IBM® DB2® ユニバーサル・データベース

管理の手引き: インプリメンテーション

バージョン 7

SC88-8511-01
(英文原典：SC09-2944-01)

IBM® DB2® ユニバーサル・データベース

管理の手引き: インプリメンテーション

バージョン 7

SC88-8511-01
(英文原典：SC09-2944-01)

ご注意!

本書、および本書がサポートする製品をご使用になる前に、 493ページの『付録M. 特記事項』にある一般的な情報を必ずお読みください。

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

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)

原 典:	SC09-2944-01 IBM® DB2® Universal Database Administration Guide: Implementation Version 7
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 2001.8

この文書では、平成明朝体™W3、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™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

本書について

3 巻で構成される本書には、 2000 年問題に対応した DB2* リレーショナル・データベース管理システム (RDBMS) 製品を使用し管理するために必要な、以下の情報が収められています。

- データベース設計についての情報 (*管理の手引き: 計画*)
- データベースの使用および管理についての情報 (*管理の手引き: インプリメンテーション*)
- パフォーマンスを向上させるための、データベース環境の構成およびチューニングについての情報 (*管理の手引き: パフォーマンス*)

本書に記載されているタスクの多くは、以下に示すさまざまなインターフェースを使用して実行することができます。

- **コマンド・ライン・プロセッサー**。グラフィカル・インターフェースから、データベースをアクセスし操作することができます。このインターフェースから、SQL ステートメントおよび DB2 ユーティリティー関数も実行することができます。本書にある例のほとんどが、このインターフェースを使った場合を例として取り上げています。コマンド・ライン・プロセッサーの使用に関する詳細は、*コマンド解説書* を参照してください。
- **アプリケーション・プログラミング・インターフェース**。アプリケーション・プログラム内で DB2 ユーティリティー関数を実行することができます。アプリケーション・プログラミング・インターフェースの使用についての詳細は、 *管理 API 解説書* を参照してください。
- **コントロール・センター**。システムの構成、ディレクトリーの管理、システムのバックアップと回復、ジョブのスケジューリング、および媒体の管理などの管理タスクをグラフィックを使用して実行することができます。またコントロール・センターには、システム間のデータの複製をグラフィックを使用してセットアップするための**複製管理**が含まれています。さらに、コントロール・センターでは、グラフィカル・ユーザー・インターフェースを介して DB2 ユーティリティー機能を実行できます。コントロール・センターを呼び出す方法は、プラットフォームによって異なります。たとえば、 OS/2 ではコマンド行で `db2cc` コマンドを使用し、 Windows プラットフォームでは DB2 フォルダーからコントロール・センター アイコンを選択するか、開始パネルを使用します。紹介のヘルプを表示するには、「コントロール・センター (Control Center)」ウィンドウの「**ヘルプ (Help)**」プルダウンから「**入門 (Getting started)**」を選択してください。 **Visual Explain** および**パフォーマンス測定プログラム**のツールは、コントロール・センターから呼び出されます。

他にも、管理タスクを行うために使用できるツールがあります。それらのツールには、以下のものがあります。

- **スクリプト・センター**は、スクリプトと呼ばれる小さなアプリケーションを保管します。これらのスクリプトには、オペレーティング・システムのコマンドだけでなく、SQL ステートメントや DB2 コマンドを含めることができます。
- **アラート・センター**は、他の DB2 操作から出されるメッセージをモニターします。
- **ツール設定**は、コントロール・センター、アラート・センター、および複製についての設定値を変更します。
- **ジャーナル**は、自動で実行するジョブをスケジュールします。
- **データウェアハウスセンター**は、ウェアハウス・オブジェクトを管理します。

本書の対象読者

本書は、ローカルまたはリモート・クライアントがアクセスするデータベースを設計、使用、および維持する必要のあるデータベース管理担当者、システム管理者、機密保護管理者、およびシステム操作員を対象としています。 DB2 リレーショナル・データベース管理システムの管理および操作について理解しておくことが必要なプログラマーや、その他のユーザーも本書をご使用になれます。

本書の構成

本書には、以下の主な項目に関する情報が記載されています。

コントロール・センターによる管理

- 『第1章 GUI ツールを使用した DB2 の管理』では、データベースを管理するのに使用されるグラフィカル・ユーザー・インターフェース (GUI) ツールを紹介します。

設計のインプリメント

- 『第2章 データベースを作成する前に』では、データベースの作成およびデータベース内のオブジェクトの作成の前に必要な前提条件について説明します。
- 『第3章 データベースの作成』では、データベースの作成およびデータベース内のオブジェクトの作成に関連する作業について説明します。
- 『第4章 データベースの変更』では、データベースの作成およびデータベース内のオブジェクトの代替または除去に関連する前提条件と作業について説明します。

