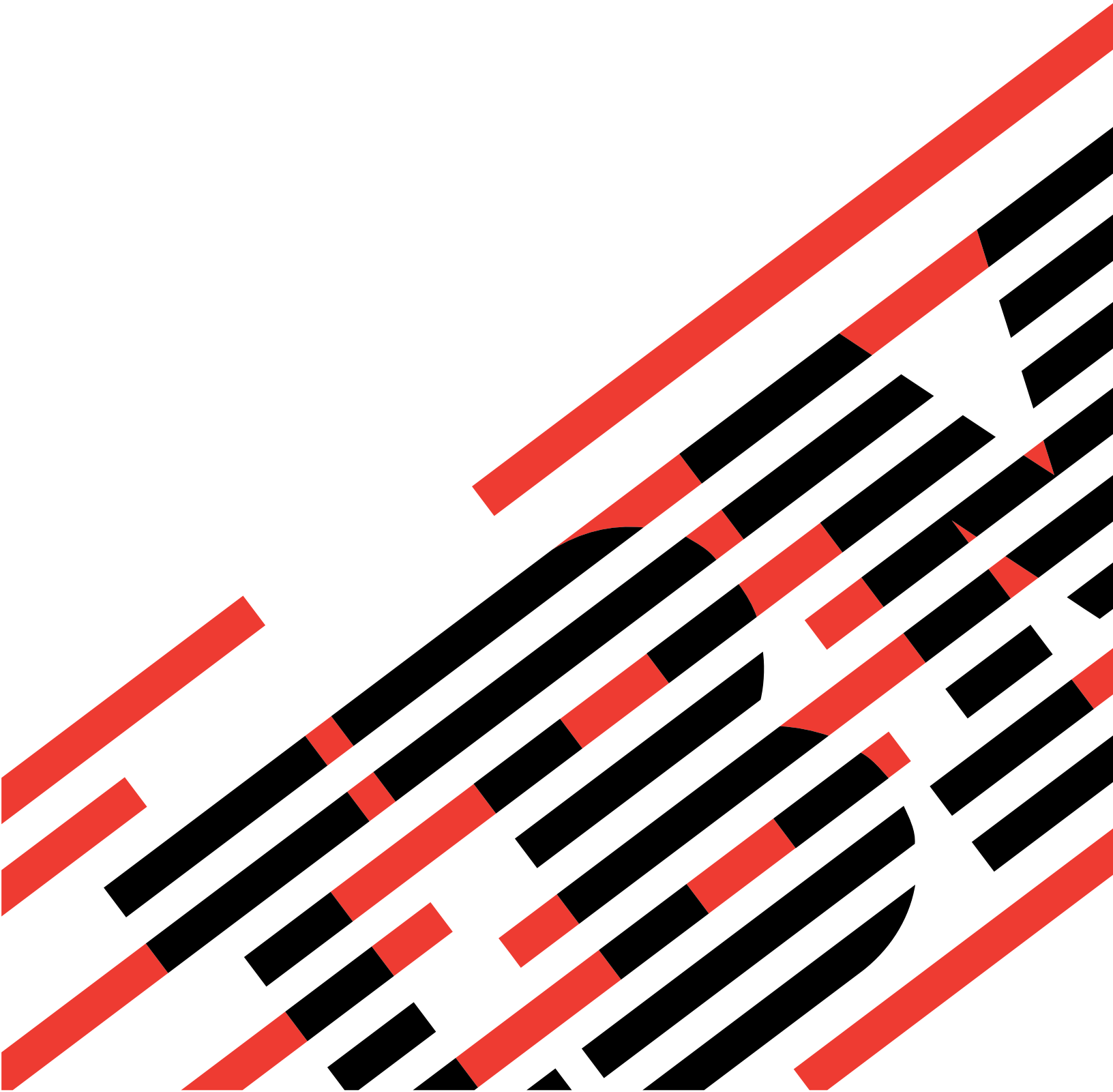
IBM

eserver

iSeries

分散データベース・プログラミング

バージョン 5 リリース 3



IBM

@server

iSeries

分散データベース・プログラミング

バージョン 5 リリース 3

お願い

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

本書は、IBM OS/400 (プロダクト番号 5722-SS1) のバージョン 5、リリース 3、モディフィケーション 0 に適用されます。また改訂版で断りがない限り、それ以降のすべてのリリースおよびモディフィケーションに適用されます。このバージョンは、すべての RISC モデルで稼動するとは限りません。また、CISC モデルでは稼動しません。

本書は、SC41-5702-03 の改訂版です。

本マニュアルに関するご意見やご感想は、次の URL からお送りください。今後の参考にさせていただきます。

http://www.ibm.com/jp/manuals/main/mail.html

なお、日本 IBM 発行のマニュアルはインターネット経由でもご購入いただけます。詳しくは

http://www.ibm.com/jp/manuals/ の「ご注文について」をご覧ください。

(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： iSeries
Distributed Database Programming
Version 5 Release 3

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 2005.8

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

分散データベース・プログラミングについて

分散データベース・プログラミング では、 OS/400 (OS/400) ライセンス・プログラムの分散リレーショナル・データベース管理機能について説明します。分散リレーショナル・データベース管理により、アプリケーションは、そのアプリケーションの外部にあり、別のコンピューター・ネットワークに存在するデータへアクセスできるようになります。

本書の詳細については、以下のトピックを参照してください。

- 本書の対象読者
- 分散データベース・プログラミング V5R3 の新機能

次に、ご使用に際して、 iSeries 分散リレーショナル・データベースでの処理、サポート、プログラミング、および管理に関する情報について、分散リレーショナル・データベースと iSeries server を参照してください。

本書の対象読者

この資料は主に、1 つ以上の iSeries server で、分散リレーショナル・データベースの開発、管理、およびサポートを担当する、アプリケーション・プログラマーを対象にしています。 iSeries データベースに詳しくないアプリケーション・プログラマーであっても、 OS/400 オペレーティング・システムによって提供される、データベース・サポートを全体的に理解することができます。 アプリケーション・プログラマーは、この情報を使って、分散リレーショナル・データベース・アプリケーションを実行するときのサーバー・コンテキストを確認できます。

本書のご使用にあたっては、すでに、一般的なプログラミングの概念および用語の知識があり、 iSeries server および OS/400 オペレーティング・システムについて総合的に理解している必要があります。

分散データベース・プログラミング V5R3 の新機能

このリリースでは、以下の更新が行われています。

- サーバー認証項目を表示する方法
- Kerberos DRDA® サービス名の定義
- RDB ディレクトリー項目追加の例
- STRTRC の使用
- TRCTCPAPP トレース・フォーマット設定
- QRWOPTIONS データ域の使用法
- 新しいレベルの DRDA プロトコル
- CONNECT ステートメントの新しい診断情報
- 接続を試行すると SQL5048N メッセージが戻される理由
- バインド・オプションの例
- Query の終了
- ユーザー ID とパスワードの長さ
- ストアード・プロシージャー、ユーザー定義関数 (UDF)、およびコミットメント制御
- 平文として流される特定のパスワードに関する注意

印刷可能な PDF

本書の PDF 版を表示またはダウンロードするには、分散データベース・プログラミング (約 2535 KB) を選択してください。

PDF ファイルの保管

表示または印刷のためにワークステーションに PDF ファイルを保管するには、次のようにします。

1. ブラウザーで PDF ファイルを右クリックします (リンク上で右クリックします)。
2. ローカルで PDF を保管するオプションをクリックします。
3. PDF ファイルを保管するディレクトリーに進みます。
4. 「**保存**」をクリックします。

Adobe Reader のダウンロード

これらの PDF を表示または印刷するには、システムに Adobe Reader をインストールしておく必要があります。Adobe Web サイト (www.adobe.com/products/acrobat/readstep.html) から無料でダウンロードできます。

コードの特記事項情報

本書には、プログラミングの例が含まれています。

IBM® は、お客様に、すべてのプログラム・コードのサンプルを使用することができる非独占的な著作使用権を許諾します。お客様は、このサンプル・コードから、お客様独自の特別のニーズに合わせた類似のプログラムを作成することができます。

すべてのサンプル・コードは、例として示す目的でのみ、IBM により提供されます。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。

ここに含まれるすべてのプログラムは、現存するままの状態で提供され、いかなる保証も適用されません。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の保証の適用も一切ありません。

第 1 章 分散リレーショナル・データベースと iSeries server

iSeries server システムでの分散リレーショナル・データベース・サポートは、 IBM* 分散リレーショナル・データベース・アーキテクチャー (Distributed Relational Database Architecture* (DRDA*)) の実装と、アプリケーション・リクエスター・ドライバー (ARD) プログラムの使用による SQL クライアントの統合で構成されます。 Operating System/400® (OS/400) と、 DB2 UDB for iSeries Query Manager and SQL Development Kit が結合することにより、このサポートを提供しています。

この章では、分散リレーショナル・データベースと、 iSeries server 上での使用方法について説明します。ここでは、以下のトピックで説明されている、分散リレーショナル・データベースのいくつかの一般的な概念について定義します。

- 分散リレーショナル・データベース処理
- 分散リレーショナル・データベース・アーキテクチャー のサポート
- DRDA および CDRA サポート
- アプリケーション・リクエスター・ドライバー・プログラム
- iSeries server における分散リレーショナル・データベース
- iSeries 分散リレーショナル・データベースの管理

これらのトピックの他に、 例: Spiffy 社の分散リレーショナル・データベースについても説明されています。この架空の企業は、分散リレーショナル・データベース・アプリケーション・プログラムで、 iSeries server を使用します。本書で使われている例はすべて、この Spiffy 社の例を背景としています。

分散リレーショナル・データベース処理

リレーショナル・データベースとは、コンピューター内の 1 つ以上のテーブルに格納されたデータのセットです。**テーブル**とは、データを 2 次元に配列したもので、 表 1 に示されているように、横の行と縦の列から成っています。各**行**には、一連の値が含まれており、テーブルの列ごとに 1 つの値が入ります。**列**には名前が付けられており、特定のデータ・タイプ (文字、10 進数、整数など) が入ります。

表 1. 代表的なテーブル

品目	品名	仕入れ先	数量
78476	野球ボール	ACME	650
78477	ラグビー・ボール	Imperial	228
78478	バスケット・ボール	ACME	105
78479	サッカー・ボール	ACME	307

サーバーでは、いろいろな方法でテーブルを定義しアクセスできます。 サーバー上のテーブルに記述およびアクセスする 1 つの方法は、**構造化照会言語 (SQL)** のような言語を使うことです。 SQL は標準的な IBM データベース言語であり、異なるサーバーの間で分散データ処理を行えるようにするために必要な整合性を備えています。

サーバーのテーブルを記述しアクセスする別の方法は、データ記述仕様 (DDS) を使用して物理および論理ファイルを記述し、ファイル・インターフェースを使用してテーブルへアクセスすることです (たとえば、read や write の高水準言語のステートメントなど)。

SQL では、iSeries server で使われる用語とは異なる用語を使用します。ほとんどの SQL オブジェクトの場合、 iSeries server 側にも、対応するサーバー・オブジェクトが存在します。 表 2 では、 SQL リレーショナル・データベースの用語と iSeries server の用語の関連が示されています。

表 2. SQL 用語とシステム用語の関係

SQL 用語	システム用語
リレーショナル・データベース。テーブルのセットと見なすことができ、データのリレーショナル・モデルに従って操作できるデータベース。ユーザーが iSeries™ server からアクセスできるリレーショナル・データベースは、システム用語の欄にリストされているように、 3 つのタイプがあります。詳細については、 iSeries Information Center の 『リレーショナル・データベース (Relational Database)』をご覧ください。	**システム・リレーショナル・データベースまたはシステム・データベース**。 iSeries サーバーに接続されたディスク上に存在し、独立補助記憶域プール上には保管されないすべてのデータベース・オブジェクト。 **ユーザー・リレーショナル・データベースまたはユーザー・データベース**。独立した補助記憶域プールには保管されていないデータベース・オブジェクトとともに、独立した単一の補助記憶域プールに存在するすべてのデータベース・オブジェクト。**注:** V5R2 では、 iSeries サーバーは、独立補助記憶域プールがサーバー上で構成されている場合、複数のリレーショナル・データベースに対するホストにすることができる。システム・リレーショナル・データベースは必ず 1 つ存在し、ユーザー・リレーショナル・データベースは 1 つ以上存在することができます。各ユーザー・データベースには、システム・データベース内のすべてのオブジェクトが入っています。**注:** ただし、システム・データベースは、コミットメント制御の観点では別のデータベースとして扱われ、 SQL の観点でもユーザー・データベースに組み込まれていると見なされることを承知しておく必要がある。詳細については、iSeries Information Center の『トランザクションとコミットメント制御』のトピックを参照してください。 **リモート・リレーショナル・データベースまたはリモート・データベース**。 iSeries または別のサーバー上に存在する、リモート・アクセス可能なデータベース。
スキーマ。ライブラリー、ジャーナル、ジャーナル・レシーバー、SQL カタログ、およびオプションのデータ・ディクショナリーで構成されます。スキーマを使うと、関連するオブジェクトがグループ化され、名前でオブジェクトを見つけられるようになります。**注:** スキーマは一般にコレクションとも呼ばれます。	**ライブラリー**。これを使うと、関連するオブジェクトがグループ化され、名前でオブジェクトを見つけられるようになります。
テーブル。列と行の集まり。	**物理ファイル**。レコードの集まり。
行。一連の列を含む、表の横方向の部分。	**レコード**。フィールドの集まり。
列。 1 つのデータ・タイプの、表の縦方向の部分。	**フィールド**。 1 つのデータ・タイプの 1 バイト以上の関連情報。
ビュー (視点)。 1 つ以上の表の列と行のサブセット。	**論理ファイル**。 32 個までの物理ファイルの、フィールドまたはレコード (あるいはその両方) のサブセット。
索引。表の列にあるデータを集めて、昇順あるいは降順に論理的に配列したもの。	論理ファイルの 1 つのタイプ。
パッケージ。アプリケーション・サーバーで使う SQL ステートメントの制御構造を含むオブジェクト。	**SQL パッケージ**。 SQL 用語での意味と同じ。

表 2. SQL 用語とシステム用語の関係 (続き)

SQL 用語	システム用語
カタログ。テーブル、パッケージ、ビュー、索引、および制約についての情報を含む、一連のテーブルとビュー。 QSYS2 のカタログ・ビューには、iSeries server における、すべてのテーブル、パッケージ、ビュー、索引、および制約についての情報が含まれています。さらに、SQL スキーマには、スキーマ内のテーブル、パッケージ、ビュー、索引、および制約についての情報だけを含む一連のビューがある。	類似のオブジェクトはありません。ただし、ファイル記述の表示 (DSPFD) および ファイル・フィールド記述の表示 (DSPFFD) コマンドを使えば、 SQL カタログの照会で入手できる情報の一部を入手できます。

分散リレーショナル・データベースは、データおよびデータそのものを使うアプリケーション・プログラムが別のシステム上に置かれている場合、またはデータを使用するプログラムが同じサーバー上の複数のデータベースに置かれている場合に存在します。後者の場合、単一のサーバー内の 1 つ以上のデータベースにアクセスするために DRDA プロトコルが使用されるという意味で、データベースは分散しています。そのような環境でのデータベースへの接続は、ローカルか DRDA のいずれかのタイプです。ほとんどの場合、1 度に存在するローカル・データベース接続は 1 つだけです。分散リレーショナル・データベースの簡単な例が 図 1 に示されています。ここでは、アプリケーション・プログラムを 1 つのマシンで実行していますが、そのデータは別のリモート・サーバー上に存在しています。

分散リレーショナル・データベースを使用する場合、アプリケーション・プログラムを実行するシステムを**アプリケーション・リクエスター (AR)** といい、リモート・データが存在するシステムを**アプリケーション・サーバー (AS)** といいます。 'クライアント' は AR と、 'サーバー' は AS と交換可能な単語としてしばしば使用されます。

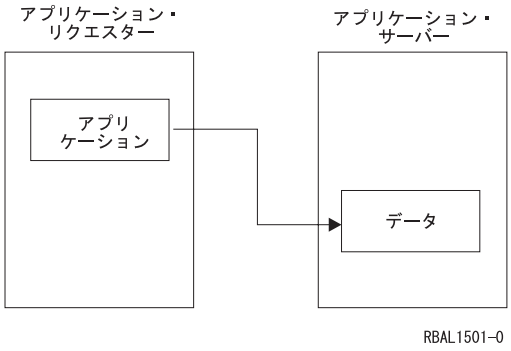


図 1. 分散リレーショナル・データベース

作業単位とは、 4 ページの図 2 に示されているように、完成した作業の一部を構成する、1 つ以上のデータベース要求および関連した処理のことです。簡単な例として、在庫管理アプリケーション・プログラムを使用して、在庫から部品を取り出すことが挙げられます。在庫プログラムでは、店の在庫アカウント・テーブルから項目を仮に削除し、その項目を、同じ場所の部品追加発注テーブルへ追加することができます。 'トランザクション' という用語は、作業単位という概念を伝える別の表現です。

上記の例では、部品が店の在庫アカウント・テーブルから削除されて、追加発注テーブルへ追加されるまで、作業単位は完了しません。要求が完了したら、アプリケーション・プログラムは、作業単位を**コミット**できます。すなわち、作業単位に関連したデータベースの変更はすべて永続的なものになる。

作業単位サポートにより、アプリケーション・プログラムは、作業単位への変更を**ロールバック**することも可能です。作業単位をロールバックすると、最後のコミットあるいはロールバック操作後に加えられた変更は適用されません。このように、アプリケーション・プログラムは、データベースに対する一連の要求を 1 単位として扱います。

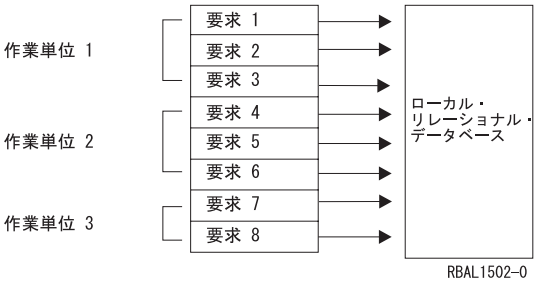


図 2. ローカル・リレーショナル・データベースにおける作業単位

作業単位の詳細については、以下のトピックを参照してください。

- リモート作業単位
- 分散作業単位
- 分散リレーショナル・データベースのその他の用語と概念

リモート作業単位

リモート作業単位 (RUW) は、分散リレーショナル・データベース処理の一形態であり、アプリケーション・プログラムは、 1 作業単位内でリモート・データベース上のデータにアクセスできます。 1 つのリモート作業単位には、複数のリレーショナル・データベース要求を含めることが可能ですが、すべての要求を同じリモート・データベースに対して発行する必要があります。リレーショナル・データベースに対する要求を別のリレーショナル・データベースへ送る前に、すべての要求を (コミットするかロールバックして) 完了しておかなければなりません。このことが、図 3 に示されています。

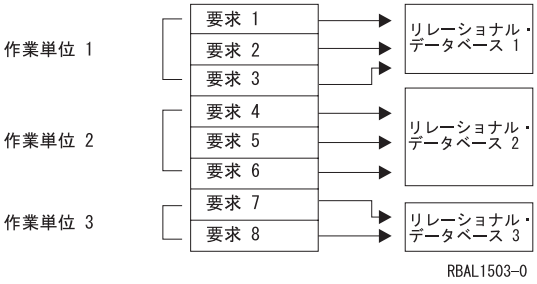


図 3. 分散リレーショナル・データベースでのリモート作業単位

アプリケーション・プログラムは、要求を発行する前に、正しいリレーショナル・データベース・システムに接続する必要があるため、リモート作業単位はアプリケーション指示による分散処理です。しかし、アプリケーション・プログラム側では、正しい接続を確立するために、リモート・データベースの名前を知っているだけで構いません。

リモート作業単位サポートによって、アプリケーション・プログラムは、複数のロケーションでデータを読み取ったり更新することができます。ただし、作業単位内でプログラムがアクセスするすべてのデータを、同じリレーショナル・データベース管理システムで管理する必要があります。たとえば、店の在庫アプリケ

ーション・プログラムは、別のロケーションに存在するテーブルを読み取ったり更新する前に、在庫および売掛管理作業単位をコミットしなければなりません。

リモート作業単位の処理では、コンピューターごとに、関連するリレーショナル・データベース管理システムと関連するアプリケーション・リクエスター・プログラムがあり、分散リレーショナル・データ要求の処理に役立てることができます。これにより、独自のアプリケーション・プログラムを使い、ローカル・リレーショナル・データを要求するときとほとんど同じ方法で、リモート・リレーショナル・データを要求できるようになります。

分散作業単位

分散作業単位 (DUW) を使うと、 図 4 に示されているように、ユーザーまたはアプリケーション・プログラムは、 1 つの作業単位内の複数のロケーションで、データを読んだり更新することが可能になります。 1 つの作業単位内で、1 つのシステムで実行しているアプリケーションは、相手システムでサポートされている SQL を使うことにより、 SQL 要求を複数のリモート・データベース管理システムへ送ることができます。たとえば、店の在庫プログラムでは、あるシステムにある在庫テーブルを更新すると同時に、同じ作業単位内の別のシステムにある売掛管理テーブルを更新することもできます。

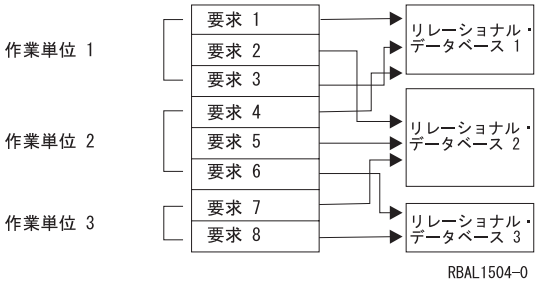


図 4. 分散リレーショナル・データベースにおける分散作業単位

要求のターゲットは、CONNECT TO や SET CONNECTION などの SQL ステートメントを使い、ユーザーまたはアプリケーションによって制御されます。各 SQL ステートメントは、それぞれ 1 つのロケーションのデータを参照しなければなりません。

アプリケーションが作業をコミットする準備ができると、アプリケーションはそのコミットを開始します。コミットの実行は、同期点管理プログラムによって調整されます。

分散作業単位によって、次のことが行えます。

- 1 つの作業単位の中で、複数のデータベース管理システムに対する更新アクセス、または、
- 別のデータベース管理システムに対する読み取りアクセスを行いながら、 1 つの作業単位の中で、1 つ以上のデータベース管理システムに更新アクセス。

1 つの作業単位の中で、アプリケーションが所定のデータベース管理システムを更新できるかどうかは、DRDA のレベル (リモート・リレーショナル・データベースへのアクセスに DRDA を使用する場合) と、接続と更新を行うときの順序に依存しています。

分散リレーショナル・データベースのその他の用語と概念

次の部分では、分散リレーショナル・データベースに関する他の概念を簡単に説明します。 IBM システムでは、いくつかの分散リレーショナル・データベース・サポートは、 DB2® Relational Connect および DataPropagator™ Relational プロダクトによって提供されています。また、 iSeries アプリケーション・プログラムの作成時に、これらの概念のいくつかを使うこともできます。

DB2 UDB for iSeries は、 OS/400® V5R1 から APPC および TCP/IP 通信でのリモート作業単位と分散作業単位の両方をサポートしています。処理の複雑さの程度が分散作業単位を超えた場合が、**分散要求**になります。 このタイプの分散リレーショナル・データベース・アクセスを使うと、ユーザーまたはアプリケーション・プログラムは、1 つの SQL ステートメントを発行して、複数のロケーションにあるデータを読み取ったり更新できます。

分散リレーショナル・データベースのテーブルは、必ずしも相互に異なっていなければならないわけではありません。テーブルによっては、他のテーブルのまったく同じコピーであるか、部分コピーである可能性があります。抽出、スナップショット、および複製は、分散処理を使用したコピーのタイプを示す用語です。

抽出 (Extract) は、ユーザーが要求したテーブルのコピーです。このコピーは、特定のデータベースから抽出され、ユーザー指定のデータベースへロードされます。アンロードとロードのプロセスを周期的に繰り返すことにより、更新したデータを取り出すことができます。抽出は、たまにしか変更されないデータの読み取り専用コピーが、 1 回限り生じるあるいは低頻度の場合に最適です。

スナップショットは、サーバーによって自動的に生成される、テーブルの読み取り専用コピーです。 サーバーは、ソース・テーブルからのこのようなコピーを、ユーザー指定の周期 (日ごと、週ごと、月ごとなど) でリフレッシュします。 スナップショットは、更新された情報を周期的に受け取るための自動的な処理を必要とするようなロケーションで最適です。

データの**複製 (replication)** とは、サーバーがテーブルのコピーを自動的に更新することです。 これは、テーブルのコピーが複数のロケーションに格納されるという点で、スナップショットと似ています。データ複製は、更新されることが少なく、高い信頼性と迅速な検索が求められる状況で、特に効果的です。

テーブルを、ネットワーク内のコンピューター・サーバー間で分けることも可能です。 そのようなテーブルのことを、**分散テーブル**といいます。分散テーブルは、ローカルでの参照と格納を容易にするために、行ごとに横方向で分けることも、列ごとに縦方向で分けることもできます。分散テーブルを縦方向に分けた列は、分散テーブルを横方向に分けた行と同様、いろいろなロケーションに配置されます。さまざまなロケーションに分けられていても、ユーザー側からは、テーブルが 1 つのロケーションに存在するかのように見えます。分散テーブルは、テーブルの特定部分にアクセスし更新する要求が、テーブルの特定部分そのものと同じロケーションから発生する場合に、特に効果的です。

ほかの用語については、 用語集を参照してください。

分散リレーショナル・データベース・アーキテクチャー のサポート

分散リレーショナル・データベース処理のための DRDA サポートは、 IBM リレーショナル・データベース・プロダクトで使われます。 DRDA サポートでは、アプリケーション・プログラムおよびリモート・リレーショナル・データベース間の通信プロトコルを定義しています。

| DRDA サポートは、IBM 環境と IBM 以外の環境の両方で、分散リレーショナル・データベース管理を提
| 供します。 IBM 環境では、次のプログラムでリレーショナル・データが管理されます。

| • DB2 Universal Database™ for iSeries
| • DB2 Universal Database for z/OS®
| • DB2 Universal Database for VSE/VM
| • DB2 Universal Database for AIX®
| • DB2 Universal Database for Linux™
| • DB2 Universal Database for HP-UX
| • DB2 Universal Database for Sun Solaris

| • DB2 Universal Database for Windows®

| DRDA サポートは、リレーショナル・データベース管理機能が**同種**環境および**異種**環境で稼働するのに必
| 要な、データベース情報にアクセスするための構造を提供します。たとえば、複数の DB2 UDB for iSeries
| 間でのリレーショナル・データのアクセスは、同種環境における分散です。一方、DB2 UDB for iSeries と
| 別のタイプのシステム、または OS/400 に組み込まれているものとは異なるクライアントとの間でのリレ
| ーショナル・データのアクセスは、異種環境での分散になります。 1 つの具体的な例としては、 DB2
| UDB for iSeries と IBM DB2 Universal Driver for SQLJ and JDBC の間で行われるリレーショナル・デー
| タのアクセスがあります。 Universal Driver に関する詳細は、『IBM DB2 Universal Driver for SQLJ and
| JDBC 1.0』を参照してください。

SQL は、標準的な IBM データベース言語です。これを使用すると、同種操作環境と異種操作環境間で分散データ処理を行えるようにするために必要な整合性を備えています。 DRDA サポートで SQL を使用すると、DRDA 実装をサポートする環境間で、データを定義、検索、および操作することが可能になります。

DRDA および CDRA サポート

分散リレーショナル・データベースを使う上で興味深い可能性の一つは、データベースを別の種類のコンピューター間で共有できるということだけでなく、それぞれのコンピューターが別の国や地域にあっても構わないということです。たとえば、すべてが iSeries server である場合のように、サーバーが同じでも、サーバーで使用する言語に応じて、データをコード化する方法は異なることがあります。 サーバーのタイプが異なれば、データをコード化する方法も異なります。 たとえば、システム/390、iSeries server、および PS/2* システムでは、数値データをそれぞれ固有の形式でコード化します。 さらに、文字データをコード化する場合には、 System/390® と iSeries server は EBCDIC コード化体系を使いますが、 PS/2® システムでは ASCII コード化体系を使います。

数値データでは、このような違いは問題になりません。異環境システムが DRDA サポートを提供している場合、数値を表示する方式が 1 つのコンピューター・システムと別のコンピューター・システムの間で異なっていれば、その違いをすべて自動的に変換します。たとえば、iSeries のアプリケーション・プログラムが DB2 UDB for iSeries データベースから数値データを読み取る場合には、 DB2 UDB for iSeries は数値データを System/390 の形式で送り、 OS/400 データベース管理システムがそれを iSeries の数値形式に変換します。

これに対して、文字データの処理は複雑になりますが、それでも分散リレーショナル・データベース内で処理することができます。文字データの処理に関する詳細については、 CDRA での文字変換を参照してください。

CDRA での文字変換

コード化スキームに違いがある (Extended Binary Coded Decimal Interchange Code (EBCDIC) と American Standard Codefor Information Interchange (ASCII) など) だけでなく、言語に関連した違いもあり得ます。たとえば、システムが異なる言語用に構成されていれば、同じコードに割り当てる文字が異なったり、同じ文字に割り当てるコードが異なったりする可能性があります。たとえば、米国英語用に構成されているシステムでは、デンマーク語用に構成されたシステムで å に割り当てるのと同じコードを、文字 } に割り当てる場合があります。さらに、これらの 2 つのシステムが、$ など、同じ文字に対して、異なるコードを割り当てる可能性もあります。

異なるサーバー間でデータを共用する場合には、文字データの認識は、ユーザーおよびアプリケーションですべて同じである必要があります。 すなわち、たとえ $ のコード化文字が各サーバーで異なっている場合

でも、ニューヨークの PS/2 とコペンハーゲンの iSeries server ユーザーは、両方とも $ を $ として認識する必要があるということです。 さらに、文字がニューヨークで保管されたものである場合に、コペンハーゲンのユーザーは、たとえその文字のコードがデンマーク語の å と同じであっても、 } として認識する必要があります。このようなことを実現するためには、$ は PS/2 システムでの適切な文字コード (すなわち、米国英語文字セットの ASCII) に変換され、さらにニューヨークからコペンハーゲンに渡る時点で、今度はデンマーク語のコード (すなわち、デンマーク語文字セットの EBCDIC) に変換されなければなりません。この種の文字変換は、iSeries server でも、他の IBM 分散リレーショナル・データベース管理プログラムの場合と同じように用意されています。この変換は、**文字データ表現体系 (CDRA)** に従って、一貫した方法で行われます。

CDRA は、サーバーによって使用する文字セットおよびコード化体系が異なっている場合でも、サーバー相互間で文字データを理解することができるように、文字データの属性を識別する方法を指定するためのものです。 サーバー相互間で変換が行われるためには、各サーバーは、それぞれ他のサーバーから受信する文字データの属性を理解しなければなりません。 CDRA では、これらの属性が**コード化文字セット識別子 (CCSID)** によって識別されるように指定します。 DB2 UDB for z/OS、DB2 UDB for VM、および OS/400 データベース管理システムの文字データには、すべて CCSID があり、これによってコード化体系、文字セット、およびコード・ページの特定の組み合わせを示します。 Extended Services® 環境の文字データは、いずれもコード・ページだけしか持っていません (ただし、他のデータベース管理プログラムは、そのコード・ページ識別子を CCSID として扱います)。**コード・ページ**とは、文字と内部コードの間の割り当ての特定のセットです。

たとえば、CCSID 37 は、コード化体系 4352 (EBCDIC)、文字セット 697 (ラテン文字、 1 バイト文字)、およびコード・ページ 37 (アメリカ/カナダ国別拡張コード・ページ) を意味します。 CCSID 5026 は、コード化体系 4865 (拡張 EBCDIC)、コード・ページ 290 の文字セット 1172 (カタカナ/漢字の 1 バイト文字セット)、およびコード・ページ 300 の文字セット 370 (カタカナ/漢字の 2 バイト文字セット) を意味します。

DRDA が使用可能なシステムには、広範囲にわたる CCSID と CCSID の間、および CCSID とコード・ページの間で文字データを変換するためのメカニズムが組み込まれています。多くの CCSID およびコード・ページの場合の文字変換は、すでにこれらのプロダクトに組み込まれています。 iSeries がサポートしている CCSID について詳しくは、iSeries Information Center の『OS/400 グローバリゼーション』トピックを参照してください。iSeries server における CCSID の使用の説明については、203 ページの『コード化文字セット識別子 (CCSID)』を参照してください。

アプリケーション・リクエスター・ドライバー・プログラム

アプリケーション・リクエスター・ドライバー (ARD) プログラムは、出口プログラムの 1 つのタイプで、 DB2 UDB for iSeries 以外のデータベース管理システムが管理するデータを SQL アプリケーションでアクセスできるようにするものです。 iSeries クライアントは、次の操作のときに、ARD プログラムを呼びます。

- 構造化照会言語パッケージの作成 (CRTSQLPKG) コマンド または CRTSQLxxx コマンドを使用して行われる、 SQL 事前コンパイルのパッケージ作成ステップにおいて、リレーショナル・データベース (RDB) パラメーターが ARD プログラムに対応する RDB 名に合致したとき。
- SQL ステートメントの処理時に、現行接続が ARD プログラムに対応する RDB 名であるとき。

このような呼び出しによって、ARD プログラムは、 SQL ステートメントとそのステートメントについての情報をリモート・リレーショナル・データベースに渡し、その結果を**アプリケーション・リクエスター (AR)** に戻すことができます。すると、AR は、その結果をアプリケーションまたはユーザーに戻します。

ARD プログラムがアクセスするリレーショナル・データベースをアクセスすることは、異環境で DRDA アプリケーション・サーバーにアクセスするのと似ています。

ARD プログラムは、 RDB ディレクトリー項目の追加 (ADDRDBDIRE) コマンドを使用することにより、システムに登録されます。指定するパラメーターの 1 つは、プログラムが置かれているライブラリーです。補助記憶域プールで構成されているシステムの場合、 ARD プログラムはシステム・データベース内のライブラリー (システム ASP または構成済みの基本 ASP の一部であるライブラリー) 内に置かれていなければなりません。

アプリケーション・リクエスター・ドライバー・プログラムについては、 iSeries Information Center の『アプリケーション・プログラミング・インターフェース (Application programming interfaces (API))』を参照してください。

iSeries server における分散リレーショナル・データベース

DB2 UDB for iSeries では、 iSeries システムのリレーショナル・データベースおよび構成済みのあらゆるユーザー・データベースで使用される、すべてのデータベース管理機能が提供されています。システムの分散リレーショナル・データベースのサポートは、通信、実行管理機能、セキュリティー機能などの機能のサポートと同様に、 OS/400 プログラムの不可欠な部分です。

iSeries は、DRDA の実装をサポートする他のサーバーとともに、分散リレーショナル・データベース・ネットワークを構成することができます。 iSeries システムは、同種または異種のどちらの環境でも、**アプリケーション・リクエスター (AR)** または**アプリケーション・サーバー (AS)** にすることができます。
iSeries システムでの分散リレーショナル・データベースの実装では、リモート作業単位 (RUW) と分散作業単位 (DUW) をサポートしています。 RUW では、1 つの作業単位の中で、1 つのデータベースに対する複数の要求を出すことができます。一方、DUW では、1 つの作業単位の中に複数のデータベースに対する複数の要求を出すことができます。

たとえば、DUW サポートでは、1 つの作業単位の中で、あるサーバーで部品の在庫量を減らし、他のサーバーで部品の在庫量を増やすことができます。そして、この作業単位の終わりで 2 フェーズ・コミット処理を使用し、これらの変更をそれぞれのリモート・データベースにコミットします。 DB2 UDB for iSeries は、分散要求をサポートしないので、各 SQL ステートメントでは、 1 つのデータベースにアクセスできるだけです。アプリケーション・プログラムに提供されるサポート・レベルは、アプリケーション・サーバー (AS) で利用できるサポート・レベルと、接続と更新が行われる順序によって決まります。詳細については、 190 ページの『分散リレーショナル・データベースへの接続』を参照してください。

ARD プログラムは、DRDA アクセスの他に、 DRDA をサポートしないデータベースのアクセスにも使用できます。 ARD プログラムによってアクセスされるリレーショナル・データベースへの接続は、異種環境サーバーへの接続のように扱われます。 このような接続は、DRDA アプリケーション・サーバーへの接続、ローカル・リレーショナル・データベースへの接続、および他の ARD プログラムへアクセスする接続と共存できます。

iSeries server では、スナップショットと複製の分散機能 (5 ページの『分散リレーショナル・データベースのその他の用語と概念』 に紹介してある) は、サーバーでは自動的には実行されません。 DataPropagator Relational Capture and Apply プロダクトを iSeries server にインストールし、構成することでこれらの機能を実行できます。また、ユーザー作成のアプリケーション・プログラムでこれらの機能を使うこともできます。分散リレーショナル・データベースでこれらの機能を編成する方法の詳細については、 『第 7 章 分散リレーショナル・データベースのデータの可用性および保護』に説明しています。

iSeries server では、 5 ページの『分散リレーショナル・データベースのその他の用語と概念』で説明されている分散要求機能は、直接にはサポートされていません。 しかし、DataJoiner® プロダクトを利用し、分散照会を実行し、さまざまなデータ・ソースからのテーブルを結合することができます。 DataJoiner は、IBM Information Warehouse ファミリー・プロダクトの包括的な情報カタログである、 DataGuide とともに機能します。 DataGuide には、企業のデータ・リソースについての情報リストを完成させるために、グラフィカル・ユーザー・インターフェースが備えられています。

OS/400 プログラムには、SQL の実行時サポートが組み込まれています。 iSeries server で分散リレーショナル・データベース要求を処理するため、または SQL コレクションを作成するために、 DB2 UDB for iSeries アプリケーション・リクエスター (AR) またはアプリケーション・サーバー (AS) に、 DB2 UDB for iSeries Query Manager and SQL Development Kit ライセンス・プログラムをインストールする必要はありません。ただし、SQL ステートメントのあるプログラムを事前コンパイルする、対話式 SQL を実行する、または DB2 UDB for iSeries Query Manager を実行するためには、 DB2 UDB for iSeries Query Manager and SQL Development Kit プログラムが必要です。

iSeries 分散リレーショナル・データベースの管理

iSeries server 上で分散リレーショナル・データベースを管理するには、 OS/400 ライセンス・プログラム内のリソースおよびツールに関する広範な知識が必要です。 本書には、iSeries server における分散リレーショナル・データベースの管理に役立つ、オペレーティング・システムで使用可能な各種機能の概略を説明しています。 本書では、iSeries server システムのネットワーク (*同種* 環境) における分散リレーショナル・データベースの機能およびタスクについて説明しています。 同種環境と異種環境における iSeries 分散リレーショナル・データベース機能の相違点については、本書では概説で紹介する程度に留めてあります。

適切に実装された分散リレーショナル・データベースを使うと、リモート・サーバーのデータベースへ簡単にアクセスしたり、存在する場所を認識することなくデータベース・ファイルを処理したり、アプリケーション・プログラムに変更を加えずに、データベースの一部を他のサーバーへ移動できるようになります。

それぞれの分散リレーショナル・データベースを効果的に実装するため、次に示すかぎとなる分野の要件に精通している必要があります。

- 分散リレーショナル・データベースの計画および設計に、分散データベースの計画および設計時に考慮すべき重要な事項が説明してあります。
- iSeries 分散リレーショナル・データベースのための通信では、分散リレーショナル・データベースを処理するために、ネットワークをセットアップしたり、既存のネットワークを変更する場合にどの通信機能を使うかについて説明しています。
- iSeries 分散リレーショナル・データベースのセキュリティー では、リモート・リレーショナル・データベースへの通信および DRDA アクセスなどの、 iSeries 分散リレーショナル・データベースのセキュリティーに関する情報が提供されています。
- iSeries 分散リレーショナル・データベースのセットアップ では、分散データベースにデータを入力する方法、ならびに iSeries server におけるサブシステムおよびリレーショナル・データベース・ディレクトリーに関する情報が提供されています。
- 分散リレーショナル・データベースの管理および操作の作業では、ネットワークの中で行われる分散リレーショナル・データベース作業を管理できる方法について説明しています。
- 分散リレーショナル・データベースのデータの可用性および保護では、iSeries server 上でプログラムおよびデータを保護し、問題が発生した場合に回復時間を短縮するためのツールおよび技法を説明しています。また、ネットワーク・ユーザーが、必要時に、確実にネットワーク内のリレーショナル・データベースおよびテーブルにアクセスできるようにする方法についても示しています。

- 分散リレーショナル・データベースのパフォーマンスでは、ネットワーク、システム、およびデータベースの設計を向上させる方法について説明します。
- 分散リレーショナル・データベースの問題の処理では、分散リレーショナル・データベースで起こり得る問題のいくつか、およびそれらの問題のトラブルシューティングの方法について説明します。
- 分散リレーショナル・データベース・アプリケーションの作成では、分散リレーショナル・データベースの場合のプログラミングの問題の概要を示しています。

iSeries 分散リレーショナル・データベースとともに実行する異なる分散リレーショナル・データベース・プラットフォームに関する考慮事項については、 247 ページの『付録 B. DRDA を使用したプラットフォーム間アクセス』で説明しています。

DRDA をサポートする他の IBM システムについてさらに知りたい場合は、該当のシステムとともに提供される情報、または分散リレーショナル・データベース・ライブラリーにリストされている資料、および 289 ページの『付録 E. 関連情報』の他の IBM 分散リレーショナル・データベースのプラットフォーム・ライブラリーを参照してください。

例: Spiffy 社の分散リレーショナル・データベース

IBM の資料には、分散リレーショナル・データベース・サポートの説明にあたって、Spiffy 社がしばしば登場しています。本書では、この架空の会社に若干の変更を加えて、 iSeries server ネットワークにおける DRDA の iSeries server サポートを示しています。本書で使用する例は、特定の機能、接続、および処理を示すためのものです。したがって、本書以外の分散リレーショナル・データベース関係の資料で使用されている例とは正確に一致しない場合がありますが、なじみやすい例になるよう心掛けてあります。

Spiffy 社は架空の企業には違いありませんが、本書で説明する同社の業務は、同種の構造を持つ数社で行われている業務をモデルにしています。ただし、本書で取り上げる例は、たとえこの架空の会社の場合でも、分散リレーショナル・データベースを使用して行えることすべてを説明するものではありません。

以下のトピックでは、 Spiffy 社の組織と分散リレーショナル・データベース・サポートの使用方法に関する情報が扱われています。

- Spiffy 社の組織とシステム・プロファイル
- Spiffy 社の自動車保守サービスの業務処理
- Spiffy 社の分散リレーショナル・データベースの管理

Spiffy 社の組織とシステム・プロファイル

Spiffy 社は、全国的な規模を持つ製品卸売業者で、地域支社や地区販売店網を介して、得意先小売業者を対象に、自動車を中心とする製品の販売および保守サービスを行っています。今日の自動車産業の競争の激しさを思えば、Spiffy 社のような企業の経営の成否は、質的に高いサービスを提供し、得意先に対する予備部品の納期を順守できるかどうかにかかっています。この競争に対処するために、Spiffy 社では、広大なサービス網を自社の販売店組織に組み込んで確立しました。

販売店組織を率いるのが、イリノイ州シカゴにある本社車両流通センターです。北米全域の数カ所に地域流通センターを置いています。そのうちの 2 つは、ミネソタ州のミネアポリスとミズーリ州のカンザス・シティーにあります。これらの流通センターでは、地域在庫を設けることによって、車両および予備部品の流通コストを最小限に抑制しています。ミネアポリス地域流通センターは、ほぼ 15 社の販売店を担当し、カンザス・シティー地域流通センターは、30 社にも及ぶ販売店を担当しています。

11 ページの『例: Spiffy 社の分散リレーショナル・データベース』には、Spiffy 社のシステム編成図が示してあります。

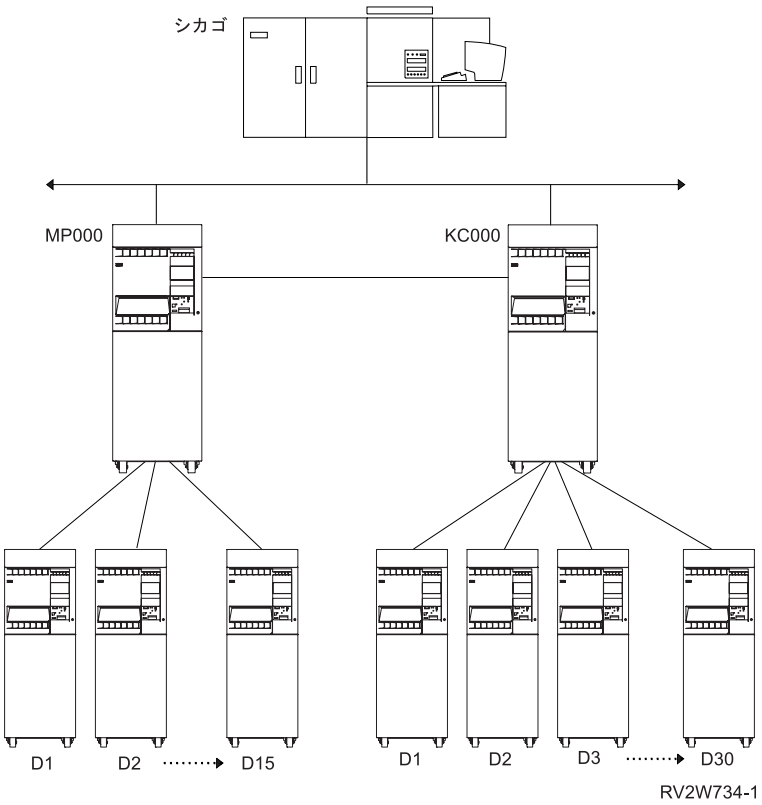


図 5. Spiffy 社のシステム編成

Spiffy 社では全国的な統合通信ネットワークを構築中です。自動車事業部では、地域流通センターおよび販売店を結ぶ iSeries server のネットワークを設けています。これを本社車両流通センターのSystem/390 に接続します。このネットワークは、競争上の優位を維持するために不可欠の事業資産と見なされます。

本社の流通センターが関係のある意思決定支援ソフトウェアをとともに、 System/390 で DB2 UDB for z/OS を実行します。このシステムを使用するのは、各種のアプリケーション・プログラムを使用して一度に処理しなければならないデータの量が多いからです。本社車両流通センターのシステムは、自動車事業部のデータ処理専用ではありません。会社の作業や処理の中には、まだ分散データベース環境での操作の対象になっていないものがあり、それらを処理しなければならないからです。地域流通センターは iSeries システムを実行しています。そこでは、SNADS および SDLC プロトコルを使用する 5250 表示装置パススルーで APPC/APPN を使用しています。

すべての販売店で iSeries server が使用されており、それらのサイズはそれぞれ異なります。これらのシステムは、SDLC プロトコルを使用して地域支社に接続されます。最大規模の販売店では、会社のデータ処理機能を担当する、パートタイムのプログラマー 1 名とシステム・オペレーター 1 名を置いています。しかし、ほとんどの販売店ではプログラミングの専門的技術を備えたものは雇用しておらず、小規模な販売店では一般的なコンピューターの知識以上のものを備えた従業員をまったく置かないところもあります。

Spiffy 社の自動車保守サービスの業務処理

Spiffy 社の自動車事業部では、業務は本書で取り上げる分散リレーショナル・データベース環境で自動化されています。例が必要以上に複雑になるのを避けるために、社内で車両保守サービスにかかわる機能だけを考えることにします。

販売店には、規模に応じて 2000 から 20,000 社の得意先リストがあります。つまり、保守サービス受注が、小規模販売店の場合で毎日 5 件、大規模販売店の場合で毎日 50 件の割合になります。これらの保守サービス受注には、定期保守、保証修理、定期修理、および部品受注があります。

販売店では、手持ち在庫は需要の多い予備部品に限っており、自社の在庫データベースを管理しています。前述した両地域流通センターでは、要求に応じて部品を提供しています。販売店在庫については、予測モデルで制御されるバッチ処理による定期的な調達も行われます。

Spiffy 社の分散リレーショナル・データベースの管理

各販売店では、それぞれが独立した企業として、自社のデータ処理リソースおよび手順を管理します。Spiffy 社では、各販売店に対して、それぞれが 1 つ以上の iSeries server を備え、それらのサーバーを特定の時間にネットワークで使えるようにしています。 ただし、サーバーのサイズおよびシステム上で自動化される業務処理数については、各販売店ごとの要件および使用可能なリソースに応じて決まります。

Spiffy 社では、販売店がすべて在庫の分散リレーショナル・データベースを使用できる状態になっている必要があります。 Spiffy 社には系列販売店があるので、これらを完全に補うための販売店ソフトウェアを備えていますが、これには分散リレーショナル・データベース環境にアクセスできるものもできないものもあります。 Spiffy 社の系列販売店では、このソフトウェア・ツールの全セットを使用します。フランチャイズ契約加盟販売店の場合も、特に Spiffy 社の営業方式に見合う調整がなされているので、ほとんどがこのソフトウェア・ツールの全セットを使用します。

地域流通センターでは、担当する地域の在庫を管理します。また、地域で使用されるすべての分散データベース・リソースの、データベース管理者としての機能も果たします。担当する責任は、各販売店のデータ処理能力のレベルによって異なります。地域流通センターは、地域の販売店が援助を必要とする際に、まず接触する連絡先になります。

ミネアポリス地域流通センターは、サーバーおよびネットワークに関する広範な経験および知識を備えた iSeries プログラマーのスタッフを備えています。 担当する販売店は他の地域流通センターの場合のほぼ半数ですが、これはミネアポリス地域流通センターがネットワーク全体の iSeries サポート機能に集中できるようにするためです。 これらの機能には、アプリケーション・プログラム開発、プログラム保守、および問題処理などがあります。

次に挙げるのは、ネットワークの各活動レベルごとのデータベースに関する責任範囲です。

販売店:

- サーバーの基本操作および管理の実行。
- 地区ユーザーの登録。

地域流通センター:

- 新規販売店のデータ処理のセットアップ。
- 解約販売店のデータベース・リソースの分散。
- 地域内ネットワーク・ユーザーの登録。
- 地域の在庫の管理。

- 販売店に関する保守サービス計画の開発。
- 販売店を対象とする援助の窓口の運営。

上記の地域流通センター活動に加えて、ミネアポリス iSeries server 専門センターでは、次の活動も行います。

- iSeries server のアプリケーション開発。
- 地域流通センターを対象とする援助の窓口の運営。
- データベース・パフォーマンスの調整。
- データベース問題の解決。

本書で使用する例は、上記の活動の 1 つ以上に関連するものです。多くの例を挙げて、得意先の保守または修理の予定に合わせて、在庫から部品を調達する過程を示しています。その他の例では、Spiffy 社の分散リレーショナル・データベース・ネットワークのサーバーのセットアップ、保護、モニター、および問題解決に使われる分散リレーショナル・データベース管理作業を示しています。

第 2 章 分散リレーショナル・データベースの計画および設計

分散リレーショナル・データベースを正しく操作するためには、計画がその第 1 要件となります。分散リレーショナル・データベースの使用を決定するにあたっては、企業としての要件および目標を考慮しなければなりません。アプリケーション・プログラムのコーディング方法、データとの関連におけるアプリケーション・プログラムの所在場所、およびアプリケーション・プログラムをデータに接続するネットワーク設計は、いずれも設計上の重要な考慮事項です。

分散リレーショナル・データベースの場合は、単一の iSeries リレーショナル・データベースを扱う場合よりも、データベース設計が重要となります。複数の iSeries server を考慮しなければならないので、ネットワーク全般にわたって整合性を持つ管理指針を開発しなければなりません。指針を作る際に特に注意が必要な操作を、以下に示します。

- 一般的な操作
- ネットワーキング・プロトコル
- システム・セキュリティー
- 会計
- 問題分析
- バックアップおよび回復処理

分散リレーショナル・データベースを準備するにあたっては、業務上の要件とリレーショナル・データベース技術の両方を理解していなければなりません。

分散リレーショナル・データベースの計画および設計は、相互に密接に結び付いているので、この章では、以下に挙げる関連タスクの説明にあたっては、計画と設計という項目を一括して扱っています。

- 要件および要望事項の識別
- アプリケーション、ネットワーク、およびデータの設計
- 管理方針の開発

分散リレーショナル・データベースの要件および要望事項の識別

分散リレーショナル・データベースの要件および要望事項の分析にあたっては、次のことを考慮してください。

- 分散リレーショナル・データベースのデータ要件。計画に見合うデータがどんなものであり、そのデータを誰が、どのような理由で、どの程度の頻度で使用するのか。
- 分散リレーショナル・データベースの能力。要件が分散リレーショナル・データベースによる解決に適しているかどうか。
- 分散リレーショナル・データベースの目標および動向。分散リレーショナル・データベースが解決策として使用できると考えられるとして、短期的および長期的にどのような目標を達成することができるか。

分散リレーショナル・データベースのデータ要件

分析の最初のステップでは、どのような要因がどのようにデータに作用するかを判別します。以下の質問項目を検討してください。

- 関与するロケーションはどこか?
- 想定するトランザクションの種類は?
- 各トランザクションでどのようなデータが必要か?
- データ項目の相互の依存関係、とくに参照限界は? たとえば、1 つの表の情報を別の表の情報と突き合わせて調べる必要があるか? (その必要があれば、両方の表は同じロケーションに保持しなければなりません。)
- 現在データが存在しているか? 存在している場合は、そのロケーションはどこか? 誰がデータを「所有」しているのか、つまり、誰がデータの正確度の維持管理を担当しているのか?
- 必要なデータの可用性に対する優先度は? ロケーション間での保全性は? 無許可アクセスからのデータの保護は?
- データに対して想定しているアクセス・パターンは? たとえば、データの読み取りか、更新か、あるいはその両方か? 頻度は? 典型的なアクセスで戻るデータは多いか、少ないか?
- 各トランザクションのパフォーマンスの期待値は? 受け入れることのできる応答時間は?

分散リレーショナル・データベースの能力

分析の 2 番目のステップでは、データ要件が分散リレーショナル・データベースによる解決に適しているかどうかを判定します。

データベース処理がほとんどローカルで行われ、リモート・データにアクセスする必要はたまにしかないようなアプリケーションの場合は、分散リレーショナル・データベースの使用が概して*適して* います。

次のような要件を伴うアプリケーションの場合は、分散リレーショナル・データベースの使用は通常*適して*いません。

- データを中央で保持し、しかも リモート・ユーザーが行う必要のある作業のほとんどが中央で行われる。
- 一貫して高いパフォーマンス、とくに一貫して高速の応答時間を必要とする。ネットワーク内でデータを移動すれば、それだけ時間がかかることになります。
- 一貫して高い可用性、とくに 24 時間、週 7 日の可用性を必要とする。ネットワークは、それにかかわるシステム、および通信回線や通信制御装置など、中間的な構成要素が多くなり、それだけ故障の機会が増えます。
- 現在利用できないか、まだ発表されていない分散リレーショナル・データベース機能を必要とする。

分散リレーショナル・データベースの目標および動向

分析の 3 番目のステップでは、短期目標および長期目標の評価を行います。

SQL は、標準的な IBM データベース言語です。目標および動向に、異環境システム間における可搬性またはリモート・データ・アクセスが含まれる場合には、 iSeries server で分散リレーショナル・データベースを使うようにします。

分散作業単位の分散データベース機能は、 DataPropagator Relational Capture and Apply の提供する追加データ・コピー機能と同様に、iSeries server で実行できる処理範囲を広げます。 ただし、分散データベース・アプリケーションで、 iSeries server で現在使用できない機能が必要な場合は、その機能がオペレーティング・システムで使用可能になるまで、他のオプションを使うことができます。 たとえば、次のいずれかを行うことができます。

- 必要な機能を独自に用意する。

- 新しい機能が使用可能になるときに備えて、分散リレーショナル・データベースに関する計画を実施する。
- 目標および要件の再評価を行って、現在使用可能または発表済みの機能で満足できないか検討する、代替となる解決法のいくつかを表 3 にリストしています。これらの代替解決法を用いれば、使用可能な機能を補ったり、あるいは使用可能な機能に代えることができます。

表 3. 分散リレーショナル・データベースに対する代替解決法

解決法	説明	利点	欠点
分散データ管理機能 (DDM)	オペレーティング・システムの機能の 1 つで、これを使用すれば、 1 つのシステムのアプリケーション・プログラムまたはユーザーは、リモート・システムに格納されているデータベース・ファイルを使用できます。この場合、システムは通信ネットワークによって接続されている必要があり、リモート・システムも DDM を使用している必要があります。	• 単純な読み取りアクセスおよび更新アクセスでは、パフォーマンスが SQL の場合より優れている。 • 既存のアプリケーションを作成し直す必要がない。 • これを使用すれば、S/38、S/36、および CICS* にアクセスすることができる。	• 複雑な機能では、SQL の方が効率がよい。 • 他の分散リレーショナル・データベース・プラットフォームにアクセスすることができない。 • CCSID および数値データの変換を行うことができない。
システム間通信機能/共通プログラミング・インターフェース (ICF/CPI 通信)	ICF は、オペレーティング・システムの機能の 1 つで、これを使用すれば、プログラムは別のプログラムまたはシステムと対話式に通信することができます。 CPI 通信は、プログラム間通信を使用するアプリケーションのために、整合性のあるアプリケーション・インターフェースを提供する、呼び出しレベルのインターフェースです。これらのインターフェースは、SNA の論理装置 (LU) 6.2 アーキテクチャーを使い、リモート・システムでのプログラムとの会話を確立し、データを送受し、制御情報を交換し、会話を終了し、パートナー・プログラムにエラーを通知します。	• これを使用すれば、アプリケーションをカスタマイズして要件に適合させることができる。 • これによって、より高いパフォーマンスが得られる。	分散リレーショナル・データベースおよび DDM に比較して、通信およびデータ交換要件をサポートするためにいくぶん複雑なプログラムが必要になる。
ディスプレイ・パススルー	通信機能の 1 つで、これを使用すれば、ユーザーは iSeries server から別の iSeries server にサインオンし、そのサーバーのプログラムやデータを使用することができます。	• リモート・システムのアプリケーションおよびデータがローカル・システムからアクセス可能である。 • データが一時的なものであり、しかも複数のサーバーのユーザーが 1 つのサーバーのデータを大量に必要とする場合に、迅速なアクセスが可能である。	画面更新での応答時間がローカル接続装置の場合より遅い。

分散リレーショナル・データベースは、通常、業務上の要件が変化し、新製品が使用可能になるのに応じて、単純なものから複雑なものへと変遷します。要件および要望事項の分析にあたっては、このことを考慮してください。

分散リレーショナル・データベース用のアプリケーション、ネットワーク、およびデータの設計

分散リレーショナル・データベースの設計には、以下に関する選択が伴います。

- アプリケーション
- ネットワークに関する考慮事項
- データに関する考慮事項

ヒント: 分散リレーショナル・データベース・アプリケーションの設計

分散リレーショナル・データベース・アプリケーションと、ローカル・データベースだけに使用するために開発されたアプリケーションとは、要件に大きな違いがあります。このような差異を考慮して適切に計画するためには、次のことを念頭に置いてアプリケーションを設計してください。

- 使用できる場合には、分散作業単位 (DUW) 機能を利用してください。

  注: バージョン 5 リリース 1 より前の OS/400 では、 iSeries server の TCP/IP で 2 フェーズ・コミット・サポートを使用できませんでした。
- 共通インターフェースを使用してプログラムをコーディングしてください。
- 複雑なアプリケーションは、それを細かく部分に分割し、分割した各部分をそれぞれの処理に最も適したロケーションに置くことを考慮します。処理するデータが入っているリモート・ロケーションで、ストアード・プロシージャーを実行するために SQL CALL ステートメントを使用するのは、アプリケーションで分散処理を行う 1 つのよい方法です。このストアード・プロシージャーは、DB2 Universal Database for iSeries アプリケーション・サーバーで実行するときは、 SQL 操作である必要はありません。統合データベース入出力を使用してもよく、別のタイプの処理を実行することもできます。
- 初期のデータベース・アプリケーションの準備、テスト、および使用の方法を研究してください。
- 可能であれば、SQL セット処理機能を活用してください。これによって、アプリケーション・サーバーとの通信が最小限になります。たとえば、この処理が可能な場合、1 つの SQL ステートメントによって、複数の行を更新することができます。
- プログラムの準備時に RUW 接続方式を使用している場合、または分散アプリケーションの他のノードが DUW をサポートしていない場合は、 1 作業単位内のデータベース操作は、単一の場所で行わなければならないことに注意してください。
- DUW 接続方式では、1 つのステートメントを複数のリレーショナル・データベースに送るのに制約があることを忘れてはなりません。
- 接続管理方式の選択によって、パフォーマンスは影響を受けます。異なるリモート・リレーショナル・データベースを切り替えながら使用する必要がない場合は、 RUW 接続管理方式を使用するのがよいと思われます。これは、DUW 接続管理で使用される 2 フェーズ・コミットでは、負担が大きくなるからです。

  ただし、複数のリモート・データベース管理システム間を頻繁に切り換える必要がある場合は、 DUW 接続管理を使用してください。 DUW 接続管理を使用して実行しているときは、あるデータベース管理システムとの通信会話は、もう 1 つのデータベース管理システムに接続を切り換えるときに、終了させる必要がありません。同種環境では、これは異環境ほど大きな影響はありません。同種環境での会話は、省略時の DDMCNV(*KEEP) ジョブ定義属性を使用することによって、活動状態のままにしておく

ことができるからです。ただし、同種環境でも、新しい接続を設定するために、カーソルをクローズし通信フローを送信するコストを除くために、 DUW を使用してパフォーマンスを上げることができます。

- 接続管理方式は、CONNECT ステートメントの働きを決定します。 RUW 接続管理方式では、CONNECT ステートメントは、リレーショナル・データベースへの新しい接続を設定する前に、既存の接続があればそれらを終了させます。 DUW 接続管理方式では、CONNECT ステートメントは、既存の接続を終了させません。

分散リレーショナル・データベースのネットワークに関する考慮事項

ネットワークの設計は、分散リレーショナル・データベースのパフォーマンスに直接影響します。特定のネットワークで十分に機能する分散リレーショナル・データベースを適切に設計するには、以下のことを行います。

- アプリケーションのパフォーマンスにとって回線速度が非常に重要になることがあるので、ネットワーク内の適切な個所に十分な容量を用意して、主な分散リレーショナル・データベース・アプリケーションに効率的なパフォーマンスが達成できるようにしてください。詳細は、V5R1 Supplemental Manuals Web サイトにある、 「*Communications Management*」  を参照してください。
- 使用可能な通信ハードウェアおよびソフトウェアを評価し、必要に応じて、上位移行能力も評価を行ってください。
- APPC 接続の場合、ネットワークの定義時に指定するセッション限界および会話限界を考慮してください。
- 必要とされる (テスト環境および実稼働環境の両方で) ハードウェア、ソフトウェア、および通信機器、ならびに分散リレーショナル・データベース・ネットワークの機器の最も優れた構成を識別してください。
- TCP/IP をサポートするために必要なスキルは、 APPC をサポートするために必要なスキルとは異なっていることを考慮してください。
- エンド・ユーザー・グループとの間の初期サービス・レベルに関する同意 (所定の分散リレーショナル・データベース・アプリケーションの場合に予測される応答時間など)、ならびに提供される実際のサービスの監視および調整についての方針を考慮に入れてください。
- 現在のスレッドで補助記憶域プール (ASP) グループに設定されている AR からデータベースにアクセスする際に、 APPC 保護 DUW 会話は使用できないことを理解していなければなりません。
- 分散リレーショナル・データベースのデータベース・オブジェクト、および分散リレーショナル・データベースの各ロケーションでの命名についての方針を決めてください。 **ロケーション**とは、分散リレーショナル・データベースに関与するリレーショナル・データベース管理システムの相互接続ネットワークの中にある、特定のリレーショナル・データベース管理システムのことです。この意味での「ロケーション」は、独立 ASP グループで構成されたシステム内のユーザー・データベースを指すこともあります。この方針を決めるにあたっては、次のことを考慮してください。
  - 分散データベースのオブジェクトの完全修飾名には 3 つ (2 つではなく) の部分があり、最上位の修飾子でオブジェクトのロケーションを識別します。
  - 分散リレーショナル・データベースの各ロケーションには、それぞれ固有の識別名を与えてください。データベースの各オブジェクトにも、それぞれ固有の識別名が必要です。識別名が重複していると、重大な問題を生じることがあります。たとえば、ロケーション名およびオブジェクト名が重複していると、アプリケーションが意図したもの以外のリモート・データベースに接続される場合があり、しかもいったん接続された場合には、意図したもの以外のオブジェクトにアクセスすることになります。ネットワークの結合時には、命名に特別の注意を払ってください。

– また、ユーザー・データベースでの各ロケーションには、固有の識別名を与えなければなりません。'PAYROLL' という名前のユーザー・データベースが 2 つのそれぞれ別個のサーバーにある場合、アプリケーションが同じサーバーからそれらの両方にアクセスしなければならないときに、名前に関する競合が発生します。独立 ASP 装置を構成するときに、ユーザーはその ASP 装置自体の名前とは異なる RDB 名をその装置に指定できるということを忘れないでください。これが、 ASP グループ内で基本装置に関連付けられている RDB 名であり、これによって、そのユーザー・データベースは認識されます。

分散リレーショナル・データベースのデータに関する考慮事項

データを必要とするアプリケーションとの関連におけるデータの配置は、分散リレーショナル・データベースの設計時の重要な考慮事項になります。配置に関する決定にあたっては、次のことを考慮してください。

- アプリケーションで必要とするパフォーマンスのレベル。
- ロケーション間でのデータのセキュリティー、現行性、整合性、および可用性に関する要件。
- 必要とされるデータの量およびデータ・アクセスの予測パターン。
- 必要とされる分散リレーショナル・データベース機能が使用可能であるかどうか。
- サーバーをサポートするのに必要な技能と実際に使用可能な技能。
- 誰がデータを「所有」しているのか、つまり、誰がデータの正確度の維持管理を担当しているのか。
- システム間セキュリティー、会計、監視と調整、問題処理、データのバックアップと回復、および変更制御に関する管理方針。
- データをネットワーク内のどこに置くか、維持管理するデータのコピーは 1 つにするか複数にするかなど、分散データベース設計に関する決定事項。

分散リレーショナル・データベースの管理方針の開発

この項では、分散リレーショナル・データベースの管理に関する以下の方針を説明します。

- 分散リレーショナル・データベースの一般的な操作
- 分散リレーショナル・データベースのセキュリティーに関する考慮事項
- 分散リレーショナル・データベースの会計
- 分散リレーショナル・データベースの問題分析
- 分散リレーショナル・データベースのバックアップおよび回復

分散リレーショナル・データベースの一般的な操作

分散リレーショナル・データベースの一般的な操作の計画にあたっては、パフォーマンスと可用性の両方を考慮してください。

以下に設計上の考慮事項が挙げてありますが、分散リレーショナル・データベースのパフォーマンスと可用性の両方の向上を図る上で役立てることができます。

- 実行頻度が高いか、またはデータの送受信量が多いトランザクションを伴うアプリケーションの場合には、データと同じロケーションにアプリケーションを保持するように努めてください。
- 異なるロケーションにあるアプリケーションによる共用の必要があるデータの場合は、活動の最も多いロケーションにデータを置いてください。

- 1 つのロケーションにあるアプリケーションと別のロケーションにあるアプリケーションが、同程度にデータを必要とする場合には、データのコピーを両方のロケーションに保持することを考慮してください。複数のロケーションにコピーを保持する場合には、管理方針に関する以下の質問事項を検討してください。
  - コピーに更新を行うことをユーザーに許可するか?
  - 最新のデータによるコピーの更新の方法および時期は?
  - コピーのバックアップは、すべてのコピーについて行わなければならないのか、コピーの 1 つについて行えば十分なのか?
  - 一般管理活動をすべてのコピーに関して一貫して行う方法は?
  - コピーの 1 つを削除することが許される時期は?
- 分散データベースの管理は中央ロケーションによるのか、各データベース・ロケーションによるのかを考慮してください。

パフォーマンスについては、以下のことを行うことによっても改善できます。

- データとアプリケーションを別々のロケーションに保持しなければならない場合には、次のようにして、パフォーマンスが許容限度内に収まるようにします。
  - アプリケーションで使用されるデータの列の検索だけに留めることによって、ネットワーク内のデータ通信量をできるだけ低く抑えてください。つまり、SELECT ステートメントの一部として列名のリストの代わりに * を使用することは避けてください。
  - リモート・ロケーションとの間で大量のデータの送信または受信を行うステートメントのコーディングを、プログラマーに行わせないようにしてください。つまり、SELECT ステートメントの WHERE 文節の使用を奨励して、データの行数を制限するようにしてください。
  - 参照保全、トリガー、およびストアード・プロシージャー (リモート・リレーショナル・データベース管理システムに対する CONNECT ステートメントの後の SQL CALL ステートメント) を使用してください。これによって、**アプリケーション・サーバー (AS)** への処理が分散され、したがって実質的な回線通信量が減り、パフォーマンスが改善されます。
  - FOR FETCH ONLY 文節を指定することによって、読み取り専用照会を適宜使用してください。
  - 照会をブロック化する規則に注意してください。たとえば、iSeries - iSeries 間の照会では、COMMIT(*NONE) についてだけ、あるいは ALWBLK(*ALLREAD) が指定されている場合には COMMIT(*CHG) と COMMIT(*CS) についてだけ、読み取り専用データのブロック化が行われます。
  - 可能な限りリモート・データではなくローカル・データを使用することによって、アクセス・データに対するアクセス数を低く抑えてください。
  - SQL SET 命令を使用して、単一の SQL 要求でアプリケーション・リクエスターの複数行を処理してください。
  - RUW 接続管理では DDMCNV(*KEEP) を使用することによって、または、DUW 接続管理を使用することによって、接続の停止を避けるようにしてください。
- 以下のようにして、十分なネットワーク容量を用意します。
  - 高速、高帯域幅の回線を導入することによって、あるいはネットワーク内の適当な点に回線を追加することによって、ネットワークの容量を増やしてください。
  - 競合を減らすか、または特定の処理装置の競合バランスを改良してください。たとえば、既存のアプリケーションをホスト・サーバーから部門サーバーに移動するか、または分散リレーショナル・データベースの作業をグループ化してバッチ処理します。
- 優れたテーブルの設計を奨励します。分散リレーショナル・データベース・ロケーションで、基本キー、テーブルの索引、および正規化技法の適切な使用を奨励してください。

- WHERE 文節で使用されるホスト変数のデータ・タイプが、対応するキー列データ・タイプの中のデータ・タイプと矛盾しないようにしてください。たとえば、浮動小数点のホスト変数は、異なるデータ・タイプの列に構築された索引では使用できません。

可用性については、以下のことを行うことによっても改善できます。

- 一般的には、ネットワーク内のデータ通信量を制限するようにしてください。
- データとアプリケーションを別々のロケーションに保持しなければならない場合には、次のようにして、可用性が許容限度内に収まるようにします。
  - 代替ネットワーク経路を確立してください。
  - 時間帯の違いが可用性に及ぼす影響を考慮してください。
    - サーバーを始動する適格者が確保できるか?
    - 時間外のバッチ作業が処理の障害になるか?
  - 優れたバックアップおよび回復機能を確保してください。
  - 誰もがバックアップおよび回復に熟練するようにしてください。

分散リレーショナル・データベースのセキュリティーに関する考慮事項

分散リレーショナル・データベース計画の一部として、分散データの保護について決定しなければならない事項があります。そのような決定事項には以下のものがあります。

- 他のロケーションのユーザーにとってアクセス可能なシステムがどれで、それらのシステムに対してアクセスができる他のロケーションのユーザーが誰なのか。
- システムに対するアクセスを規制する厳しさの程度。たとえば、リモート・ユーザーによる会話の開始時には、ユーザー・パスワードを必須とするのか?
- パスワード入力を暗号化された形式で回線上に流すのか?
- クライアント・ジョブを実行するときのユーザー・プロファイルを、接続先のリレーショナル・データベースの名前に応じて、別のユーザー識別コードまたはパスワードへ割り当てる必要があるか?
- 他のロケーションのユーザーにとってアクセス可能なデータがどれで、そのデータに対してアクセスができる他のロケーションのユーザーが誰なのか。
- ユーザーがデータに対して講じることを許される処置。
- データに対する権限の認可は中央管理かローカル管理か。
- 複数のシステムがリンクされるため、特別の予防策を講じるべきかどうか。たとえば、名前変換を使用するのか?

以上の決定を行うにあたっては、ロケーションの選択時に次のことを考慮してください。

- 物理的な保護。たとえば、入室制限を設けた部屋をロケーションに用意することができます。
- システム・セキュリティーのレベル。システム・セキュリティーのレベルは、ロケーションによって異なる場合があります。分散データベースのセキュリティー・レベルは、ネットワークで使用される最低レベルのセキュリティー以下になります。

  APPC で接続しているすべてのサーバーで、次のことを行うことができます。
  - 両方のサーバーが iSeries server である場合、暗号化された形式のパスワードで通信する。
  - 1 つのサーバーがネットワーク内の別のサーバーと通信するための要求を受信した場合に、要求側サーバーが実際に「自称どおり」であり、しかも受信側サーバーと通信する権限を認可されていることを検査する。

  すべてのサーバーで次のことを行うことができます。

– リモート・データ・アクセスの許可に先立って、検査のために、ユーザーの識別名およびパスワードをローカル・サーバーからリモート・サーバーへ渡す。
– テーブルやビューなどの SQL オブジェクトをアクセスおよび操作する特権の認可および取り消しを行う。

iSeries server には、セキュリティー監査機能が組み込まれており、これを使用すれば、無許可でデータにアクセスしようとする試みのトレース、ならびにセキュリティーにかかわるその他のイベントのトレースを行うことができます。 さらに、サーバーでは、リモート・サーバーによる分散データベースへのアクセスをすべて防止できる機能も用意しています。

– セキュリティーに関連したコスト。セキュリティーのコストを考慮するにあたっては、セキュリティー関連製品の購入コストと、情報関係スタッフが次の活動を行うための時間コストの両方を考慮してください。
  - リモート・データにアクセスするユーザーのサーバー識別名を、ローカル・サーバーとリモート・サーバーの両方で維持管理する。
  - 監査機能をサイト間で調整する。

セキュリティーの詳細については、 iSeries 分散リレーショナル・データベースのセキュリティーを参照してください。

分散リレーショナル・データベースの会計

分散データの使用に関する会計および請求が可能でなければなりません。以下の事項を考慮してください。

- 分散データの使用に関する会計では、1 つ以上のリモート・サーバーのリソースの使用、ローカル・サーバーのリソースの使用、およびサーバーを接続するネットワーク・リソースの使用が対象になります。
- 会計情報は、サーバーごとに独立して累積されます。 ネットワーク会計情報は、サーバーで累積されるデータとは別に累積されます。
- さまざまなサーバーの時間帯については、会計情報を相関付けようとするときに、考慮に入れなければならないことがあります。 各サーバー・クロックは、リモート・サーバー・クロックに同期化していないことがあります。
- 各サーバーの許容会計コード (番号) の間には、相違点が存在していることがあります。 たとえば、iSeries server では、会計コードを最大 15 文字に制限しています。

分散データの使用に関する会計では、次の機能が使用可能です。

- iSeries server ジョブ会計ジャーナル。 iSeries server では、各分散リレーショナル・データベース・アプリケーションごとに、ジョブ会計情報をジョブ会計ジャーナルに書き込みます。 ジャーナル表示 (DSPJRN) コマンドを使用すると、ジャーナル項目をデータベース・ファイルに書き込めます。次に、ユーザー作成プログラムと照会機能のどちらかを使用すれば、会計データを分析することができます。詳しくは 110 ページの『分散リレーショナル・データベースのジョブ会計』をご覧ください。
- NetView* 会計データ。 NetView® ライセンス・プログラムを使えば、ネットワーク・リソースの使用に関する会計データを記録することができます。

分散リレーショナル・データベースの問題分析

問題分析は、分散データベース環境で管理する必要があります。問題分析には、サーバーのネットワーク内で処理されるアプリケーションに関する問題の識別と解決の両方が含まれます。 以下の事項を考慮してください。

- 分散データベース処理の問題はさまざまな仕方で現れます。たとえば、問題を検出したサーバーによって、エラー戻りコードが分散データベース・アプリケーションに渡される場合があります。 さらに、応答が遅かったり、間違っていたり、あるいは存在しない場合もあります。
- 分散データベース処理の問題を診断するためのツールが使用可能です。たとえば、各分散リレーショナル・データベース・プロダクトごとに、分散データ処理の問題を診断を援助できるトレース機能が用意されています。
- サーバー障害が iSeries server によって検出された場合、サーバーは、障害の検出直後に、プログラム状況に関する情報をログ記録します。

注: IBM プログラムに対する修正が必要であり、しかもネットワーク分散管理機能 (NDM) を備えたシステム/390 (System/390*) がネットワーク内に導入されている場合には、 NDM および分散サーバー・ノード・エグゼクティブ・プロダクトを使用して、ネットワーク内の該当するシステムとの間で更新および置換を送受信することができます。

分散リレーショナル・データベースのバックアップおよび回復

単一サーバー環境では、バックアップおよび回復はローカルで行われます。 しかし、分散データベースでは、バックアップおよび回復はリモート・ロケーションにも影響します。

iSeries server では、個々のテーブル、コレクション、またはコレクション・グループをバックアップおよび回復することができます。 バックアップおよび回復が可能なのはローカルでだけですが、十分なバックアップ・サポートを持っていないサーバーには、重要性の低いデータを配置するように計画できます。 バックアップおよび回復手順は、複数のアプリケーション・サーバー上に存在するデータに対し、一貫していなければなりません。ネットワークには複数のサーバーがあるので、そのようなデータを 2 番目のサーバーにも保管して、何らかの形式で常にネットワークで使えるようにしておくことができます。 こうした方針は、データベースをネットワーク内で分散する前に、特に計画および展開する必要があります。

第 3 章 iSeries 分散リレーショナル・データベースのための通信

この章では、分散リレーショナル・データベースでサポートされている通信に関する、以下のトピックについて説明します。

- 『DRDA 実装のための通信ツール』では、通信タイプや通信回線を含む、分散リレーショナル・データベースをサポートするために使用されるさまざまな通信ツールについて説明します。
- 分散リレーショナル・データベースの通信ネットワークに関する考慮事項では、データベース処理が分散リレーショナル・データベースに依存する場合に、通信ネットワークに関して考慮しなければならない事項について説明しています。
- 『分散リレーショナル・データベースに関する通信の構成』では、ネットワーク構成のステップを紹介します。

本書には、必要な情報すべてが含まれているとは限りません。本書の目的は、お客様が自らに適切な質問をして、ご自身で回答を下す手助けをすることであり、そうすることによって、業務上の要件に基づいたリソースを最大限に活用できるようになります。

DRDA 実装のための通信ツール

iSeries 上で DRDA を実装するための通信サポートが、 IBM システム・ネットワーク体系 (SNA) のもとで、 Advanced Peer-to-Peer Networking® (APPN) の有無には関係なく、拡張プログラム間通信 (APPC) プロトコルを介して提供されます。

V4R2 以降で、 1 フェーズ・コミットでの DRDA の TCP/IP サポートが使用可能になりました。 V5R1 で、 2 フェーズ・コミットでの完全な DUW サポートが使用可能になりました。

分散リレーショナル・データベースでのシステム・ネットワーク体系: SNA は、複数の論理装置 (LU) タイプで構成される体系です。これらの論理装置は、やはり同じ LU タイプをサポートするサーバー、制御装置、および端末との間で通信する方法の体系的な定義です。 iSeries server 上の分散リレーショナル・データベースに必要な SNA サポートはすべて、 OS/400 ライセンス・プログラムの一部になっています。

ほかの通信ツールの詳細については、以下のトピックを参照してください。

- 分散リレーショナル・データベースのための APPC/APPN
- DDM と分散リレーショナル・データベースの使用

この章の例と仕様は、SNA 構成とネイティブの TCP/IP だけに通用するものです。 TCP/IP を介した

APPC の詳細は、V5R1 Supplemental Manual Web サイトにある、 *「通信構成」*  を参照してください。システム TCP/IP サポートのセットアップに関する詳細は、『TCP/IP セットアップ』のトピックを参照してください。

分散リレーショナル・データベースのための APPC/APPN

APPC は、SNA LU 6.2 および物理装置 (PU) T2.1 アーキテクチャーを、 iSeries server で実現したものです。 これを使うと、異なる処理装置に存在しているアプリケーションは、相互に対等関係で通信およびデータ交換を行えます。

APPN サポートは、PU T2.1 アーキテクチャーに対する拡張機能で、次のようなネットワーク機能を提供するものです。

- 分散ディレクトリーの探索によってネットワーク内で LU を動的に見つけること。
- アプリケーションによるセッションの要求時における選択特性に基づいた LU への経路の動的選択。
- 他の LU 6.2 パートナー間のセッションのためのノードを経由する LU 6.2 セッション・トラフィックの中間経路指定。
- 伝送優先順位に基づいたセッション・データの経路指定。
- リモート・ロケーション・パートナー定義の動的作成および開始。
- 高性能経路指定 (HPR)。これは APPN アーキテクチャーに追加されるもので、特に高速リンクの使用時に、APPN 経路指定のパフォーマンスと信頼性を拡張します。

APPC および APPN は、次のような IBM 提供の機能もサポートします。

- SNA 配布サービス (SNADS)
- iSeries server に対する表示装置パススルー

iSeries APPN および HPR については、『APPC、APPN、および HPR』で説明されています。

DDM および分散リレーショナル・データベースの使用

iSeries server における DRDA の実装では、分散データ管理機能 (DDM) アーキテクチャー・コマンドを使用して、他のサーバーと通信します。ただし、分散リレーショナル・データベースと DDM のファイル・アクセス・サポートとでは、処理の仕方が異なる機能があります。

分散リレーショナル・データベース処理の使用では、アプリケーションは、ローカル・システム上のリレーショナル・データベース・ディレクトリーを使用して、リモートに接続されます。 リレーショナル・データベース・ディレクトリーは、リレーショナル・データベース名とそのデータベースまでの通信パスの間の、必要なリンクを提供します。分散リレーショナル・データベースのもとで実行されるアプリケーションでは、データベース名を識別して、処理に必要な SQL ステートメントを実行するだけです。

DDM サポートの使用では、ローカル・システム上の DDM ファイルによって、リモート・ファイルが識別され、通信パスが提供されます。 V5R2 から、RDB ディレクトリー項目への参照によって DDM ファイルも作成できるようになりました。

リモート・コマンドの投入、ファイルのコピー、ある場所から別の場所へのデータの移動など、管理作業の分散リレーショナル・データベース処理をサポートするために DDM を使用できます。 DDM サポートを使用するためには、DDM ファイルを作成しなければなりません。これについては、83 ページの『DDM ファイルのセットアップ』に説明してあります。

iSeries server のファイル・コピー・コマンドを用いる DDM ファイルの使用については、 89 ページの『コピー・ファイル・コマンドを使用した iSeries server 相互間のデータの移動』に説明してあります。

IP ネットワークを使っている場合、 DDM 関連操作のいくつかで使用可能な他の類似の機能があり、この節で説明されています。たとえば、FTP および リモート実行 (RUNRMTCMD) コマンドを使えます。

分散リレーショナル・データベースの通信ネットワークに関する考慮事項

分散リレーショナル・データベースでは、通信使用度が高くなるので、データベース処理が通信ネットワークに依存する場合に、通信ネットワークに関して考慮しなければならない事項があります。

分散リレーショナル・データベース処理に伴う使用度が高くなるので、ローカル・ロケーションとリモート・ロケーションの両方について作成される MODE 記述について、最大セッション数パラメーター (MAXSSN) および会話の最大数パラメーター (MAXCNV) を大きくすることができます。

MODE 記述によって容量を増やすだけでなく、分散リレーショナル・データベース処理用のネットワークのパフォーマンスを向上させるためには、ネットワーク内の種々の回線の回線速度を上げることや、高品質回線を選択することを考慮できます。

分散リレーショナル・データベース・ネットワークに関する考慮事項としては、その他にアクセス可能度や可用性の問題があります。特定のデータベースが日常の、あるいは随時の企業運営にとって重要であればあるほど、ユーザーがそのデータベースにアクセスできる方法を考慮する必要が増します。つまり、必要に応じたデータの可用性を提供するために、ネットワーク内のパスおよび代替パスを検討しなければなりません。この項目の詳細については、『第 7 章 分散リレーショナル・データベースのデータの可用性および保護』で説明してあります。

ネットワークのパフォーマンスに最も影響を及ぼすのは、回線速度と通信回線を構成する方法だと思われますが、転送される情報の性質についても、回線速度と使用のタイプの両方との関連で、問題をいくつか検討してみることが大切です。以下にその例を示します。

- 転送しなければならない情報の量はどのくらいか。
- バッチ・アプリケーションの場合の代表的なトランザクションおよび作業単位は何か。
- 回線を同時に使用するアプリケーション・プログラムまたはユーザーの数はどのぐらいか。
- 対話式アプリケーションの場合の代表的なトランザクションおよび作業単位は何か、ならびに各トランザクションで送信および受信するデータの量はどのくらいか。

分散リレーショナル・データベースに関する通信の構成

iSeries server 上で DRDA を実装するための通信サポートは、分散データ管理機能 (DDM) アーキテクチャーに基づいています。 このサポートには、拡張プログラム間通信 (APPC) による IBM システム・ネットワーク体系 (SNA) だけではなく、ネイティブの TCP/IP 接続も含まれます。拡張対等ネットワーキング機能 (APPN*) を伴う場合と伴わない場合がありますが、高性能経路指定 (HPR) を伴います。さらに、OS/400 では、APPC が提供されているので、 DDM および分散リレーショナル・データベースへは、AnyNet* サポートを使用した TCP/IP 経由でアクセスします。 AnyNet® は、TCP/IP 経由の DRDA リモート作業単位サポートには必要ありませんが、 TCP/IP 経由の分散作業単位機能には役立つ場合があります。

これらの機能に関する詳細については、以下のトピックを参照してください。これらのトピックでは、基本的な構成例を提供して、ネットワーク内のシステムの構成に必要なステップを例示します。

- APPC 用の通信ネットワークの構成
- TCP/IP 用の通信ネットワークの構成
- OptiConnect を介した通信の構成

注: リレーショナル・データベースにアクセスするために、アプリケーション・リクエスター・ドライバー出口プログラムが使用できる通信には、制約はありません。詳細については、 アプリケーション・リクエスター・ドライバー・プログラムを参照してください。

APPC 用の通信ネットワークの構成

分散リレーショナル・データベースのための通信を構成する場合には、ネットワーク内でローカル・システムおよびリモート・システムを定義する必要があります。ネットワーク内のシステムをいったん定義すれ

ば、分散データ管理機能 (DDM) 機能または SNA 配布サービス (SNADS) を使用してネットワーク全体に情報を配布し、ディスプレイ・パススルーを使用してローカル・サーバー上のワークステーションから**アプリケーション・サーバー (AS)** に接続し、分散リレーショナル・データベース・ネットワーク内のサーバーのリレーショナル・データベース・ディレクトリーをセットアップできます。リレーショナル・データベース・ディレクトリーは、通信構成値を、分散リレーショナル・データベース・ネットワーク内のリレーショナル・データベースの名前に関連付けるためのものです。リレーショナル・データベース・ディレクトリーのセットアップの詳細は、 『第 5 章 iSeries 分散リレーショナル・データベースのセットアップ』を参照してください。

ネットワーク内の各 iSeries server については、各サーバーがそれ自体とネットワーク内のリモート・サーバーを識別できるように、それぞれ定義する必要があります。 ネットワーク内のサーバーを定義するには、次のことを行わなければなりません。

1. ネットワーク属性の定義
2. 必要であれば、ネットワーク・インターフェースとネットワーク・サーバー記述の作成
3. 該当する回線記述の作成
4. 制御装置記述の作成
5. APPC 接続のためのサービス・クラス記述の作成
6. APPC 接続のモード記述の作成
7. 装置記述の自動または手作業による作成

分散リレーショナル・データベースのネットワーク属性の定義

ネットワーク属性を定義するには、ネットワーク属性変更 (CHGNETA) コマンドを使います。 ネットワーク属性には、ローカル・サーバー名、省略時のローカル・ロケーション名、省略時の制御点名、ローカル・ネットワーク識別コード、およびネットワーク・ノード・タイプが含まれます。マシンがエンド・ノードである場合には、属性には、この iSeries server で使われるネットワーク・サーバーの名前も含まれます。ネットワーク属性によって、サーバーが高性能経路指定 (HPR) を使うかどうかも決まります。 HPR ネットワークの計画および構成については、 iSeries Information Center の『APPC、APPN、および HPR』を参照してください。

分散リレーショナル・データベースのネットワーク・インターフェース記述の定義

ネットワーク・インターフェース記述を作成します。ネットワーク・インターフェースの作成には、『Create Network Interface (Frame-Relay Network) (CRTNWIFR)』を参照します。

分散リレーショナル・データベースの回線記述の定義

iSeries server とネットワークの間で使われる物理回線接続およびデータ・リンク・プロトコルを記述する、回線記述を作成します。回線記述の作成には、次のコマンドを使用してください。

- 回線記述の作成 (イーサネット) (CRTLINETH)
- 回線記述の作成 (DDI ネットワーク) (CRTLINDDI)
- 回線記述の作成 (フレーム・リレー・ネットワーク) (CRTLINFR)
- 回線記述の作成 (SDLC) (CRTLINSDLC)
- 回線記述の作成 (トークンリング・ネットワーク) (CRTLINTRN)
- 回線記述の作成 (無線) (CRTLINWLS)
- 回線記述の作成 (X.25) (CRTLINX25)

分散リレーショナル・データベースのための制御装置記述の定義

制御装置記述は、ネットワーク内の隣接するサーバーを記述します。 APPN サポートの使用は、制御装置記述の作成時に、APPN(*YES) を指定することによって示します。制御装置記述の作成には、次のコマンドを使います。

- 制御装置記述の作成 (APPC) (CRTCTLAPPC)
- 制御装置記述の作成 (SNA ホスト) (CRTCTLHOST)

トークンリング、イーサネット、無線、または DDI 回線記述の AUTOCRTCTL パラメーターが、 *YES にセットされている場合には、サーバーがその回線でセッション開始要求を受信すると、制御装置記述は自動的に作成されます。

AnyNet サポートを指定するには、 制御装置記述の作成 (APPC) (CRTCTLAPPC) コマンド の LINKTYPE パラメーターに *ANYNW を指定します。

分散リレーショナル・データベースの構成に関するその他の考慮事項

追加のローカル・ロケーション、または APPN 用の特殊な特性を持つリモート・ロケーションが必要な場合には、 APPN ロケーション・リストを作成しなければなりません。 1 つのローカル・ロケーション名は、ネットワーク属性に指定されている制御点名です。 iSeries server で追加のロケーションが必要である場合には、 APPN ローカル・ロケーション・リストが必要になります。 リモート・ロケーションの特殊な特性としては、リモート・ロケーションがローカル・ロケーションとは別のネットワーク内にあるのかどうか、そしてセキュリティー要件があります。特殊な特性のリモート・ロケーションが存在する場合、APPN リモート・ロケーション・リストが必要です。 APPN ロケーション・リストは、構成リスト作成 (CRTCFGL) コマンドで作成できます。 APPN 構成リストおよびセキュリティー要件の詳細は、 APPC ネットワークでの DRDA セキュリティーの要素を参照してください。

通信記述は、構成変更 (VRYCFG) コマンドまたは構成状況処理 (WRKCFGSTS) コマンドを使い、オンに構成変更する (活動化する) ことができます。 非交換回線記述がオンに構成変更される場合には、その回線に接続されている該当の制御装置および装置もオンに構成変更されます。 WRKCFGSTS コマンドを使用すると、各接続の状況も表示されます。通信構成状況の処理の詳細は、『第 6 章 分散リレーショナル・データベースの管理および操作の作業』を参照してください。

基本構成例の例示に役立つように、『例: 分散リレーショナル・データベースのための APPN 構成』に示してある、 Spiffy 社のネットワークを考察することにします。

注:

1. 制御装置記述は、 IBM ネットワーク制御プログラム/仮想記憶通信アクセス方式 (NCP/VTAM*) PU マクロに対応するものです。制御装置記述の中の情報は、Extended Services 通信管理機能パートナー LU プロファイルに入っています。
2. 装置記述は、NCP/VTAM 論理装置 (LU) マクロに相当します。装置記述の中の情報は、Extended Services 通信管理機能パートナー LU プロファイル、および LU プロファイルに入っています。
3. モード記述は、NCP/VTAM モード・テーブルに相当します。モード記述の中の情報は、Extended Services 通信管理機能伝送サービス・モード・プロファイル、および初期セッション限界値プロファイルに入っています。

ネットワーク・サポートの構成およびロケーション・リストの処理については、 V5R1 Supplemental Manual Web サイトにある、 *「通信構成」*  、および『APPC、APPN、および HPR』のトピックに記載されています。

例: 分散リレーショナル・データベースのための APPN 構成

基本構成例の例示に役立つように、次の例に示してある、 Spiffy 社のネットワークを考察することにします。

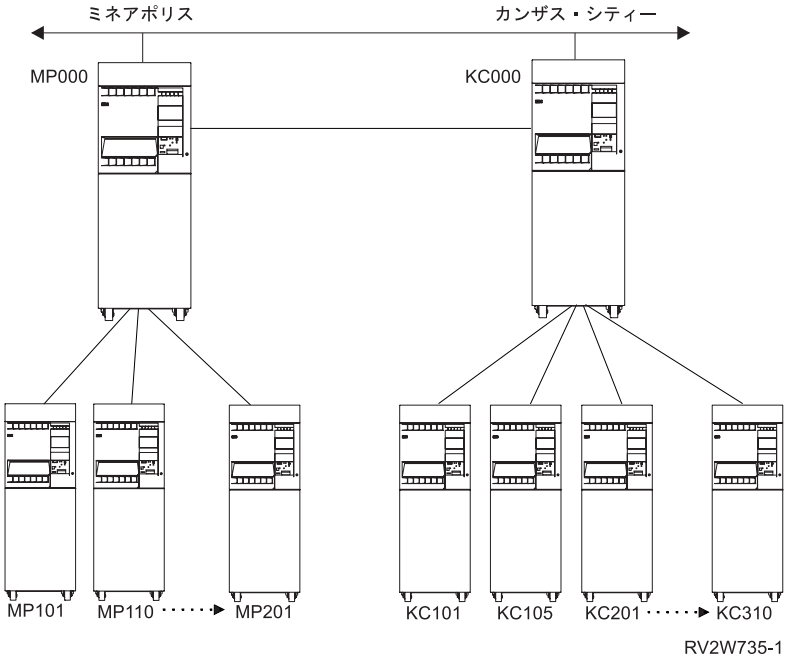


図 6. Spiffy 社のネットワーク編成

このネットワーク編成では、Spiffy 社の地域支社のうちの 2 つは、それぞれ MP000 および KC000 というネットワーク・ノード・サーバーです。 ミネアポリスの MP000 サーバーとカンザス・シティーの KC000 サーバーは、 SDLC 交換回線をバックアップ回線として備えた SDLC 非交換回線によって、相互に通信しています。 MP000 iSeries server は、KC000 サーバーおよびその他の地域ネットワーク・ノードの開発および問題処理センターとして機能しています。

次のプログラム例および説明では、ミネアポリスおよびカンザス・シティーの iSeries server を、ネットワーク内のネットワーク・ノードとして構成する方法を説明し、あわせて、ミネアポリスが担当域内販売店の 1 つに対してネットワークを構成する方法を示してあります。この例の目的は、図 6 に示してあるネットワークの構成に必要なタスクの一部に限って説明することにあるので、そのネットワークの完全な構成を示すものではありません。

ネットワーク・ノード MP000 の構成

次のプログラム例には、 MP000 (ネットワーク・ノード 1) として識別されるサーバーの構成を定義するために使用する、制御言語 (CL) コマンドを示してあります。 この例では、CL プログラムで使用するとおりにコマンドを示します。構成は、構成メニューを使って実行することも可能です。

注: xv ページの『コードの特記事項情報』に重要なリーガル情報がありますので、お読みください。

```
/*********************************************************************/
/*                                                                   */
/*  MODULE:  MP000                 LIBRARY:  PUBSCFGS                */
/*                                                                   */
/*  LANGUAGE:  CL                                                    */
/*                                                                   */
/*  FUNCTION:  CONFIGURES APPN NETWORK:                              */
/*                                                                   */
```

```
/*             THIS IS: MP000 TO KC000           (nonswitched)       */
/*                      MP000 TO KC000           (switched)          */
/*                      MP000 TO MP101 - MP299   (nonswitched)       */
/*                                                                   */
/*                                                                   */
/*                                                                   */
/*********************************************************************/
PGM
/* Change network attributes for MP000 */          1
CHGNETA    LCLNETID(APPN) LCLCPNAME(MP000) +
LCLLOCNAME(MP000) NODETYPE(*NETNODE)
/*********************************************************************/
/*                         MP000 to KC000   (nonswitched)            */
/*********************************************************************/
/* Create nonswitched line description for MP000 to KC000*/
CRTLINSDLC LIND(KC000L) RSRCNAME(LIN021)           2
/* Create controller description for MP000 to KC000     */
CRTCTLAPPC CTLD(KC000L) LINKTYPE(*SDLC)  +         3
LINE(KC000L) RMTNETID(APPN) +
RMTCPNAME(KC000) STNADR(01) +
NODETYPE(*NETNODE)
/*********************************************************************/
/*                         MP000 TO KC000 (switched)                 */
/*********************************************************************/
/* Create switched line description for MP000 to KC000  */
CRTLINSDLC LIND(KC000S) RSRCNAME(LIN022) +         4
CNN(*SWTPP) AUTOANS(*NO) STNADR(01)
/* Create controller description for MP000 to KC000     */
CRTCTLAPPC CTLD(KC000S) LINKTYPE(*SDLC) +          5
SWITCHED(*YES) SWTLINLST(KC000S) +
RMTNETID(APPN) RMTCPNAME(KC000) +
INLCNN(*DIAL) CNNNBR(8165551111 +
STNADR(01) TMSGRPNBR(3) NODETYPE(*NETNODE)

/*********************************************************************/

/*********************************************************************/
/*                         MP000 to MP101   (nonswitched)            */
/*********************************************************************/
/* Create nonswitched line description for MP000 to KC000*/
CRTLINSDLC LIND(MP101L) RSRCNAME(LIN031)           6
/* Create controller description for MP000 to MP101     */
CRTCTLAPPC CTLD(MP101L) LINKTYPE(*SDLC)  +
LINE(MP101L) RMTNETID(APPN) +
RMTCPNAME(MP101) STNADR(01) +
NODETYPE(*ENDNODE)
/*********************************************************************/
ENDPGM
```

1 **ネットワーク属性変更 (MP000)**

ネットワーク属性変更 (CHGNETA) コマンドを使用して、ネットワーク内のサーバーの属性を設定します。以下の属性が MP000 地域サーバーに関して定義され、これらの属性は、このネットワーク・ノードのネットワークの中のすべての接続に適用されます。

LCLNETID(APPN)
ローカル・ネットワークの名前は APPN です。リモート・サーバー (プログラム例の KC000) では、 制御装置記述の作成 (APPC) (CRTCTLAPPC) コマンド のリモート・ネットワーク識別コード (RMTNETID) として、この名前を指定しなければなりません。この例では、ネットワーク属性に省略時解釈されます。

LCLCPNAME(MP000)
ミネアポリス地域サーバーのローカル制御点に割り当てられる名前は MP000 です。リモー

ト・サーバーでは、 制御装置記述の作成 (APPC) (CRTCTLAPPC) コマンド のリモート制御点名 (RMTCPNAME) として、この名前を指定します。

LCLLOCNAME(MP000)
省略時のローカル・ロケーション名は MP000 です。この名前は、APPN サポートによって作成される装置記述で使用されます。

NODETYPE(*NETNODE)
ローカル・サーバー (MP000) は APPN ネットワーク・ノードです。

2 **回線記述の作成 (MP000 と KC000 間、非交換)**

この例で使われる回線は SDLC 非交換回線です。回線を作成するのに使われるコマンドは 回線記述の作成 (SDLC) (CRTLINSDLC) コマンド です。指定されるパラメーターには以下のものがあります。

LIND(KC000L)
回線記述に割り当てられる名前は KC000L です。

RSRCNAME(LIN021)
LIN021 という名前の物理通信ポートが定義されます。

3 **制御装置記述の作成 (MP000 と KC000 間、非交換)**

これは APPN 環境 (iSeries server と iSeries server 間) なので、制御装置は APPC 制御装置であり、 制御装置記述の作成 (APPC) (CRTCTLAPPC) コマンドを使用して、制御装置の属性を定義します。例で使用するコマンドで定義される属性には、以下のものがあります。

CTLD(KC000L)
制御装置記述に割り当てられる名前は KC000L です。

LINKTYPE(*SDLC)
この制御装置は SDLC 通信回線によって接続されるので、指定される値は *SDLC です。この値は、回線記述の作成コマンド (CRTLINxxx) によって定義されている回線のタイプに対応していなければなりません。

LINE(KC000L)
この制御装置が接続される回線記述の名前は KC000L です。この値は、回線記述の LIND パラメーターで指定されている名前と一致しなければなりません。

RMTNETID(APPN)
リモート制御点が存在しているネットワークの名前は APPN です。

RMTCPNAME(KC000)
リモート制御点名は KC000 です。ここで指定する名前は、リモート・サーバーでローカル制御点名に指定されている名前と一致しなければなりません。 例では、名前は、リモート・サーバー (KC000) で ネットワーク属性変更 (CHGNETA) コマンドの LCLCPNAME パラメーターによって指定されます。

STNADR(01)
リモート制御装置に割り当てられるアドレスは 16 進数の 01 です。

NODETYPE(*NETNODE)
リモート・サーバー (KC000) は APPN ネットワーク・ノードです。

4 **回線記述の作成 (MP000 と KC000 間、交換)**

この例で使われる回線は SDLC 交換回線です。回線を作成するのに使われるコマンドは 回線記述の作成 (SDLC) (CRTLINSDLC) コマンド です。指定されるパラメーターには以下のものがあります。

LIND(KC000S)
回線記述に割り当てられる名前は KC000S です。

RSRCNAME(LIN022)
LIN022 という名前の物理通信ポートが定義されます。

CNN(*SWTPP)
これは交換回線接続です。

AUTOANS(*NO)
このサーバーは着信呼び出しに自動応答しません。

STNADR(01)
ローカル・サーバーに割り当てられるアドレスは 16 進の 01 です。

5 制御装置記述の作成 (MP000 と KC000 間、交換)

これは APPN 環境 (iSeries server と iSeries server 間) なので、制御装置は APPC 制御装置であり、 制御装置記述の作成 (APPC) (CRTCTLAPPC) コマンドを使用して、制御装置の属性を定義します。例で使用するコマンドで定義される属性には、以下のものがあります。

CTLD(KC000S)
制御装置記述に割り当てられる名前は KC000S です。

LINKTYPE(*SDLC)
この制御装置は SDLC 通信回線によって接続されるので、指定される値は *SDLC です。この値は、回線記述の作成コマンド (CRTLINxxx) によって定義されている回線のタイプに対応していなければなりません。

SWITCHED(*YES)
この制御装置は交換 SDLC 回線に接続されます。

SWTLINLST(KC000S)
この制御装置を接続できる回線記述 (交換回線の) の名前は KC000S です。例では、1 回線 (KC000S) しかありません。この値は、回線記述の LIND パラメーターで指定されている名前と一致しなければなりません。

RMTNETID(APPN)
リモート制御点が存在しているネットワークの名前は APPN です。

RMTCPNAME(KC000)
リモート制御点名は KC000 です。ここで指定する名前は、リモート・サーバーでローカル制御点名に指定されている名前と一致しなければなりません。 例では、名前は、リモート・サーバーで ネットワーク属性変更 (CHGNETA) コマンドの LCLCPNAME パラメーターによって指定されます。

INLCNN(*DIAL)
初期接続は、着信呼び出しに応答するか、または呼び出しをかけるかいずれかの方法で、iSeries server によって行われます。

CNNNBR(8165551111)
リモート・カンザス・シティー制御装置の接続 (電話) 番号は 8165551111 です。

STNADR(01)
リモート・カンザス・シティー制御装置に割り当てられるアドレスは 16 進数の 01 です。

TMSGRPNBR(3)
リモート・サーバーとの伝送グループ協定のために、値 (3) が APPN サポートによって使用されます。

リモート・サーバーでは、伝送グループに同じ値を指定しなければなりません。

NODETYPE(*NETNODE)
リモート・サーバー (KC000) は APPN ネットワーク・ノードです。

6 **MP101 への回線および制御装置の作成**

例のこの部分では、 MP000 と MP101 (販売店エンド・ノード) 間の回線および制御装置の構成を示しています。これに類似する構成は、MP000 と各販売店エンド・ノードとの間でも行わなければなりません。さらに、ミネアポリスでの構成を完了するためには、販売店のそれぞれもこれに類似する構成コマンドまたはプログラムを使用して、それぞれの通信相手先となる各サーバーごとに回線および制御装置を作成しなければなりません。

30 ページの図 6 に示してあるネットワーク構成を完了するためには、同じように、KC000 サーバーについても、その各販売店エンド・ノードに対して構成を行い、各エンド・ノードでは、KC000 サーバーとの間で通信するための回線および制御装置を構成しなければなりません。

これらの接続は、例には示してありません。

ネットワーク・ノード KC000 の構成

次のプログラム例では、KC000 として識別される地域サーバーの構成を定義するときに使う、 CL コマンドが示されています。 この例では、CL プログラムで使用するとおりにコマンドを示します。構成は、構成メニューを使って実行することも可能です。

注: xv ページの『コードの特記事項情報』に重要なリーガル情報がありますので、お読みください。

```
/*********************************************************************/
/*                                                                   */
/*  MODULE:  KC000                     LIBRARY:  PUBSCFGS            */
/*                                                                   */
/*  LANGUAGE:  CL                                                    */
/*                                                                   */
/*  FUNCTION:  CONFIGURES APPN NETWORK:                              */
/*                                                                   */
/*             THIS IS: KC000 TO MP000     (nonswitched)             */
/*                      KC000 TO MP000     (switched)                */
/*                                                                   */
/*                                                                   */
/*********************************************************************/
PGM
/* Change network attributes for KC000                 */
CHGNETA    LCLNETID(APPN) LCLCPNAME(KC000) +      7
LCLLOCNAME(KC000) NODETYPE(*NETNODE)
/*********************************************************************/
/*                           KC000 TO MP000  (nonswitched)           */
/*********************************************************************/
/* Create line description for KC000 to MP000          */
CRTLINSDLC LIND(MP000L) RSRCNAME(LIN022)          8
/* Create controller description for KC000 to MP000    */
CRTCTLAPPC CTLD(MP000) LINKTYPE(*SDLC) +          9
LINE(MP000L) RMTNETID(APPN) +
RMTCPNAME(MP000) STNADR(01) +
NODETYPE(*NETNODE)
/*********************************************************************/
/*                         KC000 TO MP000   (switched)               */
```

```
/*********************************************************************/
/* Create switched line description for KC000 to MP000S */
CRTLINSDLC LIND(MP000S) RSRCNAME(LIN031) +        10
CNN(*SWTPP) AUTOANS(*NO) STNADR(01)
/* Create controller description for KC000 to MP000     */
CRTCTLAPPC CTLD(MP000S) LINKTYPE(*SDLC) +         11
SWITCHED(*YES) SWTLINLST(MP000S) +
RMTNETID(APPN) RMTCPNAME(MP000) +
INLCNN(*ANS) CNNNBR(6125551111) +
STNADR(01) TMSGRPNBR(3) NODETYPE(*NETNODE)
ENDPGM
```

7 **ネットワーク属性変更 (KC000)**

ネットワーク属性変更 (CHGNETA) コマンドを使用して、ネットワーク内のサーバーの属性を設定します。 次の属性が KC000 という地域サーバーに関して定義され、これらの属性は、このネットワーク・ノードのネットワークの中のすべての接続に適用されます。

LCLNETID(APPN)
ローカル・ネットワークの名前は APPN です。リモート・サーバー (この例では、ミネアポリス・ネットワーク・ノード) では、 制御装置記述の作成 (APPC) (CRTCTLAPPC) コマンド のリモート・ネットワーク識別コード (RMTNETID) として、この名前を指定しなければなりません。

