELETE 操作ごとに 1 行を保有することで、ソース表に対して実行された操作の履歴を保留します。 UPDATE 操作を INSERT および DELETE 操作 (パーティション化キー列用) としてキャプチャーする場合、CCD 表は、それぞれの更新ごとに 2 つの行、すなわち DELETE に 1 行、INSERT に 1 行を指定します。

コンプリートまたは非コンプリート

CCD 表をコンプリートにするか、あるいは非コンプリートにするかを選択できます。未完成の CCD 表にはソース行のコンプリート・セットが最初含まれていないため、ソース表への更新 (アプライ・プログラムが CCD 表の移植を開始してからの更新) 履歴を保持する未完成の CCD 表を作成します。

UOW (作業単位) 列の組み込み

監査機能の改善のため、UOW 表からの追加の列を組み込んでください。ユーザー指向の識別がさらに必要であれば、UOW 表で、DB2 for z/OS の相関 ID に関する列、 1 次許可 ID、または System i のジョブ名およびユーザー・プロファイルを使用することができます。

ポイント・イン・タイム表

ポイント・イン・タイム表は、ソース・データのコピーと、特定の行がソース・サーバーで挿入または更新されたおおよそのポイント・イン・タイムのタイム・スタンプを含む追加のシステム列 (IBMSNAP_LOGMARKER) を保持します。

schema.point_in_time

サーバー: ターゲット・サーバー

重要: SQL を使用してこの表を更新した場合は、アプライ・プログラムによってフル・リフレッシュが実行されたときにユーザーの更新情報が失われる危険性があります。

表 111 では、ポイント・イン・タイム表の列の要旨を示します。

表 111. ポイント・イン・タイム表の列

列名	説明
user key columns	ターゲット・キーを構成する列。
user nonkey columns	ソース表またはビューからの非キー・データ列。 このターゲット表の中の列名はソース表の中の列名と一致している必要はありませんが、データ・タイプは一致している必要があります。
user computed columns	SQL 式から派生したユーザー定義の列。ソース・データ・タイプを別のターゲット・データ・タイプに変換するために、SQL 関数で算出列を使用することができます。
IBMSNAP_LOGMARKER	キャプチャー・コントロール・サーバーにおけるおおよそのコミット時刻。この列は、フル・リフレッシュ後は NULL になります。

レプリカ表

レプリカ表には、ソース表と同じ主キー列が必要です。この類似性のために、レプリカ表は、その他のサブスクリプション・セットでソース表として使用できます。ターゲット表からソース表への変換は、ユーザーがレプリカ・ターゲット・タイプを定義し、 CHANGE DATA CAPTURE 属性を指定すると、自動的に行われます。

schema.replica

サーバー: ターゲット・サーバー

この表の情報は、SQL を使って更新できます。

517 ページの表 112 では、レプリカ表の列の要旨を示します。

表 112. レプリカ表の列

列名	説明
user key columns	ターゲット・キーを構成する列。マスター表と同じ主キーであることが必要です。
user nonkey columns	ソース表からの非キー・データ列。このターゲット表の中の列名はソース表の中の列名と一致している必要はありませんが、データ・タイプは一致している必要があります。

ユーザー・コピー表

ユーザー・コピー表は、ソース表の列のコピーを含むターゲット表です。このターゲット表では、ソース表の行または列のサブセットを使用できますが、追加列を含めることはできません。

schema.user_copy

サーバー: ターゲット・サーバー

重要: SQL を使用してこの表を更新した場合は、アプライ・プログラムによってフル・リフレッシュが実行されたときにユーザーの更新情報が失われる危険性があります。

サブセット化とデータ拡張の場合を除き、ユーザー・コピー表はソース表の正しい状態を反映しますが、それが最新の状態であるとはかぎりません。ユーザー・コピー表 (または他の任意のターゲット表タイプ) への参照により、ソース表への直接アクセスが多すぎるときに生じる競合問題を減らすことができます。ローカル・ユーザー・コピー表にアクセスすると、照会ごとにネットワークを使用してリモート・ソース表へアクセスするよりも迅速です。