データベースの機密保護

- 『第5章 データベース・アクセスの制御』では、データベース・リソースへのアクセスを制御する方法について説明します。
- 『第6章 DB2 アクティビティーの監査』では、データに対する不要なアクセスまたは予期せぬアクセスを検出してモニターする方法について説明します。

データの移動

- 『第7章 データの移動に使用するユーティリティー』では、ロード、オートローダー、インポートおよびエクスポート・ユーティリティーについて説明します。

回復

- 『第8章 データベースのリカバリー』では、データベースおよび表スペースの回復方式を選択する場合の考慮事項について説明します。回復タスクには、データベースおよび表スペースのバックアップおよび回復、およびロールフォワード回復方式の使用が含まれます。

付録

- 『付録A. 命名規則』では、データベースおよびオブジェクトを命名する際に従わなければならない規則を示します。
- 『付録B. 分散コンピューティング環境 (DCE) ディレクトリー・サービスの使用』では、 DCE ディレクトリー・サービスを使用する方法について説明します。
- 『付録C. データベース・リカバリー用のユーザー出口』では、ユーザー出口プログラムでデータベース・ログ・ファイルを使用する方法について説明し、いくつかのサンプル・ユーザー出口プログラムを示します。
- 『付録D. 複数のデータベース区画に対するコマンドの発行』では、コマンドを区分データベース環境内のすべての区画に送るための *db2_all* および *rah* シェル・スクリプトの使用法について説明します。
- 『付録E. DB2 (Windows NT 版) が Windows NT 機密保護を処理する方法』では、DB2 が Windows NT 機密保護を処理する方法について説明します。
- 『付録F. Windows NT パフォーマンス・モニターの使用』では、 DB2 (Windows NT 版) ユーザーが使用できる 2 つのパフォーマンス・モニター (DB2 パフォーマンス・モニターおよび Windows NT パフォーマンス・モニター) について説明します。
- 『付録G. Windows NT または Windows 2000 データベース区画サーバーを使った作業』では、 Windows NT と Windows 2000 で区分データベース・サーバーを処理するために使用されるユーティリティーについて説明します。
- 『付録H. 複数の論理ノードの構成』では、区分データベース環境で複数論理ノードを構成する方法について説明します。
- 『付録I. 高速ノード間通信』では、 Windows NT 環境で、仮想インターフェース・アーキテクチャーを DB2 エンタープライズ拡張エディションで使用できるようにする方法について説明します。
- 『付録J. Lightweight Directory Access Protocol (LDAP) ディレクトリー・サービス』では、 LDAP ディレクトリー・サービスを使用する方法について説明します。

- 『付録K. コントロール・センターの拡張』では、新しいアクションを割り当てた新しいツールバー・ボタンの追加、新しいオブジェクト定義の追加、および新しいアクション定義の追加により、コントロール・センターを拡張する方法について説明します。
- 『付録L. DB2 ライブラリーの使用法』では、ウィザード、オンライン・ヘルプ、メッセージ、およびマニュアルを含む DB2 ライブラリーの構造について説明します。

管理の手引きの他の巻の概要

管理の手引き: 計画

管理の手引き: 計画 は、データベース設計を扱っています。論理設計および物理設計、分散トランザクション、および高可用性について説明しています。この巻のそれぞれの章と付録は、以下のように構成されています。

DB2 ユニバーサル・データベースの世界

- 『DB2 ユニバーサル・データベースの管理』では、 DB2 ユニバーサル・データベースを紹介し、概要を説明します。

データベースの概念

- 『リレーショナル・データベースの基本的な概念』では、回復オブジェクト、記憶オブジェクト、およびシステム・オブジェクトを含むデータベース・オブジェクトの概要を説明します。
- 『連合システム』では、連合システム (federated database system: 複数のデータベースから構成されるが、単一のデータベース・イメージを提供するデータベース・システムを意味します) について説明します。連合システムはデータベース管理システム (DBMS) の一種で、アプリケーションやユーザーが 1 つのステートメント内で 2 つ以上の DBMS またはデータベースを参照する SQL ステートメントを実行依頼できます。
- 『並列データベース・システム』では、 DB2 で実現される並行性のタイプを紹介します。
- 『データウェア・ハウジングについて』では、データウェアハウジングおよびウェアハウジング・タスクについて概説します。
- 『地理情報エクステンダーについて』では、地理情報エクステンダーを紹介し、この目的とこれが処理するデータについて説明します。

データベースの設計

- 『論理データベースの設計』では、論理データベースの設計に関する概念と指針について説明します。
- 『物理データベースの設計』では、物理データベースの設計 (データ記憶域に関する考慮事項を含む) の指針について説明します。

