SYBASE®

Adaptive Server 分散トランザクション管理機能の使用

Adaptive Server® Enterprise

15.5

ドキュメント ID：DC01270-01-1550-01
改訂：2009 年 10 月

マニュアルの注文

マニュアルの注文を承ります。ご希望の方は、サイベース株式会社営業部または代理店までご連絡ください。マニュアルの変更は、弊社の定期的なソフトウェア・リリース時にのみ提供されます。

Sybase, Inc., One Sybase Drive, Dublin, CA 94568.

目次

はじめに

対象読者

このマニュアルは、Adaptive Server 分散トランザクション管理 (DTM) 機能を購入した管理者またはアプリケーション開発者を対象としています。

このマニュアルの内容

このマニュアルは、Adaptive Server とそれに関連する機能ライセンスをインストールしたあとに読んでください。

関連マニュアル

Adaptive Server® Enterprise には次のマニュアルが用意されています。必要に応じて参照してください。

- 使用しているプラットフォームの『リリース・ノート』－ マニュアルには記載できなかった最新の情報が記載されています。

  このリリース・ノートの最新バージョン (英語版) を入手できます。製品の CD がリリースされた後で、製品またはマニュアルに関する重要な情報が追加されているかを確認するには、Sybase Product Manuals Web サイトを使用してください。
- 使用しているプラットフォームの『インストール・ガイド』－ すべての Adaptive Server および関連する Sybase 製品のインストール、アップグレード、設定の手順について説明しています。
- 『新機能ガイド』－ Adaptive Server の新しい機能について説明しています。また、新しい機能をサポートするためのシステム変更や、既存のアプリケーションに影響を与える可能性がある変更についても説明しています。
- 『Active Messaging ユーザーズ・ガイド』－ Active Messaging を使用して、Adaptive Server Enterprise データベースでトランザクション (データ変更) を取得し、外部アプリケーションにイベントとしてリアルタイムで渡す方法について説明しています。
- 『コンポーネント統合サービス・ユーザーズ・ガイド』－ コンポーネント統合サービスを使用して、リモートの Sybase データベースおよび Sybase 以外のデータベースに接続する方法について説明しています。
- 使用しているプラットフォームの『設定ガイド』－ 特定の設定作業の手順について説明しています。
- 『用語解説』－ Adaptive Server マニュアルで使用されている技術用語について説明しています。
- 『Historical Server ユーザーズ･ガイド』－ Historical Server を使用して、Adaptive Server のパフォーマンス情報を入手する方法について説明しています。

- 『Adaptive Server Enterprise における Java』－ Adaptive Server データベースで Java クラスをデータ型、関数、ストアド・プロシージャとしてインストールして使用する方法について説明しています。
- 『Job Scheduler ユーザーズ・ガイド』－ コマンド・ラインまたはグラフィカル・ユーザ・インタフェース (GUI) を使用して、ローカルまたはリモートの Adaptive Server でジョブのインストール、設定、作成、スケジュールを行う方法について説明しています。
- 『マイグレーション技術ガイド』－ 別のバージョンの Adaptive Server にマイグレートするための方法とツールについて説明しています。
- 『Monitor Client Library プログラマーズ・ガイド』－ Adaptive Server のパフォーマンス・データにアクセスする Monitor Client Library アプリケーションの記述方法について説明しています。
- 『Monitor Server ユーザーズ･ガイド』－ Monitor Server を使用して､Adaptive Server のパフォーマンス統計を取得する方法について説明しています。
- 『モニタリング・テーブル・ダイヤグラム』－ モニタリング・テーブルと、そのエンティティの関係をポスター形式で図解しています。フル・サイズのダイアグラムは印刷版だけで参照できます。コンパクト版は PDF 形式で参照できます。
- 『パフォーマンス&チューニング・シリーズ』－ Adaptive Server で最高のパフォーマンスを実現するためのチューニング方法について説明しています。
  - 『基本』－ Adaptive Server のパフォーマンスに関する問題の理解と調査の基本について説明しています。
  - 『統計的分析によるパフォーマンスの向上』－ Adaptive Server で統計情報がどのように保存され、表示されるかについて説明しています。また、set statistics コマンドを使用して、サーバの統計情報を分析する方法について説明しています。
  - 『ロックと同時実行制御』－ ロック・スキームを使用してパフォーマンスを向上させる方法と、同時実行性を最小限に抑えるようにインデックスを選択する方法について説明しています。
  - 『sp_sysmon による Adaptive Server の監視』－ sp_sysmon を使用してパフォーマンスをモニタリングする方法について説明しています。
  - 『モニタリング・テーブル』－ Adaptive Server のモニタリング・テーブルに統計情報や診断情報を問い合わせる方法について説明しています。

  - 『物理データベースのチューニング』– データの物理的配置、データに割り付けられた領域、テンポラリ・データベースの管理方法について説明しています。
  - 『クエリ処理と抽象プラン』– オプティマイザがクエリを処理する方法と、抽象プランを使用してオプティマイザのプランの一部を変更する方法について説明しています。
- 『クイック・リファレンス・ガイド』– コマンド、関数、システム・プロシージャ、拡張システム・プロシージャ、データ型、ユーティリティの名前と構文の包括的な一覧表を記載したポケット版 (PDF 版は通常サイズ) のマニュアルです。
- 『リファレンス・マニュアル』– 詳細な Transact-SQL® 情報を記載しています。
  - 『ビルディング・ブロック』– データ型、関数、グローバル変数、式、識別子とワイルドカード、予約語について説明しています。
  - 『コマンド』– コマンドについて説明しています。
  - 『プロシージャ』– システム・プロシージャ、カタログ・ストアド・プロシージャ、システム拡張ストアド・プロシージャ、dbcc ストアド・プロシージャについて説明しています。
  - 『テーブル』– システム・テーブル、モニタリング・テーブル、dbcc テーブルについて説明しています。
- 『システム管理ガイド』でさらに詳しく説明しています。
  - 『第 1 巻』– 設定パラメータ、リソースの問題、文字セット、ソート順、システムの問題の診断方法に関する説明を含め、システム管理の基本の概要について説明しています。『第 1 巻』の後半は、セキュリティ管理に関する詳細な説明です。
  - 『第 2 巻』– 物理的なリソースの管理、デバイスのミラーリング、メモリとデータ・キャッシュの設定、マルチプロセッサ・サーバとユーザ・データベースの管理、データベースのマウントとマウント解除、セグメントの作成と使用、reorg コマンドの使用、データベース一貫性の検査方法についての手順とガイドラインを説明しています。『第 2 巻』の後半では、システムとユーザ・データベースをバックアップおよびリストアする方法について説明しています。
- 『システム・テーブル・ダイアグラム』– システム・テーブルと、そのエンティティとの関係をポスター形式で図解しています。フル・サイズのダイアグラムは印刷版だけで参照できます。コンパクト版は PDF 形式で参照できます。

- 『Transact-SQL ユーザーズ・ガイド』－ リレーショナル・データベース言語の拡張版である Sybase の Transact-SQL について説明しています。まだ経験の浅いデータベース管理システムのユーザは、このマニュアルをガイドブックとして使用してください。pubs2 および pubs3 サンプル・データベースの詳細も説明しています。
- 『トラブルシューティング・シリーズ』－
  - トラブルシューティング『エラー・メッセージと詳細な解決方法』－ 発生する可能性のある問題について、トラブルシューティング手順を説明しています。このマニュアルで取り上げられている問題は、Sybase 製品の保守契約を結んでいるサポート・センタに最も頻繁に寄せられるものです。
  - 『トラブルシューティング＆エラー・メッセージ・ガイド』－ 発生頻度が高い Adaptive Server のエラー・メッセージの解決方法について詳しい手順を説明しています。
- 『暗号化カラム・ユーザーズ・ガイド』－ Adaptive Server を使用して暗号化カラムを設定し、使用する方法について説明しています。
- 『インメモリ・データベース・ユーザーズ・ガイド』－ インメモリ・データベースの設定および使用方法について説明しています。
- 『Adaptive Server 分散トランザクション管理機能の使用』－ 分散トランザクション処理環境での Adaptive Server DTM 機能の設定、使用、トラブルシューティングについて説明しています。
- 『IBM® Tivoli® Storage Manager と Backup Server の使用』－ IBM Tivoli Storage Manager を設定および使用して Adaptive Server のバックアップを作成する方法について説明しています。
- 『高可用性システムにおける Sybase フェールオーバの使用』－ Sybase のフェールオーバ機能を使用して、Adaptive Server を高可用性システムのコンパニオン・サーバとして設定する方法について説明しています。
- 『Unified Agent および Agent Management Console』－ Unified Agent について説明しています。Unified Agent は、分散 Sybase リソースを管理、モニタ、制御するためのランタイム・サービスを提供します。
- 『ユーティリティ・ガイド』－ オペレーティング・システム・レベルで実行される isql および bcp などの、Adaptive Server のユーティリティ・プログラムについて説明しています。
- 『Web Services ユーザーズ・ガイド』－ Adaptive Server 用の Web サービスの設定、使用、トラブルシューティング方法について説明しています。

- 『XA インタフェース統合ガイド for CICS、Encina、TUXEDO』 – X/Open XA トランザクション・マネージャを備えた Sybase DTM XA インタフェースを使用する方法について説明しています。
- 『Adaptive Server Enterprise における XML サービス』 – データベースに XML 機能を導入する、Sybase ネイティブの XML プロセッサと Sybase Java ベースの XML のサポートについて、また XML サービスに準拠したクエリとマッピング用の関数について説明しています。

その他の情報

Sybase Getting Started CD、SyBooks™ CD、Sybase Product Manuals Web サイトを利用すると、製品について詳しく知ることができます。

- Getting Started CD には、PDF 形式のリリース・ノートとインストール・ガイド、SyBooks CD に含まれていないその他のマニュアルや更新情報が収録されています。この CD は製品のソフトウェアに同梱されています。Getting Started CD に収録されているマニュアルを参照または印刷するには、Adobe Acrobat Reader が必要です (CD 内のリンクを使用して Adobe の Web サイトから無料でダウンロードできます)。
- SyBooks CD には製品マニュアルが収録されています。この CD は製品のソフトウェアに同梱されています。Eclipse ベースの SyBooks ブラウザを使用すれば、使いやすい HTML 形式のマニュアルにアクセスできます。

  一部のマニュアルは PDF 形式で提供されています。これらのマニュアルは SyBooks CD の PDF ディレクトリに収録されています。PDF ファイルを開いたり印刷したりするには、Adobe Acrobat Reader が必要です。