表 113 では、ユーザー・コピー表の列の要旨を示します。

表 113. ユーザー・コピー表の列

列名	説明
user key columns	ターゲット・キーを構成する列。
user nonkey columns	ソース表またはビューからの非キー・データ列。 このターゲット表の中の列名はソース表の中の列名と一致している必要はありませんが、データ・タイプは一致している必要があります。
user computed columns	SQL 式から派生したユーザー定義の列。ソース・データ・タイプを別のターゲット・データ・タイプに変換するために、SQL 関数で算出列を使用することができます。

付録 A. SQL レプリケーション用の Unicode および ASCII のコード化スキーム (z/OS)

SQL replication for OS/390 and z/OS バージョン 7 以降では、Unicode と ASCII のコード化スキームをサポートします。

Unicode のコード化スキームを活用するには、少なくとも DB2 for OS/390 and z/OS バージョン 7 が必要で、以下のセクションで説明するように、SQL レプリケーションのソース、ターゲット、およびコントロールの各表を手動で作成または変換しなければなりません。ただし、既存のレプリケーション環境は、コード化スキームが変更されていなければ、SQL replication for OS/390 and z/OS バージョン 7 以降で機能します。 使用しているシステムが Unicode システムの場合は、キャプチャー、アプライ、およびレプリケーション・アラート・モニターの各プログラムの BIND PLAN コマンドと PACKAGE コマンドに、 ENCODING(EBCDIC) を付け足さなければなりません。

コード化スキームの選択の規則

ソース表、CD 表、およびターゲット表で同じコード化スキームを使用していると、レプリケーション環境でのデータ変換の必要性を最小にすることができます。

これらの表のコード化スキームを選択するときは、1 つの CCSID 規則に従ってください。

> 表スペース・データは ASCII、EBCDIC、または Unicode の CCSID を使ってエンコードされます。 ある SQL ステートメントで参照されるすべての表のコード化スキームは同じでなければなりません。また、表示または結合して使用する表もすべて、同じコード化スキームを使用しなければなりません。

単一の CCSID 規則に従わないと、DB2 はバインドまたは実行中に違反を検出して SQLCODE -873 を戻します。

どの表を ASCII または Unicode にする必要があるかは、クライアント/サーバー構成によります。具体的には、表のコード化スキームを選択するとき、以下の規則に従ってください。

- DB2 for OS/390 のソースまたはターゲット表は EBCDIC、ASCII、または Unicode にする。これらの表は、サポートされているすべての DBMS (DB2 ファミリー、または DataJoiner を使用した非 DB2) の表 (同一または別のコード化スキームを持つ) と相互にコピーできます。
- サブスクリプション・セット・メンバーの作成時に、ターゲット・タイプが USERCOPY であり JOIN_UOW_CD が Y ではない場合、DB2 for OS/390 ソース・サーバーでは、同一サーバー上の CD 表と UOW 表が同じコード化スキームを使用する必要はない。それ以外の場合、CD 表と UOW 表は同じコード化スキームを使用しなければなりません。

- IBMSNAP_SIGNAL 表は、キャプチャー・プログラムがシグナルをシグナル表から選択するときに EBCDIC に変換する必要がないように、EBCDIC にエンコードしておく必要がある。
- 同じコントロール・サーバー上のすべてのコントロール表 (ASN.IBMSNAP_SUBS_*xxxx*) は、同じコード化スキームを使用しなければならない。
- その他のコントロール表は、任意のコード化スキームを使用できる。