分散トランザクション処理

- 『分散データベースの設計』では、単一トランザクションで複数のデータベースをアクセスする方法を説明します。
- 『トランザクション・マネージャーの設計』では、 CICS などの分散トランザクション処理環境でデータベースを使用する方法について説明します。

高可用性システム

- 『高可用性とフェールオーバー・サポートの紹介』では、 DB2 が提供する高可用フェイルオーバー・サポートの概要を説明します。

付録

- 『データベース移行の計画』では、第 7 版へのデータベースの移行について説明します。
- 『リリースからリリースへの非互換性』では、第 7 版までに導入された、互換性のない機能について説明します。
- 『各国語サポート (NLS)』では、国、言語、およびコード・ページの情報を含む、DB2 各国語サポート (NLS) について説明します。

管理の手引き: パフォーマンス

管理の手引き: パフォーマンス では、パフォーマンスに関する問題、つまり、アプリケーションや DB2 ユニバーサル・データベース製品のパフォーマンスの設定、テスト、および改善に関連したトピックや問題を扱います。この巻のそれぞれの章と付録は、以下のように構成されています。

パフォーマンスの紹介

- 『パフォーマンスの要素』では、 DB2 UDB パフォーマンスを管理と改善に関する概念と考慮事項について紹介します。
- 『構造と処理の概要』では、基礎となる DB2 ユニバーサル・データベースの構造およびプロセスを紹介します。

アプリケーション・パフォーマンスのチューニング

- 『アプリケーションについての考慮事項』では、アプリケーションの設計時点で、データベースのパフォーマンスを向上させる手法をいくつか説明します。
- 『環境についての考慮事項』では、データベース環境の設定時点で、データベースのパフォーマンスを向上させる手法をいくつか説明します。
- 『システム・カタログ統計』では、最適なパフォーマンスを達成するために、データについての統計を収集し使用する方法について説明します。
- 『SQL コンパイラーに関する解説』では、 SQL コンパイラーを使用してコンパイルしたときに、 SQL ステートメントがどうなるかについて説明します。

- 『SQL Explain 機能』では、Explain 機能について説明します。この機能により、データにアクセスするために SQL コンパイラーが行った選択を調べることができます。

システムのチューニングと構成

- 『操作上のパフォーマンス』では、データベース・マネージャーがメモリーを使用する方法の概要、および実行時のパフォーマンスに影響を与えるその他の考慮事項について説明します。
- 『管理プログラムの使用法』では、データベース管理のある局面を制御するための管理プログラムの使用について説明します。
- 『構成のスケーリング』では、データベース・システムのサイズの増加に関連する考慮事項と作業について説明します。
- 『データベース区画間でのデータの再配分』では、区画間でデータを再配分するために、区分データベース環境において必要な作業について説明します。
- 『ベンチマーク・テスト』では、ベンチマーク・テストの概要とベンチマーク・テストの実行方法について説明します。
- 『DB2 の構成』では、データベース・マネージャー、データベース構成ファイルおよび構成パラメーターの値について説明します。

付録

- 『DB2 登録変数と環境変数』では、プロファイル・レジストリーの値と環境変数を説明します。
- 『Explain 表と定義』では、 DB2 Explain 機能が使用する表について、およびそれらの表の作成方法について説明します。
- 『SQL Explain ツール』では、 DB2 Explain ツールである db2expln および dynexpln の使用法について説明します。
- 『db2exfmt − Explain 表フォーマット・ツール』では、 DB2 EXPLAIN ツールを使用して Explain 表データをフォーマットする方法について説明します。

第1部 コントロール・センターによる管理

第1章 GUI ツールを使用した DB2 の管理

DB2 ユニバーサル・データベースにはグラフィカル・ユーザー・インターフェース (GUI) があり、コントロール・センターと呼ばれる集中管理点から、容易にローカルおよびリモート・データベースを管理することができます。

この章では、使用可能な DB2 ユニバーサル・データベース管理ツールを概説し、さらに作業を容易に、また効率的に行うためにこのツールを使用する方法を説明します。また、Java コントロール・センターの要約、および独自の Java で使用可能なツールを含めるためにコントロール・センターをカスタマイズする方法についても説明します。

この章では、次の事項についての情報を提供します。

管理ツール

DB2 を管理するツールは、各 DB2 ユニバーサル・データベース製品で選択可能なコンポーネントである、アドミニストレーション・クライアントの一部です。また、アドミニストレーション・クライアントは、 DB2 が使用可能なすべてのオペレーティング・システム用のアドミニストレーション・クライアントを含む、 CD-ROM のセットでも使用可能です。これによって、どんなワークステーションでもアドミニストレーション・クライアントをインストールし、使用することができます。データベースがローカルかリモートか、またデータベース・サーバーが実行しているオペレーティング・システムには関係ありません。ツールを使用すると、コマンド行プロセッサーから実行できるのと同じ機能を、グラフィカル・ユーザー・インターフェースから実行することができます。これらの機能には、DB2 コマンド、SQL ステートメント、またはシステム・コマンドの入力があります。しかし、ツールを使うとき、複雑なステートメントやコマンドを記憶する必要はありません。付加的な援助があります。