  SyBooks をインストールして起動するまでの手順については、Getting Started CD の『SyBooks インストール・ガイド』、または SyBooks CD の *README.txt* ファイルを参照してください。
- Sybase Product Manuals Web サイトは、SyBooks CD のオンライン版であり、標準の Web ブラウザを使用してアクセスできます。また、製品マニュアルのほか、EBFs/Updates、Technical Documents、Case Management、Solved Cases、ニュース・グループ、Sybase Developer Network へのリンクもあります。

  Technical Library Product Manuals Web サイトにアクセスするには、Product Manuals (http://www.sybase.com/support/manuals/) にアクセスしてください。

Web 上の Sybase 製品の動作確認情報

Sybase Web サイトの技術的な資料は頻繁に更新されます。

❖ **製品認定の最新情報にアクセスする**

1 Web ブラウザで Technical Documents を指定します。(http://www.sybase.com/support/techdocs/)

2 [Certification Report] をクリックします。

3 [Certification Report] フィルタで製品、プラットフォーム、時間枠を指定して [Go] をクリックします。

4 [Certification Report] のタイトルをクリックして、レポートを表示します。

❖ **コンポーネント認定の最新情報にアクセスする**

1 Web ブラウザで Availability and Certification Reports を指定します。(http://certification.sybase.com/)

2 [Search By Base Product] で製品ファミリとベース製品を選択するか、[Search by Platform] でプラットフォームとベース製品を選択します。

3 [Search] をクリックして、入手状況と認定レポートを表示します。

❖ **Sybase Web サイト (サポート・ページを含む) の自分専用のビューを作成する**

MySybase プロファイルを設定します。MySybase は無料サービスです。このサービスを使用すると、Sybase Web ページの表示方法を自分専用にカスタマイズできます。

1 Web ブラウザで Technical Documents を指定します。(http://www.sybase.com/support/techdocs/)

2 [MySybase] をクリックし、MySybase プロファイルを作成します。

Sybase EBF とソフトウェア・メンテナンス

❖ **EBF とソフトウェア・メンテナンスの最新情報にアクセスする**

1 Web ブラウザで Sybase Support Page (http://www.sybase.com/support) を指定します。

2 [EBFs/Maintenance] を選択します。MySybase のユーザ名とパスワードを入力します。

3 製品を選択します。

4 時間枠を指定して [Go] をクリックします。EBF/Maintenance リリースの一覧が表示されます。

鍵のアイコンは、「Technical Support Contact」として登録されていないため、一部の EBF/Maintenance リリースをダウンロードする権限がないことを示しています。未登録でも、Sybase 担当者またはサポート・コンタクトから有効な情報を得ている場合は、[Edit Roles] をクリックして、「Technical Support Contact」の役割を MySybase プロファイルに追加します。

5 EBF/Maintenance レポートを表示するには [Info] アイコンをクリックします。ソフトウェアをダウンロードするには製品の説明をクリックします。

表記規則

次の項では、このマニュアルで使用されている表記について説明します。

SQL は自由な形式の言語で、1 行内のワード数や、改行の仕方に規則はありません。このマニュアルでは、読みやすくするため、例や構文を文の句ごとに改行しています。複数の部分からなり、2 行以上にわたる場合は、字下げしています。複雑なコマンドの書式には、修正された BNF (Backus Naur Form) 記法が使用されています。

表 1 に構文の規則を示します。

表 *1:* このマニュアルでのフォントと構文規則

要素	例
コマンド名、プロシージャ名、ユーティリティ名、その他のキーワードは sans serif フォントで表記する。	select sp_configure
データベース名とデータ型は sans serif フォントで表記する。	master データベース
ファイル名、変数、パス名は斜体で表記する。	システム管理ガイド *sql.ini* ファイル *column_name* *$SYBASE/ASE* ディレクトリ

変数 (ユーザが入力する値を表す語) がクエリまたは文の一部である場合は Courier フォントの斜体で表記する。	`select column_name`
	`from table_name`
	`where search_conditions`

カッコはコマンドの一部として入力する。	compute *row_aggregate* (*column_name*)
2 つのコロンと等号は、構文が BNF 表記で記述されていることを示す。この記号は入力しない。「～と定義されている」ことを意味する。	`::=`
中カッコは、その中のオプションを 1 つ以上選択しなければならないことを意味する。コマンドには中カッコは入力しない。	`{cash, check, credit}`
角カッコは、オプションを選択しても省略してもよいことを意味する。コマンドには角カッコは入力しない。	`[cash | check | credit]`
中カッコまたは角カッコの中のカンマで区切られたオプションをいくつでも選択できることを意味する。複数のオプションを選択する場合には、オプションをカンマで区切る。	`cash, check, credit`

要素	例
パイプまたは縦線は複数のオプションのうち 1 つだけを選択できることを意味する。	`cash	check	credit`
省略記号 (...) は、直前の要素を必要な回数だけ繰り返し指定できることを意味する。	`buy thing = price [cash	check	credit] [, thing = price [cash	check	credit]]...` この例では、製品 (thing) を少なくとも 1 つ購入 (buy) し、価格 (price) を指定する必要があります。支払方法を選択できる。角カッコで囲まれた項目の 1 つを選択する。追加品目を、必要な数だけ購入することもできる。各 buy に対して、購入した製品 (thing)、価格 (price)、オプションで支払方法 (cash、check、credit のいずれか) を指定します。

- 次は、オプション句のあるコマンドの構文の例です。

  sp_dropdevice [*device_name*]

  複数のオプションを持つコマンドの例を示します。

  select *column_name*
  from *table_name*
  where *search_conditions*

  構文では、キーワード (コマンド) は通常のフォントで表記し、識別子は小文字で表記します。ユーザが提供するワードは斜体で表記します。

- Transact-SQL コマンドの使用例は次のように表記します。

  ```
  select * from publishers
  ```

- 次は、コンピュータからの出力例です。

```
pub_id    pub_name                 city          state
------    --------------------     -----------   -----
0736      New Age Books            Boston        MA
0877      Binnet & Hardley         Washington    DC
1389      Algodata Infosystems     Berkeley      CA

(3 rows affected)
```

このマニュアルでは、例に使用する文字はほとんどが小文字ですが、Transact-SQL のキーワードを入力するときは、大文字と小文字は区別されません。たとえば、SELECT、Select、select はすべて同じです。

テーブル名などのデータベース・オブジェクトの大文字と小文字を Adaptive Server が区別するかどうかは、Adaptive Server にインストールされたソート順によって決まります。シングルバイト文字セットを使用している場合は、Adaptive Server のソート順を再設定することによって、大文字と小文字の区別の取り扱い方を変更できます。詳細については、『システム管理ガイド』を参照してください。

アクセシビリティ機能

このマニュアルには、アクセシビリティを重視した HTML 版もあります。この HTML 版マニュアルは、スクリーン・リーダーで読み上げる、または画面を拡大表示するなどの方法により、その内容を理解できるよう配慮されています。

Adaptive Server HTML マニュアルは、連邦リハビリテーション法第 508 条のアクセシビリティ規定に準拠していることがテストにより確認されています。第 508 条に準拠しているマニュアルは通常、World Wide Web Consortium (W3C) の Web サイト用ガイドラインなど、米国以外のアクセシビリティ・ガイドラインにも準拠しています。

注意 アクセシビリティ・ツールを効率的に使用するには、設定が必要な場合もあります。一部のスクリーン・リーダーは、テキストの大文字と小文字を区別して発音します。たとえば、すべて大文字のテキスト (ALL UPPERCASE TEXT など) はイニシャルで発音し、大文字と小文字の混在したテキスト (Mixed Case Text など) は単語として発音します。構文規則を発音するようにツールを設定すると便利かもしれません。詳細については、ツールのマニュアルを参照してください。

Sybase のアクセシビリティに対する取り組みについては、Sybase Accessibility (http://www.sybase.com/accessibility) を参照してください。Sybase Accessibility サイトには、第 508 条と W3C 標準に関する情報へのリンクもあります。

不明な点があるときは

Sybase ソフトウェアがインストールされているサイトには、Sybase 製品の保守契約を結んでいるサポート・センタとの連絡担当の方 (コンタクト・パーソン) を決めてあります。マニュアルだけでは解決できない問題があった場合には、担当の方を通して Sybase のサポート・センタまでご連絡ください。

第 1 章

概要

Adaptive Server には、バージョン 15.0.3 以前より、次に示す機能を実現するために、いくつかの分散トランザクション管理機能が導入されています。

- XA-Server のような追加のサービスを必要とせずにリソース・マネージャとして機能するように、Adaptive Server を X/Open XA プロトコルに完全に準拠させました。
- Microsoft 分散トランザクション・コーディネータ (MSDTC) によってコーディネートされる分散トランザクションをサポートします。
- Adaptive Server のデータをリモート・プロシージャ・コール (RPC) とコンポーネント統合サービス (CIS) を介して更新するトランザクションに対して、一貫したコミットとロールバックを保証します。
- 将来追加する分散トランザクション管理プロトコルをサポートするための枠組みを提供します。

この章では、新しい分散トランザクション管理機能およびそれらの機能をサポートする Adaptive Server への変更点について説明します。

分散トランザクション管理機能

Adaptive Server には、次に示す分散トランザクション管理機能が導入されています。

- 改良されたトランザクションおよびスレッド管理。Adaptive Server はすべてのトランザクションをサーバ・リソースとして管理し、トランザクションにスレッドを付加および分離する機能を提供します。これらの新しい機能は、ローカル・サーバ・トランザクションのクライアントおよび X/Open XA と MSDTC 機能のクライアントをサポートする共通のインタフェースを提供します。「トランザクション・リソースの設定」(8 ページ) を参照してください。

- 新しい分散トランザクション・コーディネーション・サービス。Adaptive Server は、RPC と CIS を通してリモート Adaptive Server のデータを修正するトランザクションに、一貫したロールバックとコミット機能を提供します。新しいトランザクション・コーディネーション・サービスは、外部トランザクション・マネージャが存在しなくても、そのような分散トランザクションの整合性を保証します。「第 3 章 Adaptive Server トランザクション・コーディネーション・サービスの使用」を参照してください。
- 準備済みトランザクション用に改良されたリカバリ。リカバリ中、Adaptive Server は、X/Open XA プロトコルと Adaptive Server のネイティブ・トランザクション・コーディネーション・サービスによってコーディネートされた準備済みトランザクションを識別します。Adaptive Server はリカバリ前の状態にこれらのトランザクションをリストアし、以前のバージョンのサーバよりも短時間で、関連するデータベースをオンラインにします。「分散トランザクションにおけるクラッシュのリカバリ手順」(29 ページ) を参照してください。
- 分散トランザクションを自発的に完了する新しい dbcc コマンド。「トランザクションの自発的完了」(31 ページ) を参照してください。