コード化スキームの設定

表に適切なコード化スキームを指定するには、表の生成に使用される SQL を変更します。

既存の表のコード化スキームを変更するときは、その前にキャプチャー・プログラムとアプライ・プログラムを停止することをお勧めします。

注: 「*DB2 for z/OS V8 SQL Reference*」に、CCSID についての詳細が記載されています。

コード化スキームを設定するには、次のようにします。

1. 適切なコード化スキームを使用して新しいソース表とターゲット表を作成します。 その後コールド・スタートでキャプチャーを初期化してアプライ・プログラムを再始動することをお勧めします。
2. ソース表とターゲット表をすでに作成してある場合は、既存のターゲット表とソース表のコード化スキームを変更します。 既存の表は、表スペース内で同じコード化スキームを持たなければなりません。
   a. 表スペースの REORG ユーティリティーを使用して、既存の表スペースをアンロードします。
   b. 既存の表スペースをドロップします。
   c. 新しいコード化スキームを指定する表スペースを再作成します。
   d. ロード・ユーティリティーを使用して、古いデータを新しい表スペースにロードします。 LOAD と REORG の各ユーティリティーについての詳細は、「*DB2 for z/OS V8 Utility Guide and Reference*」を参照してください。
3. レプリケーション・センターを使用して、適切なコード化スキームを持つ新しいコントロール表を作成します。
4. REORG ユーティリティーとロード・ユーティリティーを使用して、既存のコントロール表と CD 表のコード化スキームを変更します。
5. ASNCLP またはレプリケーション・センターを使用して新しいレプリケーション・ソースまたはサブスクリプション・セットを作成するときに、適切なコード化スキームを指定します。

付録 B. アプリケーション (Linux、UNIX、Windows) 内部からの SQL レプリケーション・プログラムの始動

レプリケーションの 1 サイクルにおけるレプリケーション・プログラム (キャプチャー・プログラム、アプライ・プログラム、およびレプリケーション・アラート・モニター) はどれでも、アプリケーションの内部から呼び出しルーチンで始動できます。

API が同期実行のみをサポートしているため、これらのルーチンを使用するには、キャプチャー・プログラムの場合は AUTOSTOP オプション、アプライ・プログラムの場合は COPYONCE オプションを指定しなければなりません。

API のサンプルと各 MAKE ファイルは以下のディレクトリーにあります。

Windows

sqllib¥samples¥repl

Linux UNIX

sqllib/samples/repl

上記のディレクトリーには、キャプチャー・プログラムを始動するための以下のファイルが含まれています。

capture_api.c
Windows、Linux、または UNIX でキャプチャー・プログラムを始動するためのサンプル・コード。

capture_api_nt.mak
Windows でのサンプル・コードの MAKE ファイル。

capture_api_unix.mak
UNIX でのサンプル・コードの MAKE ファイル。

上記のディレクトリーには、アプライ・プログラムを始動するための以下のファイルが含まれています。

apply_api.c
Windows、Linux、または UNIX でアプライ・プログラムを始動するためのサンプル・コード。

apply_api_nt.mak
Windows でのサンプル・コードの MAKE ファイル。

apply_api_unix.mak
UNIX でのサンプル・コードの MAKE ファイル。

上記のディレクトリーには、レプリケーション・アラート・モニターを始動するための以下のファイルが含まれています。

monitor_api.c
Windows、Linux、または UNIX でレプリケーション・アラート・モニターを始動するためのサンプル・コード。

monitor_api_nt.mak
Windows でのサンプル・コードの MAKE ファイル。

monitor_api_unix.mak
UNIX でのサンプル・コードの MAKE ファイル。

付録 C. SQL レプリケーションの場合にキャプチャー・プログラムがジャーナル項目タイプを処理する方法 (System i)