注: アドミニストレーション・クライアントはインストール・オプションの 1 つです。

以下のツールをコントロール・センターのツールバーから使用できます。

- コントロール・センター。コントロール・センターは、データベースを管理するメインの DB2 図形処理ツールです。コントロール・センターから、すべてのシステム、およびローカルにカタログ化されるデータベース・オブジェクトの明確な概要をつかむことができます。
- サテライト管理センター。サテライト管理センターで、DB2 サテライト・サーバーを管理することができます。
- コマンド・センター。コマンド・センターによって、DB2 データベース・コマンド、SQL ステートメント、およびオペレーティング・システム・コマンドの発行、前のコマンドの再呼び出し、および SQL 照会のアクセス・プランのスクロールを実行できます。
- スクリプト・センター。スクリプト・センターで、オペレーティング・システムのコマンド、 DB2 コマンド・スクリプト、および SQL コマンド・スクリプトを作成、実行、およびスケジュールできます。
- アラート・センター。アラート・センターは、設定した限界値を超えた場合、またはマルチノード環境のノードが応答しなくなった場合に通知します。
- ジャーナル。ジャーナルによって、ジョブの状況を表示したり、ジョブのスケジュールを変更したり、リカバリー・ヒストリー・ログおよびメッセージ・ログを表示したりできます。
- インフォメーション・センター。インフォメーション・センターでは、DB2 製品マニュアル、およびサンプル・プログラムの情報にすばやくアクセスしたり、 Web 上の DB2 情報の他のソースにアクセスしたりすることができます。
- ライセンス・センター。ライセンス・センターでは、ライセンスの状況を表示し、さらに適切なライセンス・モニター用にシステムを構成することができます。

GUI ツールで実行できるいくつかの機能については、ウィザードの使用のオプションがあります。ウィザードは、コントロール・センターのポップアップ・メニューで起動します。実行中のタスクに必要な情報を入力する方法や、計算方法、提供した情報に基づいた推奨事項について、段階的にプロンプトを出すので、非常に役に立ちます。データベース管理者としてまだ日が浅い方、またはデータベースを頻繁に管理するわけではない方にお勧めします。

DB2 ユニバーサル・データベースには、以下のウィザードがあります。

- データベースのバックアップ。データベース中のデータ、データベースの可用性、および回復可能性要件について基本的な質問をします。その後、バックアップ・プランを提案し、ジョブ・スクリプトを作成し、スケジュールします。データベースのバックアップ・ウィザードを起動するには、バックアップしたいデータベースを表すアイコンを選択し、右マウス・ボタンをクリックしてから、**「バックアップ -> ウィザードを使用するデータベース (Backup —> Database using Wizard)」**を選択します。
- データベースの作成。このウィザードを使うと、データベースを作成し、ストレージを割り当て、基本的なパフォーマンス・オプションを選択できます。データベースの作成ウィザードを起動するには、データベース・アイコンを選択し、右マウス・ボタンをクリックしてから、**「作成 -> ウィザードを使用するデータベース (Create —> Database using Wizard)」**を選択します。
- 表の作成。このウィザードは、事前定義された列テンプレートを使って列を設計したり、表に基本キーを作成したり、1 つ以上の表スペースに表を割り当てたりするのに役立ちます。ウィザードを起動するには、表アイコンを選択し、右マウス・ボタンをクリックしてから、**「作成 -> ウィザードを使用する表 (Create —> Table using Wizard)」**を選択します。
- 表スペースの作成。このウィザードを使用して、新しい表スペースを作成し、基本記憶機構とパフォーマンス・オプションを設定します。ウィザードを起動するには、表スペース・アイコンを選択し、右マウス・ボタンをクリックしてから、**「作成 -> ウィザードを使用する表スペース (Create —> Table space using Wizard)」**を選択します。
- 索引の作成。このウィザードは、SQL ステートメントの指定されたセットで、どの索引を作成または除去するかを決定します。自分で指定する作業負荷を基本にすることをお勧めします。索引作成ウィザードを起動するには、索引フォルダーを選択し、右マウス・ボタンをクリックしてから、**「作成 -> ウィザードを使用する索引 (Create —> Index using Wizard)」**を選択します。
- パフォーマンス構成。このウィザードは、データベース、そのデータ、およびシステムの目的についての情報を要求することによって、データベースを調整するのに役立ちます。データベースおよびインスタンスに新しい構成パラメーターを推奨し、必要ならそれらを自動的に適用します。このウィザードを起動するには、データベースのアイコンを選択し、右マウス・ボタンをクリックしてから、**「ウィザードを使用する構成 (Configure using Wizard)」**を選択します。