LCLCPNAME(KC000)
ローカル制御点に割り当てられる名前は KC000 です。リモート・サーバーでは、 制御装置記述の作成 (APPC) (CRTCTLAPPC) コマンドのリモート制御点名 (RMTCPNAME) として、この名前を指定します。

LCLLOCNAME(KC000)
省略時のローカル・ロケーション名は KC000 です。この名前は、APPN サポートによって作成される装置記述で使用されます。

NODETYPE(*NETNODE)
ローカル・サーバー (KC000) は APPN ネットワーク・ノードです。

8 **回線記述の作成 (KC000 と MP000 間、非交換)**

この例で使われる回線は SDLC 非交換回線です。回線を作成するのに使われるコマンドは回線記述の作成 (SDLC) (CRTLINSDLC) コマンド です。指定されるパラメーターには以下のものがあります。

LIND(MP000L)
回線記述に割り当てられる名前は MP000L です。

RSRCNAME(LIN022)
LIN022 という名前の物理通信ポートが定義されます。

9 **制御装置記述の作成 (KC000 と MP000 間、非交換)**

これは APPN 環境 (iSeries server と iSeries server 間) なので、制御装置は APPC 制御装置であり、 制御装置記述の作成 (APPC) (CRTCTLAPPC) コマンドを使用して、制御装置の属性を定義します。例で使用するコマンドで定義される属性には、以下のものがあります。

CTLD(MP000L)
制御装置記述に割り当てられる名前は MP000L です。

LINKTYPE(*SDLC)
この制御装置は SDLC 通信回線によって接続されるので、指定される値は *SDLC です。この値は、回線記述の作成コマンド (CRTLINxxx) によって定義されている回線のタイプに対応していなければなりません。

LINE(MP000L)
この制御装置が接続される回線記述の名前は MP000L です。この値は、回線記述の LIND パラメーターで指定されている名前と一致しなければなりません。

RMTNETID(APPN)
リモート・サーバーが存在しているネットワークの名前は APPN です。

RMTCPNAME(MP000)
リモート制御点名は MP000 です。ここで指定する名前は、リモート・サーバーでローカル制御点名に指定されている名前と一致しなければなりません。 例では、名前は、ミネアポリス地域リモート・サーバー (MP000) で、 ネットワーク属性変更 (CHGNETA) コマンドの LCLCPNAME パラメーターによって指定されます。

STNADR(01)
リモート制御装置に割り当てられるアドレスは 16 進数の 01 です。

NODETYPE(*NETNODE)
リモート・サーバー (MP000) は APPN ネットワーク・ノードです。

10 回線記述の作成 (KC000 と MP000 間、交換)

この例で使われる回線は SDLC 交換回線です。回線を作成するのに使われるコマンドは 回線記述の作成 (SDLC) (CRTLINSDLC) コマンド です。指定されるパラメーターには以下のものがあります。

LIND(MP000S)
回線記述に割り当てられる名前は MP000S です。

RSRCNAME(LIN031)
LIN031 という名前の物理通信ポートが定義されます。

CNN(*SWTPP)
これは交換回線接続です。

AUTOANS(*NO)
このサーバーは着信呼び出しに自動応答しません。

STNADR(01)
ローカル・サーバーに割り当てられるアドレスは 16 進の 01 です。

11 制御装置記述の作成 (KC000 と MP000 間、交換)

これは APPN 環境 (iSeries server と iSeries server 間) なので、制御装置は APPC 制御装置であり、 制御装置記述の作成 (APPC) (CRTCTLAPPC) コマンドを使用して、制御装置の属性を定義します。例で使用するコマンドで定義される属性には、以下のものがあります。

CTLD(MP000S)
制御装置記述に割り当てられる名前は MP000S です。

LINKTYPE(*SDLC)
この制御装置は SDLC 通信回線によって接続されるので、指定される値は *SDLC です。この値は、回線記述の作成コマンド (CRTLINxxx) によって定義されている回線のタイプに対応していなければなりません。

SWITCHED(*YES)
この制御装置は交換 SDLC 回線に接続されます。

SWTLINLST(MP000S)
この制御装置を接続できる回線記述 (交換回線の) の名前は MP000S です。例では、1 回線 (MP000) しかありません。この値は、回線記述の LIND パラメーターで指定されている名前と一致しなければなりません。

RMTNETID(APPN)
リモート制御点が存在しているネットワークの名前は APPN です。

RMTCPNAME(MP000)
リモート制御点名は MP000 です。ここで指定する名前は、リモート地域サーバーでローカル制御点名に指定されている名前と一致しなければなりません。 例では、名前は、リモート・ミネアポリス地域サーバー (MP000) で、 ネットワーク属性変更 (CHGNETA) コマンドの LCLCPNAME パラメーターによって指定されます。

INLCNN(*ANS)
初期接続は、着信呼び出しに応答する方法で、iSeries server によって行われます。

CNNNBR(6125551111)
リモート・ミネアポリス制御装置の接続 (電話) 番号は 6125551111 です。

STNADR(01)
リモート・ミネアポリス制御装置に割り当てられるアドレスは 16 進数の 01 です。

TMSGRPNBR(3)
リモート・サーバーとの伝送グループ協定のために、値 (3) が APPN サポートによって使用されます。 リモート・サーバーでは、伝送グループに同じ値を指定しなければなりません。

NODETYPE(*NETNODE)
リモート・サーバー (MP000) は APPN ネットワーク・ノードです。

TCP/IP 用の通信ネットワークの構成

iSeries ナビゲーターを使い、TCP/IP ネットワークを迅速かつ簡単にセットアップすることができます。あるいは、次に示されている上級ステップを使い、 TCP/IP ネットワークをセットアップすることもできます。詳細については、 iSeries Information Center の『TCP/IP の構成』を参照してください。

1. 使用する iSeries server をローカル・ネットワーク (iSeries server を直接に接続するネットワーク) へ通知します。
   a. 回線記述が存在しているかどうかを判別します。
   b. 回線記述が存在していなければ、作成します。
   c. TCP/IP インターフェースを定義して iSeries server に IP アドレスを割り当てます。
2. TCP/IP 経路を定義します。これを定義しておくと、iSeries server はリモート TCP/IP ネットワーク (すなわち、 iSeries server を直接に接続しないネットワーク) 上のサーバーと通信できます。
3. ローカル定義域名およびホスト名を定義します。これにより、それぞれのサーバーに名前が割り当てられます。
4. ネットワーク内でのサーバー名を識別します。
   a. ローカル・ホスト・テーブルを作成します。
   b. リモート・ネーム・サーバーを識別します。
5. TCP/IP を開始します。

6. TCP/IP の動作を検査します。

OptiConnect を介した通信の構成

以下の 3 つの方式のいずれかで、OptiConnect を介した DRDA を使用できます。

- RDB ディレクトリー項目内の装置として QYCTSOC を指定します。この方式では、分散作業単位はサポートされません。
- OptiConnect 制御装置および装置をセットアップし、 RDB ディレクトリー項目でそれらを構成します。この方式には、 OptiConnect を終了する前に、制御装置をオフに変更しなければならないという欠点があります。
- OptiConnect を介した IP を構成し、 IP を介して実行するように RDB ディレクトリー項目をセットアップします。作成およびオンに変更する制御装置はありません。 IP インターフェースを自動開始に設定して、 OptiConnect の開始時にアクティブにできます。また、 LAN アダプターの IP アドレスを、関連するローカル・アドレスとして指定する、Point-to-Point インターフェースもセットアップできます。これは、OptiConnect が機能しているときには自動的に OptiConnect を介してトラフィックを送り、OptiConnect が機能していないときには LAN を介して送ります。 OptiConnect のセットアップについては、 *「OptiConnect for OS/400」*  を参照してください。

第 4 章 iSeries 分散リレーショナル・データベースのセキュリティー

iSeries server では、セキュリティーの要素がオペレーティング・システムに組み込まれていて、アプリケーション・サーバーのデータ・リソースに対するアクセスを制限しています。セキュリティー・オプションの範囲は、単純な物理的セキュリティーから、コマンドおよびデータ・オブジェクトに対する許可と組み合わされた、完全なパスワード・セキュリティーにまで及んでいます。ユーザーは、データベースがローカルであれリモートであれ、データベースにアクセスするための適正な許可を与えられていなければなりません。ユーザーはまた、アプリケーション・プログラムを実行するのに必要な、コレクション、テーブルなどのリレーショナル・データベース・オブジェクトに対する適正な許可も持っていなければなりません。つまり、分散データベースのユーザーは、ネットワーク内で使うデータベースについての、有効なユーザー・プロファイルを持っていなければならないということです。セキュリティー計画で、ネットワーク内におけるユーザーおよびアプリケーション・プログラムの要件を考慮しなければなりません。

セキュリティー機能の詳細については、『分散リレーショナル・データベースのセキュリティーの要素』を参照してください。

分散リレーショナル・データベース管理担当者は、次の 2 つのセキュリティー問題に対処しなければなりません。

- システム間保護
- リモート・サイトのユーザーの識別

2 つ以上のシステムがセットアップされていて、相互のデータベースにアクセスする場合には、通信回線の向こう側が確かに通信先として意図したロケーションであり、侵入者ではないことを確認することが大切です。 DRDA でリモート・リレーショナル・データベースにアクセスするとき、 iSeries server で拡張プログラム間通信 (APPC) およびAdvanced Peer-to-Peer Networking (APPN) の通信構成機能を使用する場合は、この必要なネットワーク・レベル・セキュリティーを行うオプションが用意されています。

分散リレーショナル・データベース管理担当者にとって 2 番目に重要になるのは、データ・セキュリティーが、データを保管しているシステムによって維持されるということです。分散リレーショナル・データベースでは、データベースがローカルであるかリモートであるかにかかわらず、データベースにアクセスするには、ユーザーは適正な権限を (システムのセキュリティー・レベルに応じて) 認可されていなければなりません。分散リレーショナル・データベースのネットワーク・ユーザーは、**アプリケーション・サーバー (AS)** でどんなジョブを実行する場合でも、 AS でユーザー ID によって正しく識別されなければなりません。 APPC/APPN および TCP/IP 通信プロトコルの両方を使用する DRDA サポートでは、接続要求とともにユーザー ID とパスワードの送信を行います。

この章では、リモート・リレーショナル・データベースに対する通信および DRDA アクセスに関連するセキュリティーについて説明します。そして、APPC ネットワーク接続の会話レベルのセキュリティーと、それに対応する DRDA アプリケーションによって開始される TCP/IP 接続のセキュリティーのレベルの相違点を説明します。これから行われるセキュリティーの説明では、ユーザー という用語は、通信ジョブを開始しているリモート・ユーザーを含みます。

それぞれの必要に適したセキュリティーに関する提案戦略は、『分散リレーショナル・データベースでの保護の方針』を参照してください。

一般的な iSeries のセキュリティーの概念については、 iSeries Information Center にある『システム・セキュリティーの手引き』をご覧ください。

分散リレーショナル・データベースのセキュリティーの要素

分散リレーショナル・データベース管理担当者は、ネットワーク内の **アプリケーション・リクエスター (AR)** によるデータのアクセスを不必要に制限せずに、ネットワーク内のアプリケーション・サーバーのリソースを保護する必要があります。

AR は、分散リレーショナル・データベース・プログラムに対して許可されたユーザーだけがアクセスできるようにオブジェクトとリレーショナル・データベースを保護します。これは、通常の iSeries server オブジェクト許可を使用してユーザーを識別してから、各ユーザー (またはユーザー・グループ) がオブジェクトに対して何を行うことができるか指定することによって可能になります。あるいは、SQL GRANT と REVOKE ステートメントを使用してテーブル、ビュー、および SQL パッケージに対する権限を許可したり取り消したりできます。 AR 上の SQL オブジェクトに対して権限レベルを提供すると、別のシステムのデータをアクセスする SQL アプリケーションに対して許可ユーザーだけがアクセスできるようにすることができます。

アプリケーション・サーバー (AS) で使用されているシステム・セキュリティー・レベルにより、 AR からの要求が受け入れられるかどうか、およびリモート・ユーザーが AS 上のオブジェクトに対して権限を認可されているかどうかが決まります。

分散リレーショナル・データベース・ネットワーク内の iSeries server に関するセキュリティー計画には、以下に挙げるような側面があります。

- 機密のテーブル、プログラム、およびパッケージなど、特定のリソースに対するユーザーのアクセスを規制するオブジェクト関連セキュリティー。
- ネットワーク内の正当な他システムであることを検査するロケーション・セキュリティー。
- ローカル・システムおよびリモート・システム上の正当なユーザーであること、およびその権限を検査するためのユーザー関連セキュリティー。
- 機密のテーブル、プログラム、およびパッケージなど、特定のリソースに対するユーザーのアクセスを規制するオブジェクト関連セキュリティー。
- 回線記述で構成することができ、経路選択プロセスで使用される、システム、モデム、通信回線、および端末を囲むドアの施錠や建物の警備保障など、物理的なセキュリティー。

ロケーション・セキュリティー、ユーザー関連セキュリティー、およびオブジェクト関連セキュリティーが使用できるのは、システム・セキュリティー・レベルが 20 またはそれ以上のレベルにセットされている場合だけです。

APPC 会話では、システムがレベル 10 のセキュリティーを使用している場合、無保護システムとして iSeries server はネットワークに接続します。 サーバーはセッション確立時にリモート・システムの身元を妥当性検査することはなく、着信するプログラム開始要求では会話セキュリティーを必要としません。 レベル 10 では、APPC リモート・ロケーションで構成されたセキュリティー情報は無視され、セッションまたは会話の確立時には使用されません。サーバー上でユーザー・プロファイルが存在しない場合には、それは作成されます。

システムがレベル 20 以上のセキュリティーを使用している場合、保護システムとして iSeries server はネットワークに接続します。 このようにして、iSeries システムでは、会話レベル・セキュリティー機能を実現でき、APPC の場合にはセッション・レベル・セキュリティーも実現できます。

ネットワーク内のシステムで同じレベルのシステム・セキュリティーを設定すると、セキュリティーの管理担当者の作業はより容易なものになります。 AS は、AR で予期されているものを指定してセッションを確立することによって、セッションおよび会話が確立できるかどうかを制御します。たとえば、AR のセキュリティー・レベルが 10 に設定されており、 AS のセキュリティー・レベルが 10 よりも大きい場合には、適切な情報が送信されない場合があり、これらシステムの中の 1 つのセキュリティー要素を変更しなければセッションは確立されないかもしれません。

DRDA アクセス用のパスワード

リモート・ユーザーにデータベース・アクセスを許可するための最も一般的な方法は、接続時にユーザー ID とパスワードを渡すことです。このためにアプリケーション・プログラマーが使用できる 1 つの方法は、組み込み SQL CONNECT ステートメントに USER/USING 文節をコーディングすることです。以下にその例を示します。

```
EXEC SQL CONNECT TO :locn USER :userid USING :pw
```

リモート・リレーショナル・データベースでの DRDA アクセスでは、会話がいったん確立されると、パスワードを再度入力する必要はありません。 RUW 接続管理方式で実行している際に RELEASE、DISCONNECT、または CONNECT ステートメントのいずれかを使用して接続を終了すると、最初の**アプリケーション・サーバー (AS)** との会話は除去される場合もあり、除去されない場合もあります。これは、接続している AS の種類と**アプリケーション・リクエスター (AR)** のジョブ属性にかかっています (特定の規則については、『DDM 会話の制御』を参照してください)。最初の AS との会話が除去されていない場合には、 2 番目の AS と接続している間は未使用のまま残されます。最初の AS に再度接続した際に会話が使用されていない場合、ユーザー ID とパスワードを入力しなくても会話は再度活動状態になります。この会話を 2 度目に使用するときにも、パスワードが再度検査されることはありません。

特定のセキュリティー・システムについての詳細は、以下のトピックを参照してください。

- APPC ネットワークでの DRDA セキュリティーの要素
- APPC ネットワークでの DRDA アプリケーション・サーバー (AS) セキュリティー
- TCP/IP を使用した DDM/DRDA セキュリティーの要素
- DRDA サーバー・アクセス制御出口プログラム
- DRDA 用のオブジェクト関連のセキュリティー
- 分散リレーショナル・データベース・オブジェクトの権限
- 分散リレーショナル・データベースの借用権限のもとで実行されるプログラム

セキュリティー・レベルについては、iSeries Information Center の『セキュリティー』および『APPC、APPN、および HPR』のセキュリティーの考慮事項についてのトピックをご覧ください。

APPC ネットワークでの DRDA セキュリティーの要素

DRDA を使用する場合、DRDA 環境での各サーバーのデータ・リソースを保護する必要があります。このことは、以下のパラメーターで制御される、3 つのグループのセキュリティー要素を使用して行います。

- システム関連のセキュリティーまたはセッションの場合、ソース・システムとターゲット・システムの間で APPC 通信セッションが最初に確立される場合には、それらのシステム間で交換されるシステム検証パスワードを示すために、各 iSeries サーバーで **LOCPWD パラメーター**が使用されます。どちらのシステムも、セッションを開始する前に同じパスワードを交換する必要があります。 (システム/36 では、このパスワードのことをロケーション・パスワードといいます。) APPC ネットワークでは、装置記述の作成 (APPC) (CRTDEVAPPC) コマンドの LOCPWD パラメーターは、このパスワードを指定します。装置は APPN を使用して自動的に作成され、リモート・ロケーション・リストのロケーション・パ

スワードは、身元を検査するために 2 つのロケーションで使用するパスワードを指定します。 構成リスト作成 (CRTCFGL) コマンドを使用して、リモート・ロケーション・リストのタイプ (*APPNRMT) を作成します。

- ユーザー関連またはロケーション・セキュリティーの場合、すでにソース・サーバーによってセキュリティーが確認された着信アクセス要求を (ターゲット・サーバーとして) 受け入れるかどうか、あるいはユーザー ID と暗号化されたパスワードが必要かどうかを示すため、各 iSeries サーバーで **SECURELOC パラメーター**が使用されます。 APPC ネットワークでは、装置記述の作成 (APPC) (CRTDEVAPPC) コマンドの SECURELOC パラメーターによって、ローカル・サーバーがリモート・サーバーによるセキュリティーの検査を許可するかどうかを指定します。装置は APPN を使用して自動的に作成され、APPN リモート構成リストのロケーション保護を使用して、ローカル・サーバーがリモート・サーバーによるユーザー・セキュリティー情報の検査を許可するかどうかが決定されます。SECURELOC 値は、リモート・ロケーションごとに、個別に指定できます。

  SECURELOC パラメーターは、以下のセキュリティー要素と共に使用されます。
  - ソース・サーバーによって送信されたユーザー ID (このパラメーターで許可されている場合)。
  - ユーザー ID と暗号化されたパスワード (このパラメーターで許可されている場合)。
  - 省略時ユーザー・プロファイルを含む、ターゲット・サーバーのユーザー・プロファイル。

  詳細は、『DDM source system security in an APPC network』のトピックを参照してください。
- オブジェクト関連のセキュリティーの場合、他のサーバーによって iSeries サーバーのファイルに完全にアクセスできるかどうか、そしてアクセスできる場合には、どのレベルのセキュリティーで着信要求を検査するかを示すため、 ネットワーク属性変更 (CHGNETA) コマンドで **DDMACC パラメーター**が使用されます。 このオブジェクト関連パラメーターの詳細は、 iSeries Information Center の『DDM Network Attribute (DDMACC Parameter)』のトピックにあります。
  - DDMACC パラメーター上で *REJECT が指定されている場合、ターゲット iSeries サーバーで受信される DRDA 要求はすべて拒否されます。
  - DDMACC パラメーターに *OBJAUT が指定されている場合には、通常のオブジェクト・レベルのセキュリティーがターゲット・サーバー上で使用されます。
  - DDMACC パラメーターでユーザー提供の任意のユーザー出口プログラム (またはアクセス制御プログラム) の名前が指定されると、追加のセキュリティーのレベルが使用されます。特定ソース・サーバーの指定ユーザーが、ターゲット・サーバーの特定のファイルを (規則的に) アクセスするのに特定のコマンドを使用できるかどうかを制御するために、ユーザー出口プログラムを使用することができます。 (詳細は、『DDM server access control exit program for additional security』を参照してください。)
  - DRDA を使用してターゲット・サーバー上でファイルを作成する場合、指定されるライブラリー名にはそのファイルが含まれます。 DRDA 要求でライブラリー名が指定されない場合、現行ライブラリー (*CURLIB) が使用されます。ファイル権限は、ファイルを作成したユーザーかターゲット・サーバーのセキュリティーの責任担当者だけがファイルにアクセスできる、という省略時値をとります。

リモート・ファイル・アクセスを制限するためのほとんどのセキュリティー管理は、ターゲット・サーバーによって扱われます。ソース・サーバーによって指定されるユーザー ID を除き、これらのすべての要素がターゲット・サーバー上で指定されて使用されます。しかし、ソース・サーバーでは、ソース・サーバー上の access to the DRDA ファイルへのアクセスを制御し、必要な場合にユーザー ID をターゲット・サーバーに送信することにより、ターゲット・サーバー・ファイルへのアクセスも制限します。

APPN 構成リスト

APPC ネットワークでは、2 つのロケーションが相互間で終端間 (エンドツーエンド) セッションを持つ場合、ロケーション・パスワードが指定されます。 中間ノードであるロケーションでは、ロケーション・パスワードを指定する必要はありません。

リモート・ロケーション・リストは構成リスト作成 (CRTCFGL) コマンドで作成され、これには、すべてのリモート・ロケーション、それぞれのロケーション・パスワード、およびリモート・ロケーションが保護されているかどうかを示すリストが含まれています。 iSeries サーバーには、システム全体のリモート・ロケーション構成リストが 1 つ存在します。中央側の iSeries サーバーは、制御言語 (CL) プログラムを送ることによって、リモート iSeries サーバーのロケーション・リストを作成できます。

構成リスト変更 (CHGCFGL) コマンドを使用して、リモート構成リストに変更を加えることができますが、変更が有効になるのは、そのロケーションのすべての装置がいずれもオフに構成変更された状態になってからです。

構成リスト表示 (DSPCFGL) コマンドが使用されたとき、パスワードの存在を示す表示はありません。 パスワードが入力された場合には、 構成リスト変更 (CHGCFGL) コマンドは *PASSWORD をフィールドに入れることによってパスワードの存在を示します。パスワードを表示する方法はありません。ロケーションのセキュリティーを設定する際に問題に直面する場合には、両方のシステムにパスワードを再度入力して、パスワードが一致することを確認する必要があります。

構成リストについては、 Information Center の『APPC、APPN、および HPR』をご覧ください。

会話レベル・セキュリティー

システム・ネットワーク体系 (SNA) 論理装置 (LU) 6.2 の体系では、異なったシステムのネットワークで一貫した会話セキュリティーを提供するために、 SNA ネットワークのさまざまなタイプのシステムが使用できる 3 つの会話セキュリティーの指定方法を示しています。 SNA セキュリティーのレベルには、以下のものがあります。

SECURITY(NONE)
通信を確立するのに、ユーザー ID もパスワードも送信されない。

SECURITY(SAME)
ローカル・サーバーと同じユーザー ID でユーザーをリモート・サーバーに署名させる。

SECURITY(PGM)
通信のために、ユーザー ID とパスワードの両方が送信される。

SECURITY(PROGRAM_STRONG)
パスワードが明白な形で送信されていない場合のみ、通信のためにユーザー ID とパスワードの両方が送信され、そうでない場合には、エラーが報告される。これは OS/400 の DRDA ではサポートされていません。

iSeries サーバーは、会話セキュリティーの 4 つの SNA レベルをすべてサポートしますが、 DRDA は最初の 3 つのレベルのみを使用します。 ターゲットでは、会話で使用される SNA 会話レベルを制御します。

SECURITY(NONE) レベルでは、ターゲットはユーザー ID またはパスワードを要求しません。ターゲット上の省略時のユーザー・プロファイルを使用して会話することが許可されています。会話で省略時のユーザー・プロファイルが使用できるかどうかは、 通信項目追加 (ADDCMNE) コマンドの DFTUSR パラメーターまたは当該サブシステムの通信項目変更 (CHGCMNE) コマンドで指定される値に依存しています。DFTUSR パラメーターの値が *NONE の場合は、**アプリケーション・サーバー (AS)** では、ターゲット上

の省略時のユーザー・プロファイルを使用する会話が許可されていないことを示しています。 SECURITY (NONE) は、パスワードとユーザー ID が提供されておらず、ターゲットで SECURELOC(*NO) が指定されているときに送信されます。

SECURITY(SAME) レベルでは、リモート・サーバーの SECURELOC 値によって、どんなセキュリティー情報が送信されるかが決まります (リモート・サーバーが iSeries であることを想定)。 SECURELOC 値が *NONE である場合、あたかも SECURITY(NONE) が要求されたかのように、ユーザー ID とパスワードは送信されません。 SECURITY(NONE) の処理方法については、前述の段落を参照してください。 SECURELOC 値が *YES である場合、ユーザー・プロファイルの名前が抽出され、ローカル・サーバーによってパスワードがすでに検査されたという旨の指示とともに送信されます。 SECURELOC 値が *VFYENCPWD である場合、ユーザー・プロファイルとそれに関連したパスワードは、パスワードが暗号化されてその値が秘密にされた後にリモート・サーバーに送信されます。したがって、DRDA を使用するには、ユーザーは両方のサーバーで同一のユーザー・プロファイル名とパスワードを持つ必要があります。

注: これら 3 つのオプションの中で SECURELOC(*VFYENCPWD) は一番安全なものであると言えます。なぜなら、SECURELOC(*VFYENCPWD) では最も多くの情報がリモート・サーバーによって検査されるからです。しかし、このオプションでは、ユーザーが複数のサーバー上で同一のパスワードを持つ必要があります。この場合、ユーザーが 1 つのサーバーのパスワードを変更しても、他のサーバーのパスワードを変更しないと問題が生じる可能性があります。

SECURITY(PGM) レベルでは、会話を行うために、ソースからのユーザー ID とパスワードの両方がターゲットによって要求されます。パスワードは会話が確立される際に検査され、それ以降の会話では無視されます。

APPC ネットワークでの DRDA アプリケーション・サーバー (AS) セキュリティー

ターゲット・サーバーが複数の要素を同時に使用する iSeries サーバーである場合、リモート・ファイルへアクセスする要求が許可されているかどうかを判別します。

ユーザー関連セキュリティー要素: ターゲット・サーバーの SECURELOC パラメーター、ソース・サーバーによって送信されるユーザー ID (許可される場合)、ソース・サーバーによって送信されるユーザー ID のパスワード、およびユーザー・プロファイルかターゲット・サーバーでの省略時ユーザー・プロファイル。

オブジェクト関連セキュリティー要素: DDMACC パラメーター、そして任意で、通常のオブジェクト権限制御を補足するための、ユーザー指定のユーザー出口プログラム。

ターゲット・セキュリティーのユーザー関連要素

分散リレーショナル・データベースの作業を処理するには、有効なユーザー・プロファイルが**アプリケーション・サーバー (AS)** 上に存在する必要があります。 iSeries server 上で通信ジョブを処理するサブシステムに、省略時のユーザー・プロファイルを指定することができます。 AS 上で通信項目追加 (ADDCMNE) コマンドの DFTUSR パラメーターに、省略時のユーザー・プロファイル名を指定できます。 ADDCMNE コマンドは、通信ジョブで使用されるサブシステム記述に会話項目を追加します。

省略時のユーザー・プロファイルが通信サブシステムで指定された場合、 AS が保護ロケーションであるかどうかによって、この要求に省略時のユーザー・プロファイルを使用するかどうかが決まります。 装置記述の作成 (APPC) (CRTDEVAPPC) コマンド上の SECURELOC パラメーター、または APPN リモート・ロケーション・リスト上での保護ロケーションの指定によって、 AS が保護ロケーションかどうかを指定します。

- SECURELOC または AS 上の保護ロケーションで *YES が指定されると、 AS は**アプリケーション・リクエスター (AR)** が保護ロケーションであると見なします。 AR からの要求とともにユーザー ID および検査済み標識が要求されます。リクエスターによって送信されるユーザー ID と一致するユーザー・プロファイルが AS 上に存在する場合、要求は許可されます。そうでない場合には、要求は拒否されます。
- AS 上の SECURELOC パラメーターで *NO が指定されると、 AS は AR を保護ロケーションであると見なしません。 AR はそれでもユーザー ID を送信しますが、 AS はそれをこの要求で使用することはありません。その代わり、この要求では AS 上の省略時のユーザー・プロファイルが使用されます (存在する場合)。 AS 上で省略時のユーザー・プロファイルが存在しない場合には、要求は拒否されます。
- AS 上の SECURELOC で *VFYENCPWD が指定されると、 AS は AR を保護ロケーションであると見なしますが、現行のユーザーが正当なユーザーかどうかを検査するため、ユーザー ID とパスワードを (暗号化された形式で) 送信することが必要になります。リクエスターによって送信されるユーザー ID と一致するユーザー・プロファイルが AS 上で存在していて、そのリクエスターが両方のシステムで同一のパスワードを持っている場合、要求は許可されます。そうでない場合には、要求は拒否されます。

表 4 には、 iSeries server 上の SNA SECURITY(PGM) を制御する要素の可能な組み合わせすべてが示されています。 列内の 『Y』 は、その要素が存在するか条件が満たされていることを示します。 PWD 列の 『M』 は、セキュリティー管理機能がユーザーのパスワードを検索し、パスワード保護が活動状態ならば、保護 (暗号化された) パスワードが送信されることを示しています。保護パスワードが送信されない場合には、パスワードは送信されません。*保護パスワード* とは、会話が開始される際に APPC がユーザー・パスワードに置き換える文字ストリングのことです。 保護パスワードは会話の両方のパートナーがパスワード保護をサポートしていて、パスワードが OS/400 バージョン 2 リリース 2 以降を実行するシステムで作成されている場合にのみ使用できます。

表 4. 分散リレーショナル・データベースへのリモート・アクセス

行	UID	PWD[1]	AVI	SEC(Y)	DFT	有効	アクセス
1	Y	Y		Y	Y	Y	UID を使用
2	Y	Y		Y	Y		拒否
3	Y	Y		Y		Y	UID を使用
4	Y	Y		Y			拒否
5	Y	Y			Y	Y	UID を使用
6	Y	Y			Y		拒否
7	Y	Y				Y	UID を使用
8	Y	Y					拒否
9	Y		Y	Y	Y	Y	UID を使用
10	Y		Y	Y	Y		拒否
11	Y		Y	Y		Y	UID を使用
12	Y		Y	Y			拒否
13	Y	M[3]			Y	Y	DFT または UID を使用 [2]
14	Y	M[3]			Y		DFT または UID を使用 [2]
15	Y	M[3]				Y	拒否または UID を使用 [2]

表 4. 分散リレーショナル・データベースへのリモート・アクセス (続き)

行	UID	PWD[1]	AVI	SEC(Y)	DFT	有効	アクセス
16	Y	M[3]					拒否または UID を使用 [2]
17				Y	Y		DFT を使用
18				Y			拒否
19					Y		DFT を使用
20							拒否

キー:

UID ユーザー ID の送信

PWD パスワードの送信

AVI 検査済み標識の設定

SEC(Y) SECURELOC(YES) の指定

DFT 通信サブシステムでの省略時のユーザー ID の指定

有効 ユーザー ID とパスワードが有効

UID を使用
提供されたユーザー ID を使用して接続する

DFT を使用
省略時のユーザー ID を使用して接続する

拒否 接続が行われなかった

注:

1. パスワード保護が活動状態であれば、保護パスワードが送信される。
2. パスワード保護が活動状態であれば、「UID を使用」となる。
3. パスワード保護が活動状態であれば、セキュリティー管理機能がユーザーのパスワードを検索し、保護パスワードを送信する。活動状態でなければ、パスワードは送信されない。

省略時のユーザー・プロファイルを使用しないようにするには、分散リレーショナル・データベース・オブジェクトにアクセスする必要のある AR ユーザーごとに、 AS 上にユーザー・プロファイルを作成してください。省略時のユーザー・プロファイルを使用することに決めた場合には、適切な権限のないユーザーにシステムの使用許可を与えないでください。たとえば、次のコマンドは省略時のユーザー・パラメーターを DFTUSER(QUSER) として指定します。これにより、システムは通信要求からユーザー ID またはパスワードを受け取らなくてもジョブ開始要求を受諾することができます。通信ジョブは、QUSER ユーザー・プロファイルを使用してサインオンされます。

```
ADDCMNE   SBSD(SAMPLE) DEV(*ALL) DFTUSER(QUSER)
```

TCP/IP を使用した DDM/DRDA セキュリティーの要素

ネイティブ TCP/IP 上の DDM/DRDA は、OS/400 通信セキュリティー・サービスや通信装置、モード、保護ロケーション属性などの概念を使用しません。また、APPC 通信と関連した会話セキュリティー・レベルも使用しません。 したがって、TCP/IP のセキュリティー設定はかなり異なったものです。

TCP/IP サーバーで使用できるセキュリティーのタイプに以下のものがあります。

- DDM/DRDA の接続セキュリティー・プロトコル
- DDM/DRDA の Secure Socket Layer (SSL)

- DDM/DRDA のインターネット・プロトコル・セキュリティー・プロトコル (IPSec)

分散データ管理機能 (DDM) 通信のセキュリティーのための新しい方法が提供されるようになったので、iSeries サーバーの管理担当者は、使用するポートをブロック化することによって特定の通信モードを制限できます。『DDM/DRDA のポートおよびポートの制限』では、これらの考慮事項のいくつかが取り扱われています。

DDM セキュリティーの詳細は、以下を参照してください。

- Application requester (AR) security in a TCP/IP network
- TCP/IP ネットワークでのアプリケーション・サーバー (AS) セキュリティー

DDM/DRDA の接続セキュリティー・プロトコル

TCP/IP 上で実装される DB2 UDB for iSeries の DDM/DRDA では、以下のいくつかの接続セキュリティー・プロトコルがサポートされています。

- ユーザー ID のみ
- クリア・テキスト・パスワード付きのユーザー ID
- 暗号化されたパスワード付きのユーザー ID
- Kerberos

暗号化されたデータ・ストリーム・サポートでは、通常の通信トレース・サポートを使用する利点はあまりありません。 TCP/IP アプリケーションのトレース (TRCTCPAPP) コマンドは、発信データ・ストリームを暗号化する前に記録して、着信データ・ストリームを暗号化解除後に記録します。このコマンドの使用方法を示す基本的な情報については、『通信トレース』のトピックを参照してください。

平文として流される特定のパスワードに関する注意

iSeries は接続パスワードの暗号化をサポートしているものの、 RDB ディレクトリー項目をセットアップする際に指定できる接続セキュリティー・オプションの 1 つに、 *USRIDPWD というオプションがあります (詳細については、『リレーショナル・データベース・ディレクトリーの処理』の、 RDB ディレクトリー項目の追加コマンドおよびリレーショナル・データベース・ディレクトリー項目の変更コマンドを参照してください)。

接続が行われるサーバーでこの *USRIDPWD セキュリティー・オプションが許可されている場合、接続のパスワードは平文で流すことが可能です。 V5R3 では、 SQL の SET ENCRYPTION PASSWORD ステートメントと ENCRYPT 関数でも、パスワードを平文でネットワーク上に流させることが可能です。現時点で、データ・ストリームの暗号化に使用できるソリューションは 2 つあります。 1 つは IPSec を使用する方法です。 48 ページの『DDM/DRDA のインターネット・プロトコル・セキュリティー・プロトコル (IPSec)』を参照してください。もう 1 つは、SSL をサポートする AR を使用していれば、 iSeries AS との間で伝送されるデータを暗号化するプロトコルが使用できます。

DDM/DRDA の Secure Socket Layer (SSL)

DB2 UDB for iSeries DRDA クライアントは、SSL をサポートしていません。しかし、インターネット・プロトコル・セキュリティー・プロトコル (IPSec) を使用して同様の機能が提供されます。

DDM TCP/IP サーバーでは、 Secure Socket Layer (SSL) データ暗号化プロトコルがサポートされています。 このプロトコルを使用して、 iSeries Toolbox for Java™ および iSeries Access Family OLE DB Provider (レコード・レベルのアクセス用の SSL をサポートする) などのクライアントで相互運用することができます。また、SSL をサポートする外部のソフトウェア・ベンダーが提供する DDM ファイル入出力クライアントでも可能です。

iSeries DDM TCP/IP サーバーで SSL を使用するには、クライアントを構成してサーバー上の割り当て済み SSL ポート 448 に接続する必要があります。

サーバー上の『DDM TCP/IP 属性の変更 (CHGDDMTCPA) コマンド』で PWDRQD(*ENCRYPTED) を指定すると、 Secure Socket Layer (SSL) で有効な任意のパスワードを使用することができます。これは、パスワードを含むデータ・ストリーム全体が暗号化されていることをサーバーが認識するので可能です。

SSL については、 iSeries Information Center の『ネットワーキング』の中の『SSL でのアプリケーションの保護』を参照してください。

必要なプログラム: PC および iSeries サーバーでの SSL の設定およびインストールについては、 iSeries Access for Windows を参照してください。

iSeries サーバー要件: iSeries が SSL 上で通信するには、システムは OS/400 V4R4 以降を実行していて、以下のものがインストールされている必要があります。

- TCP/IP 接続ユーティリティー iSeries 用、5769-TC1 (基本 TCP/IP サポート)
- Cryptographic Access Provider、5769-ACx
- IBM HTTP Server for iSeries、5769-DG1 (ディジタル認証マネージャーへのアクセス用)
- ディジタル認証マネージャー、5769-SS1 - Boss オプション 34
- クライアント暗号化サポート、5769-CEx - このプロダクトを iSeries 上にインストールする必要があり、ネットワーク内のすべての PC クライアントは必要な SSL クライアント・コードを検索する必要があります。 このプロダクトは、サーバー主導で SSL 通信を行う場合は必要ではなく、クライアントのみで必要です (注を参照)。

PC 要件 (iSeries Access Family および DRDA を使用する PC): ネットワーク上のクライアント PC が SSL で通信するには、以下のプロダクトのいずれかがインストールされている必要があります。

- クライアント暗号化サポート、40-bit 5769-CE1
- クライアント暗号化サポート、56-bit 5769-CE2
- クライアント暗号化サポート、128-bit 5769-CE3

注: SSL クライアント暗号化サポート・プロダクト (5769-CEx) のサービスは、 iSeries Access Family のサービス・パックからは独立したサービス・パックを通して処理されます。詳細は、iSeries Access Family ホーム・ページの通知 APAR II10598 を参照してください。

DDM/DRDA のインターネット・プロトコル・セキュリティー・プロトコル (IPSec)

インターネット・プロトコル・セキュリティー・プロトコル (IPSec) は、ネットワーク層のセキュリティー・プロトコルで暗号セキュリティー・サービスを提供します。これらのサービスは、インターネットまたはイントラネット上でデータの機密送達をサポートしています。

iSeries、IPSec では、仮想私設ネットワーク (VPN) サポートの構成要素は、アプリケーションが DRDA または DDM であるかどうかにはかかわりなく、 2 つの IP アドレスまたはポート間のデータがすべて暗号化されるようにします。IPSec で使用されるアドレスおよびポートを構成することができます。 IBM では、DRDA アクセスまたは DDM アクセス用の IPSec にはポート 447 を使用することをお勧めしています。VPN サポートの設定については、 iSeries Information Center の『**ネットワーキング**』の中の『VPN (仮想私設ネットワーク)』を参照してください。

IPSec とともに有効なパスワードを使用しても、普通は、サーバーで DDM TCP/IP 属性の変更 (CHGDDMTCPA) コマンドの PWDRQD(*ENCRYPTED) を指定する際に課せられる要件を満たせません。

なぜなら、アプリケーション (DRDA または DDM) は、 IPSec が使用されているかどうかを判別することができないからです。ですから、IPSec で PWDRQD(*ENCRYPTED) を使用するのは避けるべきです。

DDM/DRDA のポートおよびポートの制限

DDM/DRDA TCP/IP サーバーは、 447 ポート (割り当て済み DDM ポート) および 446 ポート (割り当て済み DRDA ポート)、さらに 448 ポート (割り当て済み SSL ポート) で listen します。 DDM の DB2 UDB for iSeries 実装では、446 と 447 という 2 つのポートの判別がされないので、 DDM および DRDA アクセスの両方は、どちらのポートでも行えます。

IPSec で推奨される規則を使用すると、 DDM の TCP/IP サーバーでは以下のようにポートが使用されます。

- 明示テキスト・データ・ストリームでは 446
- IPSec で暗号化されたデータ・ストリームでは 447 (推奨)
- SSL で暗号化されたデータ・ストリームでは 448 (必須)

TCP/IP の構成 (CFGTCP) コマンドを使用すると、サーバーでの 1 つ以上のポートの使用をブロックできます。これを行うには、そのコマンドの「TCP/IP ポート制約事項の処理」オプションを選んでください。制限を追加して、QRWTLSTN が実行されているユーザー・プロファイル (通常は QUSER) 以外の特定のユーザー・プロファイルだけが 446 などの特定のポートを使用するようにできます。これによって、446 はブロックされたことになります。 447 が IPSec 専用に構成された場合、446 をブロックすると、暗号化されたデータ・ストリームだけが、ネイティブ TCP/IP 上で DDM および DRDA アクセスで使用されることになります。 447 および 448 の両方をブロックして、SSL だけが使用できるように制限することもできます。パフォーマンス上の問題または他の理由があって (SSL が使用できるクライアントが現時点では少ないなど)、これらの例に従って設定を行うのは実際的ではないかもしれませんが、これらの例は、このような構成方法も可能であることを示すために挙げられています。

認証方式の折衝

接続性のシナリオが異なれば、異なるレベルの認証を使用することが求められます。したがって、管理者は、**アプリケーション・サーバー (AS)** へ接続するときに、より望ましい認証方式フィールドを各 RDB ディレクトリー項目に設定することにより、**アプリケーション・リクエスター (AR)** で必要な最低のセキュリティー認証方式を設定できます。さらに管理者は、より低いセキュリティー認証方式を許可するよう選択することにより、認証方式についての決定をサーバーと折衝することを可能にできます。この場合、より望ましい認証方式は試行中になりますが、AS がより望ましい方式を受け入れられない場合、サーバーのセキュリティー設定や、暗号化サポートの可用性などの他の要因に応じ、より低い方式を使用できます。たとえば、2 つのシステムが物理的に保護されていない環境にある場合、管理者は、より低いセキュリティー認証方式を許可せずに Kerberos 認証を必須とするよう選択できます。

アプリケーション・リクエスター (クライアント) 側では、 DRDA TCP/IP 接続要求のユーザー ID とともにパスワードを送信するのに 2 つの方法のいずれかを使用できます。これらの方法のどちらも使用しない場合には、 CONNECT コマンドはユーザー ID を送信できるだけです。

パスワードを送信する最初の方法は、対話式 SQL 環境からの以下の例のように、 SQL CONNECT ステートメントの USER/USING 形式を使用することです。

```
CONNECT TO rdbname USER userid USING 'password'
```

組み込み SQL を使用したプログラムでは、ユーザー ID とパスワードの値は USER/USING データベースのホスト変数に含められます。

CLI を使用するプログラムにおいて、DRDA アプリケーション・リクエスター (AR) に対し、ユーザー ID とパスワードをホスト変数で表す方法の一例は次のとおりです。

```
SQLConnect(hdbc,sysname,SQL_NTS,     /*do the connect to the application server */
     uid,SQL_NTS,pwd,SQL_NTS);
```

パスワードを提供する 2 番目の方法は、サーバー権限項目を使用して TCP/IP 上で接続要求を送信することです。サーバー権限リストは、システム上のすべてのユーザー・プロファイルと関連しています。省略時では、このリストは空です。しかし、サーバー権限項目の追加 (ADDSVRAUTE) コマンドを使用して項目を追加できます。 TCP/IP 上で DRDA 接続を試みる際には、 DB2 UDB for iSeries クライアント (AR) は、クライアント・ジョブが実行されているところのユーザー・プロファイルを調べるために、サーバー権限リストを検査します。 CONNECT ステートメントの RDB 名と権限項目の SERVER 名 (どちらも大文字でなければなりません) が一致することが分かれば、項目内の関連する USRID パラメーターが接続ユーザー ID として使用されます。 PASSWORD パラメーターが項目内で保管されている場合には、そのパスワードも接続要求で送信されます。

サーバー権限項目は、DDM ファイル入出力操作のために、TCP/IP 経由でパスワードを送信するときにも使用できます。 TCP/IP 上で DDM 接続を試みる際には、DB2 UDB for iSeries は、クライアント・ジョブが実行されているところのユーザー・プロファイルを調べるために、サーバー権限リストを検査します。RDB 名 (RDB ディレクトリー項目が使用される場合) と 'QDDMSERVER' のいずれかと、権限項目の SERVER 名が一致することが分かれば、項目内の関連する USRID パラメーターが接続ユーザー ID として使用されます。 PASSWORD パラメーターが項目内で保管されている場合には、そのパスワードも接続要求で送信されます。

サーバー許可項目の追加 (ADDSVRAUTE) コマンドを使用してパスワードを保管するには、QRETSVRSEC システム値を '1' に設定する必要があります。省略時では、この値は '0' です。この値を変更するには、次のコマンドを入力してください。

```
CHGSYSVAL QRETSVRSEC VALUE('1')
```

次の例は、RDB ディレクトリー項目を使用するときの、 サーバー許可項目の追加 (ADDSVRAUTE) コマンドの構文を示します。

```
ADDSVRAUTE USRPRF(user-profile) SERVER(rdbname) USRID(userid) PASSWORD(password)
```

USRPRF パラメーターは、アプリケーション・リクエスター・ジョブが実行されるユーザー・プロファイルを指定します。 SERVER パラメーターへ指定する内容は、通常は接続先の RDB の名前です。例外は、RDB ディレクトリーを使用する目的で作成されたわけではなかった DDM ファイルを使用する場合です。その場合、SERVER パラメーターには QDDMSERVER を指定する必要があります。 RDB 名を指定する場合、**大文字**でなければ**なりません**。 USRID パラメーターは、サーバー・ジョブが実行されるユーザー・プロファイルを指定します。 PASSWORD パラメーターは、ユーザー・プロファイルのパスワードを指定します。

USRPRF パラメーターを省略すると、 サーバー許可項目の追加 (ADDSVRAUTE) コマンドが実行されるところのユーザー・プロファイルが省略時値として使われます。 USRID パラメーターを省略すると、USRPRF パラメーターの値が省略時値として使われます。 PASSWORD パラメーターを省略したり、QRETSVRSEC 値を 0 に設定すると、項目にパスワードは保管されません。そして、その項目を使用した接続が試みられると、試行されるセキュリティー機構はユーザー ID だけになります。

| サーバー認証リストにどの認証項目が追加されているかは、 DSPSVRAUTE コマンドを使用して確認することができます。詳細は、『CL』トピックの『DSPSVRAUTE (Display Server Authentication Entries) Command Description』を参照してください。ユーザー作成プログラムの QsyRetrieveServerEntries

(QSYRTVSE) API も使用できます。 『API』のトピックの『Retrieve Server Authentication Entries (QSYRTVSE, QsyRetrieveServerEntries) API』を参照してください。

サーバー権限項目は、サーバー権限項目の除去 (RMVSVRAUTE) コマンドを使用して除去できます。サーバー権限項目は、サーバー権限項目の変更 (CHGSVRAUTE) コマンドを使用して変更できます。これらのコマンドの完全な説明については、Information Center の『制御言語 (CL)』を参照してください。

リレーショナル・データベース (RDB) でサーバー権限項目が存在していて、 CONNECT ステートメントの USER/USING 形式も使用されている場合には、後者の方が優先されます。

Kerberos ソース構成

DRDA および DDM は、両方のシステムが Kerberos 用に構成されていれば、 Kerberos 認証を利用することができます。 Kerberos 構成の詳細は、iSeries Information Center の『ネットワーク認証サービス』トピックを参照してください。ジョブのユーザー・プロファイルに有効なチケット許可チケット (TGT) がある場合、 DRDA **アプリケーション・リクエスター (AR)** はこの TGT を使用して、サービス・チケットを生成し、ユーザーをリモート・サーバーに認証します。有効な TGT を使用可能にすると、パスワードを直接入力する必要はなくなるため、サーバー認証項目を行う必要性はなくなります。しかし、ジョブのユーザー・プロファイルに有効な TGT がない場合、サーバー認証項目からユーザー ID とパスワードを取り出して、必要な TGT およびサービス・チケットを生成できます。

Kerberos を使用する場合、リモート・ホスト名として、 RDB ディレクトリー項目のリモート・ロケーション (RMTLOCNAME) を入力する必要があります。 Kerberos 認証では、IP アドレスは役立ちません。

Kerberos レルム名が DNS 接尾部名と異なる場合には、正確なレルムにマップする必要があります。これを行うには、Kerberos 構成ファイル (krb5.conf) 内に、各リモート・ホスト名を正しいレルム名にマップする項目がなければなりません。この入力されるホスト名は、リモート・ロケーション名 (RMTLOCNAME) と正確に一致している必要があります。 DSPRDBDIRE または DSPDDMF コマンドで表示されるリモート・ロケーション・パラメーターは、 krb5.conf ファイルのドメイン名と一致していなければなりません。次の図は、DSPRDBDIRE 画面の一例を示しています。

```
                    リレーショナル・データベース明細の表示

リレーショナル・データベース  : RCHASXXX

リモート・ロケーション :
  リモート・ロケーション  . . : rchasxxx.rchland.ibm.com
    タイプ  . . . . . . . . . : *IP
  ポート番号またはサービス名  : *DRDA
  リモート認証方式 :
    所望の方式  . . . . . . . : *KERBEROS
    低い認証の許可  . . . . . : *NOALWLOWER
テキスト  . . . . . . . . . . :

リレーショナル・データベース
      ・タイプ  . . . . . . . : *REMOTE




続行するには、実行キーを押してください。
   F3= 終了    F12= 取り消し
```

次に示すのは、対応する krb5.conf ファイルの内容の一部分です。ここには、リモート・ロケーション名と一致するドメイン名が示されています (注: 構成ファイルの内容を表示するために、 ファイルの表示 (DSPF) コマンドが使用されています):

```
DSPF STMF('/QIBM/UserData/OS400/NetworkAuthentication/krb5.conf')

[domain_realm]
; Convert host names to realm names. Individual host names may be
; specified. Domain suffixes may be specified with a leading period
; and will apply to all host names ending in that suffix.
 rchasxxx.rchland.ibm.com = REALM.RCHLAND.IBM.COM
```

Kerberos を使用するジョブは、krb5.conf ファイルに対して構成の変更が行われる場合に、再始動する必要があります。

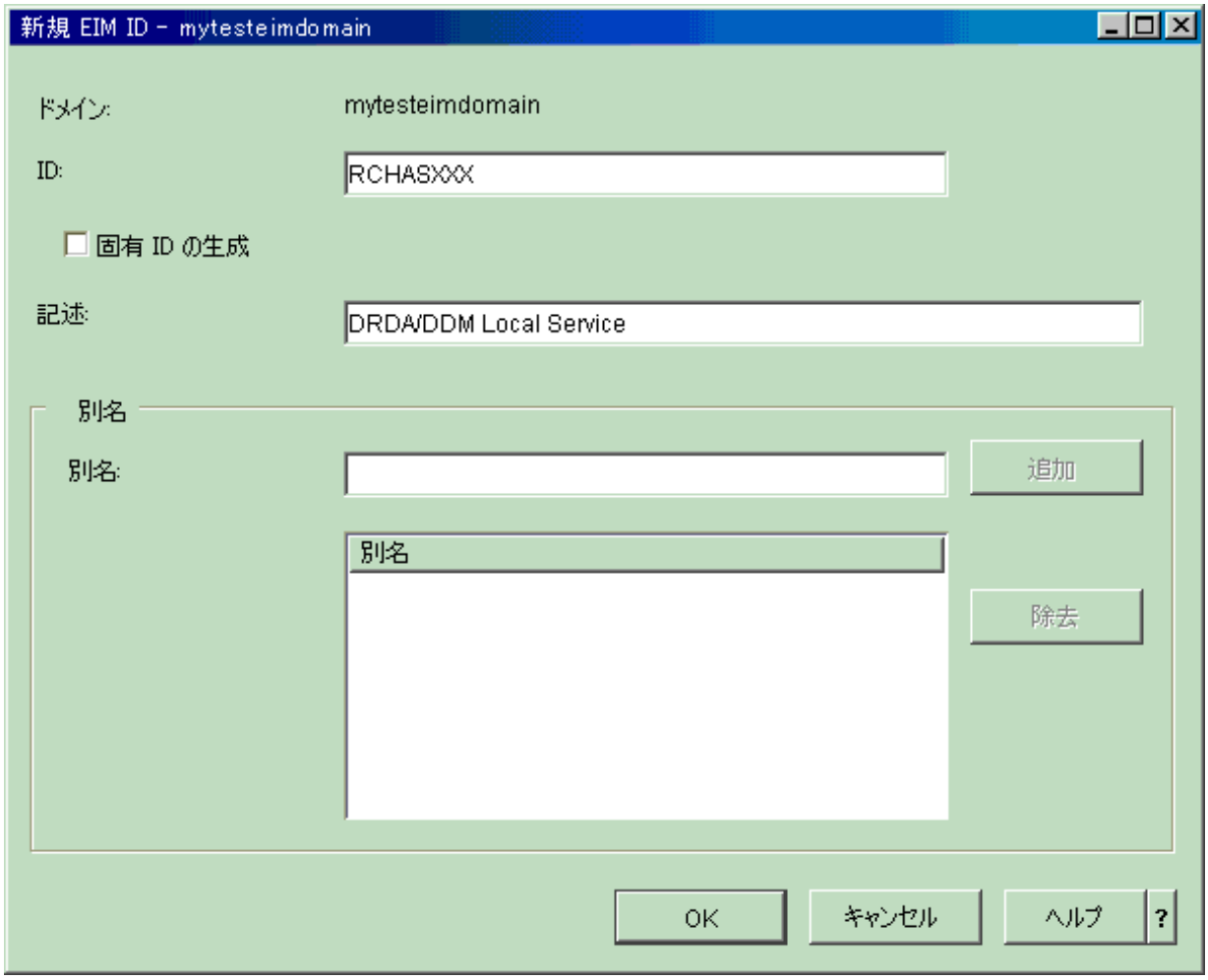
Kerberos DRDA サービス名の定義

注: DRDA レベル 5 を使用する DB2 バージョン 8 製品ファミリー・メンバーでの Kerberos 構成に関する詳細は、『ネットワーク認証サービス』を参照してください。

iSeries server 以外に接続するために Kerberos 認証を使用する場合、エンタープライズ識別マッピング (EIM) の下で、 iSeries 以外のサービス名を定義する必要があります。詳細は、iSeries Information Center の『エンタープライズ識別マッピング (EIM)』のトピックを参照してください。 DRDA サービス名を定義するには、以下のステップを実行します。

1. **iSeries ナビゲーター**を開始します。
2. 「**ネットワーク**」を展開します。
3. 「**エンタープライズ識別マッピング**」を展開します。
4. 「**ドメイン管理**」を展開します。
5. 自分の EIM ドメイン名を展開します。
6. 「**ID**」を右クリックして、「**新規 ID (New Identifier)**」を選択します。

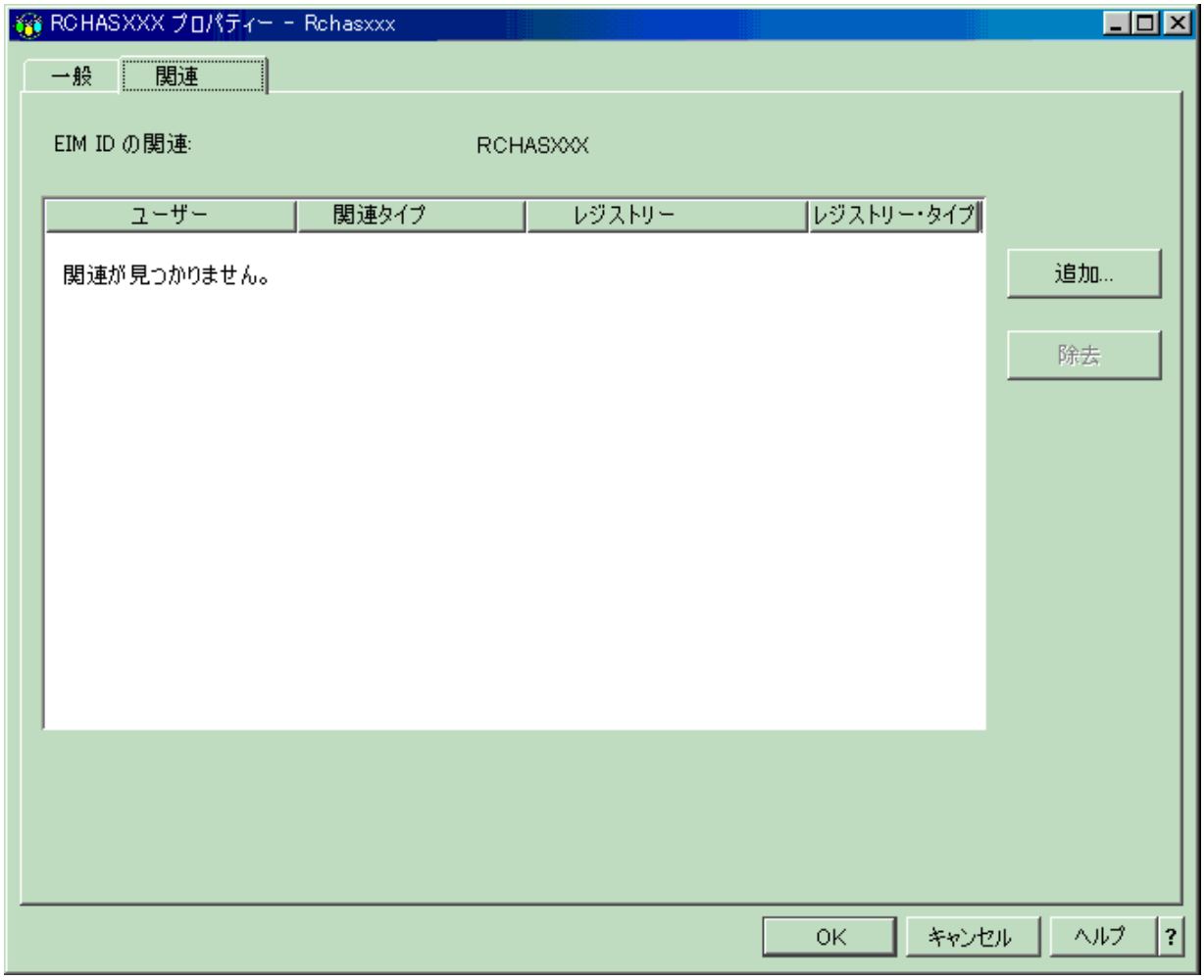
7. ID としてローカル RDB 名を入力し、必要であれば説明も入力します。



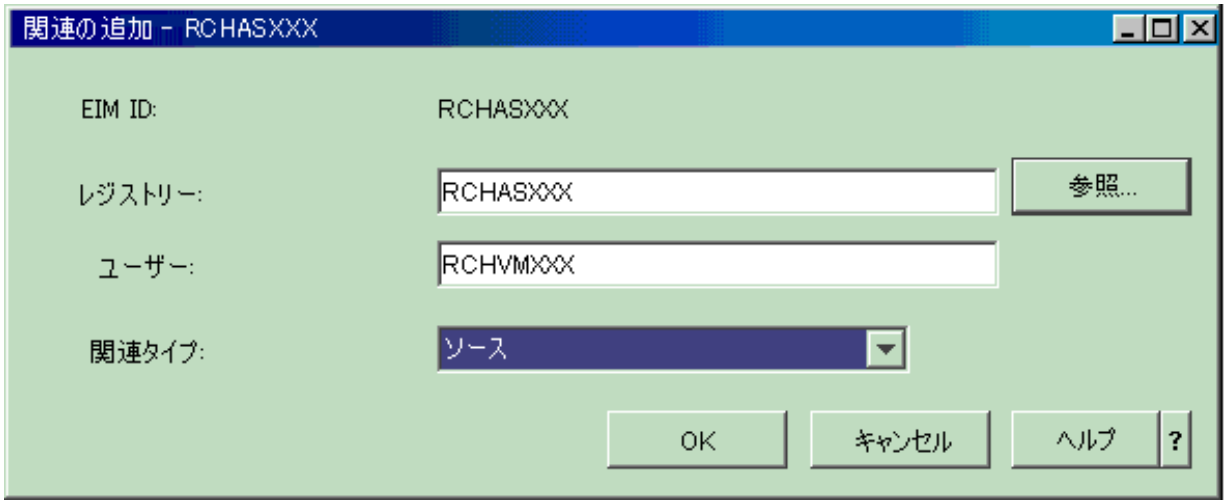
8. 「**OK**」をクリックします。

   作成した ID が iSeries ナビゲーターの右側のペインに示されます。

9. 作成した ID を右クリックし、「**プロパティー**」を選択します。
10. 「**関連**」タブをクリックします。

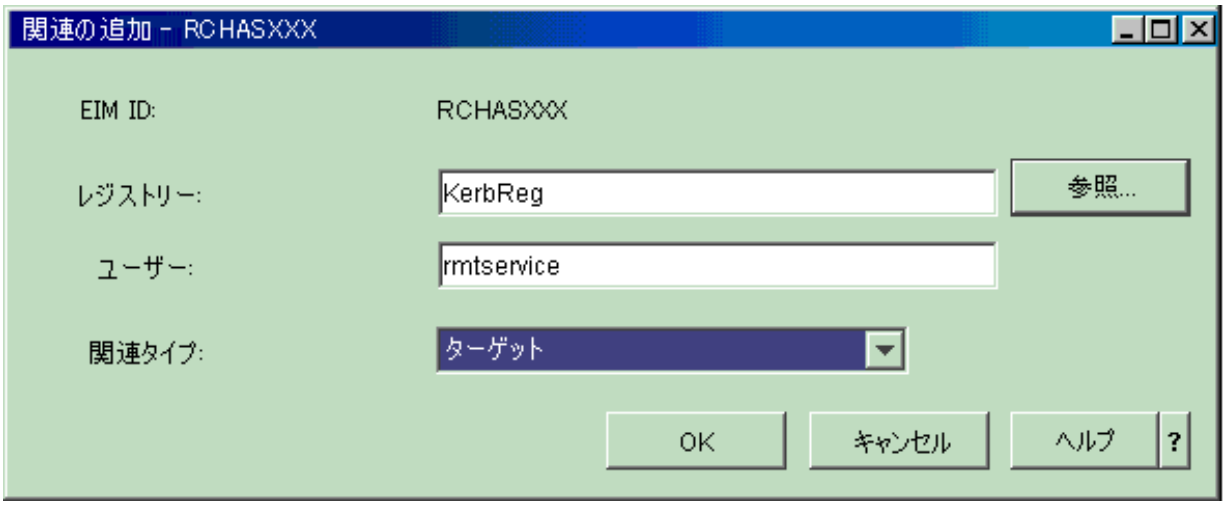


11. 「**追加**」をクリックして、新しい関連を追加します。
12. 「**ユーザー**」フィールドにリモート・ロケーション名 (RMTLOCNAME) を入力し、「関連タイプ」フィールドに**ソース**を入力します。



13. 「**OK**」をクリックします。 ID の「**プロパティー**」ダイアログに戻ります。

14. 「**追加**」をクリックして、2 番目の関連を入力します。
15. 「**レジストリー**」フィールドに、Kerberos レジストリーを入力します。「**ユーザー**」フィールドに、リモート・サーバーの Kerberos サービス名を入力します。「関連タイプ」フィールドで「**ターゲット**」を選択します。



16. 「**OK**」をクリックします。

TCP/IP ネットワークでのアプリケーション・サーバー (AS) セキュリティー

TCP/IP サーバーには、省略時のセキュリティーとして、クリア・テキストのパスワードを指定したユーザー ID が備えられています。これは、サーバーがインストールされると、着信 TCP/IP 接続要求ではサーバー・ジョブが実行されるユーザー ID とともに、最低でもクリア・テキストのパスワードが必要になることを表しています。セキュリティーは、DDM TCP/IP 属性の変更 (CHGDDMTCPA) コマンドか、iSeries ナビゲーターの「**ネットワーク**」->「**サーバー**」->「**TCP/IP**」->「**DDM サーバー (DDM server)**」プロパティーで変更できます。この設定を変更するには、*IOSYSCFG 特殊権限が必要です。

低いサーバー・セキュリティーに使用できる 2 つの設定があります。

- PWDRQD (*NO)

  パスワードは必要でない
- PWDRQD(*VLDONLY)

  パスワードは必須ではないが、送信される場合には有効でなければならない。

*NO と *VLDONLY の違う点を挙げると、パスワードがクライアント・システムから送信される場合に、*NO オプションでは無視されます。それに対して *VLDONLY オプションでは、パスワードが送信されると、そのパスワードは付随するユーザー ID の妥当性が検査されて、間違っている場合にはアクセスが拒否されます。

より高いセキュリティー・レベルでは、暗号化された必須のパスワードか PWDRQD(*ENCRYPTED) と、Kerberos か PWDRQD(*KERBEROS) を使用できます。 Kerberos を使用する場合、エンタープライズ識別マッピング (EIM) を使用して、ユーザー・プロファイルを Kerberos プリンシパルにマッピングする必要があります。詳細は、iSeries Information Center の『Enterprise Identity Mapping (EIM)』のトピックを参照してください。

| **注:** *ENCRYPTED を使用するためには、両方のシステムに暗号化製品 5669-AC2 または 5769-AC3、あるいはそれと同等のものがインストールされている必要があります。

次に示すのは、DDM TCP/IP 属性の変更 (CHGDDMTCPA) コマンドを使用して、ユーザー ID に暗号化されたパスワードが付随しなければならないことを指定する一例です。このオプションを設定するには、次のように入力します。

```
CHGDDMTCPA PWDRQD(*ENCRYPTED)
```

注: V4R4 では、DDM/DRDA TCP/IP サーバーが拡張されてパスワード置換 と呼ばれるパスワード暗号化の形式がサポートされるようになりました。 V4R5 では、より広範囲にわたって使用されているパスワード暗号化技法である Diffie-Hellman 公開キー・アルゴリズムが実装されました。これは、DRDA の標準アルゴリズムで、最新版の IBM DRDA アプリケーション・リクエスターによって使用されます。古いパスワード置換アルゴリズムは、 PC クライアントからの DDM ファイル・アクセスで主に使用されています。 V5R1 では、「強力な」パスワード置換アルゴリズムもサポートされていました。クライアントとサーバー間では使用されるセキュリティー・メカニズムの折衝が行われます。 3 つあるどの暗号化方式を使用しても PWDRQD(*ENCRYPTED) の要件が満たされ、 Secure Socket Layer (SSL) のデータ・ストリームの使用にも十分です。

DRDA サーバー・アクセス制御出口プログラム

APPC と TCP/IP の両方で使用される DRDA サーバーのセキュリティー機能は、 ネットワーク属性変更 (CHGNETA) コマンドの DDMACC パラメーターの使用を DRDA にも拡張します。このパラメーターは、以前には DDM ファイルの入出力アクセスだけに適用されました。この機能の DRDA での使用は、接続要求だけに限られているので、接続が行われた後にデータを要求するためには使われません。

このセキュリティー機能を利用したくない場合には、通常は何も行う必要はありません。唯一の例外は、現在 DDM 出口プログラムを使用していて、知られていない機能コードが受信された際に処理を拒否するようにコーディングされており、そのシステムのデータをアクセスするのに DRDA を使用している場合だけです。この場合には、出口プログラムを変更して、機能コードが 'SQLCNN ' である場合に、 '1' を戻して DRDA アクセスが行われるようにしなければなりません。

DRDA 接続をブロックまたはフィルター処理するために出口プログラムを使用する場合、新しい DRDA 出口プログラムを作成するか、既存のものを変更する必要があります。

注: システムが複数のデータベース (ASP グループ) で構成されている場合、出口プログラムは (範囲が 1 から 32 の補助記憶域プール上の) システム・データベース内のライブラリーに存在していなければなりません。

iSeries Information Center の『分散データベース管理』トピックには、 DRDA 出口プログラムを作成するための一般的な情報が記載されています。

このセキュリティー機能は、DRDA 機能コードを、入力パラメーター構造でプログラムに入力できる要求機能のリストに追加します。 'SQLCNN ' (SQL 接続要求) という名前の機能コードは、 DRDA 接続要求が処理されていることを示します (61 ページの図 7 の FUNC パラメーターを参照)。 DRDA 接続要求呼び出しでは、 APP (アプリケーション) 入力パラメーターは '*DDM ' ではなく、 '*DRDA ' と設定されます。

DRDA の出口プログラムをコーディングする際には、パラメーター構造内の以下のフィールドが役立つかもしれません。

- USER フィールドによりプログラムは、ユーザー・プロファイル ID に基づいて DRDA アクセスを許可したり拒否したりします。
- RDBNAME フィールドには、ユーザーが接続する先の RDB の名前が示されます。これは、システム・データベースかユーザー・データベース (ASP グループ) である可能性があります。このフィールドは、複数のデータベースを構成する環境内にある、 1 つ以上のデータベースへのアクセスを拒否する場合に役立ちます。
- 61 ページの図 7 の SRVNAME パラメーターは、出口プログラムの呼び出し元から設定される場合もあり、設定されない場合もあります。このパラメーターが設定される場合、クライアント・システムの名前を表します。このパラメーターが設定されない場合には、パラメーターの値は *N になります。DRDA アプリケーション・リクエスターが iSeries server であるならばいつも設定されます。
- TYPDEFN パラメーターは接続しているクライアントのタイプについて追加の情報を提供します。 IBM メインフレームでは、TYPEDEFN は QTDSQL370 になります。 iSeries server では、QTDSQL400 になります。 Intel® PC では、QTDSQLX86 になります。 RS/6000® クライアントでは、QTDSQLASC になります。
- PRDID (プロダクト ID) パラメーターは、接続しようとしているプロダクトと、そのプロダクトのリリース・レベルを識別します。次に示すのは、これらのコードの最初の 3 文字の部分リストです (出口プログラムで使用する前に非 IBM コードを検査する必要があります)。

  QSQ IBM DB2 UDB for iSeries

  DSN IBM DB2 UDB for z/OS

  SQL IBM DB2 Connect™ (以前は DDCS と呼ばれていた)

  ARI IBM DB2 UDB for VSE & VM

  GTW Oracle 社のプロダクト

  GVW Grandview DB/DC Systems のプロダクト

  XDB XDB Systems のプロダクト

  IFX Informix® Software のプロダクト

  RUM Wall Data Rumba for Database Access

  SIG StarQuest プロダクト

  STH FileTek プロダクト

| **JCC** IBM DB2 Universal Driver for SQLJ and JDBC

  フィールドの残りは vvrrm として構造化されます。ここで、vv はバージョン、rr はリリース、および m はモディフィケーション・レベルを表します。

出口プログラムが '0' という RTNCODE 値を戻して、接続要求が iSeries クライアントからのものである場合、ユーザーへの接続の失敗を示すメッセージは、SQ30060, 'ユーザーにはリレーショナル・データベース... が認可されていません' になります。 . 通常は、出口プログラムによるアクセス拒否への応答は DRDA RDBATHRM 応答メッセージで行われます。この応答メッセージは、ユーザーがリレーショナル・データベースへの許可を持っていないことを示します。 クライアントのプラットフォームが異なれば、ユーザーへのエラーも異なる場合があるので注意してください。

制限:

- 機能チェックがユーザー出口プログラムで起きる場合、プログラムによって同様の応答メッセージが戻され、接続の試みは失敗します。出口プログラムは、DB2 UDB for iSeries に対してコミット可能な更新を行うべきではありません。さもなければ、予期できない結果が起こる可能性があります。

- 出口プログラムを使用して、事前開始サーバー・ジョブの以前の呼び出しで開かれたファイルをアクセスしようとするべきではありません。
- V5R2 より前は、TCP/IP サーバーで使用される事前開始ジョブを後で使用するためにリサイクルする場合、さらに多くの制限がありました。次回にジョブを使用するよう準備するため、終結処理が行われます。この処理の一部として、値が *ELIGIBLE である ACTGRP パラメーターを指定した活動化グループの再利用 (RCLACTGRP) コマンドを使用することが関係します。その結果、事前開始サーバー・ジョブ内に残っているリンケージを、 RCLACTGRP で破棄された活動化グループに対して使用しようとすると、 MCH3402 例外 (もはや存在していないすべてまたは一部のオブジェクトを参照しようとした) が起こり得ます。この制限の回避策の 1 つは、QRWTSRVR 事前開始ジョブの MAXUSE 値を 1 に設定することです。そうすると、CHGPJE SBSD(QSYSWRK) PGM(QRWTSRVR) MAXUSE(1) のようになります。

例: DRDA サーバー・アクセス制御出口プログラム

61 ページの図 7 では、すべての DRDA 操作と、ユーザー ID が 'ALIEN' 以外の DRDA 接続をすべて許可している PL/I ユーザー出口プログラムの例を示します。

注: xv ページの『コードの特記事項情報』に重要なリーガル情報がありますので、お読みください。

```
/*******************************************************************/
/*                                                                 */
/* PROGRAM NAME: UEPALIEN                                          */
/*                                                                 */
/* FUNCTION:     USER EXIT PROGRAM THAT IS DESIGNED TO             */
/*               RETURN AN UNSUCCESSFUL RETURN CODE WHEN           */
/*               USERID 'ALIEN' ATTEMPTS A DRDA CONNECTION.        */
/*               IT ALLOWS ALL TYPES OF DDM OPERATIONS.            */
/*                                                                 */
/* EXECUTION:    CALLED WHEN ESTABLISHED AS THE USER EXIT          */
/*               PROGRAM.                                          */
/*                                                                 */
/* ALL PARAMETER VARIABLES ARE PASSED IN EXCEPT:                   */
/*                                                                 */
/* RTNCODE -    USER EXIT RETURN CODE ON WHETHER FUNCTION IS       */
/*              ALLOWED: '1' INDICATES SUCCESS; '0' FAILURE.       */
/*                                                                 */
/*******************************************************************/

UEPALIEN: PROCEDURE (RTNCODE,CHARFLD);

DECLARE RTNCODE CHAR(1);           /* DECLARATION OF THE EXIT      */
                                   /* PROGRAM RETURN CODE.  IT     */
                                   /* INFORMS REQUEST HANDLER      */
                                   /* WHETHER REQUEST IS ALLOWED.  */
DECLARE                            /* DECLARATION OF THE CHAR      */
 1  CHARFLD,                       /* FIELD PASSED IN ON THE CALL. */

 2  USER     CHAR(10),             /* USER PROFILE OF DDM/DRDA USER */
 2  APP      CHAR(10),             /* APPLICATION NAME              */
 2  FUNC     CHAR(10),             /* REQUESTED FUNCTION            */
 2  OBJECT   CHAR(10),             /* FILE NAME                     */
 2  DIRECT   CHAR(10),             /* LIBRARY NAME                  */
 2  MEMBER   CHAR(10),             /* MEMBER NAME                   */
 2  RESERVED CHAR(10),             /* RESERVED FIELD                */
 2  LNGTH    PIC '99999',          /* LENGTH OF USED SPACE IN REST  */
 2  REST,                          /* REST OF SPACE = CHAR(2000)    */

 3  LUNAME  CHAR(10),              /* REMOTE LU NAME (IF SNA)       */
 3  SRVNAME CHAR(10),              /* REMOTE SERVER NAME            */
 3  TYPDEFN CHAR(9),               /* TYPE DEF NAME OF DRDA AR      */
 3  PRDID,                         /* PRODUCT ID OF DRDA AR         */

 5 PRODUCT  CHAR(3),               /* PRODUCT CODE                  */
 5 VERSION  CHAR(2),               /* VERSION ID                    */
 5 RELEASE  CHAR(2),               /* RELEASE ID                    */
 5 MOD      CHAR(1),               /* MODIFICATION LEVEL            */
 5 RDBNAME CHAR(18),               /* RDB NAME                      */
 5 REMAING CHAR(1965),             /* REMAINING VARIABLE SPACE      */

START:
IF (USER = 'ALIEN' &          /* IF USER IS 'ALIEN' AND          */
      FUNC = 'SQLCNN') THEN        /*   FUNCTION IS DRDA CONNECT    */
  RTNCODE = '0';                   /* SET RETURN CODE TO UNSUCCESSFUL */
ELSE                               /* IF ANY OTHER USER, OR DDM     */
  RTNCODE = '1';                   /* SET RETURN CODE TO SUCCESSFUL */

END UEPALIEN;
```

図 7. 例: PL/I ユーザー出口プログラム

DRDA 用のオブジェクト関連のセキュリティー

iSeries server が**アプリケーション・サーバー (AS)** である場合、セキュリティーを適用できる 2 つのオブジェクト関連レベルがあり、リレーショナル・データベース・テーブルへのアクセスを制御します。

ネットワーク属性変更 (CHGNETA) コマンド上では DDMACC パラメーターが使用され、このサーバー上のテーブルが別のシステムでアクセスできるかどうかが示されます。アクセスできる場合には、着信 DRDA 要求がどのセキュリティー・レベルで検査されるかも示されます。

- DDMACC パラメーター上で *REJECT が指定されている場合、 AS で受信される分散リレーショナル・データベース要求はすべて拒否されます。しかし、このシステム (**アプリケーション・リクエスター (AR)** として) は SQL 要求を使用して、アクセスを許可する他のシステムにあるテーブルにアクセスすることができます。 *REJECT を指定する iSeries server では、リモート・システムによるデータベースのアクセスは行えません。

  SQL 要求がすでに使用中である時に *REJECT が指定された場合には、このシステムのデータベースに対するアクセスを要求するシステムからの*新しい* ジョブはすべて拒否され、それらのジョブにエラー・メッセージが戻されます。既存のジョブに影響はありません。
- DDMACC パラメーターに *OBJAUT が指定されている場合には、通常のオブジェクト・レベルのセキュリティーが AS 上で使用されます。

  DDMACC パラメーターは最初 *OBJAUT に設定されています。 *OBJAUT 値はリモート要求をすべて許可しますが、それらの要求はこの AS 上のオブジェクト権限によって制御されます。 DDMACC 値が *OBJAUT である場合には、このジョブで使用されるユーザー・プロファイルに、私用権限、共通権限、グループ権限、または借用権限を通して適切なオブジェクト権限が与えられているか、 AR ジョブによって必要とされるオブジェクトの権限リストにプロファイルが存在していなければなりません。システム上の各 SQL オブジェクトについて、そのオブジェクトにアクセスする権限を、すべてのユーザー、または特定のユーザーのみに (ユーザー ID によって) 与えることができます。また、誰にもその権限を与えないこともできます。

  オブジェクトで許可されるユーザー ID は、AS ジョブのユーザー ID です。 AS ジョブを実行するときのユーザー・プロファイルの種類については、『Elements of DDM Security in an APPC network』のトピックを参照してください。

  TCP/IP 接続の場合には、まず最初にサーバー・ジョブは QUSER で実行されます。ユーザー ID が検査された後に、接続要求で指定されたユーザー・プロファイルでジョブが実行されるようにユーザー・プロファイルの交換が行われます。ジョブは、そのユーザー・プロファイルの属性 (たとえば、ライブラリー・リスト) を継承します。

  値 *OBJAUT が指定されると、それ以上の検査 (iSeries オブジェクト・レベルのセキュリティーを超えるもの) は必要でないことを示します。
- DDM ジョブでは、DDMACC パラメーターでユーザー提供の任意のユーザー出口プログラム (またはアクセス制御プログラム) の名前が指定されると、追加のセキュリティーのレベルが使用されます。 DDM クライアントのユーザーが、iSeries サーバーの特定のファイルをアクセスするのに特定のコマンドを使用できるかどうかを制御するために、ユーザー出口プログラムを使用することができます。

  DRDA ジョブでは、DDMACC パラメーターでユーザー提供の任意のユーザー出口プログラム (アクセス制御プログラム) の名前が指定されると、システムは *OBJAUT が指定されたかのようにその入力を扱います (1 つの例外を除いて)。ユーザー作成の出口プログラムが DRDA ジョブに対して行えるのは、接続要求を拒否することだけです。詳細は、『DRDA サーバー・アクセス制御出口プログラム』のトピックを参照してください。

最初は *OBJAUT に設定される DDMACC パラメーターは、 ネットワーク属性変更 (CHGNETA) コマンドを使用すると前述の値のいずれかに変更できます。そして、ネットワーク属性表示 (DSPNETA) コマンドで現行値を表示できます。 また、ネットワーク属性検索 (RTVNETA) コマンドを使用して、制御言語 (CL) プログラム内の値を入手することもできます。

DDMACC パラメーター値が変更されるとその変更はすぐに反映されますが、システムで AS として開始された*新しい* 分散リレーショナル・データベース・ジョブだけが影響を受けます。この AS 上で変更前に実行されていたジョブは、古い値を使用し続けます。

DDMACC パラメーターの説明については、V5R1 Supplemental Manuals Web サイトにある、

「*Communications Management*」  のマニュアルで、ネットワーク属性変更 (CHGNETA) コマンドの説明を参照してください。

分散リレーショナル・データベース・オブジェクトの権限

SQL GRANT および REVOKE ステートメント、または制御言語 (CL) であるオブジェクト権限認可 (GRTOBJAUT) およびオブジェクト権限取り消し (RVKOBJAUT) コマンドのいずれかを使用して、リレーショナル・データベース・オブジェクトに対するユーザー権限の認可および取り消しを行うことができます。 SQL GRANT および REVOKE ステートメントは、パッケージ、テーブル、およびビューのみで動作します。ある場合には GRTOBJAUT および RVKOBJAUT を使用して、それ以外のオブジェクトに対する権限をユーザーに認可する必要があります。

SQL ステートメントの場合に検査の対象となる権限は、ステートメントが静的か、動的か、または対話式に実行されるかによって異なります。

CRTSQLxxx コマンドの USRPRF パラメーターで指定できる値の意味、および SQL ステートメントの静的および動的ステートメントとの相違点については、 『SQL プログラミングの概念』のセキュリティーのセクションを参照してください。

対話式 SQL ステートメントの場合は、そのステートメントの処理を行う人の権限の検査が行われます。対話式の SQL ステートメントに対しては、借用権限は使用されません。

分散リレーショナル・データベース・アプリケーションを実行するユーザーは、**アプリケーション・サーバー (AS)** 上で SQL パッケージを実行するための権限を必要とします。 GRANT EXECUTE ON PACKAGE ステートメントを使用すると、SQL パッケージの所有者、またはその SQL パッケージに対する管理特権を持っているユーザーは、指定したユーザーに SQL パッケージの中のステートメントを実行する特権を認可することができます。このステートメントを使用して、AS に対する権限を認可されているすべてのユーザー、または AS 上の 1 つ以上のユーザー・プロファイルのリストに、 SQL パッケージの中のステートメントを実行する特権を与えることができます。

通常、ユーザーは、CRTSQL*xxx* コマンドを使用して作成された分散アプリケーション・プログラムに対する権限を認可されている場合には、パッケージに対して処理特権を持っています。パッケージが構造化照会言語パッケージの作成 (CRTSQLPKG) コマンドを使用して作成されている場合には、そのパッケージに対する処理特権をユーザーに授与しなければならないことがあります。このステートメントは、 SQL プログラムに組み込むか、または対話式 SQL を使用して出すことができます。以下にステートメント例を示します。

```
GRANT EXECUTE
ON PACKAGE  SPIFFY.PARTS1
TO PUBLIC
```

REVOKE EXECUTE ON PACKAGE ステートメントを使用すると、SQL パッケージの所有者、またはその SQL パッケージに対する管理特権を持っているユーザーは、指定したユーザーから SQL パッケージの中のステートメントを実行する特権を取り去ることができます。 AS に対する権限を認可されているすべてのユーザー、または AS 上の 1 つ以上のユーザー・プロファイルのリストに対する EXECUTE 特権を取り去ることができます。

同一のユーザーに同一の特権を複数回授与している場合には、そのユーザーからその特権を取り消すことによって、すべての認可がいずれも無効になります。あるユーザーに以前授与した SQL パッケージに対する EXECUTE 特権を取り消した場合には、その SQL パッケージに対する EXECUTE 特権の授与は、誰が授与したものであっても、すべて無効になります。以下にステートメント例を示します。

```
REVOKE EXECUTE
ON PACKAGE  SPIFFY.PARTS1
FROM PUBLIC
```

SQL パッケージに対する権限は、オブジェクト権限認可 (GRTOBJAUT) コマンドを使用して授与し、 オブジェクト権限取り消し (RVKOBJAUT) コマンドを使用して取り消すこともできます。

分散リレーショナル・データベースの借用権限のもとで実行されるプログラム

分散リレーショナル・データベース・プログラムは、借用権限のもとで実行することができます。このことは、ユーザーが、プログラムの実行時にプログラムで使用されるオブジェクトに対するプログラム所有者の権限を借用することを意味します。プログラムが命名のための *SQL 事前コンパイラー・オプションを使用して作成された時は、そのプログラムはプログラム所有者のユーザー・プロファイルのもとで実行されます。

異環境システムからの SQL パッケージでは、そのパッケージの中の静的 SQL ステートメントのすべてについて、常にパッケージ所有者の権限を借用します。 OPTION(*SQL) を指定した CRTSQL*xxx* コマンドを使用して iSeries server 上で作成された SQL パッケージでも、そのパッケージの中の静的 SQL ステートメントのすべてについて、パッケージ所有者の権限を借用します。

分散リレーショナル・データベース管理担当者は、プログラム借用表示 (DSPPGMADP) コマンドを使用することによって、アプリケーション・サーバーにおけるセキュリティーの問題を検査することができます。DSPPGMADP コマンドを実行すると、以下に示すように、指定したユーザー・プロファイルを使用するプログラムおよび SQL パッケージが表示されます。コマンドを実行した結果をプリンターまたは出力ファイルに送ることもできます。

```
                          借用プログラムの表示

ユーザー・プロファイル . . :   MPSUP

オブジェクト    ライブラリー      タイプ    属性        テキスト
INVENT          SPIFFY            *PGM                  借用プログラム
CLIENT1         SPIFFY            *PGM                  借用プログラム
TESTINV         TEST              *PGM      CLP         テスト在庫プログラム
INVENT1         SPIFFY            *SQLPKG               SQL パッケージ
CLIENT1         SPIFFY            *SQLPKG               SQL パッケージ
TESTINV         SPIFFY            *SQLPKG               SQL パッケージ




終わり
続行するには、実行キーを押してください。

 F3= 終了    F12= 取り消し    F17= 最上部    F18= 最下部
(C) COPYRIGHT IBM CORP. 1980, 2000.
```

分散リレーショナル・データベースでの保護の方針

アプリケーション・サーバー (AS) 上の重要なデータを無許可アクセスから保護するには、 iSeries 分散リレーショナル・データベースでのネットワーク・セキュリティーを計画しなければなりません。ただし、リレーショナル・データベースの種類が分散型であるため、セキュリティー計画では、ネットワークの中におけるデータの可用性が不必要に制約されることがないようにしなければなりません。

分散リレーショナル・データベース管理担当者が行う必要のある決定の 1 つとして、ネットワークの中の各システムに適したシステム・セキュリティー・レベルの決定があります。システム・セキュリティー・レベルが 10 の場合は、アプリケーション・サーバーに対するセキュリティーは、システム・サイトにおける物理的セキュリティー以外には何も提供されません。システム・セキュリティー・レベルが 20 の場合は、ネットワーク・セキュリティー検査が行われて、ローカル・システムおよびリモート・システムが正しく識別されるので、アプリケーション・サーバーに対してある程度の保護が提供されます。ただし、このレベルのセキュリティーでは、重要なデータベース要素を無認可アクセスから保護するために必要なオブジェクト許可は提供されません。したがって、ネットワークの中のシステムが特定のシステム・オブジェクトの保護を必要とする場合には、 30 かそれ以上の iSeries server セキュリティー・レベルの選択をお勧めします。

分散リレーショナル・データベース管理担当者は、ネットワーク上の**アプリケーション・リクエスター (AR)** とアプリケーション・サーバーとの間で通信を確立する方法についても考慮しなければなりません。そのためには、以下の質問事項を解決する必要があります。

- AS 上に省略時のユーザー・プロファイルを存在させるかどうか。

  多くのユーザー・プロファイルをネットワークの中で維持管理するのは難しい場合があります。これに対して、通信サブシステムの項目の中に省略時のユーザー・プロファイルを作成すると、 AS が保護ロケーションでない場合には、 AS は受信する通信要求に対して開放されることになります。この状態は、場合によっては、受け入れ可能なこともありますが、省略時のユーザー・プロファイルのために、システム保護機能が著しく低下して、セキュリティー要件を満たせない場合もあります。

  たとえば、システムが多くの AR にサービスを提供する場合には、高いレベルのセキュリティーを必要とします。そのようなシステムのデータベースに脱落または損傷が生じた場合には、ネットワーク全体

が影響を被ることになります。アクセスを必要とする可能性のあるユーザーをすべて識別するユーザー・プロファイル、またはグループ・プロファイルを AS 上に作成することは可能であるため、データベース管理担当者は、分散リレーショナル・データベースの作業を管理する通信サブシステムのために、省略時のユーザー・プロファイルを作成することを考える必要はありません。

これに対して、ネットワークの中で他のシステムに対して AS として機能することがほとんどなく、機密性の高い重要なデータが入っていない iSeries server の場合には、分散リレーショナル・データベース作業を管理する通信サブシステムの省略時のユーザー・プロファイルを使用することができます。 これが特に効果的であるのは、ネットワークの中の他のすべてのシステムで同じアプリケーションを使用して、このデータベースに対する作業を処理する場合です。

省略時のユーザーという概念は、厳密には APPC の使用の際にのみ適用されます。ただし、TCP/IP を使用しているシステムでは、これに類似した技法を使用することができます。ユーザー ID を 1 つ確立してから、そのもとでサーバー・ジョブを実行することができます。 そして、すべての AR でサーバー許可項目の追加 (ADDSVRAUTE) コマンドを使用して、接続するユーザーすべてがそのユーザー ID を使用する必要があることを指定できます。 AS の DDM TCP/IP 属性の変更 (CHGDDMTCPA) コマンド上にある PWDRQD パラメーターの設定に応じて、サーバー権限項目でパスワードを指定するか、パスワードとして *NONE を指定することができます。この属性の省略時の値では、パスワードは必須です。

- データベース・オブジェクトに対するアクセスをどのように処理するか。

  オブジェクトに対する権限は、私用権限、グループ権限、共通権限、借用権限、および権限リストによって認可することができます。通信要求が受諾されるためには、ユーザー・プロファイル (または省略時のプロファイル) が AS 上に存在していなければなりませんが、オブジェクトに対する権限をユーザーに認可する方法によっては、パフォーマンスに影響が生じる可能性があります。

  分散リレーショナル・データベース・オブジェクトに対するアクセスの認可にあたっては、可能なかぎり、グループ権限または権限リストを使用してください。これらの検査の方が、すべての私用権限を検査する場合に比べて、時間およびシステム・リソースの節減を図ることができます。

  TCP/IP 接続では、ユーザー ID のマッピングを行うことができるので、 AS に接続できるユーザーごとに専用ユーザー ID を割り当てる必要はありません。

第 5 章 iSeries 分散リレーショナル・データベースのセットアップ

iSeries 分散リレーショナル・データベースの実行時サポートは、 OS/400 プログラムによって提供されます。したがって、オペレーティング・システムをインストールするときに、分散リレーショナル・データベース・サポートがインストールされます。ただし、とりわけ APPC 環境などにおいては、アプリケーション・リクエスターやアプリケーション・サーバーで作業の送受信が行えるようにするために、いくらかのセットアップ作業が必要になる場合があります。 1 つ以上のサブシステムを使い、対話式ジョブ、バッチ・ジョブ、スプール・ジョブ、および通信ジョブを制御することができます。また、ネットワーク内のすべてのアプリケーション・リクエスター (AR) には、リレーショナル・データベース・ディレクトリーと、そこへの接続情報がセットアップされなければなりません。さらに、ネットワーク全体のアプリケーション・サーバーのテーブルに、データを入れることも必要かもしれません。

リレーショナル・データベース・ディレクトリーには、データベース名および通信ネットワーク・パラメーターに変換される値が入ります。 AR は、ローカル・データベースや、構成されるすべてのローカル・ユーザー・データベースを含む、ネットワーク内の各データベースに、項目を持たなければなりません。これらのローカル項目は、システムに自動で追加させることもできますし、手動で追加することもできます。各ディレクトリー項目は、固有のリレーショナル・データベース名、および対応する通信パス情報で構成されています。 V5R2 では、アウトバウンド接続について、希望するパスワード・セキュリティーの情報を指定できます。 ARD プログラムの提供するアクセスに対しては、リレーショナル・データベース・ディレクトリー項目に ARD プログラム名を追加する必要があります。

データベースにデータを入力する方法はいろいろあります。 SQL アプリケーション・プログラム、その他の高水準言語アプリケーション・プログラム、または次の方法のいずれかを使用することができます。

- 対話式 SQL
- OS/400 管理機能
- データ・ファイル・ユーティリティー (DFU)
- ファイル・コピー (CPYF) コマンド

iSeries 分散リレーショナル・データベースのセットアップには、以下のトピックに関するいくらかの知識が必要です。

- iSeries server 上の実行管理機能
- ユーザー・リレーショナル・データベースと DRDA の考慮事項
- リレーショナル・データベース・ディレクトリーの使用
- DRDA セキュリティーのセットアップ
- DRDA または APPC 用の TCP/IP サーバーのセットアップ
- 対話式 SQL での SQL パッケージのセットアップ
- DDM ファイルのセットアップ
- 分散リレーショナル・データベースでのテーブルへのデータのロード

この章では、上記のトピックについて紹介し、分散リレーショナル・データベース作業のために iSeries server のセットアップを行う手助けをします。

異種環境システムの分散リレーショナル・データベース・ネットワークに関する接続とセットアップの説明は、 *Distributed Relational Database Cross-Platform Connectivity* (SG24-4311-02) を参照してください。

iSeries server 上の実行管理機能

iSeries server で行う作業はすべて、実行管理機能を通して実行要求されます。 iSeries server 上では、異なるタイプの作業を処理してサーバーの要件を充足するために、特別な操作環境を設計することができます。ただし、オペレーティング・システムをインストールすると、そのオペレーティング・システムには、対話式処理、バッチ処理、通信、およびスプール処理をサポートする実行管理環境が含まれています。

サーバーでは、ユーザー・ジョブはすべてサブシステム記述で定義された**サブシステム**と呼ばれる環境で動作します。サーバーはこの環境で処理やリソースを調整します。共通の特性を持つジョブが同一のサブシステム内に置かれている場合には、ユーザーは、それらのジョブのグループを他のジョブとは別に制御することができます。実行する作業をサポートし、所要のパフォーマンス特性を維持するために、必要に応じて、サブシステムの開始および終了を容易に行うことができます。

サーバー上で実行されるジョブの基本タイプには、対話式、通信、バッチ、スプール、自動開始、および事前開始があります。

対話式ジョブは、ワークステーションにサインオンすると開始し、サインオフすると終了します。 APPC 通信バッチ・ジョブは、別のシステムからプログラム開始要求によって開始されるジョブです。通信以外のバッチ・ジョブは、ジョブ待ち行列から開始されます。通信バッチ・ジョブを開始する時は、ジョブ待ち行列は使用されません。スプーリング機能は、入力と出力の両方で使用可能です。自動開始ジョブは、反復作業または 1 回限りの初期設定作業を行います。自動開始ジョブは、特定のサブシステムに対応しており、サブシステムが開始されると、それに対応する自動開始ジョブが開始されます。事前開始ジョブは、リモート・プログラムがプログラム開始要求を送る前に実行を開始するジョブです。

サブシステムについてのより詳細な情報は、以下のトピックを参照してください。

- DRDA 用の実行管理環境のセットアップ
- APPC 用のサブシステムのセットアップに関する考慮事項

注: V5R2 では、デフォルトで、 DRDA TCP/IP 接続に使用される DDM TCP/IP サーバー事前開始ジョブは、QUSRWRK サブシステムで実行されます。 V5R2 になる前は、これは QSYSWRK で実行されていました。 QUSRWRK は、ユーザーの作業のサブシステムです。これには、ユーザーの作業を代行処理するため、サーバーによって開始されるジョブも含まれます。 QSYSWRK では、事前開始ジョブに作業をまわす、DRDA の「リスナー」ジョブが実行されます。 TCP/IP サーバーのセットアップおよび管理について詳細は、 111 ページの『TCP/IP サーバーの管理』を参照してください。

DRDA 用の実行管理環境のセットアップ

サーバーをロードすると、**制御サブシステム**と呼ばれる 1 つのサブシステムが自動的に始動します。 2 つの制御サブシステムが IBM によって提供され、変更せずに使用することができます。第 1 の構成には、次のようなサブシステムがあります。

- QBASE は制御サブシステムで、対話式ジョブ、バッチ・ジョブ、および通信ジョブをサポートします。
- QSPL は、スプール読み取りプログラムおよびスプール書き出しプログラムの処理をサポートします。
- QSYSWRK は TCP/IP などの様々なサーバー機能をサポートします。
- QUSRWRK は、ユーザーの作業のサブシステムです。これには、ユーザーの作業を代行処理するため、サーバーによって開始されるジョブも含まれます。

QBASE は、サーバーが開始されると、自動的に始動します。 QBASE の中で自動的に開始されたジョブが QSPL を開始します。

提供されている 2 番目の制御サブシステム構成は、最初のものに比べて複雑です。この構成は、以下のサブシステムから成っています。

- QCTL は制御サブシステムで、コンソールで開始される対話式ジョブをサポートします。
- QINTER は、他のワークステーションで開始される対話式ジョブをサポートします。
- QCMN は通信ジョブをサポートします。
- QBATCH はバッチ・ジョブをサポートします。
- QSPL は、スプール読み取りプログラムおよびスプール書き出しプログラムの処理をサポートします。
- QSYSWRK は TCP/IP などの様々なサーバー機能をサポートします。
- QUSRWRK は、ユーザーの作業のサブシステムです。これには、ユーザーの作業を代行処理するため、サーバーによって開始されるジョブも含まれます。

構成を変更して QCTL 制御サブシステムを使用する場合には、このサブシステムはシステムの開始時に自動的に始動します。 QCTL の中で自動的に開始されたジョブがその他のシステムを開始します。

システム値変更 (CHGSYSVAL) コマンドのシステム値 QCTLSBSD (制御サブシステム) を QCTL に変更し、再度システムを開始することによって、サブシステム構成を QBASE から QCTL に変更することができます。

サブシステム記述変更 (CHGSBSD) コマンドを使用することによって、 IBM 提供のサブシステム記述またはユーザー作成のサブシステム記述を変更することができます。 また、このコマンドを使用して、活動サブシステムの記憶域プール・サイズ、記憶域プール活動レベル、およびサブシステム記述の最大ジョブ数を変更することができます。

iSeries server での実行管理機能、サブシステムおよびジョブについては、 iSeries Information Center の『実行管理機能』をご覧ください。また、通信および通信サブシステムの実行管理機能に関する説明については、 V5R1 Supplemental Manuals Web サイトにある、 「*Communications Management*」  を参照してください。

APPC 用のサブシステムのセットアップに関する考慮事項

SNA ネットワークを使用する分散リレーショナル・データベースの場合、管理者が各システムで管理の計画を立てる必要のある主要な作業のタイプは、通信ジョブと対話式ジョブであるといえます。ネットワーク内のサーバーは、通信ジョブを開始して、**アプリケーション・リクエスター (AR)** からの要求を処理します。他のサーバーに対する AR の通信要求は、通常、ローカル・システム上の対話式またはバッチ・ジョブから生じます。分散リレーショナル・データベース・ネットワーク・サーバーに効率的な実行管理環境をセットアップするなら、ネットワーク内の各**アプリケーション・サーバー (AS)** や AR の特定の必要に合わせてシステム・リソースを割り振ることにより、ネットワーク全体のパフォーマンスを高めることができます。

OS/400 ライセンス・プログラムが最初にインストールされた時点では、 QBASE がデフォルトの制御サブシステムです。制御サブシステムとして、QBASE は 2 つのサブシステム、 QBASE と QSPL の間にシステム・リソースを割り振ります。対話式ジョブ、通信ジョブ、バッチ・ジョブなどが、QBASE サブシステム内でリソースを割り振ります。スプール・ジョブだけが別のサブシステム QSPL のもとで管理されます。つまり、通信ジョブを対話式ジョブと対比して処理する場合は、 QCTL 制御サブシステムを使用する場合ほどには、システム・リソースの制御を行えないことを意味します。

QCTL サブシステム構成を使用すれば、システムが記憶域プールや他のシステム・リソースを割り振っている 4 つの追加サブシステムの制御を行うことができます。 QCTL サブシステムの変更、または独自のサブシステムの作成を行えば、処理リソースの柔軟性および制御はさらに増します。

Spiffy 社の分散リレーショナル・データベース・ネットワークの中のシステムの一部でシステム要件が異なる場合、最高のネットワーク効率を得るためには、異なった実行管理環境が必要になることがあります。以下の説明では、分散リレーショナル・データベース管理担当者が実行管理サブシステムをどのように計画すれば、 Spiffy 社の分散リレーショナル・データベース・ネットワークの中の各 iSeries server の要件に適合することができるかを示しています。

Spiffy 社のシステム編成では、販売店が小規模であれば、ユーザーがサーバー上で処理する様々なジョブについて、 QBASE レベルの制御で十分のことがあります。 たとえば、小規模の販売店のリレーショナル・データベースに対する地域 AR からの要求 (出荷に関する販売店在庫レベルの更新) は、通信ジョブとして処理されます。地区に現在在庫していない部品を要求するために、販売店ユーザーが地域 AS に対して行う要求は、販売店サーバーで対話式ジョブとして処理されます。 これらの活動は、販売店が小規模であり、処理する保守サービス受注件数、部品販売高などが少ないので、両方とも相対的に小さいジョブです。QBASE サブシステム内のリソースの調整によって、この企業が対話式要件および通信要件で必要とする制御のレベルが提供されます。

これに対して、販売店の規模が大きくなると、ジョブのタイプが異なるのに対応して作業負荷が異なるので、 QCTL サブシステムによって作業を管理することになると考えられます。

毎日の保守サービス受注件数が多くなる可能性があり、したがって、ローカル・リレーショナル・データベースに対する部品の照会、また販売店に在庫していない部品に関する地域センターの AS に対する照会が必要になります。このタイプの活動は、システム上で対話式ジョブを開始します。販売店では、企業人事記録の保管、営業および販売の計画および報告書作成など、分散リレーショナル・データベース関連ジョブではない多くの対話式ジョブも開始します。この販売店に対する地域センターからのパフォーマンス情報の要求や在庫または作業計画の更新要求は、販売店が別の環境で管理したいと考える通信ジョブです。大規模販売店では、地域センターで在庫切れになっている部品に関する要求を、別の販売店から受信することもあります。

販売店が大規模な場合には、サブシステム管理を QINTER と QCMN とで別にした QCTL 構成によって、サーバー作業環境の管理に関する柔軟性と制御が増します。 この例では、販売店サーバーでの対話式ジョブおよび通信ジョブには、他のタイプのジョブに比べて、多くのサーバー・リソースを割り振ることができます。 さらに、このシステムで、通信ジョブが対話式ジョブに比べて概して少ない場合には、QINTER と QCMN の両方のサブシステム記述を変更することによって、リソースを対話式ジョブに振り向けることができます。

Spiffy 社の地域センターの見地から実行管理環境を調整することも大切です。 Spiffy ネットワークでは、地域センターは、定期部品出荷データによって販売店在庫表を更新したり、特定の修理ジョブに関する新規または更新保守サービス計画によって保守サービス計画表を更新する場合には、各販売店に対して AR になります。これらのジョブの中には、システム使用度が概して低い早朝または午後遅く、対話式ジョブとして実行する (地域システムで) か、または正規の営業時間後にバッチ・ジョブとして実行する (地域サーバーで) ことができるものがあります。 管理担当者は、特定の処理時間およびリソース要件に合わせて、QINTER および QBATCH サブシステムを調整することができます。

販売店に在庫していない部品、特定の保守サービス・ジョブ (ステアリング・ラックの再組み立てなど) に関する保守サービス計画、または販売店リレーショナル・データベースに対する最後の更新以後の技術速報

やリコール通知に関して、販売店が地域リレーショナル・データベースに照会する必要がある時には、地域センターは、各販売店に対して AS になることもあります。これらの通信ジョブは、すべて QCMN で管理されます。

しかし、KC000 (カンザス・シティー) 地域センター、およびその傘下にある販売店による分散リレーショナル・データベース・ネットワーク使用の特定の側面をさらに綿密に検討してみると、カンザス・シティーの分散リレーショナル・データベース管理担当者にとってもっと別の方法があることがわかります。

KC000 サーバーがサービスを提供している販売店には、毎日数百件の保守サービス受注を処理する非常に大規模なものが数社あり、毎日の保守サービス受注処理件数が 20 件未満の小規模なものも少数あります。その他は中規模販売店で、それぞれが毎日 100 件前後の保守サービス受注を処理しています。分散リレーショナル・データベース管理担当者に提示される問題の 1 つは、他のシステムから KC000 サーバーに寄せられるすべての通信要求の公平な処理です。 大規模販売店がその要求で QCMN リソースを支配する可能性があり、ネットワークの中の他のシステムにとっては、応答時間およびコストが不満足な結果になります。

分散リレーショナル・データベース管理担当者は、追加の通信サブシステムを作成することができるので、各クラスの販売店 (小規模、中規模、または大規模) は、いずれも AS に対して支援を要求し、一般的によりよい応答を受け取ることができます。各サブシステム記述のサブシステム属性、事前開始ジョブ項目、通信実行処理項目、および経路指定項目を調整することによって、管理担当者は、あるサブシステム上で活動状態にできるジョブの数、およびそのサブシステムでジョブを処理する方法を制御します。

管理担当者は、経路指定項目を追加することができます。これにより、CMPVAL パラメーターに QCNTEDDM を指定し、また、ジョブの優先順位を制御するクラスを指定することによって、DRDA/DDM ジョブのクラス (したがって優先順位) を変更することができます。次は、その例です。

```
ADDRTGE SBSD(QCMN) SEQNBR(280) CLS(QINTER) CMPVAL('QCNTEDDM' 37)
```

また、管理担当者は、次の例のように、事前開始ジョブとして QCNTEDDM を指定することによって、DRDA/DDM ジョブのための事前開始ジョブを追加することもできます。

```
ADDPJE SBSD(QCMN) PGM(QCNTEDDM)
```

iSeries serverの実行管理機能に関する詳細は、 iSeries Information Center の『実行管理機能』のトピックを参照してください。また、通信の属性、実行処理項目および経路指定項目の変更に関する詳細は、 V5R1 Supplemental Manuals Web サイトにある、 「*Communications Management*」 を参照してください。

ユーザー・リレーショナル・データベースと DRDA の考慮事項

ユーザーは、サーバー上に独立した補助記憶域プールを構成することによって、 iSeries サーバーの上に別のリレーショナル・データベースを作ることができます。独立した補助記憶域プールの各グループが、リレーショナル・データベースを構成するのです。本書では、このようなリレーショナル・データベースを「ユーザー・データベース」と呼んでいます。このデータベースには、独立した補助記憶域プールのグループのディスク上に存在する、すべてのデータベース・オブジェクトが含まれます。加えて、その独立した補助記憶域プールがオンにされた iSeries サーバーのシステム・リレーショナル・データベース (本書ではシステム・データベースと呼びます) にあるデータベース・オブジェクトも、論理的にはすべてユーザー・リレーショナル・データベースに組み込まれます。ただし、コミットメント制御の視点から見た場合に、システム・データベースは扱われ方が異なります。詳細については、iSeries Information Center の『トランザクションとコミットメント制御』のトピックを参照してください。

ユーザー・データベースの作成と使用には、先に述べたコミットメント制御の考慮事項に示される点以外にも、いくつもの規則が関係しています。たとえば、1 つの例として、現行スレッドのユーザー・データベース (補助記憶域プール (ASP) の 1 つのグループ) に設定されている、 AR からデータベースへの接続には、APPC で保護された DUW 会話は使用できません。また、別の例では、ユーザー・データベースで作成する一切のスキーマに対して、そのユーザー・データベースや関連するシステム・データベースにすでに存在する名前を使用することはできません。このような制約についての詳細は、iSeries Information Center の『SQL 解説書』のトピックを参照してください。

ユーザー・データベースには、組み込むことのできない特定の DRDA 関連オブジェクトというものがあります。 DDM ユーザー出口プログラムは、システム・データベース内のライブラリーに常駐していなければなりません。すべてのアプリケーション・リクエスター・ドライバー・プログラムも、これと同様です。