以下の表では、キャプチャー・プログラムが様々なジャーナル項目タイプを処理する方法について説明します。

表 114. ジャーナル項目によるキャプチャー・プログラムの処理

ジャーナル・コード [1]	項目タイプ	説明	処理
C	CM	コミット済みのレコード変更のセット	UOW 表にレコードを挿入します。
C	RB	ロールバック	挿入された UOW 行はありません。
F	AY	物理ファイル・メンバーに適用されたジャーナル済みの変更	ASN2004 メッセージを発行し、ファイルのフル・リフレッシュを行います。
F	CE	物理ファイルのデータの終わりの変更	ASN2004 メッセージを発行し、ファイルのフル・リフレッシュを行います。
F	CR	消去された物理ファイル・メンバー	ASN2004 メッセージを発行し、ファイルのフル・リフレッシュを行います。
F	EJ	終了した物理ファイル・メンバーのジャーナリング	ASN200A メッセージを発行し、ファイルのフル・リフレッシュを行います。フル・リフレッシュは、ユーザーまたはシステムがジャーナリングを終了させたかにかかわらず、キャプチャー・プログラムが EJ ジャーナル項目を読み取るたびに起こります。ファイルの暗黙的なジャーナル終了イベントについての情報は、該当する System i 資料を参照してください。
F	IZ	初期化された物理ファイル・メンバー	ASN2004 メッセージを発行し、ファイルのフル・リフレッシュを行います。
F	MD	物理ファイルから削除されたメンバー (DLTLIB、 DLTF、または RMVM)	ASN200A メッセージを発行し、フル・リフレッシュを試行します。
F	MF	解放された物理ファイル・メンバーのストレージ	ASN200A メッセージを発行し、ファイルのフル・リフレッシュを行います。

表 114. ジャーナル項目によるキャプチャー・プログラムの処理 (続き)

ジャーナル・コード [1]	項目タイプ	説明	処理
F	MM	移動されたメンバーを含む物理ファイル (ライブラリーのオブジェクトの名前変更 (RNMOBJ)、ファイルのオブジェクトの移動 (MOVOBJ))	ASN200A メッセージを発行し、フル・リフレッシュを試行します。
F	MN	名前変更されたメンバー (ファイルの RNMOBJ、メンバーの名前変更 (RNMM)) を含む物理ファイル	ASN200A メッセージを発行し、フル・リフレッシュを試行します。
F	MR	リストアされた物理ファイル・メンバー	ASN2004 メッセージを発行し、ファイルのフル・リフレッシュを行います。
F	RC	物理ファイル・メンバーから削除されたジャーナル済みの変更	ASN2004 メッセージを発行し、ファイルのフル・リフレッシュを行います。
F	RG	再編成された物理ファイル・メンバー	レプリケーション・キーとしてソース表の RRN が使用されている場合には、 ASN2004 メッセージを発行して、ファイルのフル・リフレッシュを行います。
J	NR	次のジャーナル・レシーバーの ID	キャプチャー・プログラムをリセットします。
J	PR	直前のジャーナル・レシーバーの ID	ユニークのシーケンス番号カウンターを増分します。
R	DL	物理ファイル・メンバーから削除されたレコード	DLT レコードを CD 表に挿入します。
R	DR	ロールバックで削除されたレコード	DLT レコードを CD 表に挿入します。
R	PT	物理ファイル・メンバーに追加されたレコード	ADD レコードを CD 表に挿入します。
R	PX	物理ファイル・メンバーに直接追加されたレコード	ADD レコードを CD 表に挿入します。
R	UB	物理ファイル・メンバーで更新されたレコードの変更前イメージ	注 2 を参照してください。
R	UP	物理ファイル・メンバーで更新されたレコードの変更後イメージ	注 2 を参照してください。
R	BR	ロールバックで更新されたレコードの変更前イメージ	注 3 を参照してください。
R	UR	ロールバックで更新されたレコードの変更後イメージ	注 3 を参照してください。

表 114. ジャーナル項目によるキャプチャー・プログラムの処理 (続き)

ジャーナル・コード [1]	項目タイプ	説明	処理

注:

1. 以下の値が、ジャーナル・コードに使用されます。
   - **C** コミットメント・コントロール操作
   - **F** データベース・ファイル操作
   - **J** ジャーナルまたはジャーナル・レシーバー操作
   - **R** 特定のレコードに対する操作
2. 登録表の PARTITION_KEYS_CHG 列が N の場合、R-UP イメージおよび R-UB イメージが単一の UPD レコードを CD 表に形成します。そうでない場合には、 R-UB イメージが DLT レコードを CD 表に挿入し、R-UP イメージが ADD 表を CD 表に挿入します。
3. 登録表の PARTITION_KEYS_CHG 列が N の場合、 R-UR イメージおよび R-BR イメージが単一の UPD レコードを CD 表に形成します。そうでない場合には、 R-BR イメージが DLT レコードを CD 表に挿入し、 R-UR イメージが ADD 表を CD 表に挿入します。