- データベースの復元。このウィザードは、データベース回復のプロセスを示します。ウィザードを起動するには、データベースを表すアイコンを選択し、右マウス・ボタンをクリックしてから、**「復元 -> ウィザードを使用するデータベース (Restore —> Database using Wizard)」**を選択します。
- マルチサイト更新の構成。このウィザードによって、アプリケーションが複数のサイトを同時に更新できるように、データベースを構成します。すべてのサイトのデータが一貫していなければならない場合に、このことが重要です。このウィザードを起動するには、インスタンスを選択し、右マウス・ボタンをクリックしてから、**「マルチサイト更新 -> ウィザードを使用する構成 (Multisite Update —> Configure using Wizard)」**を選択します。

注: DB2 (OS/390 版) サブシステムにはウィザードは存在しません。

コントロール・センター・ツールバーから起動できる図形処理ツールの他に、コントロール・センター・ツールバーから直接起動しない、付加的な GUI ツールがあります。

- パフォーマンス・モニター。パフォーマンス・モニターは、インスタンス、データベース、表、表スペース、および接続などの DB2 オブジェクトをモニターするツールです。このツールを使って、パフォーマンス上の問題を検出し、データベースを最適なパフォーマンスに合わせて調整することができます。パフォーマンス・モニターは、コントロール・センターのポップアップ・メニューの選択項目として起動できます。
- イベント・モニター。特定のイベントが発生したときにデータベースの状態を記録することによって、リソースの使用状況を分析するのに役立つツールです。イベント・モニターは、 DB2 コマンド行で `db2emcrt` と入力することによって作成されます。
- イベント・アナライザー。イベント・モニターが収集したデータを分析するためのツールです。イベント・アナライザーは DB2 コマンド行で `db2evmon` と入力すると起動されます。
- Visual Explain。 Visual Explain 機能では、SQL ステートメントのアクセス・プランをグラフとして表示し、パフォーマンスを向上させるために SQL 照会を調整できます。バージョン 6 の前は、アクセス・プランを表示するのに、 Visual Explain ツールを使用していました。現在では、Visual Explain は別個のツールではありません。コントロール・センター、およびコマンド・センターのさまざまなデータベース・オブジェクトにある、ポップアップ・メニューから使用可能です。

これらのツールに加えて、コントロール・センターの一部ではないデータベース管理用の役立つツールがあります。それは、クライアント構成アシスタントです。クライアント構成アシスタントには、リモート・サーバーと通信するクライアントをセットアップするのに役立つウィザードが含まれています。

これらのツールは、すべて後で詳しく説明します。次の節では、ツールの機能を概説します。

共通のツール機能

以下の機能は、いくつかのツールで使用可能です。

- SQL 表示およびコマンド表示
- 関連項目表示
- DDL の生成
- フィルター
- ヘルプ

SQL 表示およびコマンド表示

ツールが SQL ステートメントを生成する場合、ツール・インターフェースで**「SQL 表示 (Show SQL)」**押しボタンが使用できます。同様に、DB2 コマンドを生成するツールでは、**「コマンド表示 (Show Command)」**押しボタンが使用可能です。これらの押しボタンのどちらかをクリックすると、以下のことを行えます。

- グラフィカル・インターフェースで行った選択に基づいてツールが生成する、 SQL ステートメントまたは DB2 コマンドを表示します。この情報によって、インターフェースの働きを理解することができます。
- ステートメントまたはコマンドを将来再利用するためにスクリプトとして保管します。この機能によって、同じステートメントまたはコマンドを再び実行したい場合に、 SQL ステートメントまたは DB2 コマンドを再入力しなくてもすみます。一度 SQL ステートメントまたは DB2 コマンドがスクリプトに保管されると、ステートメントまたはコマンドを再入力しないでスクリプトをスケジュールしたり、スクリプトを編集して変更を加えたり、同様のスクリプトを作成したりできます。

SQL ステートメントまたは DB2 コマンドを表示する方法は次のとおりです。

1. コントロール・センターから、 SQL ステートメントまたは DB2 コマンドを生成するツールを処理するためのウィンドウまたはノートブックに移動します。**「SQL 表示 (Show SQL)」**押しボタンまたは**「コマンド表示 (Show Command)」**押しボタンが使用可能であるとして表示されます。
2. **「SQL 表示 (Show SQL)」**押しボタンまたは**「コマンド表示 (Show Command)」**押しボタンをクリックします。適切なウィンドウが表示されます。