影響を受けるトランザクションのタイプ

新しい Adaptive Server の DTM 機能は、次に示すトランザクションに影響を与えます。

- 外部トランザクション・マネージャによってコーディネートされる分散トランザクション
- RPC および CIS を使用してデータを更新するトランザクション

外部トランザクション・マネージャによってコーディネートされる分散トランザクション

分散トランザクションは、外部トランザクション・マネージャが X/Open XA などの特定のプロトコルを使用してトランザクションの実行をコーディネートする環境で発生します。Adaptive Server は、Adaptive Server への DTM XA インタフェースを介した CICS、Encina、TUXEDO、MSDTC トランザクション・マネージャを使用して、トランザクションをサポートします。

注意 DTM XA インタフェースを持つ Adaptive Server は、以前 XA-Server 製品の一部であった機能を提供します。現在は、XA-Server 製品ではなく、Adaptive Server のオプションとして提供されています。DTM XA インタフェースの詳細については、『XA インタフェース統合ガイド for CICS、Encina、TUXEDO』を参照してください。

トランザクション・マネージャがコーディネートするトランザクションの動作

Adaptive Server は、XA-Library と XA-Server 製品の一部であったいくつかの機能をネイティブに実装し、X/Open XA プロトコルを通してコーディネートされる準備済みトランザクションに新しいリカバリ手順を提供します。詳細については、「トランザクション・リソースの設定」(8 ページ) および「分散トランザクションにおけるクラッシュのリカバリ手順」(29 ページ) を参照してください。

Adaptive Server への XA インタフェースは、サーバの新しい分散トランザクション管理機能を実行できるように修正されました。XA インタフェースへの変更は、X/Open XA クライアント・アプリケーションに対して透過的です。しかし、リソース・マネージャとして Adaptive Server を使用するには、X/Open XA トランザクション・マネージャに Adaptive Server の DTM XA インタフェースをリンクする必要があります。すべての XA インタフェースの変更の詳細については、『XA インタフェース統合ガイド for CICS、Encina、TUXEDO』を参照してください。

また、Adaptive Server には、MSDTC によってコーディネートされる分散トランザクションのサポートが導入されています。MSDTC クライアントは、ネイティブ・インタフェースを使用して Adaptive Server に直接通信できます。また、クライアントは DTM XA インタフェースを使用することによって、UNIX で実行している 1 つ以上の Adaptive Server と通信できます。

注意 DTM XA インタフェースを使用する MSDTC クライアントは、アクセスする Adaptive Server で dtm_tm_role を処理する必要があります。dtm_tm_role の詳細については、『XA インタフェース統合ガイド for CICS、Encina、TUXEDO』を参照してください。

Adaptive Server バージョン 15.0.3 以降の拡張されたトランザクション・マネージャ

バージョン 15.0.3 よりも前の Adaptive Server では、Adaptive Server が外部トランザクションを暗黙にアボートしたことをアプリケーションが認識しない場合、通常はこのトランザクション内で実行されるはずの DML コマンドが、明示的なトランザクションの外側で実行される可能性があります。これらの DML コマンドは Adaptive Server が起動した暗黙のトランザクション内で実行されるため、ビジネス・データに矛盾が生じることになります。この問題に対応するには、外部トランザクションがアクティブになっているかどうかをユーザ・アプリケーション側で常にチェックし、状況に応じてコマンドを発行する必要があります。

バージョン 15.0.3 以降では、外部トランザクションの暗黙なロールバックが発生した場合、トランザクション・マネージャからの分離要求がないかぎり、Adaptive Server はこの外部トランザクションに関連付けられている接続での DML コマンドの実行を許可しません。分離要求は、コマンドのバッチの終わりで外部トランザクションを実行することを意味します。

15.0.3 以降では、Adaptive Server は、分散トランザクション内で実行されるべき SQL コマンドが、分散トランザクションの外側で実行されないように自動的に制御します。つまり、各コマンドの実行前にユーザ・アプリケーションでグローバル変数 *@@trancount* をチェックして、トランザクションが暗黙にアボートされたかどうかを確認する必要がなくなりました。トランザクションが暗黙にアボードされると、エラー・メッセージ (3953)「外部トランザクションがロールバックされたため、コマンドを実行できません」が表示されます。detach transaction コマンドが発行されると、このメッセージの表示が消えます。

3953 エラー・メッセージの表示を消し、Adaptive Server で以前の動作をリストアする (つまり、DTM トランザクションがアクティブでない場合であっても SQL コマンドを実行する) には、トレース・フラグ -T3995 を使用して Adaptive Server を起動します。

RPC および CIS トランザクション

ローカル Adaptive Server トランザクションは、Transact-SQL リモート・プロシージャ・コール (RPC) およびコンポーネント統合サービス (CIS) を使用することによって、リモート・サーバにあるデータを更新できます。RPC 更新は、ローカルに作成されたトランザクション内から RPC を実行することによって行います。次に例を示します。

```
sp_addserver westcoastsrv, ASEnterprise, hqsales
begin transaction rpc_tran1
update sales set commission=300 where salesid="12OZ"
exec westcoastsrv.salesdb..recordsalesproc
commit rpc_tran1
```

上記のトランザクションは、ローカル Adaptive Server にある sales テーブルを更新しますが、RPC の recordsalesproc を使用してリモート・サーバにあるデータも更新します。

CIS は、これらのテーブルがローカルな場合に、リモート・テーブルにあるデータを更新する方法を提供します。sp_addobjectdef を使用することによって、リモート・データを参照する Adaptive Server にローカル・オブジェクトを作成することができます。ローカル・オブジェクトの更新は、リモート Adaptive Server にあるデータを修正します。次に例を示します。

```
sp_addobjectdef salesrec,
"westcoastsrv.salesdb..sales", "table"
begin transaction cis_tran1
update sales set commission=300 where salesid="12OZ"
update salesrec set commission=300 where salesid="12OZ"
commit cis_tran1
```

RPC および CIS トランザクションの新しい動作

バージョン 12.0 より前の Adaptive Server では、RPC と CIS を通してデータを更新するトランザクションはリモート・サーバの作業をロールバックできず、またこれらのトランザクションでリモート作業が実際にコミットされることを保証することもできませんでした。Adaptive Server は、新しいトランザクション・コーディネーション・サービスを提供することによって、RPC や CIS 更新が初期トランザクションでの作業のコミットまたはロールバックを保証します。詳細については、「第 3 章 Adaptive Server トランザクション・コーディネーション・サービスの使用」を参照してください。

RPC と CIS 更新の以前の動作を利用するアプリケーションに対しては、トランザクション・コーディネーション・サービスを無効にすることができます。詳細については、「enable xact coordination パラメータ」(8 ページ) を参照してください。

SYB2PC トランザクション

SYB2PC トランザクションは、Sybase 2 フェーズ・コミット・プロトコルを使用して、論理単位としてコミットまたはロールバックされる分散トランザクションの作業を保証します。

Adaptive Server は、SYB2PC トランザクションの動作を変更していません。しかし、SYB2PC トランザクションを実装するアプリケーション開発者によっては、代わりに Adaptive Server トランザクション・コーディネーション・サービスの使用を検討したい場合もあります。SYB2PC トランザクションと比較すると、Adaptive Server によって直接コーディネートされるトランザクションは、分散トランザクションの整合性を保証しながら、より少ないネットワーク接続を使用して、より短時間で実行されます。また、アプリケーションではなく Adaptive Server がリモート・トランザクションをコーディネートする場合は、アプリケーション・コードをよりシンプルにすることができます。詳細については、「第 3 章 Adaptive Server トランザクション・コーディネーション・サービスの使用」を参照してください。

第 2 章

DTM 機能の有効化

この章では、Adaptive Server の DTM 機能を有効にする方法を説明します。

ライセンス・キーのインストール

分散トランザクション管理は、Adaptive Server 機能とは別のライセンスとして使用できます。DTM 機能を有効にして使用する前に、Adaptive Server と DTM 機能の両方の有効なライセンスを購入し、インストールする必要があります。

ライセンス・キーのインストールと Sybase ソフトウェア資産管理 (SySAM) の使用については、『インストール・ガイド』を参照してください。DTM のライセンスまたは他のライセンス付き Adaptive Server 機能をご購入になる場合は、Sybase 営業担当者までご連絡ください。

DTM 機能の有効化

Adaptive Server および DTM 機能の有効なライセンスを購入して、インストールした後、sp_configure に設定パラメータ enable dtm と enable xact coordination を使用して DTM 機能を有効にできます。

enable dtm パラメータ

enable dtm パラメータにより、基本的な DTM 機能を有効にしたり無効にしたりできます。enable dtm を 1 (オン) に設定すると、Adaptive Server は MSDTC および DTM XA インタフェースを介した X/Open XA トランザクション・マネージャからの外部トランザクションをサポートします。詳細については、『XA インタフェース統合ガイド for CICS、Encina、TUXEDO』を参照してください。

基本的な DTM 機能を有効にするには、次のコマンドを使用します。

```
sp_configure 'enable dtm', 1
```

この変更を有効にするには、Adaptive Server を再起動します。

enable xact coordination パラメータ

enable xact coordination は、Adaptive Server トランザクション・コーディネーション・サービスを有効にしたり無効にしたりします。このパラメータを有効にすると、Adaptive Server では、リモート Adaptive Server データへの更新は、オリジナルのトランザクションでコミットまたはロールバックするようになります。詳細については、「第 3 章 Adaptive Server トランザクション・コーディネーション・サービスの使用」を参照してください。

トランザクション・コーディネーションを有効にするには、次のコマンドを使用します。

```
sp_configure 'enable xact coordination', 1
```

この変更を有効にするには、Adaptive Server を再起動します。

トランザクション・リソースの設定

Adaptive Server は、ローカル・サーバ・トランザクションと分散トランザクション・プロトコルによってコーディネートされる外部トランザクションの両方をサポートする、共通のインタフェースを提供します。分散トランザクション・プロトコルは、X/Open XA、MSDTC、ネイティブ Adaptive Server トランザクション・コーディネーション・サービスによってサポートされます。