他のジャーナル項目タイプはすべて、キャプチャー・プログラムによって無視されます。

製品に関する情報へのアクセス

IBM では、製品およびサービスをいくつかの方法でご説明しています。

Web 上の最新情報については以下を参照してください。
http://www.ibm.com/software/data/sw-bycategory/subcategory/SWB50.html

製品資料にアクセスするには、publib.boulder.ibm.com/infocenter/db2help/topic/ を参照してください。

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。

利用できる資料

資料は、ほとんどの Web ブラウザーで表示できる XHTML 形式で用意されています。

XHTML により、ご使用のブラウザーに設定されている表示設定に従って資料を表示できます。さらに、スクリーン・リーダーや他の支援テクノロジーを使用することもできます。

シンタックス・ダイアグラムはドット 10 進形式で提供されます。この形式は、スクリーン・リーダーを使用してオンライン資料にアクセスする場合にのみ使用できます。

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。

〒106-8711
東京都港区六本木 3-2-12
IBM World Trade Asia Corporation
Intellectual Property Law & Licensing

以下の保証は、国または地域の法律に沿わない場合は、適用されません。 IBM およびその直接または間接の子会社は、本書を特定物として現存するままの状態で提供し、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含むすべての明示もしくは黙示の保証責任を負わないものとします。国または地域によっては、法律の強行規定により、保証責任の制限が禁じられる場合、強行規定の制限を受けるものとします。

この情報には、技術的に不適切な記述や誤植を含む場合があります。本書は定期的に見直され、必要な変更は本書の次版に組み込まれます。 IBM は予告なしに、随時、この文書に記載されている製品またはプログラムに対して、改良または変更を行うことがあります。

本書において IBM 以外の Web サイトに言及している場合がありますが、便宜のため記載しただけであり、決してそれらの Web サイトを推奨するものではありません。それらの Web サイトにある資料は、この IBM 製品の資料の一部ではありません。それらの Web サイトは、お客様の責任でご使用ください。

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

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 Corporation
J46A/G4
555 Bailey Avenue
San Jose, CA 95141-1003 U.S.A.

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。

本書はプランニング目的としてのみ記述されています。記述内容は製品が使用可能になる前に変更になる場合があります。

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的創作物にも、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.

この情報をソフトコピーでご覧になっている場合は、写真やカラーの図表は表示されない場合があります。

商標

本書では、IBM の商標および IBM 以外の商標の一部につき、それぞれが最初に出現する個所でマークを付けています。

IBM の商標については、www.ibm.com/legal/copytrade.shtml を参照してください。

以下は、他社の商標または登録商標です。

Adobe®、Adobe ロゴ、PostScript®、PostScript ロゴは、Adobe Systems Incorporated 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。

Cell Broadband Engine™ は、Sony Computer Entertainment, Inc. の米国およびその他の国における商標です。

Intel®、Intel (ロゴ)、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。

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。

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。

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。

Linux は、Linus Torvalds の米国およびその他の国における商標です。

ITIL® は英国 Office of Government Commerce の登録商標および共同体登録商標であって、米国特許商標庁にて登録されています。

IT Infrastructure Library® は英国 Office of Government Commerce の一部である the Central Computer and Telecommunications Agency の登録商標です。

他の会社名、製品名およびサービス名等はそれぞれ各社の商標です。

索引

日本語, 数字, 英字, 特殊文字の順に配列されています。なお, 濁音と半濁音は清音と同等に扱われています。

[ア行]

[カ行]

[サ行]

[タ行]

[ナ行]

[ハ行]

[マ行]

[ヤ行]

[ラ行]

[数字]

A

B

C

D

E

F

G

I

J

L

M

N

O

P

Q

R

S

T

U

W

Z

［特殊文字］

Printed in Japan

SC88-4168-01

日本アイ・ビー・エム株式会社
〒106-8711　東京都港区六本木3-2-12

Spine information:

IBM Information Integration バージョン 9.5

SQL レプリケーション ガイドおよびリファレンス