SQL ステートメントおよび DB2 コマンドの保管は、 SQL ステートメントまたは DB2 コマンドが複雑である場合に特に役立ちます。

コマンド表示または **SQL 表示**押しボタンを使用する場合、後で編集可能な新しいスクリプトを作成するか、またはダイアログ・ボックスをクローズして、変更を行う元のダイアログに戻ることができます。 「スクリプトの作成 (Create Script)」押しボタンをクリックすると、「新しいコマンド・スクリプト (New Command Script)」ウィンドウが表示されます。 このウィンドウで、スクリプトを保管する前に、 SQL ステートメントまたは DB2 コマンドを編集できます。

関連項目表示

関連項目表示は、表、索引、視点、別名、トリガー、表スペース、ユーザー定義関数、およびユーザー定義タイプの間の、直接の関係を示します。たとえば、表を選択して関連する視点の表示を選んだ場合、その表に直接基づいている視点が表示されます。関連した視点に基づいた視点は、表から直接作成されていないので表示されません。

関連項目表示オブジェクトは、次のことに役立ちます。

- データベースの構造を理解する。
- 表にすでに存在する索引を判別する。
- 表スペースに保管されているオブジェクトを判別する。
- オブジェクトに関連しており、そのためにとられる処置によって影響を受ける他のオブジェクトを知る。たとえば、従属視点を持つ表を除去したい場合、**関連項目表示**によって、作動不能になる視点が表示されます。

関連項目表示機能の使用法は次のとおりです。

- コントロール・センターで、コンテンツ・ペインからオブジェクトを選択し、右マウス・ボタンをクリックします。
- **「Show Related (関連項目表示)」**を選択します。
- タブをクリックして、必要な関連オブジェクトのページをオープンします。選択したタブによって、さまざまな関連オブジェクトがリストされます。選択したオブジェクトに直接関連するオブジェクトだけが表示されます。

  選択したページの関連オブジェクトで右マウス・ボタンをクリックし、ポップアップ・メニューから「Show Related (関連項目表示)」を選択します。選択されたページが変更され、最新の選択に関連したオブジェクトが表示されます。また、選択されたオブジェクトの隣にある下矢印をクリックして、関係を表示するために以前に選択したオブジェクトのリストを表示することもできます。
- **「Close (クローズ)」**をクリックして、「Show Related (関連項目表示)」をクローズし、コントロール・センターに戻ります。

DDL の生成

DDL の生成機能によって、DDL および SQL ステートメント、および次のものの統計をスクリプト・ファイルで再作成し、保管することができます。

- データベース・オブジェクト
- 許可ステートメント
- 表スペース、ノードグループおよびバッファー・プール
- データベース統計

これによって、以下のことが可能です。

- DDL を保管して、全く同一に定義した表、データベース、および索引を、他のデータベース、たとえば、データベース・ウェアハウス・アプリケーションなどに作成する。
- DDL を使用して、テスト環境から実稼働環境、またはあるシステムから別のシステムに、データベースをコピーする。
- DDL を編集して類似のオブジェクトを作成する。

「DDL の生成 (Generate DDL)」押しボタンをクリックして、 **db2look** と呼ばれるユーティリティーにより生成されるステートメントを含む、「コマンド表示 (Show Command)」ウィンドウを表示します。 「コマンド表示 (Show Command)」ウィンドウから、**「スクリプトの保管 (Save Script)」**押しボタンをクリックして、ステートメントを保管できます。 ステートメントは、スクリプトに入れられます。**「生成 (Generate)」**ボタンをクリックすると、「スクリプトの実行 (Run Script)」ウィンドウがオープンします。

注: System/390 用のコントロール・センターを使用して作業を行っている場合は、DDL ステートメントの生成方法が異なります。具体的な違いについては、ヘルプ情報を参照してください。

DDL ステートメントを、選択したスキーマ、またはデータベース内のすべてのスキーマのどちらに生成するかを選択できます。その後、変更を加えたい場合は、実稼働環境でスクリプトを使用する前に、スクリプトを編集することができます。生成された DDL ステートメントを使用して同一のデータベースを作成するには、生成したスクリプトを使用して、新しい環境で実行します。

DDL ステートメントを生成する方法は次のとおりです。

1. DDL ステートメントを生成したいオブジェクトを強調表示し、右マウス・ボタンをクリックします。
2. **「DDL の生成 (Generate DDL)」**を選択します。「スクリプトの実行 (Run script)」ウィンドウが表示されます。
3. ユーザー ID とパスワードを入力し、**「OK」**をクリックします。 **db2look** コマンドの内容でジョブが作成されます。 DB2 メッセージ・ウィンドウに、新しいジョブのジョブ ID が表示されます。
4. **「OK」**をクリックしてメッセージ・ウィンドウをクローズします。
5. 「ジャーナル (Journal)」ノートブックの「ジョブ履歴 (Job History)」を使用して、ジョブの結果を表示し、ジョブと関連した保管されたスクリプトの内容を表示します。
6. ジョブを選択し、右マウス・ボタンをクリックします。ポップアップ・メニューから、**「結果表示 (Show Result)」**を選択します。「ジョブ結果 (Job Results)」ウィンドウがオープンします。 **db2look** コマンドの出力が、「ジョブ出力 (Job Output)」ペインに表示されます。