Adaptive Server はすべてのトランザクションを設定可能なサーバ・リソースとして管理し、システム管理者は任意のサーバの使用可能リソースの合計数を設定できます。X/Open XA 環境で Adaptive Server にアクセスするクライアント・タスクは、必要に応じてトランザクション・リソースのスレッドを中断したりジョインしたりできます。

この項では、Adaptive Server が利用可能なトランザクション・リソースの合計数を決定したり設定したりする方法について説明します。

必要なトランザクション記述子の計算

Adaptive Server はトランザクション記述子リソースを使用して、サーバ内でのトランザクションを管理します。トランザクション記述子は内部メモリ構造で、Adaptive Server はこれを使用してトランザクションを表現します。

起動時に、Adaptive Server は設定パラメータ txn to pss ratio の値に基づいて固定数のトランザクション記述子を割り当て、プールに配置します。Adaptive Server は、新しいトランザクションに対して必要になったときに、プールからトランザクション記述子を取得します。トランザクションが完了すると、記述子はプールに返却されます。使用できるトランザクション記述子がない場合は、Adaptive Server が記述子の解放を待つ間、トランザクションは遅延されます。

トランザクション記述子の数を適切に設定するには、グローバル・プールから Adaptive Server が新しい記述子をいつ取得しようとするのか、正確に理解することが重要です。新しいトランザクション記述子は、次のような場合に必要となります。

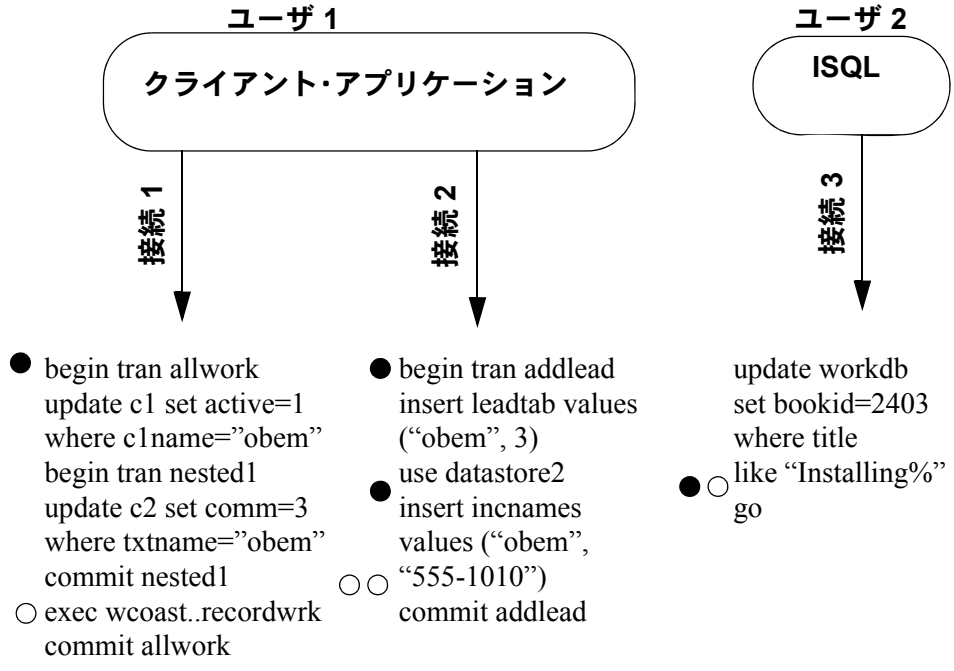
- クライアント接続が、新しい外部レベルのトランザクションを開始する場合。これは、クライアントが外部レベルの begin transaction コマンドを実行するときに、明示的に発生します。クライアントが begin transaction コマンドを入力せずにデータを修正した場合、暗黙的に発生することもあります。

  外部レベルのトランザクションが開始されると、その後のネストした begin transaction コマンドにはそれ以上のトランザクション記述子は必要ありません。トランザクション記述子の割り付けと割り付け解除は、トランザクションの最も外側のブロックによって指示されます。

- 既存のトランザクションが、2 番目のデータベースを修正した場合 (マルチデータベース・トランザクション)。マルチデータベース・トランザクションには、アクセスする各データベースに対して、専用トランザクション記述子が必要です。

図 2-1 は、Adaptive Server が異なるタイプのトランザクションに対して、トランザクション記述子を取得し、解放する方法について示しています。

図 2-1: トランザクション記述子の割り付けおよび割り付け解除



● 新しいトランザクション記述子が取得されたことを示す

○ トランザクション記述子が解放されたことを示す

図 2-1 では、Adaptive Server は 1 組のクライアント接続を介してサーバにアクセスするユーザ 1 に対して、合計 3 つのトランザクション記述子を使用しています。サーバは、トランザクション allwork に対して 1 つの記述子を割り付けます。記述子は、トランザクションがコミットされたときに解放されます。ネストされたトランザクション nested1 は、専用トランザクション記述子を必要としません。

トランザクション addlead はマルチデータベース・トランザクションで、2 つのトランザクション記述子を必要とします。1 つは、外部トランザクション・ブロック用で、もう 1 つは 2 番目のデータベース datastore2 の修正用です。このトランザクション記述子は両方とも、外部トランザクション・ブロックがコミットしたときに解放されます。

isql から Adaptive Server にアクセスしているユーザ 2 も、専用トランザクション記述子が必要です。ユーザ 2 は begin transaction を使用して外部トランザクション・ブロックを明示的に作成しませんが、Adaptive Server はトランザクション・ブロックを暗黙的に作成して、update コマンドを実行します。このブロックに関連するトランザクション記述子は、go コマンドの後に取得され、挿入が完了した後に解放されます。

トランザクション記述子は、ほかの Adaptive Server サービスが使用する分のメモリを消費することがあるので、常に、必要となるトランザクションの最大数に足りるだけの記述子を使用することが重要です。

トランザクション記述子の数の設定

システムで使用するトランザクション記述子の数を決定したら、sp_configure を使用して txn to pss ratio の値を設定します。txn to pss ratio は、サーバで使用できるトランザクション記述子の合計数を決定します。起動時に、この比率に number of user connections パラメータを掛け合わせ、トランザクション記述子プールを作成します。

```
# of transaction descriptors = number of user connections * txn
to pss ratio
```

txn to pss ratio のデフォルト値 16 は、Adaptive Server の以前のバージョンとの互換性を保ちます。バージョン 12.0 より前の Adaptive Server では、各ユーザ接続に対して 16 個のトランザクション記述子を割り付けていました。バージョン 12.0 以降では、同時トランザクションの数は、サーバで利用できるトランザクション記述子の数のみによって制限されます。

たとえば、各ユーザ接続に対して 25 個のトランザクション記述子を割り付けるには、次のコマンドを使用します。

```
sp_configure 'txn to pss ratio', 25
```

この変更を有効にするには、Adaptive Server を再起動します。

第 3 章

Adaptive Server トランザクション・コーディネーション・サービスの使用

この章では、Adaptive Server トランザクション・コーディネーション・サービスを設定し、使用する方法を説明します。

トランザクション・コーディネーション・サービスの概要

ローカル Adaptive Server トランザクションの作業は、リモート・データを修正するリモート・サーバに分散されることがあります。これは、ローカル・トランザクションが別の Adaptive Server テーブルにあるデータを更新するためにリモート・プロシージャ・コール (RPC) を実行したり、ローカル・トランザクションがコンポーネント統合サービス (CIS) を使用してリモート・テーブルにあるデータを修正する場合に発生します。

バージョン 12.0 より前の Adaptive Server では、RPC を実行したローカル・トランザクションまたは CIS でデータを更新したローカル・トランザクションは、リモート Adaptive Server で行われた作業をロールバックできませんでした。さらに、ローカル・トランザクションを実行するクライアントは、たとえばリモート・サーバがシステム障害を検出したときなどに、リモート作業が実際にコミットされることを保証できませんでした。

Adaptive Server は、リモート・サーバにトランザクションを送信し、すべてのサーバの作業をコーディネートするサービスを提供し、すべての作業が論理単位としてコミットまたはロールバックされることを保証します。トランザクション・コーディネーション・サービスを使用すると、Adaptive Server 自身が複数の Adaptive Server にあるデータを更新するトランザクションの分散トランザクション・マネージャとして動作できます。

階層的なトランザクション・コーディネーション

分散トランザクションに関連する別の Adaptive Server はリモートのパティシパントもコーディネートするため、トランザクションは階層的な方法でさらに追加のサーバに送信できます。たとえば図 3-1 では、ASE1 に接続するクライアントは、ASE2 で RPC を実行し、ASE3 で RPC を実行するトランザクションを開始します。ASE1 のコーディネーション・サービスは、ASE2 と ASE3 にトランザクションを送信します。

ASE2 はトランザクション・コーディネーション・サービスも有効なため、追加のリモート・パティシパントにトランザクションを送信できます。ここで、ASE2 は CIS を使用してデータが更新される ASE4 にトランザクションを送信します。

図 *3-1:* 階層的なトランザクション・コーディネーション

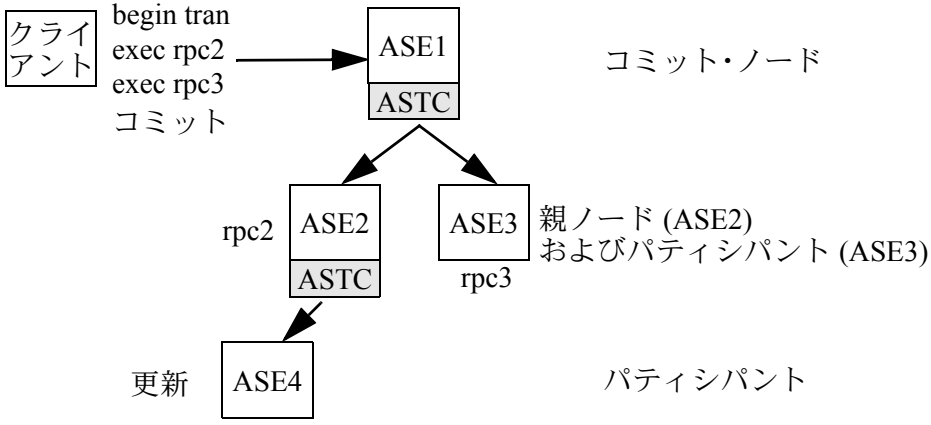


図 3-1 では、ASE1 は分散トランザクションのコミット・ノードとしています。ASE1 のトランザクションがコミットしたとき、ASE1 のコーディネーション・サービスは ASE2 と ASE3 にトランザクションを送信する準備をするよう指示します。ASE3 は、ローカル作業のコミットが準備されたときに、そのトランザクションが準備されることを示します。ASE2 はそのローカル作業を完了する必要があり、ASE3 にそのトランザクションを準備するよう指示します。トランザクションが ASE2 と ASE3 で準備されると、ASE1 のコーディネーション・サービスはオリジナルのトランザクションをコミットします。次に、従属するトランザクションをコミットする指示が ASE2、ASE3、および最終的に ASE3 に転送され、準備のための指示と同じ方法で転送されます。