注意する点として、ユーザー・データベースをオンにする処理を行うと、RDB ディレクトリーがある期間使用できなくなるため、ディレクトリーを使おうとする DRDA の**アプリケーション・リクエスター (AR)** や **アプリケーション・サーバー (AS)** による試行が、遅延されたりタイムアウトになったりする可能性があります。それで、同時に複数のユーザー・データベースをオンにするなら、データベースをオンにすることによってディレクトリーが使用できなくなるときに生じる、ディレクトリー操作をタイムアウトにするための障害は、かなり大きなものになります。この後の部分でも述べますが、ユーザー・データベースをはじめてオンにするときには、サーバーによって、そのデータベースの項目の追加が試行されます。データベースをオンにする操作を並行して行うことにより、ディレクトリーが使用できなくなるなら、項目の追加は失敗し、項目を手動で追加しなければならなくなります。

ユーザー・データベースを使用する際に考慮しなければならない別の点は、 RDB ディレクトリー内の項目の構成に関するものです。ユーザー・データベースの命名に関する規則の 1 つには、ユーザー RDB 名を、ネットワーク属性で指定されているシステム名 (ネットワーク属性表示 (DSPNETA) コマンドで表示される) と一致させることはできない、というものがあります。

RDB ディレクトリー内のローカル・ユーザー・データベース項目は、関連付けられているデータベースがはじめてオンにされるときに、自動的に追加されます。これらの項目の作成は、*IP プロトコル・タイプを使用し、LOOPBACK として指定されたリモート・ロケーションで行われます。 LOOPBACK とは、データベースがディレクトリーと同じサーバーにあることを示すものです。ですから、複数のサーバー間で切り替えて使用することを意図したユーザー・データベースには、専用の IP アドレスを関連付けて構成することが、強く推奨されています。切り替え可能なデータベースに専用の IP アドレスがないと、切り替えを行う度に、そのデータベースを参照するすべてのサーバーで、手動によるディレクトリー項目の更新が必要になるからです。専用 IP アドレスを構成する方法についての説明は、iSeries Information Center の『システム管理』の下の『クラスター』トピックにある『Manage application CRG IP addresses (アプリケーション CRG の IP アドレスの管理)』項目を参照してください。ユーザー・データベースの RDB ディレクトリー項目に関する詳細は、『リレーショナル・データベース・ディレクトリーの使用』を参照してください。

リレーショナル・データベース・ディレクトリーの使用

OS/400 プログラムは、リレーショナル・データベース・ディレクトリーを使用して、 iSeries server 上で実行されるアプリケーションでアクセスできるリレーショナル・データベース名を定義して、この接続で SNA または IP が使用されるかどうかを指定し、これらのリレーショナル・データベース名を対応するネットワーク・パラメーターに関連付けます。リレーショナル・データベース・ディレクトリーを使用すると、**アプリケーション・リクエスター (AR)** はアプリケーションからのリレーショナル・データベース名を受け入れ、通信処理のために、この名前を該当するインターネット・プロトコル (IP) アドレスまたはホスト名とポート、あるいはシステム・ネットワーク体系 (SNA) ネットワーク識別コードおよび論理装置 (LU) 名の値に変換することができます。 V5R2 と同じく、RDB ディレクトリーも、ユーザーの優先アウ

トバウンド接続セキュリティー・メカニズムを指定するために使用されます。リレーショナル・データベース・ディレクトリーを使用すると、また、ARD プログラムをリレーショナル・データベース名と関連付けることもできます。

分散リレーショナル・データベース・ネットワークの中の各 iSeries システムごとに、リレーショナル・データベース・ディレクトリーを構成しなければなりません。 システム上に存在するリレーショナル・データベース・ディレクトリーは 1 つだけです。分散リレーショナル・データベース・ネットワークの中の各 AR は、それぞれのリレーショナル・データベース・ディレクトリーの中に、それ自体のローカル・リレーショナル・データベースの項目を 1 つと、その AR がアクセスする各リモートおよびローカル・ユーザー・リレーショナル・データベースごとに項目を 1 つずつ持っていなければなりません。分散リレーショナル・データベース・ネットワークの中にあって、**アプリケーション・サーバー (AS)** としてしか機能することのないシステムの場合は、他のリモート・リレーショナル・データベースのリレーショナル・データベース名がそのディレクトリーに含まれている必要はありません。

ローカルのリレーショナル・データベースに割り当てられるリレーショナル・データベース名は、固有でなければなりません。つまり、ネットワークの中の他のリレーショナル・データベースとは異なっていなければなりません。ディレクトリーの中で他のリレーショナル・データベースに割り当てられる名前は、リモート・リレーショナル・データベースまたはローカル・ユーザー・データベースを識別しなければなりません。リモート RDB の名前は、 AS がそのローカル・システム・データベースを識別するために使用する名前か、または構成済みである場合は、そのユーザー・データベースの 1 つを識別するために使用する名前と一致しなければなりません。 AS のローカル・システム RDB 名前項目が必要なときに存在しないならば、ディレクトリーで自動的に作成されます。ディレクトリーで使用される名前は、ネットワーク属性の表示 (DSPNETA) コマンドによって表示される現行システム名です。

詳細については、以下のトピックを参照してください。

- リレーショナル・データベース・ディレクトリーの処理
- リレーショナル・データベース・ディレクトリーのセットアップ例

リレーショナル・データベース・ディレクトリーの処理

以下に挙げるコマンドは、リレーショナル・データベースを処理できるようにするものです。

ADDRDBDIRE
: リレーショナル・データベース・ディレクトリー項目の追加 (ADDRDBDIRE) コマンド

CHGRDBDIRE
: リレーショナル・データベース・ディレクトリー項目の変更 (CHGRDBDIRE) コマンド

DSPRDBDIRE
: リレーショナル・データベース・ディレクトリー項目の表示 (DSPRDBDIRE) コマンド

RMVRDBDIRE
: リレーショナル・データベース・ディレクトリー項目の除去 (RMVRDBDIRE) コマンド

WRKRDBDIRE
: リレーショナル・データベース・ディレクトリー項目の処理 (WRKRDBDIRE) コマンド

SNA 使用の場合の項目の追加

「RDB ディレクトリー項目の追加 (ADDRDBDIRE)」画面を以下に示します。この画面のプロンプトまたは RDB ディレクトリー項目の追加 (ADDRDBDIRE) コマンドを使用して、リレーショナル・データベース・ディレクトリーに項目を追加することができます。

```
RDB ディレクトリー項目の追加  (ADDRDBDIRE)

選択項目を入力して、実行キーを押してください。

リレーショナル・データベース  . . . . . .  MP311
リモート・ロケーション:
  名前またはアドレス  . . . . . .  MP311
  タイプ  . . . . . .  *SNA          *SNA, *IP
テキスト  . . . . . .  'オーク・ストリートのディーラー'
```

この例では、 MP311 というリモート・ロケーション名を持つサーバーの MP311 という名前のリレーショナル・データベースを、ローカル・サーバー上のリレーショナル・データベース・ディレクトリーに追加するために、入力が行われます。 SNA 接続の場合、リレーショナル・データベースの別名フィールドは、デフォルト値の *NONE のままにしておきます。リモート・ロケーション名は、それを使用するリレーショナル・データベース・ディレクトリー項目の作成前に、定義される必要はありません。ただし、リモート・ロケーション名は、リレーショナル・データベース・ディレクトリー項目がアプリケーションの中で使用される前に定義されなければなりません。リレーショナル・データベース名 (RDB) パラメーターおよびリモート・ロケーション名 (RMTLOCNAME) パラメーターは、 RDB ディレクトリー項目の追加 (ADDRDBDIRE) コマンドでは必須です。 RMTLOCNAME パラメーターの 2 番目の要素は、デフォルトでは *SNA に設定されます。テキスト (TEXT) パラメーターは任意指定です。この例に示してあるように、リレーショナル・データベース名をサーバー名、またはネットワーク構成でこのサーバーに指定されているロケーション名と同じにするのはよい考えです。 このようにすることにより、データベース名を識別するとともに、特にネットワークが複雑な場合に、そのデータベース名を、分散リレーショナル・データベース・ネットワークの中の特定のサーバーに結び付けるのに役立ちます。

上記以外のこのコマンドの任意指定パラメーターを表示するには、「RDB ディレクトリー項目の追加 (ADDRDBDIRE)」画面で F10 を押してください。表示される任意指定パラメーターを下に示します。

```
RDB ディレクトリー項目の追加  (ADDRDBDIRE)

選択項目を入力して、実行キーを押してください。

  リレーショナル・データベース  . . .  > MP311
  リモート・ロケーション :
    名前またはアドレス  . . . . . .  > MP311
    タイプ  . . . . . .  *SNA          *SNA, *IP
  テキスト  . . . . . .  > 'オーク・ストリートのディーラー'

装置 :
  APPC 装置記述  . . . . . .  *LOC          名前 , *LOC
ローカル・ロケーション . . . . . .  *LOC          名前 , *LOC, *NETATR
リモート・ネットワーク識別コード . .  *LOC          名前 , *LOC, *NETATR, *NONE
モード  . . . . . .  *NETATR       名前 , *NETATR
トランザクション・プログラム . . . .  *DRDA         文字値 , *DRDA
```

サーバーは、以下の省略時の *SNA 値を、追加の RDB ディレクトリー項目の追加 (ADDRDBDIRE) コマンド・パラメーターに提供します。

- 装置 (DEV)
- ローカル・ロケーション (LCLLOCNAME)
- リモート・ネットワーク ID (RMTNETID)
- モード (MODE)
- トランザクション・プログラム (TNSPGM)

注:

1. SNA 接続の場合、リレーショナル・データベースの別名フィールドは、デフォルト値の *NONE のままにしておきます。
2. iSeries serverのトランザクション・プログラム名パラメーターは TNSPGM です。 SNA ではこれは TPN です。
3. デフォルトの値を拡張プログラム間通信 (APPC) で使用する場合には、サーバーは、使用される装置、ローカル・ロケーション、およびリモート・ネットワーク ID を決めます。 ネットワーク属性に定義されているモード名が使用され、分散リレーショナル・データベース・アーキテクチャー (DRDA) サポートのトランザクション・プログラム名が使用されます。
4. デフォルトの値をAdvanced Peer-to-Peer Networking (APPN) で使用する場合には、サーバーは、装置 (DEV) パラメーターを無視し、ネットワーク属性に定義されているローカル・ロケーション名、リモート・ネットワーク ID、およびモード名を使用します。

RDB ディレクトリー項目の追加 (ADDRDBDIRE) コマンドの省略時の値は、いずれも変更することができます。たとえば、DB2 UDB for VM サーバーと通信するためには、 TNSPGM パラメーターを変更しなければならない場合があります。 DB2 UDB for VM サポートの省略時解釈では、TNSPGM は接続したい DB2 UDB for VM データベースの名前になります。 DRDA (*DRDA) のデフォルトの TNSPGM パラメーターの値は、X'07F6C4C2' です。 QCNTEDDM および DB2DRDA も X'07F6C4C2' にマップします。トランザクション・プログラム名の詳細については、以下の箇所を参照してください。

- 180 ページの『DB2 UDB for iSeries アプリケーション・リクエスターでの TPN としての QCNTSRVC の設定』
- 181 ページの『DB2 UDB for VM アプリケーション・リクエスターでの TPN としての QCNTSRVC の設定』
- 181 ページの『DB2 UDB for z/OS アプリケーション・リクエスターでの TPN としての QCNTSRVC の設定』
- 181 ページの『DB2 Connect アプリケーション・リクエスターでの TPN としての QCNTSRVC の設定』

TCP/IP 使用の場合の項目の追加

次に示す「RDB ディレクトリー項目の追加 (ADDRDBDIRE)」画面では、 RMTLOCNAME パラメーターの 2 番目の項目に *IP を入力した際のパネルの変更の様子と、 TCP/IP を使用する RDB でどんな一般的な項目が表示されるかが示されています。 TCP/IP を使用する接続でリレーショナル・データベース別名フィールドの使用が可能になっていますが、この最初の TCP/IP の例では、別名を指定しません。

```
RDB ディレクトリー項目の追加  (ADDRDBDIRE)

選択項目を入力して、実行キーを押してください。

リレーショナル・データベース     > MP311
リレーショナル・データベース別名  RDBALS
リモート・ロケーション:
    名前またはアドレス  . . . . . > MP311




    タイプ  . . . . . > *IP          *SNA, *IP
テキスト  . . . . . .  ' オーク・ストリートのディーラー '

ポート番号またはサービス・プログラム*DRDA
リモート認証方式 :
    優先方式  . . . . . .   *ENCRYPTED    *USRID, *USRIDPWD...
    より低い認証の許可  . . . . .   *ALWLOWER     *ALWLOWER, *NOALWLOWER
```

リレーショナル・データベースの別名の指定

次の例は、RDB 別名を指定するディレクトリー項目の追加を示しています。これにより、それぞれのネットワークに同じ名前のリレーショナル・データベースがあっても、 DRDA 環境でそれぞれを一意的に識別できるようになります。また別名を使用する項目が RDB ディレクトリーに追加されている場合は、その別名で項目が識別されます。項目を表示または削除するときは、別名を指定する必要があります。

下の画面では、リレーショナル・データベースの別名として RDBALS が指定されています。

```
選択項目を入力して、実行キーを押してください。

リレーショナル・データベース    > TEST
リレーショナル・データベース別名  RDBALS
リモート・ロケーション:
名前またはアドレス  . . . . . > MP311.spiffy.com



タイプ  . . . . . > *IP          *SNA, *IP
テキスト  . . . . . .  ' オーク・ストリートのディーラー '
```

WRKRDBDIRE およびオプション 1 を使用して別名の項目を追加する場合は、まず、「**項目名 (Entry)**」フィールドに実際の RDB 名を入力し、Enter を押します。次いで、「**リレーショナル・データベース別名**」フィールドに別名を入力し、その他のフィールドにも必要な値を入力すると、 RDB 項目のリストの「**項目名 (Entry)**」フィールドで、実際の RDB 名が別名に置き換えられます。なお、リモート・ロケーション名の「**タイプ**」は *SNA から *IP に変更する必要がありますので、ご注意ください。

別名の項目を除去するときには、実際の RDB 名ではなく、別名を使用して項目の指定を行います。

別名によってリモート・データベースを識別する場合は、同一ディレクトリーの中で、実際の名前も使用してデータベースを参照することはできません。

RMTLOCNAME で MP311.spiffy.com を指定しなくても、 IP アドレス (たとえば、'9.5.25.176') を指定できることに注目してください。別の iSeries server への IP 接続では、ポート 447 を使用する必要がなければ、 PORT パラメーター値を省略時の値である *DRDA のままにしてください。たとえば、IP セキュリティー (IPSec) を使用した伝送用に構成されたポート 447 があるとします。他のプラットフォーム上の IBM ユニバーサル・データベース (UDB) サーバーへの接続では、ポートを 50000 などの数字に設定する必要があるでしょう。詳しいことは、ご使用のサーバーのプロダクト資料を参照してください。 DRDA ポートで有効なサービス名を定義した場合には、数字の代わりにそのサービス名を使用することもできます。ただし、iSeries 上では、*DRDA が 'drda' サービス名の使用より優先されます。

アプリケーション・リクエスター・ドライバー (ARD) の項目の追加

RDB ディレクトリー項目の追加 (ADDRDBDIRE) コマンド・プロンプトで ARD プログラムと通信情報を指定するには、 F9 と次ページ・ボタンを押してください。 ARD プログラムで ADDRDBDIRE コマンドに指定された通信情報を使用しない場合は (通常は使用しない)、 RMTLOCNAME パラメーターに特別な値 *ARDPGM を使用します。 ARD プログラムは、システム・データベース内のライブラリー (ASP 番号 1-32) に置く必要があります。

リレーショナル・データベース・ディレクトリー項目の処理 (WRKRDBDIRE) コマンドの使用

次に示す「リレーショナル・データベース・ディレクトリー項目の処理」画面では、リレーショナル・データベース・ディレクトリー項目の追加、変更、表示、または除去を行うことのできるオプションが用意されています。

```
リレーショナル・データベース・ディレクトリー項目の処理

 位置指定 . . . . . .

オプションを入力して、実行キーを押してください。
1= 追加   2= 変更   4= 除去   5= 明細の表示   6= 明細の印刷

      リレーショナル・     リモート・
OPT   データベース         ロケーション    テキスト
__    KC000               KC000           カンザスシティ地域のデータ・ベース
__    MP000               *LOCAL          ミネアポリス地域のデータ・ベース
__    MP101               MP101           販売店データ・ベース MP101
__    MP102               MP102           販売店データ・ベース MP102
__    MP211               MP211           販売店データ・ベース MP211
__    MP215               MP215           販売店データ・ベース MP215
4_    MP311               MP311           販売店データ・ベース MP311
```

画面に示されているように、オプション 4 を使用すれば、ローカル・サーバーのリレーショナル・データベース・ディレクトリーから項目を除去することができます。 項目を除去する場合には、別の画面が表示されるので、その画面で指定した項目の除去要求を確認するか、または別のリレーショナル・データベース・ディレクトリー項目を選択することができます。 リレーショナル・データベース・ディレクトリー項目の除去 (RMVRDBDIRE) コマンドを使用する場合には、特定のリレーショナル・データベース名、総称名、すべてのディレクトリー項目、またはリモート項目だけのいずれかを指定することもできます。

「リレーショナル・データベース・ディレクトリー項目の処理」画面では、 1 つの項目の明細を表示するオプションを選択することができます。「リレーショナル・データベース・ディレクトリー項目の処理」画面の出力は画面表示です。ただし、リレーショナル・データベース・ディレクトリー項目の表示 (DSPRDBDIRE) コマンドを使用すれば、その出力をプリンターまたは出力ファイルに送ることができます。リレーショナル・データベース・ディレクトリーは iSeries オブジェクトではありません。したがって、出力ファイルを使用することによって、リレーショナル・データベース・ディレクトリーのバックアップ手段が得られます。リレーショナル・データベース・ディレクトリーのバックアップのために出力ファイルを用いる (DSPRDBDIRE) コマンドの使用に関する詳細については、 134 ページの『リレーショナル・データベース・ディレクトリーの保管と復元』を参照してください。

「リレーショナル・データベース・ディレクトリー項目の処理」画面では、リレーショナル・データベース・ディレクトリーの中の 1 つの項目を変更するオプションを選択することができます。または、リレーショナル・データベース・ディレクトリー項目の変更 (CHGRDBDIRE) コマンドを使用しても、ディレクトリーの中の項目に変更を加えることができます。任意指定のコマンド・パラメーターおよびサーバーのリモート・ロケーション名は、変更することができます。 ディレクトリー項目のリレーショナル・データベース名を変更することはできません。ディレクトリーの中のリレーショナル・データベースの名前を変更するには、そのリレーショナル・データベースの項目を除去し、新しいリレーショナル・データベース名の項目を追加します。

注: リモート・ロケーションがリレーショナル・データベース・ディレクトリー項目で変更された場合、遠隔ジャーナルを、 遠隔ジャーナルの除去 (RMVRMTJRN) コマンドまたは QjoRemoveRemoteJournal API を使用して除去し、 遠隔ジャーナルの追加 (ADDRMTJRN) コマンドまたは QjoAddRemoteJournal API を使用して再び追加する必要があります。リモート・ロケーションのタイプ、認証、または他の何かが変更された場合、遠隔ジャーナルは、 遠隔ジャーナルの変更 (CHGRMTJRN) コマンドまたは QjoChangeJournalState API を使用して終了し、再び遠隔ジャーナルの変更 (CHGRMTJRN) コマンドまたは QjoChangeJournalState API を使用して再始動するだけでかまいません。分散ファイル用に変更を使用するには、ノード・グループを削除して再作成し、それからファイルを再作成する必要があります。

***LOCAL ディレクトリー項目**

*LOCAL を含むディレクトリー項目は、ディレクトリー内にそのような項目が 1 つしかないという点で固有であり、これはローカル・システム・データベースの名前を指定します。関連した RDB 名は、SQL CONNECT ステートメントで使用して、ローカル・データベースに接続することができます[1]。この効果は、通常はこの方法で使用する必要はありませんが、 CONNECT RESET SQL ステートメントを使用する場合と似ています。

ただし、ローカル RDB 項目の名前を変更する必要がある場合には、前の段落で説明されているとおり、除去と追加を行う必要があります。しかし、その項目にはシステム全体に適用される DRDA 属性情報が入っているので、ローカル項目を除去する際には特別な注意が必要です。項目を除去しようとすると、CPA3E01 (*LOCAL ディレクトリー項目を除去すると、構成データ (C G) が失われることがあります。) メッセージが出され、その操作を取り消すかまたは継続するかを選択する機会が与えられます。そして、メッセージ・テキストは、その項目が DDM TCP/IP 属性の変更 (CHGDDMTCPA) コマンドで入力された構成データを保管するために使用されていることを知らせます。 *LOCAL 項目が除去されると、構成データが破棄される可能性があり、デフォルトの構成値が使用されるようになります。このデフォルトの値が満足のいくものではない場合には、 CHGDDMTCPA コマンドを使用して構成データを再入力する必要があります。項目を除去する前には CHGDDMTCPA コマンドで指定されている値を記録して、 *LOCAL 項目が削除されて正確なローカル RDB 名が追加された後に復元することができます。

ローカル・ユーザー・データベース用のディレクトリー項目

データベースが 1 つだけのサーバーの場合 (つまり、構成済みの独立補助記憶域プールがない)、 *LOCAL 項目とは、単一のローカル・データベースを指します。複数のデータベースがあるサーバーの場合 (1 つのシステム・データベースと、 1 つ以上のユーザー・データベース)、*LOCAL 項目とは、システム・データベースを指します。ローカル・ユーザー・データベースは、リモート *IP 項目に似た項目によって表されます。主な相違点は、Remote Location フィールドです。データベースが別のサーバーに切り替えられない場合に、このフィールドには通常は語 LOOPBACK が入ります。 LOOPBACK は、ホスト・サーバーの IP アドレスを表します。データベースを切り替えられる場合、接続先のサーバーに関係なく、特定の IP アドレスがデータベースに関連付けられるようにサーバーを構成することが勧められています。専用 IP アドレスを構成する方法についての説明は、iSeries Information Center の『システム管理』の下の『クラスター』トピックにある『Manage application CRG IP addresses (アプリケーション CRG の IP アドレスの管理)』項目を参照してください。この場合、IP アドレスは Remote Location フィールドで使用されます。

LOOPBACK を交換可能データベースに使用すると、それがローカル・サーバーから切り替えられるときはいつも、ユーザーは手動でディレクトリー項目を変更して、LOOPBACK を接続先の新規サーバーの IP アドレスで置き換え、その後データベースが切り替えられて元に戻される場合には LOOPBACK に戻す必要があります。

リレーショナル・データベース・ディレクトリーのセットアップ例

Spiffy 社のネットワークが示す例を見れば、分散リレーショナル・データベース・ネットワークの中のサーバーでリレーショナル・データベース・ディレクトリーを使用する方法が理解でき、それぞれをセットアップする方法が分かります。 この例では、通信に TCP/IP を使用するのではなく、APPC を使用することを

1. プログラム・テストなどのために、ローカル・サーバー・データベースへの DRDA 接続を作成したい場合は、この目的で使用できる ME および MYSELF という、2 つの特殊 RDB 名があります。使用例としては、ME という RDB 名、タイプ *IP、およびリモート・ロケーション名 LOOPBACK のディレクトリー項目をプログラマーが追加する場合などがあります。それからプログラマーは、プログラム内で SQL CONNECT TO ME を実行して、ローカル・システムへのソケット DRDA 接続を確立できます。ただし、これらの RDB 名の一般的な使用は勧められておらず、ある状況ではこの使用の結果として予期しない動作が生じることを警告するためだけに記述しています。

想定しています。この方が、セットアップがより容易と考えられます。ただし、この例の中のある要素は、プロトコルに依存していません。 APPC で使用される必要がある RDB ディレクトリー項目は、 TCP/IP ネットワークでも必要ですが、パラメーターが異なります。この場合、LU 名、装置記述、モード、TPN などが、ホスト名または IP アドレス、およびポート識別によって置き換えられます。

単純な関係を考察するために、次に示すような 2 つの地域支社間の関係を取り上げてみます。

図 8. サーバーが 2 つの場合のリレーショナル・データベース・ディレクトリーのセットアップ

各地域支社のリレーショナル・データベース・ディレクトリーには、ローカル・リレーショナル・データベースの項目とリモート・リレーショナル・データベースの項目が入っていなければなりません。その理由は、各サーバーがそれぞれ**アプリケーション・リクエスター (AR)** と**アプリケーション・サーバー (AS)** の両方になるからです。 MP000 サーバーのリレーショナル・データベース・ディレクトリーを作成するためのコマンドは、次のようになります。

```
ADDRDBDIRE    RDB(MP000) RMTLOCNAME(*LOCAL) TEXT('Minneapolis region database')

ADDRDBDIRE    RDB(KC000) RMTLOCNAME(KC000) TEXT('Kansas City region database')
```

上の例では、MP000 サーバーは、RMTLOCNAME パラメーターに *LOCAL を指定することによって、それ自体をローカル・リレーショナル・データベースとして識別します。 リレーショナル・データベースは、1 つの iSeries server に 1 つだけあります。 ディレクトリーの中のリレーショナル・データベース名を、ローカル・サーバーのサーバー名およびローカル・ロケーション名と同じにし、リモート・サーバーのリモート・ロケーション名と同じにすれば、ネットワーク・リレーショナル・データベースを簡単に識別することができます。

注: サーバー名は、 ネットワーク属性変更 (CHGNETA) コマンドの SYSNAME パラメーターで指定します。ローカル・サーバーは、通信構成時に、 CHGNETA コマンドの LCLLOCNAME パラメーターで識別します。 SNA (APPC) を使用するリモート・ロケーションは、通信構成時に、制御装置記述の作成 (APPC) (CRTCTLAPPC) コマンドの RMTCPNAME パラメーターで識別します。これらのコマンドの使用例については、『例: 分散リレーショナル・データベースのための APPN 構成』を参照してください。サーバー名、ネットワーク・ロケーション、およびデータベース名に同じ名前を使用すれば、特に複雑なネットワークの場合に、混乱を避ける上で役立ちます。

KC000 サーバーのリレーショナル・データベース・ディレクトリーの場合の対応する項目は、次の通りになります。

```
ADDRDBDIRE    RDB(KC000) RMTLOCNAME(*LOCAL) TEXT('Kansas City region database')

ADDRDBDIRE    RDB(MP000) RMTLOCNAME(MP000) TEXT('Minneapolis region database')
```

複雑な例を考察するために地域支社とその傘下の販売店の関係を取り上げます。たとえば、次に示すネットワーク内のリレーショナル・データベースをアクセスするには、 MP000 サーバーのリレーショナル・データベース・ディレクトリーを拡張して、その傘下の各販売店の項目を含めなければなりません。

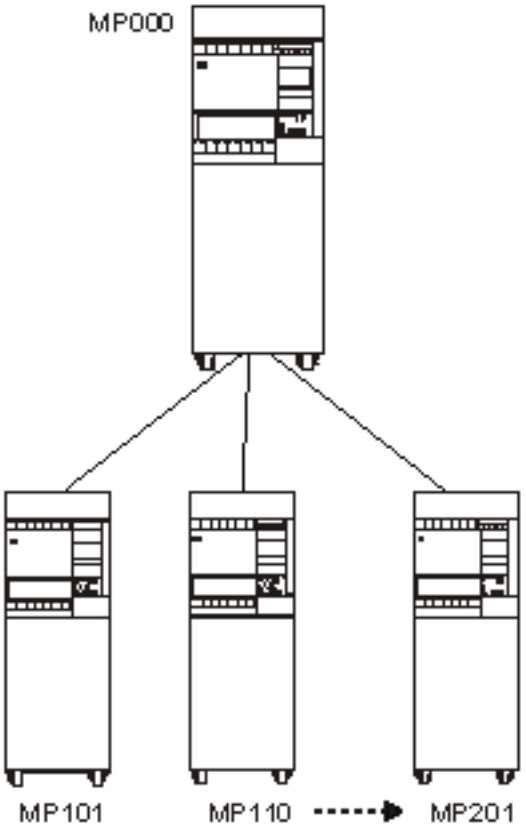


図 9. サーバーが多数の場合のリレーショナル・データベース・ディレクトリーのセットアップ

MP000 リレーショナル・データベース・ディレクトリーに、傘下の販売店データベースをすべて含めるために使用するコマンドの例を以下に示します。

注: xv ページの『コードの特記事項情報』に重要なリーガル情報がありますので、お読みください。

```
PGM
ADDRDBDIRE    RDB(MP000) RMTLOCNAME(*LOCAL) +
TEXT('Minneapolis region database')
ADDRDBDIRE    RDB(KC000) RMTLOCNAME(KC000)
TEXT('Kansas City region database')
ADDRDBDIRE    RDB(MP101) RMTLOCNAME(MP101)
TEXT('Dealer database MP101')
ADDRDBDIRE    RDB(MP002) RMTLOCNAME(MP110)
TEXT('Dealer database MP110')
.
.
.
ADDRDBDIRE    RDB(MP215) RMTLOCNAME(MP201)
TEXT('Dealer database MP201')
ENDPGM
```

上記の例では、地域の各販売店は、リモート・リレーショナル・データベースとしてミネアポリスのリレーショナル・データベース・ディレクトリーに含められます。

各販売店は、MP000 および他の販売店アプリケーション・サーバーに対して AR として機能するので、各販売店は、それぞれそれ自体をローカル・リレーショナル・データベースとし、地域支社および他のすべての販売店をリモート・リレーショナル・データベースとする項目を持つリレーショナル・データベース・ディレクトリーを持たなければなりません。データベース管理担当者は、各販売店のサーバーのリレーショナル・データベース・ディレクトリーを作成するにあたって、複数のオプションから選択することができます。

最も時間を要し、しかもエラーを生じる可能性が最も高いのは、 MP000 分散リレーショナル・データベース・ネットワークに属するすべてのサーバー上で各ディレクトリー項目を作成するための RDB ディレクトリー項目の追加 (ADDRDBDIRE) コマンドを使用することによって、各サーバーにリレーショナル・データベース・ディレクトリーを作成する方法です。

それに代わる方法として優れているのは、上記の例で MP000 の場合に示してあるような制御言語 (CL) プログラムを作成する方法です。分散リレーショナル・データベース管理担当者は、各販売店のサーバーごとにこの CL プログラムをコピーすることができます。 このプログラムを各販売店に応じてカスタマイズするには、データベース管理担当者は、MP000 サーバーのリモート・ロケーション名を MP000 に変更し、地区販売店のリモート・ロケーション名を *LOCAL に変更します。 分散リレーショナル・データベース管理担当者は、カスタマイズした CL プログラムを各販売店に配布し、そのサーバーで実行させて、それぞれ固有のリレーショナル・データベース・ディレクトリーを作成させることができます。

3 番目の方法としては、 リレーショナル・データベース・ディレクトリー項目の表示 (DSPRDBDIRE) コマンドを使用した結果として出力ファイルに送られるリレーショナル・データベース・ディレクトリー情報を読み取るプログラムを作成する方法があります。このプログラムは、 MP000 サーバーのリレーショナル・データベース・ディレクトリー項目が入っている出力ファイルとともに、販売店に配布することができます。 各サーバーでは、MP000 出力ファイルを読み取って、ローカル・リレーショナル・データベース・ディレクトリーを作成することができます。 次に リレーショナル・データベース・ディレクトリー項目の変更 (CHGRDBDIRE) コマンドを使用して、 MP000 システムのディレクトリーをローカル・システムに応じてカスタマイズできます。出力ファイルの使用によるリレーショナル・データベース・ディレクトリー項目の作成に関する詳細については、 134 ページの『リレーショナル・データベース・ディレクトリーの保管と復元』を参照してください。

DRDA セキュリティーのセットアップ

Distributed Relational Database Architecture™ (DRDA) セキュリティーについては、 39 ページの『第 4 章 iSeries 分散リレーショナル・データベースのセキュリティー』で扱われています。しかし、その記述をより充実したものにするため、DRDA を使用する前、またはネットワークで拡張プログラム間通信 (APPC) の使用から伝送制御プロトコル/インターネット・プロトコル (TCP/IP) の使用に変更する前の考慮事項としてここに記載しています。

TCP/IP のセキュリティー・セットアップは、APPC で必要とされるものとはかなり異なっています。念頭に置くべき 1 つの点は、APPC の「保護ロケーション」という概念が存在しないということです。 TCP/IP サーバーでは、クライアント・サーバーが主張しているとおりのものであることを信用することはできないので、接続要求でのパスワードの使用はより重要になります。 接続要求でのパスワードの送信をより容易なものとするため、TCP/IP サポートでは特定のユーザー・プロファイルに関連したサーバー権限リストの使用が導入されました。サーバー権限リストの項目は、 xxxSVRAUTHE コマンド (xxx は ADD、CHG、および RMV を表す) を使用して保守できます。このことは、 39 ページの『第 4 章 iSeries 分散リレーショナル・データベースのセキュリティー』、および iSeries Information Center の『制御言語 (CL)』で説明

されています。 サーバー権限項目の使用に代わる別の方法は、 SQL CONNECT ステートメントの USER/USING 形式を使用して、接続要求でパスワードを送信することです。

V5R2 では、Kerberos サポートが追加されました。これは TCP/IP を使用する場合に、さらに別のセキュリティー・オプションを提供します。 Kerberos の構成方法については、 iSeries Information Center の『ネットワーク認証サービス』を参照してください。

サーバー側のセットアップでは、着信する接続要求でどのレベルのセキュリティーを必要とするかを決定して指定することも含まれています。たとえば、暗号化されていないパスワードを使用するかどうかなどです。デフォルトの設定では、パスワードは必要です。 DDM TCP/IP 属性の変更 (CHGDDMTCPA) コマンドを使用すると、この設定を変更することができます。

DRDA 用の TCP/IP サーバーのセットアップ

TCP/IP プロトコルを使用する DRDA **アプリケーション・サーバー (AS)** を所有する場合、 DDM TCP/IP サーバーをセットアップする必要があります。これは、必要な時にサーバーが開始されるようにすることによって簡単に行えます。そして、次のコマンドを実行して、サーバーがいつも活動状態であるようにできます。

```
CHGDDMTCPA AUTOSTART(*YES)
```

サーバーをご使用の環境に調整するために、他のパラメーターも変更することを望まれるかもしれません。これらには、開始される事前開始ジョブの最初の数、ジョブの最大数、ジョブをさらに開始する際の限界値などがあります。このことについて詳細は、 111 ページの『TCP/IP サーバーの管理』を参照してください。

すべてのクライアントが接続の際に使用する共通ユーザー・プロファイルをセットアップすることもでき、リモート・ユーザーの様々なクラスで異なったセキュリティー・レベルを持つ様々なユーザー・プロファイルをセットアップすることもできます。それから、**アプリケーション・リクエスター (AR)** で サーバー認証項目の追加 (ADDSVRAUTE) コマンドを使用して、AR にある各ユーザーのプロファイル名を、 AS でそれらのユーザーが実行されるユーザー・プロファイルに対応付けます。詳細については、『認証方式の折衝』を参照してください。

対話式 SQL (ISQL) での SQL パッケージのセットアップ

このセクションは、非 iSeries アプリケーション・サーバーのみに適用されます。

以下のいずれかが該当する場合、 SQL パッケージがサーバーで作成されていることを確認する必要があります。

- DB2 UDB Query Manager and SQL Development Kit があり、その製品の対話式 SQL (STRSQL) 機能を使用する計画をしている場合。
- V5R1 より前の iSeries クライアントの TCP/IP を使用する非 iSeries DRDA サーバーの接続を計画している場合、または 2 フェーズ・コミット機能がない非 iSeries DRDA サーバーの接続を計画している場合。

STRSQL は、iSeries server 用の SQL パッケージを必要としません。通常の場合、 iSeries ではない**アプリケーション・サーバー (AS)** では、 STRSQL のユーザー用に SQL パッケージが自動的に作成されます。しかし、STRSQL の最初の接続はローカル・サーバーに対して行われ、その接続は 2 フェーズ・コミット・プロトコルによって保護されているので、問題が生じる可能性があります。 システムへの後続の接続が 1 フェーズ・コミットしか可能でなかったり、 V5R1 より前の iSeries クライアントから TCP/IP が使

用されたりすると、その接続は読み取り専用になります。そのような接続でパッケージを自動的に作成しようとすると、パッケージの作成は更新と見なされ、読み取り専用の接続では行えないので失敗します。

この問題の解決策は、リモート AS に接続する前にローカル・データベースへの接続を解除することです。これは、RELEASE ALL コマンドの後に COMMIT を実行することによって行えます。それから、リモート・サーバーへの接続を行うことができ (最初の接続なので)、そこで更新を行うことができます。

対話式 SQL を開始する際には、 *NONE 以外のコミットメント制御レベルを指定する必要があります。さらに、接続するために使用するユーザー ID は、アプリケーション・サーバーで SQL パッケージを作成するための適切な権限がなければなりません。接続を試みる際に SQLSTATE 42501 を受け取る場合には、パッケージ作成権限を持っていないのかもしれません。

詳しくは 250 ページの『異種環境アプリケーション・サーバー上での対話式 SQL および Query 管理機能のセットアップ』をご覧ください。

DDM ファイルのセットアップ

iSeries server での DRDA サポートの実装では、通信のために分散データ管理機能 (DDM) 会話を使用します。 したがって、分散リレーショナル・データベース処理に関連して DDM を使用することができます。DDM を使用して、**アプリケーション・サーバー (AS)** にリモート・コマンドを投入し、 iSeries server 間でテーブルをコピーし、非分散リレーショナル・データベース作業を別のサーバーで処理することができます。

分散リレーショナル・データベースを使用すると、**アプリケーション・リクエスター (AR)** がデータベースとの接続に必要とする情報は、リレーショナル・データベース・ディレクトリーの中で提供されます。DDM を使用する時には、アプリケーション・サーバー (AS) 上で処理したい各ファイルごとに、別々の DDM ファイルを作成しなければなりません。 DDM ファイルは、アプリケーション・サーバー (AS) 上のリモート・ファイルおよびアプリケーション・サーバー (AS) への通信経路を識別するために、アプリケーション・リクエスター (AR) のアプリケーションによって使用されます。

V5R2 の場合と同様、RDB ディレクトリー項目への参照がある DDM ファイルも作成することができます。 『第 6 章 分散リレーショナル・データベースの管理および操作の作業』に説明してあるデータベース管理タスクには、 DDM を使用してリモート・ファイルにアクセスするものがあります。 DDM ファイルは、分散データ管理機能ファイルの作成 (CRTDDMF) コマンドを使用して作成します。 DDM ファイルは、ファイルおよびファイルの中で名前を指定される通信経路が作成される前に、作成することができます。ただし、DDM ファイルの中で名前を指定されるファイル、および通信情報は、アプリケーションによる DDM ファイルの使用前に、作成されなければなりません。

以下に、DDM ファイルの作成方法の一例を示します。

```
CRTDDMF FILE (TEST/KC105TST) RMTLOCNAME(KC105)
        RMTFILE(SPIFFY/INVENT)
```

上記の例の DDM ファイル・アクセスが TCP/IP を介して行われる場合には、 RMTLOCNAME パラメーターの 2 番目の項目で *IP を指定する必要があります。

このコマンドでは、KC105TST という名前の DDM ファイルを作成し、それをアプリケーション・リクエスター (AR) の TEST ライブラリーに保管します。この DDM ファイルは、リモート・ロケーション KC105 を使用して、ターゲット iSeries server の SPIFFY ライブラリーに保管されている INVENT という名前のリモート・ファイルにアクセスします。

「DDM ファイルの処理」画面のオプションを使用して、 DDM ファイルを変更、削除、表示、または作成することができます。

DDM ファイルの使用に関しては、iSeries Information Center の『分散データ管理』を参照してください。

分散リレーショナル・データベースでのテーブルへのデータのロード

分散リレーショナル・データベース環境のアプリケーションは、テーブルに保管されているデータに対して操作を行います。一般的に、アプリケーションを使用してテーブルに情報を照会し、 1 つ以上のテーブルの行の挿入、更新、または削除を行い、あるいは新しいテーブルを作成します。その他の状況が生じて、 1 つのサーバーのデータを別のサーバーに移動しなければならない場合もあります。

このセクションでは、以下の作業で使用できる多くの方式を説明しています。

- 分散リレーショナル・データベース・テーブルへの新しいデータのロード
- iSeries server システム相互間でのデータの移動
- 非 iSeries server から iSeries server へのデータベースの移動

分散リレーショナル・データベース・テーブルへの新しいデータのロード

テーブルへのデータのロードは、各データ項目をテーブルに入力することによって行います。 iSeries server では、SQL、DB2 UDB for iSeries Query 管理機能の機能、または iSeries アプリケーション開発ツールのデータ・ファイル・ユーティリティー部分を使用して、データをテーブルに挿入するアプリケーションを作成することができます。

SQL の使用によるテーブルへのデータのロード

データをテーブルにロードする単純な方法としては、 SQL アプリケーションおよび SQL INSERT 操作を使用する方法があります。

Spiffy 社では、地域センターから販売店へ定期的な在庫品出荷が行われているので、ある地域センターがある販売店の在庫テーブルに在庫品目を定期的に追加する必要がある状況を考えてみます。

```
INSERT INTO SPIFFY.INVENT
   (PART, DESC, QTY, PRICE)
 VALUES
  ('1234567', 'LUG NUT', 25, 1.15 )
```

上記のステートメントでは、 SPIFFY という名前の SQL コレクションの中の INVENT という名前のテーブルに 1 行のデータを挿入します。

各定期出荷品目ごとに、SQL INSERT ステートメントによって、販売店の在庫テーブルに 1 行が挿入されます。上記の例で、15 の異なる品目が販売店に出荷される場合であれば、地域支社のアプリケーションには、 15 の SQL INSERT ステートメントまたはホスト変数を使用する単一の SQL INSERT ステートメントを組み込むことができます。

この例では、地域センターは SQL アプリケーションを使用して、**アプリケーション・サーバー (AS)** にあるテーブルにデータをロードしています。 SQL の実行時のサポートは OS/400 ライセンス・プログラムの中で提供されているので、 AS では IBM DB2 Query Manager and SQL Development Kit for iSeries ライセンス・プログラムは必要ありません。 ただし、アプリケーションを作成するためには、 IBM DB2 Query Manager and SQL Development Kit for iSeries ライセンス・プログラムが必要です。 SQL プログラミング言語についての説明は、 iSeries Information Center の 『SQL プログラミング 概念』と『SQL 解説書』を参照してください。

iSeries Query 管理機能の機能を使用したテーブルおよびファイル内のデータ操作

OS/400 ライセンス・プログラムでは、DB2 UDB for iSeries Query 管理機能が提供されており、この機能を使用すれば、テーブルおよびファイルの中のデータを操作することができます。照会は、SQL 照会ステートメントを使用して作成します。 CL コマンドによって、あるいはアプリケーション・プログラムの中で照会呼び出し可能インターフェースを用いることによって、照会を実行することができます。 Query 管理機能を使用すると、前のセクションで説明した在庫更新の場合に、次のようにして 1 行のデータをテーブルに挿入することができます。

ソース・メンバー INVLOAD をソース物理ファイル INVLOAD の中に作成し、次の SQL ステートメントを作成します。

注: xv ページの『コードの特記事項情報』に重要なリーガル情報がありますので、お読みください。

```
INSERT INTO SPIFFY/INVENT
   (PART, DESC, QTY, PRICE)
 VALUES
   (&PARTVALUE, &DESCVALUE, &QTYVALUE, &PRICEVALUE)
```

CL コマンドを使用して、Query 管理機能プログラム・オブジェクトを作成します。

```
CRTQMQRY QMQRY(INVLOAD) SRCFILE(INVLOAD) SRCMBR(INVLOAD)
```

次の CL コマンドを実行すると、 INSERT SQL ステートメントの結果が SPIFFY コレクションの INVENT テーブルに入ります。 Query の中で変数 (&PARTVALUE、&DESCVALUE、など) を使用すると、所要の値を STRQMQRY 呼び出しの一部として入力することができ、各行ごとに Query 管理機能プログラムを作成する必要はありません。

```
STRQMQRY QMQRY(INVLOAD) RDB(KC000)
   SETVAR((PARTVALUE '1134567'') (DESCVALUE '''Lug Nut''')
    (QTYVALUE 25) (PRICEVALUE 1.15))
```

Query 管理機能は動的です。つまり、そのアクセス・パスは、プログラムのコンパイル時ではなく、実行時に作成されることを意味します。したがって、DB2 UDB for iSeries Query 管理機能は、データをテーブルにロードする場合には、 SQL アプリケーションほど効率的に機能しません。ただし、アプリケーションを作成するには、 IBM DB2 Query Manager and SQL Development Kit for iSeries プロダクトが必要です。 SQL および Query 管理機能の実行時サポートは、 OS/400 ライセンス・プログラムの一部になっています。

Query 管理機能についての詳細は、 *「Query 管理機能 プログラミング」* を参照してください。

データ・ファイル・ユーティリティーを使用したデータの入力、テーブルの更新、および照会

データ・ファイル・ユーティリティー (DFU) は、 IBM により提供される使用可能な iSeries アプリケーション開発ツール・パッケージの一部であり、データを入力し、テーブルを更新し、照会を行うためのプログラムの作成を援助するプログラム作成プログラムです。DFU の使用には、プログラミング言語を必要としません。データ入力、保守、または照会プログラムは、一連の画面に対して応答することにより作成されます。 DFU を使用する利点の 1 つは、それが本質的に汎用的であるために、 SQL などのプログラミング言語を使用する場合に比べて、迅速にデータをテーブルにロードするためのデータベース更新プログラムを作成できるという点にあります。 DDM ファイルと DFU を使用するか、または表示装置パススルーを使用してアプリケーション・ソース (AS) で DFU を実行することによって、リモート・サーバーにあるデータを処理することができます。

DFU プログラム生成プログラムの詳細については、V5R1 Supplemental Manuals Web サイトにある、

「適用業務開発ツールセット/400 データ・ファイル・ユーティリティー (DFU)」  を参照してください。

iSeries server システム相互間でのデータの移動

企業の運営においては、 iSeries server 相互間でデータを移動する必要のあるような状況が数多く発生します。 たとえば、新しい販売店が地域内に開設され、既設の販売店の得意先の中には、所在地の都合によって、新設の販売店に移る場合もあり得ます。販売店が閉業したり、 Spiffy 社の販売および保守サービスの代理業務を停止する場合もあるでしょう。そのような販売店の在庫および必要な保守サービス情報については、地域支社または他の域内販売店に割り振らなければなりません。販売店によっては、成長が著しく、サーバーのアップグレードが必要になり、データベース全体を新しいサーバーに移動しなければならないことも考えられます。

iSeries server 相互間でデータを移動する方法としては、以下に挙げるものがあります。

- ユーザー作成のアプリケーション・プログラム
- 対話式 SQL (ISQL)
- DB2 UDB for iSeries Query 管理機能の機能
- テープまたはディスケット装置相互間でのコピー
- DDM の使用によるファイル・コピー・コマンド
- ネットワーク・ファイル・コマンド
- iSeries server の保管および復元コマンド

ユーザー作成アプリケーション・プログラムの作成

DUW 接続管理でコンパイルしたプログラムは、リモート・データベースとローカル・データベースに接続することができ、一方のシステムから FETCH し、もう一方に INSERT して、データを移動することができます。複数行 FETCH と複数行 INSERT を使用すると、レコードのブロックを一時に処理することができます。コミットメント制御を使用して、データの移動中にある時点でチェックポイントを取り、障害が起きたときにコピーを開始しなくても済むようにすることができます。

対話式 SQL を使用したデータベースの照会

SQL SELECT ステートメントおよび対話式 SQL を使用し、ローカル・サーバー上にあるテーブルの作成または更新を行うのに必要なデータに関し、別の iSeries server 上にあるデータベースに照会することができます。 SELECT ステートメントでは、所要のデータが入っているテーブル名と列、およびどの行のデータを検索するのかを決める選択基準またはフィルターを指定することができます。 SELECT ステートメントが正常に実行された場合には、指定したテーブルの 1 行または複数行が取り出される結果になります。

SQL を使用すると、1 つのテーブルからデータを取り出すだけでなく、結合操作を使用することによって、複数のテーブルに含まれている列から情報を取り出すことができます。 SELECT ステートメントが正常に実行された場合には、指定したテーブル (複数可) の 1 行以上が取り出される結果になります。戻される行の列内のデータ値は、指定した表に含まれているデータ値を合成したものになります。

対話式 SQL 照会を使用すると、照会の結果は、ローカル・サーバーのデータベース・ファイルに入れることができます。 コミットメント制御レベルが対話式 SQL 処理に指定された場合、それは**アプリケーション・サーバー (AS)** に適用されます。ローカル・サーバーのデータベース・ファイルのコミットメント制御レベルは *NONE になります。

対話式 SQL を使用すると、次のことを行うことができます。

- 選択の結果を入れる新しいファイルを作成する。
- 既存のファイルを置き換える。
- ファイル内に新しいメンバーを作成する。
- メンバーを置き換える。
- 結果を既存のメンバーに追加する。

KC105 販売店が部品番号 '1234567' の在庫全量を KC110 に転送する状況を考えてみます。 KC110 では、KC105 から調達する部品について、KC105 データベースに照会します。この在庫照会の結果は、KC110 サーバーですでに存在しているデータベース・ファイルに戻されます。 以下にこの作業を完了するために使用できる処理を示します。

SQL 対話式セッションの開始 (STRSQL) コマンドを使用して、「対話式 SQL」画面を表示します。新しいデータベースの SQL ステートメント (CONNECT 以外) を入力する前に、以下のステップを行うことによって、この操作の結果がローカル・サーバー上のデータベース・ファイルに送られるように指定します。

1. 「SQL ステートメントの入力」画面でサービスのオプションを選択してください。
2. 「サービス」画面でセッション属性変更のオプションを選択してください。
3. 「セッション属性」画面で出力装置選択のオプションを入力してください。
4. 出力装置フィールドにデータベース・ファイルを表す 3 を入力して、実行キーを押してください。次の画面が表示されます。

```
セッション属性変更

選択項目を入力して、実行キーを押してください。

   ステートメント処理 . . . . . .   *RUN           *RUN, *VLD, *SYN
   出力の選択 . . . . . . . . . .   3              1= 表示,  2= 印刷装置
                                                  3= ファイル

   出力ファイル :
     ファイル . . . . . . . . . .   QSQLSELECT     名前
       ライブラリー . . . . . . .     QGPL         名前
     メンバー . . . . . . . . . .   *FILE          名前,  *FILE, *FIRST

     オプション . . . . . . . . .   1              1= ファイルの作成
                                                  2= ファイルの置き換え
                                                  3= メンバーの作成
                                                  4= メンバーの置き換え
                                                  5= メンバーへの追加

     権限 . . . . . . . . . . . .   *LIBCRTAUT     権限リスト名
                                                  *LIBCRTAUT, *CHANGE, *ALL
                                                  *EXCLUDE, *USE

     テキスト . . . . . . . . . .


 F3= 終了    F4=プロンプト    F5= 最新表示    F12= 取り消し
```

5. 結果を受け取るデータベース・ファイルの名前を指定してください。

データベース名が指定されたら、次の例に示すように対話式 SQL 処理を開始することができます。

```
SQL ステートメントの入力

SQL ステートメントを入力して、実行キーを押してください。
現在の接続相手はリレーショナル・データベース KC000 である。
CONNECT TO KC105_____________________________________________________________
現在の接続相手はリレーショナル・データベース KC105 である。
====> SELECT * FROM INVENTORY_____________________________________________________
WHERE PART = '1234567'_____________________________________________________________
_____________________________________________________________________________
_____________________________________________________________________________
_____________________________________________________________________________
_____________________________________________________________________________
_____________________________________________________________________________
_____________________________________________________________________________
_____________________________________________________________________________
_____________________________________________________________________________
_____________________________________________________________________________
_____________________________________________________________________________
_____________________________________________________________________________
_____________________________________________________________________________
_____________________________________________________________________________
 終わり
 F3= 終了     F4= プロンプト     F6= 行の挿入     F9= コマンドの複写     F10= 行のコピー
 F12= 取り消し     F13= サービス     F24= キーの続き
```

SQL プログラミング言語および対話式 SQL についての説明は、 iSeries Information Center の 『SQL プログラミング 概念』と『SQL 解説書』を参照してください。

DB2 UDB for iSeries Query 管理機能の機能を使用したリモート・サーバーの照会

リモート・サーバーに照会して、出力ファイルの中の結果をローカル・サーバーに戻す場合には、 DB2 UDB for iSeries Query 管理機能の機能は、対話式 SQL とほとんど同じサポートを提供します。

対話式 SQL と Query 管理機能は、両方ともファイルまたはテーブルのデータ処理操作 (INSERT、DELETE、SELECT など) を実行することができ、そのためにテーブル (またはファイル) がコレクションの中にすでに存在している (ライブラリーの中に存在することができる) ことを必要としません。また、Query 管理機能では、新しいテーブルが照会の結果としてサーバー上に作成される時には、 SQL CREATE TABLE ステートメントを使用してデータ定義を提供します。 Query 管理機能によって作成される表は、SQL を使用して作成される表に適用されるものと同じ指針および制約条件に従います。

ただし、Query 管理機能を使用する場合には、結果をファイルまたはテーブルに追加したい時に、メンバーを指定することはできません。 Query 管理機能を使用した場合の結果は、 データベース・ファイルの一時変更 (OVRDBF) コマンドを使用して、Query 管理機能を開始する前に別のメンバーを指定しない限り、最初のファイル・メンバーに入れられることになります。

Query 管理機能についての詳細は、 「Query 管理機能 プログラミング」を参照してください。

テープまたはディスケット相互間でのファイルのコピー

iSeries server 上でテープへのコピー (CPYTOTAP) およびディスケットへのコピー (CPYTODKT) コマンドを使用して、テーブルまたはファイルをテープまたはディスケットへコピーすることができます。

テープまたはディスケット上のデータは、 テープからのコピー (CPYFRMTAP) およびディスケットからのコピー (CPYFRMDKT) コマンドを使用して、別のサーバーへロードすることができます。これらのコマンドの使用に関する詳細は、『ストレージ・ソリューション』のトピックを参照してください。

さらに、ファイルのコピー (CPYF) コマンドを使用して、テープ上のデータを DB2 UDB for iSeries にロードできます。これは、DB2 UDB for z/OS または DB2 UDB Server for VM (SQL/DS™) からアンロードされたデータをロードするときに特に便利です。これらのサーバーからヌル値可能データをアンロードし

て、ヌル値可能フィールドに単一バイト・フラグを関連付けることができます。 CPYF の FMTOPT パラメーターで *NULLFLAGS オプションが指定されていると、ヌル・フラグが識別されて、テープの隣接するフィールドのデータを無視します。そして DB2 UDB for iSeries のフィールドをヌルにします。 IBM メインフレームからデータをインポートする際に役立つもう 1 つの FMTOPT パラメーター値は、*CVTFLOAT 値です。これは、System/390 形式でテープ上に保管されている浮動小数点データを、 DB2 UDB for iSeries で使用されている IEEE 形式に変換することを可能にします。

コピー・ファイル・コマンドを使用した iSeries server 相互間のデータの移動

iSeries server 相互間でデータを移動する他の方法としては、ファイル・コマンドを DDM とともに使用して、データをコピーする方法があります。 ファイルのコピー (CPYF)、 ソース・ファイルのコピー (CPYSRCF)、および 照会ファイルからのコピー (CPYFRMQRYF) コマンドを使用して、ソース・システムとアプリケーション・ソース (AS) のファイル相互間でデータをコピーすることができます。リモート・データベース・ファイルとの間でローカル・リレーショナル・データベース・ファイルまたは装置ファイルをコピーすることができ、リモート・ファイルをリモート・ファイルへコピーすることもできます。

たとえば、ある販売店が閉業した場合には、分散リレーショナル・データベース管理担当者は、リモート・サーバーから地域のローカル・サーバーへ得意先テーブルおよび在庫テーブルをコピーすることができます。 管理担当者は、これらのテーブルをアクセスおよびコピーするために、適正に許可されたユーザー・プロファイルをアプリケーション・ソース (AS) 上で必要とし、コピーする各テーブルまたは各ファイルごとに、 DDM ファイルをアプリケーション・リクエスター (AR) 上に作成しなければなりません。次の例には、SPIFFY と呼ばれるコレクションの中の INVENT と呼ばれるテーブルを、 KC105 というリモート・ロケーション名を持つサーバーから、 KC000 と呼ばれる地域中央サーバーへコピーする場合に、データベース管理担当者が使用するコマンドが示してあります。 アプリケーション・リクエスター (AR) KC000 上の TEST と呼ばれるライブラリーの中の INCOPY と呼ばれる DDM ファイルが、ファイル・アクセス用として使用されます。これらのコマンドは KC000 サーバー上で実行されます。

```
CRTDDMF   FILE(TEST/INCOPY) RMTFILE(SPIFFY/INVENT)
   RMTLOCNAME(KC105)
CPYF      FROMFILE(TEST/INCOPY)  TOFILE(TEST/INVENTDDM)
   MBROPT(*ADD)
```

この例では、管理担当者は KC000 サーバー側でコマンドを実行します。 管理担当者が KC000 サーバー側にいない場合には、パススルーを使用して、KC000 サーバーでこれらのコマンドを実行しなければなりません。 SBMRMTCMD (リモート・コマンド投入) コマンドを使用して上記のコマンドを実行することはできません。 iSeries server が、同一のジョブに関して、アプリケーション・リクエスター (AR) およびアプリケーション・ソース (AS) になることはできないからです。

このコマンドを DDM で使用するにあたっては、次の事項を考慮してください。

- DDM ファイルは、 ファイルのコピー (CPYF) コマンドおよびソース・ファイルのコピー (CPYSRCF) コマンドでは、 FROMFILE パラメーターおよび TOFILE パラメーターで指定することができます。

  注: 照会ファイルからのコピー (CPYFRMQRYF)、 ディスケットからのコピー (CPYFRMDKT)、およびテープからのコピー (CPYFRMTAP) コマンドでは、DDM ファイル名は、 TOFILE パラメーターでしか指定することができません。また、ディスケットへのコピー (CPYTODKT)、およびテープへのコピー (CPYTOTAP) コマンドでは、 DDM ファイル名を指定できるのは FROMFILE パラメーターだけです。
- 削除可能ファイルを削除不能ファイルにコピーする時には、 COMPRESS(*YES) を指定しなければなりません。そうしないと、エラー・メッセージが送られて、ジョブが終了します。
- DDM ファイル上のリモート・ファイル名にメンバー名を指定する場合には、 ファイルのコピー (CPYF) コマンドでそのファイルに指定されるメンバー名は、 DDM ファイル上のリモート・ファイル

名のメンバー名と同じでなければなりません。さらに、データベース・ファイルの一時変更 (OVRDBF) コマンドでは、 DDM ファイル上のリモート・ファイル名のメンバー名と異なるメンバー名を指定することはできません。

- DDM ファイルがメンバー名を指定せず、 データベース・ファイルの一時変更 (OVRDBF) コマンドでそのファイルのメンバー名を指定した場合には、 ファイルのコピー (CPYF) コマンドでは、OVRDBF コマンドで指定したメンバー名を使用します。
- TOFILE パラメーターが存在していないファイルを参照する DDM ファイルである場合には、 CPYF でそのファイルを作成します。次に挙げるのは、ファイルのコピー (CPYF) コマンドで作成されるファイルに関する特別の考慮事項です。
  - ターゲット DDM ジョブのためのユーザー・プロファイルが、ターゲット・システム上の物理ファイルの作成 (CRTPF) コマンドに対して許可されなければなりません。
  - iSeries ターゲットでは、TOFILE パラメーターは、iSeries Information Center の『ファイル管理』に説明されているものを除いて、 FROMFILE パラメーターの属性のすべてを備えています。
- TCP/IP を使用している場合には、 分散データ管理ファイルの作成 (CRTDDMF) コマンドの RMTLOCNAME パラメーターの 2 番目の項目は *IP でなければなりません。

サーバー間の「ファイルのコピー」コマンドの使用については、 iSeries Information Center の『分散データ管理』を参照してください。

ネットワーク・ファイル・コマンドを使用したネットワークでのデータの転送

データは、SNA 配布サービス (SNADS) をサポートするネットワーク・プロトコルを用いてネットワークで転送することができます。分散リレーショナル・データベース処理で使用される APPC および APPN プロトコルに加えて、SNADS は、 2 進データ同期通信同等リンク (BSCEL) および SNA アップライン機能 (SNUF) プロトコルを使用することができます。 SNADS がサポートする iSeries server は、 ネットワーク・ファイル送信 (SNDNETF) コマンドを用いて別のサーバーへデータを送信し、 ネットワーク・ファイル受信 (RCVNETF) コマンドおよびネットワーク・ファイルの処理 (WRKNETF) コマンドを用いて、別の iSeries server からネットワーク・ファイルを受信することができます。

サーバー保管および復元コマンドを使用したテーブルの移動

オブジェクト保管 (SAVOBJ) コマンドおよびオブジェクト復元 (RSTOBJ) コマンドを使用して、別の iSeries server からテーブルを移動することができます。 保管コマンドでは、テープ、ディスケット、または保管ファイルにデータベース・ファイルを保管します。保管ファイルは、通信によって別のサーバーに配布することができます。

テーブルまたはファイルの保管および復元に使用される保管および復元コマンドには、次のものがあります。

- ライブラリーの保管 (SAVLIB) コマンドは、 1 つ以上のコレクションまたはライブラリーを保管します。
- オブジェクトの保管 (SAVOBJ) コマンドでは、 1 つ以上のオブジェクト (データベースのテーブルおよびビューを含む) を保管します。
- 変更されたオブジェクトの保管 (SAVCHGOBJ) コマンドは、コレクションまたはライブラリーが最後に保管された時点以降、または指定された日付以後に変更されたオブジェクトを保管します。
- ライブラリーの復元 (RSTLIB) コマンドでは、コレクションまたはライブラリーを復元します。
- オブジェクトの復元 (RSTOBJ) コマンドでは、 1 つ以上のオブジェクト (データベースのテーブルおよびビューを含む) を復元します。

たとえば、2 つの販売店が合併した場合には、保管および復元コマンドを使用して、 1 つのリレーショナル・データベースのコレクションおよびテーブルを保管し、その後で残っているサーバーのリレーショナル・データベースに復元することができます。 そのためには、管理担当者は次のことを行います。

1. サーバー A に対してライブラリーの保管 (SAVLIB) コマンドを使用してコレクションを保管するか、またはサーバー A に対してオブジェクトの保管 (SAVOBJ) コマンドを使用してテーブルを保管する。
2. データを保管ファイル (SNADS を使用して配布できるファイル) に保管するのか、あるいはテープまたはディスケットに保管するのかを指定する。
3. 保管ファイルをサーバー B に配布するか、あるいはテープまたはディスケットをサーバー B に送付する。
4. サーバー B に対してライブラリーの復元 (RSTLIB) コマンドを使用してコレクションを復元するか、またはサーバー B に対してオブジェクトの復元 (RSTOBJ) コマンドを使用してテーブルを復元する。

保管および復元コマンドの使用に際して考慮すべき事項の 1 つは、復元されるオブジェクトに対する所有権および許可です。現在のオブジェクト所有者の有効なユーザー・プロファイルが、オブジェクトが復元されるサーバーに存在していなくてはなりません。 現在の所有者のプロファイルがこのサーバーに存在していない場合には、オブジェクトは QDFTOWN デフォルト・ユーザー・プロファイルのもとで復元されます。 ユーザーのオブジェクトに対する許可は、デフォルト・ユーザー・プロファイル・パラメーターによって制限されます。 QSECOFR 権限を持っているユーザーは、元の所有者のプロファイルをこのサーバー上に作成して、復元されたオブジェクトに変更を加えるか、あるいはこのオブジェクトに対する新しい許可を、ローカル・ユーザーとリモート・ユーザーの両方に指定するか、どちらかを行わなければなりません。

保管および復元コマンドの詳細については、 iSeries Information Center の『バックアップおよび回復の手引き』を参照してください。

非 iSeries server から iSeries server へのデータベースの移動

別の IBM サーバーから iSeries server へ、または IBM 以外のサーバーから iSeries server へ、ファイルを移動する必要のある場合があります。 このセクションには、非 iSeries server から iSeries server へデータを移動する方法をリストしてあります。 ただし、それぞれの使用に関する特定の指示事項については、他のサーバーとともに提供されているか、またはアプリケーションとして指定されている資料を参照する必要があります。

別の IBM サーバーからのデータの移動

別の IBM サーバーから iSeries server へデータを移動する場合に使用できる方法はたくさんあります。その中には次のような方法があります。

- 高水準言語プログラムを作成して、別のサーバーからデータを取り出すことができます。 これに対応する サーバーのプログラムを使用して、データをサーバーにロードすることができます。
- 他の DRDA 実装がサポートされているサーバーの場合には、データを移動するために SQL 機能を使用することができます。 分散作業単位を使用すれば、ソース・データに対して照会をオープンし、その同じ作業単位で、サーバー上のテーブルにデータを挿入することができます。 パフォーマンスを最高にするためには、この照会でブロック化を使用し、サーバーでは複数行挿入を行ってください。 さらに詳しくは、 18 ページの『ヒント: 分散リレーショナル・データベース・アプリケーションの設計』を参照してください。
- データは、他のサーバー上のテーブルおよびファイルから抽出して、テープまたはディスケットにコピーして iSeries server に送付するか、あるいは通信回線を用いて送信することができます。
  - DB2 UDB for z/OS データベースからの場合は、データベース・マネージャーに付属している、DSNTIAUL と呼ばれるサンプル・プログラムを使用して、ファイルまたはテーブルからデータを抽出することができます。

  - DB2 UDB Server for VM (SQL/DS) データベースからの場合は、データベース・マネージャーのデータベース・サービス・ユーティリティー部分を使用して、データを抽出することができます。
  - DB2 UDB for z/OS または DB2 UDB Server for VM データベースの両方から、データを抽出するためにデータ抽出 (DXT*) を使用できます。ただし、DXT™ によるヌル・データの処理は、以下に説明するようにファイル・コピーによるヌル・データの処理と互換性がありません。したがって、 iSeries server への移行のためにリレーショナル・データベースをアンロードする目的で DXT を使用することはお勧めできません。
  - IMS/DB 階層データベースから、DXT を使用してデータを抽出することができます。
- DB2 UDB for z/OS または DB2 UDB Server for VM のデータベースからコピーする場合は、標準テープ管理技法を使用してデータをテープに書き出します。 iSeries server では、テープからコピー (CPYFRMTAP) コマンドを使用して、テープからデータをロードします。ただし、ファイル・コピー (CPYF) コマンドを使用すると、 IBM メインフレーム・コンピューターからデータを移行するための特別なサポートを受けることができます。 テープ・ファイル一時変更 (OVRTAPF) コマンドを使用して、テープ・データに CPYF を行うこともできます。 OVRTAPF コマンドでは、特別なテープ固有のパラメーターを指定することができます。このパラメーターは iSeries server 以外のサーバーからデータをインポートする際に必要な場合があります。

  特別な CPYF サポートにより、ヌル値可能データおよび浮動小数点データをインポートすることができます。メインフレームからヌル値可能データをアンロードして、ヌル値可能フィールドに単一バイト・フラグを関連付けることができます。 FMTOPT パラメーターで *NULLFLAGS オプションが指定されていると、 ファイルのコピー (CPYF) コマンドはヌル・フラグを識別して、テープの隣接するフィールドのデータを無視します。そして DB2 UDB for iSeries のフィールドをヌルにします。 IBM メインフレームからデータをインポートする際に役に立つ別の FMTOPT パラメーター値は、 *CVTFLOAT 値です。これは、System/390 形式でテープ上に保管されている浮動小数点データを、 DB2 UDB for iSeries で使用されている IEEE 形式に変換することを可能にします。

  iSeries server システムでのテープとディスケット装置の使用に関する詳細については、『ストレージ・ソリューション』のトピックを参照してください。ファイル・コピー・コマンドの使用によるサーバー相互間のコピーについては、『分散データ管理』のトピック、および iSeries Information Center の『制御言語 (CL)』のトピックを参照してください。
- 通信回線を用いてデータを送信する場合は、 iSeries server 上の SNADS サポートによって処理することができます。 SNADS サポートでは、分散リレーショナル・データベース処理に使用される APPC または APPN プロトコルの場合に加えて、 BSCEL または SNUF プロトコル用のネットワーク・ファイルを転送します。
  - MVS™ システムからの場合は、データは TSO XMIT 機能を使用して、 iSeries server に送信することができます。 サーバーは、ネットワーク・ファイルの処理 (WRKNETF) またはネットワーク・ファイルの受信 (RCVNETF) コマンドを使用して、ネットワーク・ファイルを受信します。
  - VM システムからの場合は、データは SENDFILE 機能を使用して、サーバーに送信することができます。 サーバーは、ネットワーク・ファイルの処理 (WRKNETF) またはネットワーク・ファイルの受信 (RCVNETF) コマンドを使用して、ネットワーク・ファイルを受信します。
- Microsoft® Windows からは、クライアント・データは iSeries Access (別途注文の IBM 製品) を使用して iSeries サーバーに送信することができます。
- さまざまなワークステーション・クライアントからの場合は、 DB2 Connect の IMPORT と EXPORT ユーティリティーを使用して、 iSeries server との間でデータを相互にコピーすることができます。なお IMPORT では、既存のテーブルにのみデータをインポートすることができます。 IMPORT ユーティリティーと EXPORT ユーティリティーの例については、 「Advanced Functions and Administration on

DB2 Universal Database for iSeries Redbook」を参照してください。このレッドブックには、 IMPORT ユーティリティーと ExPORT ユーティリティーで使用できるファイル・タイプとデータ・フォーマットに関する情報も含まれています。

- データはまた、非同期通信など、 SNADS をサポートしない通信回線によって送信することもできます。 OS/400 ライセンス・プログラムの一部となっているユーティリティーのファイル転送サポート (FTS) を使用して、データの送受信を行うことができます。通信および通信ファイルの処理に関する詳細については、 「*ICF Programming*」  を参照してください。

非 IBM サーバーからのデータの移動

IBM 以外のサーバーからファイルまたはテーブルをテープまたはディスケットにコピーし、それらのファイルを iSeries server システムにロードすることができます。 コピー元インポート・ファイル (CPYFRMIMPF) コマンドを使って、これを行ってください。

販売会社独自の通信機能も、別の 2 つの iSeries ライセンス・プログラムを介してサポートされます。

ローカル・エリア・ネットワークと広域ネットワークの両方のための対等接続機能が、伝送制御プロトコル/インターネット・プロトコル (TCP/IP) によって提供されています。 iSeries TCP/IP 接続ユーティリティー/400 ライセンス・プログラムのファイル転送プロトコル (FTP) を使用すれば、リモート・サーバーの機能に応じて、多くのタイプのファイルを受信することができます。 詳細については、 iSeries Information Center の『TCP/IP セットアップ』を参照してください。

OSI ファイル・サービス/400 ライセンス・プログラム (OSIFS/400) では、オープン・サーバー間相互接続 (OSI) ネットワークのためのファイル管理および転送サービスを提供しています。 OSIFS/400 は、前提ライセンス・プログラムである、 OSI コミュニケーション・サブシステム/400 との併用によって、 OSI ファイル転送アクセスおよび管理 (FTAM) 標準に適合するリモート IBM サーバーまたは IBM 以外のシステムに、iSeries server を接続します。

OSIFS/400 プログラムは、リモート・サーバーからローカル iSeries server にファイルをコピーまたは移動するために、対話式インターフェースか、アプリケーション・プログラミング・インターフェース (API) のどちらかを提供します。詳細については、*OSI Communications Subsystem Programming and Concepts Guide* を参照してください。

第 6 章 分散リレーショナル・データベースの管理および操作の作業

分散リレーショナル・データベースの管理担当者は、複数のサーバー上で行われる作業に責任を負うことになります。 ローカル・システムを起点とする**アプリケーション・リクエスター (AR)** としての作業の監視は、他の作業の監視を iSeries server 上で行うのと同じ要領で行うことができます。 ローカル・システム上で行われる**アプリケーション・サーバー (AS)** としての作業単位を追跡する際には、同じツールを使用しますが、異なる種類の情報を探すことになります。

この章では、ネットワークの中で行われる分散リレーショナル・データベース作業を管理できる方法について説明しています。ここで説明するコマンド、プロセス、およびその他のリソースは、分散リレーショナル・データベースを使用するためにだけ存在しているわけではなく、すべての iSeries server の操作のために提供されているツールです。 ここで説明している管理コマンド、プロセス、およびリソースは、いずれもすべての DB2 UDB for iSeries 管理機能とともに、OS/400 プログラムに含まれています。

iSeries server の実行管理機能を使えば、次のことを実行できるので、複数のサーバー上の作業を追跡する効果的な手段が得られます。

- リレーショナル・データベース活動のモニター
- リモート iSeries server システムの操作
- DDM 会話の制御
- プログラムで使用されるオブジェクトの表示
- 分散リレーショナル・データベースからのコレクションの除去
- 分散リレーショナル・データベースのジョブ会計
- TCP/IP サーバーの管理
- リレーショナル・データベース・ディレクトリーの監査

リレーショナル・データベース活動のモニター

次に挙げる制御言語 (CL) コマンド (すべてのコマンドがさまざまな方法で同じ情報を提供する) を使用すれば、 iSeries server での作業の概要を把握することができます。このコマンドに関する詳細は、次のトピックを参照してください。

- 分散リレーショナル・データベースのジョブの処理

  ジョブの処理 (WRKJOB) コマンドは、ジョブ名または WRKJOB コマンドを入力するジョブが分かっている場合に使用し、ジョブに固有の情報が得られます。
- 分散リレーショナル・データベースのユーザー・ジョブの処理

  ユーザー・ジョブの処理 (WRKUSRJOB) コマンドは、ジョブが実行されるユーザー・プロファイルが分かっている場合に使用すれば、より詳細な情報が提供されます。 (TCP/IP 環境では WRKUSRJOB QUSER *ACTIVE を使用してください。)
- 分散リレーショナル・データベースの活動ジョブの処理

WRKACTJOB (活動ジョブの処理) コマンドでは、サーバー上で行われている作業について、最も一般的な把握ができます。現在サーバーで実行中のすべてのジョブが表示され、各ジョブごとに何らかの統計が表示されます。

- 分散リレーショナル・データベースのコミットメント定義の処理。

  WRKCMTDFN (コミットメント定義の処理) コマンドは、コミットメント定義を表示するもので、これは、 コミットメント制御開始 (STRCMTCTL) コマンドでコミットメント制御を開始した場合に、そのコミットメント制御の情報を保管するために使用されます。

これらのコマンドを使用して、サーバーでの作業の概要を把握することに加えて、情報を追跡したり、特定のジョブを見つけたりすることもできます。詳細については、次のトピックを参照してください。

- 分散リレーショナル・データベースのジョブ・ログによる要求情報の追跡
- 分散リレーショナル・データベース・ジョブの探索

分散リレーショナル・データベースのジョブの処理

ジョブの処理 (WRKJOB) コマンドは、「ジョブ処理」メニューを表示します。 このメニューを使用すれば、オプションを選択して、指定されたジョブに関連する情報を処理または変更することができます。現在使用しているジョブについての情報を入手する場合には、パラメーターを指定しないでコマンドを入力します。あるジョブに関連する同じ情報を得たい場合には、次のようにコマンドにジョブの名前を入力することによって、ジョブを指定します。

```
WRKJOB JOB(ジョブ番号/ユーザー ID/ ジョブ名)
```

ジョブがジョブ待ち行列に入っているのか、出力待ち行列に入っているのか、または活動状態にあるのかについて、メニュー画面のオプションによって提供される情報を入手することができます。ただし、ジョブがサーバー内にあると見なされるのは、その入力データがすべて完全に読み込まれてからになります。 その時になって初めてジョブ待ち行列に項目が入ります。ジョブ情報のオプションには、次のものがあります。

- ジョブ状況属性
- ジョブ定義属性
- スプール・ファイル情報

以下に挙げるオプションに関する情報が表示されるのは、ジョブが活動状態にある時だけです。

- ジョブ実行属性
- ジョブ・ログ情報
- 呼び出しスタック情報
- ジョブ・ロック情報
- ライブラリー・リスト情報
- オープン・ファイル情報
- ファイル指定変更情報
- コミットメント制御状況
- 通信状況
- 活動化グループ
- 相互除外 (Mutex)

オプション 10 (ジョブ・ログの表示) では、活動ジョブまたはジョブ待ち行列に入っているジョブについての情報が得られます。終了したジョブについては、通常、オプション 4 (スプール・ファイルの処理) を使用することによって、同じ情報を見つけることができます。このオプションを選択すると、「ジョブ・ス

プール・ファイルの処理」画面が表示されますから、そこでオプション 5 を使用して、 QPJOBLOG という名前のファイル (リストに載っている場合) を表示することができます。

分散リレーショナル・データベースのユーザー・ジョブの処理

ジョブで使用されているユーザー・プロファイル (ユーザー名) が分かっている場合には、 ユーザー・ジョブの処理 (WRKUSRJOB) コマンドを使用して、ジョブ情報を表示または変更することができます。自分のユーザー・プロファイルがあるサーバーのジョブのリストを入手する場合には、パラメーターを指定しないでコマンドを入力してください。 次のようにコマンドに名前を入力することによって、ユーザーおよびジョブ状況を指定して、ジョブのリストを短縮することができます。

```
WRKUSRJOB USER(KCDBA)
```

「ユーザー・ジョブの処理」画面が表示されて、サーバーで実行されている (*ACTIVE) か、ジョブ待ち行列に入っている (*JOBQ) か、あるいは出力待ち行列に入っている (*OUTQ) ユーザー・ジョブの名前および状況情報が表示されます。 下の画面には、KCDBA という名前のユーザーの活動ジョブおよび終了ジョブが表示されています。

```
ユーザー・ジョブの処理                    KC105
03/29/92  16:15:33
オプションを入力して、実行キーを押してください。
2= 変更    3= 保留      4= 終了    5= 処理    6= 解放    7= メッセージの表示
8= スプール・ファイルの処理    13= 切断 ...

OPT   ジョブ       ユーザー     タイプ    ----- 状況 -----    機能
__    KC000        KCDBA        CMNEVK    OUTQ
__    KC000        KCDBA        CMNEVK    OUTQ
__    KC000        KCDBA        CMNEVK    OUTQ
__    KC000        KCDBA        CMNEVK    OUTQ
__    KC000        KCDBA        CMNEVK    ACTIVE
__    KC0001       KCDBA        CMNEVK    ACTIVE              *  -PASSTHRU
__    KC0001       KCDBA        INTER     ACTIVE              CMD-WRKUSRJOB




終わり
パラメーターまたはコマンド
===>
F3= 終了    F4= プロンプト      F5= 最新表示    F9= コマンドの複写
F11= スケジュール・データの表示    F12= 取り消し     F17= 最上部     F24= キーの続き
```

この画面には、サーバー内にあるこのユーザーのジョブがすべてリストされ、指定された状況 (この場合は *ALL) が示され、ジョブのタイプが表示されています。 また、選択したジョブについてコマンドを入力するための 8 つのオプション (2 から 8 および 13) も提供されています。オプション 5 を選択すると、上で説明した「ジョブの処理」画面が表示されます。

ユーザー・ジョブの処理 (WRKUSRJOB) コマンドは、サーバーが TCP/IP を使用している場合に DDM TCP/IP サーバー・ジョブの状況を見たい場合に役立ちます。次のコマンドを実行してください。

```
WRKUSRJOB QUSER *ACTIVE
```

文字 QRWT で始まるジョブが表示されるまでページ送りしてください。 サーバーが活動状態にある場合、(事前開始 DRDA ジョブがサーバー上で実行されていないかぎり) QRWTLSTN という名前のジョブが 1 つと、 QRWTSRVR という名前の 1 つ以上のジョブが表示されるはずです。 QRWTSRVR ジョブは、事前開始ジョブです。 QRWTLSTN ジョブが表示されていない場合には、次のコマンドを実行して、ジョブを開始してください。

```
STRTCPSVR *DDM
```

QRWTLSTN ジョブが表示されていて、QRWTSRVR ジョブが表示されておらず、 DRDA 事前開始ジョブが使用不可になっていない場合には、次のコマンドを実行し、事前開始ジョブを開始してください。

```
STRPJ subsystem QRWTSRVR
```

V5R2 より前のバージョンでは、通常 QRWTSRVR は QSYSWRK サブシステムで実行されていました。V5R1 より後のバージョンでは、QRWTSRVR は QUSRWRK で実行されます。

分散リレーショナル・データベースの活動ジョブの処理

数人のユーザーに対して実行されているジョブをモニターする場合や、ジョブを探しているものの、ジョブの名前やユーザー ID がわからない場合は、 WRKACTJOB (活動ジョブの処理) コマンドを使用してください。このコマンドを入力すると、「活動ジョブの処理」画面が表示されます。この画面には、現在サーバー上で活動状態にあるジョブのパフォーマンスおよび状況の情報が表示されます。 情報は、すべてある 1 つのジョブを基準に収集され、サブシステム別にグループ化されています。

次の画面には、KC105 システムでの典型的な 1 日の「活動ジョブの処理」画面が示されています。

```
活動ジョブの処理                          KC105
99/11/05  11:30:00
CPU %:   41.7       経過時間 :   04:37:55     活動ジョブ数 :   42

オプションを入力して、実行キーを押してください。
2= 変更   3= 保留     4= 終了   5= 処理   6= 解放   7= メッセージの表示
8= スプール・ファイルの処理   13= 切断 ...

OPT  サブシステム/  ユーザー       タイプ  CPU %       機能            状況
       ジョブ
__   QBATCH          QSYS           SBS     .0                      DEQW
__   QCMN            QSYS           SBS     .0                      DEQW
__   QINTER          QSYS           SBS     .0                      DEQW
__     DSP01         CLERK1         INT     .0        CMD-STRSQL    DSPW
__     DSP02         CLERK2         INT     .0        *  -CMDENT    DSPW




続く ...
パラメーターまたはコマンド
===>
F3= 終了   F5= 最新表示   F7= 検索       F10= 統計の再始動
F11= スレッド・データの表示  F12= 取り消し   F23= オプション続き   F24= キーの続き
```

F11 (経過データの表示) を押すと、次の画面が表示されて、詳細な状況情報が得られます。

```
活動ジョブの処理                          KC105
99/11/05  11:30:00
CPU %:   41.7       経過時間 :   04:37:55     活動ジョブ数 :   42

オプションを入力して、実行キーを押してください。
2= 変更    3= 保留      4= 終了    5= 処理    6= 解放    7= メッセージの表示
8= スプール・ファイルの処理    13= 切断 ...
--------- 経過 ---------
OPT  サブシステム/ タイプ    プール     PTY      CPU  INT    RSP  AUXIO  CPU %
       ジョブ
__    QBATCH          SBS      2         0      4.4                108     .0
__    QCMN            SBS      2         0     20.7                668     .0
__      KC000         EVK      2        50       .1                  9     .0
__      KC0001        EVK      2        50       .1                  9     .0
__      MP000         EVK      2        50       .1                 14     .0
__    QINTER          SBS      2         0      7.3                  4     .0
__      DSP01         INT      2        20       .1                  0     .0
__      DSP02         INT      2        20       .1                  0     .0

続く ...
パラメーターまたはコマンド
===>
F3= 終了    F5= 最新表示    F7= 検索       F10= 統計の再始動
F11= スレッド・データの表示   F12= 取り消し    F23= オプション続き    F24= キーの続き
```

「活動ジョブの処理」画面では、ジョブ優先順位およびサーバー使用状況、「ユーザー・ジョブの処理」画面から得られるユーザーおよびタイプの情報が得られます。 また、ジョブに対して 11 のオプション (2 から 11 および 13) が使用でき、この中にはオプション 5 が含まれていて、これを選択した場合には、選択したジョブに関する「ジョブの処理」画面が表示されます。

iSeries ナビゲーターを使用しても、ジョブ優先順位とサーバー使用状況に関する情報を表示できます。これを行うためには、以下のステップに従ってください。

1. iSeries ナビゲーター・インターフェース中のデータベースを選択します。
2. 情報を表示したいリモート・データベースを選択します。
3. プロパティーを右クリックして選択します。プロパティー・ウィンドウが表示され、このウィンドウに情報が表示されます。

分散リレーショナル・データベースのコミットメント定義の処理

サーバー上でコミットメント定義を処理する場合は、 WRKCMTDFN (コミットメント定義の処理) コマンドを使用してください。コミットメント定義 を使用すると、 コミットメント制御開始 (STRCMTCTL) コマンドでコミットメント制御が開始された場合に、そのコミットメント制御の情報を記憶できます。 これらのコミットメント定義は、活動ジョブに関連している場合もしていない場合もあります。活動ジョブに関連していないコミットメント定義は終了していますが、その論理作業単位の 1 つまたはいくつかはまだ完了していません。

WRKCMTDFN (コミットメント定義の処理) コマンドを使用すると、コミットメント定義のジョブ名、状況または論理作業単位識別コードに基づいてコミットメント定義を処理することができます。

STATUS パラメーターには、すべてのジョブを指定するか、あるいは、*RESYNC または *UNDECIDED の状況値をもつジョブだけを指定することができます。 *RESYNC は、同期点を再設定する過程においてそのリソースの再同期化に関係するジョブだけを示します。*同期点* は、すべてのリソースが整合状態にある地点です。

*UNDECIDED は、リソースのコミットまたはロールバックについての決定が未知であるジョブだけを示します。

LUWID パラメーターでは、他のサーバーのコミットメント定義を処理しているコミットメント定義を表示することができます。 このようなコミットメント定義を含んでいるジョブは、APPC 保護会話を使用して通信します。あるサーバーのコミットメント定義を表示し、それをWRKCMTDFN (コミットメント定義の処理) コマンドへの入力として使用すると、 LUWID を見つけて、対応するコミットメント定義を発見することができます。

コミットメント定義の処理 (WRKCMTDFN) コマンドを使用して未決定のジョブ内のローカル・リソースを解放することができます。ただし、これができるのは、コミットメント定義が「Prepared (準備済み) (PRP)」状態または「最終エージェント保留中 (LAP)」状態である場合のみです。コミットメント定義は、強制的にコミットまたはロールバックすることができ、したがって、保持リソースを解放できます。開始プログラムがコミットメント定義でとられた処置を知るまでは、制御はもとのコミットを出したプログラムには戻りません。

また、再同期化が他のサーバーで決して終わらないことが分かった場合に、再同期化を終わりにするためにWRKCMTDFN (コミットメント定義の処理) コマンドを使用することができます。

コミットメント制御と再同期化について詳しくは、 iSeries Information Center の『トランザクションとコミットメント制御のトラブルシューティング (Troubleshoot Transactions and Commitment Control)』のトピックを参照してください。

分散リレーショナル・データベースのジョブ・ログによる要求情報の追跡

iSeries server 上のすべてのジョブは、それぞれジョブについて入力された要求に関連する情報が入るジョブ・ログを持っています。ジョブ・ログの中の情報には、次のものがあります。

- ジョブで使用されたコマンド。
- 送られたが、プログラム・メッセージ待ち行列から除去されなかったメッセージ。
- CL プログラムの中のコマンド (そのプログラムが LOGCLPGM(*JOB) を用いて作成され、ジョブが LOGCLPGM(*YES) を指定しているか、または CL プログラムが LOGCLPGM(*YES) を用いて作成された場合)。

ジョブの終了時に、ジョブ・ログは、 QPJOBLOG という名前のスプール・ファイルを書き込むことができ、元のジョブ・ログは削除されます。ジョブ記述の LOG パラメーターを指定することによって、ジョブ・ログに書き込まれる情報を制御することができます。

ジョブ・ログを表示する方法は、ジョブの状況によって異なります。ジョブが終了していて、しかもジョブ・ログがまだ印刷されていない場合には、 ユーザー・ジョブの処理 (WRKUSRJOB) コマンドを使用してジョブを見つけてから、オプション 8 (スプール・ファイルの表示) を選択してください。 QPJOBLOG という名前のスプール・ファイルを見つけ、オプション 5 (ジョブ・ログの表示) を選択してください。ジョブの処理 (WRKJOB) コマンド、および「ジョブの処理」画面上の他のオプションを使用することによって、ジョブ・ログを表示することもできます。

バッチ・ジョブまたは対話式ジョブがまだ活動状態にあるか、あるいはジョブ待ち行列に入っていて、まだ開始されていない場合には、 WRKUSRJOB コマンドを使用してジョブを見つけてください。WRKACTJOB (活動ジョブの処理) コマンドを使用すると、活動ジョブのジョブ・ログは表示されますが、ジョブ待ち行列に入っているジョブは表示されません。オプション 5 (ジョブの処理) を選択してから、オプション 10 (ジョブ・ログの表示) を選択してください。

自分自身の対話式ジョブのジョブ・ログを表示する場合には、次のようにしてください。

- ジョブ・ログの表示 (DSPJOBLOG) コマンドを入力してください。

- ジョブの処理 (WRKJOB) コマンドを入力し、「ジョブの処理」画面でオプション 10 (ジョブ・ログの表示) を選択してください。
- 「コマンド入力」画面で F10 (詳細メッセージの表示) を押して、ジョブ・ログに示されているメッセージを表示してください。

ジョブ・ログの表示 (DSPJOBLOG) コマンドを使用すると、「ジョブ・ログ」画面が表示されます。この画面には、プログラム名に次のような特殊記号がついて表示されます。

>> 実行中のコマンドまたは次に実行されるコマンド。たとえば、CL プログラムまたは高水準言語プログラムが呼び出された場合には、プログラムに対する呼び出しが表示されます。

> コマンドは処理を完了しました。

. . コマンドはまだ処理されていません。

? 応答メッセージ。この記号によるマークが付くのは、応答を必要としているメッセージと応答がなされたメッセージの両方です。

分散リレーショナル・データベース・ジョブの探索

アプリケーション・リクエスター (AR) 上で分散リレーショナル・データベース・ジョブについての情報を探索する時に、使用されているユーザー・プロファイルが分かっている場合には、 ユーザー・ジョブの処理 (WRKUSRJOB) コマンドを使用することによって、そのジョブを見つけることができます。このコマンドは、**アプリケーション・サーバー (AS)** 上でも使用することができますが、 AR で使用されているものとは異なっている場合があることを心得ていてください。 TCP/IP サーバーの場合、ジョブ名を修飾するユーザー・プロファイルは常に QUSER になり、ジョブ名は常に QRWTSRVR になります。ログの表示 (DSPLOG) コマンドは、完全なサーバー・ジョブ名を見つけやすくするために使用できます。メッセージは、次の形式になります。

```
DDM job 031233/QUSER/QRWTSRVR servicing user XY on 10/02/97 at 22:06
```

指定したユーザー・プロファイルに関して複数のジョブがリストされていて、そのリレーショナル・データベースが DRDA を使用してアクセスされている場合は、オプション 5 (ジョブの処理) を入力して、「ジョブの処理」画面を表示してください。この画面でオプション 10 (ジョブ・ログ) を入力して、ジョブ・ログを表示してください。このジョブ・ログでは、ジョブが分散リレーショナル・データベース・ジョブであるかどうかが示され、そうである場合には、そのジョブが接続されるリモート・サーバーが示されます。(接続が APPC または TCP/IP を使用しているかどうかによって) ジョブ・ログのページを送って次のメッセージを探してください。

CPI9150
リモート・データベース・ジョブが開始された。

CPI9160
データベース接続が TCP/IP またはローカル・ソケット経由で開始された。

メッセージ CPI9150 および CPI9160 の第 2 レベル・テキストには、 AS ジョブのジョブ名が入っています。

AS で作業していて、ジョブ名は分かっていないが[2]、ユーザー名が分かっている場合には、 ユーザー・ジョブの処理 (WRKUSRJOB) コマンドを使用してください。ユーザーを指定しないでこのコマンドを使用し

2. DDM TCP/IP サーバーを使用している場合、上記で説明されているように、 ログの表示 (DSPLOG) コマンドを使用してジョブ名を見つけることができます。

3. TCP/IP の場合、ジョブ名内のユーザー・プロファイルは常に QUSER になります。

た場合には[3]、自分が使用しているユーザー・プロファイルのもとにあるジョブのリストが戻されます。「ユーザー・ジョブの処理」画面では、次の列を使用して、 APPC 接続を行っている AS ジョブの識別に役立ててください。

1 ジョブのタイプ列には、APPC 通信ジョブについて、タイプを CMNEVK としてリストしたジョブが表示されます。

2 状況の列には、ジョブが活動状態であるか、あるいは完了しているかが表示されます。ジョブのログを記録するためのサーバーのセットアップ方法によっては、活動ジョブしか表示されない場合があります。

3 ジョブの列には、ジョブ名が示されます。 AS 上のジョブ名は使用されている装置と同じになります。

```
ユーザー・ジョブの処理                    KC105
03/29/92  16:15:33
オプションを入力して、実行キーを押してください。
2= 変更   3= 保留     4= 終了   5= 処理   6= 解放   7= メッセージの表示
8= スプール・ファイルの処理   13= 切断 ...

OPT   ジョブ       ユーザー     タイプ    ----- 状況 -----    機能
__    KC000        KCDBA        CMNEVK    OUTQ
__    MP000        KCDBA        CMNEVK    OUTQ
__    MP000        KCDBA        CMNEVK    OUTQ
__    KC000        KCDBA        CMNEVK    OUTQ
__    KC000        KCDBA        CMNEVK    ACTIVE
__    KC0001       KCDBA        INTER     ACTIVE             CMD-WRKUSRJOB
      3                           1         2
```

活動 AS ジョブを探索する時に、ユーザー名が分かっていない場合には、 WRKACTJOB (活動ジョブの処理) コマンドを入力すると、サーバー上で活動状態にあるサブシステムのジョブのリストが表示されます。下の例には、探索する項目をいくつか示してあります。

```
活動ジョブの処理                          KC105
99/11/05  11:30:00
CPU %:    41.7     経過時間 :   04:37     活動ジョブ数 :   102

オプションを入力して、実行キーを押してください。
2= 変更   3= 保留     4= 終了   5= 処理   6= 解放   7= メッセージの表示
8= スプール・ファイルの処理   13= 切断 ...

OPT  サブシステム/  ユーザー       タイプ    CPU % 機能              状況
       ジョブ
     QBATCH          QSYS           SBS        .0                  DEQW
4    QCMN            QSYS           SBS        .0                  WDEQ
__     KC0001        KCCLERK        EVK        .0  *               EVTW
       5                            6
```

4 AS ジョブを処理するためにセットアップされているサブシステム[4]を探してください。この例では、AS ジョブのサブシステムは QCMN です。

5 APPC AS ジョブでは、ジョブ名は AS ジョブ用に作成された装置の装置名です。

6 リストされているジョブ・タイプ[5]は、通常、プログラム開始要求によって開始された EVK です。

---

4. TCP/IP サーバー・ジョブ用のサブシステムは、 V5R2 より前のバージョンでは QSYSWRK、 V5R1 より後のバージョンでは QUSRWRK です。

5. TCP/IP AS ジョブの場合、 (ジョブ・タイプが BCI の場合に DRDA 事前開始ジョブがサーバー上で活動状態でないかぎり) ジョブ・タイプは PJ です。

探索しているジョブと思われるものが見つかったら、オプション 5 を入力して、そのジョブを処理してください。次に、「ジョブ・メニューの処理」画面でオプション 10 を選択して、ジョブ・ログを表示してください。 DB2 Universal Database for iSeries アプリケーション・リクエスターから AS にアクセスしているジョブの分散データベース・ジョブ・ログには、最上部の近くに次のようなステートメントが入っています。

CPI3E01
ローカル・リレーショナル・データベースが *(*システム名*)* によってアクセスされた。

AS で実行中のジョブが見つかったら、AR が iSeries server である場合には、そのジョブをトレースして AR までさかのぼることもできます。 ジョブ・ログ内に以下のいずれかのメッセージが現れます。受け取ったメッセージ上にカーソルが置かれます。

CPI9152
ターゲット DDM ジョブが**アプリケーション・リクエスター (AR)** によって開始されました。

CPI9162
DDM 接続を処理するために割り当てられたターゲット・ジョブは、 TCP/IP 経由でアプリケーション・リクエスターによって開始されました。

Help キーを押すと、このステートメントの詳細なメッセージが表示されます。アプリケーション・リクエスター (AR) のジョブ名は、このジョブの原因となった AR 上のジョブです。

リモート iSeries server システムの操作

分散リレーショナル・データベースでは、管理担当者は、時にリモート iSeries server を操作しなければならない場合があります。

たとえば、リモート・サーバーを開始または停止しなければならない場合があります。 iSeries server には、リモート・サーバーを必要に応じて操作できるようにするオプションが用意されています。確かに、リモート・サーバーが確実に動作するようにする最も単純な方法は、分散リレーショナル・データベース要件に応じて、リモート・ロケーションでサーバーの電源を入れられるようにすることです。 しかし、このことは必ずしも常に可能であるとは限りません。自動電源オンおよび電源オフのスケジュールを用意するか、またはリモート・サーバーに対するリモート電源オンを可能にすることができます。 電源オンおよび電源オフのスケジュールについては、『自動的な電源オン/オフ・スケジュールの設定』をご覧ください。 IPL とリモート IPL を制御するシステム値については、『IPL を制御するサーバー値』をご覧ください。

サーバーでは、リアルタイムに、または前もって予定した時刻に、このことを行う複数の方法が用意されています。 さらに頻度が高いのは、リモート・サーバーの動作中に、そのリモート・サーバーで特定のタスクを実行する必要が生じる場合です。 そのためには 3 つの主要な方法が使用できます。表示装置パススルーを使用する方法、SBMRMTCMD (リモート・コマンド投入) コマンドを用いる方法、およびストアード・プロシージャーを使用する方法です。

SBMRMTCMD (リモート・コマンド投入) コマンドは、**アプリケーション・サーバー (AS)** 側で実行するために、分散データ管理機能 (DDM) サポートを使用する CL コマンドを投入するためのものです。まず DDM ファイルを作成する必要があります。 DDM ファイルのリモート・ロケーション情報を使用して、使用する通信回線を決めます。こうして、投入されたコマンドを受信するアプリケーション・サーバー (AS) を識別します。 DDM ファイルを使用して、アプリケーション・サーバー (AS) 側で実行するためのコマンドを投入する際には、 DDM ファイルに関連するリモート・ファイルは関与しません。 DDM ファイルの作成に関する説明については、83 ページの『DDM ファイルのセットアップ』を参照してください。

SBMRMTCMD (リモート・コマンド投入) コマンドは、バッチ環境でも、QCAEXEC システム・プログラムを介しても実行できる CL コマンドであれば、すべて投入することができます。つまり、コマンドは、ALLOW 属性に *BPGM および *EXEC の値が指定されています。 表示コマンド (DSPCMD) コマンドを使用することによって、 ALLOW 属性を表示することができます。

SBMRMTCMD (リモート・コマンド投入) コマンド の主要な目的は、**アプリケーション・リクエスター (AR)** 側のユーザーやプログラムが、アプリケーション・サーバー側にあるオブジェクトに対して、ファイル管理操作およびファイル許可活動を実行できるようにすることにあります。このコマンドの 2 次的な目的は、ユーザーが非ファイル操作 (メッセージ待ち行列の作成など) を実行できるようにすること、あるいはアプリケーション・サーバー (AS) 側で実行するためのユーザー作成コマンドを投入できるようにすることです。 CMD パラメーターを使用すれば、アプリケーション・サーバー (AS) 側で実行されるコマンドを表す文字ストリング (最大 2000 文字) を指定することができます。

投入される CL コマンド、およびそのコマンドの実行対象となるオブジェクトに関して、アプリケーション・サーバー (AS) 上で適正な権限が付与されていなければなりません。アプリケーション・リクエスター (AR) ユーザーがそのための正しい権限 (アプリケーション・サーバー (AS) のユーザー・プロファイルに決められている) を持っている場合には、以下に挙げる活動が、 SBMRMTCMD (リモート・コマンド投入) コマンドを使用してリモート・ファイルに対して実行できることの例になります。

- リモート・テーブルに対するオブジェクト権限の認可または取り消し。
- テーブルまたはその他のオブジェクトの検査。
- テーブルまたはその他のオブジェクトの保管または復元。

このコマンドを使用すれば、リモート・サーバー上のテーブルまたはその他のオブジェクトに対して多くのことができますが、このコマンドの使用が、タスクによっては、 iSeries server 上の他の方法に比べて効果的でない場合があります。 たとえば、このコマンドを使用して、リモート・ファイルのファイル記述またはフィールド属性の表示、あるいはファイルまたはその他のオブジェクトのダンプを行うことはできますが、出力はアプリケーション・サーバー (AS) に留まっています。リモート・ファイル記述およびフィールド属性をアプリケーション・リクエスター (AR) に表示するには、 SYSTEM(*RMT) を指定したファイル記述表示 (DSPFD) コマンドまたはファイル・フィールド記述表示 (DSPFFD) コマンドを使用し、リモート・ファイルに関連する DDM ファイルの名前を指定する方が、方法として優れています。

投入できる CL コマンドのリストおよびこのコマンドの使用上の制約については、『分散データ管理』を参照してください。なお、DDM が会話を共用する方法については、『DDM 会話の制御』に説明してあります。

DDM 会話の制御

注: *会話* という語には、 SNA APPC 用語では、特定の、技術的な意味があります。この語は、正式な意味では TCP/IP 用語に拡大しません。ただし、TCP/IP にはよく似た概念があります (この点について扱っている他の資料にある「ネットワーク接続」)。本書では、この語は、同様に、TCP/IP ネットワーク接続に適用して使用されています。本書の他の節では、この語は、特定の APPC の意味として使用されていますが、読者は、文脈からその意味を識別することができます。

本書のこの節では、*接続* という語は、 SQL 接続の概念を表しています。 SQL 接続は、明示的または暗黙的 SQL CONNECT が行われてから、論理 SQL 接続が、 SQL DISCONNECT または RELEASE の後に COMMIT が続いているものと同じ意味で終了されるまで存続します。複数の SQL 接続は、単一のネットワーク接続または会話上で連続して行うことができます。つまり、1 つの接続が終了すると、その接続を実行した会話が必ずしも終了するわけではありません。

アプリケーション・リクエスター (AR) 側で DRDA を使用して**アプリケーション・サーバー (AS)** に接続する際には、DDM 会話が使用されます。この会話は、AR からの SQL CONNECT ステートメントによって確立されますが、それは次の場合だけです。

- 同じリモート・ロケーション値を使用する会話が、AR ジョブについてまだ存在していない。
- 会話が同じ活動化グループを使用する。
- DDM から開始した場合は、会話のファイルは活動化グループに限っている。
- 会話は同じ会話タイプ (保護または非保護) である。

DDM 会話は、次の 3 つの状態のいずれかになります。つまり、活動状態、未使用、または停止のいずれかです。分散リレーショナル・データベースで使用される DDM 会話は、 AR が AS に接続されている間は、活動状態です。

接続を終了するには、SQL の DISCONNECT と RELEASE のステートメントが使用されます。 接続の終了は、サーバーによって暗黙に行うこともできます。 そのほか、RUW 接続管理と一緒に実行中のときは、前の接続は、CONNECT の実行によって終了します。接続の終了についての詳細は、 196 ページの『明示 CONNECT』を参照してください。接続が終了すると、DDM 会話は未使用または停止のいずれかになります。 DDM 会話が未使用の場合は、リモート・データベース管理システムに対する会話は、DDM 通信管理機能によって維持管理され、未使用とマークされます。 DDM 会話が停止すると、DDM 通信管理機能は会話を終了させます。活動状態でなくなった接続のための DDM 会話が未使用になるか停止になるかは、 DDMCNV ジョブ属性によって決まります。ジョブ属性値が *KEEP で、接続が他の iSeries server に対するものであるときは、会話は未使用になります。ジョブ属性値が *DROP であるか、または接続が他の iSeries server に対するものでないときは、会話は停止となります。

DDMCNV ジョブ属性として *KEEP の使用が望ましいのは、リモート・リレーショナル・データベースとの接続がしばしば変更される場合です。

次の場合には、*DROP の値を使用することが望まれます。

- 会話の維持管理コストが高く、しかも会話はどちらかというと直ちに使用するわけではない場合。
- RUW 接続管理を使用してコンパイルしたプログラムと、 DUW 接続管理を使用してコンパイルしたプログラムを一緒に実行する場合。リモート・ロケーションに対する RUW 接続管理を使用してコンパイルしたプログラムを実行しようとする場合は、保護会話があると、失敗します。
- DDM または DRDA のいずれかで保護会話を使用して実行する場合。未使用の保護会話に対して、コミットとロールバックに余分の負担がかかります。

DDM 会話が、DDM を介してリモート・ファイルに対する操作を行うためにも使用されている場合には、以下に挙げる条件が満たされるまで、会話は活動状態を維持します。

- 会話で使用されたすべてのファイルがクローズおよびロック解除される。
- 他に実行されている DDM 関連機能がない。
- 中断されている (たとえば、中断プログラムによって) DDM 関連機能がない。
- 保護会話において、すべての SQL プログラムが終了し、かつ、すべての DDM 関連機能が完了した後で、コミットまたはロールバックが実行された。
- AR ジョブの AS との接続が終わっている。

DDMCNV ジョブ属性の値には関係なく、ジョブ経路指定ステップの終了時、ジョブの終了時、またはジョブによるジョブ経路再指定 (RRTJOB) コマンドの開始時に、会話は停止します。活動ジョブ内の未使用会話は、DDM 会話再利用 (RCLDDMCNV) またはリソース再利用 (RCLRSC) コマンドによって停止することもできます。詳細については、『DDM リソースの再利用』を参照してください。また、通信回線障害などのエラーも、会話が停止する原因となります。

DDMCNV パラメーターは、ジョブ変更 (CHGJOB) コマンドによって変更され、 OPTION(*DFNA) を指定したジョブ表示 (DSPJOB) コマンドによって表示されます。 また、ジョブ属性検索 (RTVJOBA) コマンドを使用して、このパラメーターの値を取り出し、それを CL プログラムの中で使用することもできます。

DDM リソースの再利用

分散データ管理機能会話再利用 (RCLDDMCNV) コマンドは、ジョブの DDMCNV 属性値が *KEEP であったとしても、そのジョブで現在使用されていないすべてのアプリケーション会話を再利用します。このコマンドを使用すれば、すべてのオープン・ファイルをクローズしたり、 リソース再利用 (RCLRSC) コマンドで実行されるその他の機能のいずれかを実行せずに、未使用 DDM 会話を再利用することができます。

分散データ管理機能会話再利用 (RCLDDMCNV) コマンドは、そのコマンドが入力される**アプリケーション・リクエスター (AR)** 側のジョブの DDM 会話に適用されます。 AR ジョブで使用される DDM 会話には、対応する AS ジョブがあります。その AS ジョブは [6]、対応する DDM 会話が終了すると、自動的に終了します。

このコマンドは 1 つのジョブで使用されるすべての DDM 会話に適用されますが、このコマンドを使用しても、そのような DDM 会話のすべてが再利用されることにはなりません。会話が再利用されるのは、活動的に使用されていない場合だけです。コミットメント制御が使用されている場合には、 COMMIT または ROLLBACK 操作を行ってからでないと、DDM 会話は再利用できません。

プログラムで使用されるオブジェクトの表示

プログラム参照表示 (DSPPGMREF) コマンドを使用して、プログラムまたは SQL パッケージで使用されるテーブル、データ域、および他のプログラムを判別することができます。この情報が使用可能なのは、SQL パッケージおよびコンパイル済みプログラムの場合だけです。情報は、表示、印刷、またはデータベース出力ファイルへの書き出しができます。

プログラムまたはパッケージの作成時に、そのプログラムまたはパッケージの中で使用される特定のオブジェクトについての情報が保管されます。この情報は、後でプログラム参照表示 (DSPPGMREF) コマンドで使用することができます。 検索される情報には、次のものがあります。

- プログラムまたはパッケージの名前およびそのテキスト記述
- プログラムまたはパッケージが入っているライブラリーまたはコレクションの名前
- プログラム・パッケージによって参照されるオブジェクトの数
- サーバー・オブジェクトの修飾名
- 情報検索日付
- 参照されたオブジェクトのオブジェクト・タイプ

ファイルおよびテーブルの場合は、レコードに次のような追加のフィールドが入ります。

- プログラムまたはパッケージの中のファイルまたはテーブルの名前 (プログラムまたはパッケージの作成時に指定変更が有効であった場合には、サーバー・オブジェクト名とは異なることがあります)。