7. **「スクリプトの作成 (Create Script)」**を選択し、結果のスクリプトを作成します。「新規コマンド・スクリプト (New Command Script)」ウィンドウが表示されます。
8. 再使用したい場合は、新しいスクリプトを保管します。

フィルター

コントロール・センターでは、コンテンツ・ペインに表示される情報、または表から実際の結果セットとして取り出される情報をフィルターにかけることができます。表示されるオブジェクトの数、または戻されるオブジェクトの数を、 1 つ以上のオブジェクトにフィルターを作成することによって制限することができます。一度フィルターを設定すると、ツリーでもう一度すべてのオブジェクトを表示したい場合、フィルターをクリアまたは削除する必要があります。

表示のフィルター処理

コンテンツ・ペインに表示されるオブジェクトの数を削減して、より管理しやすくする方法は次のとおりです。

1. コントロール・センターの下部にあるコンテンツ・ペイン・ツールバーからフィルター・アイコンを選択するか、または「表示 (View)」メニュー・バーから「フィルター (Filter)」を選択します。
2. オブジェクトの数を削減するための基準を選択します。
3. 「フィルターを使用可能にする (Enable filter)」チェック・ボックスを選択し、フィルターを活動化します。

後でオブジェクトを選択してその内容を表示するとき、そのオブジェクトと関連付けたフィルターは、前に設定した基準に従って表示を制限します。

検索されたデータのフィルター処理

照会で戻される行数を削減し、応答時間を速くするためには、オブジェクトの選択時に、コンテンツ・ペインで表示される出力または結果セットを定義できます。

1. ツリーからフォルダー・オブジェクトを選択し、右マウス・ボタンをクリックします。
2. ポップアップ・メニューから、**「フィルター (Filter)」**を選択します。「フィルター (Filter)」ウィンドウがオープンします。
3. フィルター機能を使って、そのオブジェクトに所属する行を検索するため、基準の設定を定義します。

フィルターを定義して特定のデータ集合を検索する

フィルターを定義して特定のデータ集合を検索する方法は次のとおりです。

1. コントロール・センターから、プラットフォームによって「データベース (Databases)」、または「サブシステム (Subsystems)」フォルダーを展開します。
2. フィルターを定義したいオブジェクトを選択します。そのオブジェクトで右マウス・ボタンをクリックします。

3. ポップアップ・メニューから**「フィルター (Filter)」**を選択します。フィルター・ノートブックがオープンします。
4. 「位置指定 (Locate)」ページで、選択されたオブジェクトの名前または他の記述的なフィルター基準を指定します。フィルターの結果は、コントロール・センターのコンテンツ・ペインに表示される、選択されたオブジェクトと関連付けられた結果セットです。
5. 「位置指定 (Locate)」ページでは、そのページのフィールドで選択されたすべての条件に合うか、または少なくとも 1 つの条件に合えばよいかを指定するラジオ・ボタンを選択します。
6. フィルター・ノートブックの「拡張機能 (Advanced)」ページで、戻される行数をさらに制限するために、表示されるテキストを編集することによって、付加的な基準を使用することができます。
7. **「OK」**をクリックして、定義したフィルター基準を使用します。

行数に基づいてこのフィルター・ノートブックを自動的に起動するには、メニュー・バーから「ツール (Tools)」を選択し、ポップアップ・メニューから「ツールの設定 (Tools Settings)」を選択します。**「指定行数を超えた場合にフィルター処理を選択 (Select filtering when numbers of row exceeds)」**チェック・ボックスでは、どんな選択からでも戻される行の限界値を事前定義することができます。限界値に達すると、フィルター・ノートブックが表示され、定義された基準に基づいて現行の検索を制限できます。これは特に、表が予期しない大きさになり、前にフィルター処理されていない場合に役立ちます。プラットフォーム、およびデータによっては、行のサブセットだけが必要なときに、数百万行を戻そうとするようなこともあり得ます。

ヘルプ

広範なヘルプ情報が管理ツールで提供されています。メニュー・ツールバーに加えて、すべてのウィンドウおよびノートブックにヘルプ・ボタンがあります。一般ヘルプに加えて、フィールドの入力およびタスクの実行方法に関するヘルプもあります。ヘルプ・メニューから、用語の索引または参照情報、および製品マニュアルで提供される情報にもアクセスできます。