DTP 環境における X/Open XA 準拠の動作

X/Open XA プロトコルは、リモート・リソース・マネージャに送信されるトランザクションのコーディネーション・サービスを提供するリソース・マネージャが必要です。この要件は、外部トランザクション・マネージャ (場合によっては、トランザクションが発生するクライアント) がトランザクションのリモート・サーバへの送信を認識せず、リモート・トランザクションが必要に応じて完了またはアボートすることを保証できないために必要です。

新しいトランザクション・コーディネーション・サービスを使用すると、Adaptive Server が X/Open XA プロトコルに完全に準拠したリソース・マネージャとしての役割を果たすようにすることができます。分散トランザクションは RPC や CIS を使用してリモート・サーバに暗黙的に送信でき、Adaptive Server はコーディネートするリモート・サーバにグローバル・トランザクションのコミットまたはロールバック・ステータスを保存することを保証します。

要件と動作

Adaptive Server トランザクション・コーディネーション・サービスは、各リモート Adaptive Server がバージョン 12.0 以降の場合、リモート・サーバの作業が論理的にコミットまたはロールバックされることを保証できます。

トランザクション・コーディネーション・サービスは、分散トランザクションを実行するクライアントに対して透過的です。ローカル・クライアント・トランザクションが RPC を実行したり、CIS を使用してデータを更新したりする場合、コーディネーション・サービスはリモート作業に新しいトランザクション名を作成し、従属するリモート・サーバにそのトランザクションを送信します。ローカル・クライアントがローカル・トランザクションをコミットまたはロールバックすると、Adaptive Server は従属サーバのそれぞれにその要求をコーディネートし、リモート・トランザクションが同じようにコミットまたはロールバックされることを保証します。

Adaptive Server トランザクション・コーディネーション・サービスは、“ASTC HANDLER” という 1 つ以上のバックグラウンド・タスクとして実行され、sp_who を使用して参照できます。複数の Adaptive Server エンジンを使用するシステムでは、“ASTC HANDLER” プロセスの数 (最も近い整数に丸められます) は、次のようになります。

```
number of engines * 2/3
```

Adaptive Server で実行される “ASTC HANDLER” プロセスの最大数は 4 です。

次に示す sp_who からの出力は、1 つの “ASTC HANDLER” を表示します。

```
                          sp_who
fid spid status loginname origname hostname blk_spid dbname cmd block_xloid
--------------------------------------------------------------------------------
0   1    running  sa      sa        dtmsol1  0        master SELECT          0
0   2    sleeping NULL    NULL      master   0        master NETWORK HANDLER 0
0   3    sleeping NULL    NULL               0        master DEADLOCK TUNE   0
0   4    sleeping NULL    NULL               0        master MIRROR HANDLER  0
0   5    sleeping NULL    NULL               0        master HOUSEKEEPER     0
0   6    sleeping NULL    NULL               0       master CHECKPOINT SLEEP 0
0   7    sleeping NULL    NULL    metin1_dtm 0     sybsystemdb ASTC HANDLER  0
```

パティシパント・サーバ・リソースの設定

デフォルトでは、トランザクション・コーディネーション・サービスは常に有効です。システム管理者は、enable xact coordination 設定パラメータを使用してこれらのサービスを有効または無効にできます。このパラメータの完全な説明については、『システム管理ガイド』を参照してください。

また、システム管理者は、トランザクションによって要求される可能性のあるすべての RPC および CIS 更新のコーディネートに必要なリソースが Adaptive Server にあることを保証する必要があります。トランザクションが RPC または CIS 更新を発行するたびに、トランザクション・コーディネータはフリーの「**DTX** パティシパント」を取得する必要があります。DTX パティシパントまたは「分散トランザクション・パティシパント」は、Adaptive Server が従属 Adaptive Server に送信されたトランザクションをコーディネートするのに使用する内部メモリ構造体です。図 3-1 では、ASE1 は 3 つのフリー DTX パティシパントが必要で、ASE2 は 2 つのフリー DTX パティシパントが必要です (この場合、1 つの DTX パティシパントは送信されるトランザクションのローカル作業をコーディネートするのに使用されます)。

DTX パティシパント・リソースは、関連するリモート・トランザクションがコミットされるまで、コーディネーティング Adaptive Server によって使用されたままになります。初期トランザクションはすべての従属トランザクションが正常に作業の準備を整えるとすぐにコミットするため、これは一般に、初期トランザクションがコミットされたあとのある時期に発生します。

使用できる DTX パティシパントがない場合、RPC 要求と CIS 更新要求は処理されず、トランザクションがアボートされます。

number of dtx participants パラメータ

システム管理者は、number of dtx participants 設定パラメータを使用して、Adaptive Server で使用できる DTX パティシパントの合計数を設定できます。number of dtx participants は、Adaptive Server トランザクション・コーディネーション・サービスが一度に送信およびコーディネートできるリモート・トランザクションの合計数を設定します。

デフォルトでは、Adaptive Server は 500 のリモート・トランザクションを調整できます。number of dtx participants の設定値を小さくすると、サーバが管理できるリモート・トランザクションの数が減少します。使用できる DTX パティシパントがない場合、新しい分散トランザクションは開始できません。新しいリモート・トランザクションを送信するために使用できる DTX パティシパントがない場合は、進行中の分散トランザクションがアボートすることがあります。

number of dtx participants の設定値を大きくすると、Adaptive Server が処理できるリモート・トランザクション分岐の数が増加しますが、メモリの消費量も増加します。

使用しているシステムの *number of dtx participants* の最適化

ピークの時間帯に、sp_monitorconfig を使用して DTX パティシパントの使用状況を調査します。

```
                sp_monitorconfig "number of dtx participants"
Usage Information at date and time:Jun 18 1999 9:00AM.
Name             # Free    # Active % Active   # Max Ever Used   Re-used
--------------   --------  ------   --------   ---------------   -------
number of dtx    480       20       4.00       210               N/A
participants
```

#Free の値がゼロまたは非常に低い場合、新しい分散トランザクションは DTX パティシパントの不足により、開始できない可能性があります。この場合は、number of dtx participants の値を増やしてください。

#Max Ever Used の値が非常に低い場合、未使用の DTX パティシパントによってメモリが消費されているため、他のサーバ機能がメモリを使用できなくなっている可能性があります。この場合には、number of dtx participants の値を減らしてください。

異機種間環境でのトランザクション・コーディネーション・サービスの使用

Adaptive Server が別のバージョン 12.0 以降の Adaptive Server にトランザクションを送信する場合、全体としての分散トランザクションの整合性を保証できます。しかし、ローカル Adaptive Server トランザクションの作業は、バージョン 12.0 以降のトランザクション・コーディネーション・サービスをサポートしないリモート・サーバに分散されることがあります。これは、トランザクションが RPC を使用して前のバージョンの Adaptive Server でデータを更新したり、CIS サービスを使用して Sybase 以外のデータベースでデータを更新したりする場合に発生します。このような状況では、コーディネーティング Adaptive Server は、リモート・サービスの作業がオリジナル・トランザクションでコミットまたはロールバックされることを保証できません。

strict dtm enforcement パラメータ

Adaptive Server では、システム管理者は strict dtm enforcement 設定パラメータを設定することによって、分散トランザクションを論理単位としてコミットまたはロールバックする要件を強制または緩和することができます。

注意 また、セッション・レベルの set コマンドと strict_dtm_enforcement オプションを使用して、strict dtm enforcement の値を上書きできます。

---

strict dtm enforcement は、Adaptive Server トランザクション・コーディネーション・サービスが分散トランザクションの ACID プロパティを厳密に適用するかどうかを決定します。

strict dtm enforcement を 1 (オン) に設定すると、Adaptive Server がコーディネートするトランザクションに参加できるサーバだけに、トランザクションが送信されることを保証します。トランザクション・コーディネーション・サービスをサポートしないサーバにあるデータをトランザクションが更新しようとすると、Adaptive Server はそのトランザクションをアボートします。

異機種間環境では、トランザクション・コーディネーションをサポートしないサーバを使用することがあります。これには、Adaptive Server の古いバージョンや CIS を使用して設定された Sybase 以外のデータベース・ストアも含まれます。これらの状況では、strict dtm enforcement を 0 (オフ) に設定できます。これによって、Adaptive Server は従来の Adaptive Server や他のデータ・ストアにトランザクションを送信できますが、これらのサーバのリモート作業がオリジナル・トランザクションでロールバックまたはコミットされることは保証しません。

コーディネートされたトランザクションとパティシパントのモニタリング

Adaptive Server は新しいシステム・テーブル sybsystemdbdbo.syscoordinations にあるデータを使用して、従属サーバで行われた作業のステータス情報を追跡します。このテーブルの完全な説明については、『リファレンス・マニュアル』を参照してください。

sp_transactions プロシージャも、syscoordinations テーブルから、処理中のリモート・トランザクションについてのいくつかのデータを表示します。詳細については、「分散トランザクション情報の取得」(23 ページ) を参照してください。

第 4 章

DTM 管理とトラブルシューティング

この章では、Adaptive Server DTM 機能のモニタリング、管理、トラブルシューティングの方法について説明します。

トランザクションと制御スレッド

Adaptive Server バージョン 12.0 より前では、すべてのトランザクション・リソースは 1 つのサーバ・タスクによってプライベートに専有されていました。サーバは、トランザクションを開始したタスク以外のタスクを持つトランザクションを共有できませんでした。

Adaptive Server バージョン 12.5 以降では、Encina や TUXEDO などの X/Open XA 準拠のトランザクション・マネージャが使用する、「サスペンド」と「ジョイン」セマンティックのネイティブ・サポートが利用できます。トランザクションは異なる実行スレッド間で共有したり、関連するスレッドをまったく持たないようにすることもできます。

トランザクションが関連するスレッドを持たない場合、その状態を「分離されている」と表現します。分離されているトランザクションは、spid 値 0 が割り当てられます。トランザクションの spid 値は、新しい master.dbo.systransactions テーブルまたは新しい sp_transactions プロシージャの出力で参照できます。詳細については、「分散トランザクション情報の取得」(23 ページ) を参照してください。