  注: 指定変更はいずれも**アプリケーション・リクエスター (AR)** 側でのみ適用されます。

---

6. 終了する TCP/IP 会話の場合、**アプリケーション・サーバー (AS)** ジョブは、通常、事前開始ジョブであり、通常は終了されるよりも再生されます。

- ファイルまたはテーブル (入力、出力、更新、未指定、またはこれら 4 つの組み合わせ) のプログラムまたはパッケージによる使用。
- 参照されたレコード様式 (もしあれば) の数。
- ファイルまたはテーブルで使用されるレコード様式の名前、およびそのレコード様式レベル識別コード。
- 各様式ごとに参照されたフィールドの数。

プログラムの中のオブジェクトが表示されるためには、ユーザーがあらかじめそのプログラムに対して *USE 権限を持っていなければなりません。また、ライブラリー修飾子で指定されたライブラリーの中で、ユーザーが読み取り権を持っているライブラリーだけが、プログラム探索の対象となります。

表 5 には、高水準言語およびユーティリティーが情報を保管するオブジェクトを示してあります。

表 5. 高水準言語によるオブジェクトに関する情報の保管

言語	ファイル	プログラム	データ域	注を参照
CL	する	する	する	1
COBOL/400* 言語	する	する	しない	2
PL/I	する	する	該当せず	2
RPG/400* 言語	する	しない	する	3
DB2 UDB SQL	する	該当せず	該当せず	4

注:

1. ファイル、プログラム、またはデータ域を参照するすべてのサーバー・コマンドでは、コマンドを CL プログラム内でコンパイルする時に情報を保管することをコマンド定義に指定する必要があります。 変数を使用する場合、変数の名前はオブジェクト名として使用されます (例: &FILE)。式を使用する場合、オブジェクトの名前は *EXPR として保管されます。ユーザー定義コマンドで、コマンドに指定されたファイル、プログラム、またはデータ域に関する情報を保管することもできます。 iSeries Information Center 中の『制御言語 (CL)』トピック中の PARM または ELEM コマンド・ステートメントの FILE、 PGM、および DTAARA の各パラメーターの説明を参照してください。
2. プログラム名が保管されるのは、COBOL/400® 識別コード (`CALL PGM1` などの動的呼び出し) ではなく、リテラルがプログラム名に使用されている場合 (`CALL 'PGM1'` などの静的呼び出し) だけです。
3. 内部データ域の使用は保管されません。
4. SQL パッケージについての情報。

保管されたファイル情報には、使用のタイプを表す項目 (番号) が含まれています。 プログラム参照表示 (DSPPGMREF) コマンドのデータベース・ファイル出力 (OUTFILE パラメーターの使用時に作成される) では、この項目は 1 つ以上のコード (下記にリストしてあります) を表すものです。 オブジェクト 1 つにつき 1 項目しか可能ではありませんが、組み合わせで使用することができます。たとえば、7 というコードのファイルであれば、入力、出力、および更新に使用されます。

コード 意味

1 入力

2 出力

3 入力および出力

4 更新

8 未指定

詳しくは、『プログラム参照表示の例』を参照してください。

プログラム参照表示の例

アプリケーション・リクエスター (AR) プログラムで使用されるオブジェクトを表示するには、次のようなコマンドを入力することができます。

```
DSPPGMREF PGM(SPIFFY/PARTS1) OBJTYPE(*PGM)
```

リクエスター側では、プログラムで使用されるすべてのコレクションおよびテーブルのリストを取り出すことができますが、それらが入っているリレーショナル・データベースがどれであるかを表示することはできません。複数のリレーショナル・データベースに入っている場合もあります。コマンドの出力は、データベース・ファイルまたは表示スプール・ファイルに渡すことができます。その出力は次のようになります。

```
                          スプール・ファイルの表示
ファイル . . . :   QPDSPPGM                       ページ／行  1/1
制御 . . . . . .                                  桁          1 - 75
検索 . . . . . .

3/29/92                プログラム参照の表示
DSPPGMREF コマンド入力
プログラム. . . . . . . . . . . . . . . . . :   PARTS1
ライブラリー. . . . . . . . . . . . . . . . :   SPIFFY
出力. . . . . . . . . . . . . . . . . . . . :   *
SQL パッケージの組み込み. . . . . . . . . . :   *YES
プログラム. . . . . . . . . . . . . . . . . :   PARTS1
ライブラリー. . . . . . . . . . . . . . . . :   SPIFFY
テキスト ' 記述 .'. . . . . . . . . . . . . :   部品の在庫チェック
参照されるオブジェクトの数. . . . . . . . . :   3
オブジェクト. . . . . . . . . . . . . . . . :   PARTS1
ライブラリー. . . . . . . . . . . . . . . . :   SPIFFY
オブジェクトのタイプ. . . . . . . . . . . . :   *PGM
オブジェクト. . . . . . . . . . . . . . . . :   QSQROUTE
ライブラリー. . . . . . . . . . . . . . . . :   *LIBL
オブジェクトのタイプ. . . . . . . . . . . . :   *PGM
オブジェクト. . . . . . . . . . . . . . . . :   INVENT
ライブラリー. . . . . . . . . . . . . . . . :   SPIFFY
       オブジェクトのタイプ  . . . . . . . :   *FILE
       プログラム中のファイル名. . . . . . :
       ファイルの使用法. . . . . . . . . . :   入力
```

アプリケーション・サーバー (AS) SQL パッケージで使用されるオブジェクトを表示する場合には、次のようなコマンドを入力することができます。

```
DSPPGMREF PGM(SPIFFY/PARTS1) OBJTYPE(*SQLPKG)
```

コマンドの出力は、データベース・ファイルまたは表示スプール・ファイルに渡すことができます。その出力は次のようになります。

```
                          スプール・ファイルの表示
ファイル . . . :   QPDSPPGM                       ページ／行  1/1
制御 . . . . . .                                  桁          1 - 75
検索 . . . . . .

3/29/92                プログラム参照の表示
DSPPGMREF コマンド入力
プログラム. . . . . . . . . . . . . . . . . :   PARTS1
ライブラリー. . . . . . . . . . . . . . . . :   SPIFFY
出力. . . . . . . . . . . . . . . . . . . . :   *
SQL パッケージの組み込み. . . . . . . . . . :   *YES
SQL パッケージ. . . . . . . . . . . . . . . :   PARTS1
ライブラリー. . . . . . . . . . . . . . . . :   SPIFFY
テキスト ' 記述 .'. . . . . . . . . . . . . :   部品の在庫チェック
参照されるオブジェクトの数. . . . . . . . . :   1
オブジェクト. . . . . . . . . . . . . . . . :   INVENT
```

```
ライブラリー. . . . . . . . . . . . . . . . :   SPIFFY
オブジェクトのタイプ. . . . . . . . . . . . :   *FILE
プログラム中のファイル名. . . . . . . . . . :
ファイルの使用法. . . . . . . . . . . . . . :   入力
```

分散リレーショナル・データベースからのコレクションの除去

ジャーナル・レシーバーが入っているコレクションを削除しようとすると、**アプリケーション・サーバー (AS)** ジョブの QSYSOPR メッセージ待ち行列に照会メッセージが送られます。この照会に対する応答があるまで、 AS および**アプリケーション・リクエスター (AR)** ジョブは待機します。

メッセージ待ち行列に送られるメッセージを次に示します。

CPA7025

(I C) *ライブラリー* 内のレシーバー (*名前*) は完全に保管されていない。

AR ジョブが待機している時に、アプリケーションが停止したような状態になることがあります。 AR ジョブが予想以上に長時間待機したままである場合は、下記の事項を検討してください。

- 照会メッセージが QSYSOPR メッセージ待ち行列に送られていて、応答があるまで先へ進めない状態になっていないかどうかを確認する。
- AS のサーバー応答リストを用いて、メッセージに対して AS に応答させる。

注: アプリケーションがこのような「停止したような」状態になると、動かなくなります。したがって、オブジェクトの移動 (MOVOBJ) コマンドを使用して、ジャーナル・レシーバーを別のライブラリーに移動できません。また、ジャーナル・レシーバーを別のライブラリーに保管したり復元したりすることもできません。 ジャーナル・レシーバー作成 (CRTJRNRCV) コマンドを使用して新しいジャーナル・レシーバーを別のライブラリー中に作成することと、 ジャーナル変更 (CHGJRN) コマンドを使用してジャーナル・レシーバーをジャーナルに付加することだけ行えます。ジャーナル変更 (CHGJRN) コマンドに JRNRCV(*GEN) パラメーターを指定して使用した際に、システムによって新しく作成されるジャーナル・レシーバーは、新しいライブラリー中に作成されます。ジャーナルを保管する際に、別のライブラリー中のレシーバーが付加されている場合は、保管したバージョンのジャーナルを復元する際には、新しいジャーナル・レシーバーが他のライブラリー中に作成されます。ジャーナル処理に関する詳細は、 iSeries Information Center の『ジャーナル管理』のトピックを参照してください。

AS のサーバー応答リストを使用してメッセージに対して AS 応答を行うには、現在停止したような状態になっているジョブを変更するか、あるいは、システムで実行しているすべての AS ジョブのジョブ記述を変更します。しかし、まず、 応答リスト項目追加 (ADDRPYLE) コマンドを使用して、**アプリケーション・サーバー (AS)** 応答リストにメッセージ CPA7025 についての項目を追加しなければなりません。

```
ADDRPYLE  SEQNBR(...)  MSGID(CPA7025) RPY(I)
```

AS で現在実行中のジョブのジョブ記述を変更するには、 SBMRMTCMD (リモート・コマンド投入) コマンドを使用してください。下記の例では、カンザス・シティー地区にあるサーバーのデータベース管理担当者が、 KC105 サーバー (TEST/KC105TST DDM ファイルによってアドレス指定されているシステム) にあるジョブ記述を変更しています。

```
SBMRMTCMD  CMD('CHGJOB JOB(KC105ASJOB) INQMSGRPY(*SYSRPYL)')
   DDMFILE(TEST/KC105TST)
```

ジョブ記述変更 (CHGJOBD) コマンドを使用して、該当のジョブ記述を使用するすべてのジョブがサーバー応答リストを使うようにしておけば、 AS で上記のような状態が発生するのを永続的に防ぐことができます。 下記の例は、同じ AS で CHGJOBD コマンドを入力する方法を示しています。

```
CHGJOBD JOBD(KC105ASJOB)  INQMSGRPY(*SYSRPYL)
```

この方法を用いる場合は、注意が必要です。サーバー応答リストに CPA7025 を追加すると、そのサーバー応答リストを使用するすべてのジョブに影響が及びます。 ジョブ記述の変更も、そのジョブ記述を使用するすべてのジョブに影響を及ぼします。したがって、AS ジョブのそれぞれに別々のジョブ記述を作成することも考えてみてください。ジョブ記述の作成については、 iSeries Information Center 中の『実行管理機能』をご覧ください。

分散リレーショナル・データベースのジョブ会計

iSeries server のジョブ会計機能は、データを収集して、誰がサーバーを使用し、どんなサーバー・リソースを使用しているかを判別できるようにします。典型的なジョブ会計では、サーバーで実行されるジョブ、および処理装置、印刷装置、表示装置の使用など、使用されたリソース、ならびにデータベースおよび通信機能の詳細を示します。

ジョブ会計は任意選択で、サーバーにセットアップしなければなりません。 サーバーにリソース会計をセットアップするには、次のことを行わなければなりません。

1. ジャーナル・レシーバー作成 (CRTJRNRCV) コマンドを使用することによって、ジャーナル・レシーバーを作成してください。
2. ジャーナル作成 (CRTJRN) コマンドを使用することによって、 QSYS/QACGJRN という名前のジャーナルを作成してください。 QSYS/QACGJRN という名前を使用しなければならず、しかもこのジャーナルを作成するには、 QSYS に項目を追加する権限を持っていなければなりません。このコマンドには、前のステップで作成したジャーナル・レシーバーの名前を指定してください。
3. システム値処理 (WRKSYSVAL) コマンドまたはシステム値変更 (CHGSYSVAL) コマンドを使用して、会計レベルのサーバー値 QACGLVL を変更してください。

   システム値変更 (CHGSYSVAL) コマンドの VALUE パラメーターによって、いつジョブ会計ジャーナル項目が作成されるかが決まります。 *NONE の値は、サーバーがジョブ会計ジャーナルの中に項目を作成しないことを意味します。 *JOB の値は、サーバーがジョブ (JB) ジャーナル項目を作成することを意味します。 値が *PRINT であれば、印刷されるファイルごとに、直接印刷 (DP) またはスプール印刷 (SP) ジャーナル項目が作成されます。

ジョブが開始されると、ジョブ記述がジョブに割り当てられます。ジョブ記述オブジェクトには、会計コード (ACGCDE) パラメーターの値が入りますが、これは会計コードまたは省略時の値 *USRPRF です。 *USRPRF が指定された場合には、ジョブのユーザー・プロファイルの中の会計コードが使用されます。

ユーザー・プロファイル作成 (CRTUSRPRF) コマンドまたはユーザー・プロファイル変更 (CHGUSRPRF) コマンドで会計コード・パラメーター ACGCDE を使用して、会計コードをユーザー・プロファイルに追加することができます。 ジョブ記述作成 (CRTJOBD) コマンドまたはジョブ記述変更 (CHGJOBD) コマンドの ACGCDE パラメーターに所要の会計コードを指定することによって、特定のジョブ記述の会計コードを変更することができます。

ジョブ会計ジャーナルがセットアップされると、 システム値変更 (CHGSYSVAL) コマンドが有効になった後で次にサーバーに入るジョブから始めて、ジョブ会計項目がジャーナル・レシーバーに入れられます。

ジャーナル表示 (DSPJRN) コマンドで OUTFILE パラメーターを使用して、処理することのできるデータベース・ファイルに会計項目を書き出すことができます。

ジョブ会計については、 iSeries Information Center 中の『実行管理機能』をご覧ください。

TCP/IP サーバーの管理

ここでは、 TCP 上でソケットを使用して通信する DRDA/DDM サーバー・ジョブを管理する方法について説明します。また、サーバーが実行するサブシステム、サーバーに影響を及ぼすオブジェクト、およびそれらのリソースを管理する方法について説明します。

DRDA/DDM TCP/IP サーバーは、OS/400 プログラムに付属しており、通常は、正しく作動させるために既存のシステム構成に変更を加える必要はありません。それは、OS/400 のインストール時にセットアップされ、構成されます。ある時点で、要件をよりよく満たしたり、問題を解決したり、サーバーのパフォーマンスを改善したり、または単にサーバー上でジョブを見るために、システムがサーバー・ジョブを管理する方法を変更することができます。そのような変更を行って、処理要件を満たすには、オブジェクトがシステムのどの部分に影響を及ぼし、それらのオブジェクトをどのように変更するかを知っている必要があります。

ここでは、サーバー・ジョブでの作業を行なうために理解しておく必要のある、高水準のいくつかの作業管理の概念と、それらの概念やオブジェクトがどのようにサーバーに関連するかについて説明します。
iSeries サーバーを管理する方法を完全に理解するには、この部分を読み進める前に、 iSeries Information Center 内の『実行管理機能』を注意深く検討することをお勧めします。その後、この節で、 TCP/IP サーバーを管理する方法や、システムの残りの部分と適合させる方法について示します。

詳細については、以下のトピックを参照してください。

- DRDA TCP/IP サーバーの用語
- DDM の TCP/IP 通信サポートの概念
- DRDA/DDM サーバー・ジョブ
- DDM サーバー・ジョブ・サブシステムの構成
- サーバー・ジョブの識別

DRDA TCP/IP サーバーの用語

DB2 UDB for iSeries への DDM および DRDA TCP/IP の両方のアクセスのために同じサーバー・ソフトウェアが使用されます。簡潔さのために、以下の説明の中では、*DRDA/DDM サーバー* ではなく、 *DDM サーバー* という用語を使用します。 ただし、文脈により修飾子の使用が不必要なときには、時折、*TCP/IP サーバー*、*DRDA サーバー*、または単にサーバー と表現することもあります。

DDM サーバーは、2 つ以上のジョブから成り立っています。 1 つは DDM *リスナー* と呼ばれ、接続要求を聴取して、その他のジョブへの作業をディスパッチします。 その他のジョブは、最初の構成時に、初期接続が行われた後に DRDA または DDM クライアントから要求を出す事前開始ジョブです。関連したすべてのジョブのセット (リスナーおよびサーバー・ジョブ) は、ひとまとめにして *DDM サーバー* と呼ばれます。

クライアント という用語は、DRDA アプリケーション環境では、*DRDA アプリケーション・リクエスター* (または AR) と相互に交換可能な語として使用されます。クライアントという用語は、DDM (分散ファイル管理機能) アプリケーション環境では、 *DDM ソース・システム* と相互に交換可能な語として使用されます。

サーバー という用語は、DRDA アプリケーション環境では、 *DRDA アプリケーション・サーバー* (または AS) と相互に交換可能な語として使用されます。クライアントという用語は、DDM (分散ファイル管理機能) アプリケーション環境では、 *DDM ターゲット・システム* と相互に交換可能な語として使用されます。 (文脈によっては、 iSeries システム (ハードウェア) のことをサーバーや iSeries サーバーともいうことに注意してください。)

DDM の TCP/IP 通信サポートの概念

特に、DRDA および DDM によって使用される TCP/IP 通信サポートに関係のあるいくつかの概念があります。ここでは、これらの概念の詳細について説明します。

TCP/IP 上での DRDA または DDM 接続の確立

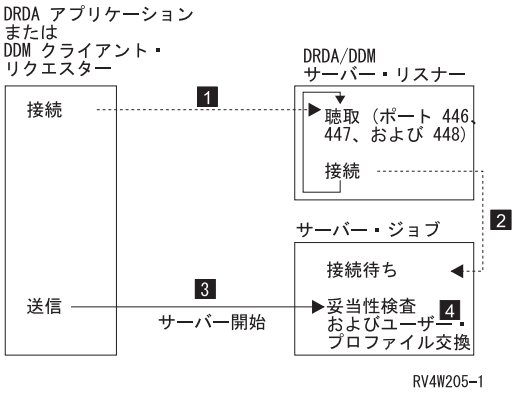


図 10. DRDA/DDM TCP/IP サーバー

TCP/IP 通信サポートを使用する DDM サーバー・ジョブを開始するには、 DRDA アプリケーション・リクエスターまたは DDM ソース・システムは、割り当て済みポート番号 446 または 447 に接続します。DDM サーバーは、ポート 448 上でも listen しますが、セキュア・ソケット (SSL) 接続 (DB2 UDB for iSeries アプリケーション・リクエスターまたは DDM クライアントでは未サポート) でのみ使用されます。 **1** 。 DDM リスナー・プログラムは、クライアントの接続要求を聴取し、受け入れるために、(TCP/IP サーバーの開始 (STRTCPSVR SERVER(*DDM)) を使用することによって) 開始されていなければなりません。 DDM リスナーは、この接続要求を受け入れるときに、クライアントの接続を DDM サーバー・ジョブに接続するように内部要求を出します **2** 。 このサーバー・ジョブは、事前開始ジョブか、またはユーザーが QRWTSRVR サブシステムから QRWTSRVR 事前開始ジョブ項目を除去した場合 (事前開始ジョブが使用されていない場合)、クライアント接続要求が処理されるときに実行依頼されるバッチ・ジョブです。サーバー・ジョブは、クライアントとのそれ以降の通信を処理します。

発生する初期データ交換には、サーバー・ジョブが実行することになっているユーザー・プロファイルを識別する要求が含まれています **3** 。いったんユーザー・プロファイルおよびパスワード (ユーザー・プロファイル ID とともに送信される場合) が妥当性検査されると、サーバー・ジョブは、ジョブを変更するだけでなくこのユーザー・プロファイルにスワップして、ユーザー・プロファイル用に定義された CCSID などの属性を使用します **4** 。

リスナー・プログラムに接続し、サーバー・ジョブにクライアント接続を接続してデータを交換しユーザー・プロファイルとパスワードを妥当性検査する機能は、 APPC プログラム開始要求が処理されるときに実行される機能に相当します。

DRDA/DDM リスナー・プログラム

DDM リスナー・プログラムはバッチ・ジョブで実行されます。 DDM リスナーと実際のサーバー・ジョブとの間には、1 対多の関係があります。 1 つのリスナーに多数の DDM サーバー・ジョブがある可能性があります。サーバー・ジョブは、通常、事前開始ジョブです。リスナー・ジョブは、 QSYSWRK サブシステム内で実行されます。

DDM リスナーにより、クライアント・アプリケーションが、インバウンド接続要求を処理し経路指定することによって、関連したサーバー・ジョブとの TCP/IP 接続を確立することができるようになります。 いったんクライアントがサーバー・ジョブとの通信を確立すると、その接続の期間中はクライアントとリスナーとの間の関連はなくなります。

DDM リスナーは、 DRDA アプリケーション・リクエスターおよび DDM ソース・システムが DDM TCP/IP サーバーとの接続を確立するために、活動状態でなければなりません。 DDM TCP/IP 属性の変更 (CHGDDMTCPA) コマンドを使用するか、または iSeries ナビゲーターを使用することによって、 DRDA リスナーが自動的に開始されることを要求することができます。 iSeries ナビゲーターで、 DDM 設定にナビゲートします: 「**ネットワーク**」->「**サーバー**」->「**TCP/IP**」。これにより、TCP/IP が開始されるときにリスナーも開始されることになります。 DRDA リスナーを開始するとき、QSYSWRK サブシステムと TCP/IP の両方が活動状態でなければなりません。

TCP/IP サーバーの開始 (STRTCPSVR) CL コマンド

TCP/IP サーバーの開始 (STRTCPSVR) コマンドは、 *DDM または *ALL という SERVER パラメーター値を使って、リスナーを開始するために使用されます。

DDM リスナーの制約事項: 一度に活動状態にできる DDM リスナーは 1 つだけです。すでに活動状態である場合にリスナーの開始要求を行うと、コマンド発行者に通知メッセージが出されます。

注: DDM サーバーは、QUSER パスワードの有効期限が切れている場合には開始しません。 QUSER プロファイルでは、パスワードの有効期限の間隔を *NOMAX に設定することをお勧めします。この値を指定すると、パスワードの有効期限はなくなります。

例: TCP/IP サーバーの開始 (STRTCPSVR) CL コマンド: 例 1: すべての TCP/IP サーバーの開始

```
STRTCPSVR SERVER(*ALL)
```

コマンドは、DDM サーバーを含むすべての TCP/IP サーバーを開始します。

例 2: DDM TCP/IP サーバーの開始

```
STRTCPSVR *DDM
```

このコマンドは、DDM TCP/IP サーバーのみを開始します。

TCP/IP サーバーの終了 (ENDTCPSVR) CL コマンド

TCP/IP サーバーの終了 (ENDTCPSVR) コマンドは、DDM サーバーを終了します。

DDM リスナーが終了され、クライアント・アプリケーションへの活動状態の接続を持つ関連したサーバーがある場合、サーバー・ジョブは、クライアント・アプリケーションとの通信が終了するまで活動状態のままです。 ただし、リスナーが再び開始されるまで、クライアント・アプリケーションからの連続接続要求は失敗します。

TCP/IP サーバーの終了に関する制約事項: DDM リスナーが活動状態でないときに、 DDM リスナーを終了するために TCP/IP サーバーの終了 (ENDTCPSVR) コマンドを使用すると、診断メッセージが発行されます。 (ENDTCPSVR) SERVER(*ALL) コマンドが発行されるときにリスナーが活動状態でない場合には、これと同じ診断メッセージは送信されません。

TCP/IP サーバーの終了の例: 例 1: すべての TCP/IP サーバーの終了

```
ENDTCPSVR *ALL
```

このコマンドは、すべての活動状態の TCP/IP サーバーを終了します。

例 2: DDM サーバーの終了

```
ENDTCPSVR SERVER(*DDM)
```

このコマンドは、DDM サーバーを終了します。

iSeries ナビゲーターでの DDM リスナーの開始

iSeries Access Family の一部である iSeries ナビゲーターを使用して DDM リスナーを管理することもできます。そのためには、次のパスに従います。「**ネットワーク**」->「**サーバー**」->「**TCP/IP**」ディレクトリー。

DRDA/DDM サーバー・ジョブ

サブシステム記述および事前開始ジョブ項目と DDM

サブシステム記述は、どのように、どこに、どのくらいの作業をサブシステムに入れるか、そしてサブシステムが作業を実行するためにどのリソースを使用するかを定義します。以下では、 QUSRWRK (V5R2 より前のバージョンでは QSYSWRK) サブシステム内の事前開始ジョブ項目が DDM サーバーに及ぼす影響について説明します。

事前開始ジョブは、アプリケーション・リクエスター (AR) がサーバーとの通信を開始する前に実行を開始するバッチ・ジョブです。事前開始ジョブは、サブシステム記述内の事前開始ジョブ項目を使用して、ジョブを開始するときにどのプログラム、クラス、および記憶域プールを使用するかを判別します。事前開始ジョブ項目内に、サブシステムが事前開始ジョブのプールを作成および管理するために使用する属性を指定しなければなりません。

事前開始ジョブは、サーバーへの接続を開始するときのパフォーマンスを向上させます。事前開始ジョブ項目は、サブシステム内で定義されます。事前開始ジョブは、サブシステムが開始されるときに活動状態になるか、または事前開始ジョブの開始 (STRPJ) および事前開始ジョブの終了 (ENDPJ) コマンドで制御できます。

DRDA/DDM 事前開始ジョブ

事前開始ジョブ (活動事前開始ジョブの表示 (DSPACTPJ) コマンドなど) に関係のあるサーバー情報は、その情報が TCP/IP 接続要求の結果として開始された事前開始ジョブに関係がある場合でさえ、事前開始ジョブを開始するために行われる要求を示すために、専ら「プログラム開始要求」という用語を使用します。

以下のリストには、 DDM TCP/IP サーバー用の初期構成値を持つ事前開始ジョブ項目属性が含まれています。 事前開始ジョブ項目属性は、 事前開始ジョブ項目変更 (CHGPJE) コマンドを使用して変更できます。

- サブシステム記述。事前開始ジョブ項目を含むサブシステムは、 V5R2 では QUSRWRK です。それ以前のリリースでは、QSYSWRK です。
- プログラム・ライブラリーおよび名前。事前開始ジョブが開始されるときに呼び出されるプログラムは、QSYS/QRWTSRVR です。
- ユーザー・プロファイル。ジョブが実行するユーザー・プロファイルは、QUSER です。これは、ジョブがユーザー・プロファイルとして示すものです。サーバーへの接続要求がクライアントから受信されると、事前開始ジョブ機能は、その要求で受信されるユーザー・プロファイルにスワップします。
- ジョブ名。事前開始ジョブが開始されるときのジョブ名は、QRWTSRVR です。
- ジョブ記述。事前開始ジョブに使用されるジョブ記述は、*USRPRF です。ユーザー・プロファイルは QUSER なので、 QUSER のジョブ記述が何であっても *USRPRF になることに注意してください。た

だし、ジョブの属性は、ユーザー ID およびパスワード (存在すれば) が検証された後に、要求しているユーザーのジョブ記述に対応するように変更されます。

- ジョブ開始。これは、サブシステムの開始時に事前開始ジョブが自動的に開始するかどうかを示します。これらの事前開始ジョブ項目には、 *YES という開始ジョブ値が出荷時に設定されています。システム IPL の実行時に不必要なジョブが開始しないように、これらの値を *NO に変更してもかまいません。**注:** DDM サーバー・ジョブが実行中でなく、 DDM リスナー・ジョブがバッチの場合は、即時 DDM サーバー・ジョブは依然として QSYSWRK サブシステムの下で実行されます。
- 初期ジョブ数。最初に構成されるとき、サブシステムの開始時に開始されるジョブの数は 1 です。この値は、特定の環境および必要に合わせて調整することができます。
- 限界値。事前開始ジョブ項目の使用可能な事前開始ジョブの最小数は 1 に設定されます。この限界値に達すると、追加の事前開始ジョブが自動的に開始されます。これは、プール内に、ある特定のジョブ数を保持するために使用されます。
- 追加のジョブ数。限界値に達するときに開始される追加の事前開始ジョブ数は、最初 2 に構成されます。
- ジョブの最大数。この項目のために活動状態になることのできる事前開始ジョブの最大数は *NOMAX です。
- 最大使用数。ジョブの最大使用数は 200 に設定されます。この値は、サーバーを開始するための要求が 200 処理された後に事前開始ジョブが終了することを示しています。ある特定の状況では、 TCP/IP サーバーが正しく機能するために MAXUSE パラメーターを 1 に設定する必要があります。サーバーがある特定の ILE ストアード・プロシージャーを実行するとき、破棄されたオブジェクトへのポインターが事前開始ジョブ環境内に残っていることがあり、事前開始ジョブを続けて使用すると、MCH3402 例外が発生します。 V5R2 では、OS/400 中に変更が加えられており、例外が発生する可能性は最小限に抑えられています。
- ジョブの待機。 *YES を設定すると、ジョブの最大数に達した場合に、クライアント接続要求は使用可能なサーバー・ジョブを待機します。
- プール識別コード。この事前開始ジョブが実行するサブシステム・プール識別コードは 1 に設定されます。
- クラス。事前開始ジョブが実行するクラスの名前およびライブラリーは QSYS/QSYSCLS20 に設定されます。

事前開始ジョブ項目の開始ジョブ値が *YES に設定されており、残りの値が初期設定値で提供されるような場合、それぞれの事前開始ジョブ項目ごとに以下のことが起こります。

- サブシステムが開始されるとき、1 つの事前開始ジョブが開始されます。
- 最初のクライアント接続要求が TCP/IP サーバーに対して処理されるとき、最初のジョブが使用され、限界値を超えます。
- 事前開始ジョブ項目内に定義されている数に基づいて、サーバーに対して追加のジョブが開始されます。
- 使用可能なジョブの数は 1 未満にはなりません。
- サブシステムは、プール内の未使用の事前開始ジョブの数を定期的に調べ、余分なジョブを終了します。サブシステムは、常に、少なくとも初期ジョブ・パラメーター内に指定されている事前開始ジョブの数を残します。

事前開始ジョブのモニター: 事前開始ジョブは、 活動事前開始ジョブの表示 (DSPACTPJ) コマンドを使用してモニターすることができます。

(DSPACTPJ) コマンドは、以下の情報を提供します。

- 事前開始ジョブの現行数
- 事前開始ジョブの平均数
- 事前開始ジョブの最大数
- 使用している事前開始ジョブの現行数
- 使用している事前開始ジョブの平均数
- 使用している事前開始ジョブの最大数
- 接続要求を待っている事前開始ジョブの現行数
- 接続要求を待っている事前開始ジョブの平均数
- 接続要求を待っている事前開始ジョブの最高数
- 平均待ち時間
- 受け入れられた接続要求の数
- 拒否された接続要求の数

事前開始ジョブの管理: 活動状態の事前開始ジョブについて表示されている情報は、「活動中の事前開始ジョブの表示」画面にいる間に F5 キーを押して最新表示することができます。特に重要なのは、プログラム開始要求に関する情報です。この情報は、使用可能な事前開始ジョブの数を変更する必要があるかどうかを示します。プログラム開始要求が、使用可能な事前開始ジョブを待っていることを示す情報がある場合、 事前開始ジョブ項目変更 (CHGPJE) コマンドを使用して事前開始ジョブを変更することができます。

プログラム開始要求が十分な速さで実行されていなかった場合、以下のいずれかの組み合わせを行うことができます。

- 限界値を増やします。
- 最初のジョブ数 (INLJOBS) パラメーター値を増やします。
- 追加のジョブ数 (ADLJOBS) パラメーター値を増やします。

肝要な点は、サーバー・ジョブを開始する要求が送信されるごとに、使用可能な事前開始ジョブが確実にあるようにすることです。

事前開始ジョブ項目の除去: サーバーが事前開始ジョブ機能を使用しないようにするには、以下のことを行わなければなりません。

1. 事前開始ジョブ終了 (ENDPJ) コマンドを使用して、事前開始ジョブを終了します。

   事前開始ジョブ項目内に開始ジョブ *YES が指定されている場合、 ENDPJ コマンドを使用して終了された事前開始ジョブは、次回サブシステムが開始されるときに開始されます。事前開始ジョブを終了するだけで、次のステップを実行しない場合、特定のサーバーの開始要求はいずれも失敗します。
2. 事前開始ジョブ項目除去 (RMVPJE) コマンドを使用して、サブシステム記述内の事前ジョブ項目を除去します。

   (RMVPJE) コマンドを使用して除去される事前開始ジョブ項目は、サブシステム記述から永久に除去されます。 いったん項目が除去されると、サーバーに対する新しい要求は成功しますが、ジョブ開始のパフォーマンス・オーバーヘッドが発生します。

項目の経路指定: OS/400 ジョブがサブシステムに経路指定されるとき、これは、サブシステム記述内の経路指定項目を使用して行われます。 QSYSWRK サブシステム内のリスナー・ジョブ用の経路指定項目は、OS/400 のインストール後に表示されます。 このジョブは、QUSER ユーザー・プロファイルの下で開始され、 QSYSNOMAX ジョブ待ち行列が使用されます。

V5R2 より前のバージョンでは、サーバー・ジョブは QSYSWRK サブシステム中で実行されていました。V5R2 では、サーバー・ジョブはデフォルトでは QUSRWRK 内で実行されます。サーバー・ジョブの特性は、OS/400 で自動的に構成されることにもなる事前開始ジョブ項目からとられます。事前開始ジョブがサーバーに対して使用されないようにするためにこの項目が除去される場合、サーバー・ジョブは、対応するリスナー・ジョブの特性を使用して開始されます。

以下に、リスナー・ジョブ用の QSYSWRK サブシステム内の初期構成を提示します。

サブシステム QSYSWRK

ジョブ待ち行列
QSYSNOMAX

ユーザー QUSER

経路指定データ
QRWTLSTN

ジョブ名 QRWTLSTN

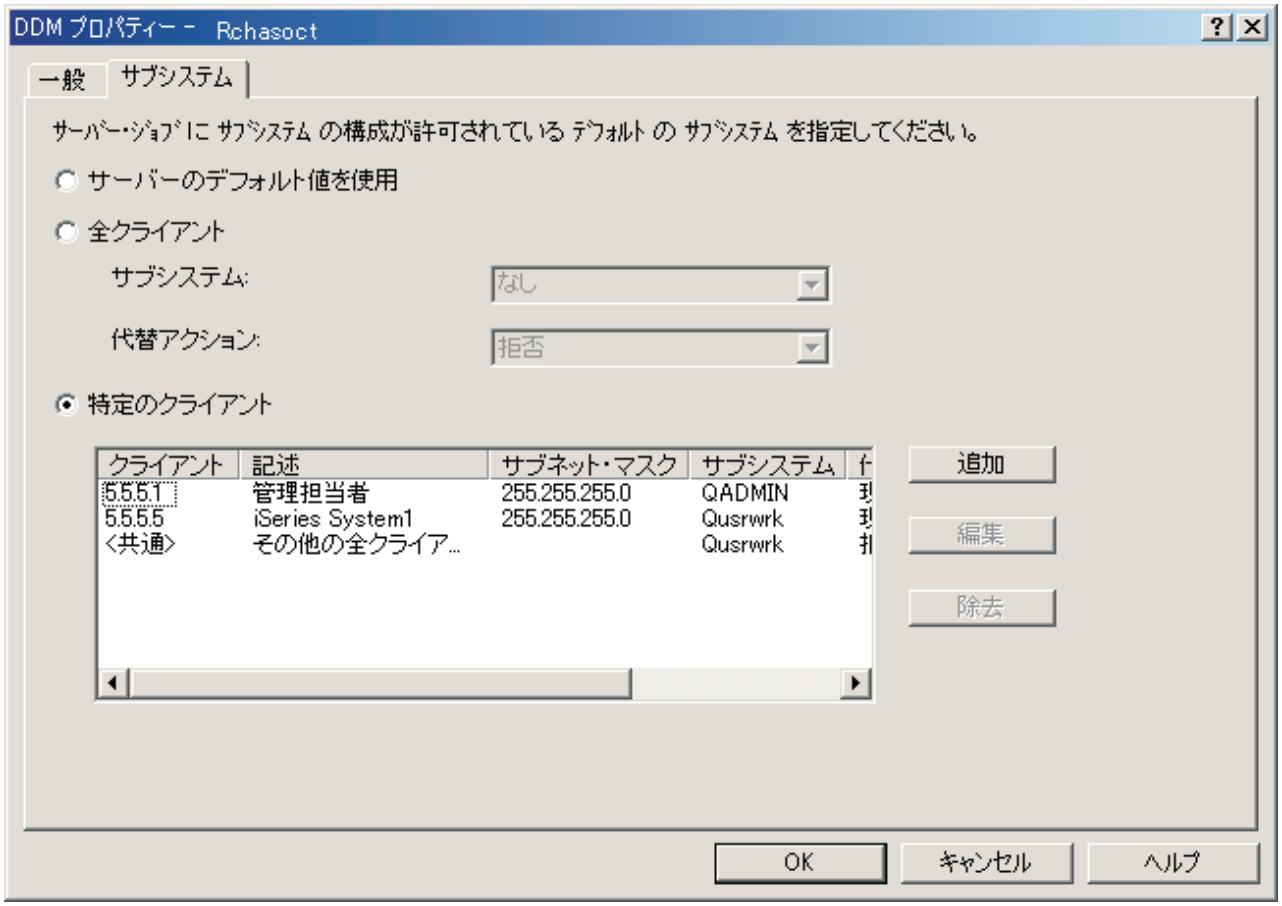
クラス QSYSCLS20

DDM サーバー・ジョブ・サブシステムの構成

V5R2 以降、デフォルトでは DDM TCP/IP サーバー・ジョブは QUSRWRK サブシステム内で実行されます。 iSeries ナビゲーターを使用して、クライアントの IP アドレスを基にした代替サブシステム中ですべてまたは特定のサーバー・ジョブを実行するように、 DDM サーバー・ジョブを構成できます。この構成をセットアップするには、次のようにします。

1. 事前開始ジョブ項目の追加 (ADDPJE) コマンドを使用して、ご希望のサブシステムの事前開始ジョブ項目を作成します。事前開始ジョブの属性について詳しくは、 114 ページの『DRDA/DDM 事前開始ジョブ』を参照してください。
2. 事前開始ジョブの開始 (STRPJ) コマンドを使用して、作成した事前開始ジョブ項目を開始します。
3. iSeries ナビゲーターで、「**ネットワーク**」を展開します。
4. 「**サーバー**」を展開します。
5. 「**TCP/IP**」をクリックします。
6. 右側のパネルに表示されているサーバーのリスト中で、「**DDM**」を右クリックし、「**プロパティー**」を選択します。
7. 「**サブシステム**」タブで、特定のクライアントとサブシステムの名前を追加します。

以下の例で、管理担当者は QADMIN に接続して実行でき、ネットワーク中の別のサーバーは QUSRWRK に接続して実行できます。他のクライアントはすべて拒否されます。



サーバー・ジョブの識別

サーバー上で開始されているサーバー・ジョブを見ると、サーバー・ジョブをある特定のアプリケーション・リクエスター・ジョブまたはある特定の PC クライアントに関連付けることは困難であることが分かるでしょう。 ある特定のジョブを識別できることは、問題を調査し、パフォーマンス・データを収集するための前提条件です。 iSeries ナビゲーターは、ジョブをより容易にするこれらのタスク用のサポートを提供します。

ここでは、iSeries ナビゲーターを使用していないときにデバッグまたはパフォーマンス調査を開始する前に、サーバー・ジョブを識別する方法に関する情報を提供します。

iSeries ジョブ名

iSeries 上で使用されているジョブ名は、以下の 3 つの部分から成り立っています。

- 単純ジョブ名
- ユーザー ID
- ジョブ番号 (昇順)

DDM サーバー・ジョブは、以下の規則に従います。

- ジョブ名は、QRWTSRVR です。
- ユーザー ID
  - 事前開始ジョブが使用されていてもいなくても常に QUSER になります。

– ジョブ・ログは、ユーザーが現在ジョブを使用していることを示します。

- ジョブ番号は、実行管理機能によって作成されます。

サーバー・ジョブの表示

サーバー・ジョブを識別するのに助けとなるために使用できる 3 つの方法があります。 1 つ目の方法は、WRKACTJOB (活動ジョブの処理) コマンドを使用することです。 2 番目の方法は、 ユーザー・ジョブの処理 (WRKUSRJOB) コマンドを使用することです。 3 番目の方法は、どのジョブがどのクライアント・ユーザーによって使用されているかを判別するためにヒストリー・ログを表示することです。

WRKACTJOB を使用した活動ジョブの表示: WRKACTJOB (活動ジョブの処理) コマンドは、すべての活動ジョブを表示します。リスナー・ジョブと同様に、すべてのサーバー・ジョブが表示されます。

以下の図は、(WRKACTJOB) コマンドを使用した状況例を示しています。 この図には、サーバーに関連したジョブのみが示されています。使用可能な事前開始ジョブを表示するには、F14 を押さなければなりません。

この図には、以下のタイプのジョブが示されています。

- **1** - リスナー・ジョブ
- **2** - 事前開始サーバー・ジョブ

```
                          活動ジョブの処理                              AS400597
                                                              99/11/05  11:35:49
CPU %:    3.1     経過時間 :   21:38:40     活動ジョブ数 :   77

オプションを入力して、実行キーを押してください。
   2= 変更    3= 保留      4= 終了    5= 処理    6= 解放    7= メッセージの表示
   8= スプール・ファイルの処理    13= 切断 ...

OPT  サブシステム/ ユーザー       タイプ  CPU %    機能       状況
         ジョブ
           .
___    QUSRWRK         QSYS           SBS             .0         DEQW
           .
___  1
       QRWTLSTN        QUSER          BCH             .0         SELW
           .
           .
___  2
       QRWTSRVR        QUSER          PJ              .0         TIMW
___    QRWTSRVR        QUSER          PJ              .0         TIMW
___    QRWTSRVR        QUSER          PJ              .0         TIMW
___    QRWTSRVR        QUSER          PJ              .0         TIMW
___    QRWTSRVR        QUSER          PJ              .0         TIMW
                                                                  続く ...
```

以下のタイプのジョブが示されています。

PJ 事前開始サーバー・ジョブ。

SBS サブシステム・モニター・ジョブ。

BCH リスナー・ジョブ。

WRKUSRJOB を使用した活動ユーザー・ジョブの表示: ユーザー・ジョブの処理 (WRKUSRJOB) コマンド USER(QUSER) STATUS(*ACTIVE) は、 QUSER の下で実行しているすべての活動サーバー・ジョブを表示します。その中には、DDM リスナー・ジョブやすべての DDM サーバー・ジョブが含まれます。このコマンドを使用すると、ほんの少数のジョブしかリストされないため、 DDM 関連のジョブを見つけるためにざっと目を通すのに便利です。

ヒストリー・ログ (活動記録ログ) の表示

クライアント・ユーザーがサーバーとの正常な接続を確立するたびに、そのジョブは、そのクライアント・ユーザーのプロファイルの下で実行するためにスワップされます。特定のクライアント・ユーザーと関連したジョブを判別するには、 ログの表示 (DSPLOG) コマンドを使用してヒストリー・ログを表示することができます。提供されている情報の例が、以下の図に示されています。

```
                    活動記録ログの内容の表示
      .
      .
DDM ジョブ 036995/QUSER/QRWTSRVR は  01/01/09 の 15:26:43 にユーザー MEL にサービス中。
      .
DDM ジョブ 036995/QUSER/QRWTSRVR は  01/01/09 の 15:45:08 にユーザー REBECCA にサービス中。
      .
DDM ジョブ 036995/QUSER/QRWTSRVR は  01/01/09 の 15:56:21 にユーザー NANCY にサービス中。
      .
DDM ジョブ 036995/QUSER/QRWTSRVR は  01/01/09 の 16:02:59 にユーザー ROD にサービス中。
      .
DDM ジョブ 036995/QUSER/QRWTSRVR は  01/01/09 の 16:48:13 にユーザー SMITH にサービス中。
      .
DDM ジョブ 036995/QUSER/QRWTSRVR は  01/01/09 の 17:10:27 にユーザー DAVID にサービス中。
      .
      .
      .
      .


続行するには、実行キーを押してください。

F3=終了    F10=すべての表示    F12=取り消し
```

注: ログの表示 (DSPLOG) コマンドに MSGID パラメーターを指定して使用し、不要な項目をフィルターで取り除く方法の例を以下に示します。

```
DSPLOG MSGID(CPI3E34)
```

QRWOPTIONS データ域に該当するオプションを設定して、これらのレコードがヒストリー・ログに書き込まれないようにすることもできます。詳しくは、『QRWOPTIONS データ域の使用法』のトピックを参照してください。

リレーショナル・データベース・ディレクトリーの監査

リレーショナル・データベース・ディレクトリーへのアクセスは、次のいずれかの場合にセキュリティー監査ジャーナルに記録されます。

- システムの QAUDLVL の値が *SYSMGT である。
- ユーザーの AUDLVL の値が *SYSMGT である。

値が *SYSMGT 値である場合、サーバーは、次のコマンドによって行われるすべてのアクセスを監査します。

- リレーショナル・データベース・ディレクトリー項目の追加 (ADDRDBDIRE) コマンド
- リレーショナル・データベース・ディレクトリー項目の変更 (CHGRDBDIRE) コマンド
- リレーショナル・データベース・ディレクトリー項目の表示 (DSPRDBDIRE) コマンド
- リレーショナル・データベース・ディレクトリー項目の除去 (RMVRDBDIRE) コマンド
- リレーショナル・データベース・ディレクトリー項目の処理 (WRKRDBDIRE) コマンド

リレーショナル・データベース・ディレクトリーは、ディレクトリー項目コマンドを使用せずに直接読み取ることのできるデータベース・ファイル (QSYS/QADBXRDBD) です。

V5R2 より前のバージョンでは、*PUBLIC に付与された操作権限を使用して、ライブラリー QSYS 内にリレーショナル・データベース (RDB) ディレクトリー・ファイル QADBXRDBD が作成されました。 V5R2 以降では、この点が変わっています。したがって、このファイルを使用して RDB ディレクトリーにアクセスする既存のプログラムは、正しく実行されなくなります。 *ALLOBJ 特殊権限がない場合に限り、QADBXRDBD に上書きして作成された QADBXRMTNM という名前の論理ファイルにアクセスする必要があります。このファイルへの直接アクセスを監査するには、 オブジェクト監査の変更 (CHGOBJAUD) コマンドによって監査をオンに設定します。

第 7 章 分散リレーショナル・データベースのデータの可用性および保護

分散リレーショナル・データベース環境では、データ可用性の観点から、ネットワーク内の個々のサーバーでのデータの保護だけでなく、ユーザーがネットワーク内のデータに確実にアクセスできるようにすることが必要です。

iSeries server では、分散リレーショナル・データベース・ネットワーク・サーバー上にあるデータを確実に使用できる状態にしておくために、以下の一連の機能が用意されています。

- 保管/復元
- ジャーナル管理およびアクセス・パス・ジャーナル処理
- コミットメント制御
- 補助記憶域プール
- チェックサム保護
- ミラー保護および無停電電源装置

通常、各サーバーのシステム・オペレーターは、そのサーバーにあるデータのバックアップと回復に責任を持ちますが (『 分散リレーショナル・データベースの回復サポート』を参照)、ネットワーク内のデータを最適な方法で利用できるようにするための方針を計画する場合には、 データ冗長性の側面に加えて、 ネットワーク冗長性の側面についても考慮する必要があります。特定のデータの重要性が企業にとって大きければ大きいほど、そのデータにアクセスする方法を多く備えていなければなりません。

分散リレーショナル・データベースの回復サポート

コンピューター・サーバーに起こり得る障害には、サーバー障害 (サーバー全体が作動しない時)、火災や洪水などの災害によるサイトの消失、またはオブジェクトの損傷ないしは消失があります。分散リレーショナル・データベースの場合は、ネットワークの中の 1 つのサーバーに障害が生じると、ネットワーク全体のユーザーが、そのサーバー上のリレーショナル・データベースにアクセスできなくなります。 そのリレーショナル・データベースが他のロケーションでの日常の業務活動にとって不可欠である場合には、 1 つのサーバーが回復するまでの間、ネットワーク全体にわたって企業運営が混乱することになりかねません。データの保護および障害後の回復に関する計画が、分散リレーショナル・データベースでは特に重要であることは明白です。

分散リレーショナル・データベースの各サーバーは、それぞれ自分のところにあるデータについて、バックアップおよび回復の責任を負います。 また、ネットワークの中の各サーバーは、それぞれがサーバー異常終了後の回復手順も処理します。 ただし、オペレーターが経験不足であったり、オペレーターがまったくいないサーバーの場合には、分散リレーショナル・データベース管理担当者がディスプレイ装置パススルーを使用して、バックアップおよび回復の手順を行うことができます。

最も一般的に発生する消失のタイプは、オブジェクトまたはオブジェクトのグループの消失です。オブジェクトの消失または損傷は、電源障害、ハードウェア障害、システム・プログラム・エラー、アプリケーション・プログラム・エラー、またはオペレーター・エラーなど、複数の要因が原因となって発生します。iSeries server には、サーバー・プログラム、アプリケーション・プログラム、およびデータの永久消失を

防ぐための方法がいくつも用意されています。障害のタイプおよび選択する保護レベルによって異なりますが、プログラムおよびデータのほとんどは保護することができ、回復時間は大幅に縮小することができます。

保護の方法には、次のようなものがあります。

- オブジェクトが保管される場所を制御する補助記憶域プール、補助記憶域プールのチェックサム保護、およびディスク関連ハードウェア構成要素のミラー保護など、 分散リレーショナル・データベースのディスク障害後のデータ回復
- リレーショナル・データベース変更やデータのジャーナル処理索引作成の補助レコードに関する、 分散リレーショナル・データベースのジャーナル管理
- リレーショナル・データベース・トランザクションを一貫した方法で適用または除去できるようにするための、コミットメント制御によるトランザクション回復
- テーブル、コレクション、パッケージ、およびリレーショナル・データベース・ディレクトリーといった、構造化照会言語 (SQL) オブジェクトが確実に保管および復元できるようにするための、 分散リレーショナル・データベースの保管および復元処理
- **補助記憶域へのデータの書き出し**

  ファイルの作成コマンドで強制書き出し率 (FRCRATIO) パラメーターを使用すると、データを補助記憶域に強制的に書き出すことができます。強制書き出し率を 1 にすると、該当のテーブルについて、すべての追加、更新、および削除要求が補助記憶域に即時に書き出されることになります。しかし、このオプションを選択すると、サーバー・パフォーマンスが低下することがあります。 したがって、データベースを保護するための第 1 の方法としては、テーブルの保管およびテーブルのジャーナル処理を考慮すべきです。
- **物理的な保護**

  電力供給が突然停止した場合にも、システムが確実に保護されるようにしておくことが、**アプリケーション・サーバー (AS)** を**アプリケーション・リクエスター (AR)** にとって確実に使用可能にしておくための重要な部分となります。無停電電源装置は、別途購入することができ、電源障害、停電、または電圧降下によって電力供給が停止した場合に、電力供給が回復するまでの間、サーバーに電源を提供することによって、サーバーを電力供給停止から保護するためのものです。 通常、無停電電源装置は、すべてのワークステーションを対象に電力を供給するものではありません。 iSeries server の場合には、無停電電源装置を使用すると、サーバーでは次のことを行うことができます。
  - 短時間の停電または一時的な電圧降下が生じている間、操作を続行する。
  - ファイルをクローズし、オブジェクトの保全性を維持することによって、正常に操作を終了する。

分散リレーショナル・データベースのディスク障害後のデータの回復

ディスク障害が起こった場合には、オブジェクトが障害の直前にすべてテープまたはディスクに保管されていない限り、最新の入力データについては回復は不可能です。前に保管されたオブジェクトが復元された後、サーバーは操作可能ですが、データベースは現行状態ではありません。

補助記憶域プール (ASP)、チェックサム保護、およびミラー保護は、 OS/400 ディスク回復機能であり、ディスク関連の障害の後に最新入力データを回復するための方法を提供するものです。これらの機能は、追加のサーバー・リソースを使用しますが、分散リレーショナル・データベースのサーバーに高水準の保護を提供します。 サーバーによっては、アプリケーション・サーバーとして他のサーバーよりも重要な機能を果たしているものがあるので、分散リレーショナル・データベース管理担当者は、どのようにすれば、上記のディスク・データ保護方式がネットワーク内の個々のシステムで最適使用できるかについて検討しなければなりません。

補助記憶域プール (ASP)、チェックサム保護、およびミラー保護については、 iSeries Information Center の『バックアップおよび回復』のトピックを参照してください。

補助記憶域プール

ASP は、同一の記憶域に割り当てられた 1 つまたは複数の物理磁気ディスク装置です。 ASP を使用すると、指定した物理磁気ディスク上に、特定のタイプのオブジェクトを分離することができます。

サーバー ASP は、サーバー・プログラム、およびサーバー・プログラムによる処理の結果として作成される一時オブジェクトを分離します。 ユーザー ASP は、ライブラリー、SQL オブジェクト、ジャーナル、ジャーナル・レシーバー、アプリケーション、およびデータなどのオブジェクトの分離に使用することができます。 iSeries server は、最大 32 の基本ユーザー ASP と、223 の独立ユーザー ASP をサポートしています。ライブラリーやオブジェクトは、ユーザー ASP の中に分離すると、他の ASP の中のディスク障害から保護されるので、回復時間が短縮されます。

オブジェクトを ASP に入れると、回復時間の短縮およびオブジェクトの分離に加えて、パフォーマンスの向上を図ることができます。ジャーナル・レシーバーはユーザー ASP の中に分離した場合には、その ASP に関連するディスクはそのジャーナル・レシーバー専用になります。データベース・ファイルを対象とする読み取りおよび書き出し操作を多く必要とする環境では、これによってその ASP の中のディスクに対するアームの競合が減り、ジャーナル処理のパフォーマンスを向上させることができます。

分散リレーショナル・データベースのチェックサム保護

チェックサム保護は、ASP の中のディスク上のデータを消失から保護します。チェックサム・ソフトウェアが、ASP データのエンコードされたコピーを、その ASP 内の特別のチェックサム・データ域に維持します。チェックサム保護 ASP の中の永続オブジェクトに加えられた変更は、すべて自動的にチェックサム・セットのチェックサム・データの中に維持されます。チェックサム・セットの中の 1 つの磁気ディスク装置のデータが消失した場合には、サーバーは、チェックサムおよびセットの中に残っている機能装置上のデータを使用して、消失した装置の内容を再構成します。 このようにして、装置のいずれかに障害が生じた場合にも、その内容を回復することができます。こうして再構成されたデータは、障害の時点でディスク上にあった最新情報を反映しています。チェックサム保護は、サーバー・パフォーマンスに著しい影響を与える可能性があります。 分散リレーショナル・データベースでは、これが問題となることがあります。

分散リレーショナル・データベースのミラー保護

ミラー保護では、ディスク制御装置、ディスク入出力装置、またはバスなど、異なるディスク関連ハードウェア構成要素を 2 重にすることによって、サーバーの可用性を増大させます。サーバーは障害後もそのまま使用可能であり、障害のあったハードウェア構成要素の保守は適当な時期に予定することができます。

ミラー保護のレベルが異なれば、それに応じてサーバー可用性のレベルも異なります。たとえば、サーバー上の磁気ディスク装置だけがミラー保護された場合には、すべての磁気ディスク装置が磁気ディスク装置レベルの保護となり、サーバーは 1 台の磁気ディスク装置の障害から保護されます。しかしこのような状態では、コントローラー、入出力装置、またはバスの障害が起こった場合には、障害部品の修理または交換が行われるまで、サーバーは運転できません。サーバー上のミラー保護された装置は、すべて同一の磁気装置レベルの保護でなければならず、また同一の ASP に存在していなければなりません。 1 つの ASP の中の装置は、ミラー保護が開始されると、サーバーによって自動的に対にされます。

分散リレーショナル・データベースのジャーナル管理

ジャーナル管理を、リレーショナル・データベースおよび索引のバックアップおよび回復の戦略の一つとして使用することができます。

ジャーナル処理に関する詳細は、 iSeries Information Center の『ジャーナル管理』のトピックを参照してください。

iSeries ジャーナル・サポートでは、監査証跡と正方向および逆方向回復が用意されています。正方向回復は、テーブルの古いバージョンを取り出し、ジャーナルにログ記録された変更をテーブルに適用するのに使用することができます。逆方向回復は、ジャーナルにログ記録された変更をテーブルから除去するのに使用することができます。

コレクションが作成されると、ジャーナルおよびジャーナル・レシーバーと呼ばれるオブジェクトがそのコレクションの中に作成されます。ジャーナル・レシーバーは、テーブルとは別の ASP にあった方が、パフォーマンスが向上します。しかし、コレクションをユーザー ASP 上に置いてしまうと、テーブルとジャーナル、そしてジャーナル・レシーバーは、すべて同じユーザー ASP に置かれることになります。これでは、高いパフォーマンスは得られません。そこで、別の ASP (このジャーナルのジャーナル・レシーバーのためだけに使用される) に新しいジャーナル・レシーバーを作成し、 ジャーナル変更 (CHGJRN) コマンドを使用してそれを付加すると、次からは、サーバーによって生成されるジャーナル・レシーバーが、すべて別のユーザー ASP に置かれるようになり、それによってパフォーマンスの向上を得ることができます。

テーブルは、作成されると、コレクションの中に作成されているジャーナル SQL に自動的にジャーナル処理されます。この後は、自分の責任でジャーナル機能を使用して、ジャーナル、ジャーナル・レシーバー、およびジャーナルへのテーブルのジャーナル処理を管理しなければなりません。たとえば、テーブルをコレクションの中へ移動する場合には、ジャーナル処理状況に対する自動変更は行われません。テーブルが復元される場合には、通常のジャーナル規則が適用されます。つまり、テーブルが保管される時にジャーナル処理されている場合には、テーブルはそのサーバー上に復元した時に同一のジャーナルにジャーナル処理されます。 テーブルが保管される時にジャーナル処理されていない場合には、テーブルは復元した時にジャーナル処理されません。ジャーナル機能を使用して、任意のテーブルに対するジャーナル処理を停止することができますが、それを行った場合には、SQL 操作はコミットメント制御下で実行できなくなります。 COMMIT(*NONE) を指定してある場合には、SQL 操作はまだ実行できますが、この場合には、ジャーナル処理およびコミットメント制御の場合と同じレベルの保全性は提供されません。

ジャーナル処理を活動状態にして、データベースに変更を加えると、その変更は、データベースに加えられる前に、ジャーナル・レシーバーの中でジャーナル処理されます。ジャーナル・レシーバーには、常に最新のデータベース情報が入っています。すべての活動は、変更がどのように行われたかに関係なく、データベース・テーブルについてジャーナル処理されます。

ジャーナル・レシーバー項目は、特定の行 (追加、変更、または削除された行)、あるいはテーブル (オープンされたテーブルまたは保管されたメンバーなど) についての活動を記録します。各項目には、活動の源、ユーザー、ジョブ、プログラム、時刻、および日付を識別する追加の制御情報が含まれています。

サーバーは、テーブルの移動およびテーブルの名前変更をも含めて、ファイル・レベルの変更のジャーナル処理を行います。 サーバーはまた、物理ファイル・メンバーの初期設定など、メンバー・レベルの変更、および初期プログラム・ロード (IPL) など、サーバー・レベルの変更のジャーナル処理も行います。 ジャーナル・レシーバーに項目を追加して、重要な事象 (ジョブ・ステップが後で再開できるように、ジョブおよびサーバーに関する情報をジャーナル処理できるチェックポイントなど) の識別、またはアプリケーションの回復の援助を行うことができます。

単一行に影響する変更の場合は、制御情報に続いて、行イメージが常に含められます。変更が行われた後の行のイメージが常に含められます。任意で、変更が加えられる前の行イメージも含めることができます。物理ファイル・ジャーナル開始 (STRJRNPF) コマンドで IMAGES パラメーターを指定し、ジャーナル処理を変更前と変更後の両方の行イメージにするか、変更後の行イメージだけにするかを制御します。

ジャーナル処理データベース・ファイルは、すべて、サーバーの開始時 (IPL 時) か、独立 ASP がオンに変更されるときに自動的にジャーナルと同期化されます。サーバーが異常終了した場合や、独立 ASP が異常な状態でオフにされた場合は、データベースに対する変更の一部がジャーナルに入っていても、まだデータベースそのものに変更が反映されていない場合があります。このような場合、サーバーは自動的にジャーナルからデータベースを更新して、テーブルを最新の状態にします。

ジャーナル処理を使用すると、データベース・テーブルの保管が一層容易で迅速なものとなります。たとえば、毎日テーブル全体を保管する代わりに、そのテーブルに対する変更が入っているジャーナル・レシーバーを保管するだけで済みます。さらにテーブル全体の保管を定期的に行うこともできます。この方法を用いると、日常の保管操作の実行に要する時間を削減できます。

ジャーナル表示 (DSPJRN) コマンドを使用すると、ジャーナル・レシーバー項目をデータベース・ファイルに変換することができます。 このようなファイルは、活動報告書、監査証跡、セキュリティー、およびプログラム・デバッグに使用できます。

索引の回復

索引とは、テーブルから行を読み取る順序を記述するものです。索引がジャーナルに記録されていれば、サーバーは、索引を回復できるため、システムが異常終了した後の IPL 時や、異常な状態でオフにされた ASP をオンに戻すときなどに、多大の時間をかけて索引を再作成しなくて済みます。

テーブルをジャーナル処理すると、テーブルの中の行に対する変更のイメージがジャーナルに書き込まれます。サーバーが異常終了したときは、これらの行イメージを使用してテーブルが回復されます。ただし、異常終了の後で、サーバーは、テーブルについて作成された索引がテーブルの中のデータと同期していないことを発見する場合があります。 アクセス・パスとそのデータが同期していない場合には、サーバーでは索引を再作成して、この 2 つが必ず同期して使用可能であるようにしなければなりません。

索引がジャーナル処理されると、サーバーでは索引のイメージをジャーナルの中に記録して、索引とデータの間に認識された同期点を用意します。 その情報がジャーナルの中に入っていることによって、サーバーでは、データと索引の両方を回復し、両方の同期を確保することができます。 このような場合には、索引の再作成に要する長い時間を回避することができます。

iSeries server には、索引の回復を援助する複数の機能が用意されています。 サーバー上のすべての索引には、その索引のメインテナンスが行われる時期を指定するメインテナンス・オプションがあります。 SQL 索引は、*IMMED メインテナンスという属性を指定して作成されます。

電源障害またはサーバーの異常障害が生じた場合には、変更中の索引は、データとの一致を確保するために、再作成が必要になることがあります。 サーバー上のすべての索引には、必要な場合に、その索引を再作成する時期を指定する回復オプションがあります。 UNIQUE という属性が指定されている SQL 索引は、すべて *IPL という回復属性を指定して作成され、これによって、これらの索引は OS/400 ライセンス・プログラムが開始される間に再作成されることを意味します。一方、こうしたオプションが指定されていない SQL 索引は、すべて *AFTIPL 回復属性で作成されることになります。この属性を持つ索引は、オペレーティング・システムが開始された後、あるいは独立 ASP がオンにされた後に再作成されます。 IPL 時や、独立 ASP をオンにするときには、画面で、再作成の必要な索引とそれらの回復オプションを確認できます。これらの回復オプションは、指定変更することが可能です。

SQL 索引は自動的にはジャーナル処理されません。 アクセス・パス・ジャーナル開始 (STRJRNAP) コマンドを使用して、 SQL 操作で作成された索引をジャーナル処理することができます。 サーバー保管および復元機能を使用すれば、オブジェクト保管 (SAVOBJ) またはライブラリー保管 (SAVLIB) コマンドで ACCPTH (*YES) を使用することによって、テーブルを保管する時に、索引を保管することができます。

テーブルを復元しなければならない場合でも、索引を再作成する必要はありません。前に保管および復元されていない索引は、いずれもデータベースによって自動的かつ非同期的に再作成されます。

索引のジャーナル処理の前に、その索引に対応するテーブルについて、ジャーナル処理を開始しなければなりません。さらに、索引およびそれに対応するテーブルには、同一のジャーナルを使用しなければなりません。

索引ジャーナル処理は、追加の出力操作を最小限に抑制するように設計されています。たとえば、サーバーは、変更された行および変更された索引のジャーナル・データを、同一の出力操作で書き出します。 ただし、索引のジャーナル処理を開始するにあたっては、ユーザー ASP の中のジャーナル・レシーバーを分離することを特に考慮しなければなりません。ジャーナル・レシーバーをそれ自体の ASP に入れれば、最高のジャーナル管理のパフォーマンスが得られると同時に、ジャーナル・レシーバーをディスク障害から保護する上でも役立ちます。

索引再作成時間を短縮するためのテーブルの設計

テーブルの設計しだいでは、索引回復時間の短縮に役立つ場合があります。たとえば、大規模のマスター・テーブルをヒストリー・テーブルとトランザクション・テーブルに分割することができます。そうすれば、トランザクション・テーブルは新しいデータを追加するのに使用し、ヒストリー・テーブルは照会専用として使用することができます。毎日、トランザクション・データをヒストリー・テーブルに組み合わせてから、翌日のデータに備えてトランザクション・ファイルを消去することができます。このような設計を採用すれば、その日にサーバーが異常終了した場合でも、比較的小規模のトランザクション・テーブルの索引を再作成するだけで済むために、索引を再作成する時間を短縮することができます。 なお、大規模なヒストリー・テーブルの索引については、1 日の大半が読み取り専用となるため、データとの同期がずれることはほとんどなく、したがって再作成する必要はなさそうです。

テーブル設計によって索引の再作成時間の短縮を図る場合と、アクセス・パス・ジャーナル処理など、サーバー提供機能を使用する場合でどちらが有利であるかを考慮してください。 上記のテーブル設計では、より複雑なアプリケーションの設計が必要となる場合があります。実際の状況を考慮した後、より複雑なアプリケーションを設計するよりも、アクセス・パスのジャーナル処理のようなサーバーが提供する機能を使うこともできます。

システム管理アクセス・パス保護 (SMAPP)

システム管理アクセス・パス保護 (SMAPP) は、アクセス・パスの自動保護を行います。 SMAPP サポートを使うと、アクセス・パス・ジャーナルの開始 (STRJRNAP) コマンド などのジャーナル処理コマンドを使用しなくても、アクセス・パスのジャーナル処理の利点が得られます。 SMAPP サポートでは、IPL 時や、独立 ASP をオンにするときにアクセス・パスを再作成するのではなく、サーバーが異常終了した後にアクセス・パスを回復します。

SMAPP サポートは、出荷時にオンになっています。

サーバーは、ユーザーが定めるターゲットのアクセス・パス回復回数に基づいて、またはサーバーで指定されたデフォルトの時間により、どのアクセス・パスを保護するかを判別します。 ターゲットのアクセス・パス回復回数は、サーバー全体の値として指定するか、または ASP を基礎として指定できます。 ユーザー定義のジャーナルにジャーナルがとられるアクセス・パスは、SMAPP 保護の対象とはなりません。それらのアクセス・パスはすでに保護されているからです。 SMAPP についての詳細は、 iSeries Information Center の『システム管理アクセス・パス保護 (SMAPP)』のトピックを参照してください。

コミットメント制御によるトランザクションの回復

コミットメント制御は、iSeries server 上のジャーナル管理機能を拡張したものです。 サーバーは、リレーショナル・データベース変更のグループを単一の作業単位 (トランザクション) として識別し処理することができます。

SQL COMMIT ステートメントは、操作のグループが完了することを保証します。 SQL ROLLBACK ステートメントは、操作のグループが取り消されることを保証します。コミットまたはロールバックすることができない SQL ステートメントは、次に挙げるものだけです。

- DROP COLLECTION
- GRANT または REVOKE (ただし、指定されたオブジェクトに権限保有者が存在している場合)

コミットメント制御の下では、トランザクション中に使用されているテーブルおよび行は、他のジョブからロックされます。したがって、そのトランザクションが完了するまで、他のジョブではそのデータを使用することができません。トランザクションが終了すると、プログラムは SQL COMMIT または ROLLBACK ステートメントを出して、行を解放します。コミット操作が実行される前に、サーバーまたはジョブが異常終了した場合には、最後にコミットまたはロールバック操作が行われた時以後にそのジョブに加えられたすべての変更がロールバックされます (元に戻される)。 影響を受けた行でまだロックされているものがあれば、そこでアンロックされます。ロック・レベルは、次のようになっています。

***NONE**
: コミットメント制御は使用されない。他のジョブのコミットされていない変更は、見ることができる。

***CHG** SQL ALTER、COMMENT ON、CREATE、DROP、GRANT、LABEL ON、および REVOKE ステートメントの中で参照されているオブジェクト、および更新、削除、挿入が行われた行は、その作業単位 (トランザクション) が完了するまで、ロックされる。他のジョブのコミットされていない変更は、見ることができる。

***CS** SQL ALTER、COMMENT ON、CREATE、DROP、GRANT、LABEL ON、および REVOKE ステートメントの中で参照されているオブジェクト、および更新、削除、挿入が行われた行は、その作業単位 (トランザクション) が完了するまで、ロックされる。選択されても更新されなかった行は、次の行が選択されるまで、ロックされる。他のジョブのコミットされていない変更は、見ることができない。

***ALL** SQL ALTER、COMMENT ON、CREATE、DROP、GRANT、LABEL ON、および REVOKE ステートメントの中で参照されているオブジェクト、および読み取り、更新、削除、挿入が行われた行は、その作業単位 (トランザクション) が完了するまで、ロックされる。他のジョブのコミットされていない変更は、見ることができない。

130 ページの表 6 には、上記のロック・レベルのそれぞれについて、レコード・ロック期間が示してあります。

プログラムのプリコンパイル時、または対話式 SQL の開始時に、 COMMIT (*CHG)、COMMIT (*CS)、または COMMIT (*ALL) を要求した場合には、SQL は、 コミットメント制御開始 (STRCMTCTL) コマンドを暗黙的に呼び出すことによって、コミットメント制御環境をセットアップします。 SQL がコミットメント制御を開始する時に指定される LCKLVL パラメーターは、 CRTSQL*xxx* コマンドの COMMIT パラメーターで指定されるロック・レベルです。 SQL がコミットメント制御を開始する時には、NFYOBJ(*NONE) が指定されます。別の NFYOBJ パラメーターを指定する場合には、 SQL を開始する前に (STRCMTCTL) コマンドを出してください。

注: コミットメント制御を用いて実行する時には、アプリケーション・プログラムの中でデータ処理言語ステートメントによって参照されているテーブルは、ジャーナル処理しなければなりません。そのようなテーブルは、プリコンパイル時にジャーナル処理する必要はありませんが、アプリケーションの実行時にはジャーナル処理しなければなりません。

リモート・リレーショナル・データベースが サーバー上のデータにアクセスし、コミットメント・レベル反復可能読み取り (*RR) を要求している場合には、照会がクローズされるまで、テーブルはロックされることになります。 カーソルが読み取り専用である場合には、テーブルは (*SHRNUP) ロックされます。カーソルが更新モードである場合には、テーブルは (*EXCLRD) ロックされます。

SQL コレクションの中に作成されたジャーナルは、通常、SQL テーブルに対するすべての変更のログを記録するのに使用されるジャーナルです。ただし、サーバー・ジャーナル機能を使用すれば、 SQL テーブルを別のジャーナルにジャーナル処理することができます。

コミットメント制御では、最高 131,072 個の行変更を 1 つの作業単位で処理することができます。COMMIT(*ALL) が指定されている場合には、読み取られた行もすべて 131,072 の限度内に含まれます。(1 つの作業単位内である 1 行が複数回、変更または読み取られた場合、 131,072 の限度に対しては 1 回としてカウントされます。) ロックが多数維持されていると、サーバー・パフォーマンスに悪影響を及ぼします。また、作業単位が完了するまで、他のユーザーはその作業単位でロックされている行にアクセスすることができません。 したがって、1 つの作業単位で処理される行の数を少ない数に抑える方が、効率的にはよいと言えます。コミットメント制御では、最高 512 のテーブルをコミットメント制御のもとでオープンにしておくか、あるいは 1 つの作業単位の中で変更を保留してクローズにしておくことができます。

COMMIT および ROLLBACK ステートメントの HOLD 値を使用すると、カーソルをオープンにしておき、OPEN を再度出さなくても別の作業単位を開始することができます。プログラムに解放されていない iSeries 以外の接続があって、呼び出しスタックにまだ SQL が入っている場合は、HOLD 値を使用できません。プログラムのプリコンパイル時に、 ALWBLK(*ALLREAD) および COMMIT(*CHG) と COMMIT(*CS) のいずれかが指定された場合には、すべての読み取り専用カーソルで列のブロック化が可能になり、 ROLLBACK HOLD ステートメントではカーソル位置をロールバックしません。

ロックされた行 (レコード) が SQL プリコンパイル・プログラム、または対話式 SQL セッションの実行で保留になっている場合には、サーバーの「コマンド入力」画面で COMMIT または ROLLBACK ステートメントを出すことができます。 それ以外の場合には、暗黙の ROLLBACK 操作がジョブの終了時に行われます。

WRKCMTDFN (コミットメント定義の処理) コマンドを使用すれば、コミットメント定義の状況を監視し、そのコミットメント制御にかかわっているロックと保持リソースの解除を行うことができます。 詳しくは 99 ページの『分散リレーショナル・データベースのコミットメント定義の処理』をご覧ください。

コミットメント制御についての詳細は、 iSeries Information Center の『トランザクションとコミットメント制御』のトピックを参照してください。

表 6. レコード・ロック期間

SQL ステートメント	COMMIT パラメーター	レコード・ロックの期間	ロック・タイプ
SELECT INTO	*NONE	ロックなし	
	*CHG	ロックなし	
	*CS	読み取りおよび解放時に行のロック	READ
	*ALL (注 2 参照)	読み取りから ROLLBACK または COMMIT まで	READ

表 6. レコード・ロック期間 (続き)

SQL ステートメント	COMMIT パラメーター	レコード・ロックの期間	ロック・タイプ
FETCH (読み取り専用カーソル)	*NONE	ロックなし	
	*CHG	ロックなし	
	*CS	読み取りから次の FETCH まで	READ
	*ALL (注 2 参照)	読み取りから ROLLBACK または COMMIT まで	READ
FETCH (更新または削除可能カーソル) 注 1 参照	*NONE	レコードが更新または削除されなかった場合、読み取りから次の FETCH まで レコードが更新または削除された場合、読み取りから UPDATE または DELETE まで	UPDATE
	*CHG	レコードが更新または削除されなかった場合、読み取りから次の FETCH まで レコードが更新または削除された場合、読み取りから UPDATE または DELETE まで	UPDATE
	*CS	レコードが更新または削除されなかった場合、読み取りから次の FETCH まで レコードが更新または削除された場合、読み取りから UPDATE または DELETE まで	UPDATE
	*ALL	読み取りから ROLLBACK または COMMIT まで	UPDATE[3]
INSERT (ターゲット・テーブル)	*NONE	ロックなし	
	*CHG	挿入から ROLLBACK または COMMIT まで	UPDATE
	*CS	挿入から ROLLBACK または COMMIT まで	UPDATE
	*ALL	挿入から ROLLBACK または COMMIT まで	UPDATE[4]
INSERT (副選択の中のテーブル)	*NONE	ロックなし	
	*CHG	ロックなし	
	*CS	読み取りの間、各レコードをロック	READ
	*ALL	読み取りから ROLLBACK または COMMIT まで	READ
UPDATE (非カーソル)	*NONE	更新の間、各レコードをロック	UPDATE
	*CHG	読み取りから ROLLBACK または COMMIT まで	UPDATE
	*CS	読み取りから ROLLBACK または COMMIT まで	UPDATE
	*ALL	読み取りから ROLLBACK または COMMIT まで	UPDATE
DELETE (非カーソル)	*NONE	削除の間、各レコードをロック	UPDATE
	*CHG	読み取りから ROLLBACK または COMMIT まで	UPDATE
	*CS	読み取りから ROLLBACK または COMMIT まで	UPDATE
	*ALL	読み取りから ROLLBACK または COMMIT まで	UPDATE
UPDATE (カーソル付き)	*NONE	レコード更新時にロック解放	UPDATE
	*CHG	読み取りから ROLLBACK または COMMIT まで	UPDATE
	*CS	読み取りから ROLLBACK または COMMIT まで	UPDATE
	*ALL	読み取りから ROLLBACK または COMMIT まで	UPDATE
DELETE (カーソル付き)	*NONE	レコード削除時にロック解放	UPDATE
	*CHG	読み取りから ROLLBACK または COMMIT まで	UPDATE
	*CS	読み取りから ROLLBACK または COMMIT まで	UPDATE
	*ALL	読み取りから ROLLBACK または COMMIT まで	UPDATE
副照会 (更新または削除可能カーソルまたは UPDATE または DELETE 非カーソル)	*NONE	読み取りから次の FETCH まで	READ
	*CHG	読み取りから次の FETCH まで	READ
	*CS	読み取りから次の FETCH まで	READ
	*ALL (注 2 参照)	読み取りから ROLLBACK または COMMIT まで	READ

表 6. レコード・ロック期間 (続き)

SQL ステートメント	COMMIT パラメーター	レコード・ロックの期間	ロック・タイプ
副照会 (読み取り専用カーソルまたは SELECT INTO)	*NONE *CHG *CS *ALL	ロックなし ロックなし 読み取りの間、各レコードをロック 読み取りから ROLLBACK または COMMIT まで	 READ READ

注:

1. 結果のテーブルが読み取り専用でない (iSeries Information Center の『SQL 解説書』のトピックにある、DECLARE CURSOR の説明を参照) 場合で、しかも次のいずれか 1 つに該当する場合には、カーソルは UPDATE または DELETE 機能についてオープンされます。
   - カーソルが FOR UPDATE 文節で定義されている。
   - カーソルが FOR UPDATE、FOR FETCH ONLY、または ORDER BY 文節なしで定義され、プログラムに少なくとも次の 1 つが入っている。
     - 同じカーソル名を参照するカーソル UPDATE
     - 同じカーソル名を参照するカーソル DELETE
     - CRTSQL*xxx* コマンドで ALWBLK(*READ) または ALWBLK(*NONE) を指定した EXECUTE または EXECUTE IMMEDIATE ステートメント
2. テーブルまたはビューは、COMMIT(*ALL) を充足するために、排他ロックすることができます。グループ化または合併が組み込まれている副選択が処理される場合、または照会の処理が一時的な結果の使用を必要とする場合には、コミットされていない変更の表示から保護するために、排他ロックが獲得されます。
3. 行が更新または削除されない場合には、ロックは *READ になります。
4. ターゲット・テーブルの行に対しては UPDATE ロックで、副選択テーブルの行に対しては READ ロックです。
5. テーブルまたはビューは、反復可能読み取りを充足するために、排他ロックすることができます。行のロックは、反復可能読み取りのもとでも行われます。獲得されるロックおよびその期間は、*ALL と同じです。

分散リレーショナル・データベースの保管および復元処理

データおよびプログラムの保管および復元を使用すれば、プログラム障害またはサーバー障害からの回復、サーバー間での情報の交換、またはオフラインでのオブジェクトまたはデータの記憶が可能です。 堅実なバックアップ方針が分散リレーショナル・データベース・ネットワークの各サーバーで確立されていれば、問題が発生した場合にも、確実にサーバーの回復を図って、ネットワーク・ユーザーのために迅速に使用可能にすることができます。

サーバーをテープなど、外部メディアに保管すれば、サーバー・プログラムおよびデータを火災や洪水などの災害から保護することができます。 しかし、情報は、保管ファイルと呼ばれるディスク・ファイルに保管することもできます。保管ファイルは、ディスクに常駐しているファイルで、データが入出力操作で使用されたり、通信回線によって別の iSeries server へ転送される時まで、データを保管するために使用されるものです。 保管ファイルを使用すると、オペレーターがディスケットやテープをロードする必要がないので、オペレーターがいなくても保管操作が可能になります。分散リレーショナル・データベースでは、保管ファイルは、保護方法と 1 つとして別のサーバーに送ることができます。

情報の復元時には、情報はディスケット、テープ、または保管ファイルから補助記憶域へ書き込まれるので、サーバー・ユーザーはそこに書き込まれた情報にアクセスすることができます。

iSeries server には、データベース・テーブルと SQL オブジェクトを保管および復元するためのコマンドの完全なセットが備わっています。

- ライブラリー保管 (SAVLIB) コマンドでは、1 つ以上のコレクションを保管します。
- オブジェクト保管 (SAVOBJ) コマンドでは、 SQL テーブル、ビュー、および索引など、1 つまたは複数のオブジェクトを保管します。
- 変更されたオブジェクト保管 (SAVCHGOBJ) コマンドでは、コレクションが最後に保管された時点、または指定された日付のいずれか以後に変更されたオブジェクトを保管します。
- 保管ファイル・データの保管 (SAVSAVFDTA) コマンドでは、保管ファイルの内容を保管します。
- システム保管 (SAVSYS) コマンドでは、オペレーティング・システム、セキュリティー情報、入出力装置構成、およびサーバー値を保管します。
- ライブラリー復元 (RSTLIB) コマンドでは、コレクションを復元します。
- オブジェクト復元 (RSTOBJ) コマンドでは、SQL テーブル、ビュー、および索引など、 1 つまたは複数のオブジェクトを復元します。
- ユーザー・プロファイル復元 (RSTUSRPRF)、 権限復元 (RSTAUT)、および構成復元 (RSTCFG) コマンドでは、 システム保管 (SAVSYS) コマンドによって保管されたユーザー・プロファイル、権限、および構成を復元します。

これらの機能およびコマンドについての詳細は、 iSeries Information Center の『バックアップおよび回復』のトピックを参照してください。

分散リレーショナル・データベース内の索引の保管と復元

SQL 索引については、復元の方が再作成より高速で行うことができます。所要時間は多くの要因に左右されますが、データベース索引の再作成には、10,000 行当たり約 1 分を要します。

索引を復元した後では、最新のジャーナル変更を適用することによって、テーブルを更新する必要がある場合があります (ジャーナル処理が活動状態にあるかどうかによります)。この余分な回復時間を考慮に入れても、索引については、再作成よりも復元の方が速いかもしれません。

サーバーは、索引の保全性を確保します。 サーバーが索引について使用不能であると判別した場合には、サーバーはその回復を試みます。 索引をいつ回復するかは、制御することができます。サーバーが異常終了した場合には、次の IPL 時に、サーバーは、索引またはビューの回復を必要とするテーブルを自動的にリストします。 索引を再作成するか、または次のいずれかの時点に索引の回復を試みるかを決めることができます。

- IPL 中
- IPL の後
- テーブルを初めて使用する時

アクセス・パスの保管と復元に関する詳細は、 iSeries Information Center の『バックアップおよび回復』のトピックを参照してください。

分散リレーショナル・データベース環境内のセキュリティー情報の保管と復元

ユーザー・プロファイルを更新したり、分散リレーショナル・データベース・ネットワーク内のユーザーの権限を更新するなど、サーバー・セキュリティー環境に頻繁に変更を加える場合は、完全なシステム保管 (SAVSYS) コマンドを実行したり、専用サーバーを使用して実行に時間のかかる処理を行わなくても、セキュリティー情報をメディアや保管ファイルに保管できます。 セキュリティー・データの保管 (SAVSECDTA) コマンドを使用すれば、専用サーバーを使用しなくても、比較的短時間でセキュリティー・データを保管できます。 SAVSECDTA コマンドを使用して保管したデータは、 ユーザー・プロファイル復元 (RSTUSRPRF) コマンドや権限復元 (RSTAUT) コマンドで復元できます。

セキュリティー・データの保管 (SAVSECDTA) コマンドやユーザー・プロファイル復元 (RSTUSRPRF) コマンドで保管および復元できるセキュリティー情報に含まれるのは、 DRDA TCP/IP サポートがリモート・サーバー・ユーザー ID およびパスワード情報を保管および検索するために使用するサーバー権限項目です。

分散リレーショナル・データベース環境内の SQL パッケージの保管と復元

リモート・サーバー上のリレーショナル・データベースを参照するアプリケーション・プログラムがプリコンパイルまたはバインドされると、アプリケーションの中の SQL ステートメントを処理するのに必要な制御構造を入れるために、 SQL パッケージが**アプリケーション・サーバー (AS)** 上に作成されます。

SQL パッケージは、iSeries オブジェクトなので、 オブジェクト保管 (SAVOBJ) コマンドを使用してメディアまたは保管ファイルに保管し、 オブジェクト復元 (RSTOBJ) コマンドを使用して復元することができます。

SQL パッケージは、保管された元のコレクションと同じ名前を持つコレクションに復元されなければならず、名前を変更することはできません。

リレーショナル・データベース・ディレクトリーの保管と復元

リレーショナル・データベース・ディレクトリーは iSeries オブジェクトではありません。リレーショナル・データベース・ディレクトリーは、IPL 時にサーバーによってオープンされたファイルから成っています。したがって、オブジェクト保管 (SAVOBJ) コマンドでこれらのファイルを直接保管することはできません。リレーショナル・データベース・ディレクトリー・データから出力ファイルを作成することによって、リレーショナル・データベース・ディレクトリーを保管することができます。そうすれば、ディレクトリーが損傷した場合には、この出力ファイルを使用して、ディレクトリーに項目を追加することができます。

項目が追加され、リレーショナル・データベース・ディレクトリーを保管するときは、 リレーショナル・データベース・ディレクトリー項目の表示 (DSPRDBDIRE) コマンドで OUTFILE パラメーターを指定して、コマンドの結果を出力ファイルに送ります。この出力ファイルは、テープ、ディスケット、または保管ファイルに保管し、サーバーに復元することができます。 リレーショナル・データベース・ディレクトリーが損傷し、サーバーが回復を必要とする場合には、制御言語 (CL) プログラムを使用して、リレーショナル・データベース項目データが入っている出力ファイルを復元することができます。 CL プログラムが復元された出力ファイルからデータを読み取り、新しいリレーショナル・データベース・ディレクトリーに項目を追加する CL コマンドを作成します。

たとえば、Spiffy 社の MP000 サーバーのリレーショナル・データベース・ディレクトリーは、次のようにして RDBDIRM という名前の出力ファイルに送られます。

```
DSPRDBDIRE OUTPUT(*OUTFILE) OUTFILE(RDBDIRM)
```

次に挙げる CL プログラム例では、出力ファイル RDBDIRM の内容を読み取り、 RDB ディレクトリー項目の追加 (ADDRDBDIRE) コマンドを使用して、 RDB ディレクトリー項目を再作成しています。なお、古いディレクトリー項目は、新しいディレクトリー項目が作成される前に除去されます。

注: xv ページの『コードの特記事項情報』に重要なリーガル情報がありますので、お読みください。

```
/*****************************************************************/
/* -  Restore RDB Entries from output file created with:         - */
/* -   DSPRDBDIRE OUTPUT(*OUTFILE) OUTFILE(RDBDIRM)              - */
/* -    FROM A V4R2 OR LATER LEVEL OF OS/400                   - */
/*****************************************************************/
PGM  PARM(&ACTIVATE)
DCL    VAR(&ACTIVATE) TYPE(*CHAR) LEN(7)
```

```
/* Declare Entry Types Variables to Compare with &RWTYPE          */
 DCL &LOCAL    *CHAR 1
 DCL &SNA      *CHAR 1
 DCL &IP       *CHAR 1
 DCL &ARD      *CHAR 1
 DCL &ARDSNA   *CHAR 1
 DCL &ARDIP    *CHAR 1
 DCL &RWTYPE   *CHAR 1
 DCL &RWRDB    *CHAR 18
 DCL &RWRLOC   *CHAR 8
 DCL &RWTEXT   *CHAR 50
 DCL &RWDEV    *CHAR 10
 DCL &RWLLOC   *CHAR 8
 DCL &RWNTID   *CHAR 8
 DCL &RWMODE   *CHAR 8
 DCL &RWTPN    *CHAR 8
 DCL &RWSLOC   *CHAR 254
 DCL &RWPORT   *CHAR 14
 DCL &RWDPGM   *CHAR 10
 DCL &RWDLIB   *CHAR 10

 DCLF  FILE(RDBSAV/RDBDIRM) /* SEE PROLOG CONCERNING THIS        */
IF COND(&ACTIVATE = SAVE) THEN(GOTO CMBLBL(SAVE))
IF COND(&ACTIVATE = RESTORE) THEN(GOTO CMDLBL(RESTORE))
 SAVE:
 CRTLIB RDBSAV
 DSPRDBDIRE OUTPUT(*OUTFILE) OUTFILE(RDBSAV/RDBDIRM)
 GOTO CMDLBL(END)

 RESTORE:
/* Initialize Entry Type Variables to Assigned Values                */
 CHGVAR &LOCAL   '0'               /* Local RDB (one per system) */
 CHGVAR &SNA     '1'               /* APPC entry (no ARD pgm)    */
 CHGVAR &IP      '2'               /* TCP/IP entry (no ARD pgm)  */
 CHGVAR &ARD     '3'               /* ARD pgm w/o comm parms     */
 CHGVAR &ARDSNA  '4'               /* ARD pgm with APPC parms    */
 CHGVAR &ARDIP   '5'               /* ARD pgm with TCP/IP parms  */

 RMVRDBDIRE    RDB(*ALL)           /* Clear out directory        */

NEXTENT:                           /* Start of processing loop   */
 RCVF                              /* Get a directory entry      */
 MONMSG     MSGID(CPF0864) EXEC(DO) /* End of file processing     */
  QSYS/RCVMSG PGMQ(*SAME (*)) MSGTYPE(*EXCP) RMV(*YES) MSGQ(*PGMQ)
  GOTO CMDLBL(LASTENT)
 ENDDO

/*               Process entry based on type code               */
 IF (&RWTYPE = &LOCAL) THEN(     +
   QSYS/ADDRDBDIRE RDB(&RWRDB) RMTLOCNAME(&RWRLOC) TEXT(&RWTEXT) )

 ELSE IF (&RWTYPE = &SNA) THEN(  +
   QSYS/ADDRDBDIRE RDB(&RWRDB) RMTLOCNAME(&RWRLOC) TEXT(&RWTEXT) +
                   DEV(&RWDEV) LCLLOCNAME(&RWLLOC) +
                   RMTNETID(&RWNTID) MODE(&RWMODE) TNSPGM(&RWTPN) )

 ELSE IF (&RWTYPE = &IP) THEN(  +
   QSYS/ADDRDBDIRE RDB(&RWRDB) RMTLOCNAME(&RWSLOC *IP) +
                   TEXT(&RWTEXT) PORT(&RWPORT) )

 ELSE IF (&RWTYPE = &ARD) THEN(  +
   QSYS/ADDRDBDIRE RDB(&RWRDB) RMTLOCNAME(&RWRLOC) TEXT(&RWTEXT) +
                    ARDPGM(&RWDLIB/&RWDPGM) )

 ELSE IF (&RWTYPE = &ARDSNA) THEN(  +
   QSYS/ADDRDBDIRE RDB(&RWRDB) RMTLOCNAME(&RWRLOC) TEXT(&RWTEXT) +
```

```
                    DEV(&RWDEV) LCLLOCNAME(&RWLLOC) +
                    RMTNETID(&RWNTID) MODE(&RWMODE) TNSPGM(&RWTPN) +
                     ARDPGM(&RWDLIB/&RWDPGM) )

  ELSE IF (&RWTYPE = &ARDIP) THEN(  +
    QSYS/ADDRDBDIRE RDB(&RWRDB) RMTLOCNAME(&RWSLOC *IP) +
                     TEXT(&RWTEXT) PORT(&RWPORT) +
                     ARDPGM(&RWDLIB/&RWDPGM) )

  GOTO CMDLBL(NEXTENT)

 LASTENT:
  RETURN
  DLTLIB RDBSAV
  END


 ENDPGM
```

前述のようなタイプの出力ファイルが利用できない場合には、次のような代替手段があります。これは、保管されているサーバーからオブジェクトを抽出して、それを他の何らかのライブラリーに復元し、 RDB ディレクトリー項目の追加 (ADDRDBDIRE) コマンドを使用して手動で項目を入力する、という方法です。

システム保管 (SAVSYS) コマンドでは、リレーショナル・データベース・ディレクトリーを構成しているファイルが保管されます。リレーショナル・データベース・ディレクトリーが入っている物理ファイルは、下記のオブジェクト復元 (RSTOBJ) コマンドによって保管メディアからライブラリーに復元することができます。

```
RSTOBJ     OBJ(QADBXRDBD)  SAVLIB(QSYS)
           DEV(TAP01)  OBJTYPE(*FILE)
           LABEL(Q/ppppppp/vrmxx 0003)
           RSTLIB(your lib)
```

この例では、リレーショナル・データベース・ディレクトリーはテープから復元されます。 LABEL パラメーターの中の文字 *ppppppp* は、 OS/400 の製品コード (たとえば、バージョン 5 リリース 3 の場合には 5722SS1) です。 LABEL パラメーターの中の *vrm* は、 OS/400 のバージョン、リリース、およびモディフィケーション・レベルです。また、LABEL パラメーターの中の *xx* は、現行のサーバー言語の値の最後の 2 桁です。 たとえば、英語の場合は 2924 です。したがって、*xx* の値は 24 です。

このファイルをライブラリーに復元したら、そのファイルの情報を使用して、手動でリレーショナル・データベース・ディレクトリーを再作成できます。

分散リレーショナル・データベースのネットワークの冗長性

ネットワーク冗長性によって、分散リレーショナル・データベース・ネットワーク上のユーザーは、そのネットワーク内のリレーショナル・データベースにアクセスする手段を他にも得られることになります。**アプリケーション・リクエスター (AR)** から**アプリケーション・サーバー (AS)** に至る通信経路が 1 つしかない場合は、その通信回線が故障すると、AR 側のユーザーは AS リレーショナル・データベースにアクセスできなくなります。 したがって、Spiffy 社の分散リレーショナル・データベース管理担当者にとっても、ネットワーク冗長性は重要な問題となります。

たとえば、ある販売店の保守サービス受注または得意先部品購入の問題を考えてみましょう。ある得意先が保守サービス待機中または部品購入待ちである場合、サービス担当員には、作業の予定を立てたり部品を販売したりするために、企業情報に関して許可されているすべてのテーブルへのアクセス権が必要です。

ローカル・サーバーが故障している場合には、作業は行えません。 ローカル・サーバーが作動していても、作業を処理するためにはリモート・サーバーに対する要求が必要であり、そのリモート・サーバーが故障している場合には、その要求は処理できません。 Spiffy 社の例で言えば、これはある販売店が地域在庫センターに部品情報を要求することができないような場合に該当します。また、多くの AR ジョブを処理している AS が故障した場合には、それらの AR のいずれも要求を完了することができません。 Spiffy 社のネットワークの例では、ある地域センターが故障している場合には、その支援下にあるアプリケーション・サーバーはどれも部品を発注することができません。

地域の販売店に対して地域在庫データへのアクセスを提供することは、 Spiffy 社の分散リレーショナル・データベース管理担当者にとって重要なことです。ネットワーク内でデータに至る経路の提供は、複数の方法が可能です。 Spiffy 社の元々のネットワーク構成では、エンド・ノード販売店をそれぞれのネットワーク・ノード地域センターにリンクしていました。

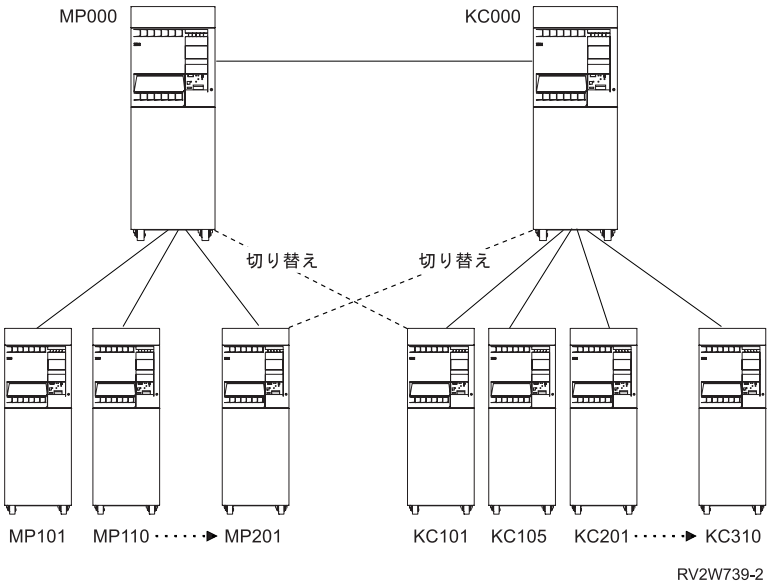


図 11. 代替ネットワーク経路

販売店によっては、別の地域センターとの交換回線接続という代替方法を取ることができます。ローカル地域センターがネットワークで使用不能になった場合には、別の AS にアクセスすることによって、要求側の販売店では、作業に必要な情報を入手することができます。 図 11 では、 MP000 サーバーからサービスを提供されている一部の販売店が KC000 サーバーとのリンクを確立し、 MP000 サーバーが使用不能になった場合にはいつでも利用できるようにしています。サーバー・オペレーターや分散リレーショナル・データベース管理者は、 構成変更 (VRYCFG) コマンドや構成状況処理 (WRKCFGSTS) コマンドを使用して、必要なときに回線をオンにし、1 次 AS が使用可能になったら回線をオフにすることができます。

区域内で比較的大規模な販売店の 1 つが他の販売店に対する AS としても機能するならば、これもやはり別の代替方法になり得ます。 138 ページの図 12 に示すように、1 つのエンド・ノードは、他のエンド・ノードに対しては、ネットワーク・ノードを介した AS に過ぎなくなります。 図 11 では、ミネアポリスとのリンクが故障した場合には、販売店 (エンド・ノード) に在庫を照会することはできません。上に図示した構成を変更して、販売店の 1 つを APPN ネットワーク・ノードとして構成し、他地区の販売店からその販売店に至る回線を設けることができます。

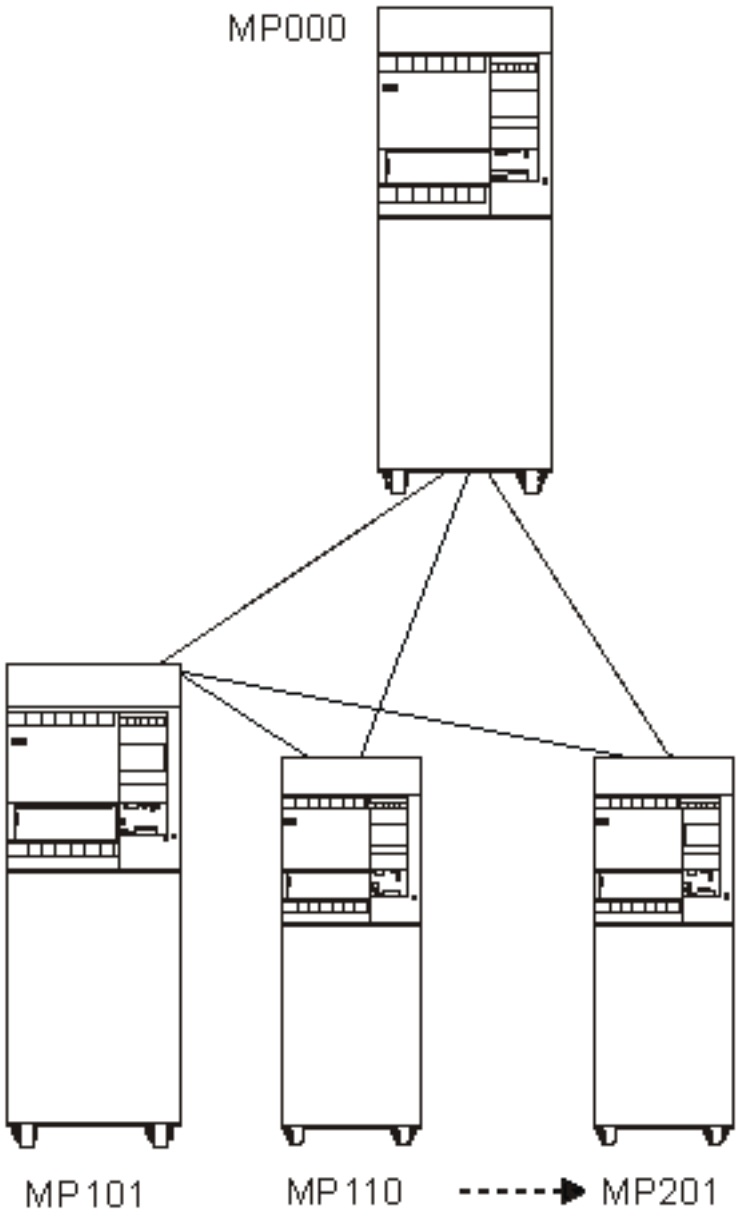


図 12. 代替アプリケーション・サーバー

分散リレーショナル・データベース・ネットワーク内のデータの冗長性

分散リレーショナル・データベースのデータの冗長性によっても、分散リレーショナル・データベース・ネットワーク上のユーザーは、そのネットワーク内のリレーショナル・データベースにアクセスする手段を他に得られることになります。データの冗長性についての方針を立てる場合に、分散リレーショナル・データベース管理担当者が検討する問題は、データに至る通信経路の可用性を確保する場合に比べて複雑です。ネ

ットワーク内のサーバー相互間でテーブルをコピーすることもでき、データのスナップショットを使用してデータの可用性に備えることもできます。 DataPropagator Relational Capture and Apply/400 プロダクトは、この機能を提供することができます。

下の図は、MP000 サーバーの分散リレーショナル・データベースのコピーは、 KC000 サーバー上に保管することができ、KC000 サーバーの分散リレーショナル・データベースのコピーは、 MP000 サーバー上に保管することができることを示しています。一方の地域の**アプリケーション・リクエスター (AR)** は、他方の**アプリケーション・サーバー (AS)** にリンクして、自らのリレーショナル・データベースの複製コピーを照会または更新することができます。

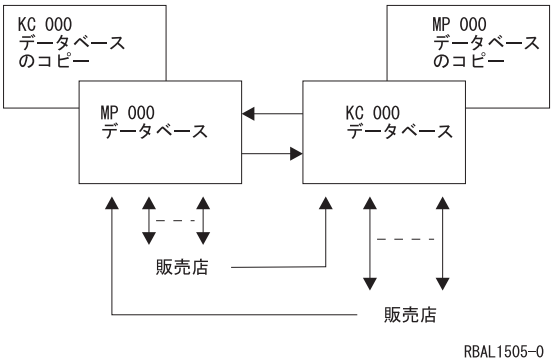


図 13. データ冗長性の例

管理担当者は、分散リレーショナル・データベース処理を可能にする方針として、最も効率的で効果的なものを決定しなければなりません。方針としては次のようなものがあります。

方針の 1 つは、MP000 が使用不能になった時には、その地域の AR は KC000 サーバーに接続して、MP000 分散リレーショナル・データベースの読み取り専用スナップショットを照会し、保守サービス作業のスケジュールを決めることができるようにするというものです。

DataPropagator Relational/400 は、テーブルの読み取り専用コピー (またはスナップショット) を定期的にリモート・サーバーに提供することができます。 Spiffy 社の場合であれば、これは毎営業日の終了時または開始時に行うようにすることができます。この例では、MP000 データベースのスナップショットは、販売店がスケジュール専用として使用するように、24 時間前の最終時点状況を提供します。 MP000 サーバーがオンラインに復帰したら、その AR は MP000 分散リレーショナル・データベースに照会して、スナップショットに照会していた在庫要求またはその他の作業を完全に処理します。

ある地域の AS が使用不能になった時には、その域内販売店ユーザーは、別の地域の AS にある複製テーブルを更新できるようにしたいと Spiffy 社が考える場合には、別の方針を立てることになります。

たとえば、通常は MP000 データベースに接続されている AR が、 KC000 サーバー上にある複製 MP000 データベースに接続して、作業を処理することができるようにします。 MP000 サーバーが再び使用可能になったら、MP000 リレーショナル・データベースは、 KCOOO ロケーションにあるその複製テーブルに生じている活動によるジャーナル項目を適用することによって、更新することができます。 このようなジャーナル項目が元の MP000 テーブルに適用されてしまえば、分散リレーショナル・データベース・ユーザーは、AS としての MP000 に再びアクセスすることができます。

各地域サーバーでのジャーナル管理処理によって、すべてのリレーショナル・データベースを更新します。この状況におけるジャーナル管理コピー活動は、これらのサーバーでパフォーマンス上の悪影響を及ぼす可能性があるので、その量を調べなくてはなりません。

第 8 章 分散リレーショナル・データベースのパフォーマンス

どのような種類のアプリケーション・プログラムをサーバーで実行する場合でも、パフォーマンスは常に重要な問題となります。 分散リレーショナル・データベースの場合、ネットワーク、サーバーおよびアプリケーションのパフォーマンスすべてが重要になります。 サーバー・パフォーマンスは、主記憶域および補助記憶域のサイズおよび編成によって影響を受けることがあります。 SQL プログラムの長所と短所を心得ていれば、パフォーマンスを向上させることができます。詳細については、『第 9 章 分散リレーショナル・データベースの問題の処理』を参照してください。

ネットワーク、サーバー、およびデータベースの設計をより良くする詳しい方法は、以下のトピックを参照してください。

- ネットワーク全体での分散リレーショナル・データベースのパフォーマンスの向上
- サーバー全体での分散リレーショナル・データベースのパフォーマンスの向上
- データベース全体での分散リレーショナル・データベースのパフォーマンスの向上

ネットワーク全体での分散リレーショナル・データベースのパフォーマンスの向上

ネットワークのパフォーマンスは、さまざまな方法で改善できます。その方法には、次のものがあります。

- 回線速度
- ペーシング
- フレーム・サイズ
- RU のサイズ
- 接続タイプ (非交換と交換)

注: 接続が RUW 接続管理を使用しているプログラムから行われる場合、または接続を行うプログラムがコミットメント制御の下で動いていない場合、あるいは、接続先のデータベースが、使用するプロトコルの 2 フェーズ・コミットをサポートしていない場合には、 DRDA 接続のために非保護会話が使われます。データに、トランザクションが 1 つのデータベース管理システムだけに影響するなどの特性がある場合、 RUW 接続管理使用のプログラムから、またはコミットメント制御を使わずに実行するプログラムから接続を確立すると、 2 フェーズ・コミット・フローに関連したオーバーヘッドを避けられます。

さらに、会話が DDMCNV(*KEEP) で活動状態になっていて、なおかつその会話が保護会話であると、その作業単位で DRDA または DDM 処理のどちらのために会話が使用されたかに関係なく、2 フェーズ・コミット・フローが送信されます。したがって、DDMCNV(*KEEP) を使って実行しているのであれば、可能なら非保護会話を使用する方が望ましいといえます。保護会話で実行しているときは、DDMCNV(*DROP) で実行するようにします。そして、それ以降の作業単位でその会話を使わない場合は、次のコミットで、 RELEASE ステートメントを使用して接続と会話を終わらせます。

詳細は、V5R1 Supplemental Manual Web サイトにある、 *「Communications Management」*  を参照してください。 RU のサイズおよびペーシングについては、『APPC、APPN、および HPR』をご覧ください。

この他の通信関連のパフォーマンスに関する考慮事項は、 iSeries Information Center の『TCP/IP セットアップ』のトピックを参照してください。

サーバー全体での分散リレーショナル・データベースのパフォーマンスの向上

効率的なサーバー・パフォーマンスを達成するには、サーバー・リソース間の適正なバランスが必要です。過剰に使用しているリソースがあれば、パフォーマンスに悪影響があります。

ここでは、サーバー・パフォーマンスの観察に役立つサーバー・コマンドについて説明します。

iSeries パフォーマンス・ツール・ライセンス・プログラムを使用して、パフォーマンスの分析に役立てることができます。加えて、サーバー・パフォーマンスの観察に役立ついくつかのシステム・コマンドも使用できます。

- システム状況の処理 (WRKSYSSTS) コマンド
- ディスク状況の処理 (WRKDSKSTS) コマンド
- WRKACTJOB (活動ジョブの処理) コマンド

それらのコマンドを使う場合、一般的なレベルの活動時にサーバー・パフォーマンスを観察するようにします。 たとえば、サーバー上でジョブがまったく実行されていないときに統計を収集しても、サーバー・パフォーマンスの評価を行う上でほとんど価値がありません。 サーバー・パフォーマンスを観察するときには、以下のステップを完了してください。

1. (WRKSYSSTS)、(WRKDSKSTS)、または (WRKACTJOB) コマンドを入力する。
2. サーバーが最低 5 分間データを収集できるようにする。
3. F5 (再表示) を押して、画面を再表示し、パフォーマンス・データを表示する。
4. 新しいデータに基づいてサーバーを調整する。