システム管理者の作業

分離されているトランザクションは、クライアント・アプリケーションがトランザクションにオリジナル・スレッドを再度付加したり、新しいスレッドを付加できるようにするため、Adaptive Server に保持されます。スレッドはトランザクションに付加されていないため、システム管理者は関連する spid を強制終了してトランザクションをロールバックできなくなりました。

分離状態のトランザクションは、dump transaction コマンドを使用してログがトランケートされるのを防ぐことができます。極端な状況では、自発的にトランザクションを完了する新しい dbcc complete_xact コマンドを使用して、分離されたトランザクションをロールバックできます。詳細については、「トランザクションの自発的完了」(31 ページ) を参照してください。

dtm detach timeout period パラメータ

システム管理者は、トランザクションが分離されてから Adaptive Server が自動的にロールバックするまでサーバワイドな時間を指定することもできます。dtm detach timeout period は、分散トランザクション分岐を分離状態に維持する時間を、分単位で設定します。この時間が過ぎると、Adaptive Server は分離されたトランザクションをロールバックします。

たとえば、分離の 30 分後に自動的にロールバックするには、次のコマンドを使用します。

```
sp_configure 'dtm detach timeout period', 30
```

分離されたトランザクションをサポートするロック・マネージャへの変更点

バージョン 12.0 以前の Adaptive Server では、ロック・マネージャはトランザクション・スレッドの値 spid を使用してトランザクションのロックをユニークに識別できました。新しいトランザクション・マネージャでは、トランザクションはオリジナルのスレッドから分離でき、関連する spid を持ちません。さらに、異なる spid 値を持つ複数のスレッドが同じトランザクション・ロックを共有し、分散トランザクションの作業を実行できなければなりません。

これらの変更を反映するため、Adaptive Server バージョン 12.5 以降のロック・マネージャは、spid ではなく、ユニークなロック所有者 ID を使用してトランザクションのロックを識別します。ロック所有者 ID は、トランザクションを作成した spid とは独立しており、トランザクションがスレッドから分離されても保持されます。ロックの所有者 ID を使って、トランザクションが関連するスレッドを持たない場合、または新しいスレッドがトランザクションに付加される場合に、トランザクションのロックをサポートできます。

ロックの所有者 ID は、master.dbo.syslocks の新しい loid カラムに保管されます。トランザクションの loid 値を確認するには、sp_lock または sp_transactions 出力を調べます。

sp_transactions 出力からの spid と loid カラムを調べると、トランザクションとその制御スレッドについての情報を取得できます。spid 値がゼロの場合は、トランザクションがその制御スレッドから分離されていることを示します。spid 値がゼロ以外の場合は、現在制御スレッドがトランザクションに付加されていることを示します。

sp_transactions 出力の loid 値が偶数のときは、ローカル・トランザクションがロックを所有しています。loid 値が奇数のときは、外部トランザクションがロックを所有していることを示します。

sp_transactions 出力の詳細については、「分散トランザクション情報の取得」(23 ページ) を参照してください。

分散トランザクション情報の取得

Adaptive Server には、すべてのサーバ・トランザクションについての情報を保管するシステム・テーブル master.dbo.systransactions があります。systransactions は、各トランザクションを識別し、トランザクションの状態とトランザクションに関連するスレッドについての情報を管理します。

新しいシステム・プロシージャ sp_transactions は、systransactions と syscoordinations テーブルからの情報を変換して、アクティブなトランザクションのステータス状況を表示します。

systransactions のトランザクション識別

Adaptive Server は varchar(255) のカラム (前バージョンでは varchar(64)) にトランザクション名を保管し、異なる分散トランザクション・プロトコルから提供されるトランザクション名の長さとフォーマットを収めます。たとえば、X/Open XA プロトコルでは、分散トランザクションにはグローバル・トランザクション ID (gtrid) と分岐修飾子の両方から構成されるトランザクション名が割り当てられます。Adaptive Server 内では、この情報は systransactions テーブルの xactname カラムに結合されます。

systransactions.xactname は、X/Open XA トランザクション・マネージャまたは MSDTC によって定義される外部作成の分散トランザクションと、ローカル・サーバ・トランザクションの両方の名前を保管します。ローカル・トランザクションを定義するクライアントは、これらのトランザクションに varchar(255) カラムの制限内で任意の名前を付けることができます。同じように、外部トランザクション・マネージャは、さまざまなフォーマットを使用して分散トランザクションの名前を付けられます。

トランザクション・キー

トランザクション・キーは systransactions の xactkey カラムに保管され、サーバ・トランザクションへのユニークな内部ハンドルとして動作します。ローカル・トランザクションの場合、xactkey はトランザクション名がサーバに対してユニークでないときも、トランザクションがお互いに区別できることを保証します。

Adaptive Server バージョン 12.0 から、すべてのシステム・テーブルは systransactions.xactkey を参照してユニークにトランザクションを識別します。sysprocesses と syslogshold テーブルは、この規則の唯一の例外です。これらのテーブルは、systransactions.xactname を参照し、varchar(64) (sysprocesses の場合) と varchar(67) (syslogshold の場合) の長さに値をトランケートし、Adaptive Server の以前のバージョンとの下位互換性を維持します。

sp_transactions を使用したアクティブ・トランザクションの表示

sp_transactions プロシージャは、systransactions と syscoordinations からの情報を変換して、アクティブ・トランザクションについての情報を提供します。キーワードなしで sp_transactions を使用すると、すべてのアクティブ・トランザクションの情報を表示します。

```
                        sp_transactions
xactkey                      type        coordinator starttime
 state             connection dbid   spid   loid
 failover                   srvname                        namelen
xactname
 ------------------------------ ----------- -----------
 --------------------
 ----------------- ---------- ------ ------ -----------
 -------------------------- ------------------------------ -------
 ------------------------------------------------------------------
 0x00000b1700040000dd6821390001 Local       None        Jun 1 1999 3:47PM
 Begun             Attached        1   1      2
 Resident Tx               NULL                                 17
$user_transaction
 0x00000b1700040000dd6821390001 Remote      ASTC        Jun 1 1999 3:47PM
 Begun             NA              0   8      0
 Resident Tx               caserv2                          108

 00000b1700040000dd6821390001-aa01f04ebb9a-00000b1700040000dd6821390001-aa01f04ebb9a-
caserv1-caserv1-0002
```

ローカル・トランザクション、リモート・トランザクション、外部トランザクションの識別

“type” カラムは、トランザクションがローカル・トランザクション、リモート・トランザクション、外部トランザクションのいずれであるかを示します。ローカル・トランザクションは、ローカル・サーバ (sp_transactions を実行するサーバ) で実行します。トランザクションは現在のサーバで実行されるため、ローカル・トランザクション用の “srvname” カラムの値は null です。

リモート・トランザクションの場合、sp_transactions は “srvname” カラムにトランザクションを実行するサーバの名前をリストします。前述の sp_transactions 出力は、caserv2 という名前のサーバで実行されているリモート・トランザクションを示しています。

外部トランザクションとは、CICS、Encina、別の Adaptive Server の “ASTC HANDLER” プロセスなど、外部トランザクション・コーディネータによってコーディネートされるトランザクションのことを示します。外部トランザクションも、“srvname” カラムの値は null です。

トランザクション・コーディネータの識別

“coordinator” カラムは、トランザクションを管理するのに使用されるメソッドまたはプロトコルを示します。前述の出力では、ローカル・トランザクション $user_transaction には外部コーディネータはありません。caserv2 でのリモート・トランザクションのコーディネータの値は “ASTC” です。これは、トランザクションはネイティブ Adaptive Server コーディネーション・サービスを使用してコーディネートされていることを示しています。このサービスについては、「第 3 章 Adaptive Server トランザクション・コーディネーション・サービスの使用」で説明します。

使用可能なコーディネータ値の全リストと説明については、『ASE リファレンス・マニュアル』の「sp_transactions」を参照してください。

トランザクションの制御スレッドの表示

spid カラムは、トランザクションに付加されたプロセスのプロセス ID (トランザクションが制御スレッドから分離されている場合は 0) を示します。ローカル・トランザクションの場合、spid 値はローカル・サーバで実行されているプロセス ID を示します。リモート・トランザクションの場合、spid 値は指定したリモート・サーバで実行されているタスクのプロセス ID を示します。前述の出力では、spid 値として、リモート・サーバ caserv2 で実行されているプロセス ID 8 を表示しています。

トランザクション・ステータス情報の理解

“state” カラムは、各トランザクションの現在のステータス情報を表示します。ローカル・トランザクションまたは外部トランザクションでは、状況によってコマンドの実行、アボート、コミットなどが行われている可能性があります。さらに、分散トランザクションは準備ステータスである場合や、または自発的完了や自発的ロールバックを行っていることもあります。

“connection” カラムは、トランザクションの接続のステータス情報を表示します。この情報を使用して、トランザクションが現在プロセスに付加されているか分離されているかを判断します。X/Open XA 環境のトランザクションは、トランザクション・マネージャからの要求に対応して、それらの開始プロセスから分離されることがあります。

使用可能なコーディネータ値の全リストと説明については、『ASE リファレンス・マニュアル』の「sp_transactions」を参照してください。

sp_transactions 出力の特定ステータスへの限定

sp_transactions と state キーワードを使用して、指定したトランザクションのステータスだけに出力を制限できます。次に例を示します。

```
sp_transactions "state", "Prepared"
```

準備済み分散トランザクションの情報だけを表示します。

トランザクション・フェールオーバ情報

“failover” カラムは、可用性の高い環境で動作しているサーバの特別な情報を表示します。可用性の高い環境では、オリジナルのサーバで重大な障害が発生した場合、準備済みトランザクションがセカンダリ・コンパニオン・サーバに転換されることがあります。“failover” カラムに表示される、トランザクションを実行している方法や場所を示すフェールオーバ・ステータスには、次の 3 種類があります。

- “Resident Tx” は、通常の動作状態で、Adaptive Server 高可用性機能を利用しないシステムで表示されます。“Resident Tx” は、トランザクションがプライマリ Adaptive Server で起動されて実行中であることを示します。
- “Failed-over Tx” は、セカンダリ・コンパニオン・サーバのフェールオーバがあった後で表示されます。“Failed-over Tx” は、トランザクションが最初はプライマリ・サーバで開始され、準備ステータスに達したが、プライマリ・サーバでのシステム障害などのために、自動的にセカンダリ・コンパニオン・サーバにマイグレートされたことを意味します。準備済みトランザクションのマイグレーションは、外部コーディネーティング・サービスに対して透過的に発生します。