経過時間カウンター機構を再始動するには、F10 (再始動) を押してください。

サーバー状況の処理およびディスク状況の処理方法については、『実行管理機能』をご覧ください。

前述のWRKACTJOB (活動ジョブの処理) コマンドが持つ 1 つの機能は、サーバー・パフォーマンスの計測です。 98 ページの『分散リレーショナル・データベースの活動ジョブの処理』に示されている「活動ジョブの処理」画面が表示されます。

システムのパフォーマンスを観察する際は、 システム状況の処理 (WRKSYSSTS) コマンドと WRKACTJOB (活動ジョブの処理) コマンドを両方使用します。各観察期間ごとに、サーバー・パフォーマンスの測定値を、あらかじめ設定してある目標値と対比して、検討および評価してください。

一般的な測定値には、次のものが含まれます。

- (WRKACTJOB) 画面から得られる対話式スループットおよび応答時間。
- バッチ・スループット。活動状態バッチ・ジョブの AuxIO および CPU% 値を観察します。
- スプール・スループット。活動書き出しプログラムの AuxIO および CPU% 値を観察します。

調整を加えるたびに、主要なパフォーマンス測定値のすべてを、測定し比較する必要があります。調整の実行および評価は、一度に 1 つずつ行ってください。

データベース全体での分散リレーショナル・データベースのパフォーマンスの向上

分散リレーショナル・データベースのパフォーマンスは、 15 ページの『第 2 章 分散リレーショナル・データベースの計画および設計』で言及したように、データベースの総合的な設計の影響を受けます。分散データを配置する位置、使用するコミットメント制御のレベル、および SQL 索引の設計など、すべてがパフォーマンスに影響します。

データベース・パフォーマンスの最適化については、以下のトピックを参照してください。

- DRDA データ・ロケーションの決定
- DRDA のブロック化に影響を与える要因
- DRDA 照会ブロックのサイズに影響を与える要因

DRDA データ・ロケーションの決定

アプリケーションとアプリケーションで必要とするデータとの間にネットワークを介在させれば、パフォーマンスの低下を招くと考えられるので、データを配置する場所の決定にあたっては、次のことを考慮してください。

- データを使用するトランザクション
- トランザクションが実行される頻度
- トランザクションで送受信されるデータの量

実行頻度が高いか、またはデータの送受信量が多いトランザクションを伴うアプリケーションの場合には、データと同じロケーションにアプリケーションを保持するように努めてください。たとえば、1 秒間に何度も実行されたり、一度に何百行ものデータを受信するアプリケーションの場合には、アプリケーションとデータが同じサーバー上にあれば、パフォーマンスはそれだけ向上します。 これに対して、アプリケーションに伴うトランザクションの実行頻度が低かったり、一度に送受信するデータの量が少ない場合には、データは、それを必要とするアプリケーションとは別のロケーションに配置することを考えてください。

DRDA のブロック化に影響を与える要因

パフォーマンスに影響する最も重要な要因は、**アプリケーション・リクエスター (AR)** と**アプリケーション・サーバー (AS)** の間でデータが転送されるときに、ブロック化が行われるかどうかということです。データのブロックとして伝送される行のグループの場合、同じデータを一度に 1 行ずつ伝送する場合と比べて、通信のオーバーヘッドははるかに少なくて済みます。別の iSeries server に接続されている場合に、ブロック化を制御する方法の 1 つとして、 OS/400 オペレーティング・システムのバージョン 2 リリース 2 またはそれ以降を用いて、 SQL の複数行 INSERT ステートメントおよび複数行 FETCH ステートメントを使う方法があります。 複数行 FETCH を使うと、ハード・エラーの発生やデータ終わりになった場合を除き、 FOR n ROWS 文節に指定された行数のブロック化が強制的に行われます。以下の説明では、単一行 FETCH の場合に、ブロック化が行われるかどうかを判別するための規則を紹介します。

AR と AS との間の照会データのブロック化を禁止する条件も、以下の説明の中で示しています。これらの条件は、複数行 FETCH ステートメントを使用する場合には、適用されません。以下のそれぞれのケースでリストされている条件が 1 つでも当てはまれば、ブロック化は行われません。

ケース 1: DB2 UDB for iSeries から DB2 UDB for iSeries へのブロック化

次の場合にはブロック化は行われません。

- カーソルが更新可能である (注 1 参照)。

- カーソルが潜在的に更新可能である (注 2 参照)。
- ALWBLK(*NONE) プリコンパイル・オプションが使用された。
- コミットメント制御レベルが *CS で、 OS/400 のレベルがバージョン 3 リリース 1 より前である。
- コミットメント制御レベルが *ALL で、外側の副選択に次のいずれも含まれていない。
  - DISTINCT キーワード
  - UNION 演算子
  - ORDER BY 文節 (その文節内のフィールドの長さの合計がソートを必要としている)
  - サーバー・データベース・ファイル参照 (サーバー・データベース・ファイルとは、 QADBxxxx という名前のライブラリー QSYS に入っているファイルおよびそれらのファイルに構築されているビュー)
- 行サイズが 2K より大きいか、あるいは、 SBMRMTCMD (リモート・コマンド投入) コマンドまたはストアード・プロシージャーをデフォルト AS データベース・バッファーのサイズ拡張のために使用している場合に、行サイズが、データベース・ファイルによる指定変更 (OVRDBF) コマンドの SEQONLY レコード数パラメーターの指定に基づいたデータベース・バッファーのサイズの約半分より大きくなっている。 ((OVRDBF) コマンドをリモートで実行する場合、 OVRSCOPE(*JOB) を指定する必要があることに注意してください。)
- カーソルがスクロール可能として宣言され (DECLARE...SCROLL CURSOR...)、 FETCH ステートメントに指定されたスクロール・オプションが次のいずれかである: RELATIVE、PRIOR、または CURRENT (ただし、前述のように複数行 FETCH が実行されている場合はその限りではない。)

ケース 2: DB2 UDB for iSeries から非 DB2 UDB for iSeries へのブロック化

次の場合にはブロック化は行われません。

- カーソルが更新可能である (注 1 参照)。
- カーソルが潜在的に更新可能である (注 2 参照)。
- ALWBLK(*NONE) プリコンパイル・オプションが使用されている。
- 行サイズが約 16K より大きい。

ケース 3: 非 DB2 UDB for iSeries から DB2 UDB for iSeries へのブロック化

次の場合にはブロック化は行われません。

- カーソルが更新可能である (注 1 参照)。
- カーソルが潜在的に更新可能である (注 2 参照)。
- パッケージのデフォルト値が「単一行強制」プロトコルになるプリコンパイルまたはバインド・オプションが使用されている。
  - DB2 の場合、このようなオプションはありません。
  - DB2 UDB for VM の場合、これは SQLPREP の NOBLOCK キーワードです (デフォルト値)。
  - DB2/2 の場合、これは SQLPREP または SQLBIND の /K=NO です。
- 行サイズが約 0.5*QRYBLKSIZ より大きい。 QRYBLKSIZ のデフォルト値と最大値は次のようになっています。

表 7. QRYBLKSIZ

DB2 製品名	デフォルト QRYBLKSIZ	バージョン 8 の最大 QRYBLKSIZ
DB2 Universal Driver for SQLJ and JDBC	32K	32K
DB2 UDB for z/OS	32K	64K

表 7. QRYBLKSIZ (続き)

DB2 製品名	デフォルト QRYBLKSIZ	バージョン 8 の最大 QRYBLKSIZ
DB2 UDB for VM	8K	32K
DB2 Connect	32K	64K

最新レベルの DRDA では、戻される結果セットに固定 Query ブロック・サイズの制限がないモードで処理を行うよう、サーバーに選択させることができます。

- カーソルはスクロール可能として定義されており、ブロック・カーソルはアプリケーションで使用されていません。

DRDA ブロック化規則の要約

要約すると、前述の規則 (注も含む) の意味するところは、一定の特別な条件または通常ではない条件がなければ、次のいずれの場合もブロック化が行われるということです。

- カーソルが読み取り専用 (注 3 参照) であって、かつ、次の場合。
  - アプリケーション・リクエスターまたはアプリケーション・サーバーのどちらかが非 DB2 Universal Database for iSeries である
  - アプリケーション・リクエスターとアプリケーション・サーバーの両方が DB2 Universal Database for iSeries で、 ALWBLK(*ALLREAD) と指定してあり、COMMIT(*ALL) の指定は**ない**
- COMMIT(*ALL) の指定がなく、次のすべてを満たしている場合。
  - SELECT の中に FOR UPDATE OF 文節がなく、しかも、
  - プログラム中にカーソルに対する UPDATE または DELETE WHERE CURRENT OF ステートメントがなく、しかも、
  - プログラムに動的 SQL ステートメントが入っていない場合、または、限定ブロック・プロトコル (DB2 Connect では /K=ALL、 DB2 UDB for iSeries では ALWBLK(*ALLREAD)、DB2 UDB for z/OS では CURRENTDATA(NO)、 DB2 UDB for VM では SBLOCK) の要求にプリコンパイル/バインド・オプションが使用された場合。

注:

1. カーソルは、それが読み取り専用ではなく (注 3 参照)、しかも以下のいずれかに該当する場合には更新可能です。
   - 選択ステートメントに、FOR UPDATE OF 文節が含まれている。または
   - プログラム中に、カーソルに対する UPDATE または DELETE WHERE CURRENT OF がある。
2. カーソルは、それが読み取り専用 (注 3 参照) ではなく、しかもプログラムに EXECUTE または EXECUTE IMMEDIATE ステートメント (あるいは、非 iSeries server システムに接続されている場合は、動的ステートメント) が組み込まれており、パッケージのデフォルト値を単一行プロトコルにするプリコンパイルまたはバインド・オプションが使用されている場合、潜在的に更新可能です。
   - DB2 Universal Database for iSeries の場合、これは ALWBLK(*READ) プリコンパイル・オプション (デフォルト) です。
   - DB2 の場合、これは BIND PACKAGE の CURRENTDATA(YES) (デフォルト) です。
   - DB2 UDB for VM の場合、これは SQLPREP の SBLOCK キーワードです。
   - DB2/2 の場合、これは SQLPREP または SQLBIND の /K=UNAMBIG (デフォルト) です。
3. カーソルは、次の条件の 1 つまたはいくつかに該当する場合は読み取り専用です。
   - DECLARE CURSOR ステートメントに ORDER BY 文節の指定があり、FOR UPDATE OF 文節の指定はない。

- DECLARE CURSOR ステートメントに FOR FETCH ONLY 文節の指定がある。
- DECLARE CURSOR ステートメントに、 DYNAMIC の指定のない SCROLL キーワードを指定した (OS/400 のみ)。
- 以下の条件のうち 1 つ以上が、カーソルについて、あるいはそのカーソルの参照先の外側の副選択内で参照されているビューまたは論理ファイルについて当てはまる。
  - 外側の副選択に、DISTINCT キーワード、GROUP BY 文節、HAVING 文節、または列関数が含まれている。
  - その選択に結合関数が含まれている。
  - その選択に UNION 演算子が含まれている。
  - その選択に最も外側の副選択のテーブルと同じテーブルを参照する副照会が含まれている。
  - その選択に一時ファイルにコピーしなければならない複合論理ファイルが含まれている。
  - 選択された列のすべてが、式、スカラー関数、または定数である。
  - 参照された論理ファイルの列のすべてが入力専用である (OS/400 のみ)。

DRDA 照会ブロックのサイズに影響を与える要因

照会で大量のデータが戻される場合、照会データのブロックのサイズを大きくすることにより、パフォーマンスが向上する場合があります。これを行う方法は、その照会に関与するサーバーの種類によって異なります。 異種環境においては、照会ブロックのサイズは、照会オープン・コマンドによって送られるパラメーターによって、アプリケーション・リクエスターで決定されます。 iSeries server が**アプリケーション・リクエスター (AR)** の場合は、必ず最初に 32K の照会ブロック・サイズを要求します。このため、伝送に複数のブロックを必要とする大規模な照会のために要求された正常な照会ブロックでは、その分それぞれのブロックのサイズが大きくなります。他のタイプの AR では、ユーザーが使用するブロック・サイズを選択できます。 DB2 Universal Driver for SQLJ and JDBC、DB2 UDB for z/OS、 DB2 UDB for VM、および DB2 Connect のデフォルト照会ブロック・サイズは、それぞれ 32K、32K、8K、32K です。 DB2 UDB for iSeries サーバーが異環境 AR に接続しているときに、 AR として使用されるプラットフォームについては、そのプロダクトの資料を参照してください。

DB2 UDB for iSeries と DB2 UDB for iSeries との間の環境では、照会ブロック・サイズは、データベース・マネージャーによって使用されるバッファーのサイズによって決まります。デフォルトのサイズは 4K です。このサイズは、バージョン 2 リリース 3 以降のアプリケーション・サーバーで変更できます。この変更を行うには、SBMRMTCMD (リモート・コマンド投入) コマンドを使用して、**アプリケーション・サーバー (AS)** に データベース・ファイルによる指定変更 (OVRDBF) コマンドを送信し実行します。指定変更するファイルの名前の他に、(OVRDBF) コマンドには、 OVRSCOPE(*JOB) および SEQONLY(*YES nnn) を含める必要があります。 ブロック当たりの所要レコード数を、SEQONLY パラメーターの nnn に指定します。データベース・バッファーのサイズを大きくすることにより、通信オーバーヘッドを軽減できるばかりでなく、行の検索に必要なデータベース・マネージャーへの呼び出しの回数も減らせます。

非 iSeries server から、または iSeries server 間で、 SQL CALL ステートメント (ストアード・プロシージャー) を使用して、照会ブロック・サイズを変更することもできます。

第 9 章 分散リレーショナル・データベースの問題の処理

分散リレーショナル・データベースへのアクセスで問題が生じた場合、管理担当者は以下の作業を行わなければなりません。

- 問題の性質を判別する。
- その問題がアプリケーションに関連する問題か、あるいはローカル・システムまたはリモート・システムに関連する問題かを判別する。

それから問題を解決するか、そうでなければ、顧客サポートの援助を受けて、問題を解決しなければなりません。そのためには、以下のことを行う必要があります。

- OS/400 プログラム・サポートについて理解する。
- 問題が、**アプリケーション・リクエスター (AR)** と、**アプリケーション・サーバー (AS)** のどちらにあるかを判別する。
- OS/400 の問題管理機能の使用方法を習得する。

分散リレーショナル・データベース問題の詳細については、以下のトピックを参照してください。

- iSeries の問題処理の概要
- 分散リレーショナル・データベースの問題の分離
- 分散リレーショナル・データベースのユーザーとの共同作業
- アプリケーションの問題
- 障害の報告のためのデータの入手
- 第 1 障害データ検知 (FFDC) データの検索
- アプリケーション・サーバー問題の診断のためのサービス・ジョブの開始
- システムおよび通信の問題

分散リレーショナル・データベースでの問題の診断方法の詳細は、*「分散関係データベース 問題判別の手引き」* を参照してください。

iSeries の問題処理の概要

ローカルおよびリモート iSeries server で、ユーザー検出またはシステム検出のどちらの問題が発生した場合でも、その問題を管理するのに OS/400 プログラムが役立ちます。問題処理のサポートには次のものが含まれます。

- 初期問題処理に関する情報が含まれるメッセージ
- システム検出の問題の自動警報
- 統合問題ロギングおよび追跡
- 第 1 障害データ検知 (FFDC) サポート
- エレクトロニック支援のサービス要求
- エレクトロニック支援、プログラム一時修正 (PTF) 要求

iSeries server とその接続装置は、特定のタイプの問題を検出することができます。これを**システム検出問題**と呼びます。 問題が検出されると、次のようないくつかの操作が行われます。

- プロダクト活動記録ログ内に項目が作成される
- 問題記録が作成される
- メッセージが QSYSOPR メッセージ待ち行列に送られる

情報は、エラー・ログと問題記録に記録されます。深刻な問題が検出された場合は、FFDC 情報のスプール・ファイルも作成されます。エラー・ログと問題記録には、次のような情報が入れられます。

- 重要プロダクト・データ
- 構成情報
- 参照コード
- 対応する装置の名前
- 補足的な障害情報

通常、**ユーザー検出**問題は、次のような問題を引き起こす原因となるプログラム・エラーに関連したものです。

- ジョブ問題
- 正しくない出力
- プログラム障害を示すメッセージ
- システムでは検出されない装置障害
- 劣悪なパフォーマンス

ユーザーが問題を検出しても、問題分析を実行するか、または、操作援助機能 (Operational Assistant*) の「USERHELP」メニューで、問題の解決に役立つ情報を保管するためのオプションを選択しない限り、サーバーでは何も情報は集められません。

iSeries server は、問題ログと問題管理機能を使って、ユーザー検出とシステム検出の両方の問題を追跡します。 問題が検出されたとき (OPENED) から、それが解決される (CLOSED) まで、問題プログラム状態は保持されるので、追跡に利用することができます。詳しくは 170 ページの『iSeries 問題ログ』をご覧ください。

分散リレーショナル・データベースの問題の分離

分散リレーショナル・データベース・アプリケーションの実行時に検出される問題は、次のような 2 つの一般的な徴候を示す可能性があります。

- ユーザーが正しくない出力を受け取る
- アプリケーションが予定時間内に完了しない

以下の図と手順は、アプリケーション・プログラム問題、パフォーマンス関連問題、およびサーバー関連問題のいずれかに問題を分類する一般的な方法を示しています。このように分類すれば、iSeries server の標準の問題分析方法を使って、問題を解決することができます。

DRDA の誤った出力の問題

エラー・メッセージが出された場合、エラー・メッセージ、SQLCODE、または SQLSTATE を使って、問題の原因を判別します。 149 ページの図 14を参照してください。メッセージの説明に、どのような問題であるかが示され、訂正アクションが指示されています。エラー・メッセージが出なかった場合、分散リレーショナル・データベースが障害の原因かどうかを自分で判断しなければなりません。そのためには、障害が起きたステートメントを**アプリケーション・サーバー (AS)** 上でローカル実行するか、または対話式構造化照会言語 (SQL) を使ってステートメントを AS 上で実行します。問題をローカルで生成できる場合、問題

の原因は分散リレーショナル・データベース・サポートにはありません。この操作の結果に応じて、iSeries server の問題分析方法を使って、サポート担当者に該当情報を提出してください。

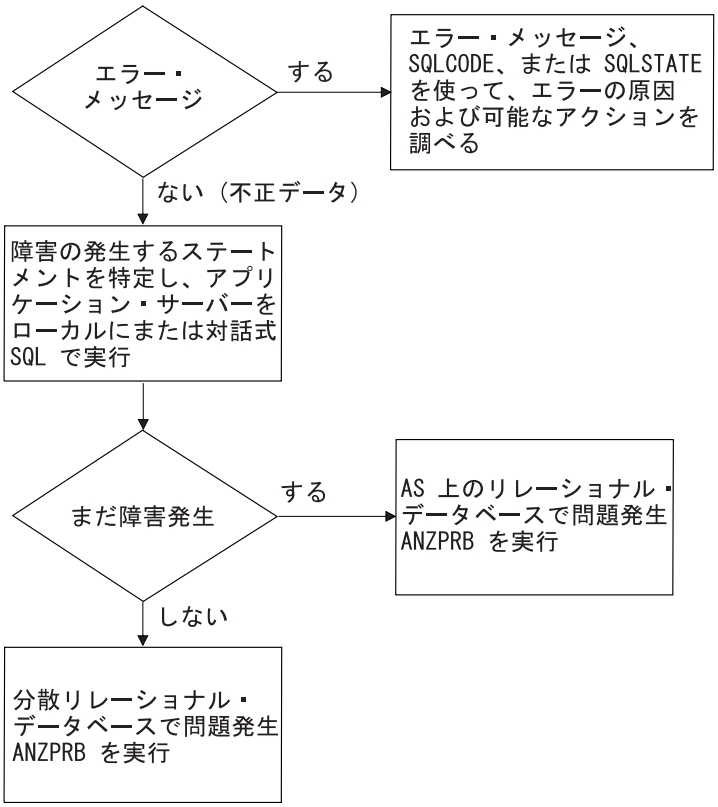
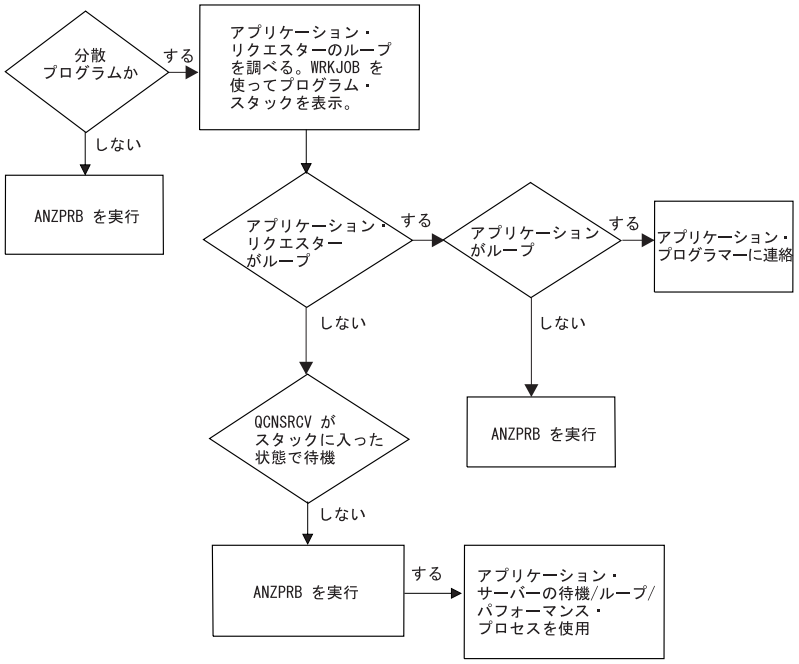


図 14. 正しくない出力の問題の解決

予定時間内に完了しないアプリケーションの問題

要求が完了するのに想定された時間より長くかかっている場合、まず**アプリケーション・リクエスター (AR)** で検査を行います。ジョブ・ログを調べて、リレーショナル・データベースへの接続が完了したことを知らせるメッセージ SQL7969 を確かめます。これは、アプリケーションが分散リレーショナル・データベース・アプリケーションであることを知らせるメッセージです。 AR を検査してループを探します。そのためには、 ジョブ処理 (WRKJOB) コマンドを使って、プログラム・スタックを表示し、そのスタックを調べてシステムがループになっているかどうかを判別します。 150 ページの図 15 を参照してください。アプリケーションそのものがループになっている場合、解決のためアプリケーション・プログラマーに連絡してください。スタック上に QAPDEQUE および QCNSRCV が見つかった場合、 AR は**アプリケーション・サーバー (AS)** 待ちになっています。 151 ページの図 16 を参照してください。システムが通信待ち状態でない場合、問題分析手順を使って、パフォーマンス上の問題があるかどうか、またはどこか他の場所が待ち状態になっているかどうかを確かめます。



RV2W732-2

図 15. アプリケーション・リクエスター側の待機、ループ、またはパフォーマンスの問題の解決

AS 上のジョブ・ログを探せば、AR ジョブ名を見つけることができます。 AS 上でのジョブの検索方法の詳細は、 101 ページの『分散リレーショナル・データベース・ジョブの探索』を参照してください。 AS ジョブを検査する必要が生じたときは、 ジョブ処理 (WRKJOB)、 WRKACTJOB (活動ジョブの処理)、またはユーザー・ジョブ処理 (WRKUSRJOB) コマンドを使って、 AS 上のジョブを見つけ出します。これらのコマンドの使用について詳細は、以下のトピックを参照してください。

- 96 ページの『分散リレーショナル・データベースのジョブの処理』
- 97 ページの『分散リレーショナル・データベースのユーザー・ジョブの処理』
- 98 ページの『分散リレーショナル・データベースの活動ジョブの処理』

これらのジョブのうちのいずれかの表示でプログラム・スタックを調べて、 AS がループになっているかどうかを確認します。ループになっている場合、問題分析を使って問題を処理します。ループになっていない場合、プログラム・スタックが、 QCNTRCV での待ち状態になっているかどうかを調べます。そうなっていると、AS は AR 待ちであることを意味します。どちらのサーバーもこのような通信待ち状態であれば、ご自分のネットワークに問題があるということになります。 AS が待ち状態でなければ、対処する必要のあるパフォーマンス上の問題があるということです。

照会のパフォーマンスの低下に共通する 2 つの原因として以下のことが挙げられます。

- アクセスされているテーブルに、索引がない。そのような場合、適切な 1 つ以上のフィールドをキーとして使って索引を作成します。
- 照会要求で戻された行がブロック化されていない。行がブロック化されているかどうかによって、照会のパフォーマンスに大きな差が生じることがあります。ブロック化に影響を与える要因を理解して、その利点を活用できるようアプリケーションを調整することが大切です。詳しくは 143 ページの『DRDA のブロック化に影響を与える要因』をご覧ください。

DB2 JDBC Universal Driver や DB2 Connect などの製品を使用して、ワークステーションから DB2 UDB for iSeries に初めてアクセスする際、その製品用の SQL パッケージが DB2 UDB for iSeries 内にあらかじめ作成されていない場合は、そのパッケージが自動的に作成されます。また、NULLID コレクションも同じように自動的に作成される必要がある場合があります。このため、初期接続後に発行される最初の SQL ステートメントの 1 つに関して、サーバーからの応答にいくぶん長い遅れが生じます。

TCP/IP を介して接続しようとしているサーバーが使用可能になっていないと、長い遅延が生じることになります。 数分間の遅延の後、メッセージ `A remote host did not respond within the timeout period` が出されます。 RDB ディレクトリー内の IP アドレスが正しくない場合にも、このような動作を生じます。

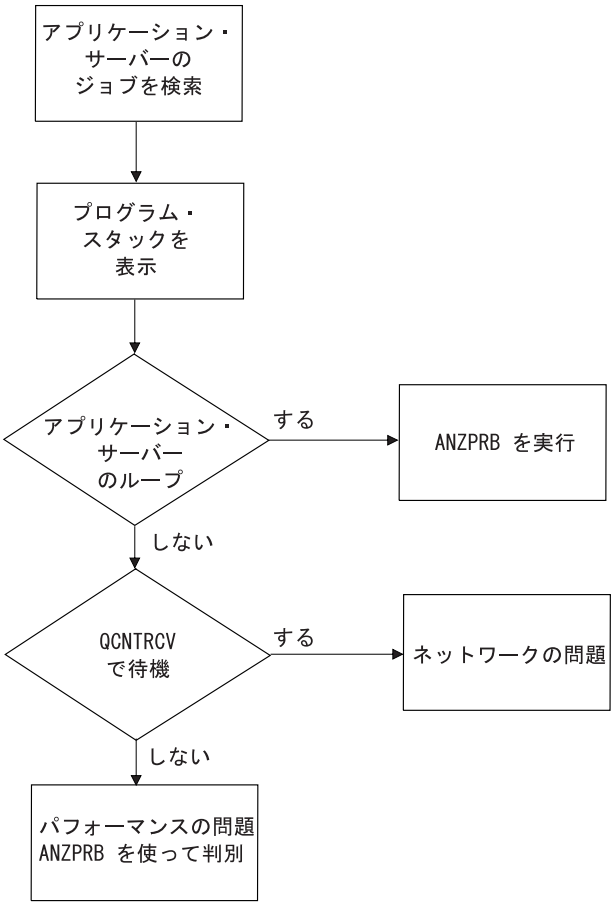


図 16. アプリケーション・サーバー側の待機、ループ、またはパフォーマンスの問題の解決

分散リレーショナル・データベースのユーザーとの共同作業

通常、問題の調査は、ユーザーと一緒に行います。 ユーザーは、プログラムを実行したときに所定の結果を得られなかったり、問題を知らせるメッセージを受け取ったりすることがあります。問題を診断して解決するには、場合によっては、ユーザーと一緒に手順をたどっていくのが最良の方法です。そのためには、画面コピー機能を使用して、ユーザーと一緒にリアルタイムで、または、ユーザーが前に見た表示のファイルを調べながら作業します。

また、表示の下端に表示されるテキスト行だけでなく、 メッセージからもさらに情報を得ることができます。この項では、別のユーザーが表示していた画面をコピーする方法と、分散リレーショナル・データベースの作業中に出されるメッセージに関する追加情報の入手方法について説明します。

プログラミング問題に加え、問題の開始またはサーバーへの接続に関する問題を抱えることがあります。これらの問題の処理方法の詳細については、以下のトピックを参照してください。

- APPC のプログラム開始要求障害の処理
- TCP/IP の接続要求障害の処理

コピー画面

画面コピー開始 (STRCPYSCN) コマンドを使って、ワークステーションにサインオンしてから、別のワークステーションで誰か他の人が見ているのと同じ画面を見ることができます。ユーザーと同じ iSeries server にサインオンする必要があります。 そのユーザーがリモート・サーバーにいる場合、ディスプレイ・パススルーを使ってそのシステムにサインオンしてから、 (STRCPYSCN) コマンドを入力してその画面を見ることができます。画面イメージをデータベース・ファイルにコピーできるのは、別のワークステーションにそのイメージがコピーされるときと同時、または別のワークステーションが使えなくなっているときです。コピーしておけば、そのデータを後で処理して、問題の発生時に行われる操作用の監査証跡を準備することができます。

画面イメージを別のワークステーションにコピーするためには、次の要件が満たされなければなりません。

- 両方の画面がともにサーバーに対して定義されている。
- 両方の画面がともにカラーであるか、またはともにモノクロームであり、一方がカラーで他方がモノクロームであるということはない。
- 両方の画面が縦横ともに同数の文字位置を持っている。

送信側装置として自分自身のステーション ID を入力した場合、画面イメージのコピーの開始時に、受信側のディスプレイは、サインオン画面を表示していなければなりません。グラフィックスはブランクとしてコピーされます。

同じサーバーにまだサインオンしていない場合、次のようなプロセスを使って、他方のユーザーがリモート・サーバー上で見ている画面を表示します。

注: xv ページの『コードの特記事項情報』に重要なリーガル情報がありますので、お読みください。

1. パススルー開始 (STRPASTHR) コマンドを入力します。

   ```
   STRPASTHR RMTLOCNAME(KC105)
   ```

2. アプリケーション・ソース (AS) にログオンします。
3. (STRCPYSCN) コマンドを入力します。

   ```
   STRCPYSCN SRCDEV(KC105)
             OUTDEV(*REQUESTER)
             OUTFILE(KCHELP/TEST)
   ```

- SRCDEV には、ソース装置 (表示イメージを送信するディスプレイ) の名前を指定します。表示コマンドを別の装置に送るには、このパラメーターに *REQUESTER 値を入力します。
- OUTDEV には、表示イメージの送信先の出力装置の名前を指定します。この例では、表示イメージは、コマンド (*REQUESTER) を入力した人のディスプレイに送られます。また、別のディスプレイや別の装置 (第三者が見ているもの) を指定することもできます。あるいは、どの装置にも送らないという指定も可能です (*NONE)。 *NONE 値を使うときは、表示イメージ用の出力ファイルを指定します。
- OUTFILE は、コマンドが活動状態にある間に表示されるすべての画面のイメージが入る出力ファイルの名前を指定します。

4. ソース装置に照会メッセージが送られ、その装置のユーザーに対して、別の装置またはファイルに画面がコピーされることが知らされます。要求側の装置にイメージを送るには、g (Go) と入力します。

送信側のディスプレイの画面が、他方のディスプレイにコピーされます。受信側のディスプレイに表示されるイメージは、送信側のディスプレイのものの 1 画面後をたどります。送信側のディスプレイのユーザーが、活動状態にないキー (Home キーなど) を押すと、両方のディスプレイに同じ画面が表示されます。

画面のコピー中は、受信側表示装置のオペレーターは、画面のコピーが終了するまで、表示装置で他の作業を行うことはできません。

送信側のディスプレイでコピー画面機能を終了するには、任意のコマンド行から画面コピー終了 (ENDCPYSCN) コマンドを入力して、実行キーを押します。

```
ENDCPYSCN
```

コピー画面機能を開始したときに表示されていた画面が表示されます。

メッセージ

iSeries server は、さまざまなシステム・メッセージを送って、単純なタイプ入力エラーから、サーバー装置またはプログラムでのエラーまでの各種の状態を示します。メッセージは以下のいずれかです。

- 現在の画面に表示されるエラー・メッセージ。

  このメッセージは、ジョブに割り込んだり、アラームを鳴らしたりすることがあります。このメッセージを表示するには、任意のコマンド行に DSPMSG と入力します。
- サーバー・オペレーター・メッセージ待ち行列に送られて別の「メッセージの処理」画面に表示される、サーバー問題に関するメッセージ。

  これらのメッセージを表示するには、任意のサーバー・コマンド行に DSPMSG QSYSOPR と入力します。
- 装置記述に指定されたメッセージ待ち行列に送られる、サーバー問題に関するメッセージ。

  これらのメッセージを表示するには、任意のサーバー・コマンド行に DSPMSG message-queue-name と入力します。

サーバー・イベントによって、サーバーは、通知メッセージまたは照会メッセージを送ります。 **通知メッセージ**は、サーバーが何を行っているかの状況を知らせます。 **照会メッセージ**は、サーバーに関する情報を知らせるとともに、応答を要求します。

メッセージ画面によっては、次のような文字や数字コードがメッセージに添えられていることがあります。

```
CPF0083
```

最初の 2 つまたは 3 つの文字は、メッセージのカテゴリーを示します。以下に、分散リレーショナル・データベースのいくつかのメッセージ・カテゴリーを示します。

表 8. メッセージ・カテゴリー

カテゴリー	説明	ライブラリー
CPA から CPZ	オペレーティング・システムからのメッセージ	QSYS/QCPFMSG
MCH	ライセンス内部コード・メッセージ	QSYS/QCPFMSG
SQ および SQL	構造化照会言語 (SQL) メッセージ	QSYS/QSQLMSG
TCP	TCP/IP メッセージ	QTCP/QTCPMSGF

その後の 4 桁 (接頭部が SQ の場合は 5 桁) の数字は、メッセージの順序番号を示します。例に示されているメッセージ ID は、これがオペレーティング・システムからの番号 0083 のメッセージであることを示しています。

画面のメッセージ行に表示されているか、またはメッセージ待ち行列に入っているメッセージについて情報の続きを入手したい場合には、次のようにします。

1. カーソルをメッセージと同じ行に移動してください。
2. ヘルプ・キーを押します。「追加のメッセージ情報」画面が表示されます。

```
追加のメッセージ情報

 メッセージ ID  . . . . :   CPD6A64       重大度 . . . . . . . . . :   30
 メッセージ・タイプ . . :    診断
 送信日付 . . . . . . . :   03/29/92      送信時刻 . . . . . . . . :   13:49:06

  メッセージ . . :    指定したメニューの選択項目が正しくない。

  原因－－指定された選択項目は，次のいずれかの理由により正しくありません。
     -- 選択された番号が無効であった。
     -- メニュー・オプション以外のものがオプション行に入力された。
  回復手順－－正しいオプションを選択して，実行キーを押すか，あるいは HELP キー
       をもう一度押してください。



終わり
続行するには、実行キーを押してください。

F3= 終了    F6= 印刷    F9= メッセージ詳細の表示
F10= ジョブ・ログ中のメッセージの表示    F12= 取り消し    F21= 援助レベルの選択
```

メッセージ識別コードとそれが置かれているライブラリーが分かっていれば、画面に示されていないメッセージの詳細を知ることができます。この情報を得るには、次のようなメッセージ記述表示 (DSPMSGD) コマンドを入力します。

```
DSPMSGD RANGE(SQL0204) MSGF(QSYS/QSQLMSG)
```

このコマンドは、メッセージに関する次のような情報を選択するための画面を表示します。

- メッセージ・テキスト
- フィールド・データ
- メッセージ属性
- 上記のすべて

このテキストは、「追加のメッセージ情報」画面に表示されるのと同じメッセージおよびメッセージ・ヘルプ・テキストです。フィールド・データは、そのメッセージとその属性に定義されているすべての置換変数のリストです。メッセージの属性は、メッセージの重大度、ログ記録、レベルの値 (定義されている場合)

と、デフォルト・プログラム、デフォルト応答、およびダンプ・パラメーターです。この情報を使って、メッセージが出されたときにユーザーが何をしていたかを判断することができます。

メッセージ・タイプ

「追加のメッセージ情報」画面に、メッセージのメッセージ・タイプと重大度コードが表示されます。 表 9 は、さまざまな iSeries メッセージとそれに関連した重大度コードを示しています。

表 9. メッセージ重大度コード

メッセージ・タイプ	重大度コード
通知メッセージ。通知だけを目的としており、応答は必要ありません。このメッセージは、機能が進行中であるか、または正常に完了したことを示すことがあります。	00
警告。エラー条件の可能性が存在します。プログラムは、欠落データを補うなどのデフォルト処置をとった可能性があります。操作の結果は正常と想定されます。	10
エラー。エラーが見つかりましたが、自動回復手順が行われたと考えられる種類のエラーです。処理は続行されました。誤ったデータの代わりにデフォルト値がとられた可能性があります。操作の結果は正しくない可能性があります。機能は完了しなかったかもしれません。たとえば、リスト内の項目には、正しく実行されたものと、そうでないものとがあるかもしれません。	20
重大エラー。見つかったエラーは、自動回復手順を行うには重大すぎ、デフォルト値はとられません。エラーがソース・データ内にあった場合、データ・レコード全体がスキップされました。プログラム中にエラーが起きた場合、プログラムの異常終了 (重大度 40) に至ります。操作の結果は正しくありません。	30
重大エラー: プログラムまたは機能の異常終了。プログラムが、正しくないデータを処理することができなかったか、またはユーザーが取り消したために、操作は終了しました。	40
ジョブまたはプログラムの異常終了。ジョブが開始されなかったかまたは正常に開始しなかったか、ジョブ・レベルの機能が所定どおりに行われなかったか、またはジョブが取り消されたかのいずれかです。	50
システム状況。システム操作員メッセージ待ち行列に対してのみ出されます。これは、装置、サブシステム、またはシステムに関する状況または警告を示します。	60
装置保全性。システム操作員メッセージ待ち行列に対してのみ出され、装置が正しく作動していないか、または作動不能になっていることを示します。	70
システム警報およびユーザー・メッセージ。システムをただちに停止するほど重大ではないけれども、予防措置を講じない限り重大度が増す可能性のある事態が存在しています。	80
システム保全性。システム操作員メッセージ待ち行列に対してのみ出されます。サブシステムまたはシステムが作動できない事態があることを示しています。	90

表 9. メッセージ重大度コード (続き)

メッセージ・タイプ	重大度コード
処置。応答の入力またはプリンター用紙の交換などの、何らかの手作業による処置が必要です。	99

分散リレーショナル・データベース・メッセージ

アプリケーション・サーバー (AS) または**アプリケーション・リクエスター (AR)** でエラー・メッセージが出された場合、障害の理由を示すサーバー・メッセージがジョブ・ログにログ記録されます。ジョブ・ログを使用し、AS 上のジョブ・ログを見つける方法の詳細は、 100 ページの『分散リレーショナル・データベースのジョブ・ログによる要求情報の追跡』を参照してください。

DB2 Universal Database for iSeries プログラムがサポートする SQL ステートメントからの SQLCODE ごとに、サーバー・メッセージが存在します。 このメッセージは、プリコンパイラー・リスト内、対話式 SQL 上、またはデバッグ・モードでの実行時にジョブ・ログに示されます。ただし、iSeries server 以外の AS で作業するときは、次のような場合にはエラー条件ごとに固有のメッセージがあるとは限りません。

- エラーが iSeries server では使用されない機能に関連している場合。

  たとえば、DB2 UDB for iSeries では特殊レジスターの CURRENT SQLID をサポートしていないので、 SQLCODE -411 (SQLSTATE 56040)「CURRENT SQLID はリモート・オブジェクトを参照するステートメントでは使用できない」というメッセージはありません。
- エラーがプロダクトに固有のもので、DB2 UDB for iSeries では決して起こらない場合。

  DB2 UDB for iSeries では、SQLCODE -925 (SQLSTATE 56021) 「IMS™ または CICS® 環境では SQL コミットまたはロールバックは無効です」が出されることはありません。

対応するメッセージをもたない SQLCODE の場合、未認識の SQLCODE、SQLSTATE、およびトークンを示した一般メッセージと、そのメッセージを生成した AS のリレーショナル・データベースの名前が戻されます。個々の条件を判別したりトークンの解釈方法を確かめたい場合は、接続先の AS のリリースに対応する製品資料を調べてください。 SQLCODE の詳細は、 164 ページの『SQLCODE および SQLSTATE』を参照してください。

CPx3E00 から CPx3EFF および CPI9100 から CPI91FF の範囲のメッセージは、分散リレーショナル・データベース・サーバー・メッセージに使われます。 以下のリストは、すべてが網羅されているわけではありませんが、 iSeries server の分散データベース・ジョブ・ログに、比較的よく出されるサーバー・メッセージを示しています。 分散リレーショナル・データベースの SQL メッセージのリストについては、『SQL プログラミング 概念』を参照してください。

表 10. 分散リレーショナル・データベースのメッセージ

MSG 識別コード	説明
CPA3E01	*LOCAL ディレクトリー項目を除去すると、構成データが失われることがある。
CPC3EC5	リレーショナル・データベースのディレクトリー項目の一部のパラメーターが無視された。
CPD3E30	指定されたリモート・ネットワーク ID の値が矛盾している。
CPD3E35	リモート・ロケーション・タイプ &2 にはこのリモート・ロケーションは正しくない。
CPD3E36	ポート識別コードが正しくない。
CPD3E38	リモート・ロケーションがそのタイプと矛盾している。
CPD3E39	パラメーター &2 に値 &3 を使用することはできない。
CPD3E3B	サーバー &2 のサーバー許可情報を検索中にエラーが起こった。

表 10. 分散リレーショナル・データベースのメッセージ (続き)

MSG 識別コード	説明
CPD3ECA	リレーショナル・データベース・ディレクトリーの操作が完了していない可能性がある。
CPD3E01	混合バイトまたは図形 CCSID はサポートされていない。
CPD3E03	ローカル・リレーショナル・データベース名がディレクトリーにない。
CPD3E05	DDM 会話パスが見つからない。
CPD3E31	DDM TCP/IP サーバーが活動状態になっていない。
CPD3E32	DDM TCP/IP サーバーの終了時にエラーが起こった。
CPD3E33	理由コード &1 で DDM TCP/IP サーバーにエラーが起こった。
CPD3E34	&1 で DDM TCP/IP 通信エラーが起こった。
CPD3E37	理由コード &1 で IP アドレスに対するホスト名のマップが失敗した。
CPF3E30	DDM TCP/IP サーバーの開始中にエラーが起こった。
CPF3E31	DDM TCP/IP サーバーを開始することができない。
CPF3EC6	DDM TCP/IP 属性変更が正しく実行されなかった。
CPF3EC9	RDB 割り込みの範囲メッセージ
CPF3E0A	リソースの限界を超えた。
CPF3E0B	照会がオープンされない
CPF3E0C	データ・タイプ ID の限界に達した。
CPF3E0D	リレーショナル・データベース終了要求を使用することはできない。
CPF3E01	この DDM パラメーター値はサポートされていない。
CPF3E02	アプリケーション・リクエスターが要求された機能をサポートすることができない。
CPF3E04	1 バイト CCSID がサポートされていない。
CPF3E05	指定された SQL パッケージ・バインド・プロセスが活動状態でない。
CPF3E06	リレーショナル・データベース &1 が見つからない。
CPF3E07	パッケージ・バインド・プロセスが活動状態の時には、このコマンドは正しくない。
CPF3E08	照会オープン障害
CPF3E09	パッケージ・バインド・プロセスの開始の試みが正常に実行されなかった。
CPF3E10	混合バイトまたは図形データはサポートされていない。
CPF3E12	COMMIT または ROLLBACK CL コマンドを使用することはできない。
CPF3E13	コミットメント制御操作が正常に実行されなかった。
CPF3E14	リレーショナル・データベース終了要求が正常に実行されなかった。
CPF3E16	リレーショナル・データベース &1 のアクセスは認可されていない。
CPF3E17	リレーショナル・データベース終了要求が進行中である。
CPF3E18	リモート・データベースでの COMMIT または ROLLBACK が正常に実行されなかった。
CPF3E19	コミットメント制御操作が正常に実行されなかった。
CPF3E20	DDM 会話パスが見つからない。
CPF3E21	リレーショナル・データベース終了要求が正常に実行されなかった。
CPF3E22	コミットでアプリケーション・サーバーのロールバックが起こった。
CPF3E23	DDM データ・ストリームが会話処理機能に違反している。
CPF3E30	DDM TCP/IP サーバーの開始中にエラーが起こった。
CPF3E32	クライアント要求の処理中に DDM TCP/IP サーバー・エラーが起こった。
CPF3E80	DDM データ・ストリームで構文エラーが検出された。

表 10. 分散リレーショナル・データベースのメッセージ (続き)

MSG 識別コード	説明
CPF3E81	受け取ったデータ記述子が正しくない。
CPF3E82	リレーショナル・データベースがすでにアクセスされた。
CPF3E83	データのミスマッチ・エラー。
CPF3E84	DDM 会話式プロトコル・エラーが検出された。
CPF3E85	リレーショナル・データベース &4 がアクセスされなかった。
CPF3E86	分散データベースの処理時にエラーが起こった。
CPF3E87	永続エラー条件が検出された。
CPF3E88	リモート・システムで SQL カーソルがすでにオープンされていた。
CPF3E89	照会がオープンされない
CPF3E99	リモート・ユーザーによってリレーショナル・データベース要求が終了された。
CPI9150	リモート・データベース・ジョブが開始された。
CPI9152	ターゲット DDM ジョブがアプリケーション・リクエスター (AR) によって開始された。
CPI9160	データベース接続が TCP/IP またはローカル・ソケット経由で開始された。
CPI9161	データベース TCP/IP またはローカル・ソケット接続が終了した。
CPI9162	DDM 接続を処理するために割り当てられたターゲット・ジョブは TCP/IP 経由でソース・システムによって開始された。
CPI9190	分散データベースでの許可の障害。
CPI3E01	ローカル・リレーショナル・データベースが &1 によってアクセスされた。
CPI3E02	ローカル・リレーショナル・データベース・アクセスが終了した。
CPI3E04	リレーショナル・データベース &1 に対する接続が終了した。
CPI3E30	DDM TCP/IP サーバーはすでに活動状態である。
CPI3E31	DDM TCP/IP サーバーは機密保護機構タイプをサポートしない。
CPI3E32	DDM サーバー・ジョブが正常に投入された。
CPI3E33	DDM サーバーは正常に終了した。
CPI3E34	DDM ジョブ xxxx は hh:mm:ss の mm/dd/yy にユーザー yyy をサービス中。 (これは、QRWOPTIONS を使って抑制できます。)
CPI3E35	QSYS のプログラム QRWTSRVR 事前開始ジョブ項目が存在していない。
CPI3E36	リレーショナル・データベース &1 への接続が終了した。
SQ30082	分散データベースの接続の試みで認可が正常に実行されない。
SQL7992	TCP/IP 経由のリレーショナル・データベース &1 に対する CONNECT が完了した。
SQL7993	すでにリレーショナル・データベース &1 に接続されている。

APPC のプログラム開始要求障害の処理

アプリケーション・サーバー (AS) 上の OS/400 サブシステムは、プログラム開始要求を受け取ると、その要求と一緒に送られてきた情報に基づいてジョブの開始を試みます。アプリケーション・サーバー (AS) に対する**アプリケーション・リクエスター (AR)** ユーザーの権限、要求されたデータベースの存在、およびその他多くの項目が検査されます。

AS サブシステムは、ジョブを開始できないと判断した場合 (たとえば、 AS 上にユーザー・プロファイルが存在しない、ユーザー・プロファイルは存在するが使用不可になっている、または AS 上の要求対象のオブジェクトに対してユーザーが正しい許可を受けていないなど)、メッセージ CPF1269 を QSYSMSG メ

ッセージ待ち行列に送ります (ただし、 QSYSMSG が存在しなければ QSYSOPR に送ります)。 CPF1269 メッセージには、理由コードが 2 つあります (理由コードの片方が 0 であることがあり、その場合は無視できます)。

ゼロ以外の理由コードは、プログラム開始要求が拒否された理由を示しています。リモート・ジョブは AS で開始されたことになっているので、メッセージと理由コードは、アプリケーション・リクエスター (AR) ではなく アプリケーション・サーバー (AS) で示されます。 AR のユーザーには、プログラム開始要求が失敗したことだけが知らされ、失敗した理由は知らされません。 AR のユーザーは、その要求が失敗した理由を知るには、 AS のシステム・オペレーターに問い合わせるか、または AS へのディスプレイ・パススルーを使用しなければなりません。

すべての理由コードとその意味の解説は、 *ICF Programming* に記載されています。

TCP/IP の接続要求障害の処理

TCP/IP を使うよう構成された DRDA サーバーでの接続要求が失敗した場合、 DDM TCP/IP サーバーの未開始、許可エラーの発生、またはマシンの未稼働がその主な原因です。

サーバーの未開始または正しくないポート ID

DDM TCP/IP サーバーが開始されていない場合に出されるエラー・メッセージは次のような CPE3425 です。

リモート・ホストが接続操作を拒絶した。

リレーショナル・データベース・ディレクトリー項目の追加 (ADDRDBDIRE) またはリレーショナル・データベース・ディレクトリー項目の変更 (CHGRDBDIRE) コマンドに誤ったポートを指定した場合も、このメッセージが出されます。 DB2 UDB for iSeries サーバーの場合、ポートは通常は *DRDA (446 の DRDA ウェルノウン・ポート) でなければなりません。ただし、IPSec と一緒に使用するようポート 447 を構成している場合、機密データの送信にはそのポートを使うのがよいかもしれません。 Secure Socket Layer (SSL) をサポートする DRDA クライアントを使用する場合、サーバー上のポート 448 に接続しなければなりません。

リモート・サーバーで DDM サーバーを開始するには、 TCP/IP サーバーの開始 (STRTCPSVR) *DDM コマンドを実行します。 DDM TCP/IP 属性の変更 (CHGDDMTCPA) AUTOSTART(*YES) コマンドを実行すれば、 TCP/IP の開始のたびにこのサーバーを開始するよう要求することができます。

DRDA 接続許可の障害

許可障害の場合に出されるエラー・メッセージは次のような SQ30082 です。

分散データベースの接続の試行の際の許可の障害。

このメッセージの原因のセクションに、理由コードと、考えられる理由コードの意味のリストが示されます。理由コード 17 は、サポートされていないセキュリティー・メカニズム (SECMEC) があることを意味します。

DB2 UDB for iSeries は、iSeries アプリケーション・リクエスター (AR) が使用できるいくつかの DRDA SEMEC を実装します。

- ユーザー ID のみ
- ユーザー ID とパスワード
- 暗号化パスワード・セキュリティー・メカニズム (V4R5 以降)

- Kerberos (V5R2)

暗号化パスワードが送られるのは、接続の開始時にパスワードが使用可能になっている場合のみです。

iSeries server のデフォルトの SECMEC では、パスワードとともにユーザー ID が必要です。**アプリケーション・リクエスター (AR)** が、デフォルトのセキュリティー構成を使って、パスワードなしでユーザー ID をサーバーに送ると、理由コード 17 付きの SQ30082 が出されます。

サポートされていない SECMEC の障害の解決法は次のとおりです。

- DDM TCP/IP 属性の変更 (CHGDDMTCPA) PWDRQD(*NO) コマンドを使用して、サーバーでユーザー ID だけで SEMEC を使えるようにする。
- サーバーで PWDRQD(*YES) が有効にされている場合には、少なくとも、接続要求で平文のパスワードを送信する。
- サーバーで PWDRQD(*ENCRYPTED) が有効にされている場合に、暗号化パスワードを送信する。
- サーバーで RWDRQD (*KERBEROS) が有効にされている場合に、クライアントで Kerberos を使用する。

パスワードを送るには、USER/USING 形式の SQL CONNECT ステートメントを使用するか、または、サーバー許可項目の追加 (ADDSVRAUTE) コマンドを使って、接続を試みるときに使用するユーザー・プロファイル用のサーバー許可項目内に、リモート・ユーザー ID とサーバー許可を追加します。 V4R5 以降のシステムでは、暗号化されたパスワードの送信が自動的に試みられます。 V4R5 より前の iSeries server では、暗号化パスワードを送ることも、 V4R5 iSeries AR から送られてくるタイプの暗号化パスワードを復号することもできないことに注意してください。

リモート・パスワードをサーバーの許可項目に保管するには、システム値 QRETSVRSEC (サーバーのセキュリティー・データの保存) を 1 に設定する必要があることに注意してください。

重要: DRDA で使用する サーバー許可項目の追加 (ADDSVRAUTE) コマンド上の RDB 名を**大文字**で入力しなければなりません。そうしないと、接続処理中にその名前は認識されず、許可項目内の情報は使われなくなります。

サーバーが利用不能

リモート・サーバーが立ち上がって作動していないか、またはアプリケーション・ソース (AS) 用の IP アドレスを RDB ディレクトリーに誤って入力した場合、次のような CPE3447 メッセージが出されます。

リモート・ホストがタイムアウト期間内に応答しなかった。

通常、このメッセージが出されるまでに、数分の遅延があります。その時点で、何かがハングアップしたか、またはループが起きたように見えることがあります。

対話式 SQL に特有の接続障害

対話式 SQL から CONNECT ステートメントを実行しているときに、場合によっては、 SQ30080 一般メッセージ (分散データベースの処理時に通信エラーが起こった) が出されます。エラーの詳細を知るには、対話式 SQL を終了してジョブ・ログを調べる必要があります。

単一フェーズ・コミット機能しかないサーバーに初めて接続した (どのレベルのコミットメント制御であっても) ときに、メッセージ SQL7020 (SQL パッケージの作成が正常に実行されなかった) が出された場合は、リモート・サーバーに読み取り専用サーバーとしてアクセスしたのに、それを更新して SQL パッケージを作成しようとしたことが原因と考えられます。

ジョブ・ログ内のメッセージを調べれば、それを確認できます。 それを解決するには、RELEASE ALL と COMMIT を行って、まずすべての接続を取り除いてから接続すれば、接続は更新可能になります。 82 ページの『対話式 SQL (ISQL) での SQL パッケージのセットアップ』を参照してください。

サーバーでの事前開始ジョブが不十分

QSYSWRK サブシステムの QRWTSRVR 事前開始ジョブ項目において、 TCP/IP サーバーに関連付けられる事前開始ジョブの数が限定されている場合に、 1 つの接続ですべての事前開始ジョブが使われていると、新たに接続を試みても、次のようなメッセージとともに失敗します。

CPE3426
リモート・ソケットとの接続はそのソケットによってリセットされた。

CPD3E34
() で DDM TCP/IP 通信エラーが起こった。

サーバーでこのような問題が起きないようにするには、 QTWTSRVR 項目用の 事前開始ジョブ項目変更 (CHGPJE) コマンドの MAXJOBS パラメーターをもっと大きな数または *NOMAX に設定するか、ADLJOBS パラメーターを 0 以外の値に設定します。

アプリケーションの問題

アプリケーションでの問題を処理するには、実動段階に入る前が最も適した時期です。しかし、アプリケーションが一般の使用に供されるとき、そのアプリケーションで起こり得る状態をすべて予測することは不可能です。**アプリケーション・リクエスター (AR)** または**アプリケーション・サーバー (AS)** のジョブ・ログに、パッケージに障害が起きたことが示されていたり、プログラムまたはパッケージのリストに、その障害の理由が示されていたりすることがあります。 SQL コンパイラーには診断テストが用意されています。そこでは、事前コンパイル・プロセスで生成された SQLCODE および SQLSTATE が診断リスト内に示されます。

統合言語環境 (ILE*) の事前コンパイルでは、任意で OPTION(*XREF) および OUTPUT(*PRINT) を指定して、事前コンパイル・ソースと相互参照表を印刷することができます。 ILE 以外の事前コンパイルでは、SQL プログラムの作成 (CRTSQL*xxx*) コマンドの OPTIONS パラメーターに任意で *SOURCE および *XREF を指定して、事前コンパイルおよび相互参照を印刷することができます。

リスト

162 ページの図 17 に示されている SQL プログラムの作成 (CRTSQL*xxx*) コマンドでのリストの作成では、次のような情報が提供されます。

- 事前コンパイル・コマンドのパラメーターに提供される値
- ソース・プログラム
- 識別コード相互参照
- 事前コンパイルの結果のメッセージ

事前コンパイラーのリスト

注: xv ページの『コードの特記事項情報』に重要なリーガル情報がありますので、お読みください。

```
5763ST1 V3R1M0 940909            SQL ILE C オブジェクトの作成 UPDATEPGM 04/19/94 14:30:10   ページ   1
ソース仕様タイプ..............C
オブジェクト名.................TST/UPDATEPGM
ソース・ファイル..............*LIBL/QCSRC
メンバー......................*OBJ
オプション....................*XREF
プログラムの印刷..............*PRINT
ターゲット・リリース..........*CURRENT
INCLUDE ファイル..............*LIBL/*SRCFILE
コミット......................*CHG
データのコピー可能............*YES
SQL カーソルのクローズ.........*ENDACTGRP
ブロック化可能.................*READ
PREPARE 遅延..................*NO
生成レベル....................10
マージン......................*SRCFILE
印刷装置ファイル..............*LIBL/QSYSPRT
日付の形式....................*JOB
日付区切り記号................*JOB
時刻の形式....................*HMS
時刻区切り記号................*JOB
置き換え......................*YES
リレーショナル・データベース...RCHASLKM
ユーザー .....................*CURRENT
RDB 接続方式..................*DUW
省略時のコレクション..........*NONE
パッケージ名..................*OBJLIB/*OBJ
作成オブジェクト・タイプ.......*PGM
デバッグ・ビュー..............*NONE
動的ユーザー・プロファイル.....*USER
ソート順序....................*JOB
言語 ID.......................*JOB
IBM SQL フラグづけ............*NOFLAG
ANS フラグ付け................*NONE
テキスト......................*SRCMBRTXT
ソース・ファイルの CCSID.......37
ジョブの CCSID.................65535
04/19/94  14:25:33 にソース・メンバーが変更された。
5763ST1 V3R1M0 940909            SQL ILE C オブジェクトの作成  UPDATEPGM 04/19/94 14:30:10   ページ   2
レコード*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 SEQNBR  最終変更
1   /*********************************************************************/               100
2   /*  This program is called to update the DEPTCODE of file RWDS/DPT1 */               200
3   /* to NULL.  This is run once a month to clear out the old          */               300
4   /* data.                                                            */               400
5   /*                                                                  */               500
6   /*  NOTE: Because this program was compiled with an RDB name, it is */               600
7   /* not necessary to do a connect, as an implicit connect will take  */               700
8   /* place when the program is called.                                */               800
9   /*********************************************************************/               900
10   #include <stdio.h>                                                            1000
11   #include <stdlib.h>                                                           1100
12   exec sql include sqlca;                                                        1200
13                                                                                  1300
14   main()                                                                         1400
15     {                                                                            1500
16        /*  Just update RWDS/DPT1, setting deptcode = NULL         */             1600
17        exec sql update RWDS/DPT1                                                 1700
18               set deptcode = NULL;                                               1800
19     }                                                                            1900
* * * * *  ソ ー ス の 終 わ り  * * * * *
```

図 17. 事前コンパイラーによるリスト (1/2)

```
5763ST1 V3R1M0 940909            SQL ILE C オブジェクトの作成 UPDATEPGM 04/19/94 14:30:10   ページ     3
相互参照
データ名                         定義     参照
DEPTCODE                         ****     COLUMN
18
DPT1                             ****     TABLE IN RWDS
17
RWDS                             ****     COLLECTION
17
5763ST1 V3R1M0 940909            SQL ILE C オブジェクトの作成 UPDATEPGM 04/19/94 14:30:10   ページ     4
                                  診断メッセージ
MSG ID  SEV  レコード  テキスト
SQL0088   0       17  桁 15 UPDATE はテーブル全体に適用される。
SQL1103  10       17  RWDS のファイル DPT1 のフィールド定義が見つからない。
                                 メッセージの要約
合計    通知    警告    エラー    重大度    端末装置
2     1       1       0       0         0
ソースに 10 レベルの重大度エラーが見つかった。
19 ソース・レコードが処理された。
* * * * *  リ ス ト の 終 わ り  * * * * *
```

図 17. 事前コンパイラーによるリスト (2/2)

CRTSQLPKG リスト

図 18 に示されている構造化照会言語パッケージの作成 (CRTSQLPKG) コマンドでの作成リストでは、次のような 2 つのタイプの情報が提供されます。

- コマンドのパラメーターに使用される値
- エラーのステートメント (ある場合)
- 構造化照会言語パッケージの作成 (CRTSQLPKG) コマンドを実行した結果のメッセージ

```
5763SS1 V3R1M0 940909               Create SQL package 04/19/94 14:30:31   Page   1
レコード*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 SEQNBR  最終変更
プログラム名.................TST/UPDATEPGM
リレーショナル・データベース..*PGM
ユーザー .....................*CURRENT
置き換え.....................*YES
省略時のコレクション.........*PGM
生成レベル...................10
印刷装置ファイル.............*LIBL/QSYSPRT
オブジェクト・タイプ.........*PGM
モジュール・リスト...........*ALL
テキスト.....................*PGMTXT
ソース・ファイル.............TST/QCSRC
メンバー.....................UPDATEPGM
```

図 18. CRTSQLPKG によるリスト (1/2)

```
5763SS1 V3R1M0 940909              Create SQL package 04/19/94 14:30:31   Page   2
レコード*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 SEQNBR  最終変更
17   UPDATE RWDS/DPT1 SET deptcode = NULL
診断メッセージ
MSG ID  SEV  レコード  テキスト
SQL0204  10        17  桁 17 RWDS のタイプ *FILE の DPT1 が見つからない。
SQL5057               TST の SQL パッケージ UPDATEPGM がモジュール UPDATEPGM から KC000 に作成された。
メッセージの要約
合計     通知     警告     エラー     重大度     端末装置
1        0        1        0          0          0
ソースに 10 レベルの重大度エラーが見つかった。
* * * * *  リ ス ト の 終 わ り  * * * * *
```

図 18. CRTSQLPKG によるリスト (2/2)

SQLCODE および SQLSTATE

SQL を使用するプログラム・インターフェースは、エラーが発生したとき、エラー情報をアプリケーション・プログラムに戻します。 SQLSTATE とそれに対応する SQLCODE は、 SQL 通信域 (SQLCA) か SQL 診断域にあるアプリケーション・プログラムに戻されます。 SQLCA は、アプリケーションによって提供されたスペースの制御ブロックにある変数の集まりであり、これは、データベース管理システムによって、最後に実行された SQL ステートメントに関する情報で更新されます。 SQL 診断域は、データベース・マネージャーによって提供されたスペースにある、もっと複雑なストレージ域で、最後に更新された SQL ステートメントに関するより広範な情報を通信するように設計されています。

SQL エラーが検出されると、そのエラーの性質は、SQLSTATE と呼ばれる 5 文字のグローバル変数によって識別されます。また、SQLSTATE に加えて、整数 SQLCODE も使用できます。ただし、SQLCODE は、現行の 4 種類の IBM リレーショナル・データベース製品において、同じエラー条件に対して同じ戻りコードを戻すわけではありません。 SQLSTATE は、アプリケーションが接続する DB2 製品が何であるかに関係なく、アプリケーション・プログラムが特定のエラー条件やエラーのクラスをテストできるように、設計されています。

ステートメントの処理時に SQL がハード・エラーを検出すると、 SQLCODE は負の数 (たとえば、SQLCODE -204) になります。ステートメントの処理時に SQL が、例外ではあるが有効な条件 (警告) を検出すると、 SQLCODE は正の数 (たとえば、SQLCODE +100) になります。ステートメントの処理時に SQL が何もエラーや例外条件を検出しないと、 SQLCODE は 0 になります。どの DB2 Universal Database for iSeries SQLCODE にも、対応するメッセージが、ライブラリー QSYS のメッセージ・ファイル QSQLMSG 内にあります。 たとえば、SQLCODE -204 は、メッセージ ID SQL0204 として記録されます。

戻されるエラー情報は貴重な問題診断ツールであるので、戻された SQLCA や SQL 診断域に入っている特定の情報を表示するのに必要な指示を、アプリケーション・プログラムに組み込んでおくとよいでしょう。また、この後で扱うメッセージ・トークンも、問題診断に非常に有用です。

- SQLSTATE

  戻りコード。

- SQLCODE (SQLCA) または DB2_RETURNED_SQLCODE (SQL 診断域)

  戻りコード。

- SQLERRD(3) (SQLCA) または ROW_COUNT (SQL 診断域)

  SQL によって更新、挿入、または削除された行数。

SQLCA および SQL 診断域に関する詳細は、『SQL リファレンス』トピックの SQLCA、SQL 診断域、および SQLDA 制御ブロックに関する情報を参照してください。

『SQL メッセージおよびコード』のトピックに、各 SQLCODE とそれに関連付けられているメッセージ ID、SQLSTATE、およびメッセージのテキストのリストがあります。 メッセージ記述表示 (DSPMSGD) コマンドを使って、すべてのメッセージをオンラインで見ることができます。

分散リレーショナル・データベースの SQLCODE と SQLSTATE

以下のリストは、分散リレーショナル・データベース処理に関連した一部の共通の SQLCODE と SQLSTATE を示しています。すべての SQLCODE と SQLSTATE は、『SQL プログラミング 概念』を参照してください。ここでの簡単な SQLCODE (およびそれに関連した SQLSTATE) の説明では、メッセージ・データ・フィールドはアンパーサンド (&) と数字で示されています (たとえば &1)。このフィールドの置換テキストは、アプリケーション・プログラムが SQLCA を使用していれば SQLCA 内の SQLERRM に、アプリケーション・プログラムが SQL 診断域を使用していれば DB2_ORDINAL_TOKEN_n (n はトークン番号) に保管されます。 メッセージ記述表示 (DSPMSGD) コマンドを使って、すべての SQLCODE のさらに詳細な原因と回復情報を検索することができます。

表 11. SQLCODE および SQLSTATE

SQLCODE	SQLSTATE	説明
+100	02000	この SQLSTATE は、空テーブル上での SQL 操作のため、「No Data」という例外警告を報告する。 SQL UPDATE または SQL DELETE ステートメント内で識別されるゼロ行、あるいは SQL FETCH ステートメント内のカーソルが結果テーブルの最終行の後にあった。
+114	0A001	リレーショナル・データベース名 &1; が現行サーバー &2; と同じではない。
+304	01515	この SQLSTATE は、ホスト変数が検索値を保持するために十分な大きさでなかったために起こったホスト変数リストまたは構造に、 FETCH または SELECT での警告を報告する。 FETCH または SELECT が、指定された SELECT 項目にデータを戻さない。標識変数は、NULL 値の戻りを示すために -2 に設定される。処理が続行される。
+331	01520	文字変換が実行できない。
+335	01517	文字変換の結果が置換文字になった。
+551	01548	タイプ &3 の &2 の中のオブジェクト & に対して許可されていない。
+552	01542	&1; に対して許可されていない。
+595	01526	コミット・レベル &1; は &2; ロックにエスカレートされている。

表 11. SQLCODE および SQLSTATE (続き)

SQLCODE	SQLSTATE	説明
+802	01519	この SQLSTATE は、 SQL 選択ステートメントの SELECT リスト、SELECT、 UPDATE、または DELETE ステートメントの検索条件、あるいは UPDATE ステートメントの SET 文節内にあった SQL 算術関数または算術式の処理中に起こった算術例外警告を報告する。エラーのあるそれぞれの式ごとに、標識変数は、NULL 値の戻りを示すために -2 に設定される。関連したデータ変数は未変更のままである。処理が続行される。
+863	01539	リレーショナル・データベース &1; には SBCS 文字だけが許可されている。
+990	01587	この SQLSTATE は、2 フェーズ処理中に少なくとも 1 人の参加者からの保留応答または混合結果を報告する。
+30104	01615	バインド・オプションは無視される。
-114	42961	リレーショナル・データベース &1; は現行サーバー &2; と同じではない。
-144	58003	セクション番号 &1; は無効である。現在の上位セクション番号は &3;。理由は &2;。
-145	55005	異機種のアプリケーション・サーバー用の反復はサポートされていない。
-175	58028	コミット操作が失敗した。
-189	22522	コード化文字セット識別コード &1; は無効。
-191	22504	混合データ値は無効である。
-250	42718	ローカル・リレーショナル・データベースがディレクトリーに定義されていない。
-251	2E000 42602	リレーショナル・データベース名 &1; の中の文字が有効ではない。
-300	22024	NUL 終了入力ホスト変数またはパラメーターには NUL が含まれていなかった。
-302	22001	入力ホスト変数 &2; での変換エラー。
-330	22021	文字変換が実行できない。
-331	22021	文字変換が実行できない。
-332	57017	CCSID &1; と CCSID &2; との間の文字変換が無効である。

表 11. SQLCODE および SQLSTATE (続き)

SQLCODE	SQLSTATE	説明
-334	22524	文字変換の結果が切り捨てに終わった。
-351 -352	56084	選択リストまたは入力リスト内にサポートされない SQLTYPE が見つかった。
-426	2D528	アプリケーション実行環境に無効な操作。この SQLSTATE は、EXCSQLIMM または EXCSQLSTT を使用して、動的 COMMIT 制限環境で COMMIT を実行する試みを報告する。
-427	2D529	アプリケーション実行環境に無効な操作。
-501 -502 -507	24501	無効なカーソル状態のため、実行が失敗した。指定されたカーソルがオープンされない。
-510	42828	この SQLSTATE は、ブロック化プロトコルを使用して行を取り出しているカーソル上での DELETE WHERE CURRENT OF CURSOR または UPDATE WHERE CURRENT OF CURSOR への試みを報告する。
-525	51015	ステートメントがエラーである。
-551	42501	&2; タイプ *&3; 内のオブジェクト &1; に対して許可されていない。
-552	42502	&1; に対して許可されていない。
-683	42842	指定されたタイプの場合は、FOR DATA 文節または CCSID 文節が有効ではない。
-752	0A001	アプリケーション・プロセスは接続可能状態にない。理由コード &1;。
-802	22003 22012	数値が範囲外であり、ゼロによる除算は無効である。
-805	51002	&2; 内に SQL パッケージ &1; が見つからない。
-818	51003	整合性トークンが一致しない。
-842	08002	接続はすでに存在する。
-862	55029	ローカル・プログラムが、リモート・リレーショナル・データベースへの接続を試みた。
-871	54019	指定された CCSID 値が多過ぎる。
-900	08003	接続が存在しない。
-918	51021	アプリケーション・プロセスがロールバック操作を実行するまで、 SQL ステートメントを実行できない。

表 11. SQLCODE および SQLSTATE (続き)

SQLCODE	SQLSTATE	説明
-922	42505	この SQLSTATE は、アプリケーション・サーバーへの接続処理中にエンド・ユーザーを認証する際の障害を報告する。
-925 -926	2D521	現行環境では、SQL COMMIT または ROLLBACK ステートメントは無効である。
-950	42705	リレーショナル・データベース &1; はリレーショナル・ディレクトリー内にない。
-952	57014	ENDRDBRQS コマンドによって SQL ステートメントの処理が終了された。
-969	58033	アプリケーション・リクエスター・ドライバー・プログラムに要求を渡すときにエラーが起きた。
-7017	42971	コミットメント制御が受動側 DDM に対してすでに活動状態である。
-7018	42970	COMMIT HOLD または ROLLBACK HOLD が使用できない。
-7021	57043	アプリケーション・サーバー上でローカル・プログラムを実行しようとした。
-30000	58008	Distributed Relational Database Architecture (DRDA) プロトコルのエラー。
-30001	57042	分散 SQL プログラムの呼び出しは許可されていない。
-30020	58009	Distributed Relational Database Architecture (DRDA) プロトコルのエラー。
-30021	58010	リモート・サーバーでは分散リレーショナル・データベースはサポートされていない。
-30040	57012	リレーショナル・データベース &1; の DDM リソース &2; はアクセス不能になっている。
-30041	57013	リレーショナル・データベース &1; の DDM リソースはアクセス不能になっている。
-30050	58011	バインド・プロセスの進行中は DDM コマンド &1; は無効である。
-30051	58012	指定されたパッケージ名と一貫性トークンを使用するバインド・プロセスは活動中ではない。

表 11. SQLCODE および SQLSTATE (続き)

SQLCODE	SQLSTATE	説明
-30052	42932	プログラム準備の前提事項が正しくない。
-30053	42506	所有者 &1; 用のパッケージの作成を許可されていない。
-30060	08004	ユーザーはリレーショナル・データベース &1; に対する許可を受けていない。
-30061	08004	リレーショナル・データベース &1; が見つからない。
-30070	58014	分散データ管理機能 (DDM) コマンド &1; はサポートされていない。
-30071	58015	分散データ管理機能 (DDM) オブジェクト &1; はサポートされていない。
-30072	58016	分散データ管理機能 (DDM) パラメーター &1; はサポートされていない。
-30073	58017	分散データ管理機能 (DDM) パラメーター値 &1; はサポートされていない。
-30074	58018	分散データ管理機能 (DDM) 応答メッセージ &1; はサポートされていない。
-30080	08001	分散データベース処理中に通信エラーが起きた。
-30082	08001	分散データベースの接続の試行の際の許可の障害。
-30090	25000 2D528 2D529	読み取り専用アプリケーション・サーバーの場合は、変更要求は有効ではない。
-30104	56095	無効なバインド・オプション。この SQLSTATE は、1 つ以上のバインド・オプションがサーバーで無効だったことを報告する。バインド・オプションは終了する。エラーのある最初のバインド・オプションは SQLERRMC 内に報告される。
-30105	56096	バインド・オプションが矛盾している。バインド・オプションは終了する。矛盾のあるバインド・オプションは SQLERRMC 内に報告される。
Unrecognized by AR	58020	エラーまたは警告用の SQLSTATE 値が定義されていない。

システムおよび通信の問題

システムまたはその通信に関する問題が起こるとき、メッセージが生成されます。システムが検出した問題は、問題ログに自動的に入れられ、それらの問題を表示したり分析したりできます。

詳細は、『iSeries 問題ログ』を参照してください。

iSeries 問題ログ

システムが検出した問題は、問題ログに自動的に入れられます。また、ユーザーが検出した問題も、問題ログに入れることができます。任意のコマンド行から問題分析 (ANZPRB) コマンドを入力すれば、記録されている問題の分析をいつでも実行することができます。 このコマンドは、分析手順を順にたどって、問題関連の追加情報を問題ログに保管します。

問題ログを見るには、問題処理 (WRKPRB) コマンドを使います。 以下の画面は、問題ログの 2 つのビューを示しています。

```
問題の処理
システム:   KC000
位置指定 . . . . . .                問題 ID

オプションを入力して、実行キーを押してください。
2=変更      4=削除    5=明細の表示      6=明細の印刷    8=問題の処理
9=警報の処理   12=テキストの入力

OPT  問題 ID    状況           問題の記述
__   9114350131  READY          ユーザーが別の AS/400 でハードウェアの問題を検
__   9114326436  OPENED         (C R) システムが制御装置を呼び出すことができな
__   9114326281  OPENED         トークン・リングへのデータの挿入中に回線障害
__   9114324416  OPENED         装置障害。回復装置は停止しました。
__   9114324241  OPENED         (C R) システムが制御装置を呼び出すことができな
__   9114324238  OPENED         (C R) システムが制御装置を呼び出すことができな
__   9114324234  OPENED         (C R) システムが制御装置を呼び出すことができな
__   9114324231  OPENED         (C R) システムが制御装置を呼び出すことができな
__   9114324227  OPENED         (C R) システムが制御装置を呼び出すことができな
__   9114324224  OPENED         (C R) システムが制御装置を呼び出すことができな
__   9114324218  OPENED         (C R) システムが制御装置を呼び出すことができな
続く...
F3=終了        F5=最新表示    F6=リストの印刷      F11=日付および時刻の表示
F12=取り消し   F16=報告準備済みの問題    F24=キーの続き
```

最初のビューで F11 キーを押すと、次のような画面が表示されます。

```
問題の処理
システム:   KC000
位置指定 . . . . . .                問題 ID

オプションを入力して、実行キーを押してください。
2=変更      4=削除    5=明細の表示      6=明細の印刷    8=問題の処理
9=警報の処理    12=テキストの入力

OPT  問題 ID     日付      時刻      起点
__   9114350131  03/29/92  14:36:05  APPN.KC000
__   9114326436  03/29/92  07:41:59  APPN.KC000
__   9114326281  03/29/92  07:39:17  APPN.KC000
__   9114324416  03/29/92  07:06:42  APPN.KC000
__   9114324241  03/29/92  07:03:38  APPN.KC000
__   9114324238  03/29/92  07:03:35  APPN.KC000
__   9114324234  03/29/92  07:03:31  APPN.KC000
__   9114324231  03/29/92  07:03:27  APPN.KC000
__   9114324227  03/29/92  07:03:24  APPN.KC000
__   9114324224  03/29/92  07:03:20  APPN.KC000
__   9114324218  03/29/92  07:03:14  APPN.KC000
続く...
F3=終了    F5=最新表示    F6=リストの印刷    F11=資源情報の表示    F12=取り消し
F16=報告準備済みの印刷    F24=キーの続き
```

iSeries の問題ログ・サポートを使って、ローカル・サーバーに記録されているすべての問題のリストを表示することができます。 また、次に示すような個々の問題に関する詳細情報を表示することもできます。

- 問題が生じている装置の型式および製造番号
- 問題の日付および時刻
- 障害を起こした部品およびその位置
- 問題状況

問題ログを使って、問題を分析したり、問題を報告したり、行われたすべてのサービス活動を判別したりすることができます。

障害の報告のためのデータの入手

以下の項では、 iSeries server で分散リレーショナル・データベース内の問題を診断するときに印刷して利用できるようなデータについて説明します。 このデータは、OS/400 プログラムで作成されます。また、システム・オペレーター・メッセージとアプリケーション・プログラム (データも一緒に) を使って、問題を診断することもできます。

- ジョブ・ログの印刷
- TCP/IP サーバー事前開始ジョブからのジョブ・ログの検出
- 製品の活動記録ログの印刷
- ジョブ・トレース
- 通信トレース

ジョブ・ログの印刷

iSeries server のどのジョブにも、そのジョブで入力された要求に関連した情報の入ったジョブ・ログがあります。 **アプリケーション・リクエスター (AR)** で問題が起きたら、その問題を診断するのにジョブ・ログ内の情報が役立つことがあります。その情報を入手するには、次のようなコマンドを使ってそのユーザーをサインオフさせるのが 1 つの簡単な方法です。

```
SIGNOFF *LIST
```

このコマンドは、そのユーザーのジョブ・ログを印刷するか、または、印刷用の出力待ち行列にそのログを入れます。

ジョブ・ログを印刷する別の方法としては、アプリケーション・ジョブ記述に LOG(4 00 *SECLVL) を指定します。ジョブが終了すると、すべてのメッセージは、そのジョブ用のジョブ・ログに記録されます。出力待ち行列上でジョブ・ログを見つけ出してから、印刷手順を実行すれば、ジョブ・ログを印刷することができます。サーバー上でジョブとジョブ・ログを見つけ出す方法の詳細は、 100 ページの『分散リレーショナル・データベースのジョブ・ログによる要求情報の追跡』を参照してください。

アプリケーション・サーバー (AS) のジョブ・ログも、問題の診断に役立つことがあります。 AS ジョブのジョブ名を見つける方法の詳細は、 101 ページの『分散リレーショナル・データベース・ジョブの探索』を参照してください。

TCP/IP サーバー事前開始ジョブからのジョブ・ログの検出

DDM TCP/IP サーバーに関連した QRWTSRVR 事前開始ジョブの 1 つを利用していた接続が終了すると、その事前開始ジョブはリサイクルされて、別の接続で使われます。その場合、終了した接続に関連したジョブ・ログは消去されます。ただし、ある特定の場合、終了した接続に関連したジョブ・ログを消去する前に、ジョブ・ログはプリンター・ファイルにスプールされます。クライアント・ユーザー ID およびパスワードが正常に妥当性検査されなかった場合、ジョブ・ログはスプールされません。妥当性検査が正常に行われた場合、以下は、ジョブ・ログがスプールされる条件です。

- V5R1 以上で、サーバー・ジョブのメッセージ・ロギング・テキスト・レベルが *SECLVL または *MSG である場合
- 要求の処理中に、接続を終了させるような重大なエラーが発生したことをサーバー要求ハンドラーの経路指定プログラムが検出した場合。
- 事前開始ジョブがサービスされていた場合 (サービス・ジョブの開始 (STRSRVJOB) コマンドの使用による)。
- クライアントまたはサーバー上の QRWOPTIONS データ域が、サーバー・ジョブによって満たされたジョブ・ログ出力条件を指定した場合。詳しくは、『QRWOPTIONS データ域の使用法』のトピックを参照してください。

いくつかの理由で、ジョブ・ログ情報を入手したいと思われることでしょう。ジョブ・ログ情報は、エラーを診断するために明らかに役立ちます。パフォーマンス問題を分析するためにも役立ちます。たとえば、デバッグ下で実行時に生成される SQL 最適化プログラム・データを入手したい場合、手動でサービス・ジョブを開始してデバッグの開始 (STRDBG) コマンドを実行するか、またはジョブ・ログを保存するように QRWOPTIONS データ域内に 1 つ以上の適当なオプションを設定することができます。

接続フェーズ中に障害が起こるジョブのログは、上記で説明されているプロセスによって保管されません。そのプロセスによって保管されるジョブは、事前開始ジョブ ID の下には保管されません。ジョブ・ログを見つけ出すには、次のようなコマンドを実行します。

```
WRKJOB userid/QPRTJOB
```

ここで、*userid* は、**アプリケーション・サーバー (AS)** への接続で使われるユーザー ID です。このユーザー ID が不明の場合、 AS 上で ログの表示 (DSPLOG) コマンドを使って見つけることができます。

以下のようにパラメーターを使用することによって、不要なメッセージをフィルターで取り除くことができます。

```
DSPLOG PERIOD(('11:00')) MSGID(CPI3E34)
```

次のようなメッセージを探してください。ただし、このメッセージ (CPI3E34) を抑制するために QRWOPTIONS データ域が使用されている場合、このメッセージはヒストリー・ログ内には表示されないことに注意してください。

```
DDM job xxxx servicing user yyy on ddd at ttt.
```

製品の活動記録ログの印刷

iSeries server の製品活動記録ログは、マシン・チェック、装置エラー、およびテープとディスケットの統計のレコードです。 またこれには、各 FFDC ダンプの最初の 1000 バイトを含めた FFDC 情報も入っています。これらのエラーを検討すれば、問題の特性を判別できる可能性があります。

自分がサインオンしているサーバーの活動記録ログを印刷するには、次のようにします。

1. 任意のコマンド行にエラー・ログの印刷 (PRTERRLOG) コマンドを入力してから、 F4 キー (プロンプト) を押します。 「エラー・ログの印刷」画面が表示されます。
2. 印刷したいログ情報の種類のパラメーター値を入力してから、実行キーを押します。そのログ情報は、ジョブで指定されている出力待ち行列に送られます。
3. ジョブの処理 (WRKJOB) コマンドを入力します。 「ジョブの処理」画面が表示されます。
4. スプール・ファイルを処理するためのオプションを選択します。「ジョブ・スプール・ファイルの処理」画面が表示されます。
5. スプール・ファイル・リストの下端またはその近くで、作成したばかりのログ・ファイルを探します。
6. ログ・ファイルの次の「*Opt*」列に、印刷状況の処理オプションを入力します。「印刷状況の処理」画面が表示されます。
7. 「印刷状況の処理」画面で、状況変更オプションを使用して、ファイルの状況を変更し、印刷装置をファイルの印刷に指定してください。

ジョブ追跡

場合によっては、問題を追跡していってもプログラムを特定できないことがあります。このようなケースでは、トレース開始 (STRTRC) およびジョブ・トレース (TRCJOB) を使用して、 CL コマンドを含むモジュール・フローと OS/400 のデータ収集をトレースすることができます。これらのツールは、問題分析手順で問題について十分な情報が得られない場合に使用してください。分散データベース・アプリケーションの場合、これらのコマンドは、分散データベースの要求と応答のデータ・ストリームを収集するのに便利です。

TRCJOB

TRCJOB は、2 つの追跡ツールの中で古い方のツールです。各トレース・レコードは、生成されるたびに内部トレース記憶域に保管されます。トレースが終了したら、トレース・レコードをスプール印刷装置ファイル (QPSRVTRC) に書き込むことができますが、データベース出力ファイルに送ることもできます。

以下に、トレースのシナリオの例を示します。

```
TRCJOB SET(*ON) TRCTYPE(*ALL) MAXSTG(2000)
       TRCFULL(*WRAP) EXITPGM($SCFTRC)
CALL QCMD
TRCJOB SET(*OFF) OUTPUT(*PRINT)
WRKOUTQ output-queue-name
```

QPSRVTRC という名前のスプール・ファイルが示されます。このスプール・ファイルには、行ったトレースが入っています。ジョブのトレースの使用法の詳細は、 261 ページの『付録 C. ジョブのトレースと FFDC データの解釈』を参照してください。

STRTRC

トレースの実行には、トレース開始 (STRTRC) コマンドを使用することも可能です。 STRTRC は、TRCJOB に比べて柔軟性が高く、侵入の少ないツールです。このツールでは、複数のジョブにまたがって追跡を行い、モジュール・フローに関してより綿密に詳細を表示することができます。各トレース・レコードは、生成されるたび、セッション ID によって識別された内部トレース記憶域に保管されます。トレース終了 (ENDTRC) を使用してトレースが終了すると、トレース・レコードは、一連のデータベース・ファイルとしてユーザー指定のライブラリーに置かれます。こうしてライブラリーに置かれたファイルは、スプール印刷装置ファイル (QPSRVTRCJ) に書き込むこともできますし、 PRTTRC を発行してデータベース出力ファイルに送ることもできます。詳細は、『制御言語 (CL)』トピックの、『STRTRC』および『 ENDTRC』の項を参照してください。

以下に、トレースのシナリオの例を示します。

注: xv ページの『コードの特記事項情報』に重要なリーガル情報がありますので、お読みください。

```
STRTRC SSNID(DRDATRACE) JOB((*ALL/QUSER/QRWTSRVR)) MAXSTG(160000)
  TRCFULL(*STOPTRC)
```

障害のある DRDA のシナリオを実行します。

```
ENDTRC SSNID(DRDATRACE) DTALIB(TRACELIB)
PRTTRC DTAMBR(DRDATRACE) DTALIB(TRACELIB)
```

アプリケーション・サーバー・ジョブのジョブ・トレースを取得する必要がある場合、サーバーでサービス・ジョブを始動する必要があります。 180 ページの『アプリケーション・サーバー問題の診断のためのサービス・ジョブの開始』を参照してください。

通信トレース

DRDA を使って分散リレーショナル・データベースにアクセスしようとして CPF3Exx の範囲または CPF91xx の範囲のメッセージが出されたら、通信トレースを実行する必要があります。以下に、表示される可能性のあるこれらの範囲の共通メッセージを示します。

表 12. 通信トレース・メッセージ

MSG 識別コード	説明
CPF3E80	DDM データ・ストリームの構文エラー。
CPF91xx	DDM プロトコル・エラー。
CPF3E83	FD0:CA 記述子が無効。
CPF3E84	データのミスマッチ・エラー。

実行できる通信トレースは 2 通りあります。 1 つは、標準の通信トレースです。もう 1 つは TRCTCPAPP 機能です。 IPSec または保護ソケット・プロトコルにより暗号化されたデータ・ストリームがある場合に、 TRCTCPAPP は判別可能なトレースを提供します。これは、暗号化前と暗号化解除後のデータを取り込みます。ただし、暗号化されていないデータ・ストリームのトレースの場合にも十分に働きます。これは、LOOPBACK が使用されるシステム内の DRDA フローのトレースを行うために必要です。この後の項に、この 2 つのタイプのトレースの実行法が説明されています。

標準の通信トレース

通信トレース機能を使って、通信構成オブジェクトでのデータのトレースを開始したり停止したりすることができます。データのトレースの実行を完了した後、印刷または表示用にそのデータをフォーマットすることができます。印刷装置ファイルは、出力待ち行列の中でしか見ることはできません。

通信トレース・オプションは、システム保守ツール (SST) のもとで実行します。 SST を使うと、通信トレースの活動中に構成オブジェクトを使用することができます。分散データベース・ネットワークで使えるどの通信タイプのデータでも、トレースしてフォーマットすることができます。

サーバーに接続されている任意の画面から、iSeries 通信トレースを実行することができます。 *SERVICE の特殊権限 (SPCAUT) をもつユーザーはだれでも、iSeries server 上でトレースを実行することができます。通信トレースは、すべての回線速度をサポートします。通信制御装置で使えるプロトコルでの最大合計回線速度の詳細は、 V5R1 Supplemental Manual Web サイトにある、 *「Communications Management」*


を参照してください。

通信トレースは、次のような状況下で使用してください。

- 問題分析手順では、問題について十分な情報が得られない場合。
- プロトコル違反の問題があると疑われる場合。
- 回線ノイズが問題であると疑われる場合。
- システム・ネットワーク体系 (SNA) バインド問題があることをエラー・メッセージが示している場合。

通信トレースで生成されるデータを正しく解釈するには、使用する回線プロトコルの詳しい知識が必要です。 DRDA データ・ストリームの解釈についての詳細は、 262 ページの『例: RW トレース・データの分析』を参照してください。

可能な限り、回線をオンに構成変更する前に通信トレースを開始してください。そうすれば、オンに構成変更されたときの回線の最も正確なサンプルが得られます。

APPC トレースを実行してその出力を処理するには、どの回線、制御装置、および装置で実行しているかが分かっていなければなりません。その情報がそろっていない場合、 176 ページの『回線、制御装置、および装置記述の検索』を参照してください。

TCP/IP トレースの出力で不要なデータをフォーマットし、回避するには、ソースおよびアプリケーション・ソース (AS) の IP アドレスを指定することができます。その代わりに、単にポート番号を指定するだけで十分な場合もあり、より容易です。

次のコマンドを使って、通信トレースを開始、終了、印刷、および削除することができます。

通信トレースの開始 (STRCMNTRC) コマンド
指定された回線またはネットワーク・インターフェース記述の通信トレースを開始します。「トレースするバイト数 (Number of bytes to trace)」パラメーターの「先頭バイト (Beginning bytes)」の値に *MAX を指定します。通信トレースは、通信トレースの終了 (ENDCMNTRC) コマンドが実行されるまで続きます。

通信トレースの終了 (ENDCMNTRC) コマンド
指定された回線またはネットワーク・インターフェース記述で実行している通信トレースを終了します。

通信トレースの印刷 (PRTCMNTRC) コマンド
指定した回線またはネットワーク・インターフェース記述の通信トレース・データを、スプール・ファイルまたは出力ファイルに移動します。 SNA データのみのフォーマット・パラメーターには *YES を指定します。

通信トレースの削除 (DLTCMNTRC) コマンド
指定した回線またはネットワーク・インターフェース記述の通信トレースを削除します。

回線、制御装置、および装置記述の検索: アプリケーション・サーバー・ジョブの始動に使われる制御装置と装置を見つけるには、 構成状況処理 (WRKCFGSTS) コマンドを使います。 以下にその例を示します。

```
WRKCFGSTS CFGTYPE(*DEV)
          CFGD(*LOC)
          RMTLOCNAME(DB2ESYS)
```

RMTLOCNAME キーワードの値は、アプリケーション・サーバー名です。

構成状況処理 (WRKCFGSTS) コマンドは、指定されたサーバー名をリモート・ロケーション名としてもつすべての装置を表示します。一度に 1 つの装置しかオンに構成変更できないため、どの装置を使用中であるかが分かります。オプション 8 を使って装置記述を処理してから、オプション 5 でそれを表示します。接続された制御装置のフィールドに、制御装置の名前が示されます。 (WRKCFGSTS) コマンドを使って制御装置と装置の記述を処理することができます。 以下にその例を示します。

```
WRKCFGSTS CFGTYPE(*CTL)
          CFGD(PCXZZ1205)    /* workstation */
WRKCFGSTS CFGTYPE(*CTL)
          CFGD(LANSLKM)      /* AS/400 on token ring */
```

CFGD 値は装置記述名ですが、これは、この項の最初の例で装置記述から得たものです。

構成状況処理 (WRKCFGSTS) コマンドの出力には、通信トレースを処理するときに必要な回線記述の名前も入っています。オプション 8 を選んでからオプション 5 を選んで制御装置記述を表示した場合、活動中の交換回線パラメーターが回線記述の名前を表示します。 LAN リモート・アダプター・アドレスに、リモート・サーバーのトークンリング・アドレスが示されます。

回線名を見つけ出す別の方法では、回線記述の処理 (WRKLIND) コマンドを使って、サーバーのすべての回線記述を表示します。

TRCTCPAPP トレース

この機能が有効なのは、通信に TCP/IP を使用する場合だけです。

| このトレース・ツールの 1 つの使い方は、データが暗号化されている環境で伝送の平文を表示するというものです。トレース・データは、暗号化前に送信側でキャプチャーし、暗号化後に受信側でキャプチャーします。ただし、このトレース・ツールは、他の環境でも有用です。

TCP/IP アプリケーションのトレース (TRCTCPAPP) コマンドを使うには、 *SERVICE 特殊権限を受けたユーザー・プロファイルがなければなりません。このトレースを開始するには、次のように入力します。

```
TRCTCPAPP *DDM
```

トレースを特定のポートにだけ限定したい場合は (例えば、SSL 用のポート 448)、次の例のようにします。

```
TRCTCPAPP *DDM *ON RMTNETADR(*INET *N '255.255.255.255' 448)
```

追跡している通信が終了したら、次のようなコマンドを実行してから、処理結果のスプール・ファイルを調べます。

```
TRCTCPAPP *DDM *OFF
```

| 複数の接続をトレースする場合は、各 QRWTSRVR ジョブにスプール・ファイルを配置し、突き合わせる必要があります。スプール・ファイル名は QZBSTRC で、ジョブは QRWxxxxxx です。この xxxxxx の部分には、スプール・ファイルのユーザー・データにあるジョブ番号が入ります。

*DDM アプリケーションでの使用時の制約事項

*DDM アプリケーションで TCP/IP アプリケーションのトレース (TRCTCPAPP) コマンドを使用する場合、単一の送受信メッセージごとにトレースできるデータの最大量は、6000 バイトに限定されます。

TRCTCPAPP トレース・フォーマット設定

TCP/IP アプリケーションのトレース (TRCTCPAPP) コマンドは、 DRDA および DDM フローを読みやすい論理表記に解析するのに使用できます。このコマンドは、情報を ASCII 形式で表示することもできるため、異種環境でも役立ちます。この形式を要求する場合は、通信トレースの末尾に続けて以下を入力します。

```
TRCTCPAPP APP(*DDM) SET(*OFF) ARGLIST('lvl=2')
```

以下は、フォーマット設定されていないトレースの例です。ただし、この例は、本書の幅に合わせて編集してあります。

```
0080D00010001007A 200100162110D9C3 C8C1E2D5E3E24040
4040404040404040 *..}....:......RCHASNTS *
0006210F2407000D 002FD8E3C4E2D8D3 F4F0F0000C112ED8
E2D8F0F5F0F3F000 *..........QTDSQL400....QSQ05030.*
0A00350006119C00 2500062121241E00 062120241E0010D1
2A01000000000000 *.......................J........*
0000000000001621 35C1D7D7D54BD3D7 F0F6F6C1C2B9191C
F706F90005213BF1 *.........APPN.LP066AB...7.9....1*
```

以下は、同じトレースを TRCTCPAPP でフォーマット設定したものです。

```
-Datastream-------------------------------------------------------------
DATA:             (ASCII)           (EBCDIC)
0080D00010001007A  200100162110D9C3  .8'.....a...b.êã .0}....:......RC
C8C1E2D5E3E24040  4040404040404040   ç ë+èë.......... HASNTS
0006210F2407000D  002FD8E3C4E2D8D3  .Lb.f"....éèàëê< ..........QTDSQL
F4F0F0000C112ED8  E2D8F0F5F0F3F000  .......êëé...... 400....QSQ05030.
0A00350006119C00  2500062121241E00  C...L.ó...LbbfK. ......ä.........
062120241E0010D1  2A01000000000000  LbafK..¢k....... .......J........
0000000000001621  35C1D7D7D54BD3D7  ......¸.b. &&+.<& .........APPN.LP
F0F6F6C1C2B9191C  F706F90005213BF1  ... â¨.c.L...bB. 066AB¾..7.9....1
-Parsed----------------------------------------------------------------

RECV(AS) RQSDSS - Request Data Stream Structure

LL: 128 CORR: 0001 CHAINED: n CONT ON ERR: n SAME CORR FOR NEXT DSS: n

NM: ACCRDB - Access RDB

LL: 122 CP: 2001

NM: RDBNAM - Relational Database Name

LL: 22 CP: 2110

ASCII: êãç ë+èë..........

EBCDIC: RCHASNTS

NM: RDBACCCL - RDB Access Manager Class

LL: 6 CP: 210F

CODE POINT DATA: 2407

NAME: SQLAM - SQL Application Manager

NM: TYPDEFNAM - Data Type Definition Name
```

```
LL: 13 CP: 002F

ASCII: éèàëé<...

EBCDIC: QTDSQL400

NM: PRDID - Product-Specific Identifier

LL: 12 CP: 112E

DATA:                (ASCII)       (EBCDIC)

D8E2D8F0F5F0F3F0     éëé.....     QSQ05030

NM: TYPDEFOVR - TYPDEF Overrides

LL: 10 CP: 0035

NM: CCSIDSBC - CCSID for Single-Byte Characters

LL: 6 CP: 119C

DATA:      (ASCII)     (EBCDIC)

0025       ..           ..

NM:  STTDECDEL - Statement Decimal Delimiter

LL: 6 CP: 2121

CODE POINT DATA: 241E

NAME: DFTPKG - Package Default

NM: STTSTRDEL - Statement String Delimiter

LL: 6 CP: 2120

CODE POINT DATA: 241E

NAME: DFTPKG - Package Default

NM: SXXPRDDTA - Extended Product Data

LL: 16 CP: D12A

DATA:              (EBCDIC)

0100000000000000   00000000 ............

NM: CRRTKN - Correlation Token

LL: 22 CP: 2135

DATA:             (ASCII)            (EBCDIC)

C1D7D7D54BD3D7F0  F6F6C1C2B9191CF7  &&+.<&... â¨.c. APPN.LP066AB¾..7

06F9 L. .9

NM: TRGDFTRT - Target Default Value Return

LL: 5 CP: 213B

BOOLEAN: TRUE
```

第 1 障害データ検知 (FFDC) データの検索

注: FFDC データは、QSFWERRLOG システム値が *LOG に設定されていない限り生成されません。

以下に、iSeries server で FFDC データを見つけ出す方法のヒントを示します。 この情報が最も役に立つのは、**アプリケーション・サーバー (AS)** で FFDC データの出力の原因になる障害が発生した場合です。通常、**アプリケーション・リクエスター (AR)** の FFDC データは、アプリケーション・プログラムを実行しているジョブに関連したスプール・ファイル内で見つけることができます。

1. メッセージの表示 (DSPMSG) QSYSOPR コマンドを実行してから、 QSYSOPR メッセージ・ログ内で `Software problem detected in Qccxyyyy` というメッセージを探します。 (プログラム名中の `cc` は通常は RW ですが、 CN または SQ である場合もあります。) このメッセージがあると、FFDC データが作成されたことを示します。ヘルプ・キーを使って、メッセージに関する詳細を入手することができます。メッセージ・ヘルプに問題 ID が示されるので、それを使って、 問題処理 (WRKPRB) コマンドで提示されたリストの中で問題を確認することができます。このステップはスキップすることができます。問題記録 (ある場合) は、リストの最上部またはその近辺にあるかもしれないからです。
2. 問題処理 (WRKPRB) コマンドを入力し、 `Software problem detected in Qccxyyyy` メッセージで見つけたプログラム名 (Qccxyyyy) を指定します。そのプログラム名を使って、不要なリスト項目をフィルターで取り除きます。問題リストが提示されたら、問題 ID の入った行にオプション 5 を指定して、徴候ストリングやエラー・ログ ID などの問題の詳細を入手します。
3. エラー・ログ ID がある場合、 システム保守ツール開始 (STRSST) コマンドを入力します。最初の画面で、「保守ツールの開始」を選択します。 次の画面で 1 を入力して、「エラー・ログ・ユーティリティー」を選択します。その次の画面で 2 を入力し、「エラー・ログ ID による表示または印刷」を選択します。 さらに次の画面で、次のいずれかを行うことができます。
   - そのエラー・ログ ID を入力する。
   - Y を入力して 16 進表示を行う。
   - 印刷または表示のオプションを選択する。

表示オプションでは、行あたり 32 バイトではなく、16 バイトが用意されています。これは、80 文字のワークステーション・プリンターで画面をオンライン表示したり印刷したりする場合に便利です。表示オプションを選択した場合、実行キーを押した後、 16 進データを見るには F6 キーを押します。

16 進データには、FFDC ダンプ・データの最初の 1K バイトが入っていますが、その前に他の何らかのデータが入っています。 FFDC データの先頭は、FFDC データ索引で識別されます。ターゲット・ジョブの名前 (アプリケーション・サーバーの場合) は、データ索引の前に置かれます。 FFDC ダンプ・スプール・ファイルがまだ削除されていないときに、そのスプール・ファイルを検索するには、この完全修飾ジョブ名を使います。スプール・ファイルがなくなった場合、次のいずれかを行います。

- エラー・ログに保管されている最初の 1K のダンプを使用します。
- 1K の FFDC データでは足りない場合はその問題を再現します。

エラー・ログから FFDC データを解釈するとき、エラー・ログ内の FFDC データは、スプール・ファイル内のデータと同様、読み取り用にフォーマットされていません。エラー・ログ内の FFDC ダンプの各セクションの前には、4 バイトのヘッダーが付けられます。このヘッダーの最初の 2 バイトは、後続のセクションの長さ (接頭部は含まれません) です。その次の 2 バイトは、セクション番号を示しますが、索引内のセクション番号に対応します (271 ページの『FFDC ダンプ出力の説明』を参照)。

アプリケーション・サーバー問題の診断のためのサービス・ジョブの開始

アプリケーションが DRDA を使用すると、SQL ステートメントは、アプリケーション・サーバー・ジョブ内で実行されます。そのため、OS/400 オペレーティング・システム上で実行されるアプリケーション・サーバー・ジョブのデバッグまたはジョブ・トレースを行う必要があります。これを行うための技法は、APPC または TCP/IP のどちらを使用しているかによって異なります。サーバー問題を診断するためのサービス・ジョブの開始の詳細については、以下のトピックを参照してください。

- APPC サーバー用のサービス・ジョブ
- 独自の TPN の作成および QCNTSRVC の設定
- TCP/IP サーバー用のサービス・ジョブ
- QRWOPTIONS データ域の使用法

APPC サーバー用のサービス・ジョブ

DB2 UDB for iSeries アプリケーション・サーバーは、特別トランザクション・プログラム名 (TPN) を認識すると、アプリケーション・サーバーからシステム・オペレーターにメッセージを送信させてから、応答待ちになります (1 を参照)。詳細については、『独自の TPN の作成および QCNTSRVC の設定』を参照してください。それによって、サービス・ジョブの開始 (STRSRVJOB) コマンドを出して、アプリケーション・サーバー・ジョブのジョブ・トレースまたはデバッグを開始することができます。 次に示すステップを使って、 DB2 UDB for iSeries アプリケーション・サーバー・ジョブをいったん停止してから、デバッグ・モードで再始動することができます。

1. アプリケーション・リクエスターで、トランザクション・プログラム名 (TPN) として QCNTSRVC を指定します。これを行う方法は、プラットフォームごとに異なります。以下の項では、さまざまな方法について説明します。
2. OS/400 アプリケーションは、QCNTSRVC の TPN を受け取ると、 QSYSOPR にメッセージ CPF9188 を送り、G (go) の応答を待ちます。
3. G 応答を入力する前に、 サービス・ジョブの開始 (STRSRVJOB) コマンドを使用して、アプリケーション・サーバー・ジョブのサービス・ジョブを開始し、そのジョブをデバッグ・モードにします。 (ジョブ名を表示するには、CPF9188 メッセージでヘルプを要求します。)
4. デバッグの開始 (STRDBG) コマンドを入力します。
5. アプリケーション・サーバー・ジョブのデバッグを開始したら、 QSYSOPR メッセージに G を応答します。
6. G の応答を受信すると、アプリケーション・サーバーは通常の DRDA 処理を続行します。
7. アプリケーションの実行後、アプリケーション・サーバーのジョブ・ログを表示して、 SQL デバッグ・メッセージを調べることができます。

独自の TPN の作成および QCNTSRVC の設定

DB2 UDB for iSeries アプリケーション・リクエスターでの TPN としての QCNTSRVC の設定

リレーショナル・データベース・ディレクトリー項目の追加 (ADDRDBDIRE) または リレーショナル・データベース・ディレクトリー項目の変更 (CHGRDBDIRE) コマンドの TNSPGM パラメーターに QCNTSRVC を指定します。

RDB ディレクトリー項目のテキスト内の特殊 TPN を書き留めておけば、デバッグの終了後にその TPN を元に戻すときの覚え書きになるので便利です。

DB2 UDB for iSeries アプリケーション・サーバー (AS) ジョブのデバッグ用の独自の TPN の作成

末尾にデバッグ・ステートメントと TFRCTL QSYS/QCNTEDDM ステートメントの入った制御言語プログラムをコンパイルすれば、独自の TPN を作成することができます。これを作成した場合の利点は、接続を行うときに手動介入しなくて済むことにあります。そのようなプログラムの例を次に示します。

注: xv ページの『コードの特記事項情報』に重要なリーガル情報がありますので、お読みください。

```
PGM
   MONMSG CPF0000
      STRDBG UPDPROD(*YES)  PGM(CALL/QRWTEXEC) MAXTRC(9999)
      ADDBKP STMT(CKUPDATE) PGMVAR((*CHAR (SQLDA@))) OUTFMT(*HEX) +
               LEN(1400)
      ADDTRC PGMVAR((DSLENGTH ()) (LNTH ()) (FDODTA_LNTH ()))
      TRCJOB *ON TRCTYPE(*DATA) MAXSTG(2048) TRCFULL(*STOPTRC)
      TFRCTL QSYS/QCNTEDDM
ENDPGM
```

アプリケーション・リクエスター (AR) の RDB ディレクトリー項目内の TPN 名は、プログラマーが指定する名前です。特殊 TPN が使用中であったり、デバッグを完了した後で TPN 名を必ず元に戻すよう指示したりする警告を指定するには、テキスト・フィールドを使います。

RDB の TPN をいったん変更すると、再び元に戻さない限り、その AR からのすべての接続はその新しい TPN を使い続けることに注意してください。その場合、パフォーマンスの低下やオペレーター応答の長時間待ちが生じたり、記憶域がデバッグ・データでいっぱいになったりして、事情を知らないユーザーの不意を突くことになります。

DB2 UDB for VM アプリケーション・リクエスターでの TPN としての QCNTSRVC の設定

UCOMDIR NAMES ファイルを変更して、TPN タグに QCNTSRVC を指定します。

以下にその例を示します。

```
        :nick.RCHASLAI :tpn.QCNTSRVC
                               :luname.VM4GATE RCHASLAI
                               :modename.MODE645
                               :security.NONE
```

次に、SET COMDIR RELOAD USER を発行します。

DB2 UDB for z/OS アプリケーション・リクエスターでの TPN としての QCNTSRVC の設定

SYSIBM.LOCATIONS テーブルを更新して、 DB2 UDB for iSeries アプリケーション・サーバーの RDB-NAME の入った行について、 TPN 列に QCNTSRVC を指定します。リリース 5 より前のバージョンを実行しているシステムでは、上記の説明を SYSIBM.SYSLOCATIONS テーブルと LINKATTR 列に置き換えて読んでください。

DB2 Connect アプリケーション・リクエスターでの TPN としての QCNTSRVC の設定

DB2 Connect および Universal Database を処理しており、製品のこのファミリー上に TPN をセットアップする方法についての指示が必要な場合には、 Web ページ 『Knowledge Base: DB2 Universal Database and DB2 Connect for Windows, OS/2®, UNIX®』を参照してください。そこで、異なるバージョンに特定の

いくつかの資料を見つけることができます (ただし、この資料で説明されているすべての機能がすべてのバージョンでサポートされているわけではないことに注意してください)。

TCP/IP サーバー用のサービス・ジョブ

DDM TCP/IP サーバーは、APPC サーバーのように TPN を使用することはありません。しかし、TCP/IP サーバーで事前開始ジョブを使えば、その環境でサービス・ジョブを開始する手段になります。ただし、QRWOPTIONS データ域の使用法と関連した機能の紹介では、大抵の場合、サービス・ジョブを開始する必要がないことに注意してください。その機能を使用すると、トレースを開始したり、その他の診断機能を開始することができます。ジョブの接続フェーズのトレースが必要な場合には、サービス・ジョブを開始する必要が生じることがあります。

以下のステートメントが真である場合、 ログの表示 (DSPLOG) コマンドを使用して、指定された接続に使用されるサーバー・ジョブの名前を報告している CPI3E34 メッセージを見つけることができます。

- 接続操作中にサーバーのアクションをトレースする必要はありません。
- QRWOPTIONS 機能を使用しないことを選択します。
- 対話式 SQL からなど、サーバー上にいくつかのセットアップを行うことができるまで、**アプリケーション・リクエスター (AR)** ジョブの実行を遅らせる機能を持っています。

次に、前の項に説明されているとおり、サービス・ジョブの開始 (STRSRVJOB) コマンドを使用することができます。

接続ステートメントをトレースする必要があるか、または、接続後にサーバー上で手動セットアップを行う時間がない場合、接続が確立される前に、その接続でどの事前開始ジョブが使われるかを知っている必要があります。そのためには、可能であれば、テスト中に他のユーザーが接続できないようにしてから、 1 つ以外の事前開始ジョブをすべて終了するのが 1 つの方法です。

事前開始ジョブの数を強制的に 1 つにするには、 QSYSWRK 内で実行される QRWTSRVR 用の 事前開始ジョブ項目変更 (CHGPJE) コマンドで、以下のパラメーターを、それぞれ示してある値に設定します。

- 最初のジョブ数: **1**
- 限界値: **1**
- 追加のジョブ数: **0**
- 最大ジョブ数: **1**

この技法を使用した場合、必ずパラメーターを後で変更して、各自の環境に適した値に戻すよう気を付けてください。そうしないと、1 つの事前開始ジョブが使用中のときにユーザーが接続を試みると、 `'A connection with a remote socket was reset by that socket'` というメッセージが出されます。

QRWOPTIONS データ域の使用法

QRWOPTIONS データ域

DDM/DRDA TCP/IP サーバー・ジョブは、開始されると、ユーザーが診断や他のオプションを指定できるデータ域を探します。このデータ域の名前は QRWOPTIONS で、このデータ域は QGPL ライブラリーに常駐している場合にのみ有効になります。このデータ域は、48 文字のストリングから成っています。

注: データ域内の情報は、CCSID 37 または 500 に大文字で入力されなければなりません。

データ域の形式は、以下のとおりです。

表 13. データ域の形式

列	内容
1-15	スイッチの値として 'I' が指定されるとき (その他の場合には無視される) に使用される小数点付き 10 進数のクライアント IP アドレス。
16	サーバーによって無視される予約済み域 (人間が使用できる文字を含めることができる)
17-26	スイッチの値として 'U' が指定されるとき (その他の場合には無視される) に比較されるユーザー・プロファイル名
27	'A'、'I'、または 'U' に設定される場合にジョブ・ログが保持されるようにするスイッチ (注 1 および 2 を参照)
28	'A'、'I'、または 'U' に設定される場合に DSPJOB 出力が印刷されるようにするためのスイッチ (注 1 および 2 を参照)
29	'A'、'I'、または 'U' に設定される場合にジョブがトレースされるようにするスイッチ (注 1 および 2 を参照)。
30	'A'、'I'、または 'U' に設定される場合にジョブにデバッグが開始されるようにするスイッチ (注 1 を参照)。
31	'A'、'I'、または 'U' に設定される場合に QRYOPTLIB 値を含む 照会変更属性 (CHGQRYA) コマンドを呼び出すためのスイッチ。 QRYOPTLIB 値は、正しい QAQQINI ファイルを含むライブラリーの名前を含んでいなければならない列 39 から 48 から抽出されます (注 1 を参照)。 **注:** この列に 'I' または 'A' が指定されている場合、 QUSER は、それが実施される *JOBCTL 特殊権限を持っていなければなりません。
32	'A'、'I'、または 'U' に設定されている場合、クライアント・デバッグ・オプションをシャドー作成するためのスイッチ (注 1 を参照)。
33	'T' に設定される場合、ジョブ・トレースのために新しい STRTRC の代わりに古い TRCJOB を使用するためのスイッチ。列 29 はトレースを要求する。 **注:** この列が 'T' に設定される場合、ジョブ・トレースのために TRCJOB が使用される。 STRTRC を使用するには、この列にブランクまたは 'S' に設定する。
34	ヒストリー・ログ内の CPI3E34 メッセージを抑制するには、この列を 'N' に設定する (これは PTF SI02613 とともにのみ V5R1 で使用可能である)
35	'A'、'I'、または 'U' に設定される場合に特殊なサブルーチン・トレースを開始するスイッチ (注 1 および 2 を参照)。
36-38	予約済み
39-48	汎用データ域 (列 31 内の適当な値によって 照会変更属性 (CHGQRYA) コマンドがトリガーされる場合にはライブラリー名を含む)

注:

1. 以下は、表示される列に対応する機能を活動化するスイッチの値です。
   - 'A' は、サーバー・ジョブをすべて使用するための機能を活動化します。
   - 'I' は、列 1 から 15 で指定されたクライアント IP アドレスが、接続の試みで使用されるクライアント IP アドレスと一致する場合に機能を活動化します。
   - 'U' は、列 17 から 26 で指定されたユーザー ID が、接続の試みで使用されるユーザー ID と一致する場合に機能を活動化します。
2. この機能を活動化した結果生じるスプール・ファイルを見つけるには、 ジョブ処理コマンド (WRKJOB user-profile/QPRTJOB) を使用します。ここで、user-profile は、接続要求で使用されるユーザー ID です。オプション 4 を選択します。1 つ以上のスプール・ファイルが表示されます。

表 14. WRKJOB user-profile/QPRTJOB コマンドからのファイル・リスト

ファイル	装置および待ち行列	ユーザー・データ
QPJOBLOG	QEZJOBLOG	QRWTSRVR
QPDSPJOB	PRT01	
QPSRVTRC	PRT01	

3. 特殊な DRDA サブルーチン・トレースを含むファイルはライブラリー QGPL に作成され、QRWDBmmddy (mm は月、dd は日、y は年の最後の 1 桁を表し、トレースが記録された日を示します) のフォーマットで命名されます。ただし、すべてのサーバー・プログラムがトレースされるわけではありません。

詳細については、以下の例を参照してください。

例: データ域を作成するための CL コマンド

```
CRTDTAARA DTAARA(QGPL/QRWOPTIONS) TYPE(*CHAR) LEN(48)
          VALUE('9.5.114.107    :MYUSERID  AAUIU TN    INILIBRARY')
          TEXT('DRDA TCP SERVER DIAGNOSTIC OPTIONS'
```

この例では、テーブル 15 に示されている機能を要求します。

注: この例にある正しいスペーシングが重要なので、 VALUE パラメーターの内容は、テーブル形式で繰り返されます。

表 15. CRTDTAARA の VALUE パラメーター内のデータ・エレメントの説明

列	内容	説明
1 から 11	9.5.114.107	列 30 にあるスイッチで使用するための IP アドレス。
16	:	IP アドレス・フィールドの終わりにマークを付けるための文字
17 から 24	MYUSERID	列 29 および 31 にあるスイッチで使用するためのユーザー ID
27	A	すべてのユーザーのためのサーバー・ジョブ・ログ (QRWTSRVR 用) をスプールする。
28	A	すべてのサーバーを使用するために DSPJOB 出力をスプールする。
29	U	接続要求上のユーザー ID がデータ域の列 17 から 26 (この例では 'MYUSERID') に指定されているユーザー ID と一致する場合、 (TRCJOB) コマンドを使用してジョブをトレースする。
30	I	クライアント ID アドレス (この例では、'9.5.114.107') がデータ域の列 1 から 15 に指定されているクライアント IP アドレスと一致する場合、 デバッグの開始 (STRDBG) コマンド (プログラムを指定しない) を使用してデバッグを開始する。
31	U	接続要求上のユーザー ID がデータ域の列 17 から 26 (この例では 'MYUSERID') に指定されているユーザー ID と一致する場合、 照会変更属性 (CHGQRYA) QRYOPTLIB(INILIBRARY) コマンドを呼び出す。 **注:** ライブラリー名は、データ域の列 39 から 48 から選択する。
32		サーバーに、クライアント・デバッグ・オプションをシャドー作成しない。
33	T	ジョブ・トレースのために古い TRCJOB 機能を使用する。

表 15. CRTDTAARA の VALUE パラメーター内のデータ・エレメントの説明 (続き)

列	内容	説明
34	N	ヒストリー・ログ内に CPI3E34 メッセージを入れない。
35		サブルーチン・トレースを開始しない。
39-48	INILIBRARY	スイッチ 31 で使用されるライブラリー。

第 10 章 分散リレーショナル・データベース・アプリケーションの作成

プログラマーは、iSeries 分散アプリケーション・プログラム用の SQL ステートメントを使用する、高水準言語プログラムを作成することができます。 ローカル処理用に作成されるプログラムとの主な違いは、リモート・データベースに接続することができ、 SQL パッケージを作成することができる点にあります。CONNECT SQL ステートメントを使用して、アプリケーション・リクエスターをアプリケーション・サーバーに明示的に接続するか、またはプログラムの作成時に、リレーショナル・データベースの名前を指定して、暗黙接続することができます。また、SET CONNECTION、RELEASE、および DISCONNECT ステートメントを使用して、分散作業単位を使用するアプリケーションのために接続を管理できます。

SQL パッケージとは、分散リレーショナル・データベースでのみ使用される iSeries オブジェクトのことです。 SQL のプリコンパイル処理の結果として作成するか、またはコンパイル済みプログラム・オブジェクトから作成することができます。 SQL パッケージは、アプリケーション・サーバー側に存在します。このパッケージには、SQL ステートメント、ホスト変数属性、およびアプリケーション・サーバーがアプリケーション・リクエスターの要求を処理するのに使うアクセス・プランが入っている。

アプリケーション・プログラムは多くの異なるサーバーに接続できるので、プログラマーは、サーバー間のデータ変換には十分に注意を払う必要があります。 iSeries server では、さまざまなタイプのデータ変換を用意しており、これには文字情報の管理用のコード化文字セット識別子 (CCSID) サポートも含まれています。

ローカル処理のアプリケーションに SQL 言語を使用するのと同じように、 SQL 言語を使用して iSeries server で分散リレーショナル・データベース用のプログラムを作成し、保守することができます。 以下の高水準言語の 1 つまたはいくつかを使用して、静的および動的構造化照会言語 (SQL) ステートメントを組み込むことができます。

- iSeries PL/I
- ILE C for AS/400 (ILE C/400*)
- COBOL/400
- ILE COBOL/400
- FORTRAN/400*
- RPG/400®
- ILE RPG/400

分散アプリケーションを開発するためのプロセスは、ローカル処理用の SQL アプリケーションを開発するプロセスに類似しています。異なっているのは、分散処理のアプリケーションでは、接続されるリレーショナル・データベースの名前を指定しなければならないという点です。これはプログラムの事前コンパイル時、またはアプリケーション内で行うことができます。

使用される SQL オブジェクトは、1 つを除いて、ローカル・アプリケーションでも分散アプリケーションでも同じです。つまり、SQL パッケージというオブジェクトが 1 つだけ分散リレーショナル・データベースの専用になります。 SQL プログラム作成 (CRTSQL*xxx*) コマンドを使用して、プログラムを作成します。このコマンドの *xxx* は、ホスト言語である、CI、CBL、CBLI、FTN、PLI、RPG、または RPGI を指

します。 SQL パッケージはこのプロセスにおける事前コンパイルによって作り出される場合があります。既存の分散 SQL プログラムの場合は、 構造化照会言語パッケージの作成 (CRTSQLPKG) コマンドによって SQL パッケージを作成します。

SQL ステートメントを組み込んだプログラムを事前コンパイルするためには、 DB2 UDB Query Manager and SQL Development Kit ライセンス・プログラムをインストールしておかなければなりません。 ただし、サーバーにインストールされているコンパイル済みプログラムだけを使用して、既存の分散 SQL プログラムから SQL パッケージを作成することができます。 DB2 UDB Query Manager and SQL Development Kit ライセンス・プログラムを使用すると、対話式 SQL によって分散リレーショナル・データベースにアクセスすることができます。 これはプログラムをデバッグするときに役立ちます。その理由は、プログラムを事前コンパイルしたりコンパイルしたりせずに、 SQL ステートメントをテストすることができるからです。

この章では、分散リレーショナル・データベースの場合のプログラミングの問題の概要を示しています。以下のトピックに関する詳細は、 iSeries Information Center の『SQL プログラミング 概念』を参照してください。

- 分散リレーショナル・データベース・アプリケーションのプログラミングに関する考慮事項
- 分散リレーショナル・データベース・プログラムの作成
- SQL パッケージの処理

分散リレーショナル・データベース・アプリケーションのプログラミングに関する考慮事項

iSeries server 上の分散リレーショナル・データベース・アプリケーションのプログラミングに関する考慮事項は、次の 2 つのカテゴリーに分けられます。つまり、ローカル・サーバー上でサポートされる機能を扱うものと、他のサーバーに接続しなければならないことに由来するものです。 このセクションでは、以下のことを説明しながら、両方のカテゴリーを扱っています。

- 命名規則
- 他のサーバーとの接続
- 分散 SQL ステートメントおよび共存
- DRDA 作業単位の終了
- コード化文字セット識別コード (CCSID)
- データ変換
- 分散データ管理 (DDM) ファイルおよび SQL プログラム

分散リレーショナル・データベース・アプリケーションを設計するときに考慮すべき追加の情報に関しては、 18 ページの『ヒント: 分散リレーショナル・データベース・アプリケーションの設計』で説明されています。

新しいレベルの DRDA プロトコル

IBM DB2 製品ファミリーの他のメンバーでは、バージョン 8 のリリースにおいて、新しいレベルのプロトコルとデータ構造 (DRDA レベル 5) を必要とする新しい機能が導入されています。 OS/400 V5R3 は、新しく設計された機能のサブセットに対する互換性を保ち、これを利用するために、レベル 5 で作動するように設計されています。OS/400 V5R3 に組み込まれた機能には、次のようなものがあります。

- よりわかりやすい Kerberos 構成。 Kerberos 構成に関する詳細は、『ネットワーク認証サービス』のトピックを参照してください。

- SQL ステートメントの実行を準備する際にそのステートメントの属性を指定できる機能。 『SQL リファレンス』のトピックの PREPARE ステートメントの項を参照してください。
- SQL ステートメントの完了時に入手できる、新しい、より複雑な診断情報。このデータのコレクションは、従来の SQLCA 制御ブロックに代わるものとして用意されたものですが、 SQLCA 制御ブロックと合わせて使用することも可能です。ただし、必ずしもすべての DB2 ファミリー・メンバーで、初めから追加情報が提供されるわけではありませんのでご注意ください。詳細は、『SQL リファレンス』の GET DIAGNOSTICS ステートメントの項を参照してください。
- ストアード・プロシージャーが Query の結果セットを戻すとき、 1 つのプロシージャー呼び出しの Query を開いたまま、後続の呼び出しで新しいインスタンスの Query を開くことができる機能。詳細は、 『SQL プログラミングの概念』の『DRDA ストアード・プロシージャーに関する考慮事項』を参照してください。
- Query 中にトリガーや関数によって実行されるデータベース更新が、 2 フェーズ・コミット・プロセスの実行中にも考慮されるようになった点。詳細は、『SQL プログラミングの概念』の『分散サポート』を参照してください。
- iSeries がアプリケーション・サーバーとして機能している場合における、異機種スクロール可能カーソル機能の限定セットのサポート (読み取り専用アクセス)。ただし、スクロール可能カーソルを使用した LOB フィールド (BLOB や CLOB など) の検索はサポートされていません。詳細は、 『SQL プログラミングの概念』の『スクロール可能カーソル』を参照してください。

分散リレーショナル・データベース・オブジェクトの命名

SQL オブジェクトは、iSeries server オブジェクトとして作成され、保守されます。

DB2 Universal Database for iSeries プログラミングでは、次の命名方式のいずれかを使用することができます。すなわち、システムと (*SYS) と SQL (*SQL) のいずれかです。 使用する命名規則によって、ファイル名およびテーブル名を修飾する方法に影響が生じます。また、セキュリティーおよび対話式 SQL 画面で使用される用語にも影響があります。分散リレーショナル・データベース・アプリケーションが別の iSeries server 上のオブジェクトにアクセスするのであれば、命名規則は、2 つのうちどちらを使用することもできます。 しかし、プログラムが iSeries server 以外のシステム上のリレーショナル・データベースにアクセスする場合には、 SQL 名だけしか使用できません。 SQL の開始 (STRSQL) コマンドの NAMING パラメーター、または CRTSQL*xxx* コマンドの 1 つの OPTION パラメーターを使用して、命名規則を選択してください。

システム (*SYS) 命名規則

システム命名規則を使用するときは、ファイルは、*ライブラリー/ファイル* という形式によってライブラリー名で修飾されます。テーブルがこの命名規則を使用して作成される場合には、テーブルが作成されるライブラリーの共通権限が前提となります。テーブル名が明示的に修飾されず、デフォルトのコレクション名が CRTSQL*xxx* コマンドまたは CRTSLQPKG コマンドの DFTRDBCOL パラメーターで使用される場合には、デフォルトのコレクション名が静的 SQL ステートメントで使用されます。ファイル名が明示的に修飾されず、デフォルトのコレクション名が指定されない場合には、以下の規則が適用されます。

- 特定の CREATE ステートメント以外のすべての SQL ステートメントで、 SQL は、修飾されていないファイルを見つけるために、ライブラリー・リスト (*LIBL) を検索します。
- CREATE ステートメントでは、修飾されていないオブジェクトについて、以下のように解決します。
  - CREATE TABLE: テーブル名は明示的に修飾されなければなりません。
  - CREATE VIEW: ビューは、副選択で参照された最初のライブラリーの中に作成されます。
  - CREATE INDEX: 索引は、その索引が作成されるテーブルが入っているコレクションまたはライブラリーの中に作成されます。

SQL (*SQL) 命名規則

SQL 命名規則を使用するときは、テーブルは、*コレクション.テーブル* という形式によってコレクション名で修飾されます。テーブル名が明示的に修飾されず、デフォルトのコレクション名が CRTSQLxxx コマンドまたは構造化照会言語パッケージの作成 (CRTSQLPKG) コマンドのデフォルトのリレーショナル・データベース・コレクション (DFTRDBCOL) パラメーターで指定される場合には、デフォルトのコレクション名が使用されます。テーブル名が明示的に修飾されず、デフォルトのコレクション名が指定されない場合には、以下の規則が適用されます。

- 静的 SQL では、デフォルトの修飾名はプログラム所有者のユーザー・プロファイルです。
- 動的 SQL または対話式 SQL では、デフォルトの修飾名はステートメントを実行するジョブのユーザー・プロファイルです。

デフォルトのコレクション名

プログラムの事前コンパイル時に、 CRTSQL*xxx* コマンドの DFTRDBCOL パラメーターにデフォルトのコレクション名を指定することによって、デフォルトのコレクション名が SQL プログラムで使用されるように指定することができます。 DFTRDBCOL パラメーターは、 *SYS 命名規則が使用される場合、修飾されていないファイルのライブラリーとして、あるいは、*SQL 命名規則が使用される場合、修飾されていないテーブルのコレクションとして、コレクション名をプログラムに提供します。プログラムの事前コンパイル時にデフォルトのコレクション名を指定しなかった場合には、それぞれの命名規則に応じて、上述したように、修飾されていない名前に関する規則が適用されます。コレクション名が適用されるのは、静的 SQL ステートメントの場合だけです。

構造化照会言語パッケージの作成 (CRTSQLPKG) コマンドで DFTRDBCOL パラメーターを使用して、パッケージのデフォルトのコレクションを変更することもできます。また、SQL プログラムがコンパイルされた後で、新規 SQL パッケージを作成して、デフォルトのコレクションを変更することができます。(CRTSQLPKG) コマンドのすべてのパラメーターの説明については、 214 ページの『SQL パッケージの作成 (CRTSQLPKG) コマンドの使用』を参照してください。

分散リレーショナル・データベースへの接続

分散リレーショナル・データベース・アプリケーションが*分散される* のは、それが別のサーバー上のリレーショナル・データベースへの接続可能性を備えているからです。

CONNECT ステートメントには、同じ構文でしかも働きの異なる 2 つのタイプがあります。

- CONNECT (タイプ 1) は、リモート作業単位用に使用されます。
- CONNECT (タイプ 2) は、分散作業単位用に使用されます。

プログラムが使用する CONNECT タイプは、 CRTSQLxxx コマンドの RDBCNNMTH パラメーターで指示します。

DRDA リモート作業単位

リモート作業単位 は、 SQL ステートメントのリモートでの準備および実行のために提供されているものです。コンピューター・サーバー A の活動化グループは、コンピューター・サーバー B のアプリケーション・サーバーに接続することができます。その後、その活動化グループは、1 つ以上の作業単位の中で、B のオブジェクトを参照する静的または動的 SQL ステートメントを何回でも実行できます。コンピューター・サーバー B での作業単位が終了した後、活動化グループは、コンピューター・サーバー C などのアプリケーション・サーバーに接続することができます。

ほとんどの SQL ステートメントは、以下の制約付きでリモートで準備し、実行することができます。

- 1 つの SQL ステートメントで参照するオブジェクトは、すべて、同じアプリケーション・サーバーで管理しなければなりません。
- 1 つの作業単位の中の SQL ステートメントは、すべて、同じアプリケーション・サーバーで実行しなければなりません。

DRDA リモート作業単位接続管理: 活動化グループは、いつも 3 つの状態のいずれかになっています。
接続可能で接続済み
接続不能で接続済み
接続可能で未接続

次の図は、状態遷移を示しています。

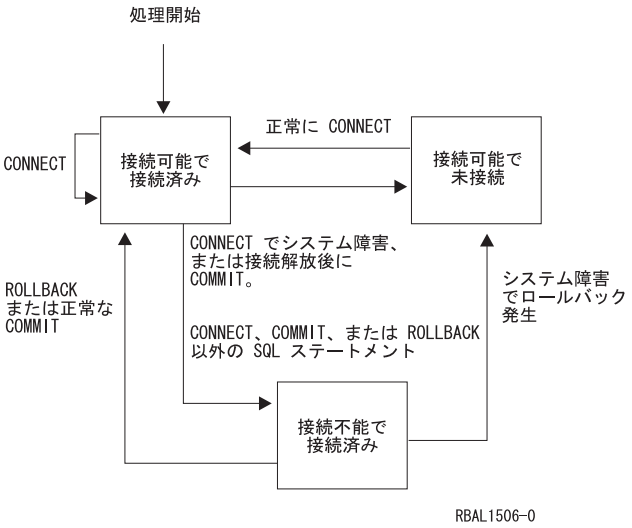


図 19. リモート作業単位の活動化グループの接続状態遷移

活動化グループの初期の状態は、*接続可能* で*接続済み* です。活動化グループが接続されるアプリケーション・サーバーは、 CRTSQLxxx と STRSQL コマンドの RDB パラメーターによって判別され、暗黙の CONNECT 操作を含むことができます。暗黙の CONNECT 操作は、暗黙または明示の CONNECT 操作が、正常であったかどうかは別として、すでに行われている場合には行われません。したがって、活動化グループは、 2 度以上アプリケーション・サーバーに暗黙に接続されることはありません。

接続可能で接続済みの状態: 活動化グループはアプリケーション・サーバーに接続され、 CONNECT ステートメントを実行することができます。活動化グループがこの状態に入るのは、ロールバックまたは正常なコミットを接続不能で接続済みの状態から完了したとき、または接続可能で未接続の状態から CONNECT ステートメントが正常に実行されたときです。

接続不能で接続済みの状態: 活動化グループはアプリケーション・サーバーに接続されていますが、アプリケーション・サーバーを変更するための CONNECT ステートメントは正常に実行されませんでした。活動化グループが接続可能で接続済み状態からこの状態に入るのは、 CONNECT、COMMIT、または ROLLBACK 以外の SQL ステートメントを実行したときです。

接続可能で未接続の状態: 活動化グループはアプリケーション・サーバーに接続されていません。実行できる唯一の SQL ステートメントは、CONNECT です。

活動化グループがこの状態になるのは、以下の場合です。

- 接続は前に解放されており、COMMIT が正常に実行された。

- 接続が SQL DISCONNECT ステートメントを使用して切断されている。
- 接続は接続可能状態だったが、CONNECT ステートメントは正常に行われなかった。

活動化グループは、CONNECT によって接続可能状態でなくなることはないので、 CONNECT ステートメントを連続して正常に実行することができます。活動化グループが現在接続されているアプリケーション・サーバーに CONNECT を行うと、その他の場合の CONNECT ステートメントと同様に実行されます。

(COMMIT(*NONE) で実行しているのでない限り) CONNECT、 COMMIT、 DISCONNECT、 SET CONNECTION、 RELEASE、または ROLLBACK 以外の SQL ステートメントに続く CONNECT は、正しく実行されません。エラーを避けるために、CONNECT ステートメントを実行する前にコミット、またはロールバックの操作を行ってください。

アプリケーション指示による分散作業単位

アプリケーション指示による分散作業単位 でも、リモート作業単位と同じ方法で、SQL ステートメントをリモートで準備し、実行できます。リモート作業単位の場合と同じように、コンピューター・サーバー A の活動化グループは、コンピューター・サーバー B のアプリケーション・サーバーに接続することができ、その作業単位の終わる前に、 B のオブジェクトを参照する静的または動的な SQL ステートメントをいくつでも実行することができます。 1 つの SQL ステートメントで参照するオブジェクトは、すべて、同じアプリケーション・サーバーで管理しなければなりません。ただし、リモート作業単位の場合とは異なり、同じ作業単位の中で、任意の数のアプリケーション・サーバーと関係を持つことができます。コミットまたはロールバックの操作で作業単位が終了します。

アプリケーション指示による分散作業単位の接続管理: どのような場合でも、以下のことが言えます。

- 活動化グループは、常に *接続済み* 状態であるか、または *未接続* の状態です。すなわち接続がゼロであるか、1 つ以上の接続を持っています。活動化グループの各接続は、接続しているアプリケーション・サーバーの名前によって唯一のものとして識別されます。
- SQL 接続は、常に以下のいずれかの状態です。
  - 現行で保持の状態
  - 現行で解放の状態
  - 休止で保持の状態
  - 休止で解放の状態

活動化グループの初期状態: 活動化グループは、最初は接続済み状態で、接続を 1 つ持っています。接続の初期状態は、*現行で保持* の状態です。

次の図は、状態遷移を示しています。

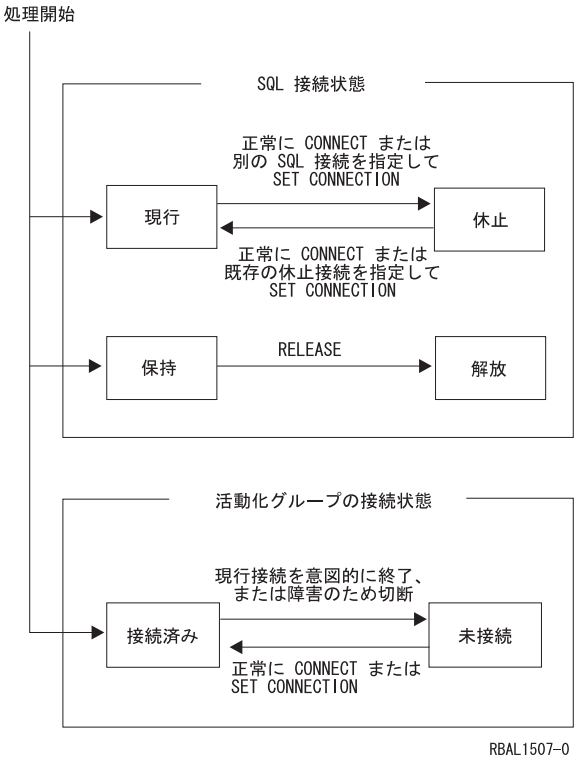


図 20. アプリケーション指示による分散作業単位接続と活動化グループの接続状態遷移

接続状態: アプリケーションが CONNECT ステートメントを実行したとき、そのサーバー名がアプリケーション・リクエスターに認識されているが、それが活動化グループの既存の接続セットの中に入っていない場合は、以下のようになります。

- 現行接続は休止状態で保持状態に置かれます。
- そのサーバー名は、接続セットに加えられ、新しい接続が現行状態で保持状態になります。

そのサーバー名がすでに活動化グループの既存の接続セットに入っている場合は、エラーとなります。

SET CONNECTION ステートメントを使用すると、休止状態の接続が現行状態になります。接続が現行状態になると、前の現行状態接続は (それがあった場合は)、休止状態になります。活動化グループの既存の接続セットの中で、現行状態になることができるのはいつでも 1 つだけです。接続状態が現行状態から休止状態に、または休止状態から現行状態に変わっても、その保持状態または解放状態には影響がありません。

RELEASE ステートメントを使用すると、接続状態は解放状態になります。活動化グループがコミット操作を実行すると、その活動化グループの各解放接続は終了します。接続状態が保持状態から解放状態に変わっても、その現行状態または休止状態には影響ありません。したがって、解放状態の接続は次のコミット操作までは使用することができます。接続を解放状態から保持状態に変える方法はありません。

活動化グループの接続状態: CONNECT ステートメントを暗黙にまたは明示的に実行することによって、異なるアプリケーション・サーバーを設定することができます。以下の規則が適用されます。

- 活動化グループは、同時に同一のアプリケーション・サーバーに複数の接続を持つことはできない。
- 活動化グループが SET CONNECTION ステートメントを実行するときの指定のロケーション名は、その活動化グループの接続セットの中の既存の接続でなければならない。
- 活動化グループが CONNECT ステートメントを実行するときの指定のサーバー名は、その活動化グループの接続セットの既存接続であってはならない。

活動化グループに現行接続がある場合は、その活動化グループは*接続済み* 状態です。

現行接続のアプリケーション・サーバー名は、 CURRENT SERVER という特殊レジスターに入っています。活動化グループは、そのアプリケーション・サーバーが管理するオブジェクトを参照する SQL ステートメントを実行することができます。

未接続状態の活動化グループは、 CONNECT または SET CONNECTION ステートメントの実行に成功すると接続済み状態になります。

活動化グループに現行接続がない場合は、その活動化グループは*未接続* 状態です。このときの CURRENT SERVER 特殊レジスターは、ブランクになっています。実行できる SQL ステートメントは、CONNECT、DISCONNECT、SET CONNECTION、 RELEASE、COMMIT、および ROLLBACK だけです。

接続済み状態の活動化グループが未接続状態になるのは、その現行接続が意図的な終了となったか、または障害が発生し、アプリケーション・サーバーでロールバックが行われて、接続が失われたために SQL ステートメントが正常に実行されなかった場合です。接続が意図的に終了となるのは、活動化グループがコミット操作を正常に実行し、その接続が解放状態になっているとき、またはアプリケーション・プロセスが正常に DISCONNECT ステートメントを実行したときです。

接続が終了するとき: 接続が終了すると、接続によって活動化グループが獲得していたすべてのリソースが割り振り解除されます。また、接続の作成と保守に使用されていたすべてのリソースも割り振り解除されます。たとえば、活動化グループが RELEASE ステートメントを実行すると、次のコミット操作で接続が終了するときにすべてのオープン・カーソルはクローズされます。

また、通信障害が起きた場合も接続は終了することがあります。この場合は、その活動化グループは未接続状態になります。活動化グループのすべての接続は、その活動化グループが終了すると、終了します。

RUW と DUW の両方の接続管理を使用した実行: RUW 接続管理でコンパイルしたプログラムは、DUW 接続管理でコンパイルしたプログラムで呼び出すことができます。 RUW 接続管理でコンパイルしたプログラムは、 SET CONNECTION、 RELEASE、および DISCONNECT ステートメントを使用して、任意の活動接続を処理することができます。ただし、DUW 接続管理でコンパイルしたプログラムが、RUW 接続管理でコンパイルしたプログラムを呼ぶときは、 RUW 接続管理でコンパイルしたプログラムで実行される CONNECT が、 CONNECT 機能の一部としてその活動化グループのすべての活動接続を終了しようとします。

そのような CONNECT は、活動状態の接続が使用している会話が保護会話である場合、失敗します。さらに、非活動状態の接続のために保護会話が使用されていて、DDMCNV ジョブ属性が *KEEP であると、これらの未使用の DDM 会話も、RUW 接続管理でコンパイルされたプログラムの接続を失敗させます。このような事態を避けるためには、ジョブ属性は DDMCNV(*DROP) を使用して、 RUW 接続管理でコンパイルしたプログラムを呼び出して CONNECT を実行する前に、 RELEASE と COMMIT を実行してください。

同様に、RUW 接続管理でコンパイルしたプログラムのためのパッケージを作成した後で、 DUW 接続管理でコンパイルしたプログラムのためのパッケージを作成するときは、 DDMCNV(*DROP) を指定して実行するか、または DUW 接続管理でコンパイルしたプログラムのためのパッケージを作成した後で RCLDDMCNV を実行してください。

DUW 接続管理でコンパイルしたプログラムは、 RUW 接続管理でコンパイルしたプログラムから呼び出すこともできます。 DUW 接続管理でコンパイルしたプログラムが CONNECT を実行しても、 RUW 接続管理でコンパイルしたプログラムによる接続は切断されません。この接続を、DUW 接続管理でコンパイルしたプログラムで使用することができます。

デフォルトの活動化グループのための暗黙の接続管理

アプリケーション・リクエスターは、暗黙にアプリケーション・サーバーに接続することができます。暗黙に接続されるのは、デフォルトの活動化グループのための最初の活動状態 SQL プログラムが最初の SQL ステートメントを発行したことをアプリケーション・リクエスターが検出し、以下の項目が該当する場合です。

- 発行された SQL ステートメントがパラメーターをもつ CONNECT ステートメントでない。
- デフォルトの活動化グループで SQL が活動状態になっていない。

分散プログラムの場合、暗黙の接続は RDB パラメーターに指定されたリレーショナル・データベースへの接続になります。非分散プログラムの場合、暗黙の接続はローカル・リレーショナル・データベースへの接続になります。

SQL が活動状態でなくなるとき、 SQL はデフォルトの活動化グループのすべての活動状態の接続を終了します。 SQL は以下の時点で活動状態でなくなります。

- アプリケーション・リクエスターが、そのプロセスの最初の活動状態 SQL プログラムが終了したことを検出し、さらに以下が該当する場合。
  - 保留中の SQL 変更がない
  - 保護会話を使用している接続がない
  - SET TRANSACTION ステートメントが活動状態ではない
  - CLOSQLCSR(*ENDJOB) で事前コンパイルされたプログラムを実行していなかった

  保留中の変更、保護会話、または活動中の SET TRANSACTION があると、SQL は終了状態になります。 CLOSQLCSR(*ENDJOB) を指定して事前コンパイルしたプログラムを実行する場合は、 SQL はそのジョブが終わるまでデフォルトの活動化グループのために活動状態のままになっています。
  - SQL が終了状態で、作業単位の終わりになっている場合。これは、SQL プログラムの外側で COMMIT または ROLLBACK コマンドが発行された場合に起きます。
  - ジョブの終わり。

デフォルトでない活動化グループのための暗黙の接続管理

アプリケーション・リクエスターは、暗黙にアプリケーション・サーバーに接続することができます。暗黙に接続されるのは、その活動化グループのために最初の SQL ステートメントを発行したことをアプリケーション・リクエスターが検出し、それがパラメーター付きの CONNECT ステートメントではない場合です。

分散プログラムの場合、暗黙の接続は RDB パラメーターに指定されたリレーショナル・データベースへの接続になります。非分散プログラムの場合、暗黙の接続はローカル・リレーショナル・データベースへの接続になります。

プロセスの以下の時点では、暗黙の切断が起きることがあります。

- 活動化グループが終了したときに、コミットメント制御が活動状態ではないか、活動化グループ・レベルのコミットメント制御が活動状態か、またはジョブ・レベルのコミットメント定義が作業単位境界にある場合。

ジョブ・レベルのコミットメント定義が活動中で、作業単位境界になっていないときは、SQL は終了状態になります。
- ジョブ・レベルのコミットメント定義がコミットされるか、ロールバックされるときに、SQL が終了状態の場合。
- ジョブの終わり。

以下のプログラム例は分散ではありません (接続は必要ありません)。これは Spiffy 社の地域支社で実行されるプログラムであり、ローカルの修理情報を報告書の形で収集するためのものです。

注: xv ページの『コードの特記事項情報』に重要なリーガル情報がありますので、お読みください。

```
CRTSQLxxx PGM(SPIFFY/FIXTOTAL) COMMIT(*CHG) RDB(*NONE)

    PROC: FIXTOTAL;
       .
       .
       .
      SELECT * INTO :SERVICE       A
               FROM REPAIRTOT;
    EXEC SQL
    COMMIT;
       .
       .
       .
    END FIXTOTAL;
```

A ローカル・リレーショナル・データベースに対して実行されるステートメント。

以下の例のような、別のプログラムでは、カンザス・シティー地域の Spiffy 社の販売店から同様の情報を収集することができます。これは、暗黙に接続および切断が行われる分散プログラムの例です。

```
CRTSQLxxx PGM(SPIFFY/FIXES) COMMIT(*CHG) RDB(KC101) RDBCNNMTH(*RUW)

    PROC: FIXES;
       .
       .
       .
    EXEC SQL
      SELECT * INTO :SERVICE              B
               FROM SPIFFY.REPAIR1;

    EXEC SQL   C
     COMMIT;
       .
       .
       .
    END FIXES;   D
```

B **アプリケーション・サーバー (AS)** への暗黙の接続。ステートメントは AS に対して実行されます。

C 作業単位の終了。 COMMIT が正常に行われた場合には、**アプリケーション・リクエスター (AR)** は接続可能で接続済みの状態に入ります。

D SQL プログラムの終了時の暗黙切断。

明示 CONNECT

CONNECT ステートメントを使用して、指定された**アプリケーション・サーバー (AS)** に明示的に**アプリケーション・リクエスター (AR)** を接続します。この SQL ステートメントはアプリケーション・プログラムに組み込むことも、対話式 SQL を使用して発行することもできます。 CONNECT ステートメントは、TO 文節または RESET 文節を指定して使用します。 TO 文節を指定した CONNECT ステートメントを使

用すると、特定の AS リレーショナル・データベースへの接続を指定することができます。 RESET 文節を指定した CONNECT ステートメントでは、ローカル・リレーショナル・データベースへの接続を指定します。

TO 文節または RESET 文節を指定した CONNECT ステートメントを発行する (またはプログラムが発行する) ときは、指定された AS がリレーショナル・データベース・ディレクトリーに記述されていなければなりません。このディレクトリーを処理する方法の詳細については、72 ページの『リレーショナル・データベース・ディレクトリーの使用』を参照してください。また、CONNECT ステートメントが正常に実行されるためには、 AR が接続可能状態でなければなりません。

CONNECT ステートメントの効果は、使用している接続管理によって異なります。 RUW 接続管理では、CONNECT ステートメントの効果は以下の通りです。

- TO 文節または RESET 文節を指定した CONNECT ステートメントが正常に実行されると、以下のようになります。
  - アプリケーション・プロセスが、 COMMIT HOLD または ROLLBACK HOLD SQL ステートメントの使用によって接続可能状態に入った場合、またはアプリケーション・プロセスが COMMIT(*NONE) を実行中である場合には、オープンのカーソルはいずれもクローズされ、準備されていたステートメントはいずれも廃棄され、保持されていたリソースはいずれも前の AS から解放されます。
  - アプリケーション・プロセスが、前に接続されていた AS (もしあれば) から切断され、指定された AS に接続されます。
  - AS の名前が現行サーバー特殊レジスターに置かれます。
  - エラーを戻したシステム・モジュールを識別する情報は、 SQL 通信域 (SQLCA) の SQLERRP フィールドか、 SQL 診断域の DB2_MODULE_DETECTING_ERROR フィールドに置かれます。
- 何らかの理由で CONNECT が成功しないと、アプリケーションは接続可能で未接続の状態のままになります。接続可能で未接続の状態のアプリケーションでは、 CONNECT ステートメントだけを実行できます。
- AR は、CONNECT によって接続可能状態でなくなることはないので、 CONNECT ステートメントを連続して正常に実行することができます。 AR が現在接続されている AS に CONNECT を行うと、その他の場合の CONNECT ステートメントと同様に実行されます。
- コミットメント制御で実行中に、 CONNECT、 SET CONNECTION、 COMMIT、 ROLLBACK、 DISCONNECT、または RELEASE 以外の SQL ステートメントに続く CONNECT ステートメントは、正しく実行されません。エラーを避けるために、CONNECT ステートメントを実行する前に、 COMMIT または ROLLBACK の操作を行ってください。コミットメント制御を使用しないで実行する場合には、CONNECT はいつでも使用することができます。

DUW 接続管理では、CONNECT ステートメントの効果は以下の通りです。

- TO 文節または RESET 文節を指定した CONNECT ステートメントが正常に実行されると、以下のようになります。
  - AS の名前が現行サーバー特殊レジスターに置かれます。
  - エラーを戻したシステム・モジュールを識別する情報は、 SQL 通信域 (SQLCA) の SQLERRP フィールドか、 SQL 診断域の DB2_MODULE_DETECTING_ERROR フィールドに置かれます。
  - 接続のタイプに関する情報も、 SQLCA および SQL 診断域に置かれます。これらのフィールドにエンコードされるのは、以下の情報です。
    - アプリケーションが接続状態か未接続状態かの情報は、 SQLCA の SQLERRD(5) または SQL 診断域の DB2_CONNECTION_STATE にあります。

- リモート接続が保護された会話を使用しているか無保護の会話を使用しているかの情報は、SQLCA の SQLERRD(4) または SQL 診断域の DB2_CONNECTION_TYPE にあります。
- 接続が常に読み取り専用であるか、常に更新可能であるか、あるいは更新可能と更新不可能が各作業単位間で変更できるかに関する情報は、 SQLCA の SQLERRD(4) または SQL 診断域の DB2_CONNECTION_STATUS にあります。

SQLCA の SQLERRD フィールドに関する詳細、および SQL 診断域の接続情報については、『SQL プログラミング 概念』のトピックを参照してください。

- AR が接続可能状態でないか、またはサーバー名 がローカル・リレーショナル・データベース・ディレクトリーにないために、 TO 文節または RESET 文節を指定した CONNECT ステートメントが正常に実行されなかった場合には、 AR の接続状態は変更されません。
- 現行接続になっている AS への接続は、エラーとなります。
- TO 文節または RESET 文節を持たない接続を使用して、現行接続の情報を入手することができます。この情報には、以下のような情報が含まれます。
  - 状況を戻したシステム・モジュールを識別する情報は、 SQL 通信域 (SQLCA) の SQLERRP フィールドか、 SQL 診断域の DB2_MODULE_DETECTING_ERROR フィールドに置かれます。
  - 他の状況情報は、 SQLERRD(4) と SQLERRD(5) の内容およびそれに対応する SQL 診断域の情報について扱ったセクションの前の部分で説明されています。

アプリケーション・プロセスによって実行される最初の SQL ステートメントを CONNECT ステートメントにするのは適切な例です。ただし、プログラムに CONNECT ステートメントを組み込んでいるときに、そのプログラムが複数の AS に接続するのであれば、AS 名を動的に変更することができます。アプリケーションを複数のサーバーで実行する場合には、以下に示すように、ホスト変数を指定した CONNECT ステートメントを指定して、プログラムにリレーショナル・データベース名が渡されるようにすることができます。

```
CONNECT TO  : host-variable
```

CONNECT ステートメントを使用しない場合、AS を変更するには、新規リレーショナル・データベース名を使用してプログラムを再コンパイルするだけでよいことになります。

以下の例では、アプリケーション・プログラム中に組み込まれた 2 つの形式 (**1** および **2**) の CONNECT ステートメントが示されています。

```
CRTSQLxxx PGM(SPIFFY/FIXTOTAL) COMMIT(*CHG) RDB(KC105)

  PROC: FIXTOTAL;
  EXEC SQL   CONNECT TO KC105; 1

   ⋮
  EXEC SQL
    SELECT * INTO :SERVICE
             FROM REPAIRTOT;

   ⋮
  EXEC SQL COMMIT;

   ⋮
  EXEC SQL CONNECT TO MPLS03 USER :USERID USING :PW;  2

   ⋮
  EXEC SQL SELECT ...

   ⋮
  EXEC SQL COMMIT;
```

```
  ⋮
END FIXTOTAL;
```

例 (**2**) は、 USER/USING 形式の CONNECT ステートメントの使用法を示しています。この形式の CONNECT ステートメントがプログラムに組み込まれているときは、ホスト変数でユーザー ID とパスワードを指定しなければなりません。 TCP/IP を使用している場合、適切なパラメーターを指定した サーバー許可項目の追加 (ADDSVRAUTE) コマンドを使用してユーザー ID とパスワードを保管してあれば、接続時にセキュリティー・オブジェクトからユーザー ID とパスワードを抽出できます。

以下の例は、対話式 SQL の中での両方の CONNECT ステートメント形式を示しています。パスワードは単一引用符で囲まなければならないことに注意してください。

```
SQL ステートメントを入力して、実行キーを押してください。
現在の接続相手はリレーショナル・データベース KC105 である。
CONNECT TO KC000
   ⋮
COMMIT
===> CONNECT TO MPLS03 USER JOE USING 'X47K'_____________________________
_____________________________________________________________________________
_____________________________________________________________________________
```

分散リレーショナル・データベースに特有の SQL および SQL CALL

分散 DB2 UDB for iSeries アプリケーションの事前コンパイル処理中に、 OS/400 プログラムは **アプリケーション・サーバー (AS)** で実行される SQL パッケージを作成することができます。コンパイル後、分散 SQL プログラムとパッケージは、アプリケーション・レシーバーおよびアプリケーション・サーバーとして使用するサーバーと互換性がなければなりません。事前コンパイル・プロセスの変更および SQL パッケージの追加に関する詳細については、 207 ページの『分散リレーショナル・データベース・プログラムの作成』で説明しています。

このセクションでは、分散リレーショナル・データベース・サポートとともに使用される SQL ステートメントの概要、および他のサーバーとの共存に関する考慮事項がいくつか示されています。これらの項目についての詳細は、iSeries Information Center の『SQL 解説書』、および『SQL プログラミング 概念』を参照してください。

分散リレーショナル・データベース・ステートメント

SQL 言語を使用して組み込まれる以下のステートメントは、特に分散リレーショナル・データベースをサポートするためのものです。

- CONNECT
- SET CONNECTION
- RELEASE
- DISCONNECT
- DROP PACKAGE
- GRANT EXECUTE ON PACKAGE
- REVOKE EXECUTE ON PACKAGE

SQL CALL ステートメントはローカルで使用することができます。ただし、その本来の目的はリモート・サーバーからプロシージャーを呼び出せるようにすることです。

190 ページの『分散リレーショナル・データベースへの接続』では、**アプリケーション・リクエスター (AR)** および **アプリケーション・サーバー (AS)** の間の接続を管理するための CONNECT、SET CONNECTION、RELEASE、および DISCONNECT ステートメントの使用方法が説明されています。 SQL GRANT EXECUTE ON PACKAGE および REVOKE EXECUTE ON PACKAGE ステートメントを使用して、 SQL パッケージに対するユーザーの権限の許可および取り消しを行う場合については、 63 ページの『分散リレーショナル・データベース・オブジェクトの権限』に説明しています。

SQL DROP PACKAGE ステートメントは、SQL パッケージを除去するのに使用するものであり、 214 ページの『SQL パッケージの処理』で説明しています。

SQL CALL ステートメント (ストアード・プロシージャー)

注: V5R1 より前の DB2 UDB for iSeries では、ストアード・プロシージャーからの結果の戻りセットはサポートしていません。 V5R1 では、DRDA サーバーに、非 iSeries クライアントからのストアード・プロシージャー呼び出しのサポートが追加されています。 V5R2 では、SQL 用の CLI インターフェースを使用するアプリケーションに、 iSeries クライアント・サイド・サポートが追加されています。しかし、V5R1 iSeries サーバーが V5R2 iSeries クライアントにストアード・プロシージャーの結果セットを戻せるようにするには、 V5R1 iSeries サーバーに PTF を適用しなければなりません。 PTF の説明は、「V5R2 iSeries DRDA クライアントへのストアード・プロシージャーの結果セットの戻りをサポートする V5R1 DRDA サーバー PTF」です。この機能に必要な V5R1 の PTF に関する詳細は、情報 APAR ii13348 を参照してください。

結果セットは、ストアード・プロシージャーで、 SQL SELECT ステートメントに関連した 1 つ以上の SQL カーソルをオープンすることによって生成できます。さらに、最高で 1 つの配列結果セットを戻すことができます。結果セットを戻すストアード・プロシージャーの作成について詳しくは、 iSeries Information Center の『SQL 解説書』にある SET RESULT SETS および CREATE PROCEDURE ステートメントの説明を参照してください。

SQL CALL ステートメントは、実際は分散リレーショナル・データベースに特有のものではありませんが、その主な価値は分散アプリケーションの論理および処理の中にあるので、ここでその説明を行います。リモート・プロシージャー呼び出し (RPC) メカニズムが Open Software Foundation** (OSF**) 分散コンピューター環境 (DCE) で提供しているものとほぼ類似した機能を、 CALL ステートメントは DRDA 環境で提供しています。事実、リモート・リレーショナル・データベースでのプログラムに対する SQL CALL は、実際にはリモート・プロシージャー呼び出しです。この RPC のタイプにはいくつかの利点があります。たとえば、インターフェース定義のコンパイル、またはスタブ・プログラムの作成は不要です。

SQL CALL (あるいは手法としてストアード・プロシージャー とも呼ばれる) を使用するのは、以下のような理由によります。

- **アプリケーション・リクエスター (AR)** と **アプリケーション・サーバー (AS)** 間のメッセージ・フローの数を減らし、与えられている機能を実行するため。 SQL 操作のセットが実行される場合、サーバーのプログラムにステートメントおよび相互接続論理を含めたほうがより効率的です。
- リモート・ロケーション固有のデータベース操作を行えるようにするため。
- SQL を使用して非データベース操作 (たとえば、メッセージの送信またはデータ待ち行列操作の実行) を行うため。

  注: データベース操作の場合と異なり、これらの操作は、サーバーによるコミットメント制御で保護されません。
- リモート・サーバーでサーバーのアプリケーション・プログラミング・インターフェース (API) にアクセスするため。

ストアード・プロシージャーとアプリケーション・プログラムは、同じ活動化グループ内で稼働することも、別個の活動化グループ内で稼働することもできます。 AR 側のアプリケーション・プログラムと AS 側のストアード・プロシージャーとの間に一貫性を持たせるため、 ACTGRP(*CALLER) を指定してストアード・プロシージャーをコンパイルすることをお勧めします。ストアード・プロシージャーが結果セットを戻すように設計されている場合は、それを *NEW 活動化グループで実行するように作成すべきではありません。そのようにすると、プロシージャーが呼び出し元に戻るときに、結果セットに関連したカーソルが早くにクローズし、活動化グループは破棄される可能性があります。

照会メッセージを発行するストアード・プロシージャーが呼び出されると、メッセージが QSYSOPR メッセージ待ち行列に送られます。ストアード・プロシージャーは照会メッセージへの応答を待ちます。ストアード・プロシージャーが照会メッセージに応答するようにするには、応答リスト項目追加 (ADDRPYLE) コマンドを使用し、ストアード・プロシージャー内の ジョブ変更 (CHGJOB) コマンドの INQMSGRPY パラメーターに *SYSRPYL を指定します。

COMMIT または ROLLBACK がデフォルトの活動化グループの AS ジョブ内で実行されている場合は、それをストアード・プロシージャーで実行することはできません。ストアード・プロシージャーおよびアプリケーション・プログラムが別々のコミットメント定義の下で実行されるときには、アプリケーション・プログラム中の COMMIT ステートメントおよび ROLLBACK ステートメントは、自らのコミットメント定義にのみ影響を及ぼします。他の手段を用いて、ストアード・プロシージャーで変更をコミットしなければなりません。

SQL CALL に関する詳細は、iSeries Information Center の『SQL 解説書』を参照してください。

DB2 Universal Database (UDB) に対する SQL CALL を使用したストアード・プロシージャーの呼び出し: DB2 UDB を実行するプラットフォームで呼び出される C で記述されたストアード・プロシージャーは、パラメーターとして argc および argv を使用することができません (つまり、main() タイプになることができません)。この点は iSeries ストアード・プロシージャーとは異なります。そこでは argc および argv を使用しなければなりません。 DB2 UDB プラットフォームのストアード・プロシージャーの例については、\SQLLIB\SAMPLES (または /sqllib/samples) サブディレクトリーを参照してください。C サブディレクトリーで outsrv.sqc および outcli.sqc を見つけてください。

iSeries server で呼び出される UDB ストアード・プロシージャーでは、プロシージャー名が大文字になっているか確認してください。 iSeries server では現在プロシージャー名を大文字に変換しています。 つまり、UDB サーバー上のプロシージャーには、同一のプロシージャーで小文字であるものはありません。iSeries server 上のストアード・プロシージャーでは、プロシージャー名は大文字です。

iSeries server 上のストアード・プロシージャーは、同じ活動化グループの中で呼び出しプログラム (ストアード・プロシージャーを作成するための適切な方法) として実行するために作成される場合、その中に COMMIT を持つことができません。 UDB では、ストアード・プロシージャーは COMMIT を持つことができますが、コミットが起こる DB2 UDB for iSeries の部分については認知できないということをアプリケーション設計者は理解している必要があります。

DB2 UDB for iSeries の共存

SQL 言語を使用して、分散リレーショナル・データベースのためのプログラムを作成および保守する場合は、分散リレーショナル・データベース・ネットワーク内の他のサーバーについて考慮する必要があります。作成および保守するプログラムは、以下のものとの間で互換性がなければなりません。

- 他の iSeries server
- 以前の iSeries server のリリース
- iSeries server 以外のサーバー

分散 SQL プログラムの中の SQL ステートメントは**アプリケーション・サーバー (AS)** 上で実行されるということに注意してください。プログラムが**アプリケーション・リクエスター (AR)** 上で実行される場合でも、 SQL ステートメントは AS 上で実行される SQL パッケージの中に入っています。それらのステートメントは AS によってサポートされ、 AS 上に存在しているコレクション、テーブル、およびビューとの間で互換性がなければなりません。また、AR 側でプログラムを実行するユーザーは、 AS 上の SQL パッケージおよびその他の SQL オブジェクトに対して許可されていなければなりません。

CRTSQLxxx コマンドを使用し、 AS に対してリレーショナル・データベース名 (RDB パラメーター) を指定して、プログラムを再度作成することによって、非分散組み込み SQL プログラムを分散組み込み SQL プログラムに変換することができます。この場合には、DB2 Universal Database for iSeries の分散リレーショナル・データベース・サポートを使用してプログラムを再度コンパイルし、 AS 上で必要とされる SQL パッケージを作成します。

iSeries server ではないアプリケーション・サーバーで実行される DB2 UDB for iSeries プログラムを作成することもできます。これらの他のプラットフォームは、多少なりとも SQL 機能をサポートすることができます。 DB2 UDB for iSeries AR でサポートされていないステートメントは、AS がその機能をサポートするときには、サーバー上で使用およびコンパイルすることができます。 本書で説明しているレベルのサポートを提供するのは、 iSeries server AS 上で実行するために作成した SQL プログラムだけです。他のシステムが提供するレベルの機能を判別したい場合には、他のシステムのサポート資料を参照してください。

DRDA 作業単位の終了

非コミット作業で SQL プログラムを終了する場合には注意が必要です。プログラムが非コミット作業で終了すると、リレーショナル・データベースへの接続は活動状態のままになります。 (システム命名の活動化グループで実行されるプログラムを組み込んでいる場合は、システムはプログラムの終了時に自動コミットを行います。)

この状態は他のシステムの場合とは異なります。つまり、OS/400 オペレーティング・システムでは、COMMIT および ROLLBACK は、コマンド行から、または CL プログラムでコマンドとして使用することができるためです。ただし、上記のシナリオではその状態についてあらかじめ計画していた場合を除き、次の SQL プログラムの実行時に予期しない結果を招く場合があります。たとえば、次に対話式 SQL (STRSQL コマンド) を実行した場合、非コミット作業を持つ前の**アプリケーション・サーバー (AS)** へ接続されている状態で対話式セッションが開始されます。別の例として、上記のシナリオの後で暗黙接続を行う 2 番目の SQL プログラムを開始した場合、最後に使用された AS 上で該当するパッケージを見つけて使用しようとする試みが行われます。この AS は予定していた AS ではない場合があります。このような不測の事態を回避するために、アプリケーション・プログラムを終了する場合は、必ずその前に最後の作業単位をコミットするかまたはロールバックするようにしてください。

ストアード・プロシージャー、ユーザー定義関数 (UDF)、およびコミットメント制御

DRDA 接続に対してコミットメント制御がアクティブになっていない状態 (COMMIT(*NONE)) で対話式 SQL などのアプリケーションが稼働している場合は、呼び出されたストアード・プロシージャーやユーザー定義関数 (UDF) が、iSeries サーバー上でコミットメント制御を開始する可能性があります。これは、クライアントとサーバーの間でコミットメント制御の不一致を生み、それによってアプリケーション終了時に更新がコミットされなくなる場合があります。

このような状態は避けなければなりません。とはいえ、もしこのような状態が発生してしまった場合には、1 つの解決方法として、コミットメント制御下で稼働しているストアード・プロシージャーや UDF に対

| し、そのデータベース更新すべてを明示的にコミットすることができます。また、これが行われない場合で
| も、 V5R3 サーバーを開始する際には、サーバーが未接続プロセスでの保留中の更新を検出し、その保留
| 中の作業を自動的にコミットします。

コード化文字セット識別子 (CCSID)

国別の言語サポートには、最小限の文字セットの適正な処理が必要です。文字情報管理のためのシステム間サポートは、 IBM 文字データ表現体系 (CDRA) によって提供されます。 CDRA では、文字を表すために使用されるコード・ポイントを識別し、それらの意味を保持するための必要に応じて、それらのコード (文字データ) を変換するためのコード化文字セット識別コード (CCSID) を定義します。

CDRA などのアーキテクチャーおよび対応する変換プロトコルの使用が重要になるのは、次のような場合です。

- 複数の国別言語バージョンが iSeries server にインストールされている場合。
- 複数の iSeries server が、異なる各国別言語の基本バージョンを使用している異なる国のシステム間でデータを共用する場合。
- iSeries server および iSeries server 以外のシステムが、異なる各国別言語の基本バージョンを使用している異なる国のシステム間でデータを共用する場合。

タグ付けは、コード化表示文字に意味を割り当てるための基本的な手段です。タグは、データ・オブジェクトに関連するデータ構造の中にある場合 (明示タグ付け) もあれば、ジョブまたはシステム事態など、オブジェクトから受け継がれる場合 (暗黙タグ付け) もあります。

DB2 UDB for iSeries は、CCSID 文字の列にタグ付けをします。 **CCSID** は、コード化体系識別コード、文字セット識別コード、コード・ページ識別コード、および使用されるコード化表示文字の表示を固有に識別する追加のコード化関連情報から成る特定のセットを識別する 16 ビットの数です。アプリケーションの実行に際して、データは別のシステムに送信されるときには変換されません。データはタグ付けされたまま CCSID とともに送信されます。受信側ジョブでは、そのデータのタグ付け方式が異なる場合、自動的にデータを独自の CCSID に変換します。

CDRA では、以下に挙げる範囲の値を CCSID に定義しています。

00000 次の階層の CCSID を使用する。

00001 から 28671
IBM 登録 CCSID

28672 から 65533
予約済み

65534 下位の階層の CCSID を参照する

65535 変換は行われない

OS/400 CCSID のリストについては、iSeries Information Center の『各国語サポート』をご覧ください。また、CDRA CCSID の完全なリストについては、*Character Data Representation Architecture - Level 1, Registry* を参照してください。 CCSID 処理についての詳細は、iSeries Information Center の『SQL 解説書』、および『SQL プログラミング 概念』を参照してください。

次の図には、CCSID の各部が示してあります。

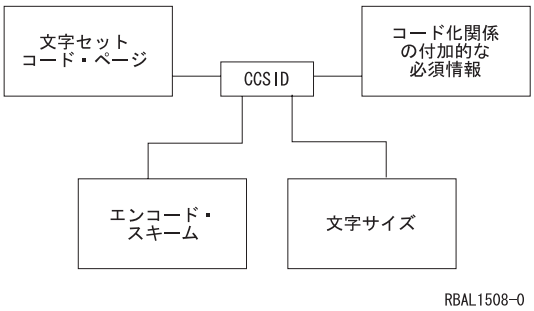


図 21. コード化文字セット識別子 (CCSID)

iSeries server のサポート

iSeries server 上のジョブのデフォルトの CCSID は、 ジョブ変更 (CHGJOB) コマンドを使用して指定されます。 CCSID がこのようにして指定されない場合には、ジョブ CCSID は、ユーザー・プロファイルの CCSID 属性から得られます。 CCSID が指定されていない場合には、システムはそれを QCCSID システム値から取得します。この QCCSID 値は最初は 65535 に設定されています。サーバーが異種システムとの分散リレーショナル・データベースにある場合は、 CCSID 65535 を使用できない場合があります。 異種環境で操作するときに考慮すべき事項については、 247 ページの『付録 B. DRDA を使用したプラットフォーム間アクセス』を参照してください。

アプリケーション・リクエスター (AR) と **アプリケーション・サーバー (AS)** の間を流れる制御情報は、すべて CCSID 500 (DRDA 標準) です。これは、コレクション名、テーブル名、および記述テキストなどの情報です。制御情報に可変レコード文字を使用すると、これらの名前が変換されることになり、パフォーマンスに影響を与える可能性があります。パッケージ名も CCSID 500 で送られます。パッケージ名の中で可変レコード文字を使用すると、パッケージ名が変換されることになります。つまり、そのパッケージが実行時に見つからないことになります。

ジョブが開始された後で、ジョブ変更 (CHGJOB) コマンドを使用することによって、ジョブの CCSID を変更することができます。そのためには、以下のようにします。

1. ジョブ処理 (WRKJOB) コマンドを入力して、「ジョブの処理」画面を表示してください。
2. オプション 2 (ジョブ定義属性の表示) を選択してください。

   これで現行 CCSID 値が見つかるので、後でジョブを元の CCSID 値にセットし直すことができます。
3. 新しい CCSID 値を指定した ジョブ変更 (CHGJOB) コマンドを入力してください。

新しい CCSID 値は即時にジョブに反映されます。ただし、変更するジョブ CCSID が AR ジョブである場合には、次の CONNECT までは新しい CCSID が実行中の作業に影響することはありません。

重要: AS ジョブの CCSID を変更した場合には、結果は予想することができません。

CCSID がソース物理ファイルの作成 (CRTSRCPF)または物理ファイルの作成 (CRTPF) コマンドでソース・ファイルに明示的に指定されない場合には、ソース・ファイルはジョブの CCSID でタグ付けされます。 CCSID がデータ記述仕様 (DDS)、対話式データ定義ユーティリティー (IDDU)、または CREATE TABLE SQL ステートメントで明示的に指定されない場合には、外部記述データベース・ファイルおよびテーブルはジョブの CCSID でタグ付けされます。

ソース・ファイルおよび外部記述ファイルでは、ジョブ CCSID が 65535 であれば、オペレーティング・システムの言語に基づいたデフォルトの CCSID が使用されます。プログラム記述ファイルは、CCSID 65535 でタグ付けされます。ビューは、対応するテーブルの CCSID または列レベル・タグでタグ付けされます。複数のテーブルにまたがって定義されているビューの場合には、列レベルでタグ付けされ、基礎を成している列のタグが想定されます。ビューは、CCSID で明示的にタグ付けすることはできません。 CCSID が等しくなく、しかもどちらの CCSID も 65535 に等しくない場合には、システムはジョブとテーブルの間でデータを自動的に変換します。

タグ付けされたテーブルの CCSID を変更すると、そのテーブルは列レベルでタグ付けすることも、ビューを定義することもできません。タグ付けされた CCSID を変更するには、物理ファイルの変更 (CHGPF) コマンドを使用します。列レベルのタグ付けがあるテーブルを変更するには、そのテーブルを再度作成し、ファイル・コピー (CPYF) コマンドの FMT(*MAP) を使用して、データを新しいテーブルにコピーしなければなりません。 テーブルに 1 つ以上のビューが定義されている場合には、テーブルを変更するには以下のようにしなければなりません。

1. ビューおよびテーブルをそれらのアクセス・パスとともに保管する。
2. ビューを削除する。
3. テーブルを変更する。
4. 作成したテーブルにビューおよびそのアクセス・パスを復元する。

DB2 Universal Database for iSeries に移行されるソース・ファイルおよび外部記述ファイルで、タグ付けされていないもの、または CCSID 65535 でタグ付けされているものは、インストールされているオペレーティング・システムの言語に基づいたデフォルトの CCSID でタグ付けされます。 このようなファイルには、新しいリリースのインストール時にシステムにあるファイル、および DB2 Universal Database for iSeries に復元されるファイルが含まれます。

AR と AS の間で送信されるデータは、すべて変換されないで送信されます。さらに、CCSID も送信されます。受信側ジョブでは、そのデータのタグ付け方式が異なる場合、自動的にデータを独自の CCSID に変換します。たとえば、販売店システム KC105 で実行される次のアプリケーションを考慮します。

注: xv ページの『コードの特記事項情報』に重要なリーガル情報がありますので、お読みください。

```
CRTSQLxxx PGM(PARTS1) COMMIT(*CHG)  RDB(KC000)

    PROC: PARTS1;
       .
       .
    EXEC SQL
      SELECT * INTO :PARTAVAIL
               FROM INVENTORY
               WHERE ITEM = :PARTNO;
       .
       .
    END PARTS1;
```

上記の例では、ローカル・システム (KC105) は QCCSID システム値が CCSID 37 に設定されています。リモート地域センター (KC000) では CCSID 937 を使用し、そのテーブルはすべて CCSID 937 でタグ付けされています。 CCSID 処理は次のようにして行われます。

- KC105 システムが入力ホスト変数 (:PARTNO) を CCSID 37 で送信します。 (ジョブの CCSID がホスト変数に適さない場合には、 DECLARE VARIABLE SQL ステートメントが使用できます。)
- KC000 システムが :PARTNO を CCSID 937 に変換し、必要なデータを選択し、そのデータを CCSID 937 で KC105 に送り返します。

- KC105 では、そのデータを入手すると、 CCSID 37 に変換し、ローカル使用に備えて :PARTAVAIL に入れます。

その他の DRDA データ変換

リモート・システムで処理を行っている場合には、時として、プログラムがデータを 1 つのシステムから変換して、他方のシステムでも使用できるようにする必要が生じることがあります。 iSeries server 上の DRDA サポートは、 DRDA サポートを使用する他のシステムとの間でデータを自動的に変換します。 DB2 Universal Database for iSeries **アプリケーション・リクエスター (AR)** は、**アプリケーション・サーバー (AS)** に接続するとき、そのタイプを識別する情報を送信します。同様にして、AS では、その処理装置タイプ (たとえば、 S/390* ホストまたは iSeries server) を識別する情報をサーバーに送り返します。 次いで、2 つのシステムは、この接続で定義されているように、相互間でデータを自動的に変換します。つまり、システム間の体系的な違いについてはプログラムで考慮する必要はないことを意味します。

DRDA サポートを使用する IBM システム間でのデータ変換には、以下のようなデータ・タイプが含まれます。

- 浮動小数点表示
- ゾーン 10 進数表示
- バイト反転
- 混合データ・タイプ
- 次のような iSeries 固有データ・タイプ
  - DBCS 専用
  - DBCS 択一
  - 精度および位取り付き整数

DDM ファイルと SQL

iSeries DDM サポートを使用して、SQL 分散リレーショナル・データベース・サポートを使用するプログラム内で分散リレーショナル・データベース・タスクを実行するのに役立てることができます。 たとえば、大量のレコードを取り出すには、 DDM およびファイル・コピー (CPYF) コマンドを使用する方が、SQL FETCH ステートメントを使用するよりも迅速にできる場合があります。 また、DDM を使用すると、コンパイル時に取り込まれたリモート・システム・データの外部ファイル記述を、分散リレーショナル・データベース・アプリケーションで使用するために取り出すことができます。そのためには、『第 3 章 iSeries 分散リレーショナル・データベースのための通信』および『第 5 章 iSeries 分散リレーショナル・データベースのセットアップ』で説明してあるように DDM を使用する必要があります。

次の例には、リレーショナル・データベース・ディレクトリー項目を追加し、 DDM ファイルを作成して、**アプリケーション・サーバー (AS)** と**アプリケーション・リクエスター (AR)** で同じジョブが使用できるようにする方法を示してあります。

注: 共用される会話に関しては、両方の接続が保護であるか、または両方の接続が非保護であるかどちらかでなければなりません。

注: xv ページの『コードの特記事項情報』に重要なリーガル情報がありますので、お読みください。

Relational Database Directory:

```
ADDRDBDIRE    RDB(KC000) +
              RMTLOCNAME(KC000)
              TEXT('Kansas City regional database')
```

DDM File:

```
CRTDDMF  FILE(SPIFFY/UPDATE)
         RMTFILE(SPIFFY/INVENTORY)
         RMTLOCNAME(KC000)
         TEXT('DDM file to update local orders')
```

以下に挙げるのは、リモート・サーバー上の同じジョブの中で、リレーショナル・データベース・ディレクトリー項目と DDM ファイルの両方を使用するプログラム例です。

```
CRTSQLxxx PGM(PARTS1) COMMIT(*CHG)  RDB(KC000) RDBCNNMTH(*RUW)

    PROC :PARTS1;
    OPEN  SPIFFY/UPDATE;
       .
       .
       .
    CLOSE SPIFFY/UPDATE;
       .
       .
       .
    EXEC SQL
      SELECT * INTO :PARTAVAIL
               FROM INVENTORY
               WHERE ITEM = :PARTNO;
    EXEC SQL
      COMMIT;
       .
       .
       .
    END PARTS1;
```

iSeries DDM サポートの使用方法についての詳細は、 iSeries Information Center の『分散データ管理』を参照してください。

分散リレーショナル・データベース・プログラムの作成

SQL 言語を使用してプログラムを作成するときは、ホスト・プログラムに SQL ステートメントを組み込むことができます。**ホスト・プログラム**とは、次に挙げる**ホスト言語**のいずれか 1 つで書かれ、 SQL ステートメントが含まれているプログラムのことです。すなわち、iSeries PL/I、 ILE C/400®、COBOL/400、 ILE COBOL/400、 FORTRAN/400、 RPG/400、または ILE RPG/400 プログラミング言語です。 ホスト・プログラムの中では、**ホスト変数**と呼ばれる変数を使用します。 これらは、SQL ステートメントの中で使用され、ホスト・プログラムに対して識別可能な変数のことです。 RPG では、フィールド名と呼ばれています。これらは FORTRAN、PL/I、および C では変数といい、COBOL ではデータ項目と呼ばれます。

分散 DB2 Universal Database for iSeries プログラムは、分散ではない DB2 UDB for iSeries プログラムの場合と似た方法でコーディングすることができます。 ホスト言語を使用して、SQL ステートメントをホスト変数とともに組み込みます。また、分散でない DB2 UDB for iSeries プログラムの場合と同じように分散 DB2 UDB for iSeries プログラムを次のようなプロセスを使用して準備します。

- 事前コンパイル
- テストとデバッグ
- アプリケーションのバインド
- アプリケーション・プログラムのコンパイル

ただし、分散 DB2 UDB for iSeries プログラムでは、データにアクセスするために SQL パッケージも**アプリケーション・サーバー (AS)** 上で作成しなければなりません。

このセクションでは、分散 DB2 UDB for iSeries プログラムの場合の違いを概説し、上記のプロセスのステップを説明します。

SQL ステートメントを組み込んだプログラムの事前コンパイル

組み込み SQL ステートメントが入っているアプリケーション・プログラムは、その実行に先立って、事前コンパイルとコンパイルが必要です。 そのようなプログラムの事前コンパイルは、SQL コンパイラーによって行われます。 SQL 事前コンパイラーは、アプリケーション・プログラムの各ステートメントを走査し、以下のことを行います。

- SQL ステートメント、およびホスト変数名の定義を検索する
- 各 SQL ステートメントが有効で、構文エラーがないかどうかを検査する
- データベースの中の記述を使用して、SQL ステートメントの妥当性検査をする
- 各 SQL ステートメントをホスト言語でのコンパイルができるように準備する
- 各事前コンパイル済み SQL ステートメントについての情報を作成する

アプリケーション・プログラミングのステートメントと組み込み SQL ステートメントが、 SQL 事前コンパイラーに対する 1 次入力になります。 SQL プリコンパイラーでは、ホスト言語ステートメントが構文上正しいことを前提としています。ホスト言語ステートメントが構文上正しくない場合には、事前コンパイラーは SQL ステートメントおよびホスト変数宣言を正しく識別できないことがあります。

SQL 事前コンパイル処理では、リストと一時ソース・ファイル・メンバーが作られます。また、事前コンパイラー・コマンドの OPTION および RDB パラメーターに指定される値によっては、 SQL パッケージが作られます。このコマンドの詳細については、 210 ページの『アプリケーション・プログラムのコンパイル』を参照してください。

リスト

出力リストは、 CRTSQL*xxx* コマンドの PRTFILE パラメーターで指定された印刷装置ファイルに送られます。以下の項目が印刷装置ファイルに書き込まれます。

- 事前コンパイラー・オプション

  これは、CRTSQL*xxx* コマンドで指定されたすべてのオプション、およびソース・メンバーが最後に変更された日付のリストです。

- 事前コンパイラー・ソース

  この出力が作成されるのは、非 ILE 事前コンパイラーに *SOURCE オプションが、または ILE 事前コンパイラーに OUTPUT(*PRINT) パラメーターが使用されている場合です。 各事前コンパイラー・ソース・ステートメントが、事前コンパイラーによって割り当てられたそのレコード番号、原始ステートメント入力ユーティリティー (SEU) の使用時に表示される順序番号 (SEQNBR)、およびレコードが最後に変更された日付とともに示されます。

- 事前コンパイラー相互参照

  この出力が作成されるのは、OPTION パラメーターに *XREF が指定された場合です。ホスト変数または SQL エンティティー (テーブルおよび列など) の名前、その名前が定義されているレコード番号、その名前が定義されていること、および名前が現れるレコード番号が示されます。

- 事前コンパイラー診断リスト

この出力は診断メッセージを提供し、エラー状態のステートメントの事前コンパイラー・レコード番号を示します。

一時ソース・ファイル・メンバー

事前コンパイラーが処理したソース・ステートメントは、 QTEMP ライブラリーの QSQLTEMP (CRTSQLRPGI を使用して作成されたプログラムでは QTEMP ライブラリーの QSQLTEMP1) に書き込まれます。事前コンパイラーによって変更されたソース・コードでは、SQL ステートメントは、 SQL インターフェース・モジュール QSQROUTE、QSQLOPEN、QSQLCLSE、および QSQLCMIT に対する注記および呼び出しに変換されています。一時ソース・ファイル・メンバーの名前は、 CRTSQL*xxx* の PGM パラメーターに指定されている名前と同じです。このメンバーは、コンパイラーに対する入力として使用される前には変更できません。

コンパイルを後にしたい場合には、 QSQLTEMP または QSQLTEMP1 を事前コンパイルの後で永続ライブラリーに移すことができます。一時ソース・ファイル・メンバーのレコードを変更する場合には、コンパイルを後で試みると、正常に行われません。

SQL パッケージの作成

CRTSQLxxx コマンドのコンパイル時に、事前コンパイル・プロセスの一部として、 SQL パッケージと呼ばれるオブジェクトを作成することができます。これらの処理の一部としてのパッケージ作成に影響する状況の説明については、 210 ページの『アプリケーション・プログラムのコンパイル』および 210 ページの『アプリケーションのバインド』を参照してください。 SQL パッケージ、およびパッケージを処理するのに使用できるコマンドの詳細については、 214 ページの『SQL パッケージの処理』を参照してください。

事前コンパイラー・コマンド

DB2 UDB Query Manager and SQL Development Kit プログラムには、各ホスト言語に対して 1 つずつ、7 つの事前コンパイラーがあります。

ホスト言語	コマンド
iSeries PL/I	CRTSQLPLI
ILE C/400 言語	CRTSQLCI
COBOL/400 言語	CRTSQLCBL
ILE COBOL/400 言語	CRTSQLCBLI
FORTRAN/400 言語	CRTSQLFTN
RPG III (RPG/400 言語の一部)	CRTSQLRPG
ILE RPG/400 言語	CRTSQLRPGI

各言語ごとに別のコマンドが存在しているので、各言語はその言語だけに適用されるパラメーターを持つことができます。たとえば、オプション *APOST および *QUOTE は COBOL に固有のものです。これらは他の言語用のコマンドには含まれていません。事前コンパイラーは、 SQL 事前コンパイラー・コマンドの 1 つが呼び出すときに指定するパラメーターによって制御されます。これらのパラメーターは、入力の処理方法および出力の提示方法を指定します。

プログラムの事前コンパイルには、プログラム・ソース・ステートメントを含んでいるメンバーの名前を CRTSQL*xxx* コマンドの PGM パラメーター (非 ILE 事前コンパイラーの場合)、または OBJ パラメーター (ILE 事前コンパイラーの場合) として指定する以外は、何も指定する必要がありません。 SQL は、すべての事前コンパイラー・パラメーターに対してデフォルト値を割り当てます (ただし、値を明示的に指定すれば、デフォルト値を指定変更することができます)。

以下に、分散リレーショナル・データベースをサポートするために使用される、すべての CRTSQLxxx コマンドに共通なパラメーターを簡単に説明します。パラメーターの構文と記述、およびサポートされる値については、『SQL プログラミング 概念』を参照してください。

RDB
SQL パッケージ・オプションを作成するリレーショナル・データベースの名前を指定します。 *NONE を指定すると、プログラムまたはモジュールは非分散オブジェクトになり、 構造化照会言語パッケージの作成 (CRTSQLPKG) コマンドは使用できなくなります。リレーショナル・データベースの名前は、ローカル・データベースの名前とすることができます。

RDBCNNMTH
CONNECT ステートメントの働き方のタイプ、すなわち、リモート作業単位 (RUW) または分散作業単位 (DUW) の別を指定します。

SQLPKG
SQL パッケージの名前とライブラリーを指定します。

USER
会話の開始時にリモート・サーバーに送られるユーザー名を指定します。このパラメーターは、会話が事前コンパイル・プロセスの一部として開始される場合だけ使用します。

PASSWORD
会話を開始するときに、リモート・サーバーで使用されるパスワードを指定します。このパラメーターは、会話が事前コンパイル・プロセスの一部として開始される場合だけ使用します。

REPLACE
事前コンパイルの一部として作成されるオブジェクトが、既存のオブジェクトを置き換えられるようにする場合に指定します。

以下の例では、INVENT という名前の COBOL プログラムを作成し、それを SPIFFY という名前のライブラリーに保管します。 SQL 命名規則が選択され、指定されたテーブルから選択された行は、すべてが回復単位の終了までロックされます。プログラムと同じ名前の SQL パッケージが、 KC000 という名前のリモート・リレーショナル・データベースに作成されます。

```
CRTSQLCBL PGM(SPIFFY/INVENT) OPTION(*SRC *XREF *SQL)
          COMMIT(*ALL) RDB(KC000)
```

アプリケーション・プログラムのコンパイル

DB2 Universal Database for iSeries 事前コンパイラーは、事前コンパイルが正常に終了すると、 *NOGEN オプションが指定されていなければ、自動的にホスト言語コンパイラーを呼びます。 このコンパイラー・コマンドは、プログラム名、ソース・ファイル名、事前コンパイラー作成のソース・メンバー名、テキスト、およびユーザー・プロファイルを指定して、実行されます。その他のパラメーターも、ホスト言語に応じて、コンパイラーに渡されます。

これらのパラメーターについての詳細は、 iSeries Information Center の『SQL プログラミング 概念』を参照してください。

アプリケーションのバインド

アプリケーション・プログラムで実行できるようにするには、その前にプログラムと参照されるテーブルおよびビューとの間の関係が確立されていなければなりません。この処理が**バインド**と呼ばれるものです。バインドの結果がアクセス・プランです。 **アクセス・プラン**は、各 SQL 要求を満たすのに必要なアクションを記述する制御構造です。アクセス・プランには、プログラムおよびプログラムで使用するデータに関する情報が入ります。分散リレーショナル・データベース作業では、アクセス・プランは SQL パッケージの

中に保管され、 SQL パッケージとともにサーバーによって管理されます。 SQL パッケージの詳細については、 214 ページの『SQL パッケージの処理』を参照してください。

コンパイルが成功して、プログラムまたはサービス・プログラムのオブジェクトが作成されると、 SQL は自動的にアクセス・プランのバインドと作成を試みます。コンパイルが正常に行われなかった場合、またはコンパイルの結果がモジュール・オブジェクトでない場合は、アクセス・プランは作成されません。実行時に、アクセス・プランが有効でないこと、またはパフォーマンスを向上させるような変更 (たとえば、索引の追加) がデータベースに加えられたことをデータベース・マネージャーが検出した場合には、新規アクセス・プランが自動的に作成されます。**アプリケーション・サーバー (AS)** が iSeries server ではない場合には、構造化照会言語パッケージの作成 (CRTSQLPKG) コマンドを使用して、バインドをやり直さなければなりません。バインドでは、以下の 3 つのことを行います。

- データベースの中の記述を使用して、SQL ステートメントの妥当性を再検査する。

  バインド・プロセス中に、SQL ステートメントは、テーブル、ビューおよび列名が有効かどうかを検査されます。参照されたテーブルまたはビューが事前コンパイル時またはコンパイル時に存在していない場合には、妥当性検査は実行時に行われます。テーブルまたはビューが実行時に存在していない場合には、負の SQLCODE が戻されます。
- プログラムで処理したいデータにアクセスするために必要なアクセス・パスを選択する。

  アクセス・パスの選択にあたっては、SQL がアクセス・プランを作成する際に、索引、テーブル・サイズ、およびその他の要因が考慮されます。バインド処理では、データにアクセスするのに使用できるすべての索引を考慮し、データへの経路の選択にあたって使用する索引を (ある場合) 決定します。
- アクセス・プランを作成しようと試みる。

  すべての SQL ステートメントが有効である場合には、バインド処理はアクセス・プランを作成し、それをプログラムの中に保管します。

プログラムがアクセスするテーブルまたはビューの特性が変更された場合には、アクセス・プランは有効ではなくなります。有効でないアクセス・プランを使用しようとすると、サーバーは自動的にアクセス・プランの再作成を試みます。 アクセス・プランが再作成できない場合には、負の SQLCODE が戻されます。この場合には、プログラムの SQL ステートメントを変更し、 CRTSQL*xxx* コマンドを再度発行して、状況を修正しなければならないことがあります。

たとえば、プログラムの中に TABLEA の中の COLUMNA を参照する SQL ステートメントが入っていて、しかもユーザーが TABLEA を削除および再作成したために COLUMNA が存在していない場合には、プログラムを呼び出したときに、COLUMNA が存在していないために、自動再作成は正常に行われません。プログラムのソースを変更し、 CRTSQL*xxx* コマンドを再度発行しなければなりません。

テストとデバッグ

分散 SQL プログラムのテストとデバッグは、ローカル SQL プログラムのテストとデバッグに類似していますが、プロセスの面では、一部異なっているところがあります。

テストには、最終的に 1 つ以上のサーバーが必要になります。プログラムの再コンパイル、プログラムに対する入力パラメーターの変更、プログラムのソースの小さな修正を行うことによって、リレーショナル・データベースが容易に変更できるようにアプリケーションがコーディングされている場合には、ほとんどのテストは単一のサーバーを使用して実施することができます。

プログラムはローカル・データでテストした後で、分散リレーショナル・データベース・ネットワーク上での最終的なテストを行えるようにします。アプリケーションをリモート接続でテストするときは、**アプリケ**

ーション・サーバー (AS) となるサーバーでそのアプリケーションをローカルにテストし、テストを分散環境に移すときはプログラムを移しさえすればよい、という方法を考えてください。

分散 SQL プログラムのデバッグは、ローカル SQL プログラムのデバッグと同じ技法を使用します。 デバッグの開始 (STRDBG) コマンドを使用して、デバッガーを始動させ、アプリケーションをデバッグ・モードにします。 ブレークポイントの追加、ステートメントのトレース、および変数の内容の表示を行うことができます。

ただし、分散 SQL プログラムをデバッグする場合には、 UPDPROD パラメーターに *YES の値を指定しなければなりません。その理由は、OS/400 分散リレーショナル・データベース・サポートではライブラリー QSYS の中のファイルを使用し、 QSYS は実動ライブラリーだからです。これにより、実動ライブラリーの中のデータを**アプリケーション・リクエスター (AR)** 上で変更することができます。 AR 上でデバッグの開始 (STRDBG) コマンドを発行しても、デバッグ・モードになるのは AR ジョブだけであり、 AS 上でのデータ操作能力が変化することはありません。

AR 上でのデバッグ・モード中は、実行された各 SQL ステートメントごとに、通知メッセージがジョブ・ログに入れられます。これらのメッセージは、各 SQL ステートメントの結果について情報を与えるものです。分散リレーショナル・データベースでの SQL 戻りコードのリストおよびエラー・メッセージのリストは、 『第 9 章 分散リレーショナル・データベースの問題の処理』で挙げられています。

サーバーが SQL ステートメントの処理効率を最大にする方法を示す通知メッセージも、デバッグ・モードの結果として発行されます。 効率の最大化は AS 側で行われるので、これらのタイプのメッセージは AR ジョブ・ログの中には現れません。この情報を得るためには、AS ジョブがデバッグ・モードでなければなりません。

TCP/IP を使用している場合、サーバーでデバッグ・モードを開始する比較的簡単な方法は、QRWOPTIONS データ域を使用する方法です。しかし、この機構では、特定のプログラムを指定してデバッグすることはできません。セットアップの詳細については、 QRWOPTIONS データ域の使用法を参照してください。データ域はデバッグを開始するためだけでなく、ジョブ・トレースを開始し、ジョブ・ログを要求してジョブ出力を表示するためや、その他の事柄を行うためにも使用できます。 iSeries AR 上で QRWOPTIONS セットアップを行って、 iSeries サーバー上ではこのオプションを隠すこともできます。

AR および AS が iSeries server であり、それらが APPC に接続している場合には、 SBMRMTCMD (リモート・コマンド投入) コマンドを使用して、 AS ジョブでのデバッグ・モードを開始することができます。 83 ページの『DDM ファイルのセットアップ』で説明しているように、DDM ファイルを作成してください。 DDM ファイルの中の通信情報は、アクセスされるリレーショナル・データベースに関するリレーショナル・データベース・ディレクトリー項目の中の情報に一致しなければなりません。そこで、次のコマンドを入力してください。

```
SBMRMTCMD CMD('STRDBG UPDPROD(*YES)') DDMFILE(ddmfile name)
```

(SBMRMTCMD) コマンドは、AS がまだ存在していない場合には、 AS ジョブを始動させ、そのジョブでデバッグ・モードを開始させます。 95 ページの『リレーショナル・データベース活動のモニター』に説明してある方法を使用して AS ジョブ・ログを調べ、ジョブを見つけてください。

詳細については、 200 ページの『SQL CALL ステートメント (ストアード・プロシージャー)』を参照してください。

AS ジョブをデバッグ・モードにする以下の方法は、どの AR および DB2 Universal Database for iSeries AS でも使用できますが、特定の制限があります。この方法は、アプリケーションがセットアップをするた

めに接続を行った後で、一時停止できることを前提としています。また、この方法では、トレースあるいはデバッグする事柄が、接続が確立された後に発生することを想定しています。

- AS にサインオンして、AS ジョブを見つけてください。
- 下に示すように、サービス・ジョブの開始 (STRSRVJOB) コマンドを対話式ジョブ (AS ジョブを見つけるために使用するジョブ) から出してください。

  ```
  STRSRVJOB (job-number/user-ID/job-name)
  ```

  (STRSRVJOB) コマンドのジョブ名は AS ジョブの名前です。 このコマンドを出せば、AS ジョブに影響する特定のコマンドを対話式ジョブから出すことができます。そのようなコマンドの 1 つにデバッグの開始 (STRDBG) コマンドがあります。
- 対話式ジョブで UPDPROD パラメーターに *YES の値を使用して、 (STRDBG) コマンドを発行してください。 これで AS ジョブはデバッグ・モードになり、 AS ジョブ・ログにデバッグ・メッセージが作られます。

このデバッグ・セッションを終了させるには、サインオフすることによって対話式ジョブを終了するか、またはデバッグの終了 (ENDDBG) コマンドに続いて サービス・ジョブの終了 (ENDSRVJOB) コマンドを使用してください。

SQL ステートメントを実行する前に、AS ジョブをデバッグにしなければならないので、アプリケーションを変更して AS 上でデバッグをセットアップする時間を確保しなければなりません。 AS ジョブは、AS へのアプリケーションの接続の結果として開始します。アプリケーションは AS への接続後、デバッグが AS で開始されるまで、待ち状態に入るようにコーディングすることができます。

TCP/IP 接続で使用される事前開始ジョブが起きる前、たとえば作業の待ちが 1 つしかない場合や他のクライアントからの妨害がない場合などに、それを予想できる場合、遅延を行う必要はありません。

プログラム参照

プログラムが作成されると、OS/400 ライセンス・プログラムは、 SQL ステートメントの中で参照されるすべてのコレクション、テーブル、ビュー、 SQL パッケージおよび索引についての情報を SQL プログラムに保管します。

プログラム参照表示 (DSPPGMREF) コマンドを使用して、プログラムの中のすべてのオブジェクト参照を表示することができます。 SQL 命名規則が使用されている場合には、ライブラリー名は次の 3 つの方法のいずれか 1 つで保管されます。

- SQL 名が完全修飾されている場合には、コレクション名が名前修飾子として保管されます。
- SQL 名が完全修飾されておらず、DFTRDBCOL パラメーターが指定されていない場合には、ステートメントの権限 ID が名前修飾子として保管されます。
- SQL 名が完全修飾されておらず、DFTRDBCOL パラメーターが指定されている場合には、DFTRDBCOL パラメーターで指定されたコレクション名が名前修飾子として保管されます。

サーバー命名規則が使用されている場合には、ライブラリー名は次の 3 つの方法のいずれか 1 つで保管されます。

- オブジェクト名が完全修飾されている場合には、ライブラリー名が名前修飾子として保管されます。
- オブジェクトが完全修飾されておらず、 DFTRDBCOL パラメーターが指定されていない場合には、*LIBL が保管されます。
- SQL 名が完全修飾されておらず、DFTRDBCOL パラメーターが指定されている場合には、DFTRDBCOL パラメーターで指定されたコレクション名が名前修飾子として保管されます。

SQL パッケージの処理

SQL パッケージは、特に分散リレーショナル・データベース・アプリケーションによって使用される SQL オブジェクトです。これには、**アプリケーション・サーバー (AS)** 上のデータにアクセスする各 SQL ステートメントの制御構造が含まれます。これらの制御構造は、アプリケーション・プログラムが SQL ステートメントを使用してデータを要求する実行時に、 AS によって使用されます。

SQL パッケージ作成用の SQL ステートメントはないので、制御言語 (CL) コマンドを使用して SQL パッケージを作成しなければなりません。 SQL パッケージは、以下の 2 つの方法で作成することができます。

- RDB パラメーターにリレーショナル・データベース名を指定した CRTSQL*xxx* コマンドを使用する。 208 ページの『SQL ステートメントを組み込んだプログラムの事前コンパイル』を参照してください。
- SQL パッケージの作成 (CRTSQLPKG) コマンドの使用。

SQL パッケージの作成の他に、以下を行うことができます。

- 『SQL パッケージの管理』
- DLTSQLPKG コマンドを使用した SQL パッケージの削除
- SQL DROP PACKAGE ステートメントの使用

SQL パッケージの作成 (CRTSQLPKG) コマンドの使用

アプリケーション・サーバー (AS) に SQL パッケージを作成するのに、 DB2 UDB Query Manager and SQL Development Kit ライセンス・プログラムは必要ありません。 SQL パッケージの作成 (CRTSQLPKG) コマンドを入力して、コンパイル済みの分散リレーショナル・データベース・プログラムから SQL パッケージを作成することができます。また、このコマンドを使用すれば、前に作成されていた SQL パッケージを置き換えることもできます。新規 SQL パッケージは、 RDB パラメーターによって定義されたリレーショナル・データベース上に作成されます。新規 SQL パッケージは、CRTSQL*xxx* コマンドの PKG パラメーターで指定されているものと同じ名前を持ち、同じライブラリーに入れられます。

詳細は、『制御言語 (CL)』トピックの SQL パッケージの作成コマンドの項を参照してください。

SQL パッケージの管理

SQL パッケージの作成後は、iSeries server 上の他のオブジェクトの管理と同じ要領で、 SQL パッケージを管理することができます。 それを保管および復元し、他のサーバーへ送り、パッケージに対するユーザーの権限の認可および取り消しを行うことができます。また、構造化照会言語パッケージの削除 (DLTSQLPKG) コマンドまたは DROP PACKAGE SQL ステートメントを入力することによって、 SQL パッケージを削除することもできます。

分散 SQL プログラムが作成されると、 SQL パッケージの名前および内部整合性トークンがプログラムに保管されます。これらは、SQL パッケージを見つけ、 SQL パッケージがそのプログラムにとって正しいかどうかを検査するために実行時に使用されます。 SQL パッケージの名前は SQL プログラムの実行にとって重要なので、 SQL パッケージを別のライブラリーに移動、名前変更、複写、または復元することはできません。

SQL パッケージの削除 (DLTSQLPKG) コマンド

構造化照会言語パッケージの削除 (DLTSQLPKG) コマンドを使用して、 1 つまたは複数の SQL パッケージを削除することができます。削除する SQL パッケージが存在している iSeries server で、 (DLTSQLPKG) コマンドを入力しなければなりません。

詳細は、『制御言語 (CL)』トピックの SQL パッケージの削除コマンドの項を参照してください。

SQL パッケージに対しては *OBJEXIST 権限が、また SQL パッケージが入っているコレクションに対しては、少なくとも *EXECUTE 権限が必要です。

パッケージを除去するためには、SQL を使用する方法も何種類かあります。

- DB2 UDB Query Manager and SQL Development Kit ライセンス・プログラムをインストールしている場合は、対話式 SQL を使用して**アプリケーション・サーバー (AS)** を接続し、次に SQL DROP PACKAGE ステートメントを使用してパッケージを除去します。
- 接続する SQL プログラムを実行し、次にパッケージを除去します。
- Query 管理機能を使用してパッケージを接続し、除去します。

以下のコマンドは、SPIFFY コレクションの中の SQL パッケージ PARTS1 を削除します。

```
DLTSQLPKG  SQLPKG(SPIFFY/PARTS1)
```

リモート iSeries server 上の SQL パッケージを削除するには、 SBMRMTCMD (リモート・コマンド投入) コマンドを使用して、リモート・サーバーに対して構造化照会言語パッケージの削除 (DLTSQLPKG) コマンドを実行してください。また、ディスプレイ・パススルーを使用して、リモート・サーバーにサインオンし、 SQL パッケージを削除することもできます。 リモート・サーバーが iSeries server ではない場合には、リモート・ワークステーション・プログラムを使用してそのサーバーにパススルーしてから、 SQL パッケージ削除コマンドをそのサーバーにローカルで投入してください。

SQL DROP PACKAGE ステートメント

DROP PACKAGE ステートメントには、分散リレーショナル・データベースに関する PACKAGE パラメーターが組み込まれています。 DROP PACKAGE ステートメントは、プログラムに組み込んで発行することも、対話式 SQL を使用して発行することもできます。 DROP PACKAGE を発行すると、SQL パッケージとその記述が**アプリケーション・サーバー (AS)** から削除されます。この結果は、構造化照会言語パッケージの作成 (DLTSQLPKG) コマンドをローカル・サーバーに入力した場合と同じになります。 SQL パッケージに依存している他のオブジェクトが、このステートメントの結果として削除されることはありません。

SQL パッケージを正常に削除するには、その SQL パッケージに対して次の特権を持っていなければなりません。

- 参照コレクションに対するシステム権限 *EXECUTE
- SQL パッケージに対するシステム権限 *OBJEXIST

以下の例には、DROP PACKAGE ステートメントを発行する方法が示してあります。

```
DROP PACKAGE  SPIFFY.PARTS1
```

プログラムは、現在使用している SQL パッケージに対して DROP PACKAGE ステートメントを出すことはできません。

付録 A. アプリケーション・プログラミング例

この付録には、RPG/400、COBOL/400 および ILE C/400 プログラミング言語で作成された、分散リレーショナル・データベースを使用する RUW のアプリケーション例が示してあります。この例では、分散リレーショナル・データベースを作業に使用する方法を示しています。

分散リレーショナル・データベース例の業務上の要件

この例の分散リレーショナル・データベース用のアプリケーションは、自動車販売店または代理店ネットワーク内での部品在庫管理です。

このプログラムでは、地区部品在庫テーブルの各部品ごとに在庫のレベルを検査します。レベルが追加発注点以下である場合には、プログラムは中央のテーブルを検査して未処理のままになっている既存発注の有無、および各発注に対して出荷されている数量を調べます。

正味数量 (地区在庫 + 発注量 - 出荷量) が追加発注点以下である場合には、中央サーバーの該当するテーブルに行を挿入することによって、該当の部品の発注を行います。 地区サーバー側で報告書が印刷されます。

技術上の注記

コミットメント制御

このプログラムでは、ローカルおよびリモートの論理作業単位 (LUW) の概念を使用しています。このプログラムでは、リモート作業単位を使用しているので、別のサーバーで新しい作業単位を開始する前に、このサーバーの現行 LUW をクローズ (COMMIT) する必要があります。

カーソル位置変更

LUW がコミットされて、アプリケーションが別のデータベースに接続すると、すべてのカーソルはクローズされます。このアプリケーションでは、部品在庫ファイルを読み取るカーソルは、次の部品番号で再オープンされる必要があります。このことを実現するために、カーソルは、部品番号が部品番号の現行値より大きいところで始まり、部品番号によって順序付けられるように定義されています。

注: 同じ部品番号に重複行がある場合には、この技法は機能しません。

この例の詳細は、以下のトピックを参照してください。

- 例: コレクションおよびテーブルの作成
- 例: テーブルへのデータの挿入
- 例: RPG プログラム
- 例: COBOL プログラム
- 例: C プログラム
- 例: プログラム出力

例: コレクションおよびテーブルの作成

注: xv ページの『コードの特記事項情報』に重要なリーガル情報がありますので、お読みください。

```
5738PW1 V2R1M1  920327                SEU SOURCE LISTING          03/29/92 17:16:50           PAGE    1
ソース・ファイル . . . .  DRDA/QLBLSRC
メンバー . . . . . .  CRTDB
SEQNBR*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 ...+... 9 ...+... 0
100        IDENTIFICATION DIVISION.                                                          03/29/92
200        PROGRAM-ID. CRTDB.                                                                03/29/92
300        ENVIRONMENT DIVISION.                                                             03/29/92
400        DATA DIVISION.                                                                    03/29/92
500        WORKING-STORAGE SECTION.                                                          03/29/92
600             EXEC SQL INCLUDE SQLCA END-EXEC.                                             03/29/92
700        PROCEDURE DIVISION.                                                               03/29/92
800        MAIN.                                                                             03/29/92
900       *    ------------------------------------------------------------                 03/29/92
1000       *  LOCATION TABLE                                                                03/29/92
1100       *  ------------------------------------------------------------*/--               03/29/92
1200             EXEC SQL                                                                    03/29/92
1300              CREATE COLLECTION DRDA                                                     03/29/92
1400             END-EXEC.                                                                   03/29/92
1500            EXEC SQL                                                                    03/29/92
1600              CREATE TABLE DRDA/PART_STOCK                                              03/29/92
1700                    (PART_NUM    CHAR(5)      NOT NULL,
1800                     PART_UM     CHAR(2)      NOT NULL,
1900                     PART_QUANT  INTEGER      NOT NULL WITH DEFAULT,                   03/29/92
2000                     PART_ROP    INTEGER      NOT NULL,                                03/29/92
2100                     PART_EOQ    INTEGER      NOT NULL,                                03/29/92
2200                     PART_BIN    CHAR(6)      NOT NULL WITH DEFAULT                    03/29/92
2300                    ) END-EXEC.                                                         03/29/92
2400             EXEC SQL                                                                   03/29/92
2500             CREATE UNIQUE INDEX DRDA/PART_STOCI                                        03/29/92
2600                    ON DRDA/PART_STOCK                                                  03/29/92
2700                    (PART_NUM ASC) END-EXEC.                                            03/29/92
2800             EXEC SQL                                                                   03/29/92
2900              CREATE TABLE DRDA/PART_ORDER                                              03/29/92
3000                    (ORDER_NUM   SMALLINT     NOT NULL,
3100                     ORIGIN_LOC  CHAR(4)      NOT NULL,
3200                     ORDER_TYPE  CHAR(1)      NOT NULL,
3300                     ORDER_STAT  CHAR(1)      NOT NULL,
3400                     NUM_ALLOC   SMALLINT     NOT NULL WITH DEFAULT,
3500                     URG_REASON  CHAR(1)      NOT NULL WITH DEFAULT,
3600                     CREAT_TIME  TIMESTAMP    NOT NULL,
3700                     ALLOC_TIME  TIMESTAMP,
3800                     CLOSE_TIME  TIMESTAMP,
3900                     REV_REASON  CHAR(1)                                                03/29/92
4000                    ) END-EXEC.                                                         03/29/92
4100             EXEC SQL                                                                   03/29/92
4200               CREATE UNIQUE INDEX DRDA/PART_ORDEI                                      03/29/92
4300                     ON DRDA/PART_ORDER                                                 03/29/92
4400                     (ORDER_NUM ASC) END-EXEC.                                          03/29/92
4500             EXEC SQL                                                                   03/29/92
4600              CREATE TABLE DRDA/PART_ORDLN                                              03/29/92
4700                    (ORDER_NUM   SMALLINT     NOT NULL,
4800                     ORDER_LINE  SMALLINT     NOT NULL,
4900                     PART_NUM    CHAR(5)      NOT NULL,
5000                     QUANT_REQ   INTEGER NOT NULL,                                      03/29/92
5100                     LINE_STAT   CHAR(1)      NOT NULL                                  03/29/92
5200                    ) END-EXEC.                                                         03/29/92
5300             EXEC SQL                                                                   03/29/92
5400              CREATE UNIQUE INDEX PART_ORDLI                                            03/29/92
5500                     ON DRDA/PART_ORDLN                                                 03/29/92
5600                     (ORDER_NUM ASC,                                                    03/29/92
5700                      ORDER_LINE ASC) END-EXEC.                                         03/29/92
```

図 22. コレクションおよびテーブルの作成 (1/2)

```
5738PW1 V2R1M1  920327                SEU SOURCE LISTING        03/29/92  17:16:50           PAGE    2
ソース・ファイル . . . .  DRDA/QLBLSRC
メンバー . . . . . .  CRTDB
SEQNBR*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 ...+... 9 ...+... 0
5800            EXEC SQL                                                                    03/29/92
5900              CREATE TABLE DRDA/SHIPMENTLN                                              03/29/92
6000                   (SHIP_NUM    SMALLINT     NOT NULL,
6100                    SHIP_LINE   SMALLINT     NOT NULL,
6200                    ORDER_LOC   CHAR(4)      NOT NULL,
6300                    ORDER_NUM   SMALLINT     NOT NULL,
6400                    ORDER_LINE  SMALLINT     NOT NULL,
6500                    PART_NUM    CHAR(5)      NOT NULL,
6600                    QUANT_SHIP  INTEGER NOT NULL,                                     03/29/92
6700                    QUANT_RECV  INTEGER NOT NULL WITH DEFAULT                         03/29/92
6800                   ) END-EXEC.                                                        03/29/92
6900               EXEC SQL                                                              03/29/92
7000                CREATE UNIQUE INDEX SHIPMENTLI                                       03/29/92
7100                     ON DRDA/SHIPMENTLN                                              03/29/92
7200                     (SHIP_NUM ASC,                                                  03/29/92
7300                      SHIP_LINE ASC) END-EXEC.                                       03/29/92
7400               EXEC SQL                                                              03/29/92
7500                     COMMIT         END-EXEC.                                       03/29/92
7600           STOP RUN.                                                                 03/29/92
* * * *  ソース仕様の終わり  * * * *
```

図 22. コレクションおよびテーブルの作成 (2/2)

例: テーブルへのデータの挿入

注: xv ページの『コードの特記事項情報』に重要なリーガル情報がありますので、お読みください。

```
5738PW1 V2R1M1  920327                SEU SOURCE LISTING          03/29/92 17:16:54           PAGE    1
ソース・ファイル . . . .  DRDA/QLBLSRC
メンバー . . . . . .  INSDB
SEQNBR*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 ...+... 9 ...+... 0
100        IDENTIFICATION DIVISION.                                                                  03/29/92
200        PROGRAM-ID. INSDB.                                                                        03/29/92
300        ENVIRONMENT DIVISION.                                                                     03/29/92
400        DATA DIVISION.                                                                            03/29/92
500        WORKING-STORAGE SECTION.                                                                  03/29/92
600              EXEC SQL INCLUDE SQLCA END-EXEC.                                                    03/29/92
700        PROCEDURE DIVISION.                                                                       03/29/92
800        MAIN.                                                                                     03/29/92
900                                                                                                  03/29/92
1000                                                                                                 03/29/92
1100       *-------------------------------------------------------------------                    03/29/92
1200       *  PART_STOCK TABLE                                                                       03/29/92
1300       *-------------------------------------------------------------*/--                      03/29/92
1400                                                                                                 03/29/92
1500                                                                                                 03/29/92
1600            EXEC SQL                                                                             03/29/92
1700             INSERT INTO PART_STOCK                                                              03/29/92
1800             VALUES                                                                              03/29/92
1900             ('14020','EA',038,050,100,' ') END-EXEC.                                            03/29/92
2000            EXEC SQL                                                                             03/29/92
2100             INSERT INTO PART_STOCK                                                              03/29/92
2200             VALUES                                                                              03/29/92
2300             ('14030','EA',043,050,050,' ') END-EXEC.                                            03/29/92
2400            EXEC SQL                                                                             03/29/92
2500             INSERT INTO PART_STOCK                                                              03/29/92
2600             VALUES                                                                              03/29/92
2700             ('14040','EA',030,020,030,' ') END-EXEC.                                            03/29/92
2800            EXEC SQL                                                                             03/29/92
2900             INSERT INTO PART_STOCK                                                              03/29/92
3000             VALUES                                                                              03/29/92
3100             ('14050','EA',010,005,015,' ') END-EXEC.                                            03/29/92
3200            EXEC SQL                                                                             03/29/92
3300              INSERT INTO PART_STOCK                                                             03/29/92
3400              VALUES                                                                             03/29/92
3500              ('14060','EA',110,045,090,' ') END-EXEC.                                           03/29/92
3600            EXEC SQL                                                                             03/29/92
3700              INSERT INTO PART_STOCK                                                             03/29/92
3800              VALUES                                                                          03/29/92
3900              ('14070','EA',130,080,160,' ') END-EXEC.                                           03/29/92
4000               EXEC SQL                                                                          03/29/92
4100              INSERT INTO PART_STOCK                                                             03/29/92
4200              VALUES                                                                             03/29/92
```