- “Tx by Failover-Conn” は、セカンダリ・コンパニオン・サーバにフェールオーバされた後に表示されます。“Tx by Failover-Conn” は、アプリケーションまたはクライアントがプライマリ・サーバでトランザクションを開始したが、プライマリ・サーバが接続フェールオーバによって使用できないことを示します。この状況が発生すると、トランザクションは自動的にセカンダリ・コンパニオン・サーバで開始され、トランザクションは “Tx by Failover-Conn” としてマーク付けされます。

Adaptive Server のフェールオーバ機能の詳細については、「トランザクション・フェールオーバ情報」(26 ページ) を参照してください。

sp_transactions によるコミット・ノードと *gtrid* の指定

sp_transactions に xid キーワードを使用すると、「sp_transactions を使用したアクティブ・トランザクションの表示」(24 ページ) に説明されている出力に加えて、コミット・ノード、親ノード、特定のトランザクションの gtrid が表示されます。この sp_transactions のフォームには、特定のトランザクション名を指定する必要があります。次に例を示します。

```
sp_transactions "xid", "00000b1700040000dd6821390001-aa01f04ebb9a-
00000b1700040000dd6821390001-aa01f04ebb9a-caserv1-caserv1-0002"
xactkey                          type        coordinator starttime
 state             connection dbid   spid   loid
 failover                   srvname                         namelen
xactname
 commit_node parent_node
gtrid
 ------------------------------ ----------- -----------
 --------------------
 ----------------- ---------- ------ ------ -----------
 -------------------------- ------------------------------ -------
 ------------------------------------------------------------------
 -------------
 -------------
 -------------
 0x00000b2500080000dd6821960001 External    ASTC        Jun 1 1999 3:47PM
 Begun             Attached        1      8         139
 Resident Tx                NULL                                108

 00000b1700040000dd6821390001-aa01f04ebb9a-00000b1700040000dd6821390001-aa01f04ebb9a-
caserv1-caserv1-0002

 caserv1 caserv1
 00000b1700040000dd6821390001-aa01f04ebb9a
```

コミット・ノードと親ノード

Adaptive Server がコーディネートする分散トランザクションの場合、“commit node” カラムは分散トランザクションの最上部の分岐を実行するサーバの名前をリストします。このトランザクションは、トランザクションのすべての分岐のコミットまたはロールバック・ステータスを決定します。詳細については、「階層的なトランザクション・コーディネーション」(14 ページ) を参照してください。

“parent node” カラムは、トランザクションを開始したサーバの名前をリストします。上記の sp_transactions 出力では、“commit node” カラムと “parent node” カラムには同じサーバ caserv1 がリストされています。これは、分散トランザクションが caserv1 で発生し、caserv1 が現在のサーバにトランザクションの分岐を送信したことを示します。

グローバル・トランザクション ID

“gtrid” カラムは、Adaptive Server によってコーディネートされた分散トランザクションのグローバル・トランザクション ID を表示します。同じ分散トランザクションの一部であるトランザクション分岐は、同じ gtrid を共有します。特定の gtrid に sp_transactions gtrid キーワードを使用して、現在のサーバで実行されている他のトランザクション分岐のステータスを調べられます。これは、システム管理者が、分散トランザクションの特定の分岐に自発的コミットまたはロールバックを行うべきかどうかを判断するのに役立ちます。sp_transactions と gtrid キーワードの使用例については、「Adaptive Server トランザクションのコミット・ステータスの確認」(34 ページ) を参照してください。

注意 X/Open XA に準拠するトランザクション・マネージャ、MSDTC、または SYB2PC によって調整されるトランザクションの場合、gtrid カラムには、外部コーディネータによって提供される完全なトランザクション名が示されます。

外部トランザクションを実行する手順

すべてのバージョンにおいて、Adaptive Server は外部トランザクションを実行するために次の手順を行います。

1 TM が begin transaction を開始します。

2 TM が attach transaction を開始します。

注意 TM は手順 1 と 2 を同時に実行する場合があります。

3 アプリケーションが DML コマンドを実行します。

4　TM が detach transaction を開始します。

5　必要に応じて、手順 2 ～ 4 を繰り返します。

6　トランザクションがロールバックされない場合、TM は prepare transaction を開始します。

7　TM が commit transaction または rollback transaction を開始します。

手順 3 を実行すると、分散トランザクションがロールバックされます。

すべてのコマンドを発行する前にグローバル変数を確認することは面倒であるため、多くのユーザ・アプリケーションはこの確認をまったく行いません。バージョン 15.0.3 以前では、分散トランザクションがロールバックされた場合に、Adaptive Server はユーザ・アプリケーションが引き続き SQL コマンドを発行することを許可していました。これらのコマンドは、独立したトランザクションとして分散トランザクションの外側で実行されました。ロールバック・トランザクションに含める必要のある SQL コマンドが独立してコミットされることで、トランザクションにデータの矛盾が発生しました。

15.0.3 以降では、Adaptive Server は、分散トランザクション内で実行されるべき SQL コマンドが、分散トランザクションの外側で実行されないように自動的に制御します。つまり、すべてのコマンドを発行する前に、ユーザ・アプリケーションでグローバル変数を確認する必要がなくなりました。トランザクションが暗黙的にアボートされると、「外部トランザクションがロールバックされたため、コマンドを実行できません」というエラー・メッセージ (3953) が表示されます。detach transaction コマンドが発行されると、このメッセージの表示が消えます。

3953 エラー・メッセージの表示を消し、Adaptive Server で以前の動作をリストアする (つまり DTM トランザクションがアクティブでない場合であっても SQL コマンドを実行する) には、トレース・フラグ -T3995 を使用して Adaptive Server を起動します。

分散トランザクションにおけるクラッシュのリカバリ手順

クラッシュのリカバリ中、Adaptive Server は準備済みステータスで発見された分散トランザクションを解決する必要があります。準備済みトランザクションの解決に使用する方法は、分散トランザクションの管理に使用されたコーディネーション方法およびコーディネーション・プロトコルによって異なります。

注意 次に示すクラッシュのリカバリ手順は、load database または load transaction コマンドを使用した通常のデータベース・リカバリ中には実行されません。load database または load transaction が準備済みまたは疑わしいトランザクションを適用した場合、Adaptive Server は関連するデータベースをオンラインにする前に、それらのトランザクションをアボートします。

MSDTC によってコーディネートされたトランザクション

MSDTC を使用してコーディネートされている準備済みトランザクションは、マスタ・トランザクションのコミット・ステータスに応じてロールフォワードまたはロールバックされます。リカバリ中、Adaptive Server は MSDTC とのコンタクトを開始し、マスタ・トランザクションのコミット・ステータスを判断して、それに従って準備済みトランザクションをコミットまたはロールバックします。MSDTC とコンタクトできない場合、リカバリ処理はコンタクトができるまで待機します。それ以上のリカバリは、Adaptive Server が MSDTC とのコンタクトが取れるまで実行されません。

Adaptive Server または X/Open XA によってコーディネートされたトランザクション

クラッシュのリカバリ中、Adaptive Server は、Adaptive Server トランザクション・コーディネーション・サービスまたは X/Open XA プロトコルを使用してコーディネートされた準備済みトランザクションを検出します。これらのトランザクションを検出すると、コーディネートしている Adaptive Server または外部トランザクション・コーディネータがコンタクトを開始し、準備済みトランザクションがコミットまたはロールバックを実行すべきか判定するまで、ローカル・サーバは待機しなければなりません。

リカバリ処理の速度を上げるため、Adaptive Server はこれらの各トランザクションを障害の前の状態に戻します。トランザクション・マネージャはオリジナルのトランザクション ID を使って新しいトランザクションを作成し、ロック・マネージャはロックを適用してオリジナル・トランザクションが修正したデータを保護します。リストアされたトランザクションは準備済みステータスのままですが、関連するスレッドを持たず、分離されます。

トランザクション・コーディネータが Adaptive Server にコンタクトすると、トランザクション・マネージャはトランザクションをコミットまたはロールバックできます。

このリカバリ・メカニズムを使用すると、サーバはコーディネートする Adaptive Server または外部トランザクション・マネージャがまだ準備済みトランザクションを解決していなくても、データベースをオンラインにすることができます。準備済みトランザクションはリカバリの前に実行したロックを保持するため、他のクライアントやトランザクションはローカル・データでの作業を再開できます。準備済みトランザクション自体は、そのコーディネータによってコンタクトが行われると、コミットまたはロールバックを準備します。

制御している Adaptive Server または外部トランザクション・マネージャがトランザクションを完了できない場合、システム管理者はそのロックとトランザクション・リソースを解放するために、トランザクションの自発的完了を実行できます。詳細については、「トランザクションの自発的完了」(31 ページ) を参照してください。

SYB2PC によってコーディネートされたトランザクション

SYB2PC プロトコルを使用してコーディネートされた準備済みトランザクションは、マスタ・トランザクションのコミット・ステータスに応じてロールフォワードまたはロールバックされます。リカバリ中、Adaptive Server はコミット・サービスとのコンタクトを開始し、マスタ・トランザクションのコミット・ステータスを判断して、それに従って準備済みトランザクションをコミットまたはロールバックします。コミット・サービスにコンタクトできない場合、Adaptive Server はデータベースをオンラインにしません。しかし、Adaptive Server はシステム内の他のデータベースのリカバリ処理を進めます。

このリカバリ方法は、Adaptive Server の以前のバージョンでの SYB2PC トランザクションで利用されており、Adaptive Server バージョン 12.5 以降でも変更はありません。

トランザクションの自発的完了

Adaptive Server の dbcc complete_xact コマンドは、トランザクションの自発的完了を行います。dbcc complete_xact はその作業をコミットまたはロールバックすることによってトランザクションを解決し、トランザクションが使用していたリソースを解放します。

dbcc complete_xact は、システム管理者だけが準備済みトランザクションを正しく解決できる場合、またはシステム管理者がトランザクション・コーディネータを待たずにトランザクションを解決しなければならない場合に使用するように提供されています。