図 23. テーブルへのデータの挿入 (1/4)

```
5738PW1 V2R1M1  920327                SEU SOURCE LISTING        03/29/92  17:16:54           PAGE    2
ソース・ファイル . . . .  DRDA/QLBLSRC
メンバー . . . . . .  INSDB
SEQNBR*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 ...+... 9 ...+... 0
4300             ('18020','EA',013,025,050,' ') END-EXEC.                                    03/29/92
4400           EXEC SQL                                                                      03/29/92
4500             INSERT INTO PART_STOCK                                                      03/29/92
4600             VALUES                                                                      03/29/92
4700             ('18030','EA',015,005,010,' ') END-EXEC.                                    03/29/92
4800           EXEC SQL                                                                      03/29/92
4900             INSERT INTO PART_STOCK                                                      03/29/92
5000             VALUES                                                                      03/29/92
5100             ('21010','EA',029,030,050,' ') END-EXEC.                                    03/29/92
5200           EXEC SQL                                                                      03/29/92
5300             INSERT INTO PART_STOCK                                                      03/29/92
5400             VALUES                                                                      03/29/92
5500             ('24010','EA',025,020,040,' ') END-EXEC.                                    03/29/92
5600           EXEC SQL                                                                      03/29/92
5700             INSERT INTO PART_STOCK                                                      03/29/92
5800             VALUES                                                                      03/29/92
5900             ('24080','EA',054,050,050,' ') END-EXEC.                                    03/29/92
6000           EXEC SQL                                                                      03/29/92
6100             INSERT INTO PART_STOCK                                                      03/29/92
6200             VALUES                                                                      03/29/92
6300             ('24090','EA',030,025,050,' ') END-EXEC.                                    03/29/92
6400           EXEC SQL                                                                      03/29/92
6500             INSERT INTO PART_STOCK                                                      03/29/92
6600             VALUES                                                                      03/29/92
6700             ('24100','EA',020,015,030,' ') END-EXEC.                                    03/29/92
6800           EXEC SQL                                                                      03/29/92
6900             INSERT INTO PART_STOCK                                                      03/29/92
7000             VALUES                                                                      03/29/92
7100             ('24110','EA',052,050,080,' ') END-EXEC.                                    03/29/92
7200           EXEC SQL                                                                      03/29/92
7300             INSERT INTO PART_STOCK                                                      03/29/92
7400             VALUES                                                                      03/29/92
7500             ('25010','EA',511,300,600,' ') END-EXEC.                                    03/29/92
7600           EXEC SQL                                                                      03/29/92
7700             INSERT INTO PART_STOCK                                                      03/29/92
7800             VALUES                                                                      03/29/92
7900             ('36010','EA',013,005,010,' ') END-EXEC.                                    03/29/92
8000           EXEC SQL                                                                      03/29/92
8100             INSERT INTO PART_STOCK                                                      03/29/92
8200             VALUES                                                                      03/29/92
8300             ('36020','EA',110,030,060,' ') END-EXEC.                                    03/29/92
8400           EXEC SQL                                                                      03/29/92
8500             INSERT INTO PART_STOCK                                                      03/29/92
8600             VALUES                                                                      03/29/92
8700             ('37010','EA',415,100,200,' ') END-EXEC.                                    03/29/92
8800           EXEC SQL                                                                      03/29/92
8900             INSERT INTO PART_STOCK                                                      03/29/92
9000             VALUES                                                                      03/29/92
9100             ('37020','EA',010,020,040,' ') END-EXEC.                                    03/29/92
9200           EXEC SQL                                                                      03/29/92
9300             INSERT INTO PART_STOCK                                                      03/29/92
9400             VALUES                                                                      03/29/92
9500             ('37030','EA',154,055,060,' ') END-EXEC.                                    03/29/92
9600           EXEC SQL                                                                      03/29/92
9700             INSERT INTO PART_STOCK                                                      03/29/92
9800             VALUES                                                                      03/29/92
9900             ('37040','EA',223,120,120,' ') END-EXEC.                                    03/29/92
10000           EXEC SQL                                                                     03/29/92
10100             INSERT INTO PART_STOCK                                                     03/29/92
10200             VALUES                                                                     03/29/92
10300             ('43010','EA',110,020,040,' ') END-EXEC.                                   03/29/92
10400           EXEC SQL                                                                     03/29/92
10500             INSERT INTO PART_STOCK                                                     03/29/92
10600             VALUES                                                                     03/29/92
10700             ('43020','EA',067,050,050,' ') END-EXEC.                                   03/29/92
10800           EXEC SQL                                                                     03/29/92
```

図 23. テーブルへのデータの挿入 (2/4)

```
5738PW1 V2R1M1  920327                SEU SOURCE LISTING        03/29/92  17:16:54           PAGE    3
ソース・ファイル . . . .  DRDA/QLBLSRC
メンバー . . . . . .  INSDB
SEQNBR*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 ...+... 9 ...+... 0
10900              INSERT INTO PART_STOCK                                                    03/29/92
11000              VALUES                                                                    03/29/92
11100              ('48010','EA',032,030,060,' ') END-EXEC.                                  03/29/92
11200                                                                                        03/29/92
11300       *------------------------------------------------------------ --                 03/29/92
11400       *  PART_ORDER TABLE                                                              03/29/92
11500       *------------------------------------------------------------*/--                 03/29/92
11600                                                                                        03/29/92
11700                                                                                        03/29/92
11800                                                                                        03/29/92
11900            EXEC SQL                                                                    03/29/92
12000              INSERT INTO PART_ORDER                                                    03/29/92
12100              VALUES                                                                    03/29/92
12200           (1,'DB2B','U','O',0,' ','1991-03-12-17.00.00',NULL,NULL,NULL)                03/29/92
12300              END-EXEC.                                                                 03/29/92
12400            EXEC SQL                                                                    03/29/92
12500              INSERT INTO PART_ORDER                                                    03/29/92
12600              VALUES                                                                    03/29/92
12700               (2,'SQLA','U','O',0,' ','1991-03-12-17.01.00',                           03/29/92
12800               NULL,NULL,NULL)                                                          03/29/92
12900              END-EXEC.                                                                 03/29/92
13000            EXEC SQL                                                                    03/29/92
13100              INSERT INTO PART_ORDER                                                    03/29/92
13200              VALUES                                                                    03/29/92
13300               (3,'SQLA','U','O',0,' ','1991-03-12-17.02.00',                           03/29/92
13400               NULL,NULL,NULL)                                                          03/29/92
13500              END-EXEC.                                                                 03/29/92
13600            EXEC SQL                                                                    03/29/92
13700              INSERT INTO PART_ORDER                                                    03/29/92
13800              VALUES                                                                    03/29/92
13900               (4,'SQLA','U','O',0,' ','1991-03-12-17.03.00',                           03/29/92
14000               NULL,NULL,NULL)                                                          03/29/92
14100              END-EXEC.                                                                 03/29/92
14200            EXEC SQL                                                                    03/29/92
14300              INSERT INTO PART_ORDER                                                    03/29/92
14400              VALUES                                                                    03/29/92
14500               (5,'DB2B','U','O',0,' ','1991-03-12-17.04.00',                           03/29/92
14600               NULL,NULL,NULL)                                                          03/29/92
14700              END-EXEC.                                                                 03/29/92
14800                                                                                        03/29/92
14900       *------------------------------------------------------------------            03/29/92
15000       *  PART_ORDLN TABLE                                                            03/29/92
15100       *------------------------------------------------------------*/--                 03/29/92
15200                                                                                        03/29/92
15300                                                                                        03/29/92
15400            EXEC SQL                                                                    03/29/92
15500              INSERT INTO PART_ORDLN                                                    03/29/92
15600              VALUES                                                                    03/29/92
15700              (1,1,'24110',005,'O') END-EXEC.                                           03/29/92
```

図 23. テーブルへのデータの挿入 (3/4)

```
5738PW1 V2R1M1  920327                SEU SOURCE LISTING        03/29/92  17:16:54           PAGE    5
ソース・ファイル . . . .  DRDA/QLBLSRC
メンバー . . . . . .  INSDB
SEQNBR*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 ...+... 9 ...+... 0
15800           EXEC SQL                                                                    03/29/92
15900             INSERT INTO PART_ORDLN                                                    03/29/92
16000             VALUES                                                                    03/29/92
16100             (1,2,'24100',021,'O') END-EXEC.                                           03/29/92
16200           EXEC SQL                                                                    03/29/92
16300             INSERT INTO PART_ORDLN                                                    03/29/92
16400             VALUES                                                                    03/29/92
16500             (1,3,'24090',018,'O') END-EXEC.                                           03/29/92
16600           EXEC SQL                                                                    03/29/92
16700             INSERT INTO PART_ORDLN                                                    03/29/92
16800             VALUES                                                                    03/29/92
16900             (2,1,'14070',004,'O') END-EXEC.                                           03/29/92
17000           EXEC SQL                                                                    03/29/92
17100             INSERT INTO PART_ORDLN                                                    03/29/92
17200             VALUES                                                                    03/29/92
17300             (2,2,'37040',043,'O') END-EXEC.                                           03/29/92
17301           EXEC SQL                                                                    03/29/92
17302             INSERT INTO PART_ORDLN                                                    03/29/92
17303             VALUES                                                                    03/29/92
17304             (2,3,'14030',015,'O') END-EXEC.                                           03/29/92
17305           EXEC SQL                                                                    03/29/92
17306             INSERT INTO PART_ORDLN                                                    03/29/92
17307             VALUES                                                                    03/29/92
17308             (3,2,'14030',025,'O') END-EXEC.                                           03/29/92
17400           EXEC SQL                                                                    03/29/92
17500             INSERT INTO PART_ORDLN                                                    03/29/92
17600             VALUES                                                                    03/29/92
17700             (3,1,'43010',003,'O') END-EXEC.                                           03/29/92
17800           EXEC SQL                                                                    03/29/92
17900             INSERT INTO PART_ORDLN                                                    03/29/92
18000             VALUES                                                                    03/29/92
18100             (4,1,'36010',013,'O') END-EXEC.                                           03/29/92
18200           EXEC SQL                                                                    03/29/92
18300             INSERT INTO PART_ORDLN                                                    03/29/92
18400             VALUES                                                                    03/29/92
18500             (5,1,'18030',005,'O') END-EXEC.                                           03/29/92
18600                                                                                       03/29/92
18700                                                                                       03/29/92
18800       *-------------------------------------------------------------                03/29/92
18900       * SHIPMENTLN TABLE                                                            03/29/92
19000       *-------------------------------------------------------------*/--            03/29/92
19100                                                                                       03/29/92
19200                                                                                       03/29/92
19300           EXEC SQL                                                                    03/29/92
19400             INSERT INTO SHIPMENTLN                                                    03/29/92
19500             VALUES                                                                    03/29/92
19600             (1,1,'DB2B',1,1,'24110',5,5) END-EXEC.                                    03/29/92
19700           EXEC SQL                                                                    03/29/92
19800             INSERT INTO SHIPMENTLN                                                    03/29/92
19900             VALUES                                                                    03/29/92
20000             (1,2,'DB2B',1,2,'24100',10,1) END-EXEC.                                   03/29/92
20100           EXEC SQL                                                                    03/29/92
20200             INSERT INTO SHIPMENTLN                                                    03/29/92
20300             VALUES                                                                    03/29/92
20400             (2,1,'SQLA',2,1,'14070',4,4) END-EXEC.                                    03/29/92
20500           EXEC SQL                                                                    03/29/92
20600             INSERT INTO SHIPMENTLN                                                    03/29/92
20700             VALUES                                                                    03/29/92
20800             (2,2,'SQLA',2,2,'37040',45,25) END-EXEC.                                  03/29/92
20801           EXEC SQL                                                                    03/29/92
20802             INSERT INTO SHIPMENTLN                                                    03/29/92
20803             VALUES                                                                    03/29/92
20804             (2,3,'SQLA',2,3,'14030', 5,5) END-EXEC.                                   03/29/92
20805           EXEC SQL                                                                    03/29/92
20806             INSERT INTO SHIPMENTLN                                                    03/29/92
20807             VALUES                                                                    03/29/92
20808             (3,1,'SQLA',2,3,'14030', 5,5) END-EXEC.                                   03/29/92
20900                                                                                       03/29/92
21000           EXEC SQL COMMIT END-EXEC.                                                   03/29/92
21100           STOP RUN.                                                                   03/29/92
* * * *  ソース仕様の終わり  * * * *
```

図 23. テーブルへのデータの挿入 (4/4)

例: RPG プログラム

注: xv ページの『コードの特記事項情報』に重要なリーガル情報がありますので、お読みください。

```
5738PW1 V2R1M1  920327                SEU SOURCE LISTING          03/29/92 17:12:48           PAGE    1
ソース・ファイル . . . .  DRDA/QRPGSRC
メンバー . . . . . .  DDBPT6RG
SEQNBR*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 ...+... 9 ...+... 0
100       ****************************************************************                      03/29/92
200       *                                                              *                       03/29/92
300       *   DESCRIPTIVE NAME = D-DB SAMPLE APPLICATION                 *                      03/29/92
400       *                      REORDER POINT PROCESSING                *                      03/29/92
500       *                      AS/400                                  *                      03/29/92
600       *                                                              *                      03/29/92
700       *   FUNCTION =  THIS MODULE PROCESS THE PART_STOCK TABLE AND   *                      03/29/92
800       *               FOR EACH PART BELOW THE ROP (REORDER POINT)    *                      03/29/92
900       *               CREATES A SUPPLY ORDER AND PRINTS A REPORT.    *                      03/29/92
1000       *                                                             *                      03/29/92
1100       *                                                             *                      03/29/92
1200       *      INPUT = PARAMETERS EXPLICITLY PASSED TO THIS FUNCTION:  *                      03/29/92
1300       *                                                             *                      03/29/92
1400       *              LOCADB         LOCAL DB NAME                   *                      03/29/92
1500       *              REMODB         REMOTE DB NAME                  *                      03/29/92
1600       *                                                             *                      03/29/92
1700       *   TABLES =  PART-STOCK       - LOCAL                        *                      03/29/92
1800       *             PART_ORDER       - REMOTE                       *                      03/29/92
1900       *             PART_ORDLN       - REMOTE                       *                      03/29/92
2000       *             SHIPMENTLN       - REMOTE                       *                      03/29/92
2100       *                                                             *                      03/29/92
2200       *   INDICATORS =   *IN89       - '0' ORDER HEADER  NOT  DONE  *                      03/29/92
2300       *                                '1' ORDER HEADER  IS   DONE  *                      03/29/92
2400       *                  *IN99       - '1' ABNORMAL END (SQLCOD<0)  *                      03/29/92
2500       *                                                             *                      03/29/92
2600       *   TO BE COMPILED WITH COMMIT(*CHG) RDB(remotedbname)        *                      03/29/92
2700       *                                                             *                      03/29/92
2800       *   INVOKE BY : CALL DDBPT6RG PARM(localdbname remotedbname)  *                      03/29/92
2900       *                                                             *                      03/29/92
3000       *   CURSORS WILL BE CLOSED IMPLICITLY (BY CONNECT) BECAUSE    *                      03/29/92
3100       *   THERE IS NO REASON TO DO IT EXPLICITLY                    *                      03/29/92
3200       *                                                             *                      03/29/92
3300       ****************************************************************                      03/29/92
3400       *                                                                                     03/29/92
3500      FQPRINT  O   F      33     OF     PRINTER                                             03/29/92
3600      F*                                                                                    03/29/92
3700      I*                                                                                    03/29/92
3800      IMISC        DS                                                                       03/29/92
3900      I                                    B   1   20SHORTB                                 03/29/92
4000      I                                    B   3   60LONGB                                  03/29/92
4100      I                                    B   7   80INDNUL                                03/29/92
4200      I                                        9  13 PRTTBL                                 03/29/92
4300      I                                       14  29 LOCTBL                                 03/29/92
4400      I I            'SQLA'                   30  33 LOC                                    03/29/92
4500      I*                                                                                    03/29/92
4600      I*                                                                                    03/29/92
4700      C*                                                                                    03/29/92
4800      C           *LIKE     DEFN SHORTB    NXTORD          NEW ORDER NR                    03/29/92
4900      C           *LIKE     DEFN SHORTB    NXTORL          ORDER LINE NR                   03/29/92
5000      C           *LIKE     DEFN SHORTB    RTCOD1          RTCOD NEXT_PART                 03/29/92
5100      C           *LIKE     DEFN SHORTB    RTCOD2          RTCOD NEXT_ORD_                 03/29/92
5200      C           *LIKE     DEFN SHORTB    CURORD          ORDER NUMBER                    03/29/92
5300      C           *LIKE     DEFN SHORTB    CURORL          ORDER LINE                      03/29/92
5400      C           *LIKE     DEFN LONGB     QUANTI          FOR COUNTING                    03/29/92
5500      C           *LIKE     DEFN LONGB     QTYSTC          QTY ON STOCK                    03/29/92
5600      C           *LIKE     DEFN LONGB     QTYORD          REORDER QTY                     03/29/92
```

図 24. RPG プログラム例 (1/8)

```
5738PW1 V2R1M1  920327                SEU SOURCE LISTING        03/29/92  17:12:48           PAGE    2
ソース・ファイル . . . .  DRDA/QRPGSRC
メンバー . . . . . .  DDBPT6RG
SEQNBR*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 ...+... 9 ...+... 0
5700      C           *LIKE     DEFN LONGB     QTYROP          REORDER POINT                     03/29/92
5800      C           *LIKE     DEFN LONGB     QTYREQ          QTY ORDERED                       03/29/92
5900      C           *LIKE     DEFN LONGB     QTYREC          QTY RECEIVED                      03/29/92
6000      C*                                                                                     03/29/92
6100      C*                                                                                     03/29/92
6200      C*************************************************************                        03/29/92
6300      C*     PARAMETERS                                             *                        03/29/92
6400      C*************************************************************                        03/29/92
6500      C*                                                                                     03/29/92
6600      C           *ENTRY    PLIST                                                            03/29/92
6700      C                     PARM           LOCADB 18       LOCAL DATABASE                    03/29/92
6800      C                     PARM           REMODB 18       REMOTE DATABASE                   03/29/92
6900      C*                                                                                     03/29/92
7000      C*                                                                                     03/29/92
7100      C*************************************************************                        03/29/92
7200      C*    SQL CURSOR DECLARATIONS                                 *                        03/29/92
7300      C*************************************************************                        03/29/92
7400      C*                                                                                     03/29/92
7500      C* NEXT PART WHICH STOCK QUANTITY IS UNDER REORDER POINTS QTY                          03/29/92
7600      C/EXEC SQL                                                                             03/29/92
7700      C+     DECLARE NEXT_PART CURSOR FOR                                                    03/29/92
7800      C+         SELECT PART_NUM,                                                            03/29/92
7900      C+                PART_QUANT,                                                          03/29/92
8000      C+                PART_ROP,                                                            03/29/92
8100      C+                PART_EOQ                                                             03/29/92
8200      C+         FROM   PART_STOCK                                                           03/29/92
8300      C+         WHERE  PART_ROP > PART_QUANT                                                03/29/92
8400      C+           AND  PART_NUM > :PRTTBL                                                   03/29/92
8500      C+         ORDER BY PART_NUM ASC                                                       03/29/92
8600      C/END-EXEC                                                                             03/29/92
8700      C*                                                                                     03/29/92
8800      C* ORDERS WHICH ARE ALREADY MADE FOR CURRENT PART                                    03/29/92
8900      C/EXEC SQL                                                                             03/29/92
9000      C+     DECLARE NEXT_ORDER_LINE CURSOR FOR                                              03/29/92
9100      C+         SELECT A.ORDER_NUM,                                                         03/29/92
9200      C+                ORDER_LINE,                                                          03/29/92
9300      C+                QUANT_REQ                                                            03/29/92
9400      C+         FROM   PART_ORDLN A,                                                        03/29/92
9500      C+                PART_ORDER B                                                         03/29/92
9600      C+         WHERE  PART_NUM  = :PRTTBL                                                  03/29/92
9700      C+         AND    LINE_STAT  <> 'C'                                                    03/29/92
9800      C+         AND    A.ORDER_NUM = B.ORDER_NUM                                            03/29/92
9900      C+         AND    ORDER_TYPE  = 'R'                                                    03/29/92
10000      C/END-EXEC                                                                            03/29/92
10100      C*                                                                                    03/29/92
10200      C*************************************************************                       03/29/92
10300      C*     SQL RETURN CODE HANDLING                               *                       03/29/92
10400      C*************************************************************                       03/29/92
10500      C/EXEC SQL                                                                            03/29/92
10600      C+     WHENEVER SQLERROR GO TO DBERRO                                                 03/29/92
10700      C/END-EXEC                                                                            03/29/92
10800      C/EXEC SQL                                                                            03/29/92
10900      C+     WHENEVER SQLWARNING CONTINUE                                                   03/29/92
11000      C/END-EXEC                                                                            03/29/92
```

図 24. RPG プログラム例 (2/8)

```
5738PW1 V2R1M1  920327                SEU SOURCE LISTING        03/29/92  17:12:48           PAGE    3
ソース・ファイル . . . .  DRDA/QRPGSRC
メンバー . . . . . .  DDBPT6RG
SEQNBR*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 ...+... 9 ...+... 0
11100      C*                                                                                          03/29/92
11200      C*                                                                                          03/29/92
11300      C*************************************************************                              03/29/92
11400      C*    PROCESS  - MAIN PROGRAM LOGIC                          *                              03/29/92
11500      C*    MAIN PROCEDURE WORKS   WITH LOCAL DATABASE             *                              03/29/92
11600      C*************************************************************                              03/29/92
11700      C*                                                                                          03/29/92
11800      C*CLEAN UP TO PERMIT RE-RUNNING OF TEST DATA                                               03/29/92
11900      C                     EXSR CLEANU                                                           03/29/92
12000      C*                                                                                          03/29/92
12100      C*                                                                                          03/29/92
12200      C           RTCOD1    DOUEQ100                                                            03/29/92
12300      C*                                                                                          03/29/92
12400      C/EXEC SQL                                                                                03/29/92
12500      C+          CONNECT   TO  :LOCADB                                                         03/29/92
12600      C/END-EXEC                                                                                03/29/92
12700      C/EXEC SQL                                                                                03/29/92
12800      C+          OPEN      NEXT_PART                                                           03/29/92
12900      C/END-EXEC                                                                                03/29/92
13000      C/EXEC SQL                                                                                03/29/92
13100      C+          FETCH     NEXT_PART                                                           03/29/92
13200      C+           INTO     :PRTTBL,                                                            03/29/92
13300      C+                    :QTYSTC,                                                            03/29/92
13400      C+                    :QTYROP,                                                            03/29/92
13500      C+                    :QTYORD                                                             03/29/92
13600      C/END-EXEC                                                                                03/29/92
13700      C                     MOVE SQLCOD    RTCOD1                                               03/29/92
13800      C/EXEC SQL                                                                                03/29/92
13900      C+          COMMIT                                                                        03/29/92
14000      C/END-EXEC                                                                                03/29/92
14100      C           RTCOD1    IFNE 100                                                            03/29/92
14200      C                     EXSR CHECKO                                                         03/29/92
14300      C                     ENDIF                                                               03/29/92
14400      C*                                                                                          03/29/92
14500      C                     ENDDO                                                               03/29/92
14600      C*                                                                                          03/29/92
14700      C                     GOTO SETLR                                                          03/29/92
14800      C*                                                                                          03/29/92
14900      C*                                                                                          03/29/92
15000      C*****************************************************************                        03/29/92
15100      C*    SQL RETURN CODE HANDLING ON ERROR SITUATIONS               *                        03/29/92
15200      C*****************************************************************                        03/29/92
15300      C*                                                                                          03/29/92
15400      C           DBERRO    TAG                                                                 03/29/92
15500      C*         *--------------*                                                               03/29/92
15600      C                     EXCPTERRLIN                                                         03/29/92
15700      C                     MOVE *ON       *IN99                                                03/29/92
15800      C/EXEC SQL                                                                                03/29/92
15900      C+          WHENEVER  SQLERROR  CONTINUE                                                  03/29/92
16000      C/END-EXEC                                                                                03/29/92
```

図 24. RPG プログラム例 (3/8)

```
5738PW1 V2R1M1  920327                SEU SOURCE LISTING        03/29/92  17:12:48           PAGE    4
ソース・ファイル . . . .  DRDA/QRPGSRC
メンバー . . . . . .  DDBPT6RG
SEQNBR*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 ...+... 9 ...+... 0
16100      C/EXEC SQL                                                                                  03/29/92
16200      C+          ROLLBACK                                                                        03/29/92
16300      C/END-EXEC                                                                                  03/29/92
16400      C/EXEC SQL                                                                                  03/29/92
16500      C+          WHENEVER  SQLERROR  GO TO  DBERRO                                               03/29/92
16600      C/END-EXEC                                                                                  03/29/92
16700      C*                                                                                          03/29/92
16800      C*                                                                                          03/29/92
16900      C           SETLR     TAG                                                                   03/29/92
17000      C*         *-------------*                                                                  03/29/92
17100      C/EXEC SQL                                                                                  03/29/92
17200      C+          CONNECT   RESET                                                                 03/29/92
17300      C/END-EXEC                                                                                  03/29/92
17400      C                     MOVE *ON       *INLR                                                  03/29/92
17500      C*                                                                                          03/29/92
17600      C*************************************************************                             03/29/92
17700      C*    THE END OF THE PROGRAM                                  *                             03/29/92
17800      C*************************************************************                             03/29/92
17900      C*                                                                                          03/29/92
18000      C*                                                                                          03/29/92
18100      C*************************************************************                             03/29/92
18200      C* SUBROUTINES TO WORK WITH REMOTE DATABASES                  *                             03/29/92
18300      C*************************************************************                             03/29/92
18400      C*                                                                                          03/29/92
18500      C*                                                                                          03/29/92
18600      C           CHECKO    BEGSR                                                                 03/29/92
18700      C*         *---------------*                                                                03/29/92
18800      C*************************************************************                             03/29/92
18900      C* CHECKS WHAT IS CURRENT ORDER AND SHIPMENT STATUS FOR THE PART *                           03/29/92
19000      C* IF ORDERED AND SHIPPED IS LESS THAN REORDER POINT OF PART,    *                           03/29/92
19100      C* PERFORMS A SUBROUTINE WHICH MAKES AN ORDER.                   *                           03/29/92
19200      C*************************************************************                             03/29/92
19300      C*                                                                                          03/29/92
19400      C                     MOVE 0         RTCOD2                                                 03/29/92
19500      C                     MOVE 0         QTYREQ                                                 03/29/92
19600      C                     MOVE 0         QTYREC                                                 03/29/92
19700      C*                                                                                          03/29/92
19800      C/EXEC SQL                                                                                  03/29/92
19900      C+          CONNECT   TO   :REMODB                                                          03/29/92
20000      C/END-EXEC                                                                                  03/29/92
20100      C/EXEC SQL                                                                                  03/29/92
20200      C+          OPEN      NEXT_ORDER_LINE                                                       03/29/92
20300      C/END-EXEC                                                                                  03/29/92
20400      C*                                                                                          03/29/92
20500      C           RTCOD2    DOWNE100                                                              03/29/92
20600      C*                                                                                          03/29/92
20700      C/EXEC SQL                                                                                  03/29/92
20800      C+          FETCH     NEXT_ORDER_LINE                                                       03/29/92
20900      C+           INTO     :CURORD,                                                              03/29/92
```

図 24. RPG プログラム例 (4/8)

```
5738PW1 V2R1M1  920327                SEU SOURCE LISTING        03/29/92  17:12:48           PAGE    5
ソース・ファイル . . . .  DRDA/QRPGSRC
メンバー . . . . . .  DDBPT6RG
SEQNBR*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 ...+... 9 ...+... 0
21000      C+                    :CURORL,                                                          03/29/92
21100      C+                    :QUANTI                                                           03/29/92
21200      C/END-EXEC                                                                              03/29/92
21300      C*                                                                                      03/29/92
21400      C           SQLCOD    IFEQ 100                                                          03/29/92
21500      C                     MOVE 100       RTCOD2                                             03/29/92
21600      C                     ELSE                                                              03/29/92
21700      C                     ADD  QUANTI    QTYREQ                                             03/29/92
21800      C*                                                                                      03/29/92
21900      C/EXEC SQL                                                                              03/29/92
22000      C+          SELECT    SUM(QUANT_RECV)                                                   03/29/92
22100      C+            INTO    :QUANTI:INDNUL
22200      C+            FROM    SHIPMENTLN                                                        03/29/92
22300      C+            WHERE   ORDER_LOC  = :LOC                                                 03/29/92
22400      C+              AND   ORDER_NUM  = :CURORD                                              03/29/92
22500      C+              AND   ORDER_LINE = :CURORL                                              03/29/92
22600      C/END-EXEC                                                                              03/29/92
22700      C*                                                                                      03/29/92
22800      C           INDNUL    IFGE 0                                                            03/29/92
22900      C                     ADD  QUANTI    QTYREC                                             03/29/92
23000      C                     ENDIF                                                             03/29/92
23100      C*                                                                                      03/29/92
23200      C                     ENDIF                                                             03/29/92
23300      C                     ENDDO                                                             03/29/92
23400      C*                                                                                      03/29/92
23500      C/EXEC SQL                                                                              03/29/92
23600      C+     CLOSE NEXT_ORDER_LINE                                                            03/29/92
23700      C/END-EXEC                                                                              03/29/92
23800      C*                                                                                      03/29/92
23900      C           QTYSTC    ADD  QTYREQ    QUANTI                                             03/29/92
24000      C                     SUB  QUANTI    QTYREC                                             03/29/92
24100      C*                                                                                      03/29/92
24200      C           QTYROP    IFGT QUANTI                                                       03/29/92
24300      C                     EXSR ORDERP                                                       03/29/92
24400      C                     ENDIF                                                             03/29/92
24500      C*                                                                                      03/29/92
24600      C/EXEC SQL                                                                              03/29/92
24700      C+          COMMIT                                                                    03/29/92
24800      C/END-EXEC                                                                              03/29/92
24900      C*                                                                                      03/29/92
25000      C                     ENDSR                          CHECKO                             03/29/92
25100      C*                                                                                      03/29/92
25200      C*                                                                                      03/29/92
25300      C           ORDERP    BEGSR                                                             03/29/92
25400      C*         *---------------*                                                            03/29/92
```

図 24. RPG プログラム例 (5/8)

```
5738PW1 V2R1M1  920327                SEU SOURCE LISTING         03/29/92  17:12:48           PAGE    7
ソース・ファイル . . . .  DRDA/QRPGSRC
メンバー . . . . . .  DDBPT6RG
SEQNBR*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 ...+... 9 ...+... 0
25500      C*************************************************************                      03/29/92
25600      C* MAKES AN ORDER. IF FIRST TIME, PERFORMS THE SUBROUTINE, WHICH *                    03/29/92
25700      C* SEARCHES FOR NEW ORDER NUMBER AND MAKES THE ORDER HEADER.     *                    03/29/92
25800      C* AFTER THAT MAKES ORDER LINES USING REORDER QUANTITY FOR THE   *                    03/29/92
25900      C* PART. FOR EVERY ORDERED PART WRITES A LINE ON REPORT.         *                    03/29/92
26000      C*************************************************************                      03/29/92
26100      C*                                                                                   03/29/92
26200      C           *IN89     IFEQ *OFF                       FIRST ORDER ?                03/29/92
26300      C                     EXSR STRORD                                                    03/29/92
26400      C                     MOVE *ON       *IN89            ORD.HEAD.DONE                03/29/92
26500      C                     EXCPTHEADER                     WRITE HEADERS                03/29/92
26600      C                     ENDIF                                                          03/29/92
26700      C*                                                                                   03/29/92
26800      C                     ADD  1         NXTORL           NEXT ORD.LIN                 03/29/92
26900      C/EXEC SQL                                                                           03/29/92
27000      C+          INSERT                                                                   03/29/92
27100      C+            INTO    PART_ORDLN                                                     03/29/92
27200      C+                   (ORDER_NUM,                                                     03/29/92
27300      C+                    ORDER_LINE,                                                    03/29/92
27400      C+                    PART_NUM,                                                      03/29/92
27500      C+                    QUANT_REQ,                                                     03/29/92
27600      C+                    LINE_STAT)                                                     03/29/92
27700      C+          VALUES   (:NXTORD,                                                       03/29/92
27800      C+                    :NXTORL,                                                       03/29/92
27900      C+                    :PRTTBL,                                                       03/29/92
28000      C+                    :QTYORD,                                                       03/29/92
28100      C+                    'O')                                                           03/29/92
28200      C/END-EXEC                                                                           03/29/92
28300      C*                                                                                   03/29/92
28400      C           *INOF     IFEQ *ON                                                       03/29/92
28500      C                     EXCPTHEADER                                                    03/29/92
28600      C                     END                                                            03/29/92
28700      C                     EXCPTDETAIL                                                    03/29/92
28800      C*                                                                                   03/29/92
28900      C                     ENDSR                           ORDERP                       03/29/92
29000      C*                                                                                   03/29/92
29100      C*                                                                                   03/29/92
29200      C           STRORD    BEGSR                                                        03/29/92
29300      C*         *---------------*                                                        03/29/92
29400      C*************************************************************                      03/29/92
29500      C* SEARCHES FOR NEXT ORDER NUMBER AND MAKES AN ORDER HEADER      *                    03/29/92
29600      C* USING THAT NUMBER. WRITES ALSO HEADERS ON REPORT.             *                    03/29/92
29700      C*************************************************************                      03/29/92
29800      C*                                                                                   03/29/92
29900      C/EXEC SQL                                                                           03/29/92
30000      C+          SELECT    (MAX(ORDER_NUM) + 1)                                           03/29/92
30100      C+            INTO    :NXTORD                                                        03/29/92
30200      C+            FROM    PART_ORDER                                                     03/29/92
30300      C/END-EXEC                                                                           03/29/92
```

図 24. RPG プログラム例 (6/8)

```
5738PW1 V2R1M1  920327                SEU SOURCE LISTING           03/29/92  17:12:48         PAGE    8
ソース・ファイル . . . .  DRDA/QRPGSRC
メンバー . . . . . .  DDBPT6RG
SEQNBR*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 ...+... 9 ...+... 0
30400      C/EXEC SQL                                                                                    03/29/92
30500      C+          INSERT                                                                            03/29/92
30600      C+            INTO    PART_ORDER                                                              03/29/92
30700      C+                   (ORDER_NUM,                                                              03/29/92
30800      C+                    ORIGIN_LOC,                                                             03/29/92
30900      C+                    ORDER_TYPE,                                                             03/29/92
31000      C+                    ORDER_STAT,                                                             03/29/92
31100      C+                    CREAT_TIME)                                                             03/29/92
31200      C+          VALUES   (:NXTORD,                                                                03/29/92
31300      C+                    :LOC,                                                                   03/29/92
31400      C+                    'R',                                                                    03/29/92
31500      C+                    'O',                                                                    03/29/92
31600      C+                    CURRENT TIMESTAMP)                                                      03/29/92
31700      C/END-EXEC                                                                                    03/29/92
31800      C                     ENDSR                           STRORD                                  03/29/92
31900      C*                                                                                            03/29/92
32000      C*                                                                                            03/29/92
32100      C           CLEANU    BEGSR                                                                   03/29/92
32200      C*         *---------------*                                                                  03/29/92
32300      C*************************************************************                               03/29/92
32400      C* THIS SUBROUTINE IS ONLY REQUIRED IN A TEST ENVIRONMENT                                     03/29/92
32500      C* TO RESET THE DATA TO PERMIT RE-RUNNING OF THE TEST                                         03/29/92
32600      C*************************************************************                               03/29/92
32700      C*                                                                                            03/29/92
32800      C/EXEC SQL                                                                                    03/29/92
32900      C+          CONNECT   TO   :REMODB                                                            03/29/92
33000      C/END-EXEC                                                                                    03/29/92
33100      C/EXEC SQL                                                                                    03/29/92
33200      C+          DELETE                                                                            03/29/92
33300      C+            FROM    PART_ORDLN                                                              03/29/92
33400      C+           WHERE    ORDER_NUM IN                                                            03/29/92
33500      C+                   (SELECT    ORDER_NUM                                                    03/29/92
33600      C+                      FROM    PART_ORDER                                                   03/29/92
33700      C+                      WHERE   ORDER_TYPE = 'R')                                            03/29/92
33800      C/END-EXEC                                                                                    03/29/92
33900      C/EXEC SQL                                                                                    03/29/92
34000      C+          DELETE                                                                            03/29/92
34100      C+            FROM    PART_ORDER                                                              03/29/92
34200      C+           WHERE    ORDER_TYPE = 'R'                                                        03/29/92
34300      C/END-EXEC                                                                                    03/29/92
34400      C/EXEC SQL                                                                                    03/29/92
34500      C+          COMMIT                                                                            03/29/92
34600      C/END-EXEC                                                                                    03/29/92
34700      C*                                                                                            03/29/92
34800      C                     ENDSR                           CLEANU                                  03/29/92
34900      C*                                                                                            03/29/92
35000      C*                                                                                            03/29/92
```

図 24. RPG プログラム例 (7/8)

```
5738PW1 V2R1M1  920327                SEU SOURCE LISTING           03/29/92  17:12:48        PAGE    9
ソース・ファイル . . . .  DRDA/QRPGSRC
メンバー . . . . . .  DDBPT6RG
SEQNBR*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 ...+... 9 ...+... 0
35100      C*************************************************************                     03/29/92
35200      O* OUTPUTLINES FOR THE REPORT                                *                     03/29/92
35300      O*************************************************************                     03/29/92
35400      O*                                                                                 03/29/92
35500      OQPRINT  E    2           HEADER                                                  03/29/92
35600      O                                +  0 '***** ROP PROCESSING'                       03/29/92
35700      O                                +  1 'REPORT *****'                               03/29/92
35800      O*                                                                                 03/29/92
35900      OQPRINT  E 2              HEADER                                                  03/29/92
36000      O                                +  0 '   ORDER NUMBER = '                         03/29/92
36100      O                         NXTORDZ +  0                                            03/29/92
36200      O*                                                                                 03/29/92
36300      OQPRINT  E 1              HEADER                                                  03/29/92
36400      O                                +  0 '------------------------'                  03/29/92
36500      O                                +  0 '---------'                                 03/29/92
36600      O*                                                                                 03/29/92
36700      OQPRINT  E 1              HEADER                                                  03/29/92
36800      O                                +  0 '   LINE     '                              03/29/92
36900      O                                +  0 'PART          '                            03/29/92
37000      O                                +  0 'QTY    '                                   03/29/92
37100      O*                                                                                 03/29/92
37200      OQPRINT  E 1              HEADER                                                  03/29/92
37300      O                                +  0 '  NUMBER   '                              03/29/92
37400      O                                +  0 'NUMBER      '                             03/29/92
37500      O                                +  0 'REQUESTED '                               03/29/92
37600      O*                                                                                 03/29/92
37700      OQPRINT  E 11             HEADER                                                  03/29/92
37800      O                                +  0 '------------------------'                  03/29/92
37900      O                                +  0 '---------'                                 03/29/92
38000      O*                                                                                 03/29/92
38100      OQPRINT  EF1              DETAIL                                                  03/29/92
38200      O                         NXTORLZ +  4                                            03/29/92
38300      O                         PRTTBL  +  4                                            03/29/92
38400      O                         QTYORD1 +  4                                            03/29/92
38500      O*                                                                                 03/29/92
38600      OQPRINT  T 2      LRN99                                                           03/29/92
38700      O                                +  0 '------------------------'                  03/29/92
38800      O                                +  0 '---------'                                 03/29/92
38900      OQPRINT  T 1      LRN99                                                           03/29/92
39000      O                                +  0 'NUMBER OF LINES '                          03/29/92
39100      O                                +  0 'CREATED = '                                03/29/92
39200      O                         NXTORLZ +  0                                            03/29/92
39300      O*                                                                                 03/29/92
39400      OQPRINT  T 1      LRN99                                                           03/29/92
39500      O                                +  0 '------------------------'                  03/29/92
39600      O                                +  0 '---------'                                 03/29/92
39700      O*                                                                                 03/29/92
39800      OQPRINT  T 2      LRN99                                                           03/29/92
39900      O                                +  0 '*********'                                 03/29/92
40000      O                                +  0 ' END OF PROGRAM '                          03/29/92
40100      O                                +  0 '********'                                  03/29/92
40200      O*                                                                                 03/29/92
40300      OQPRINT  E 2              ERRLIN                                                  03/29/92
40400      O                                +  0 '** ERROR **'                               03/29/92
40500      O                                +  0 '** ERROR **'                               03/29/92
40600      O                                +  0 '** ERROR **'                               03/29/92
40700      OQPRINT  E 1              ERRLIN                                                  03/29/92
40800      O                                +  0 '* SQLCOD:'                                 03/29/92
40900      O                         SQLCODM +  0                                            03/29/92
41000      O                                  33 '*'                                        03/29/92
41100      OQPRINT  E 1              ERRLIN                                                  03/29/92
41200      O                                +  0 '* SQLSTATE:'                               03/29/92
41300      O                         SQLSTT  +  2                                            03/29/92
41400      O                                  33 '*'                                        03/29/92
41500      OQPRINT  E 1              ERRLIN                                                  03/29/92
41600      O                                +  0 '** ERROR **'                               03/29/92
41700      O                                +  0 '** ERROR **'                               03/29/92
41800      O                                +  0 '** ERROR **'                               03/29/92
```

図 24. RPG プログラム例 (8/8)

例: COBOL プログラム

注: xv ページの『コードの特記事項情報』に重要なリーガル情報がありますので、お読みください。

```
5738PW1 V2R1M1  920327                 SEU SOURCE LISTING           03/29/92  17:12:35         PAGE    1
ソース・ファイル . . . .  DRDA/QLBLSRC
メンバー . . . . . .  DDBPT6CB
SEQNBR*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 ...+... 9 ...+... 0
100        IDENTIFICATION DIVISION.
200       *-----------------------
300        PROGRAM-ID. DDBPT6CB.                                                              03/29/92
400       ****************************************************************                    03/29/92
500       *   MODULE NAME = DDBPT6CB                                                         03/29/92
600       *
700       *   DESCRIPTIVE NAME = D-DB SAMPLE APPLICATION
800       *                      REORDER POINT PROCESSING
900       *                      AS/400                                                      03/29/92
1000       *                      COBOL
1100       *
1200       *   FUNCTION =  THIS MODULE PROCESS THE PART_STOCK TABLE AND
1300       *               FOR EACH PART BELOW THE ROP (REORDER POINT)
1400       *               CHECKS THE EXISTING ORDERS AND SHIPMENTS,                           03/29/92
1500       *               CREATES A SUPPLY ORDER AND PRINTS A REPORT.                         03/29/92
1600       *
1700       *      DEPENDENCIES = NONE                                                         03/29/92
1800       *
1900       *      INPUT = PARAMETERS EXPLICITLY PASSED TO THIS FUNCTION:
2000       *
2100       *              LOCAL-DB       LOCAL DB NAME                                       03/29/92
2200       *              REMOTE-DB      REMOTE DB NAME                                      03/29/92
2300       *
2400       *   TABLES =  PART-STOCK       - LOCAL                                           03/29/92
2500       *             PART_ORDER       - REMOTE                                          03/29/92
2600       *             PART_ORDLN       - REMOTE                                          03/29/92
2700       *             SHIPMENTLN       - REMOTE                                          03/29/92
2800       *                                                                                03/29/92
2900       *   CRTSQLCBL  SPECIAL PARAMETERS                                                03/29/92
3000       *    PGM(DDBPT6CB) RDB(remotedbname) OPTION(*APOST *APOSTSQL)                    03/29/92
3100       *                                                                                03/29/92
3200       *   INVOKE BY : CALL DDBPT6CB PARM(localdbname remotedbname)                     03/29/92
3300       *                                                                                03/29/92
3400       ****************************************************************                03/29/92
3500        ENVIRONMENT DIVISION.
3600       *--------------------
3700        INPUT-OUTPUT SECTION.
3800        FILE-CONTROL.
3900            SELECT RELAT  ASSIGN TO PRINTER-QPRINT.                                     03/29/92
4000        DATA DIVISION.
4100       *-------------
4200        FILE SECTION.
4300       *-------------                                                                   03/29/92
4400        FD  RELAT
4500            RECORD CONTAINS 33 CHARACTERS
4600            LABEL RECORDS ARE OMITTED
4700            DATA RECORD IS REPREC.
4800        01  REPREC          PIC  X(33).
4900        WORKING-STORAGE SECTION.
5000       *-----------------------                                                         03/29/92
5100       *    PRINT LINE  DEFINITIONS                                                    03/29/92
5200        01  LINE0           PIC  X(33) VALUE  SPACES.
5300        01  LINE1           PIC  X(33) VALUE
```

図 25. COBOL プログラム例 (1/8)

```
5738PW1 V2R1M1  920327                SEU SOURCE LISTING           03/29/92  17:12:35        PAGE    2
ソース・ファイル . . . .  DRDA/QLBLSRC
メンバー . . . . . .  DDBPT6CB
SEQNBR*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 ...+... 9 ...+... 0
5400           '***** ROP PROCESSING REPORT *****'.
5500        01  LINE2.
5600          05  FILLER        PIC  X(18) VALUE  '   ORDER NUMBER = '.
5700          05  MASK0         PIC  ZZZ9.
5800          05  FILLER        PIC  X(11) VALUE  SPACES.
5900        01  LINE3           PIC  X(33) VALUE
6000           '--------------------------------'.
6100        01  LINE4           PIC  X(33) VALUE
6200           '   LINE     PART         QTY     '.
6300        01  LINE5           PIC  X(33) VALUE
6400           '  NUMBER   NUMBER     REQUESTED  '.
6500        01  LINE6.
6600          05  FILLER        PIC  XXXX  VALUE SPACES.
6700          05  MASK1         PIC  ZZZ9.
6800          05  FILLER        PIC  XXXX  VALUE SPACES.
6900          05  PART-TABLE    PIC  XXXXX.
7000          05  FILLER        PIC  XXXX  VALUE SPACES.
7100          05  MASK2         PIC  Z,ZZZ,ZZZ.ZZ.
7200        01  LINE7.
7300          05  FILLER        PIC  X(26) VALUE
7400            'NUMBER OF LINES CREATED = '.
7500          05  MASK3         PIC  ZZZ9.
7600          05  FILLER        PIC  XXX   VALUE  SPACES.
7700        01  LINE8           PIC  X(33) VALUE
7800           '********* END OF PROGRAM ********'.
7900       *    MISCELLANEOUS DEFINITIONS                                                  03/29/92
8000        01  WHAT-TIME       PIC  X     VALUE '1'.
8100           88  FIRST-TIME             VALUE '1'.
8200        01  CONTL           PIC  S9999 COMP-4 VALUE ZEROS.                          03/29/92
8300        01  CONTD           PIC  S9999 COMP-4 VALUE ZEROS.                          03/29/92
8400        01  RTCODE1         PIC  S9999 COMP-4 VALUE ZEROS.                          03/29/92
8500        01  RTCODE2         PIC  S9999 COMP-4.                                      03/29/92
8600        01  NEXT-NUM        PIC  S9999 COMP-4.                                      03/29/92
8700        01  IND-NULL        PIC  S9999 COMP-4.                                      03/29/92
8800        01  LOC-TABLE       PIC  X(16).
8900        01  ORD-TABLE       PIC  S9999 COMP-4.                                      03/29/92
9000        01  ORL-TABLE       PIC  S9999 COMP-4.                                      03/29/92
9100        01  QUANT-TABLE     PIC  S9(9) COMP-4.                                      03/29/92
9200        01  QTY-TABLE       PIC  S9(9) COMP-4.                                      03/29/92
9300        01  ROP-TABLE       PIC  S9(9) COMP-4.                                      03/29/92
9400        01  EOQ-TABLE       PIC  S9(9) COMP-4.                                      03/29/92
9500        01  QTY-REQ         PIC  S9(9) COMP-4.                                      03/29/92
9600        01  QTY-REC         PIC  S9(9) COMP-4.                                      03/29/92
9700       * CONSTANT  FOR LOCATION NUMBER                                              03/29/92
9800        01  XPARM.                                                                  03/29/92
9900            05   LOC        PIC  X(4)  VALUE 'SQLA'.                               03/29/92
10000       * DEFINITIONS FOR ERROR MESSAGE HANDLING                                    03/29/92
10100        01  ERROR-MESSAGE.                                                         03/29/92
10200            05   MSG-ID.                                                          03/29/92
10300            10   MSG-ID-1     PIC  X(2)                                          03/29/92
10400                 VALUE 'SQ'.                                                     03/29/92
10500            10   MSG-ID-2     PIC 99999.                                         03/29/92
```

図 25. COBOL プログラム例 (2/8)

```
5738PW1 V2R1M1  920327                SEU SOURCE LISTING           03/29/92  17:12:35         PAGE    3
ソース・ファイル . . . .  DRDA/QLBLSRC
メンバー . . . . . .  DDBPT6CB
SEQNBR*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 ...+... 9 ...+... 0
10600       ******************************                                                  03/29/92
10700       *    SQLCA INCLUDE           *                                                  03/29/92
10800       ******************************                                                  03/29/92
10900            EXEC SQL INCLUDE SQLCA    END-EXEC.
11000                                                                                       03/29/92
11100        LINKAGE SECTION.                                                               03/29/92
11200       *----------------                                                               03/29/92
11300        01  LOCAL-DB        PIC  X(18).                                               03/29/92
11400        01  REMOTE-DB       PIC  X(18).                                               03/29/92
11500                                                                                       03/29/92
11600        PROCEDURE DIVISION USING LOCAL-DB REMOTE-DB.                                   03/29/92
11700       *------------------                                                             03/29/92
11800       ******************************                                                  03/29/92
11900       *    SQL CURSOR DECLARATION *                                                   03/29/92
12000       ******************************                                                  03/29/92
12100       * RE-POSITIONABLE CURSOR : POSITION AFTER LAST PART_NUM                         03/29/92
12200            EXEC SQL DECLARE NEXT_PART CURSOR FOR
12300                 SELECT PART_NUM,
12400                        PART_QUANT,
12500                        PART_ROP,
12600                        PART_EOQ
12700                 FROM   PART_STOCK
12800                 WHERE  PART_ROP > PART_QUANT
12900                   AND  PART_NUM > :PART-TABLE                                       03/29/92
13000                 ORDER BY PART_NUM ASC                                             03/29/92
13100            END-EXEC.
13200       * CURSOR FOR ORDER LINES                                                       03/29/92
13300            EXEC SQL DECLARE NEXT_ORDER_LINE CURSOR FOR
13400                 SELECT A.ORDER_NUM,
13500                        ORDER_LINE,
13600                        QUANT_REQ
13700                 FROM   PART_ORDLN A,                                              03/29/92
13800                        PART_ORDER B
13900                 WHERE  PART_NUM  = :PART-TABLE
14000                 AND    LINE_STAT  <> 'C'                                          03/29/92
14100                 AND    A.ORDER_NUM = B.ORDER_NUM
14200                 AND    ORDER_TYPE  = 'R'
14300            END-EXEC.
14400       ******************************                                                  03/29/92
14500       *    SQL RETURN CODE HANDLING*                                                  03/29/92
14600       ******************************                                                  03/29/92
14700            EXEC SQL WHENEVER SQLERROR GO TO DB-ERROR END-EXEC.
14800            EXEC SQL WHENEVER SQLWARNING CONTINUE  END-EXEC.                          03/29/92
14900                                                                                       03/29/92
15000        MAIN-PROGRAM-PROC.                                                             03/29/92
15100       *------------------                                                             03/29/92
15200            PERFORM START-UP THRU START-UP-EXIT.                                       03/29/92
15300            PERFORM MAIN-PROC THRU MAIN-EXIT UNTIL RTCODE1 = 100.                      03/29/92
15400        END-OF-PROGRAM.                                                                03/29/92
```

図 25. COBOL プログラム例 (3/8)

```
5738PW1 V2R1M1  920327                SEU SOURCE LISTING           03/29/92  17:12:35        PAGE    4
ソース・ファイル . . . .  DRDA/QLBLSRC
メンバー . . . . . .  DDBPT6CB
SEQNBR*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 ...+... 9 ...+... 0
15500       *---------------                                                                    03/29/92
15600       ****                                                                                03/29/92
15700            EXEC SQL CONNECT RESET END-EXEC.                                               03/29/92
15800       ****
15900            CLOSE RELAT.
16000            GOBACK.
16100        MAIN-PROGRAM-EXIT. EXIT.                                                           03/29/92
16200       *------------------                                                                 03/29/92
16300                                                                                           03/29/92
16400        START-UP.                                                                          03/29/92
16500       *----------                                                                         03/29/92
16600            OPEN OUTPUT RELAT.                                                             03/29/92
16700       ****                                                                                03/29/92
16800            EXEC SQL COMMIT END-EXEC.                                                      03/29/92
16900       ****                                                                                03/29/92
17000            PERFORM CLEAN-UP THRU CLEAN-UP-EXIT.                                           03/29/92
17100       ********************************                                                    03/29/92
17200       *    CONNECT TO LOCAL DATABASE *                                                    03/29/92
17300       ********************************                                                    03/29/92
17400       ****                                                                                03/29/92
17500            EXEC SQL CONNECT TO :LOCAL-DB END-EXEC.                                        03/29/92
17600       ****                                                                                03/29/92
17700        START-UP-EXIT. EXIT.                                                               03/29/92
17800       *------------                                                                       03/29/92
17900            EJECT
18000        MAIN-PROC.
18100       *---------
18200            EXEC SQL OPEN NEXT_PART END-EXEC.                                              03/29/92
18300            EXEC SQL
18400                 FETCH NEXT_PART
18500                 INTO  :PART-TABLE,
18600                       :QUANT-TABLE,
18700                       :ROP-TABLE,
18800                       :EOQ-TABLE
18900            END-EXEC.
19000            IF SQLCODE = 100
19100               MOVE 100 TO RTCODE1                                                         03/29/92
19200               PERFORM TRAILER-PROC THRU TRAILER-EXIT                                      03/29/92
19300            ELSE
19400               MOVE 0 TO RTCODE2
19500               MOVE 0 TO QTY-REQ
19600               MOVE 0 TO QTY-REC
19700       * --- IMPLICIT "CLOSE" CAUSED BY COMMIT ---                                         03/29/92
19800       ****                                                                                03/29/92
19900            EXEC SQL COMMIT END-EXEC                                                       03/29/92
20000       ****                                                                                03/29/92
20100       ********************************                                                    03/29/92
20200       *    CONNECT TO REMOTE DATABASE *                                                   03/29/92
20300       ********************************                                                    03/29/92
```

図 25. COBOL プログラム例 (4/8)

```
5738PW1 V2R1M1  920327                SEU SOURCE LISTING           03/29/92  17:12:35         PAGE    5
ソース・ファイル . . . .  DRDA/QLBLSRC
メンバー . . . . . .  DDBPT6CB
SEQNBR*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 ...+... 9 ...+... 0
20400       ****                                                                                    03/29/92
20500               EXEC SQL CONNECT TO :REMOTE-DB END-EXEC                                         03/29/92
20600       ****                                                                                    03/29/92
20700               EXEC SQL OPEN NEXT_ORDER_LINE END-EXEC                                          03/29/92
20800               PERFORM UNTIL RTCODE2 = 100
20900                  EXEC SQL                                                                     03/29/92
21000                       FETCH NEXT_ORDER_LINE
21100                       INTO  :ORD-TABLE,
21200                             :ORL-TABLE,
21300                             :QTY-TABLE
21400                  END-EXEC
21500                  IF SQLCODE = 100
21600                     MOVE 100 TO RTCODE2
21700                     EXEC SQL CLOSE NEXT_ORDER_LINE END-EXEC
21800                  ELSE
21900                     ADD QTY-TABLE TO QTY-REQ
22000                     EXEC SQL
22100                          SELECT SUM(QUANT_RECV)                                              03/29/92
22200                          INTO   :QTY-TABLE:IND-NULL
22300                          FROM   SHIPMENTLN                                                   03/29/92
22400                          WHERE  ORDER_LOC  = :LOC
22500                          AND    ORDER_NUM  = :ORD-TABLE
22600                          AND    ORDER_LINE = :ORL-TABLE
22700                     END-EXEC
22800                     IF IND-NULL NOT < 0
22900                        ADD QTY-TABLE TO QTY-REC
23000                     END-IF
23100                  END-IF
23200               END-PERFORM
23300               IF ROP-TABLE > QUANT-TABLE + QTY-REQ - QTY-REC
23400                  PERFORM ORDER-PROC THRU ORDER-EXIT
23500               END-IF
23600            END-IF.
23700       ****                                                                                    03/29/92
23800            EXEC SQL COMMIT END-EXEC.                                                          03/29/92
23900       ****                                                                                    03/29/92
24000       **********************************                                                      03/29/92
24100       *    RECONNECT TO LOCAL DATABASE *                                                      03/29/92
24200       **********************************                                                      03/29/92
24300       ****                                                                                    03/29/92
24400            EXEC SQL CONNECT TO :LOCAL-DB END-EXEC.                                           03/29/92
24500       ****                                                                                    03/29/92
24600        MAIN-EXIT. EXIT.
24700       *---------------
24800        ORDER-PROC.
24900       *----------
25000            IF FIRST-TIME
25100               MOVE '2' TO WHAT-TIME
25200               PERFORM CREATE-ORDER-PROC THRU CREATE-ORDER-EXIT.                              03/29/92
25300            ADD 1 TO CONTL.
```

図 25. COBOL プログラム例 (5/8)

```
5738PW1 V2R1M1  920327                SEU SOURCE LISTING           03/29/92  17:12:35        PAGE    7
ソース・ファイル . . . .  DRDA/QLBLSRC
メンバー . . . . . .  DDBPT6CB
SEQNBR*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 ...+... 9 ...+... 0
25400           EXEC SQL
25500                INSERT
25600                INTO    PART_ORDLN                                                          03/29/92
25700                       (ORDER_NUM,
25800                        ORDER_LINE,
25900                        PART_NUM,
26000                        QUANT_REQ,
26100                        LINE_STAT)
26200                VALUES (:NEXT-NUM,
26300                        :CONTL,
26400                        :PART-TABLE,
26500                        :EOQ-TABLE,
26600                        'O')
26700           END-EXEC.
26800           PERFORM DETAIL-PROC THRU DETAIL-EXIT.
26900       ORDER-EXIT. EXIT.
27000       *---------------
27100                                                                                           03/29/92
27200        CREATE-ORDER-PROC.                                                                 03/29/92
27300       *-----------------                                                                  03/29/92
27400       *GET NEXT ORDER NUMBER                                                              03/29/92
27500           EXEC SQL                                                                       03/29/92
27600                SELECT (MAX(ORDER_NUM) + 1)                                               03/29/92
27700                INTO   :NEXT-NUM:IND-NULL                                                 03/29/92
27800                FROM   PART_ORDER                                                         03/29/92
27900           END-EXEC.                                                                      03/29/92
28000           IF IND-NULL < 0                                                                03/29/92
28100             MOVE 1 TO NEXT-NUM.                                                          03/29/92
28200           EXEC SQL                                                                       03/29/92
28300                INSERT                                                                    03/29/92
28400                INTO    PART_ORDER                                                        03/29/92
28500                       (ORDER_NUM,                                                       03/29/92
28600                        ORIGIN_LOC,                                                      03/29/92
28700                        ORDER_TYPE,                                                      03/29/92
28800                        ORDER_STAT,                                                      03/29/92
28900                        CREAT_TIME)                                                      03/29/92
29000                VALUES (:NEXT-NUM,                                                        03/29/92
29100                        :LOC, 'R', 'O',                                                  03/29/92
29200                        CURRENT TIMESTAMP)                                               03/29/92
29300              END-EXEC.                                                                   03/29/92
29400           MOVE NEXT-NUM TO MASK0.                                                        03/29/92
29500           PERFORM HEADER-PROC THRU HEADER-EXIT.                                          03/29/92
29600        CREATE-ORDER-EXIT. EXIT.                                                          03/29/92
29700       *-----------------                                                                  03/29/92
29800                                                                                          03/29/92
29900        DB-ERROR.                                                                         03/29/92
30000       *--------                                                                          03/29/92
30100           PERFORM ERROR-MSG-PROC THRU ERROR-MSG-EXIT.                                    03/29/92
30200       **********************                                                             03/29/92
30300       *    ROLLBACK THE LUW *                                                            03/29/92
```

図 25. COBOL プログラム例 (6/8)

```
5738PW1 V2R1M1  920327                SEU SOURCE LISTING           03/29/92  17:12:35         PAGE    8
ソース・ファイル . . . .  DRDA/QLBLSRC
メンバー . . . . . .  DDBPT6CB
SEQNBR*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 ...+... 9 ...+... 0
30400       **********************                                                                 03/29/92
30500            EXEC SQL WHENEVER SQLERROR CONTINUE END-EXEC.                                     03/29/92
30600       ****                                                                                   03/29/92
30700            EXEC SQL ROLLBACK WORK END-EXEC.                                                  03/29/92
30800       ****                                                                                   03/29/92
30900            PERFORM END-OF-PROGRAM THRU MAIN-PROGRAM-EXIT.                                    03/29/92
31000       * -- NEXT LINE INCLUDED TO RESET THE "GO TO" DEFAULT --                                03/29/92
31100            EXEC SQL WHENEVER SQLERROR GO TO DB-ERROR END-EXEC.                               03/29/92
31200                                                                                              03/29/92
31300        ERROR-MSG-PROC.                                                                       03/29/92
31400       *----------                                                                            03/29/92
31500            MOVE  SQLCODE   TO  MSG-ID-2.                                                     03/29/92
31600            DISPLAY 'SQL STATE =' SQLSTATE ' SQLCODE =' MSG-ID-2.                             03/29/92
31700       * -- ADD HERE ANY ADDITIONAL ERROR MESSAGE HANDLING --                                 03/29/92
31800        ERROR-MSG-EXIT. EXIT.                                                                 03/29/92
31900       *----------------                                                                      03/29/92
32000                                                                                              03/29/92
32100       ******************                                                                     03/29/92
32200       * REPORT PRINTING *                                                                    03/29/92
32300       ******************                                                                     03/29/92
32400        HEADER-PROC.                                                                          03/29/92
32500       *-----------                                                                           03/29/92
32600            WRITE REPREC FROM LINE1 AFTER ADVANCING PAGE.
32700            WRITE REPREC FROM LINE2 AFTER ADVANCING 3 LINES.
32800            WRITE REPREC FROM LINE3 AFTER ADVANCING 2 LINES.
32900            WRITE REPREC FROM LINE4 AFTER ADVANCING 1 LINES.
33000            WRITE REPREC FROM LINE5 AFTER ADVANCING 1 LINES.
33100            WRITE REPREC FROM LINE3 AFTER ADVANCING 1 LINES.
33200            WRITE REPREC FROM LINE0 AFTER ADVANCING 1 LINES.
33300        HEADER-EXIT. EXIT.
33400       *-----------------
33500        DETAIL-PROC.
33600       *-----------
33700            ADD 1 TO CONTD.
33800            IF CONTD > 50
33900               MOVE 1 TO CONTD
34000               PERFORM HEADER-PROC THRU HEADER-EXIT
34100            END-IF
34200            MOVE CONTL     TO MASK1.
34300            MOVE EOQ-TABLE TO MASK2.
34400            WRITE REPREC FROM LINE6 AFTER ADVANCING 1 LINES.
34500        DETAIL-EXIT. EXIT.
34600       *-----------------
34700        TRAILER-PROC.
34800       *------------
34900            MOVE CONTL TO MASK3.
35000            WRITE REPREC FROM LINE3 AFTER ADVANCING 2 LINES.
35100            WRITE REPREC FROM LINE7 AFTER ADVANCING 2 LINES.
35200            WRITE REPREC FROM LINE3 AFTER ADVANCING 2 LINES.
35300            WRITE REPREC FROM LINE8 AFTER ADVANCING 1 LINES.
35400        TRAILER-EXIT. EXIT.
35500       *------------------
```

図 25. COBOL プログラム例 (7/8)

```
5738PW1 V2R1M1  920327                SEU SOURCE LISTING           03/29/92  17:12:35        PAGE    8
ソース・ファイル . . . .  DRDA/QLBLSRC
メンバー . . . . . .  DDBPT6CB
SEQNBR*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 ...+... 9 ...+... 0
35600       ******************************************************                                  03/29/92
35700       * THIS PARAGRAPH IS ONLY REQUIRED IN A TEST ENVIRONMENT*                                03/29/92
35800       * TO RESET THE DATA TO PERMIT RE-RUNNING OF THE TEST   *                                03/29/92
35900       ******************************************************                                  03/29/92
36000        CLEAN-UP.                                                                             03/29/92
36100       *---------                                                                             03/29/92
36200       *********************************                                                     03/29/92
36300       *    CONNECT TO REMOTE DATABASE *                                                      03/29/92
36400       *********************************                                                     03/29/92
36500       ****                                                                                  03/29/92
36600            EXEC SQL CONNECT TO :REMOTE-DB END-EXEC.                                         03/29/92
36700       ****                                                                                  03/29/92
36800       *--------------------DELETE ORDER ROWS FOR RERUNABILITY                                03/29/92
36900               EXEC SQL                                                                      03/29/92
37000                    DELETE                                                                   03/29/92
37100                    FROM    PART_ORDLN                                                       03/29/92
37200                    WHERE   ORDER_NUM IN                                                     03/29/92
37300                               (SELECT  ORDER_NUM                                            03/29/92
37400                                FROM    PART_ORDER                                           03/29/92
37500                                WHERE   ORDER_TYPE = 'R')                                    03/29/92
37600               END-EXEC.                                                                     03/29/92
37700               EXEC SQL                                                                      03/29/92
37800                    DELETE                                                                   03/29/92
37900                    FROM    PART_ORDER                                                       03/29/92
38000                    WHERE   ORDER_TYPE = 'R'                                                 03/29/92
38100               END-EXEC.                                                                     03/29/92
38200       ****                                                                                  03/29/92
38300            EXEC SQL COMMIT END-EXEC.                                                        03/29/92
38400       ****                                                                                  03/29/92
38500        CLEAN-UP-EXIT. EXIT.                                                                 03/29/92
38600       *-------------                                                                         03/29/92
* * * *  ソース仕様の終わり  * * * *
```

図 25. COBOL プログラム例 (8/8)

例: C プログラム

注: xv ページの『コードの特記事項情報』に重要なリーガル情報がありますので、お読みください。

```
5738PW1 V2R1M1  920327                SEU SOURCE LISTING           03/29/92  17:11:13         PAGE    1
ソース・ファイル . . . .  DRDA/QCSRC
メンバー . . . . . .  DDBPT6C
SEQNBR*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 ...+... 9 ...+... 0
100      /*********************************************************************/                03/29/92
200      /*   MODULE NAME = DDBPT6C                                           */                03/29/92
300      /*                                                                   */                03/29/92
400      /*   DESCRIPTIVE NAME:  D-DB SAMPLE APPLICATION                      */                03/29/92
500      /*                      REORDER POINT PROCESSING                     */                03/29/92
600      /*                      AS/400                                       */                03/29/92
700      /*                      C/400                                        */                03/29/92
800      /*                                                                   */                03/29/92
900      /*   FUNCTION:   THIS MODULE PROCESS THE PART_STOCK TABLE AND        */                03/29/92
1000      /*              FOR EACH PART BELOW THE ROP (REORDER POINT)          */               03/29/92
1100      /*              CREATES A SUPPLY ORDER.                              */               03/29/92
1200      /*                                                                   */               03/29/92
1300      /*   OUTPUT:  BATCH : SPOOLFILE                                      */               03/29/92
1400      /*            INTER : DISPLY                                         */               03/29/92
1500      /*                                                                   */               03/29/92
1600      /*   LOCAL TABLES:    PART_STOCK                                     */               03/29/92
1700      /*                                                                   */               03/29/92
1800      /*   REMOTE TABLES:   PART_ORDER, PART_ORDLN, SHIPMENTLN             */               03/29/92
1900      /*                                                                   */               03/29/92
2000      /*   COMPILE OPTIONS:                                                */               03/29/92
2100      /*   CRTSQLC    PGM(DDBPT6C) COMMIT(*CHG) RDB(rdbname)               */               03/29/92
2200      /*                                                                   */               03/29/92
2300      /*   INVOKED BY: CALL PGM(DDBPT6C) PARM('lcldbname' 'rmtdbname')     */               03/29/92
2400      /*********************************************************************/               03/29/92
2500                                                                                           03/29/92
2600 #include <stdlib.h>
2700 #include <string.h>                                                                       03/29/92
2800 #include <stdio.h>
2900                                                                                           03/29/92
3000               EXEC SQL BEGIN DECLARE SECTION;                                            03/29/92
3100                                                                                          03/29/92
3200       char loc       [4] = "SQLA";            /* dealer's database name */
3300       char remote_db [18] = "              "; /* sample remote database */
3400       char local_db [18] = "               ";  /* sample local  database */
3500                                                                                          03/29/92
3600       char part_table [5] = " ";      /* part number in table part_stock */
3700       long  quant_table;             /* quantity in stock, tbl part_stock   */          03/29/92
3800       long  rop_table;              /* reorder point    , tbl part_stock     */          03/29/92
3900       long  eoq_table;              /* reorder quantity , tbl part_stock     */          03/29/92
4000                                                                                          03/29/92
4100       short next_num;               /* next order nbr,table part_order       */          03/29/92
4200                                                                                          03/29/92
4300       short  ord_table;             /* order nbr.       , tbl order_line     */          03/29/92
4400       short  orl_table;             /* order line       , tbl order_line     */          03/29/92
4500       long qty_table;               /* ordered quantity , tbl order_line     */          03/29/92
4600       long line_count = 0;          /* total number of order lines           */          03/29/92
4700       short ind_null;               /* null indicator for qty_table          */          03/29/92
4800       short  contl = 0;             /* continuation line, tbl order_line     */          03/29/92
4900                                                                                          03/29/92
5000               EXEC SQL END   DECLARE SECTION;                                            03/29/92
5100               EXEC SQL INCLUDE SQLCA;                                                    03/29/92
5200               EXEC SQL WHENEVER SQLERROR go to error_tag;                                03/29/92
5300               EXEC SQL WHENEVER SQLWARNING CONTINUE;                                     03/29/92
5400                                                                                          03/29/92
```

図 26. C プログラム例 (1/5)

```
5738PW1 V2R1M1  920327                SEU SOURCE LISTING           03/29/92  17:11:13        PAGE    2
ソース・ファイル . . . .  DRDA/QCSRC
メンバー . . . . . .  DDBPT6C
SEQNBR*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 ...+... 9 ...+... 0
5500 /*****************************************************************************/              03/29/92
5600 /* Other Variables                                                           */              03/29/92
5700 /*****************************************************************************/              03/29/92
5800                                                                                             03/29/92
5900       char first_time, what_time;                                                           03/29/92
6000       long  qty_rec = 0, qty_req = 0;                                                       03/29/92
6100                                                                                             03/29/92
6200 /*****************************************************************************/              03/29/92
6300 /* Function Declaration                                                      */              03/29/92
6400 /*****************************************************************************/              03/29/92
6500                                                                                             03/29/92
6600 declare_cursor () {                                                                         03/29/92
6700                                                                                             03/29/92
6800 /* SQL Cursor declaration and reposition for local UW                        */              03/29/92
6900                                                                                             03/29/92
7000                EXEC SQL DECLARE NEXT_PART CURSOR FOR                                        03/29/92
7100                         SELECT PART_NUM, PART_QUANT, PART_ROP, PART_EOQ                     03/29/92
7200                         FROM   PART_STOCK                                                   03/29/92
7300                         WHERE  PART_ROP > PART_QUANT                                        03/29/92
7400                         AND    PART_NUM > :part_table                                       03/29/92
7500                         ORDER BY PART_NUM;                                                  03/29/92
7600 /* SQL Cursor declaration and connect for RUW                                */              03/29/92
7700                                                                                             03/29/92
7800                EXEC SQL DECLARE NEXT_OLINE CURSOR FOR                                       03/29/92
7900                         SELECT A.ORDER_NUM, ORDER_LINE, QUANT_REQ                           03/29/92
8000                         FROM   PART_ORDLN A,                                                03/29/92
8100                                PART_ORDER B                                                 03/29/92
8200                         WHERE  PART_NUM  = :part_table                                      03/29/92
8300                         AND    LINE_STAT  <> 'C'                                            03/29/92
8400                         AND    A.ORDER_NUM = B.ORDER_NUM                                    03/29/92
8500                         AND    ORDER_TYPE  = 'R';                                           03/29/92
8600                                                                                             03/29/92
8700  /* upline exit function in connectable state                                */              03/29/92
8800                                                                                             03/29/92
8900                EXEC SQL COMMIT;                                                             03/29/92
9000 goto function_exit;                                                                         03/29/92
9100 error_tag:                                                                                  03/29/92
9200        error_function();                                                                    03/29/92
9300 function_exit: ;                                                                            03/29/92
9400  }  /* function declare_cursor                                                 */              03/29/92
9500                                                                                             03/29/92
9600 delete_for_rerun () {                                                                       03/29/92
9700                                                                                             03/29/92
```

図 26. C プログラム例 (2/5)

```
5738PW1 V2R1M1  920327                SEU SOURCE LISTING           03/29/92  17:11:13         PAGE    3
ソース・ファイル . . . .  DRDA/QCSRC
メンバー . . . . . .  DDBPT6C
SEQNBR*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 ...+... 9 ...+... 0
9800 /* Clean up for rerunability in test environment                          */              03/29/92
9900                EXEC SQL CONNECT TO :remote_db;                                          03/29/92
10000                EXEC SQL  DELETE                                                        03/29/92
10100                          FROM    PART_ORDLN                                        03/29/92
10200                          WHERE   ORDER_NUM IN                                      03/29/92
10300                                  (SELECT  ORDER_NUM                                03/29/92
10400                                   FROM    PART_ORDER                               03/29/92
10500                                   WHERE   ORDER_TYPE = 'R');                       03/29/92
10600                EXEC SQL  DELETE                                                        03/29/92
10700                          FROM    PART_ORDER                                        03/29/92
10800                          WHERE   ORDER_TYPE = 'R';                                 03/29/92
10900  /* upline exit function in connectable state                              */              03/29/92
11000                EXEC SQL COMMIT;                                                        03/29/92
11100                EXEC SQL CONNECT TO :local_db;                                          03/29/92
11200 goto function_exit;                                                                    03/29/92
11300 error_tag:                                                                             03/29/92
11400        error_function();                                                               03/29/92
11500 function_exit: ;                                                                       03/29/92
11600  }  /* function delete_for_rerun                                              */              03/29/92
11700                                                                                        03/29/92
11800 calculate_order_quantity () {                                                          03/29/92
11900                                                                                        03/29/92
12000 /* available qty = Stock qty + qty in order - qty received                    */              03/29/92
12100                                                                                        03/29/92
12200       EXEC SQL    OPEN NEXT_PART;                                                      03/29/92
12300       EXEC SQL    FETCH NEXT_PART                                                      03/29/92
12400                   INTO  :part_table, :quant_table, :rop_table, :eoq_table;             03/29/92
12500                                                                                        03/29/92
12600            if (sqlca.sqlcode == 100) {                                                 03/29/92
12700               printf("--------------------------------¥n");                          03/29/92
12800               printf("NUMBER OF LINES CREATED = %d\n",line_count);                    03/29/92
12900               printf("--------------------------------¥n");                          03/29/92
13000               printf("*****  END OF PROGRAM  *********¥n");                          03/29/92
13100               rop_table = 0;            /* no (more) orders to process      */              03/29/92
13200            }                                                                           03/29/92
13300            else  {      qty_rec = 0;                                                   03/29/92
13400                         qty_req = 0;                                                   03/29/92
13500 /*                                                                             */              03/29/92
13600                   EXEC SQL    COMMIT;                                                  03/29/92
13700                   EXEC SQL CONNECT TO :remote_db;                                      03/29/92
13800                   EXEC SQL OPEN NEXT_OLINE;                                            03/29/92
13900                   do {                                                                 03/29/92
14000                         EXEC SQL  FETCH NEXT_OLINE                                     03/29/92
14100                                   INTO  :ord_table, :orl_table, :qty_table;            03/29/92
14200                         qty_rec = qty_rec + qty_table;                                 03/29/92
14300                      } while(sqlca.sqlcode != 100);                                    03/29/92
14400                      EXEC SQL CLOSE NEXT_OLINE;                                        03/29/92
```

図 26. C プログラム例 (3/5)

```
5738PW1 V2R1M1  920327                SEU SOURCE LISTING           03/29/92  17:11:13         PAGE    4
ソース・ファイル . . . .  DRDA/QCSRC
メンバー . . . . . .  DDBPT6C
SEQNBR*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 ...+... 9 ...+... 0
14500                     EXEC SQL  SELECT SUM(QUANT_RECV)                                      03/29/92
14600                               INTO   :qty_table:ind_null                                 03/29/92
14700                               FROM   SHIPMENTLN                                          03/29/92
14800                               WHERE  ORDER_LOC  = :loc                                   03/29/92
14900                               AND    ORDER_NUM  = :ord_table                             03/29/92
15000                               AND    ORDER_LINE = :orl_table;                            03/29/92
15100                     if (ind_null != 0)                                                   03/29/92
15200                         qty_rec = qty_rec + qty_table;                                   03/29/92
15300                 } /* end of else branch                                    */           03/29/92
15400 goto function_exit;                                                                    03/29/92
15500 error_tag:                                                                             03/29/92
15600       error_function();                                                                 03/29/92
15700 function_exit: ;                                                                       03/29/92
15800 } /* end of calculate_order_quantity                                        */           03/29/92
15900                                                                                        03/29/92
16000 process_order () {                                                                     03/29/92
16100                                                                                        03/29/92
16200 /* insert order and order_line in remote database                          */           03/29/92
16300                                                                                        03/29/92
16400     if (contl == 0) {                                                                  03/29/92
16500                                                                                        03/29/92
16600           EXEC SQL  SELECT (MAX(ORDER_NUM) + 1)                                        03/29/92
16700                     INTO   :next_num                                                   03/29/92
16800                     FROM   PART_ORDER;                                                 03/29/92
16900           EXEC SQL INSERT INTO    PART_ORDER                                           03/29/92
17000                    (ORDER_NUM, ORIGIN_LOC, ORDER_TYPE, ORDER_STAT, CREAT_TIME)         03/29/92
17100                     VALUES (:next_num, :loc, 'R', 'O', CURRENT TIMESTAMP);             03/29/92
17200           printf("*****  ROP PROCESSING  *********¥n");                                03/29/92
17300           printf("ORDER NUMBER = %d ¥n¥n",next_num);                                   03/29/92
17400           printf("--------------------------------¥n");                               03/29/92
17500           printf("   LINE     PART       QTY      ¥n");                               03/29/92
17600           printf("   NBR      NBR        REQUESTED¥n");                              03/29/92
17700           printf("--------------------------------¥n");                               03/29/92
17800           contl = contl + 1;                                                           03/29/92
17900        }  /* if contl == 0                                                 */          03/29/92
18000                                                                                        03/29/92
18100     EXEC SQL INSERT INTO PART_ORDLN                                                    03/29/92
18200              (ORDER_NUM, ORDER_LINE, PART_NUM, QUANT_REQ, LINE_STAT)                   03/29/92
18300              VALUES (:next_num, :contl, :part_table, :eoq_table, 'O');                 03/29/92
18400     line_count = line_count + 1;                                                       03/29/92
18500     printf("   %d        %.5s      %d¥n",                                             03/29/92
18600             line_count,part_table,eoq_table);                                        03/29/92
18700     contl = contl + 1;                                                                 03/29/92
18800                                                                                        03/29/92
18900  /* upline exit function in connectable state                               */          03/29/92
19000           EXEC SQL COMMIT;                                                             03/29/92
```

図 26. C プログラム例 (4/5)

```
5738PW1 V2R1M1  920327                SEU SOURCE LISTING           03/29/92  17:11:13         PAGE    5
ソース・ファイル . . . .  DRDA/QCSRC
メンバー . . . . . .  DDBPT6C
SEQNBR*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 ...+... 9 ...+... 0
19100 /* RECONNECT TO LOCAL DATABASE                                          */             03/29/92
19200            EXEC SQL CONNECT TO :local_db;                                              03/29/92
19300                                                                                        03/29/92
19400 goto function_exit;                                                                    03/29/92
19500 error_tag:                                                                             03/29/92
19600          error_function();                                                             03/29/92
19700 function_exit: ;                                                                       03/29/92
19800 }  /* end of function process_order                                          */             03/29/92
19900                                                                                        03/29/92
20000 error_function () {                                                                    03/29/92
20100 /*                                                                            */             03/29/92
20200            printf("************************¥n");                                       03/29/92
20300            printf("*      SQL ERROR       *¥n");                                       03/29/92
20400            printf("************************¥n");                                       03/29/92
20500            printf("SQLCODE    = %d¥n",sqlca.sqlcode);                                  03/29/92
20600            printf("SQLSTATE    = %5s",sqlca.sqlstate);                                 03/29/92
20700            printf("¥n**********************¥n");                                       03/29/92
20800            EXEC SQL WHENEVER SQLERROR CONTINUE;                                        03/29/92
20900            EXEC SQL ROLLBACK;                                                          03/29/92
21000            EXEC SQL CONNECT RESET;                                                     03/29/92
21100 exit (999);                                                                            03/29/92
21200 }                                                                                      03/29/92
21300                                                                                        03/29/92
21400 main(int argc, char *argv[])  {
21500   memcpy(local_db,argv[1],strlen(argv[1]));
21600   memcpy(remote_db,argv[2],strlen(argv[2]));
21700 /* clean up                                                                  */             03/29/92
21800               declare_cursor();                                                        03/29/92
21900               delete_for_rerun();                                                      03/29/92
22000                                                                                        03/29/92
22100 /* main-line,  state is connectable                                         */             03/29/92
22200                                                                                        03/29/92
22300      do {                                                                              03/29/92
22400                 calculate_order_quantity ();                                           03/29/92
22500                 if (rop_table > quant_table + qty_req - qty_rec) {                     03/29/92
22600                     process_order();                                                   03/29/92
22700                     quant_table = qty_req = qty_rec = 0;                               03/29/92
22800                 }                                                                      03/29/92
22900         } while (sqlca.sqlcode == 0);                                                  03/29/92
23000 /* RECONNECT TO APPLICATION SERVER                                          */             03/29/92
23100            EXEC SQL CONNECT RESET;                                                     03/29/92
23200 exit(0);                                                                               03/29/92
23300                                                                                        03/29/92
23400                                                                                        03/29/92
23500 } /* end of main                                                             */             03/29/92
* * * *  ソース仕様の終わり  * * * *
```

図 26. C プログラム例 (5/5)

例: プログラム出力

注: xv ページの『コードの特記事項情報』に重要なリーガル情報がありますので、お読みください。

```
*****  ROP PROCESSING  *********
ORDER NUMBER = 6
--------------------------------
LINE     PART       QTY
NBR      NBR        REQUESTED
--------------------------------
1        14020      100
2        14030      50
3        18020      50
4        21010      50
5        37020      40
--------------------------------
NUMBER OF LINES CREATED = 5
--------------------------------
*****  END OF PROGRAM  *********
```

図 27. プログラム出力例

付録 B. DRDA を使用したプラットフォーム間アクセス

本書では、iSeries server のネットワーク (*同種* 環境) における分散リレーショナル・データベースのための iSeries サポートについて集中的に説明しています。 DRDA をサポートするプラットフォームから成るネットワークの中には、分散リレーショナル・データベースを実装したものが数多く存在しています。この付録では、iSeries server を*異種* DRDA 環境で使用する際に必要とされるヒントおよび技法のリストを示します。

この付録では、他の特定の IBM プロダクトを処理するときに考慮が必要とされるいくつかの状況について説明します。ただし、この付録の目的は包括的なリストを提示することではありません。この付録で説明されているような問題または条件の多くは、アプリケーションに左右されることが少なくありません。様々な IBM プラットフォーム間の相違点の詳細については、 *DB2 UDB SQL 解説書 (第 2 巻)* (SD88-7269)、または *DRDA 適用業務プログラミングの手引き* (N:SC26-4773) を参照してください。

DRDA を使用したプラットフォーム間アクセスについて詳しくは、以下のトピックを参照してください。

- CCSID に関する考慮事項
- 異種環境アプリケーション・サーバー上での対話式 SQL および Query 管理機能のセットアップ
- DB2 コネクト (DB2 Connect) ユーザーからよく尋ねられる質問
- DB2 コネクトおよび DB2 UDB for iSeries を使用してワークステーションを操作するための他のヒント
- **DB2 UDB Server for VM での対話式 SQL パッケージの作成**

  DB2 UDB Server for VM では、コレクション名はユーザー ID と同義です。 DB2 UDB Server for VM アプリケーション・サーバー上で対話式 SQL または iSeries Query Manager によって使用されるパッケージを作成するには、 OS/400 システム上で QSQL400 というユーザー ID を作成してください。このユーザー ID を使用すれば、 DB2 UDB Server for VM アプリケーション・サーバー上で必要なパッケージをすべて作成することができます。そうすれば、ユーザーは自分自身のユーザー ID を使用して、OS/400 上で、対話式 SQL または iSeries Query Manager によって DB2 UDB Server for VM にアクセスすることができます。

CCSID に関する考慮事項

異種環境で分散リレーショナル・データベースを処理するにあたっては、コード化文字セット識別子 (CCSID) を適正にセットアップおよび使用することが必要です。 iSeries server では、出荷時にデフォルト値が設定されていますが、異種環境で使用する場合には、これを変更する必要がある場合があります。また、サーバーでサポートしている DBCS の CCSID の中には、 DB2 UDB for z/OS および DB2 UDB Server for VM のデータベース・マネージャーではサポートされていないものがあります。このセクションでは、上記の 2 つの条件について説明し、それに対処する方法を示します。

詳細は、以下のセクションを参照してください。

- iSeries システム値 QCCSID
- DB2 UDB for z/OS および DB2 UDB server for VM データベース・マネージャーの CCSID 変換に関する考慮事項
- **非 iSeries アプリケーション・リクエスター (AR) のための CCSID 変換に関する考慮事項**

非 iSeries アプリケーション・リクエスター (AR) から DB2 UDB for iSeries Application Server (AS) に接続する場合、CCSID 65535 でタグ付けされている列は変換できません。これらの列が入っているファイルに明示的に CCSID を指示した列が含まれていない場合は、すべての文字列の CCSID を別の CCSID の値に変更することができます。 CCSID を変更するには、物理ファイルの変更 (CHGPF) コマンドを使用します。 物理ファイルを基礎にする論理ファイルが作成されている場合は、受け取ったエラー・メッセージ (CPD322D) の回復セクションで示されている指示に従ってください。

iSeries システム値 QCCSID

iSeries server は、出荷時に QCCSID システム値が 65535 に設定されています。この CCSID でタグ付けされたデータは、受信側サーバーで変換することができません。 iSeries server **アプリケーション・リクエスター (AR)** がこの CCSID を使用していると、異種サーバーに接続できない場合があります。また、この CCSID でタグ付けされているソース・ファイルは、異種サーバー上でアプリケーションを作成するために使用できない場合もあります。

203 ページの『コード化文字セット識別子 (CCSID)』で述べているように、接続時点で使用される CCSID は、ジョブの CCSID によって決まります。ジョブが開始されると、ジョブの CCSID は、そのジョブが実行されるユーザー・プロファイルによって決まります。ユーザー・プロファイルは、デフォルトと同様に、システム値 QCCSID を使用することができます。

サーバーのデフォルト CCSID をサポートしないサーバーに接続する場合は、ジョブ CCSID を変更する必要があります。 ジョブ変更 (CHGJOB) コマンドを使用することによって、ジョブの CCSID を変更することができます。 ただし、この解決法が有効なのは、現在処理中のジョブの場合だけです。次にはまたジョブの CCSID を変更しなければなりません。

分散リレーショナル・データベースの中で使用されるユーザー・プロファイルで指定されている CCSID を変更すれば、上記の方法に比べてより永続的な解決策になります。ユーザー・プロファイルを変更すると、データを変換する必要のあるユーザーだけに影響が生じます。 DB2 Universal Database for iSeries **アプリケーション・サーバー (AS)** を使用する場合は、AS で使用するユーザー・プロファイルを変更する必要があります。

異なるアプリケーション・リクエスター (AR) が、ジョブ CCSID 65535 を使用して DB2 UDB for iSeries Application Server (AS) に接続する場合、ジョブは、ジョブ・デフォルト CCSID を使用するように切り替えられます。ジョブ・デフォルト CCSID は、ジョブの言語 ID (LANDID) によって決定されます。この場合は、ジョブの CCSID を 65535 以外の値に切り替えたほうが高いパフォーマンスが得られます。たとえば、サーバー・ジョブが実行されるユーザー・プロファイルの値に CCSID 値を変更できます。

ユーザー・プロファイルの中のデフォルトの CCSID は *SYSVAL です。これは QCCSID システム値を参照します。 このシステム値、すなわち、すべてのユーザー・プロファイルによって使用されるデフォルト値は、 サーバー値変更 (CHGSYSVAL) コマンドを使って変更することができます。 このようにする場合、サーバー上のほとんどの (すべてではないが) ユーザーが使用できる CCSID を選択するとよいでしょう。 使用できる CCSID と、それによって表される言語のリストについては、『各国語サポート』を参照してください。

サーバーまたはジョブによって使用される CCSID をサポートしていないサーバーで作業を行っていると思われる場合には、ジョブ・ログ、SQLCA または SQL 診断域の中に次の標識があるかどうか探索してください。

メッセージ

SQ30073

| **SQLCODE または DB2_RETURNED_SQLCODE**
| -30073

| **SQLSTATE**
| 58017

| **テキスト**
| 分散データ管理 (DDM) パラメーター値 X'0035' はサポートされていない。

| **メッセージ**
| SQL0332

| **SQLCODE または DB2_RETURNED_SQLCODE**
| -332

| **SQLSTATE**
| 57017

| **テキスト**
| CCSID &1 と CCSID &2 の間の文字変換は正しくない。

DB2 UDB for z/OS および DB2 UDB server for VM データベース・マネージャーの CCSID 変換に関する考慮事項

DB2 Universal Database for iSeries と他の DB2* データベースの間の違いの 1 つに、 iSeries システムの方がより大きな CCSID のセットをサポートすることが挙げられます。そのため、他のデータベース・マネージャーがデータに対して文字変換を行おうとすると、エラーとなる可能性があります (SQLCODE -332 および SQLSTATE 57017)。

DB2 UDB SQL テーブルの中の特定のフィールドには、 DBCS 混用データ・タイプを持つよう定義することができます。 これは、 2 バイト文字セット (DBCS) 文字と 1 バイト文字セット (SBCS) 文字を両方とも使用できるデータ・タイプです。これらのフィールド・タイプの CCSID は、サーバーに付属しているデフォルトの CCSID に基づいています。

このフィールドが DB2 UDB for z/OS または DB2 UDB Server for VM **アプリケーション・リクエスター (AR)** から選択されると、 DB2 UDB for z/OS および DB2 UDB Server for VM データベースではこの CCSID への変換をサポートしていないことがあるために、 SELECT ステートメントが正常に実行されない場合があります。

このエラーを避けるためには、DB2 UDB for z/OS データベースまたは DB2 UDB Server for VM AR を変更して、以下のいずれかで実行されるようにしなければなりません。

- iSeries SQL カタログ・テーブル内の DBCS-OPEN フィールドと同じ混合バイト CCSID。
- iSeries SQL カタログ・テーブル内の DBCS-OPEN フィールドの混合バイト CCSID からのデータであるときに、サーバーでデータの変換ができる CCSID。 この CCSID は、 iSeries SQL カタログ・テーブルの DBCS-OPEN フィールド内のデータがすべて 1 バイト・データである場合には、 1 バイト CCSID になります。

サーバーに正しい変更を加えられるようにするために、 DB2 UDB for z/OS または DB2 UDB Server for VM 上でサポートされている CCSID 変換の分析が必要です。このエラーの処理方法に関する固有の情報については、*DB2 UDB for z/OS Administration Guide* を参照してください。

異種環境アプリケーション・サーバー上での対話式 SQL および Query 管理機能のセットアップ

対話式 SQL および iSeries Query Manager は、ユーザーの実行オプション (日付形式、コミットメント制御など) に基づいて、異種環境アプリケーション・サーバー上で必要に応じてパッケージを作成します。これらのパッケージは、アプリケーション・サーバー上で QSQL400 と呼ばれるコレクションに作成されます。パッケージ名は QSQLabcd であり、 'abcd' は、そのパッケージに使用される特定のオプションを参照する数字に対応します。

'abcd' の値は以下のようにオプションに対応しています。

位置	オプション	値
a	日付形式	0 = ISO、JIS 日付形式。 1 = USA 日付形式。 2 = EUR 日付形式。
b	時刻形式	0 = JIS 時刻形式。 1 = USA 時刻形式。 2 = EUR、ISO 時刻形式。
c	コミットメント制御小数点文字	0 = *CS コミットメント制御ピリオド小数点文字。 1 = *CS コミットメント制御コンマ小数点文字。 2 = *RR コミットメント制御ピリオド小数点文字。 3 = *RR コミットメント制御コンマ小数点文字。
d	ストリング区切り文字のデフォルト文字サブタイプ	0 = アポストロフィ・ストリング区切り文字、1 バイト文字サブタイプ。 1 = アポストロフィ・ストリング区切り文字、2 バイト文字サブタイプ。 2 = 2 重引用符ストリング区切り文字、1 バイト文字サブタイプ。 3 = 2 重引用符ストリング区切り文字、2 バイト文字サブタイプ。

たとえば、パッケージが次のオプションを指定して、対話式 SQL により異種環境アプリケーション・サーバーに作成されたとします。 USA 日付形式、USA 時刻形式、*CS のコミットメント制御レベル、小数点文字がピリオド、ストリング区切り文字がアポストロフィー、デフォルト文字サブタイプが 1 バイト文字とすると、パッケージの名前は 'QSQL1100' になります。パッケージが特定のオプションのセットを使用して作成されると、そのアプリケーション・サーバーに対して同じオプションを使用して実行する対話式 SQL または iSeries Query Manager ユーザーは、すべてそのパッケージを使用します。

別の箇所で指摘しましたが、パッケージが作成されたときに**アプリケーション・サーバー (AS)** への更新可能な接続がなければなりません。パッケージを作成するために、 AS に接続する前に RELEASE ALL および COMMIT を実行する必要があるかもしれません。

非 iSeries アプリケーション・リクエスター (AR) ユーザーからよく尋ねられる質問 (FAQ)

このセクションでは、非 iSeries アプリケーション・リクエスター (AR) を介して iSeries のデータにアクセスしようとするワークステーション・ユーザーからよく尋ねられる、以下のような質問に答えていきます。

- 『DB2 コネクトから接続しようとするとメッセージ SQL5048N が戻されるのはなぜか?』
- iSeries ファイルはジャーナルする必要があるか?
- パフォーマンスを向上させるために、照会データはどのような場合にブロック化されるのか?
- DBM SQL0969N エラー・メッセージで報告される SQLCODE と関連のトークンの解釈方法は?
- WHERE 文節のホスト変数のタイプはパフォーマンスにどのような影響を与えるのか?
- ライブラリー・リストを使用して、修飾されていないテーブルおよびビューの名前を解決できるか?
- CCSID についてはどんなことを考慮しなければならないか?
- DB2 コネクトのユーザーは、通常の EBCDIC 順序の代わりに、 iSeries サーバー上の DRDA ジョブの NLSS ソート・シーケンス・テーブルを使用することを指定できるか?
- 照会を実行するときに、戻される行がない理由は?
- DB2 UDB for iSeries V5R3 との対話式処理を行うためには、どのレベルの DB2 UDB for Linux/Unix/Windows (LUW) または DB2 Connect LUW が必要か?
- DB2 Connect バージョン 8 から iSeries V5R3 へ、どのようにスクロール可能カーソルのサポートを使用可能にできるか?

DB2 コネクトから接続しようとするとメッセージ SQL5048N が戻されるのはなぜか?

メッセージ SQL5048N の定義は、データベース・サーバーのリリース・レベルがデータベース・クライアントのリリース・レベルをサポートしていないことを示すものです。しかし、このメッセージは、時おり誤って戻されることがあります。この問題は、一般に次のようなことが原因で発生します。

1. Client Application Enabler だけがインストールされている状態のとき、このエラー・メッセージが表示されます。この場合は、ゲートウェイ・サーバーを通してクライアント・システムを iSeries サーバーに接続させる必要があります。直接接続はサポートされていません。
2. 接続の手動構成中にどこかでエラーが発生していた場合にも、このエラーが戻される可能性があります。

クライアント構成アシスタント (CCA) を使用することによって、 SQL5048N が戻されないようにすることができます。

この他にこの問題の原因として考えられるのは、コレクション NULLID に関係する問題です。 DB2 コネクト、 IBM DB2 Universal Driver for SQLJ and JDBC、および他のアプリケーション・リクエスターでは、必要な SQL パッケージを構築する際にコレクション NULLID を使用します。コレクションとパッケージは、最初の接続で作成されます。ユーザー・プロファイルにコレクションを作成する権限がない場合は、最初に接続するプロファイルを、それよりも高い権限を持つ別のプロファイルにして、これらのオブジェクトが作成できるようにしてください。

このエラーに別の原因があると思われる場合は、 Authorized Problem Analysis Report Web サイトを参照してください。「**Search**」フィールドに「**APAR II12722**」と入力してください。

iSeries ファイルはジャーナルする必要があるか?

この質問への回答は、『パフォーマンスを向上させるために、照会データはどのような場合にブロック化されるのか?』 の質問と密接に関係しています。クライアント・アプリケーションが非コミット (NC) または非コミット読み取り (UR) の分離レベルを使用していて、しかも DB2 UDB SQL 機能が、照会データはブロック化できると判別した場合は、ジャーナル処理は必要ありません。 その場合、コミットメント制御は使用可能ではなく、それによってジャーナル処理は不要になります。

分離レベルを変更する方法の例は、以下のとおりです。

- DB2 コネクトのプリコンパイラーが、ISOLATION UR パラメーターを使用して、非コミット読み取りを指定します。
- DB2 コネクトのコマンド行プロセッサー (CLP) が、コマンド DBM CHANGE SQLISL TO UR を使用して、非コミット読み取りを指定します。
- DB2 コネクトのコマンド行プロセッサー (CLP) が、コマンド DBM CHANGE SQLISL TO NC を使用して、非コミットを指定します。
- JDBC クライアントが、接続のプロパティー分離レベルを TRANSACTION_READ_UNCOMMITTED に設定して、非コミット読み取りを指定します。

パフォーマンスを向上させるために、照会データはどのような場合にブロック化されるのか?

照会データは、以下の条件のいずれも該当しない場合にブロック化されます。

- カーソルが更新可能である (注 1 参照)。
- カーソルが潜在的に更新可能である (注 2 参照)。
- SQLPREP または SQLBIND で BLOCKING NO 事前コンパイラーまたはバインド・オプションが使用されている。

BLOCKING NO の事前コンパイル/バインド・オプションを付けた単一行プロトコルを強制しないかぎり、以下の条件のどちらの場合も、ブロック化が行われます。

- カーソルが読み取り専用である (注 3 参照)。
- 以下のすべてが該当する。
  - SELECT の中に FOR UPDATE OF 文節がなく、しかも、
  - プログラム中にカーソルに対する UPDATE または DELETE WHERE CURRENT OF ステートメントがなく、しかも、
  - プログラムに動的 SQL ステートメントが入っていないか、または BLOCKING ALL が使用されている。

注:

1. カーソルは、それが読み取り専用ではなく (注 3 参照)、しかも以下のいずれかに該当する場合には更新可能です。
   - 選択ステートメントに、FOR UPDATE OF 文節が含まれている。または
   - プログラム中に、カーソルに対する UPDATE または DELETE WHERE CURRENT OF がある。
2. カーソルは、それが読み取り専用ではなく (注 3 参照)、しかもプログラムに動的ステートメントが含まれており、 SQLPREP または SQLBIND で BLOCKING UNAMBIG 事前コンパイルまたはバインド・オプションが使用されている場合は、潜在的に更新可能です。
3. カーソルは、以下の 1 つ以上の条件に該当する場合は読み取り専用です。

- DECLARE CURSOR ステートメントに ORDER BY 文節の指定があり、FOR UPDATE OF 文節の指定はない。
- DECLARE CURSOR ステートメントに FOR FETCH ONLY 文節の指定がある。
- 以下の条件のうち 1 つ以上が、カーソルについて、あるいはそのカーソルの参照先の外側の副選択内で参照されているビューまたは論理ファイルについて当てはまる。
  - 外側の副選択に、DISTINCT キーワード、GROUP BY 文節、HAVING 文節、または列関数が含まれている。
  - その選択に結合関数が含まれている。
  - その選択に UNION 演算子が含まれている。
  - その選択に最も外側の副選択のテーブルと同じテーブルを参照する副照会が含まれている。
  - その選択に一時ファイルにコピーしなければならない複合論理ファイルが含まれている。
  - 選択された列のすべてが、式、スカラー関数、または定数である。
  - 参照された論理ファイルの列のすべてが入力専用である。

DBM SQL0969N エラー・メッセージで報告される SQLCODE と関連のトークンの解釈方法は?

DB2 コネクトで使用されるクライアント・サポートは、同等のコードを持たないホスト SQLCODE およびトークンを報告する場合にメッセージ SQL0969N を戻します。以下はメッセージ SQL0969N の例です。

```
SQL0969N  There is no message text corresponding to SQL error
"-7008" in the Database Manager message file on this workstation.
The error was returned from module "QSQOPEN" with original
tokens "TABLE1    PRODLIB1    3".
```

コードおよびトークンを解釈するには、メッセージ記述表示 (DSPMSGD) コマンドを使用します。

```
DSPMSGD SQL7008 MSGF(QSQLMSG)
```

オプション 1 (メッセージ・テキストの表示) を選択すると、サーバーは「定様式メッセージ・テキストの表示」画面を表示します。 メッセージの中の 3 つのトークンは、画面では、&1、&2、および &3 によって表されます。この例のメッセージの理由コードは 3 で、これは、画面に示されているコード 3 を指し示します。

```
                          定様式メッセージ・テキストの表示
システム :   KC000
 メッセージ ID . . . . . :   SQL7008
 メッセージ・ファイル . . . . :   QSQLMSG
 ライブラリー . . . . . :     QSYS

 メッセージ . . :    操作には &2 の &1 が正しくない。
 原因--理由コードは &3 です。理由コードとその意味は次の通りです。
 1 -- &1 にメンバーがない。
 2 -- &1 は記憶域を解放して保管された。
 3 -- &1 がジャーナル処理されていないか，あるいはジャーナルに対する権限が
 ない。CASCADE，SET NULL，SET DEFAULT の RI 制約処置をもつファイルは，同じ
 ジャーナルにジャーナル処理しなければなりません。
 4 および 5 -- &1 は実動ライブラリーに入っていてそこで作成中であるが，ユー
 ザーはデバッグ・モード UPDPROD(*NO) を指定している。
 6 -- コレクションを作成中であるが，ユーザーは UPDPROD(*NO) でデバッグ・モ
 ードになっている。
 7 -- ビューの作成で使用された基になったテーブルが無効である。テーブルがプ
 ログラム記述テーブルであるか，または一時ライブラリーに入っています。
                                                                  続く ...
 続行するには，実行キーを押してください。

 F3= 終了    F11= 不定様式メッセージ・テキストの表示    F12= 取り消し
```

WHERE 文節のホスト変数のタイプはパフォーマンスにどのような影響を与えるのか?

iSeries server における性能低下の考えられる原因の 1 つは、クライアントが C プログラムで、 SELECT ステートメントの WHERE 文節の中の比較に浮動小数点変数を使用していることです。 OS/400 がその列のデータを変換する必要がある場合は、その列の索引は使用できません。比較の中で用いる列、リテラル、およびホスト変数には常に同じタイプを使用するようにする必要があります。データベース内の列がパックまたはゾーン 10 進数として定義されている場合に、ホスト変数が他のタイプになっていると、C で問題が発生する可能性があります。

詳細は、 iSeries Information Center の『データベース・パフォーマンス向上のためのプログラミング方法』を参照してください。

ライブラリー・リストを使用して、修飾されていないテーブルおよびビューの名前を解決できるか?

iSeries サーバーでは、 DB2 コネクト製品を使用するクライアント・プログラムなどの非 iSeries DRDA クライアント・プログラムから DB2 UDB for iSeries データにアクセスするときに、 OS/400 のシステム命名オプションを使用するための限定機能をサポートしています。それまでは、異種 DRDA クライアントから接続するときには SQL 命名オプションしか使用できませんでした。システム命名によって、DB2 UDB for iSeries の特性がいくつか変更されています。以下にその例を示します。

1. 修飾されていないテーブルおよびビューの名前の解決で、ライブラリー・リストが検索される。
2. CREATE SQL ステートメントを実行するときに、修飾されていないオブジェクトが現行ライブラリーに作成される。
3. 修飾されているオブジェクト名を、それらが入っているライブラリーまたはコレクションから分けるために、ピリオド (.) の代わりにスラッシュ (/) が使用される。
4. ある権限特性が変更されている。

詳細は、iSeries の SQL 解説書のサーバー命名に関する項を参照してください。権限に関する説明については、『分散リレーショナル・データベースの計画および設計』を参照してください。

| DB2 コネクトは、事前コンパイル (PREP) コマンドと (BIND) コマンドというその 2 つのプログラム準
| 備コマンドでの、総称バインド・オプションの指定をサポートしています。 OS/400 命名は、これらのコ
| マンドのいずれかで、 Windows バッチ・ファイルからの抜粋である以下の例のようにして指定できます。

| DB2 コネクト V8 以降の場合。

```
| DB2 PREP %1.SQC BINDFILE OS400NAMING SYSTEM ...
| DB2 BIND %1.BND OS400NAMING SYSTEM ...
```

| DB2 コネクト V8 より前の場合。

```
| DB2 PREP %1.SQC BINDFILE GENERIC 'OS400NAMING SYSTEM' ...
| DB2 BIND %1.BND GENERIC 'OS400NAMING SYSTEM' ...
```

| Windows 開発プラットフォームでは総称オプションの名/値の組みの前後に単一引用符 (アポストロフィ) を使用し、 AIX または UNIX プラットフォームでは二重引用符を使用しますので、注意してください。

注: iSeries V4R5 および V5R1 では、オプションの名前は AS400NAMING であり、OS400NAMING ではありません。

SYSTEM のほかに有効な OS400NAMING オプションの値は、デフォルト値の SQL だけで、このフィーチャーが導入される前は、これが非 iSeries クライアントから使用可能な唯一のオプションでした。

OS400NAMING オプションを (BIND) コマンドで使用し、(PREP) コマンドでは使用しない場合は、サーバー・プラットフォームで SQL エラーが検出されてもバインド・ファイルを作成するよう指示するパラメーターを (PREP) コマンド上にコーディングする必要があります。 DB2 コネクトの場合は、 SQLERROR CONTINUE パラメーターがこの目的で使用するパラメーターです。この機能は「限定」と呼ばれていますが、これは、特定の状況では、クライアント・サイドのソフトウェアがリモート・サーバーで実行することを意図した SQL ステートメントを構文解析することがあるためです。スキーマ ID をテーブル ID と区切るために、サーバーの命名で求められているように、ピリオドではなくスラッシュを使用すると、そのステートメントは、不適切な構文が含まれているとしてリジェクトされる可能性があります。

DB2 コネクトのユーザーは、通常の EBCDIC 順序の代わりに、 iSeries サーバー上の DRDA ジョブの NLSS ソート・シーケンス・テーブルを使用することを指定できるか?

iSeries は、汎用バインド・オプションを認識します。これにより、DB2 コネクトや汎用バインド・オプションをサポートする他のクライアントから実行するプログラムを作成する際、クライアントの要求が実行されるサーバー・ジョブに関連付けられた NLSS ソート・シーケンスを iSeries サーバーで使用するよう要求できるようになりました。この機能は、V5R1 の PTF SI00174 で可能になります。後のリリースでは、この機能は基本オペレーティング・システムに入っています。

この拡張機能を利用する場合は、新しいソート・シーケンス・オプションの必要な DB2 UDB for iSeries 上にあるすべての SQL パッケージを、総称バインド・オプション SORTSEQ をクライアント・システムの JOBRUN の値と共に使用することによって再作成する必要があります。

バインド・オプションにより、ユーザーは、通常の EBCDIC 順序の代わりに、 iSeries サーバー上の DRDA ジョブの NLSS ソート・シーケンス・テーブルを使用することを指定できます。以前は、異種 DRDA クライアントから接続するときには、 EBCDIC 順序を使用するデフォルトの *HEX オプションしか使用できませんでした。

このフィーチャーは、 DRDA 総称バインド機能をサポートする DRDA アプリケーション・リクエスターから利用できます。これには、 Windows 上で実行する DB2 コネクト 6.1 FixPak 1 をクライアント開発プラットフォームおよび実行環境として使用した限定検査を施行済みです。 DB2 コネクトは、事前コンパイル (PREP) コマンドと (BIND) コマンドというその 2 つのプログラム準備コマンドでの、総称バインド・オプションの指定をサポートしています。 JOBRUN ソート・シーケンスは、これらのコマンドのいずれかで、 Windows バッチ・ファイルからの抜粋である以下の例のようにして指定できます。

```
DB2 PREP %1.SQC BINDFILE SORTSEQ JOBRUN...
DB2 BIND %1.BND SORTSEQ JOBRUN...
```

注: Windows 開発プラットフォームでは、総称オプションの名/値の組みの前後に単一引用符 (アポストロフィ) が使用されますが、 AIX または UNIX プラットフォームでは二重引用符が使用されます。

SORTSEQ オプションの他の唯一の有効値は、デフォルト値の HEX で、このフィーチャーが導入される前は、これが非 iSeries クライアントから使用可能な唯一のオプションでした。

照会を実行するときに、戻される行がない理由は?

この問題の原因として考えられることの 1 つは、 DB2 コネクト・データベース通信サービス・ディレクトリーへの iSeries server に関する項目の追加が失敗したということです。

DB2 UDB for iSeries V5R3 との対話式処理を行うためには、どのレベルの DB2 UDB for Linux/Unix/Windows (LUW) または DB2 Connect LUW が必要か?

対話式処理には、以下のフィックスパックが必要です。

- DB2 UDB LUW バージョン 7 フィックスパック 10
- DB2 UDB LUW バージョン 8 フィックスパック 4
- DB2 Connect LUW バージョン 7 フィックスパック 10
- DB2 Connect バージョン 8 フィックスパック 4

これらの フィックスパック は、 DB2 Technical Support Web サイト から入手できます。

DB2 Connect バージョン 8 から iSeries V5R3 へ、どのようにスクロール可能カーソルのサポートを使用可能にできるか?

クライアント側でフィックスパック 4 以降を使用する必要があります。フィックスパック 4 を使用している場合は、以下のいずれかを行ってください。

- 次のコマンドを実行します。

  ```
  UPDATE CLI CFG FOR SECTION iSeries dbname USING CURSORTYPES 1
  ```

  iSeries dbname の部分には、ご使用の iSeries データベース名を入れてください。
- 次の構文を使用して、db2cli.ini ファイルを編集します。

  ```
  CURSORTYPES = 1
  ```

異種環境間での対話式処理に関するその他のヒント

以下のセクションでは、 DB2 コネクトおよび DB2 UDB とともに DB2 UDB for iSeries を使用するための追加情報が示されています。これらのヒントは、 OS/2 プラットフォーム上の製品で行われたテストの経験に基づくものですが、移植されたすべての環境でも当てはまるものと思われます。

DB2 コネクト対 DB2 UDB

アプリケーション・リクエスター (クライアント) 機能に対して、 DRDA アプリケーション・サーバー機能を実行するにはどのプロダクトが必要なのか、ユーザーは、時々混乱します。 AR は DB2 コネクトと呼び、 AS は DB2 Universal Database (UDB) と呼びます。 DB2 UDB の中には、以下が含まれます。

- DB2 UDB for AIX
- DB2 UDB for HP-UX
- DB2 UDB for Linux
- DB2 UDB for Sun Solaris
- DB2 UDB for Windows

適切な構成および保守レベル

プロダクト・マニュアルにあるインストールおよび構成の指示に必ず注意深く従ってください。プロダクトのレベルが最新のものであることを確認してください。まだ行っていない場合は、適切な fix パックを適用してください。

テーブル名およびコレクション名

DRDA アプリケーションがアクセスする SQL テーブルには 3 部からなる名前があります。最初の部分はデータベース名、2 番目はコレクション *ID*、そして 3 番目はベース・テーブル名です。最初の 2 つの部分はオプションです。 DB2 UDB for iSeries は、テーブル名を第 2 レベルで、コレクション名 (またはライブラリー名) によって修飾します。テーブルは、DB2 UDB for iSeries データベースに常駐します。

V5R2 より前、独立補助記憶域プール (IASP) が出現するまでは、各 iSeries server には 1 つのデータベースしかありませんでした。しかし DB2 UDB では、テーブルは、ユーザー ID (そのテーブルの作成者のユーザー ID) によって与えられ、プラットフォーム上の、複数のデータベースのいずれかに常駐する可能性があります。 DB2 コネクトにも、コレクション ID に対するユーザー ID の使用に関して同じことが言えます。

DB2 コネクトから DB2 UDB for iSeries に対して行われる動的照会の際、指定された照会先のテーブルにコレクション名がない場合、ターゲット側ジョブ (iSeries server 上) のユーザー ID が使用されます。 これは、ユーザーにとって予想外である場合があり、このために、テーブルが見つからない可能性もあります。

DB2 UDB for iSeries から DB2 UDB に対して行われる動的照会の際、 *qualifier.table-name* という形式の照会で指定されない場合、暗黙のテーブル修飾子を持つ場合があります。第 2 レベルの UDB テーブル修飾子は、デフォルトにより、照会を作成するユーザーのユーザー ID になります。

DB2 UDB データベースおよびテーブルを、共通ユーザー ID で作成したい場合があります。しかし、UDB では DB2 UDB for iSeries のような物理コレクションが存在しないことにご注意ください。 UDB には、テーブル修飾子しかなく、それは作成者のユーザー ID です。

特権の授与

UDB データベースにアクセスする iSeries server 上で作成されたすべてのプログラムで、忘れずに以下の UDB コマンドを実行してください (コマンド行プロセッサーから入力できます)。

1. GRANT ALL PRIVILEGES ON TABLE テーブル名 TO ユーザー (ユーザーは 'PUBLIC' でもよい)
2. GRANT EXECUTE ON PACKAGE パッケージ名 (通常は iSeries プログラム名) TO ユーザー (ユーザーは 'PUBLIC' でもよい)

APPC 通信のセットアップ

AR として DB2 コネクト、または AS として UDB のどちらかを APPC とともに使用する際には、ワークステーション用に作成されたコントローラーおよび装置とともに、 OS/400 通信を適切に構成しなければなりません。

RDB ディレクトリーのセットアップ

iSeries サーバーと接続する各 UDB データベースの RDB ディレクトリーに項目を追加してください。RDB ディレクトリー項目の追加 (ADDRDBDIRE) コマンドを使用してください。 RDB 名は UDB データベース名になります。

APPC 通信を使用する場合、リモート・ロケーション名はワークステーション名になります。

TCP/IP を使用する場合は、リモート・ロケーション名はワークステーションのドメイン名、またはその IP アドレスになります。 UDB DRDA サーバーが使用するポートは、通常、 iSeries server が使用する DRDA ポート (*DDM)、446 ではありません。

UDB プロダクトの資料を調べて、ポート番号を判別してください。共通値として 50000 が使用されています。以下は、 RDB 項目が UDB サーバーに対して適切に構成されたことを示す DSPRDBDIRE 画面の例です。

```
リレーショナル・データベース明細の表示
    リレーショナル・データベース . . . :   SAMPLE
    リモート・ロケーション :
      リモート・ロケーション . . . . . :   9.5.36.17
        タイプ . . . . . . . . . . . . . :   *IP
        ポート番号またはサービス名 . . :   50000
      テキスト . . . . . . . . . . . . . :   My UDB server
```

DB2 コネクト用の SQL パッケージのセットアップ

DB2 コネクトを使用して DB2 UDB for iSeries 上のデータにアクセスする前に、アプリケーション・プログラム用、および DB2 コネクト・ユーティリティー用に、 iSeries server 上に SQL パッケージを作成しなければなりません。

DB2 (PREP) コマンドを使用すれば、組み込み SQL を持つアプリケーション・プログラムのソース・ファイルを処理することができます。 この処理により、 SQL ステートメントのホスト言語呼び出しを含む変更ソース・ファイルが作成されます。そしてデフォルトで、現在の接続先データベースに SQL パッケージを作成します。

DB2 コネクトを DB2 UDB for iSeries サーバーにバインドするには、以下のようにします。

1. CONNECT TO rdbname
2. Bind path@ddcs400.lst BLOCKING ALL SQLERROR CONTINUE MESSAGES DDCS400.MGS GRANT PUBLIC

   上記の path@ddcs400.lst パラメーターの 'path' を、デフォルトのパス C:¥SQLLIB¥BND¥ (非 INTEL のプラットフォームの場合は c:/sqllib/bin/) に置き換えてください。また、インストール先がデフォルトのディレクトリーでなかった場合は、インストール時に指定した値に置き換えてください。
3. CONNECT RESET

対話式 SQL の DB2 UDB での使用

対話式 SQL を使用するためには、 DB2 UDB Query Manager and SQL Development Kit プロダクトを OS/400 にインストールしておく必要があります。 UDB 上のデータへのアクセスは、以下のように行います。

1. STRSQL でセッションを開始する際、 NAMING(*SQL)、DATFMT(*ISO)、および TIMFMT(*ISO) のセッション属性を使用します。 *ISO 以外の形式 (すべてではない)、および日付形式 (DATFMT) で使用される形式を、時刻形式 (TIMFMT) でも使用しなければなりません。
2. iSeries server での COLLECTION と、 UDB でのテーブル修飾子 (作成者のユーザー ID) との対応関係に留意してください。
3. 最初の対話式セッションでは、UDB で作成されるパッケージを入手するために、以下のような順序で SQL ステートメントを実行しなければなりません。 (1) RELEASE ALL、(2) COMMIT、(3) CONNECT TO rdbname ('rdbname' は、特定のデータベースに置き換えられます)。

対話式 SQL の使用のセットアップの一環として、 ″GRANT EXECUTE ON PACKAGE QSQL400.QSQLabcd TO PUBLIC (または特定のユーザー)″ ステートメントを実行し、他のユーザーも対話式 SQL 用の PC で作成された SQL PKG を使用できるようにしてもよいでしょう。上記の GRANT ステートメントの 'abcd' に入る実際の値は、 250 ページの『異種環境アプリケーション・サーバー上での対話式 SQL および Query 管理機能のセットアップ』で示されているテーブルから判別することができま

す。そこには、パッケージが作成されたときの、様々なオプション・セットの実際のパッケージ名が示されています。たとえば、パッケージが作成されたときに以下のオプションが使用されている場合は、″GRANT EXECUTE ON PACKAGE QSQL400.QSQL0200 TO some-user″ ステートメントを、ある特定のユーザーに対して実行することもできます。 *ISO 日付、*ISO 時刻、*CS コミットメント制御、アポストロフィー・ストリング区切り文字、1 バイト文字サブタイプ。

Query の終了

DB2 コネクトには、暗黙的、明示的を問わず Query が終了されたときに読み取りロックの解放を要求するオプションが追加されました。なお、サーバーがこの要求を尊重しない場合、それはエラーとはみなされません (iSeries サーバーの場合)。 DB2 コネクトには、読み取る行がなくなったとき、サーバーで非スクロール可能カーソルに対して暗黙的に Query を終了させるかどうかを指定する、別の新しいオプションがあります。これは、サーバーによって前もって決定されます。 iSeries AS では、V5R3 でこの新しい機能がサポートされています。

ユーザー ID とパスワードの長さ

アプリケーション・リクエスター (AR) として稼働する DB2 UDB for iSeries では、異種のアプリケーション・サーバー (AS) に対して実行されるとき、 10 文字以上のユーザ ID とパスワードを使用することができます。正確な文字数の制限は、使用されている特定のインターフェースの記述で指定されます。例として、SQL CONNECT ステートメントの場合の制限については、『SQL 解説書』のトピックを参照してください。

付録 C. ジョブのトレースと FFDC データの解釈

この付録では、追加の問題分析情報を記載しています。これは、問題判別を担当するスペシャリストに役立ちます。またこの付録は、 Distributed Relational Database Architecture に準拠して設計されたソフトウェア製品を提供し、 iSeries server との接続性をテストする必要のある方にも参考になります。

この付録には、ジョブ・トレースから得られる RW 構成要素トレース・データの例と、トレース・データ出力の説明も記載されています。この情報は、通信トレース・データを解釈するのに役立ちます。またこの付録では、記憶域の第 1 障害データ検知の印刷出力の例が示され、その出力の説明が記載されています。

ジョブのトレースと FFDC データの例については、以下のトピックを参照してください。

- ジョブのトレースの RW 構成要素のデータ項目の解釈
- 第 1 障害データ検知 (FFDC)

ジョブのトレースの RW 構成要素のデータ項目の解釈

DRDA サポートの大半を含んでいるのは、OS/400 ライセンス・プログラムの RW 構成要素です。TRCTYPE(*ALL) または TRCTYPE(*DATA) を指定されたジョブ・トレース (TRCJOB) コマンドが発行されると、この構成要素は、特定のタイプの診断情報を作成します。 RW トレース・ポイントは、図 28 に示されているタイプのものです。検索引き数としてストリング '>>' を使って検索すれば、 RW トレース・ポイントを簡単に見つけることができます。詳細は、トピック『RW トレース・ポイントの説明』を参照してください。 '<<<...' 区切り文字を見つけ出せば、各トレース・ポイントでダンプされたデータの末尾を判別することができます。データの末尾には、最終行を埋めるのに十分な数の 1 つ以上の '<' 区切り文字があります。

```
データ   FF  6E6ED9E6D8E840D9C37A0016D052000100102205000611490000  *>>RWOQ RC:  }             *
データ   FF  0006210224170025D0530001001F241A0C76D00500023100030A  *        }        }        *
データ   FF  00080971E0540001D000010671F0E00000002CD0530001002624  *    ¥   }   0¥    }       *
データ   FF  1BFF0000000100F1F1F1411000000000000000FF0000000200F2F2  *       111             22 *
データ   FF  F241200000000000000026D0520001002022 0B00061149000400  *2         }             *
データ   FF  162110C4C2F2C5E2E8E2404040404040404040404040400056D00300  *   DB2ESYS            }   *
データ   FF  0100502408000000064F0F2F0F0F0C4E2D5E7D9C6D54000C4C2  *  &      02000DSNXRFN  DB *
データ   FF  F2C5E2E8E2404040404040404040404040FFFFFF92000000000000  *2ESYS             k       *
データ   FF  0000FFFFFFFF0000000000000000404040404040404040404000  *                           *
データ   FF  0000004C4C4C4C4C4C4C4C4C4C4C4C4C4C4C4C4C4C4C4C4C4C4C  *   <<<<<<<<<<<<<<<<<<<<<<< *
```

図 28. ジョブ・トレース RW 構成要素情報の例

データ・ストリームを構成するエレメントについては、トピック『例: RW トレース・データの分析』を参照してください。

注: データの終わりの判別のための '<' 区切り文字の使用法には、例外が 1 つあります。受信データ・ストリームがダンプされる特殊な状況の下では、トレース・データを書き込むモジュールは、データ・ストリームの末尾がどこかを判断できません。そのような場合、このプログラムは、受信バッファー全体をダンプし、ダンプされるデータはデータ・ストリームよりも長いという警告として、 '<<<...' 区切り文字を '(' 文字のストリングに置き換えます。

接頭部 '>>' の後には、トレース・ポイントを識別する 7 文字のストリングが続きます。最初の 2 文字 'RW' は、構成要素を識別します。 2 番目の 2 文字は、実行される RW 機能を識別します。 'QY' は、

DDM コマンド OPNQRY、CNTQRY、および CLSQRY に対応する照会機能を示します。 'EX' は、DDM コマンド EXCSQLSTT、EXCSQLIMM、および PRPSQLSTT に対応する実行 (EXECUTE) 機能を示します。

これらの機能のおのおのにどのプログラム・モジュールが対応するかは、分散 SQL アクセス操作の**アプリケーション・リクエスター (AR)** と**アプリケーション・サーバー (AS)** のどちら側でジョブ・トレースがとられたかによって異なります。 AR で処理と照会の機能を実行するモジュールは、QRWSEXEC と QRWSQRY です。 AS 側のモジュールは QRWTEXEC と QRWTQRY です。

7 バイトのトレース・ポイント ID の最後の 2 文字は、ダンプ・データの特性を示すか、またはダンプが取られた地点を示します。たとえば、SN は AR または AS から送られたデータ・ストリームに対応し、RC は AR で受信されたデータ・ストリームに対応します。

例: RW トレース・データの分析

261 ページの図 28 の例は、分散 SQL 照会機能において受信されたデータ・ストリームを示しています。このトレースは、接続の**アプリケーション・リクエスター (AR)** 側で実行されたものです。したがって、それに関連した、データを作成したプログラム・モジュールは QRWSQRY です。

以下の説明では、この例のデータ・ストリームを構成する要素が考察されます。 DRDA データ・ストリームの解釈についての詳細は、「*Distributed Relational Database Architecture Reference*」および「*Distributed Data Management Level 4.0 Architecture Reference*」を参照してください。この資料は、Web 上の http://www.opengroup.org/dbiop/index.htm でご覧になれます。

トレース・データは、トレース・ポイント ID の末尾を示す ':' の後に続きます。ここの例では、データ・ストリームの最初の 6 バイトに、 DDM データ・ストリーム構造 (DSS) ヘッダーが入っています。この DSS ヘッダーの最初の 2 バイトは、長さフィールドです。 3 番目のバイト X'D0' は、すべての DDM データの登録済み SNA アーキテクチャー ID です。 4 番目のバイトは、フォーマット ID です (この後に詳述されています)。 5 番目と 6 番目のバイトには、DDM 要求相関 ID が入っています。

次の 2 バイト、X'0010' (10 進数 16) は、次の DDM オブジェクトの長さを示します。この場合、このオブジェクトは、後に続く X'2205' によって識別され、 OPNQRYRM 応答メッセージのコード・ポイントとなります。

16 バイトの応答メッセージの後には、応答メッセージの後に続く応答オブジェクト用の 6 バイトの DSS ヘッダーが続きます。最初の応答オブジェクトは、X'241A' コード・ポイントで識別されます。それは QRYDSC オブジェクトです。例の中の 2 番目の応答オブジェクトは QRYDTA 構造であり、これは、X'241B' コード・ポイントで識別されます (トレース出力内では 2 行にまたがります)。 OPNQRYRM コード・ポイントの場合と同様、先頭の 2 バイトがオブジェクトの長さを示しています。

| QRYDTA オブジェクトをさらに詳しく調べると、 X'241B' コード・ポイントの後に X'FF' が見つかります。これはヌルの SQLCAGRP (ワイヤー上を流れる SQLCA または SQL 診断域の形式) です。ヌル形式の SQLCAGRP は、関連データに関するエラーや警告の情報が入っていないことを示します。その場合、関連データは、SQL SELECT 操作からのデータ行になります。それは、ヌル SQLCAGRP の後に続きます。ただし、SQLCAGRP と同じくデータ行にもヌルを使えるので、ヌル SQLCAGRP の後に続く最初のバイトは、データ行はヌルではないことを示す X'00' の入った標識になります。ヌル標識バイトの意味は、最初のビットで決まります。この桁に '1' があると、'ヌル' を示します。しかし通常は、標識がヌル・オブジェクトを表すときは、8 ビットすべてがオンに設定されます。

データ行の形式は、前に置かれた QRYDSC オブジェクトで指示されます。この例では、QRYDSC は、行にはヌル可能な SMALLINT 値、ヌル可能な CHAR(3)、およびヌル可能でない倍精度の浮動小数点値が入

っていることを示します。ヌル SQLCAGRP の後の 2 番目のバイトは、SMALLINT フィールドに関連したヌル標識です。これは、フィールドがヌルではないことと、その後の X'0001' はフィールド・データであることを示します。その後にはヌル可能な CHAR(3) があり、それには '111' が入っています。さらにその後に続く浮動小数点値の後には X'00' バイトは付きません。これは、ヌルにできないと定義されているからです。

最初のデータ行の後には、ヌル SQLCAGRP をもつ 2 番目のデータ行が続き、さらにその後には別の 6 バイトの DSS ヘッダーが続きます。そのヘッダーに入っている形式バイトの後半 (X'2') は、それに対応する DSS が REPLY であることを示します。前の DSS の形式バイト (X'53') は、それが OBJECT DSS であることを示していました。 3 番目の DSS に ENDQRYRM 応答メッセージが入っていると、 REPLY DSS 内にそのメッセージを入れる必要があります。 ENDQRYRM コード・ポイントは X'220B' です。この応答メッセージには、X'0004' という重大度コードと、照会データを戻した RDB の名前 ('DB2ESYS') が入っています。

この例では、3 番目の DSS の後に 4 番目の最後の DSS が続きます。これの形式バイトは X'03' です。3 は、これが OBJECT DSS であることを示し、その前にある 0 は、この連鎖の最後の DSS であることを示します (連鎖ビットはオフになっています)。

この DSS 内のオブジェクトは、非ヌルの SQLCAGRP の入った SQLCARD です。 X'2408' SQLCARD コード・ポイントの後に続く最初のバイトは、 SQLCAGRP がヌルではないことを知らせる標識です。次の 4 バイト (X'00000064') は、+100 SQLCODE を表しますが、それは、'row not found' (行が見つからない) 条件が出されて照会が終了したことを意味します。残りのフィールドは、SQLCA 内の他のフィールドに対応します。残りのフィールドは、SQLCA または SQL 診断域内の他のフィールドに対応します。 SQLCA および SQL 診断域のフィールドに対する SQLCAGRP フィールドのマッピングは、「*Distributed Relational Database Architecture Reference*」を参照してください。この資料は、Web 上の http://www.opengroup.org/dbiop/index.htm で利用できます。

RW トレース・ポイントの説明

RWff RC - 受信データ・ストリームのトレース・ポイント

このデータ・ストリームには、**アプリケーション・サーバー (AS)** プログラムからの DDM 応答が入っています。 DSS ヘッダーは、このデータ・ストリーム内にあります。これは、 261 ページの図 28 に示されているトレース・ポイントです。

実行されている DRDA 機能の (ff) ID を以下に示します。

ff	DRDA 機能
AC	RDB にアクセスする。
OQ	照会をオープンする。
CQ	照会を続行する。
EQ	照会をクローズする。
PS	SQL ステートメントを準備する。
XS	SQL ステートメントを実行する。
XI	直ちに SQL ステートメントを実行する。
DT	Table ステートメントを記述する。
DS	Statement ステートメントを記述する。

SY TCP/IP の同期点 (**SY**nc point) 操作を実行する。

RWff SN - 送信データ・ストリームのトレース・ポイント

このデータ・ストリームには、**アプリケーション・リクエスター (AR)** プログラムからの DDM 要求、または**アプリケーション・サーバー (AS)** プログラムからの DDM 応答が入っています。その要求または応答は、ヘッダーの追加とワイヤー経由の伝送のために下位レベルの CN 構成要素に渡される前の状態になっています。受信データ・ストリームと送信データ・ストリームにおけるトレース情報の主な相違は、その内容が異なる他に、後者の場合は 6 バイトの DSS ヘッダー情報がないことです。送信データ・ストリームのトレース域内の最初の DSS では、ヘッダー全体が省略され、その後の DSS では、6 バイトのゼロが存在しますが、これらは、後で CN 構成要素モジュールで構成されるときに、上からヘッダーをオーバーレイされます。

実行される DRDA 機能の ID は、263 ページの『RWff RC - 受信データ・ストリームのトレース・ポイント』に示されているものと同じです。

RWQY S1 - 部分送信データ・ストリームのトレース・ポイント 1

このトレース・ポイントは、同種環境内で QRYDTA の作成に新しい照会ブロックが必要になったときに、 QRWTQRY モジュールの NEWBLOCK ルーチン内において発生します。同種環境では、照会ブロックは、送信される前に満たされている必要があるわけではなく、このポイントで常にワイヤー上に置かれているので、バッファー・スペースを再利用することができます。他の送信データ・ストリームの場合と同じように DSS ヘッダーはありません。

RWQY S2 - 部分送信データ・ストリームのトレース・ポイント 2

このトレース・ポイントは、異種環境内で QRYDTA の作成に新しい照会ブロックが必要になったときに、 QRWTQRY モジュールの NEWBLOCK ルーチン内において発生します。異種環境では、最後のものを除くすべての照会ブロックが満たされないと、新規のブロックの作成を開始できません。またこれらのブロックは、すべてのブロックが作成されないと送信されません。

RWQY BP - 正しく行われた取り出しのトレース・ポイント

このトレース・ポイントは、SQFCHCRS マクロの呼び出しで、 BPCA 構造を指す非ヌルのポインターが戻された (BPCA バッファーに 1 つ以上のレコードが戻されたという意味) ときに、 QRWTQRY モジュールの FETCH ルーチンにおいて発生します。ダンプ・データは BPCA 構造 (関連バッファーではない) であり、これは特に、いくつのレコードが戻されたかを示します。

RWQY NB - 失敗した取り出しのトレース・ポイント

このトレース・ポイントは、SQFCHCRS マクロの呼び出しで、 BPCA 構造を指すヌルのポインターが戻された (BPCA バッファーに何もレコードが戻されなかったという意味) ときに、 QRWTQRY モジュールの FETCH ルーチンにおいて発生します。ダンプされるデータは SQLSTATE です。

RWQY P0 - 結果セット Pseudo-Open

関連する情報は、パッケージ・リスト項目です。

RWQY AR - 処理済み配列結果セット

関連する情報は、配列結果セットの制御ブロックです。

RWQY DA - 配列結果セット SQLDA

関連する情報は、配列結果セット SQLDA です。

RWQY DO - デバッグ・オプション

関連する情報は、QRWOPTIONS ストリングの変更バージョンです。

RWQY L1 および RWEX L1 - アウトバウンド LOB テーブルへの保管のトレース・ポイント

このトレース・ポイントは、後でアプリケーション・リクエスターに送信するために QRWTQRY または QRWTEXEC に保管されたラージ・オブジェクト (LOB) 列に関するアドレスおよびその他の情報を記録します。

RWQY L2 および RWEX L2 - LOB テーブルからのデータ・ストリームへの組み込みのトレース・ポイント

このトレース・ポイントは、QRWTQRY または QRWTEXEC によって通信バッファーにコピーされたラージ・オブジェクト (LOB) 列に関するアドレスおよびその他の情報を記録します。

RWQY L0 および RWEX L0 - インバウンド LOB テーブルへの保管のトレース・ポイント

このトレース・ポイントは、データベース管理システム (DBMS) への入力用の SQL 記述子域 (SQLDA) を後で作成するために QRWTQRY または QRWTEXEC によって保管されたラージ・オブジェクト (LOB) 列に関するアドレス、およびその他の情報を記録します。

RWAC RQ - RDB へのアクセス要求のトレース・ポイント

このトレース・ポイントは、DRDA **アプリケーション・リクエスター (AR)** では QRWSARDB モジュール、**アプリケーション・サーバー (AS)** では QRWTARDB モジュールに入った時点において発生します。どちらであるかに応じて、内容は異なります。 AS でトレースがとられる場合、データの内容は、QRWTARDB によって実行される予定の DDM コマンドを識別する 2 バイトの DDM コード・ポイントであり、その後にそのコマンドの英字名が続きます。この名前は、切断の場合は SXXDSCT に、終結処理の場合は SXXCLNUP に、接続の場合は ACCRDB になります。 AR でトレースがとられる場合、データの内容は次のとおりです。

注: xv ページの『コードの特記事項情報』に重要なリーガル情報がありますので、お読みください。

```
   OFFSET TYPE    CONTENT
       --  ------- ------------------------------------------------
        0  BIN(8)  FUNCTION CODE
        1  CHAR(8) INTERPRETATION OF FUNCTION CODE
        9  BIT(8)  BIT FLAGS
       10  CHAR(1) COMMIT SCOPE
       11  CHAR(1) SQLHOLD value
       12  CHAR(1) CMTFAIL value
       13  BIN(15) Index of last AFT entry processed by RWRDBCMT

 The function codes are:
       0  'CONNECT ' ==>  CONNECT
       1  'DISCONNE' ==>  DISCONNECT
       2  'CLEANUP ' ==>  CLEANUP
       3  'RELEASE ' ==>  RELEASE
       4  'EXIT    ' ==>  EXIT
       5  'PRECMT  ' ==>  PRE-COMMIT
       6  'POSTCMT ' ==>  POST-COMMIT
       7  'PREROLLB' ==>  PRE-ROLLBACK
       8  'POSTROLL' ==>  POST-ROLLBACK
       9  'FORCED D' ==>  FORCED DISCONNECT
```

RWAC cb - RDB 制御ブロックのトレース・ポイント

次に示すトレース・ポイントは、 QRWSARDB モジュールが提供する機能に関連付けられた制御ブロックを識別します。

cb **制御ブロックの名前**

LV ローカル変数

DD コミット定義ディレクトリー

CD コミット定義制御ブロック

RI TSSCNAFT「リモート情報」構造

CB アクセス RDB 制御ブロック

DE RDB ディレクトリー項目

TE 活動ファイル・テーブル項目

RWSY FN: SYNCxxx [TYPE:x] -- ソース TCP SYNC/RESYNC トレース・ポイント

このソース側のトレース・ポイントでは、 TCP/IP 2 フェーズ・コミット操作の実行時にやり取りされる様々なコマンドや応答が記録されます。上で 'xxx' によって表されているデータのセグメントには、以下が入ります。

- CTL - 制御コマンドを表す
- RSY - 再同期コマンドを表す
- CRD - 制御コマンドからの応答データを表す
- RRD - 再同期コマンドからの応答データを表す

CTL や RSY のレコードには、コマンドに関連付けられている TYPE コードもあります。これは、表示可能な文字ではないため、レコードの 16 進データ部分でのみ確認できます。このコードは、ストリング 'TYPE:' の後に入ります。

RWSY xx: yyyyyyy... -- ターゲット TCP SYNC/RESYNC トレース・ポイント

このターゲット側のトレース・ポイントには、様々な情報が記録されます。情報のタイプは、上の xx で表される 2 つの文字によって識別されます。詳細は、yyyyyyy 部分の可変長ストリングに示されます。

- タイプ RC は、受け取ったコマンド (SYNCCTL または SYNCRSY) を記録します。
- タイプ RW は、パラメーター構造 WrwSYData を記録します。
- タイプ LG は、受け取った同期ログ (複数のオカレンスになる場合があります) を記録します。
- タイプ SN は、エラーが何も発生しなかった場合の送信バッファーを記録します。
- タイプ GE は、一般の例外が発生した場合のローカル変数を記録します。
- タイプ TE は、 TN 構成要素への要求が失敗した場合の送信バッファーとローカル変数を記録します (2 つのレコードのオカレンス)。
- タイプ CP は、会話のプロトコル・エラーが検出された場合の送信バッファーとローカル変数を記録します (2 つのレコードのオカレンス)。

RW_ff_m - アプリケーション・リクエスター・ドライバー (ARD) 制御ブロックのトレース・ポイント

このトレース・ポイントは、実行できるさまざまなタイプの ARD 呼び出しでの ARD 制御ブロックの内容を表示します。入力形式、出力形式、および SQLCA の 3 つのタイプの制御ブロックが表示されます。表示される呼び出しのタイプと制御ブロックのタイプは、トレース・ポイント ID にエンコードされま

す。その ID の形式は RW_ff_m になります。 ff は、呼び出しタイプ ID であり、m は制御ブロック・タイプのコードです。呼び出しタイプ ID (ff) と制御ブロック・タイプ・コード (m) は次のとおりです。

注: xv ページの『コードの特記事項情報』に重要なリーガル情報がありますので、お読みください。

```
ff   Call Type                                m  Ctl Blk Type
--   ----------------------                   -  ------------
CN   Connect                                  I  Input Format
DI   Disconnect                               O  Output Format
BB   Begin bind                               C  SQLCA
BS   Bind Statement
EB   End bind
PS   Prepare Statement
PD   Prepare and Describe Statement
XD   Execute Bound Statement with Data
XB   Execute Bound Statement without Data
XP   Execute Prepared Statement
XI   Execute Immediate
OC   Open Cursor
FC   Fetch from Cursor
CC   Close Cursor
DS   Describe a Statement
DT   Describe an Object
```

第 1 障害データ検知 (FFDC)

iSeries server には、分散リレーショナル・データベースに関するエラー情報を取り込んで報告するための手段が備えられています。その機能を**第 1 障害データ検知 (FFDC)** と呼びます。 FFDC サポートの主要な目的は、 OS/400 システムの DDM 構成要素内で検出されたエラーに関する包括的情報を提供して、プログラム診断依頼書 (APAR) を作成できるようにすることです。

またこの機能を利用して、システム関連のアプリケーション問題を診断することもできます。この機能を使用すれば、主な構造と DDM データ・ストリームは自動的にスプール・ファイルにダンプされます。このように、エラーの最初の発生時にそのエラーの情報を自動的にダンプすることには、サービス・サポートにそのエラーを報告するときに、同じ障害を再現する必要をなくすという目的があります。 FFDC は、アプリケーション・リクエスターとアプリケーション・サーバーの両方で活動状態になります。

注意する必要のある点は、負の数の SQLCODE の場合に常にダンプが行われるとは限らないことです。APAR 状態を示すものだけがダンプされます。

第 1 障害データ検知 (FFDC) とダンプについての詳細は、以下のトピックを参照してください。

- FFDC ダンプ
- FFDC ダンプ出力の説明
- DDM エラー・コード
- 第 1 障害データ検知 (FFDC) データの検索

FFDC ダンプ

| システムが検出した内部障害は、FFDC データのダンプをトリガーします。 QSFWERRLOG システム値を
| *NOLOG に設定すれば、FFDC 出力を使用不可にできますが、 FFDC ダンプ処理を使用不可にしないこと
| を強くお勧めします。 FFDC ダンプが行われると、通知メッセージ `*Software problem detected in Qxxxxxxx` (Qxxxxxxx は OS/400 モジュール ID) が、QSYSOPR メッセージ待ち行列に記録されます。

FFDC ダンプ操作による出力を見るには、 スプール・ファイルの処理 (WRKSPLF) コマンドを使って QPSRVDMP を表示します。 このダンプ出力に入っている情報は次のとおりです。

- DDM 機能
- 障害のある DDM モジュールについての特定情報
- DDM ソースまたはターゲットの主制御ブロック
- DDM 内部制御構造
- DDM 通信制御ブロック
- アプリケーション・リクエスターにおける場合、障害のある DDM モジュールの入力および出力パラメーター・リスト
- 要求および応答データ・ストリーム

データの最初の 1K バイトは、エラー・ログに入れられます。しかし、スプール・ファイルに入れられたデータのほうが、常により完全であってしかも簡単に処理できます。複数の DDM 会話を確立していた場合、ダンプ出力は、複数のスプール・ファイルに入っていることがあります。スプール・ファイルごとに項目は 32 個しか入らないという限度があるからです。その場合、前にアスタリスク (*) の付いた複数の「ソフトウェア問題」メッセージが QSYSOPR メッセージ待ち行列内に入れられています。

注: xv ページの『コードの特記事項情報』に重要なリーガル情報がありますので、お読みください。

```
 A                                                    B
5738SS1 V2R1M1                AS/400 ダンプ             090454/SRR/SRRS1        02/27/91 15:12:52        ページ  1
検出されたエラーのダンプが取られました
                                    C
.疑いのあるもの -                 QRWSQRY                 ライブラリー-     S
..ライセンス・プログラム -        5738SS1    V2R1M1
..機能 -                          5001
..ロード -                        0000
..PTF-
                                    D
.検出機能 -                       QRWSQRY                 ライブラリー-     S
..ライセンス・プログラム -        5738SS1    V2R1M1
..機能 -                          5001
..ロード -                        0000
..PTF-
.徴候ストリング -
       E     F       G
5738 MSGCPF3E86 F/QRWSQRY RC10000002
                                    H
.空間 -                           01                                                    I
000000   F0F17EC9 D5C4E740 F0F27EC6 C3E34E40   F0F37EC5 D4E2C740 F0F47ED7 D9D4E240  *01=INDX 02=FCT+ 03=EMSG 04=PRMS *
000020   F0F57EE2 D5C4C240 F0F67ED9 C3E5C240   F0F77EC1 D9C4C240 F0F87ED8 C4E3C140  *05=SNDB 06=RCVB 07=ARDB 08=QDTA *
000040   F0F97EC9 D5C4C140 F1F07EE2 D8C3C140   F1F17EE6 D9C3C140 F1F27ED9 C6D4E340  *09=INDA 10=SQCA 11=WRCA 12=RFMT *
000060   F1F37EC1 C6E34040 F1F47EE2 D4C3C240   F1F57EE3 E2D3D240 F1F67EE5 C1D9E240  *13=AFT  14=SMCB 15=TSLK 16=VARS *
000080   4DD9C5E2 E340C9E2 40C3C3C2 6BD7C3C2   E26BE2C1 E36BD7D4 C1D76BD9 C3E5C240  *(REST IS CCB,PCBS,SAT,PMAP,RCVB *
0000A0   D7C5D940 C3C3C25D                                                         *PER CCB)                        *
.空間 -                    L      02
000000   200C1254 0102F5F8 F0F0F9                                                  *      58009                     *
.空間 -                           04                                                    J
000000   D8D7C1D9 D4E20000 D67FC01D A60065A0   00000000 F0F10000 00000434 00000000  *QPARMS  O"          01          *
000020   D9C3C8C1 E2F2F6F6 40404040 40404040   4040E2D9 D9404040 40404040 40404040  *RCHAS266           SRR           *
000040   40404040 D7E3F140 40404040 40404040   40404040 4040700F 70DB33C0 00BB0005  *    PT1                          *
.空間 -                           05
000000   00000000 0056D051 00010050 200C0044   2113D9C3 C8C1E2F2 F6F64040 40404040  *           &      RCHAS266      *
000020   40404040 E2D9D940 40404040 40404040   40404040 4040D7E3 F1404040 40404040  *    SRR               PT1       *
000040   40404040 40404040 700F70DB 33C000BB   00050008 21140000 7FFF0021 D0030001  *                                *
000060   001B2412 00100010 0676D004 00000671   E4D00001 0007147A 000002            *                                *
.空間 -                           06
000000   0016D052 00010010 22050006 11490000   00062102 24170052 D0530001 0022241A  *                                *
000020   0F76D004 00002600 03020000 0A000009   71E05400 01D00001 0671F0E0 0000002A  *                                *
000040   241BFF00 0001F0F0 F1000000 013FF000   00000000 00FF0000 02F0F0F2 00000002  *      001     0          002    *
000060   40000000 00000000 0010D052 0001000A   220B0006 11490004 0069D003 00010063  *                                *
0000E0   FF                                                                        *                                *
.空間 -                    K      07
000000   D9C3C8C1 E2F2F6F6 40404040 40404040   4040D9C3 C8C1E2F2 F6F64040 40404040  *RCHAS266           RCHAS266      *
000020   40404040 E2D9D940 40404040 40404040   40404040 4040D7E3 F1404040 40404040  *    SRR               PT1       *
000040   40404040 40404040 700F70DB 33C000BB   D8E3C4E2 D8D3F4F0 F0D8E2D8 F0F2F0F1  *                QTDSQL400QSQ0201*
000060   F1002500 00000000 25000000 000010F0   F4F5F1F7 F461E2D9 D961C4E2 F3F7F840  *1              045174/SRR/DS378 *
000080   40404040 40404040 40404040 40404040   40404040 40404040 40404040 40404040  *                                *
0000A0 から 00015F の行は上と同じ
000160   40404040 40404040 40404040 4040A000   2434E2D9 D9404040 40404040 00000000  *                  SRR           *
000180   C1D7D7D5 4BD9C3C8 C1E2F3F7 F8A7CCA7   54137200 40404000 00000000 00000000  *APPN.RCHAS378x x            *
0001A0   00000000 00000000                                                         *                                *
```

```
.空間 -                        09
000000   E2D8D3C4 C1404040 00000060 00010001   01F40002 00000400 00000040 40404040  *SQLDA            4             *
000020   80000000 00000000 007FC01E 11000334   00000000 00000000 00000000 00000000  *                               *
000040   00080000 00250000 00000000 00000000   00000000 00000000 00000000 00000000  *                               *
.空間 -                        10
000000   E2D8D3C3 C1404040 00000088 FFFF8ABC   00041254 01020000 00000000 00000000  *SQLCA                          *
000020   00000000 00000000 00000000 00000000   00000000 00000000 00000000 00000000  *                               *
000040   00000000 00000000 00000000 00000000   00000000 00000000 00000000 00000000  *                               *
000060   00000000 00000000 00000000 00000000   00000000 00000000 40404040 40404040  *                               *
000080   404040F5 F8F0F0F9                                                         *   58009                       *
.空間 -                        11
000000   E2D8D3C3 C1404048 00000088 00000000   00000000 00000000 00000000 00000000  *SQLCA                          *
000020   00000000 00000000 00000000 00000000   00000000 00000000 00000000 00000000  *                               *
000040   00000000 00000000 00000000 00000000   00000000 00000000 00000000 00000000  *                               *
000060   00000000 00000000 00000000 00000000   00000000 00000000 40404040 40404040  *                               *
000080   404040F0 F0F0F0F0                                                         *   00000                       *
.空間 -                        13
000000   00001BB0 00310001 F0F0F0F0 F0F0F0F0   00000000 00000000 00000000 00000000  *        00000000               *
000020   00000470 000002C0 7023C382 57000048   80000000 00000000 007FA083 A3000820  *                               *
000040   80000000 00000000 007FA083 E7000100   D9C3C8C1 E2F2F6F6 40404040 40404040  *               RCHAS266        *
000060   40405CD3 D6C34040 40404040 5CD5C5E3   C1E3D940 D9C3C8C1 E2F2F6F6 5CD3D6C3  *  *LOC      *NETATR RCHAS266*LOC*
0000A0 から 001B9F の行は上と同じ
001BA0   00000000 00000000 00000000 00000000                                     *                               *
.空間 -                        14
000000   E2D4C3C2 20000100 00000010 F0F9F0F4   F5F461E2 D9D961E2 D9D9E2F1 00000000  *SMCB        090454/SRR/SRRS1    *
000020   00000000 00000000 E5F0F2D9 F0F1D4F0   F1D9C3C8 C1E2F3F7 F8000000 00800000  *        V02R01M01RCHAS378       *
000040   0302C3D5 E2E2D5D9 C3E5D8D3 F7F9F7F1   80000000 00000000 007FA083 E9000106  *  CNSSNRCVQL7971                *
000060   F1000000 00710000 00000000 00000000   00000470 000002C0 7023C382 57000048  *1                              *
.空間 -                        15
000000   00000000 00000000 007FA083 E60019FF   00000000 00000000 00000000 00000000  *                               *
000020   00000000 00400000                                                         *                               *
.空間 -                        16
000000   00000000 00000000 00000000 00000002   00000017 000000E1 00000000 00000071  *                               *
000020   00000000 00007FFF 00000003 00170000   001B0000 FF000000 00002410 00F0F060  *                               *
000040   E70400                                                                   *X                              *
.空間 -                        17
000000   E2C3C3C2 5CD3D6C3 40404040 40405CD5   C5E3C1E3 D9405CD3 D6C34040 4040D9C3  *SCCB*LOC      *NETATR *LOC    RC*
000020   C8C1E2F2 F6F65CD3 D6C34040 404007F6   C4C24040 40405CC4 D9C4C140 40404040  *HAS266*LOC     6DB    *DRDA     *
000040   40404040 40404040 4000001E 00110000   00000000 00000000 00000000 00000000  *                               *
000060   00000000 00000000 00000000 00000000                                     *                               *
.空間 -                        18
000000   E2D7C3C2 00000000 007FA083 A3000810   00000470 000002C0 7023C382 57000048  *SPCB                           *
000020   00000000 00000000 00000000 00000000   00000000 00000000 00000000 00000000  *                               *
000040   00000000 00000000 00000000 00000000                                     *                               *
.空間 -                        19
000000   C5E7C3C2 00000076 00000003 00000079   00000009 00000082 00000010 00000092  *EXCB                           *
000020   00000008 00000000 00000018 00200003   00030003 00030003 00030001 00030003  *                               *
000040   00000000 00000000 00000000 00000000   00000000 0000C4C4 D4E5F0F2 D9F0F1D4  *
DDMV02R01M*000060   F0F1F0F4 F5F1F7F4 61E2D9D9 61C4E2F3   F7F8D9C3 C8C1E2F2 F6F6   *
01045174/SRR/DS378RCHAS266                      *
.空間 -                        20
000000   00000030 000002B6 00000430 0000043E   00010000 00000000 00000000 00000000  *                               *
000020   00000000 00000000 00000000 00000000   00000000 00000000 00000000 00000000  *                               *
000040   80000000 00000000 007FA083 D2000100   00000000 0000029A 0000005C 22050000  *                               *
000060   00060000 02B60000 00B00000 00000000   00000000 00000000 00000000 00000000  *                               *
0000E0 から 00017F の行は上と同じ
```

```
.空間 -                          21
000000   0016D052 00010010 22050006 11490000   00062102 24170052 D0530001 0022241A  *                                *
000020   0F76D004 00002600 03020000 0A000009   71E05400 01D00001 0671F0E0 0000002A  *                                *
000040   241BFF00 0001F0F0 F1000000 013FF000   00000000 00FF0000 02F0F0F2 00000002  *      001     0          002     *
000060   40000000 00000000 0010D052 0001000A   220B0006 11490004 0069D003 00010063  *                                *
000080   24080000 000064F0 F2F0F0F0 D8E2D8C6   C5E3C3C8 00D9C3C8 C1E2F2F6 F6404040  *       02000QSQFETCH RCHAS266   *
.空間 -                          22
000000   E2C3C3C2 5CD3D6C3 40404040 40405CD5   C5E3C1E3 D9405CD3 D6C34040 4040D9C3  *SCCB*LOC      *NETATR *LOC    RC*
000020   C8C1E2F2 F6F65CD3 D6C34040 404007F0   F0F14040 4040E77D F0F7C6F0 C6F0C6F1  *HAS266*LOC     001    X'07F0F0F1*
000040   7D404040 40404040 40000014 00110000   00000000 00000000 00000000 00000000  *                                *
000060   00000000 00000000 00000000 00000000   00008F00 00000700 F0F0F100 00000000  *                          001     *
.空間 -                          23
000000   C5E7C3C2 00000076 00000003 00000079   00000009 00000082 00000010 00000092  *EXCB                    b       k*
000020   00000008 00000000 00000018 00200003   00030003 00030003 00030001 00030003  *                                *
000040   00000000 00000000 00000000 00000000   00000000 0000C4C4 D4E5F0F2 D9F0F1D4  *                        DDMV02R01M*
000060   F0F1F0F4 F5F1F7F2 61E2D9D9 61C4E2F3   F7F8D9C3 C8C1E2F2 F6F6              *01045172/SRR/DS378RCHAS266      *
.空間 -                          24
000000   00000030 0000005C 00000000 000000CC   00010000 00000000 00000000 00000000  *       *                        *
000020   00000000 00000000 00000000 00000000   00000000 00000000 00000000 00000000  *                                *
000040   80000000 00000000 007FA083 A4000100   00000000 00000000 0000005C D2010000  *                            *K   *
.空間 -                          25
000000   0010D002 0001000A D2010006 11490000   E2000D11 5AE5F0F2 D9F0F1D4 F0F1000C  *                      V02R01M01  *
000020   116DD9C3 C8C1E2F2 F6F60014 115EF0F4   F5F1F7F2 61E2D9D9 61C4E2F3 F7F80064  *  RCHAS266    045172/SRR/DS378  *
000040   14041403 00031423 00031405 00031406   00031407 00031444 00031458 00011457  *                                *
000060   0003140C 00031419 0003141E 00031422   0003240F 000314A0 00041432 00031433  *
ダンプの終わり
* * * * *  リ ス ト の 終 わ り  * * * * *
```

FFDC ダンプ出力の説明

以下に、前ページの図のような FFDC ダンプ出力に示されるデータ域および情報のタイプについて説明します。

注:

1. FFDC ダンプ出力は、それぞれ内容は異なっていますが、一般に形式は同じです。各データ・セクションの内容とロケーションを理解するのに役立つよう、索引 (**I**) が用意されています。
2. 各データ・セクションは、『SPACE-』 と番号で識別されます。たとえば、SPACE- ... 01 などです。ダンプ出力に示されるデータ・セクションは、障害時の操作とその進行状況によって異なります。
3. 各データ・セクションには、たとえば SQCA のような名前が付けられます。 SQCA というのは、DB2 UDB QUERY マネージャーおよび SQL 開発キットの SQL 診断域からのデータのセクション名です。SQL データを見つけ出すには、索引 (**I**) 内で SQCA を探します。サンプルのダンプ索引では、SQCA はデータ・セクション 10 (10=SQCA) 内で見つかります。 SQL データを表示するには、SPACE- 10 に移動します。
4. ダンプできるモジュールには、2 つの基本的なクラスがあります。
   - **アプリケーション・リクエスター (AR)** モジュール
   - **アプリケーション・サーバー (AS)** モジュール

   サンプル・ダンプ出力は、AR モジュールからの典型的なダンプです。通常、AR ダンプ出力には、索引で識別されている固定数のデータ・セクションがあります。 (たとえば、サンプルのダンプ出力では SPACE- 01から16 が示されています。) さらに、可変数の他のデータ・セクションもあります。これらのセクションは、索引には含まれていません。 (たとえば、サンプル・ダンプ出力では、SPACE- 17 から 25 は索引内に示されていません。)

   通常、アプリケーション・サーバー・ダンプ出力のほうが単純です。これは、これらの出力が固定数のデータ・セクション (すべて索引で識別される) だけから構成されているためです。

5. 現行のダンプ出力内に実際にデータ・セクションがあるかどうかに関係なく、すべてのデータ・セクションについて索引項目があります。たとえば、サンプル・ダンプ出力では、SPACE- 08 はありません。索引では 08 は QDTA (照会データ) に等しくなっています。 SPACE- 08 がないということは、照会データは戻されていないので、ダンプする照会データがないことを意味します。
6. サンプル・ダンプ出力では、索引の最終項目は 『(REST IS CCB, PCBS, SAT, PMAP, RCVB, PER CCB)』 です。この項目は、SPACE- 17 以上に、 1 つ以上の通信制御ブロック (CCB) が入っていることを意味します。そのおのおのに、次のものが入っています。
   - 0 個、1 個、または複数の経路制御ブロック (SPCB)。通常は 1 個だけです。
   - 交換サーバー属性制御ブロック (EXCB)
   - 構文解析プログラム・マップ・スペース
   - 通信制御ブロック用の受信バッファー

   データ・セクション番号は、17 以上では、制御ブロックが 1 つダンプされるたびに 1 ずつ増えていきます。たとえば、サンプル・ダンプ出力では、次に示すとおり、データ・セクション SPACE- 17 から SPACE- 21 は、最初のダンプ・データ制御ブロック (CCB 1) のものであるのに対して、データ・セクション SPACE- 22 から SPACE- 25 は、 2 番目のダンプ・データ制御ブロック (CCB 2) のものです。

   17 CCB (目印は :'SCCB:'。 アプリケーション・サーバー・モジュールの場合、目印は :'TCCB:' です。)

   18 CCB 1 の PCB (目印は :'SPBC:')

   19 CCB 1 の SAT (目印は :'EXCB:')

   20 CCB 1 の PMAP (目印はなし)

   21 CCB 1 の RCVB (目印はなし)

   22 CCB 2 (目印は :'SCCB:')

   -- (会話が活動中ではないので CCB 2 の PCB はなし)

   23 CCB 2 の SAT (目印は :'EXCB:')

   24 CCB 2 の PMAP (目印はなし)

   25 CCB 2 の RCVB (目印はなし)

   A ダンプがとられたサーバーの名前およびリリース情報。

   B ダンプ出力を作成したジョブの名前。

   C 障害発生の疑いがあるオペレーティング・システム内のモジュールの名前。

   D 障害を検出したモジュールの名前。

徴候ストリングの内容は次のとおりです。

E メッセージ ID。

F FFDC ダンプがとられる原因となったと思われるモジュールの名前。

G 障害点を識別する戻りコード (RC)。

RC の後の最初の数字は、この障害に関連したダンプ・ファイルの番号を示します。割り振られていた会話数によっては、複数のダンプ・ファイルがある可能性があります。サンプル・ダンプ出力ではその数字は 『1』 ですが、これは、この障害に関連した最初の (そして唯一のものと考えられる) ダンプ・ファイルであることを示します。

戻りコードの右端に、エラーのタイプを示す 4 桁 (ゼロでない) がある場合があります。

- AR によって検出されたエラーについてのコードとして考えられるのは次のとおりです。

  0001 リモート・データベースへの接続において障害が発生した。

  0002 「まだ受信中」標識がオンであってはいけないときにオンになった。

  0003 AR は、AS から受け取ったデータ・ストリーム内で、認識されていないオブジェクトを検出した。

  0097 AR DDM 通信管理プログラムによってエラーが検出された。

  0098 AR の DDM 構成要素よって会話プロトコル・エラーが検出された。

  0099 機能チェック

- AS によって検出されるエラーのコードとして考えられるのは次のとおりです。

  0099 機能チェック

  4415 会話プロトコル・エラー

  4458 エージェント永続エラー

  4459 リソースの限界に達した

  4684 データ・ストリームの構文が無効

  4688 コマンドがサポートされていない

  4689 パラメーターがサポートされていない

  4690 値がサポートされていない

  4691 オブジェクトがサポートされていない

  4692 コマンド・チェック

  8706 照会がオープンされない

  8708 リモート・データベースがアクセスされない

  8711 リモート・データベースは事前にアクセスされている

  8713 パッケージ・バインド処理が活動状態

  8714 FDO:CA 記述子が無効

  8717 作業単位の異常終了

  8718 データと記述子の一方または両方が不一致

  8719 照会はすでにオープンされている

  8722 照会オープン障害

  8730 リモート・データベースが使用可能でない

H データ・セクションを識別する SPACE- 番号。この番号は、索引によってデータ・セクション名に関連付けられます。データ・セクション名は、下記の **I** に定義されています。

I 各データ・セクションの内容とロケーションを理解するのに役立つよう、索引と SPACE- 番号 (**H** に定義されています) が用意されています。モジュールが異なれば、ダンプ出力ごとに、さまざまなデータ・セクションの順序も異なることがあります。データ・セクション名の意味は次のとおりです。

- AFT: すべての会話情報の入った DDM 活動ファイル・テーブル。

- ARDB: AR および AS 接続情報の入ったアクセス・リモート・データベース制御ブロック。
- ARDP: ユーザー・スペースの先頭にある ARD プログラム・パラメーター。
- BDTA: SELECT INTO ステートメントからのバッファー処理通信域 (BPCA) と関連データ・レコード。
- Bind: SQL バインド・テンプレート
- BPCA: BPCA 構造 (データ・レコードなし)。
- DATA: BPCA に関連したデータ・レコード。このセクション内のレコードは、BPCA バッファー全体の内容を反映していない可能性があります。処理済みのレコードは含まれていないかもしれません。
- DOFF: エラーが検出された照会データ・ストリーム (QRYDTA) 内のオフセット。
- EICB: エラー情報制御ブロック。
- EMSG: 機能チェックまたは DDM 通信管理プログラム・エラーに関連したエラー・メッセージ。
- FCT: DDM 機能コード・ポイント (2 バイト)。
- FDOB: 実行操作内の構文解析プログラムへの FDO:CA 記述子入力。
- FDTA: 以下のもので構成される FDO:CA データ構造。
  - FDO:CA データ・ストリーム (FDODTA) の長さを定義する 4 バイト・フィールド。
  - FDODTA
- HDRS: 通信管理プログラム・コマンド・ヘッダー・スタック。
- IFMT: ARD プログラム入力フォーマット。
- INDA: 挿入、選択、削除、更新、オープン、および実行操作用のユーザー定義 SQLDA の入った入力 SQLDA。
- INDX: データ・セクション名をデータ・セクション SPACE- コードにマップする索引。索引内のすべての項目に、対応するデータ・セクションがあるわけではありません。ダンプ・データは、発生したエラーと、エラー発生時の操作の進行状況に基づいたものです。スプール・ファイルごとに最大 32 の項目をダンプできます。
- INST: SQL ステートメント。
- ITKN: 割り込みトークン。
- OFMT: ARD プログラム出力フォーマット。
- PKGN: 入力パッケージ名、整合性トークン、およびセクション番号。
- PMAP: AS ダンプ出力内の構文解析プログラム・マップ。
- PRMS: DDM モジュールの入力または出力のパラメーター構造。
- PSOP: 入力構文解析プログラム・オプション。
- QDTA: 以下のものから成る照会データ構造。
  - 照会データ・ストリーム (QRYDTA) の長さを定義する 4 バイト・フィールド
  - QRYDTA
- RCVB: 受信データ・ストリーム。内容は以下のように、場合によって異なります。
  - ダンプがアプリケーション・サーバーで行われた場合、そのセクションには、アプリケーション・リクエスターから送られた DDM 要求データが入ります。

– ダンプがアプリケーション・リクエスターで行われた場合、そのセクションには、アプリケーション・サーバーから送信されてきた DDM 応答データが入ります。このセクションが存在しない場合、受信データは、ダンプの可変部内の受信バッファー内で見つかることがあります。

- RDBD: リレーショナル・データベース・ディレクトリー。
- RFMT: レコード様式構造。
- RMTI: コミットメント制御ブロック内のリモート・ロケーション情報。
- RTDA: 戻された SQLDA (ARD プログラムから)。
- SMCB: 他の DDM 接続制御ブロックと内部 DDM 制御ブロックを指し示すポインターの入った DDM ソース・マスター制御ブロック。
- SNDB: 送信データ・ストリーム。内容は以下のように、場合によって異なります。
  – ダンプがアプリケーション・リクエスターで行われた場合、そのバッファーには、アプリケーション・サーバーに送信されたか、または送信準備が行われていた DDM 要求が入ります。

    例の中の SPACE- 05 の先頭に、4 バイトのゼロがあることに気を付けてください。これらのゼロがあっても、これらはデータ・ストリームの一部を成しません。これらは、DDM 要求で DDM ラージ・オブジェクトを送信する必要が生じたときしか使われないバッファー内のスペースを表します。その場合、DDM 要求ストリームは、左に 4 バイト・シフトされます。
  – ダンプがアプリケーション・サーバーで行われる場合、そのバッファーには、アプリケーション・リクエスターに送信準備が行われていた DDM 応答データが入ります。
- SQCA: ユーザーに戻される出力 SQL。
- SQDA: FDO:CA 構文解析プログラムによって作成される SQLDA。
- TBNM: 入力リモート・データベース・テーブル名。
- TMCB: ターゲットの主制御ブロック。
- TSLK: DDM 活動ファイル・テーブルおよび他の内部 DDM 制御ブロックを指し示すポインターを含む、ターゲットまたはソース接続制御ブロック。
- VARS: ダンプされているモジュールのローカル変数。
- WRCA: オープン操作 (OPNQRYRM) の場合にだけ戻される警告 SQLCA。
- XSAT: 交換サーバー属性制御ブロック。
- その他: エラー時のジョブのすべての DDM ジョブ用の複数の会話制御ブロック。各会話制御ブロックには、次のものが入っています。
  – 確立済みの会話に関する情報の入ったパス制御ブロック。 1 つの会話制御ブロックに対して複数のパス制御ブロックがあることもあります。
  – 1 つの交換サーバー情報制御ブロック。アプリケーション・リクエスターとアプリケーション・サーバーについての情報を含んでいます。
  – 1 つの DDM 構文解析プログラム・マップ域。すべての DDM コマンド、オブジェクト、および応答のローカルと値を含んでいます。
  – 1 つの受信バッファー。アプリケーション・サーバーで受信された要求データ・ストリームが入っています。 6 (272 ページ) も参照してください。

  データ・セクション番号は、制御ブロックが 1 つダンプされるたびに 1 ずつ増えていきます。

J 目印域。ダンプされたいくつかの区域のデータ・タイプを識別する情報。

K 障害時に進行中であった会話の作業論理単位 ID (LUWID) は、アクセス RDB 制御ブロック内で見つけることができます。このデータ域は、FFDC 索引内ではストリング ‘ARDB’ で識別されて

います。この例では、SPACE- 07 です。 LUWID はオフセット 180 から開始します。ネットワーク ID (NETID) は APPC です。これは、その後に続く論理装置 (LU) 名 RCHAS378 とピリオドで区切られます。 LU 名の後には、6 バイトの LUW インスタンス番号 X'A7CCA7541372' が続きます。

DDM エラー・コード

このエラー・コードは、 DDM エラー条件を識別する FFDC ダンプ (サンプル・ダンプ出力では **L**) に入っています。このような条件は、DDM 体系で定義されていることもいないこともあります。

コマンド検査コード

FCT+ (SPACE- 02) が、バイト 3 および 4 に 1254 を含んでいる場合、バイト 6 でこれらのコードの 1 つを探してください。

01 リレーショナル・データベース (RDB) への接続失敗。

02 DDM データ・ストリームの状態が誤り。

03 データ・ストリーム内に認識されないオブジェクト。

04 SQL から受け取った CCSID ステートメントが認識できない。

05 EXCSQLSTT OUTEXP 値が実行中の SQL ステートメントと矛盾する。

06 アプリケーション・サーバー (AS) に送られた DDM コマンドまたはオブジェクトが DRDA2 アーキテクチャーに対する OS/400 拡張に違反。

07 AS から受け取った DDM 応答またはオブジェクトが DRDA2 アーキテクチャーに違反。

08 SQLDA データ・ポインターが、ヌルであってはならないのにヌルである。

09 プロダクト・データ構造が無効。

0A XLATECC 失敗。

0B EXTJOBDI 失敗。

0C 名前からの ASP の取得に失敗。

0D ASP からの RDB 名の取得に失敗。

0E 予期しないエラー・データ。

0F DDM/DRDA 要求が認識されない。

10 予定されていた LOB は受信されなかった。

11 プレースホルダーと受信データとで LOB の長さが不一致。

12 LOB の使用が不一致。

13 LOB に対して XMIT モードが誤り。

14 バッファー拡張に失敗。

15 正常なオープンの後の取り出しで SQLCODE が負。

16 スペース割り振りエラー。

17 結果セット応答 (SQRY) が不一致。

18 結果セット応答 (SQRY) に予期せぬ RM。

19 応答の作成中にエラー。

1A SQ コンポーネントが SQL コード -30020 を戻した。

1B SQL 診断域の更新中にエラー。

1C 応答の作成中にエラー。

88 BPCA 内にレコードがない。

89 予期せぬ BGNBND オブジェクト。

8A DDM ラージ・オブジェクトのヘッダーのサイズはサポートされていない。

8B LOB テーブル・エラー。

8C LOB が要求されたが、使用可能な LOB がない。

8D SET_LELAST エラー 1。

8E SET_LELAST エラー 2。

8F 予期しない非ゼロ QRYINSID。

90 非ゼロ QRYINSID。

91 P オープン時に OPNQFL。

92 通常オープン時に OPNQFL。

97 DDM 通信管理プログラムがエラーを検出した。

98 会話プロトコル・エラーが DDM モジュールによって検出された。

99 機能チェック。 EMSG セクションを検索のこと。通常は SPACE- 03 内にある。

FF SQ オープン (TQRY) でエラー。

会話プロトコル・エラー・コードの説明

FCT+ (SPACE- 02) で、バイト 3 および 4 に 1245 が入っている場合、バイト 6 で以下のいずれかのコードを探してください。

01 ターゲットの通信管理プログラムによって RPYDSS が受信された。

02 連鎖のない複数の DSS が送られた、あるいは複数の DSS 連鎖が送られた。

03 OBJDSS が送信許可されていないときに送信された。

04 RQSDSS の要求相関 ID が、連鎖内の前の RQSDSS 要求相関 ID と同じかそれより小さい値である。

2 つの RQSDSS に同じ要求相関 ID がある場合、要求相関 ID から 1 を引いて、 RPYDSS 内で PRECCNVRM を送信しなければなりません。

05 OBJDSS の要求相関 ID が、前の RQSDSS の要求相関 ID と同じではない。

06 EXCSAT が、接続後の最初のコマンドではなかった。

DA SQLDA は、ラベルを収容できるよう 2 倍になっていない。

DF FDODSC が受け取られたが、付随の FDODTA がない。

E0 OPNQRY (照会のオープン) 応答メッセージがない。

E1 ENDQRYRM (照会応答メッセージ終了) 上の RDBNAM が無効。

E2 OPEN が、QRYDSC (照会応答設定記述) のない QRYDTA (照会応答設定データ) を獲得した。

E3 予期しない OPNQRY 応答オブジェクト。

E4 予期しない CXXQRY 応答オブジェクト。

E5 OPEN 上に QRYDTA (単一行)。

E6 OPNQRYRM 後の RM が無効。

E7 割り込み応答メッセージがない。

E8 アプリケーション・サーバー (AS) がサポートされていないところで LOB 要求があった。

E9 標準バージョンの SQLDA を予期していたのに省略バージョンが戻された。

FD エラー RM に続くヌルの SQLCARD (SQLCA 応答データ)。

FE ヌルの SQLCA にヌルの QRYDTA が続いている。

FF 予期していた SQLCARD がない。

DDM 構文エラー・コードの説明

FCT+ (SPACE- 02) で、バイト 3 および 4 に 124C が入っている場合、バイト 6 で以下のいずれかのコードを探してください。

01 DSS ヘッダー長が 6 より短い。

02 DSS ヘッダー長が、見つかったデータのバイト数と一致していない。

03 DSS ヘッド C バイトが X'D0' でない。

04 DSS ヘッダー F バイトが、認識されていないか、またはサポートされていない。

05 DSS の続行が指定されたが、見つからない。たとえば、最後の DSS で DSS の続行が指定されましたが、 SNA LU 6.2 通信プログラムから SEND 標識が戻されました。

06 DSS 連鎖が指定されたが、DSS が見つからない。たとえば、最後の DSS で DSS の連鎖が指定されましたが、 SNA LU 6.2 通信プログラムから SEND 標識が戻されました。

07 オブジェクトの長さが 4 より短い。たとえば、コマンド・パラメーターの長さが 2 と指定されたか、またはコマンドの長さが 3 と指定されました。

08 オブジェクト長が、見つかったデータのバイト数と一致していない。たとえば、長さが 150 の RQSDSS に、長さが 125 のコマンドが入っているか、または、SRVDGN (サーバー診断情報) パラメーターに 200 の長さが指定されたのに、 DSS 内には 50 バイトしか残っていません。

09 オブジェクトの長さが、指定できる最大長より長い。たとえば、RECCNT パラメーターに長さ 5 を指定しましたが、それは、時間フィールドが、全体ではなく、半分しかないことを示します。

0A オブジェクトの長さが、必要最小限の長さより短い。たとえば、SVRCOD パラメーターに 5 の長さを指定しましたが、そのパラメーターは、6 の固定長をもつと定義されています。

0B オブジェクト長が許容されていない。たとえば、FILEXPDT パラメーターに長さ 11 を指定しましたが、それは、時間フィールドが、全体ではなく、半分しかないことを示します。

0C ラージ・オブジェクトの拡張長さフィールドが正しくない (DSS の説明を参照)。たとえば、拡張長さフィールドがありますが、長さが 3 バイトしかありません。これは、2 バイトの倍数の長さと定義されています。

0D オブジェクト・コード・ポイントの索引がサポートされていない。たとえば、X'8032' というコード・ポイントが検出されましたが、 X'8' は予約済みのコード・ポイント索引です。

0E 必須オブジェクトが見つからない。たとえば、CLRFIL コマンドに FILNAM パラメーターがないか、または、 MODREC コマンドの後に RECORD コマンド・データ・オブジェクトが続いていません。

0F 送信したコマンド・データ・オブジェクトが多すぎる。たとえば、MODREC コマンドの後に 2 つの RECORD コマンド・データ・オブジェクトが続いているか、または DELREC コマンドの後に RECORD オブジェクトが続いています。

10 相互に排他的なオブジェクトが存在する。たとえば、CRTDIRF コマンドに DCLNAM パラメーターと FILNAM パラメーターを指定しました。

11 送信したコマンド・データ・オブジェクトが少なすぎる。たとえば、RECCNT(5) を指定した INSRECEF コマンドの後に、 4 つの RECORD コマンド・データ・オブジェクトしか続いていません。

12 オブジェクトが重複している。たとえば、LSTFAT コマンドに 2 つの FILNAM パラメーターを指定しました。

13 指定した要求相関 ID は無効。そのエラー・コードの代わりに、 X'04' または X'05' の PRCCNVCD を指定して PRCCNVRM を使ってください。このエラー・コードは、レベル 1 のアーキテクチャーとの互換性のために保持されています。

14 必要な値が見つからない。

15 予約済みの値は使えない。たとえば、INSRECEF コマンドに RECCNT(0) パラメーターを指定しました。

16 DSS 連結が 2 以下である。たとえば、DSS 連結の長さバイトの値が 1 になっています。

17 オブジェクトが、規定の順序になっていない。たとえば、RECAL オブジェクトに RECORD オブジェクトが入っていて、その後に RECNBR オブジェクトが続いていますが、これは指定された順序ではありません。

18 DSS 連鎖ビットが 2 進数 1 ではないが、DSSFMT ビット 3 が 2 進数 1 に設定されている。

19 前の DSS が現行 DSS が同じ要求相関を持っていることを示したが、要求相関 ID が同じではない。

1A DSS 連鎖ビットが 2 進数 1 ではないが、エラー連結が要求されている。

1B 相互に排他的なパラメーター値を指定した。たとえば、OPEN コマンドに PRPSHD(TRUE) と FILSHR(READER) を指定しました。

1D コード・ポイントが有効なコマンドではない。たとえば、RQSDSS 内の最初のコード・ポイントが、辞書内にないか、またはコマンド用のコード・ポイントではありません。

付録 D. 用語集

［ア行］

アクセス・プラン (access plan)
DB2 UDB for iSeries において、プログラムの実行時に出現した SQL ステートメントの処理に使用される、コンパイル時に生成される制御構造。

アプリケーション・サーバー (AS) (application server (AS))
分散リレーショナル・データベースを使用している場合に、リモート・データが置かれるシステム。

アプリケーション・リクエスター (AR) (application requester (AR))
分散リレーショナル・データベースを使用している場合に、アプリケーション・プログラムが実行されるシステム。

アプリケーション・リクエスター・ドライバー (ARD) (application requester driver (ARD))
OS/400 の SQL クライアント統合機能で使用される出口プログラム。 OS/400 リレーショナル・データベース以外のデータベース管理システムが管理するデータを SQL アプリケーションでアクセスできるようにする。

暗号化 (encryption)
コンピューター・セキュリティーにおいて、オリジナル・データを入手できないように、または暗号化解除プロセスを使用することによってのみ入手できるように、データを理解できない形態に変換するプロセス。

異種環境 (unlike environment)
異種環境は、以下のいずれかの条件で発生する。

- DB2 UDB for iSeries データベース管理システム (DBMS) でない AR が、 DB2 UDB for iSeries DBMS にアクセスする。
- DB2 UDB for iSeries DBMS が、DB2 UDB for iSeries DBMS でない AS にアクセスする。

注: Universal Driver for JDBC and SQLJ は、 iSeries サーバー上で AR として稼働させることができたとしても、 iSeries AS に接続されると異種 AR とみなされる。これは、DBMS の一部ではないためである。

エンタープライズ識別マッピング (EIM) (Enterprise Identity Mapping (EIM))
EIM とは、企業全体に渡って、個人またはエンティティーをさまざまなレジストリー内の適切なユーザー ID にマップする (関連付ける) ためのメカニズムのこと。 EIM には、これらの ID マッピング関係を作成したり管理したりするための API と、この情報を照会するためにアプリケーションが使用する API が用意されている。

［カ行］

拡張対等通信ネットワーキング機能 (APPN) (Advanced Peer-to-Peer Networking(R) (APPN))
ネットワーク内の直接接続する必要のない複数の APPC システム間において、データの経路を定めるデータ通信サポートに関する用語。 IBM のシステム・ネットワーク体系 (SNA) の一部。

拡張プログラム間通信 (APPC)
iSeries サーバー上のプログラムが、互換性のある通信サポートを備えた他のシステム上のプログラムと通信できるようにするデータ通信サポート。 iSeries サーバー上 の APPC は、SNA LU タイプ 6.2、およびノード・タイプ 2.1 のアーキテクチャーに準拠したアプリケーション・プログラミング・インターフェースを提供する。 IBM のシステム・ネットワーク体系 (SNA) の一部。

カタログ (catalog)
テーブル、パッケージ、ビュー、索引、および制約についての情報を含む、一連のテーブルとビュー。QSYS2 のカタログ・ビューには、iSeries サーバーにおける、すべてのテーブル、パッケージ、ビュー、索引、および制約についての情報が含まれている。さらに、SQL スキーマには、スキーマ内のテーブル、パッケージ、ビュー、索引、および制約についての情報だけを含む一連のビューがある。

コード化文字セット識別子 (CCSID) (coded character set identifier (CCSID))
使用されるコード化図形文字表現を一意的に識別する、エンコード・スキーム ID、文字セット ID、コード・ページ ID、およびその他の関連情報の特定のセットを識別する 16 ビットの数値。

コード・ページ (code page)
文字と内部コードとの間における割り当ての特定のセット。

高性能経路指定 (HPR) (high-performance routing (HPR))
データ経路指定のパフォーマンスと信頼性を改善する (特に高速リンクを使用する場合)、 Advanced Peer-to-Peer Networking (APPN) アーキテクチャーの追加機能。

構造化照会言語 (SQL) (Structured Query Language (SQL))
ホスト・プログラム言語の中で使用するか対話式に使用することにより、データベースに情報を書き込んだり、データベースから選択した情報を取得して編成したりすることができる言語。 SQL は、データベース・リソースへのアクセスを制御することもできる。 SQL は、異なるサーバー間で分散データ処理を行えるようにするために必要な整合性を提供する。

コミット (commit)
要求が完了したら、アプリケーション・プログラムは、作業単位をコミットできる。すなわち、作業単位に関連したデータベースの変更はすべて永続的なものになる。

コミットメント制御 (commitment control)
コミット可能リソースの操作をグループ化して、コミット可能リソースの変更処理のグループをコミット・コマンドによって 1 単位として処理できるようにしたり、コミット可能リソースの変更内容のグループをロールバック・コマンドによって 1 単位として削除できるようにしたりする手段。

［サ行］

作業単位 (unit of work)
作業単位とは、完成した作業の一部を構成する、 1 つ以上のデータベース要求および関連した処理のこと。トランザクションと同じ。

サブシステム (subsystem)
システムが処理とリソースの調整を行う、サブシステム記述によって定義される操作環境。

サブシステム記述 (subsystem description)
システムが制御する操作環境特性を定義する情報が入っているシステム・オブジェクト。オブジェクト・タイプのシステム認識 ID は *SBSD。

システム・サービス制御点 (SSCP) (system services control point (SSCP))
他のシステムおよび装置を管理し、ネットワーク・オペレーターの要求と問題分析要求を調整し、またネットワーク・ユーザーのためにディレクトリーの経路指定などのセッション・サービスを提供する SNA ネットワーク内のフォーカル・ポイント。

システム・ネットワーク体系 (SNA) (Systems Network Architecture (SNA))
IBM ネットワークにおいて、階層化論理構造、形式、プロトコル、および操作手順の記述。これは、ネットワークを介して情報単位を伝送したり、ネットワークの構成および操作を制御したりするために使用される。

システム・リレーショナル・データベースまたはシステム・データベース (System Relational Database, or System Database)
iSeries サーバーに接続されたディスク上に存在し、独立補助記憶域プール上には保管されないすべてのデータベース・オブジェクト。

ジャーナル (journal)
ジャーナル記録されるオブジェクト、現行ジャーナル・レシーバー、およびシステム上にあるそのジャーナルに対するすべてのジャーナル・レシーバーを識別するシステム・オブジェクト。オブジェクト・タイプのシステム認識 ID は *JRN。

スキーマ (schema)
ライブラリー、ジャーナル、ジャーナル・レシーバー、SQL カタログ、およびオプションのデータ・ディクショナリーで構成される。スキーマを使うと、関連するオブジェクトがグループ化され、名前でオブジェクトを見つけられるようになる。注: スキーマは一般にコレクションとも呼ばれます。

スプール (spool)
(1) ファイルやジョブをディスク記憶域に入れて、後で処理または印刷するシステム機能。 (2) 周辺装置とコンピューターのプロセッサーとの間でデータを転送する際に、補助記憶装置をバッファー・ストレージとして使用することによって、処理の遅延を低減すること。

制御サブシステム (controlling subsystem)
システムの始動時に最初に自動的に開始され、システム・オペレーターがシステムを制御するために使用する対話式サブシステム。

［タ行］

第 1 障害データ検知 (FFDC) (first-failure data capture (FFDC))
問題認識、診断データの選択ダンプ、症状ストリングの生成、および問題ログの記録を行う FFST™ アーキテクチャーを OS/400 にインプリメントしたもの。

対話式構造化照会言語 (ISQL) (Interactive Structured Query Language (ISQL))
SQL ステートメントをバッチ・モードではなく動的に実行できるようにする DB2 UDB Query Manager and SQL Development Kit ライセンス・プログラムの機能。すべての対話式 SQL ステートメントが、ワークステーションから読み取られ、準備されて動的に実行される。

対話式処理 (interactive processing)
オペレーターがアクションを行うたびに、プログラムまたはシステムが応答する処理方式。「バッチ処理 (batch processing)」と対比。

チケット認可チケット (TGT) (ticket-granting ticket (TGT))
サービス・チケットが要求されたときにチケット認可サーバーにプリンシパルが渡すチケット。チケット認可サービスは、このチケット認可チケットを使用して、サービス・チケットの要求を認可する前に認証サーバーに対してプリンシパルが認証されていることを検証する。

ディスプレイ・パススルー (display station pass-through)
通信機能の 1 つで、これを使用すれば、ユーザーは iSeries server から別の iSeries server にサインオンし、そのサーバーのプログラムやデータを使用することができる。

伝送制御プロトコル/インターネット・プロトコル (TCP/IP) (Transmission Control Protocol/Internet Protocol (TCP/IP))
(1) ローカルおよび広域ネットワークの両方について対等接続機能をサポートする通信プロトコル・セット。 (2) インターネット上で使用される基本通信プロトコル。TCP/IP は内部ネットワークにおいても使用できる。

同期データ・リンク制御 (SDLC) (synchronous data link control (SDLC))
(1) コマンドを用いて通信回線によるデータの転送を制御する通信回線制御の形式。 (2) 通信回線によ

る同期、コード透過性、ビット直列情報の転送に関して、米国規格協会 (ANSI) の拡張データ通信制御手順 (ADCCP) のサブセット、および国際標準化機構 (ISO) の高水準データ・リンク制御 (HDLC) に準拠している通信規則。伝送交換は、交換回線または非交換回線上で、全二重でも半二重でも行うことができる。接続の構成は、2 地点間、マルチポイント、またはループとすることができる。

同種環境 (Like environment)
分散リレーショナル・データへのアクセスが、複数の DB2 UDB for iSeries データベース管理システム (DBMS) 間で行われる場合の環境。

独立補助記憶域プールまたは独立ディスク・プール (Independent auxilary storage pool, or independent disk pool)
アドレス可能なディスク記憶域を構成する、ディスク装置またはディスク装置サブシステムから定義される 1 つ以上の記憶装置。独立ディスク・プールには、オブジェクト、オブジェクトが入っているディレクトリー、および権限所有属性などの他のオブジェクト属性が含まれる。独立ディスク・プールは、システムを再始動することなく、使用可能に (オンに変更) したり無効に (オフに変更) することができる。独立ディスク・プールは、a) クラスター化環境において複数システム間で切り替え可能にするか、 b) 単一システムに専用接続することができる。

ドメイン・ネーム・システム (DNS) (Domain Name System (DNS))
インターネット・プロトコル・スイートにおいて、ドメイン・ネームを IP アドレスにマップするために使用される分散データベース・システム。

トランザクション・プログラム名 (TPN) (transaction program name (TPN))
LU 6.2 会話に参加している各プログラムを識別する名前。通常は、接続の起動側が、接続先であるもう一方の LU にあるプログラム名を識別する。 LU 名と組み合わされて使用されるとき、TPN はネットワーク内の特定トランザクション・プログラムを識別する。

［ハ行］

バインド (binding)
(1) 統合化言語環境 (Integrated Language Environment® (ILE)) モジュールをパッケージし、これらのモジュール間で渡されるシンボルを解決することによって、プログラムを作成するプロセス。たとえば、アプリケーション・プログラムで実行できるようにするには、その前にプログラムと参照されるテーブルおよびビューとの間の関係が確立されていなければならない。 (2) DRDA において、アプリケーション・サーバー (AS) 上で SQL パッケージを作成するプロセス。

パッケージ (package)
「SQL パッケージ (SQL package)」 を参照。

バッチ処理 (batch processing)
単独のプログラム、または一連のプログラムを実行する方式の 1 つ。この方式では、1 つまたは 複数のレコード (バッチ) が、ユーザーまたはオペレーターからのアクションをほとんど、またはまったく必要とせずに処理される。「対話式処理 (interactive processing)」と対比。

物理装置 (PU) (physical unit (PU))
SNA における、3 種類のネットワーク・アドレス可能単位のうちの 1 つ。物理装置は、SNA ネットワークの各ノードに存在し、システム・サービス制御点論理装置 (SSCP-LU) セッションによって要求されたときに、ノードのリソース (接続されたリンク・ステーションや、隣接リンク・ステーションなど) を管理し、監視する。

分散作業単位 (DUW) (distributed unit of work (DUW))
分散作業単位 (DUW) を使うと、ユーザーまたはアプリケーション・プログラムは、 1 つの作業単位で、複数の場所にあるデータを読んだり更新することが可能になる。

分散データ管理機能 (DDM) (distributed data management (DDM))
オペレーティング・システムの機能の 1 つで、これを使用すれば、 1 つのシステムのアプリケーション・プログラムまたはユーザーは、リモート・システムに格納されているデータベース・ファイルを使用できる。システムは通信ネットワークによって接続されている必要があり、リモート・システムも DDM を使用している必要がある。この用語は、基礎をなす通信アーキテクチャーにも適用される。

分散リレーショナル・データベース (distributed relational database)
分散リレーショナル・データベースは、データを使うアプリケーション・プログラムおよびデータそのものが別のマシンに置かれている場合、または、プログラムが同じサーバー上の複数のデータベースに置かれているデータを使用する場合に存在する。後者の場合、単一のサーバー内の 1 つ以上のデータベースにアクセスするために DRDA プロトコルが使用されるという意味で、データベースは分散している。

分散リレーショナル・データベース体系 (DRDA) (Distributed Relational Database Architecture (DRDA))
DRDA は、以下の目的で一連のフォーマットとプロトコルを指定する。

- あるアプリケーションから 1 つ以上のリモート・データベース管理システムへ接続する。
- 両者の間でデータ交換を行う。
- トランザクションの保全性と一貫性を確保するためにトランザクションを管理する。

DRDA プロトコルは、分散データ管理機能アーキテクチャーに基づいて構築される。

ペーシング (pacing)
SNA において、受信側システムが送信側システムの速度を制御して、オーバーランを防ぐ技法。

補助記憶域プール (ASP) (auxiliary storage pool (ASP))
補助記憶域を構成する、記憶装置または記憶装置サブシステムから定義された 1 つまたは複数の記憶装置。 ASP では、記憶装置の故障による影響を制限し、回復時間を短縮するデータの編成方法が採用される。

ホスト言語 (host language)
DB2 UDB for iSeries SQL において、SQL ステートメントを組み込むことができる、C、COBOL、RPG などのプログラム言語。

ホスト変数 (host variable)
DB2 UDB for iSeries SQL アプリケーション・プログラムにおいて、組み込み SQL ステートメントによって参照される変数。 RPG では、これはフィールド名と呼ばれる。C では、これは変数と呼ばれる。 COBOL では、これはデータ項目と呼ばれる。

ホスト・プログラム (host program)
DB2 UDB for iSeries において、組み込み SQL ステートメントを含むホスト言語で書かれたプログラム。

［マ行］

文字データ表現体系 (CDRA) (Character Data Representation Architecture (CDRA))
IBM アーキテ クチャーの 1 つ。 iSeries サーバーおよび CDRA をサポートする他のタイプのサーバー間で、一貫性のある文字 (データ) の表示、処理、および交換を実現するために、一連の ID、サービス、サポート・リソース、および規則を定義する。

［ヤ行］

ユーザー・リレーショナル・データベースまたはユーザー・データベース (User Relational Database,or User Database)
独立した補助記憶域プールには保管されていないデータベース・オブジェクトとともに、独立した単一の補助記憶域プールに存在するすべてのデータベース・オブジェクト。**注:** V5R2 では、 iSeries サー

バーは、独立補助記憶域プールがサーバー上で構成されている場合、複数のリレーショナル・データベースに対するホストにすることができる。システム・リレーショナル・データベースは必ず 1 つ存在し、ユーザー・リレーショナル・データベースは 1 つ以上存在することができる。各ユーザー・データベースには、システム・データベース内のすべてのオブジェクトが入っている。**注:** ただし、システム・データベースは、コミットメント制御の観点では別のデータベースとして扱われ、 SQL の観点でもユーザー・データベースに組み込まれていると見なされることを承知しておく必要がある。

［ラ行］

リモート作業単位 (RUW) (remote unit of work (RUW))
リモート作業単位 (RUW) とは、分散リレーショナル・データベース処理の一方式であり、アプリケーション・プログラムは、1 作業単位内でリモート・データベース上のデータにアクセスできる。 1 つのリモート作業単位には、複数のリレーショナル・データベース要求を含めることが可能だが、すべての要求を同じリモート・データベースに対して発行する必要がある。

リモート・リレーショナル・データベースまたはリモート・データベース (Remote Relational Database, or Remote Database)
iSeries または別のサーバー上に存在する、リモート・アクセス可能なデータベース。

リレーショナル・データベース (relational database)
テーブルのセットと見なすことができ、データのリレーショナル・モデルに従って操作できるデータベース。ユーザーが iSeries サーバーからアクセスできるリレーショナル・データベースには、システム・リレーショナル・データベース (システム・データベース)、ユーザー・リレーショナル・データベース (ユーザー・データベース)、およびリモート・リレーショナル・データベース (リモート・データベース) という 3 つのタイプがある。

ロールバック (roll back)
作業単位サポートにより、アプリケーション・プログラムは、作業単位への変更をロールバックすることも可能。作業単位をロールバックすると、最後のコミットあるいはロールバック操作後に加えられた変更は適用されない。このように、アプリケーション・プログラムは、データベースに対する一連の要求を 1 単位として扱う。

ロケーション (location)
ロケーションとは、分散リレーショナル・データベースに関与するリレーショナル・データベース管理システムの相互接続ネットワークの中にある、特定のリレーショナル・データベース管理システムのこと。この意味での「ロケーション」は、独立 ASP グループで構成されたシステム内のユーザー・データベースを指すこともある。

論理装置 (LU) (logical unit (LU))
SNA において、ユーザーが通信ネットワークにアクセスする際に経由するポートとして機能する、3 種類のネットワーク・アドレス可能単位の 1 つ。残りの 2 つには、物理装置 (PU) とシステム・サービス制御点 (SSCP) が含まれる。

D

DB2 Query Manager
DB2 Query Manager and SQL Development Kit ライセンス・プログラム、つまり、iSeries データベースからの情報を取得するのに使用されるツールの集まりの一部。 DB2 Query Manager を使用して、QUERY 定義を作成したり、新規または既存の QUERY 定義を実行したり、QUERY 情報を形式設定したりすることができる。

DB2 Query Manager and SQL Development Kit
IBM ライセンス・プログラムであり、DB2 UDB ファミリーの製品の 1 つ。ユーザーは、Query Manager を使用して SQL 照会と報告書を作成できる。プログラマーは、SQL Development Kit を使用して、SQL アプリケーションを開発できる。

DB2 Universal Database for iSeries (DB2 UDB for iSeries)
iSeries における統合リレーショナル・データベース・マネージャー。データへのアクセスとデータの保護を可能にする。参照保全や並列データベース処理などの拡張機能も提供する。

DB2 Universal Driver for SQLJ and JDBC
分散およびローカル DB2 アクセスのための、アーキテクチャーに中立な JDBC ドライバー。

I

IP セキュリティー・アーキテクチャー (IPSec) (IP Security Architecture (IPSec))
Internet Engineering Task Force (IETF) 規格の集合で、さまざまなセキュリティー・サービスを使用して IP トラフィックを保護するために、インターネット・プロトコル (IP) 層のアーキテクチャーを定義する。

K

Kerberos
マサチューセッツ工科大学 (MIT) の Project Athena のセキュリティー・システムを指す。対称鍵暗号を使用して、ネットワーク内のユーザーにセキュリティー・サービスを提供する。

Kerberos 構成ファイル (krb5.conf) (Kerberos configuration file (krb5.conf))
Kerberos レルム名が DNS 接尾部名と異なる場合には、正確なレルムにマップする必要がある。これを行うには、Kerberos 構成ファイル (krb5.conf) 内に、各リモート・ホスト名を正しいレルム名にマップする項目がなければならない。

Kerberos 領域 (Kerberos Realm)
Kerberos 領域とは、 Kerberos サーバー内で登録される Kerberos プリンシパルのセットのこと。

R

RDB 別名 (RDB Alias)
AR 上で関連付けられたリレーショナル・データベース名。便宜上の目的で、実際のリレーショナル・データベース名の代わりに使用できる。

S

Secure Sockets Layer (SSL)
Netscape Communications Corp. および RSA Data Security, Inc. によって開発された普及しているセキュリティー機構。 SSL を使用すれば、クライアントはサーバーを認証でき、すべてのデータと要求を暗号化することができる。 SSL によって保護されたセキュア・サーバーの URL は、http ではなく https から始まる。

SNA アップライン機能 (SNUF) (SNA upline facility (SNUF))
iSeries サーバーが、ホスト・システム上の CICS/VS および IMS/VS アプリケーション・プログラムと通信できるようにする通信サポート。たとえば、DHCF は HCF と通信し、 DSNX は NetView 分散管理プログラムと通信する。

SNA 配布サービス (SNADS) (SNA distribution services (SNADS))
システムのネットワーク内で電子メールの受信、経路指定、および送信を行うための一連の規則を定義する IBM 非同期配布サービス。

SQL 診断域 (SQL diagnostic area)
データベース・マネージャーによって提供されたスペースにある、情報の集まり。最後の実行された SQL ステートメントに関する情報で更新される。

SQL パッケージ (SQL package)
SQL パッケージとは、分散リレーショナル・データベースでのみ使用される iSeries オブジェクトのこと。 SQL のプリコンパイル処理の結果として作成するか、またはコンパイル済みプログラム・オブジェクトから作成することができる。 SQL パッケージは、アプリケーション・サーバー側に存在する。このパッケージには、SQL ステートメント、ホスト変数属性、およびアプリケーション・サーバーがアプリケーション・リクエスターの要求を処理するのに使うアクセス・プランが入っている。

付録 E. 関連情報

この参考文献の項では、本書に関連して IBM から提供されている資料を、4 種類に分けて説明しています。その分類は次のとおりです。

- iSeries server ライブラリー
- 分散リレーショナル・データベース・ライブラリー
- 他の IBM 分散リレーショナル・データベースのプラットフォーム・ライブラリー
- アーキテクチャー資料
- IBM レッドブック™

iSeries server 情報

以下の iSeries 関連資料には、必要になる可能性のある情報が記載されています。ご注文と参照の際に役立つよう、オーダー番号が付記されています。

- *DSNX Support*  。リモート管理サポート (分散ホスト・コマンド機能)、変更管理サポート (分散システム・ノード管理機能)、および問題管理サポート (警報) を使えるよう iSeries server を構成する方法の詳細が述べられています。
- iSeries Information Center の『バックアップおよび回復』。システム・データを保管および復元するのに使用することができるさまざまなメディアについて、またデータベース・ファイルに加えられた変更の記録方法と、その情報をシステム回復と活動報告書に利用する方法についての詳細を、システム・プログラマーを対象に説明しています。
- iSeries Information Center の『制御言語 (CL)』。オブジェクトおよびライブラリーの一般説明、制御言語 (CL) プログラミング、プログラム相互の流れの制御と通信、CL プログラムにおけるオブジェクトの処理、および CL プログラムの作成などの広範囲にわたるプログラミングの解説を述べています。他に、事前定義メッセージと即時メッセージ、およびメッセージ処理法について、さらに、デバッグ・モード、ブレークポイント、トレース、および表示機能も含め、ユーザー定義のコマンドとメニューの定義と作成、およびアプリケーションのテストの方法についても述べています。

- *Communications Management*  。これには、通信状況の処理に関する詳細、通信関連の作業の管理に関する項、通信エラー、パフォーマンス、回線速度とサブシステム記憶域についての解説があります。
- iSeries Information Center の『分散データ管理 (Distributed Database Management)』。アプリケーション・プログラマーを対象に、リモート・ファイル処理に関する情報を提供します。 OS/400 分散データ管理機能 (DDM) に対するリモート・ファイルの定義方法、 DDM ファイルの作成方法、DDM でサポートされるファイル・ユーティリティー、および他のシステムとの関連における OS/400 DDM の要件を説明しています。

- *Local Device Configuration*  。システム・オペレーターまたはシステム管理者を対象に、初期ローカル・ハードウェアの構成方法とその構成の変更方法について説明しています。また、装置構成の概念情報と、9406、9404、および 9402 システム装置での装置構成の計画情報も記載しています。

- *ADTS/400: Data File Utility*  。V5R1 Supplemental Manuals Web サイト上で、アプリケーション・プログラマー、プログラマー、またはヘルプ・デスク担当者を対象に、アプリケーション開発ツール、デ

ータ・ファイル・ユーティリティー (DFU) について説明し、データをファイルへ入力し、ファイルを更新し、ファイル内を照会し、 DFU プログラムを実行するためのプログラムの作成法を示しています。またこの資料では、ワークステーション・オペレーター向けに、 DFU を習得するための活動と教材が用意されています。

- *SNA Distribution Services*  。これは、V5R1 Supplemental Manuals Web サイト上で、システム・プログラマーまたはネットワーク管理者を対象に、配布サービス (SNADS) および仮想計算機/多重仮想記憶域 (VM/MVS) ブリッジ用の通信ネットワークの構成について説明しています。さらに、オブジェクト配布機能、文書ライブラリー・サービス、およびシステム配布ディレクトリー・サービスについても述べています。

- *ICF Programming*  。アプリケーション・プログラマーを対象に、 iSeries 通信および ICF ファイルを使用するアプリケーション・プログラムの作成に必要な情報が記載されています。また、データ記述仕様 (DDS) キーワード、システム提供形式、戻りコード、ファイル転送サポート、およびプログラミング例も示されています。

- *LAN, Frame-Relay and ATM Support*  。トークンリング・ネットワーク、イーサネット・ネットワーク、またはブリッジ・ネットワーク環境での iSeries server の使用法について説明しています。

- *Remote Work Station Support*  。アプリケーション・プログラマーまたはシステム・プログラマーを対象に、構成コマンドについての情報と、回線、コントローラー、および装置の定義についての情報を提供します。
- iSeries Information Center の『Query Management Programming (Query 管理機能 プログラミング)』。この資料は、アプリケーション・プログラマーを対象に、報告書用に照会するデータベース・ファイルの判別方法、構造化照会言語 (SQL) の Query 定義の定義方法、および照会管理コマンドを使用するプロシージャーの使用と作成の方法を説明しています。この資料では、照会グローバル変数サポートの使用法について、また、OS/400 の Query 管理機能と iSeries Query ライセンス・プログラムとの関係についても解説されています。

- *Remote Work Station Support*  。ディスプレイ・パススルー、分散ホスト・コマンド機能、および 3270 リモート接続機構などのリモート・ワークステーション・サポートのセットアップと使用の方法について説明しています。
- iSeries Information Center の『セキュリティー』。システム・プログラマー (またはセキュリティーの責任担当者) を対象に、システム・セキュリティーの概念、セキュリティーの計画、およびシステムでのセキュリティーのセットアップについて詳述しています。
- iSeries Information Center の『SQL プログラミングの概念 (SQL Programming Concepts)』。アプリケーション・プログラマー、プログラマー、またはデータベース管理担当者を対象に、 SQL ステートメントの設計、作成、実行、およびテストの方法の概要を述べています。さらに、対話式構造化照会言語 (SQL) についても説明しています。
- iSeries Information Center の『SQL リファレンス (SQL Reference)』。アプリケーション・プログラマー、プログラマー、またはデータベース管理担当者を対象に、 SQL ステートメントおよびそのパラメーターに関して詳述しています。

- *X.25 Network Support*  。 X.25 ネットワークでの iSeries server の使用法を説明しています。

分散リレーショナル・データベース・ライブラリー

以下の資料は、IBM Distributed Relational Database Architecture の実装について、その背景と一般的なサポート情報を提供します。

- *DRDA: Every Manager's Guide* (GC26-3195)。分散リレーショナル・データベースと配布ファイルについての簡潔でしかもハイレベルな教材としてお使いいただけます。この資料では、分散データ・システムの開発に対する IBM のサポートについて述べ、分散データ用の現在の IBM 製品と発表済みサポートについて説明しています。この資料の内容は、管理職、管理者、および技術担当者が、分散データの概念を理解したいときに参考にしていただくためのものです。
- *DRDA: 分散リレーショナル・データベースの計画* (N: SC26-4650)。分散リレーショナル・データの計画に役立ちます。段階的なステップ、決定事項、および決定にあたって選択するオプションについて説明しています。また、現在入手可能であるか、またはすでに発表済みの、分散リレーショナル・データベース製品とその機能についても述べており、将来における分散リレーショナル・データのサポートに関する弊社の方針を説明しています。この資料の内容は、計画担当者を対象としています。
- *DRDA: 接続の手引き* (SC88-7070)。 Distributed Relational Database Architecture をサポートする IBM 製品の相互接続の方法を説明しています。分散リレーショナル・データベースとネットワーク・システムに関連した概念と用語について説明しています。また、分散環境内の異種システムの接続方法について述べています。「接続の手引き」の記載事項は、どの製品資料にも含まれていません。この資料の内容は、システム管理者、データベース管理者、通信管理者、およびシステム・プログラマーを対象としています。
- *DRDA: 適用業務プログラミングの手引き* (N: SC26-4773)。 IBM リレーショナル・データベース管理システムにアクセスするアプリケーション・プログラムの設計、作成、および変更の方法を説明しています。この資料では、異種環境用の分散リレーショナル・データベース・アプリケーションを作成する場合に、プログラマーがそれぞれに応じて変えなければならない点に焦点が置かれています。記載内容には、プログラムの設計、準備、および実行に加え、パフォーマンス上の考慮事項も含まれています。IBM C で作成されたプログラミング例も入っています。この資料の情報は、IBM 高水準言語のうちの少なくとも 1 つと、構造化照会言語 (SQL) を使って作業するアプリケーション・プログラマーを対象とします。
- *DRDA: 問題判別の手引き* (N: SC26-4782)。分散リレーショナル・データベース環境において問題の原因を究明するのに役立ちます。この資料は、各製品をよく知らない人を対象に、それぞれの製品の紹介を記載し、各製品ごとに問題の診断と報告の方法を詳述しています。この手引きは、各ホスト・システムごとに固有の手順およびツールと、各種のシステムに共通の手順およびツールについて説明しています。この資料の情報は、分散リレーショナル・データベースの問題を IBM サポート・センターに報告する担当者を対象とします。
- *IBM SQL Reference, Volume 2* (SC26-8416)。 DRDA を参照して、以下の機能を比較しています。
  - IBM SQL リレーショナル・データベース製品
  - IBM SQL
  - ISO-ANSI SQL (SQL92E)
  - X/Open SQL (XPG4-SQL)
  - ISO-ANSI SQL 呼び出しレベル・インターフェース (CLI)
  - X/Open CLI
  - Microsoft オープン・データベース・コネクティビティー (ODBC) バージョン 2.0

他の IBM 分散リレーショナル・データベースのプラットフォーム・ライブラリー

DB2 Connect および DB2 Universal Database

DB2 Connect および Universal Database で作業している場合で、さらに情報が必要な場合は、 Web ページ 『Knowledge Base: DB2 Universal Database and DB2 Connect for Windows, OS/2, UNIX』を参照してください。このページには、以下の資料と、各バージョンに固有の情報があります (ただし、マニュアルで説明されているすべての機能が、すべてのバージョンでサポートされているわけではありません)。

- *DB2 Connect Enterprise Edition Quick Beginning*
- *DB2 Connect Personal Edition Quick Beginning*
- *DB2 Connect User's Guide*
- *DB2 UDB for OS/2 Quick Beginnings*
- *DB2 UDB for UNIX Quick Beginnings*
- *DB2 UDB for Windows NT® Quick Beginnings*
- *DB2 UDB Personal Edition Quick Beginnings*
- *DB2 UDB SQL Getting Started*
- *DB2 UDB Administration Guide*
- *DB2 UDB SQL Reference*
- *DB2 UDB Command Reference*
- *DB2 UDB Messages Reference*
- *DB2 UDB Troubleshooting Guide*

DB2 for OS/390® and z/OS

DB2 for OS/390 and z/OS で作業している場合で、さらに情報が必要な場合は、Web ページ DB2 for OS/390 and z/OS を参照してください。このページには、以下の資料と、各バージョンに固有の情報があります (ただし、マニュアルで説明されているすべての機能が、すべてのバージョンでサポートされているわけではありません)。

- *DB2 for OS/390 Command Reference*
- *DB2 for OS/390 Reference for Remote DRDA*
- *DB2 for OS/390 SQL Reference*
- *DB2 for OS/390 Utility Guide and Reference*
- *DB2 for OS/390 Messages and Codes*

DB2 Server for VSE & VM

DB2 Server for VSE & VM で作業している場合で、さらに情報が必要な場合は、 Web ページ DB2 Server for VSE & VM を参照してください。このページには、以下の資料と、各バージョンに固有の情報があります (ただし、マニュアルで説明されているすべての機能が、すべてのバージョンでサポートされているわけではありません)。

- *SBOF for DB2 Server for VM*
- *DB2 and Data Tools for VSE and VM*
- *DB2 Server for VM Database Administration*
- *DB2 Server for VM Application Programming*

- *DB2 Server for VM Database Services Utilities*
- *DB2 Server for VM Messages and Codes*
- *DB2 Server for VM Master Index and Glossary*
- *DB2 Server for VM Operation*
- *DB2 Server for VSE & VM Quick Reference*
- *DB2 Server for VSE & VM SQL Reference*
- *DB2 Server for VM System Administration*
- *DB2 Server for VM Diagnosis Guide*
- *DB2 Server for VM Interactive SQL Guide*
- *DB2 Server Data Spaces Support for VM/ESA®*
- *DB2 Server for VSE & VM LPS*
- *DB2 Server for VSE & VM Data Restore*
- *DB2 for VM Control Center Installation*
- *DB2 Server for VM/VSE Training Brochure*

アーキテクチャー資料

- *Character Data Representative Architecture: Overview* (GC09-2207)
- *Character Data Representative Architecture: Details* (SC09-2190)

  この資料には CD-ROM が添付されています。それには、オンライン・ブック形式の CDRA 資料、バイナリー・フォームの変換テーブル、多数の変換バイナリー用のマッピング・ソース、コード・ページと文字セット・リソースの集合体、 IBM で使用されている文字命名情報が入っています。また CD には、付属媒体で使用するための表示ユーティリティーも入っています。ビューアーは、OS/2、Windows 3.1、および Windows 95 でご利用いただけます。

- *DRDA Vol. 1: Distributed Relational Database Architecture (DRDA)*

  このテクニカル資料は、 Distributed Relational Database Architecture の仕様を文書化した 3 つの資料のうちの 1 つです。この巻では、アプリケーション・プログラムが分散リレーショナル・データにアクセスするのに使用できるリレーショナル・データベースの管理機能の相互接続について説明しています。分散環境におけるアプリケーションとリレーショナル・データベース管理システムとの間に必要な接続、関係者の責務と、いつやりとりを行う必要があるか、また、分散リレーショナル・データベース管理システムの処理に必要なフォーマットとプロトコルについて述べています。分散データベース管理システムの処理用の API は説明されていません。この資料は、Web 上の http://www.opengroup.org/dbiop/index.htm で利用できます。

- *DRDA Vol. 2: Formatted Data Object Content Architecture (FD:OCA)*

  このテクニカル資料は、 Distributed Relational Database Architecture の仕様を文書化した 3 つの資料のうちの 1 つです。この巻では、定様式データ・オブジェクト内容体系 (FD:OCA) を構成する機能とサービスについて説明しています。この体系を使って、別々のデータ・タイプとデータ表示方式を利用するそれぞれの環境相互の接続のギャップを埋めることができます。 FD:OCA は、DRDA に組み込まれています。この資料は、Web 上の http://www.opengroup.org/dbiop/index.htm で利用できます。

- *DRDA Vol. 3: Distributed Data Management (DDM) Architecture*

  このテクニカル資料は、 Distributed Relational Database Architecture の仕様を文書化した 3 つの資料のうちの 1 つです。この巻では、DDM データ・ストリームの体系化コマンド、パラメーター、オブジェ

クト、およびメッセージについて説明しています。このデータ・ストリームは、さまざまな DDM モデルが互いにデータを交換するのに使われます。この資料は、Web 上の http://www.opengroup.org/dbiop/index.htm で利用できます。

レッドブック

- *Distributed Relational Database: Using DDCS/6000 DRDA Support with DB2 and DB2/400* (GG24-4155)
- *Setup and Usage of SQL/DS in a DRDA Environment* (GG24-3733)
- *DRDA Client/Server Application Scenarios* (GG24-4193)
- *DRDA Client/Server for VM &VSE Setup* (GG24-4275)
- *DATABASE 2/400 Advanced Database Functions* (GG24-4249)
- *Distributed Relational Database Cross Platform Connectivity and Application* (GG24-4311)
- *Getting Started with DB2 Stored Procedures: Give Them a Call through the Network* (GG24-4693)
- *WOW! DRDA Supports TCP/IP: DB2 Server for OS/390 (SG24-2212)*

付録 F. 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あり、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。

〒106-0032
東京都港区六本木 3-2-31
IBM World Trade Asia Corporation
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
Software Interoperability Coordinator
3605 Highway 52 N
Rochester, MN 55901-7829
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 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。

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。お客様は、IBM 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。

商標

以下は、IBM Corporation の商標です。

Advanced Peer-to-Peer Networking
AIX
AnyNet
C/400
CICS
COBOL/400
DataJoiner
DataPropagator
DB2
DB2 Connect
DB2 Universal Database
Distributed Relational Database Architecture
DRDA
DXT
Extended Services
FFST
IBM
IMS
Informix
iSeries
Language Environment
MVS
NetView
Operating System/400
OS/2
OS/390
OS/400
PS/2
Redbooks
RPG/400
RS/6000
SQL/DS

System/390
VM/ESA
z/OS

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または登録商標です。

Microsoft、Windows、Windows NT および Windows ロゴは、Microsoft Corporation の米国およびその他の国における商標です。

Intel、Intel Inside (ロゴ)、MMX および Pentium は、Intel 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。

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。

コードの特記事項情報

本書には、プログラミングの例が含まれています。

IBM は、お客様に、すべてのプログラム・コードのサンプルを使用することができる非独占的な著作使用権を許諾します。お客様は、このサンプル・コードから、お客様独自の特別のニーズに合わせた類似のプログラムを作成することができます。

すべてのサンプル・コードは、例として示す目的でのみ、IBM により提供されます。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。

ここに含まれるすべてのプログラムは、現存するままの状態で提供され、いかなる保証も適用されません。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の保証の適用も一切ありません。

資料に関するご使用条件

お客様がダウンロードされる資料につきましては、以下の条件にお客様が同意されることを条件にその使用が認められます。

個人使用: これらの資料は、すべての著作権表示その他の所有権表示をしていただくことを条件に、非商業的な個人による使用目的に限り複製することができます。ただし、IBM の明示的な承諾をえずに、これらの資料またはその一部について、二次的著作物を作成したり、配布 (頒布、送信を含む) または表示 (上映を含む) することはできません。

商業的使用: これらの資料は、すべての著作権表示その他の所有権表示をしていただくことを条件に、お客様の企業内に限り、複製、配布、および表示することができます。ただし、IBM の明示的な承諾をえずにこれらの資料の二次的著作物を作成したり、お客様の企業外で資料またはその一部を複製、配布、または表示することはできません。

ここで明示的に許可されているもの以外に、資料や資料内に含まれる情報、データ、ソフトウェア、またはその他の知的所有権に対するいかなる許可、ライセンス、または権利を明示的にも黙示的にも付与するものではありません。

資料の使用が IBM の利益を損なうと判断された場合や、上記の条件が適切に守られていないと判断された場合、IBM はいつでも自らの判断により、ここで与えた許可を撤回できるものとさせていただきます。

お客様がこの情報をダウンロード、輸出、または再輸出する際には、米国のすべての輸出入関連法規を含む、すべての関連法規を遵守するものとします。IBM は、これらの資料の内容についていかなる保証もしません。これらの資料は、特定物として現存するままの状態で提供され、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含むすべての明示もしくは黙示の保証責任なしで提供されます。

これらの資料の著作権はすべて、IBM Corporation に帰属しています。

お客様が、このサイトから資料をダウンロードまたは印刷することにより、これらの条件に同意されたものとさせていただきます。

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]

A

C

D

E

F

I

K

L

N

Q

R

S

T

V

W

Printed in Japan

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