たとえば、図 3-1 (14 ページ) に示すように、すべてのリモート Adaptive Server がトランザクションの準備を完了していても、ASE1 へのネットワーク接続が永久に失われた場合に、自発的完了を検討できます。リモート Adaptive Server は、ASE1 からのコーディネーション・サービスによってコンタクトされるまで、それらのトランザクションを準備ステータスのままにします。この場合、ASE2、ASE3、ASE4 のシステム管理者は、準備済みトランザクションを適切に解決できます。ASE3 における準備済みトランザクションの自発的完了は、トランザクションとロック・リソースを解放し、後にトランザクション・コーディネータが使用するために systransactions にコミット・ステータスを記録します。ASE2 におけるトランザクションの自発的完了も、ASE4 に送信されたトランザクションを完了します。

準備済みトランザクションの完了

警告！ 準備済みトランザクションの自発的完了は、分散トランザクション全体において一貫性のない結果となる場合があります。システム管理者がトランザクションを自発的にコミットまたはロールバックすることに決定すると、Adaptive Server またはトランザクション・プロトコルを調整した場合の決定内容と矛盾してしまう可能性があります。

トランザクションの自発的完了を実行する前に、システム管理者はコーディネーティング Adaptive Server またはトランザクション・プロトコルが分散トランザクションのコミットまたはロールバックを決断するかどうかを調べる必要があります (詳細については、「Adaptive Server トランザクションのコミット・ステータスの確認」(34 ページ) 参照)。

dbcc complete_xact を使用して、システム管理者は Adaptive Server に分散トランザクションの分岐をコミットまたはロールバックするよう強制します。準備済みトランザクションの自発的完了を実行した後、Adaptive Server は master.dbo.systransactions にトランザクションのコミット・ステータスを記録するので、トランザクションのコーディネータ (Adaptive Server、MSDTC、または X/Open XA トランザクション・マネージャ) はトランザクションがコミットまたはロールバックされたかどうかを確認できます。

Adaptive Server はコマンドを送信して、トランザクション分岐をコーディネートした対象のサーバへのトランザクションを自発的にコミットするか、またはアボートします。たとえば、図 3-1 (14 ページ) に示すように、ASE2 でトランザクションに自発的コミットを実行すると、ASE2 は ASE4 にコマンドを送信し、ASE4 でのトランザクションもコミットします。

dbcc complete_xact には、アクティブ・トランザクション名と希望のトランザクション結果を指定する必要があります。たとえば、次のコマンドはトランザクションの自発的コミットを実行します。

```
dbcc complete_xact "00000b1700040000dd6821390001-
aa01f04ebb9a-00000b1700040000dd6821390001-aa01f04ebb9a-
caserv1-caserv1-0002", "commit"
```

トランザクションの自発的完了の削除

システム管理者が準備済みトランザクションに自発的完了を実行すると、Adaptive Server は master.dbo.systransactions にトランザクションのコミット・ステータスについての情報を保持します。この情報を管理することによって、外部トランザクション・コーディネータは自発的に完了されたトランザクションの存在を検出できます。

外部コーディネータが別の Adaptive Server の場合、サーバはコミット・ステータスを調べ、自発的完了が分散トランザクションのコミット・ステータスと矛盾しているときは警告メッセージをログします。コミット・ステータスを確認したら、コーディネーティング Adaptive Server は systransactions からコミット・ステータス情報をクリアします。

外部コーディネータが X/Open XA 準拠のトランザクション・マネージャの場合、トランザクション・マネージャは自発的完了が分散トランザクションと矛盾していても警告メッセージをログしません。しかし、X/Open XA 準拠のトランザクション・マネージャは、systransactions からコミット・ステータス情報をクリアします。

コミット・ステータスの手動クリア

dbcc forget_xact は、systransactions から自発的に完了されたトランザクションのコミット・ステータスをパージします。これは、システム管理者がコーディネーティング・サービスにトランザクションが自発的に完了されることを認識させたくない場合、または外部コーディネータが systransactions からの情報をクリアできない場合に使用できます。

dbcc forget_xact の使用方法の詳細については、『リファレンス・マニュアル』の「dbcc」を参照してください。

準備済みでないトランザクションの完了

dbcc complete_xact は、Adaptive Server によってコーディネートされた準備ステータスになっていないトランザクションをロールバックするのに使用することもできます。コーディネーティング・サーバはトランザクションが作業の準備に失敗したことを認識できるため、準備されていないトランザクションの自発的ロールバックは分散トランザクションに対してリスクにはなりません。このような状況では、コーディネーティング Adaptive Server は一貫性を保つために分散トランザクション全体をロールバックできます。

準備されていない Adaptive Server トランザクションに自発的ロールバックを行う場合、Adaptive Server は systransactions に自発的ロールバックを記録しません。代わりに、情報メッセージが画面に出力され、サーバのエラー・ログに記録されます。

Adaptive Server トランザクションのコミット・ステータスの確認

コミットまたはロールバックしたい分散トランザクション分岐が Adaptive Server によってコーディネートされているときは、sp_transactions を使用して分散トランザクションのコミット・ステータスを確認できます。これを行うには、次の手順に従います。

注意 これらの手順は、X/Open XA プロトコル、MSDTC、または SYB2PC によってコーディネートされている分散トランザクションには使用できません。

1 完了するトランザクション分岐を実行しているサーバでは、sp_transactions と xid キーワードを使用して、トランザクションの情報を表示します。コミット・ノードとトランザクションの gtrid を記録します。次に例を示します。

```
sp_transactions "xid", "00000b1700040000dd6821390001-
aa01f04ebb9a-00000b1700040000dd6821390001-aa01f04ebb9a-
caserv1-caserv1-0002"
xactkey                         type        coordinator starttime
 state             connection dbid   spid   loid
 failover                   srvname                          namelen
 xactname
 commit_node parent_node
 gtrid
 ------------------------------ ----------- -----------
 --------------------
 ---------------- ---------- ------ ------ -----------
 ------------------------- ------------------------------ -------
 ------------------------------------------------------------------
 -------------
 -------------
 -------------
 0x00000b2500080000dd6821960001 External    ASTC        Jun 1 1999 3:47PM
 Begun             Attached        1      8         139
 Resident Tx               NULL                                 108

 00000b1700040000dd6821390001-aa01f04ebb9a-00000b1700040000dd6821390001-
aa01f04ebb9a-caserv1-caserv1-0002

 caserv1 sfserv
 00000b1700040000dd6821390001-aa01f04ebb9a
```

この例では、分散トランザクションのコミット・ノードは “caserv1” で、gtrid は “00000b1700040000dd6821390001-aa01f04ebb9a” です。

2 コミット・ノードによって示されたサーバにログオンします。次に例を示します。

```
isql -Usa -Psa_password -Scaserv1
```

3　sp_transactions と gtrid キーワードを使用して、手順 1 で取得した gtrid を持つ分散トランザクションのコミット・ステータスを確認します。

```
sp_transactions "gtrid", "00000b1700040000dd6821390001-
aa01f04ebb9a"
xactkey type coordinator starttime
state connection dbid   spid   loid
failover srvname namelen
xactname
commit_node
parent_node
------------------------------ ----------- -----------
--------------------
---------------- ---------- ------ ------ -----------
------------------------ ------------------------------ -------
 ------------------------------------------------------------------
 -------------
 -------------
 0x00000b1700040000dd6821390001 Local       None        Jun 1 1999 3:47PM
 Committed Attached        1      1           2
 Resident Tx                NULL                                17
$user_transaction

 caserv1
 caserv1
```

この例では、“state” カラムによって示されているように、指定した gtrid を持つローカル・トランザクションはコミットされています。システム管理者は、手順 1 で確認した準備済みトランザクションに自発的コミットを実行する必要があります。

4　システム管理者権限を持つアカウントを使用して、完了させるトランザクション分岐を実行するサーバにログオンします。

```
isql -Usa -Psa_password -Ssfserv
```

5　dbcc complete_xact を使用して、トランザクションをコミットします。この例では、システム管理者は commit キーワードを使用して、分散トランザクションとの一貫性を保つ必要があります。

```
dbcc complete_xact "00000b1700040000dd6821390001-
aa01f04ebb9a-00000b1700040000dd6821390001-aa01f04ebb9a-
caserv1-caserv1-0002", "commit"
```

プログラミングと設定の考慮事項

この項では、トラブルシューティングを行うときに考慮が必要な設定オプションについて説明します。

分散トランザクションでの DDL の動作

X/Open XA プロトコルを使用する外部トランザクション・マネージャ、または別の Adaptive Server の Adaptive Server トランザクション・コーディネーション・サービスによってトランザクションがコーディネートされる場合、DDL コマンドはトランザクション内で使用できません。この動作は、データベース・オプション ddl in tran が有効になっている場合でも適用されます。

外部トランザクションでの Adaptive Server の暗黙のロールバック

外部トランザクションでエラー (デッドロック、更新トリガのアボートなど) が発生した場合、Adaptive Server によって外部トランザクションがアボートされることがあります。

Adaptive Server は障害に関するエラー・メッセージを送信しますが、アプリケーションは常にメッセージを確認するわけではありません。特に、DBA によって追加されたトリガのような簡単な挿入は、アプリケーションで認識されない場合があります。XA トランザクションが終了したことが、エラー・メッセージで確認できない場合もあります。

Adaptive Server が外部トランザクションをアボートして SQLException をスローした場合は、`select @@trancount` を発行することができます。*@@trancount* の値がゼロの場合は、DTM トランザクションがアボートされたことを意味します。

アプリケーションはトランザクション・マネージャ (通常はアプリケーション・サーバ) を呼び出して、トランザクションがアボートされたことを知らせる必要があります。エラー・メッセージを無視すると、後続の更新が DTM トランザクションのコンテキスト外 (たとえばローカル・トランザクション) で実行される可能性があります。エラー・メッセージのログを取り *@@transtate* または *@@trancount* を調べて、更新が行われたことを確認することができます。

Adaptive Server に外部トランザクションをロールバックさせるトリガを次に示します。失敗する可能性のあるトリガは insert 文に含まれています。Adaptive Server が insert を発行できない場合は、更新で ut.commit 関数が実行されます。

次の例 (疑似コード) は、JTA/XA UserTransaction を実行していることを前提としています。

```
try {

insert into table values (xx....)
update table
ut.commit();

} catch (SQLException sqe) {

if this is a known error then process
else
select @@trancount into count
if count == 0
then ut.rollback() }
```

rollback 関数を含めない場合、ほかの更新は JTA/XA トランザクション外で行われます。

索引

記号

A

B

C

D

E

F

I

L

M

N

O

R

S

T

V

X

い

お

か

き

く

こ

し

す

せ

た

て

と

な

の

は

ふ

ま

も

ら

り

ろ