コントロール・センター

コントロール・センターは、管理の中心として、システム、DB2 インスタンス、データベースや、表、視点、ユーザー・グループなどのデータベース・オブジェクトを管理します。また、コントロール・センターを使用して、 DB2 (OS/390 版) サブシステムにアクセスすることもできます。コントロール・センターに表示される前に、すべての DB2 データベースをカタログ化する必要があります。 12ページの図1 は、コントロール・センターの主な機能を示しています。オペレーティング・システムの違いのため、ご使用のシステム上のコントロール・センターが図とは異なる場合があります。

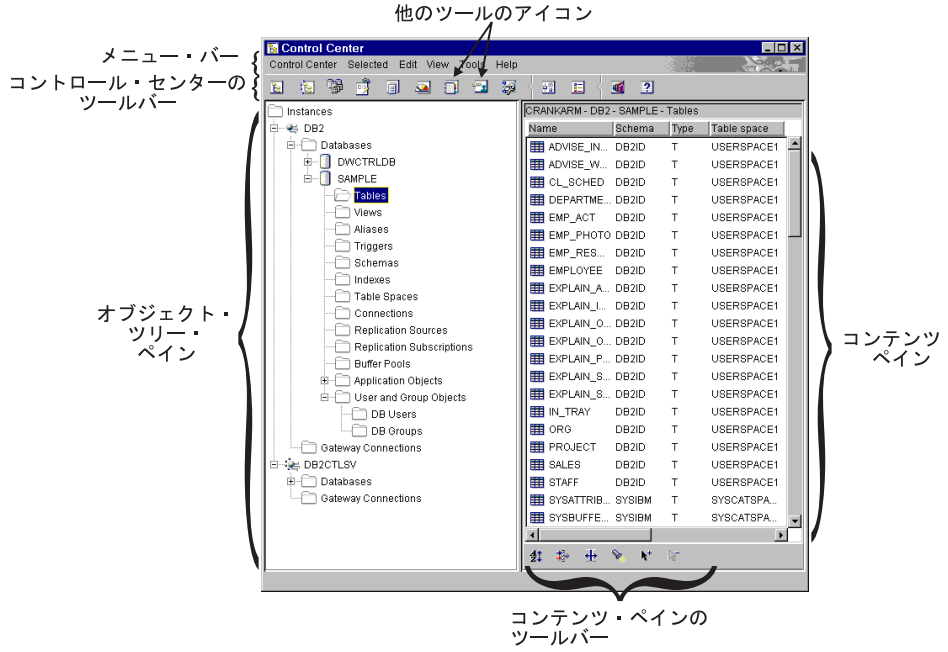


図 1. コントロール・センターの機能

コントロール・センターの主なエレメント

コントロール・センターの主なエレメントは次のとおりです。

- メニュー・バー。画面の上部にあります。メニュー・バーからメニューを選択すると、DB2 ツールのシャットダウン、図形処理ツールへのアクセス、オンライン・ヘルプおよび製品情報へのアクセスの獲得など、多くの機能を実行できます。メニュー・バー上の各項目をクリックすると、これらの機能についてよく知ることができます。
- コントロール・センター・ツールバー。コントロール・センターおよび他のツールのアイコンは、コントロール・センター・ツールバーにあります。カーソルがアイコンの上にくると、吹き出しヘルプが表示されて各アイコンを識別します。

  コントロール・センターのツールバーから「ツール設定 (Tools Settings)」アイコンを選択すると、これらのツールの設定を変更できます。
- オブジェクト・ツリー。オブジェクト・ツリーは画面の左の区画にあります。すべてのデータベース・サーバー、およびコントロール・センターから管理できるオブジェクトのアイコンを表示します。リモート・データベース・サーバーがオブジェクト・ツリー・ペインに表示される前に、まずカタログ化する必要があります。オブジェクト・ツリー・ペインのいくつかのオブジェクトについては、他のオブジェクトが入っています。オブジェクトの左側にある正符号 (+) は、オブジェクトが縮小されている

ことを示します。正符号をクリックすると、展開することができます。展開されると、オブジェクトの左側に負符号 (-) が表示されます。オブジェクトを縮小するには、負符号をクリックします。

- コンテンツ・ペイン。コンテンツ・ペインは、画面の右の区画にあります。このページは、オブジェクト・ツリー・ペインで選択されたオブジェクトに含まれる、すべてのオブジェクトが表示されます。たとえば、オブジェクト・ツリー・ペインの表フォルダーを選択した場合、すべての表がコンテンツ・ペインに表示されます。データベース・フォルダーを選択すると、コンテンツ・ペインが変更されて、すべてのデータベースが表示されます。コンテンツ・ペイン・ツールバーの「フィルター (Filter)」アイコンをクリックし、必要な情報を指定することによって、コンテンツ・ペインに表示される列をフィルター処理することができます。または、ツールバーの**「ツール (Tools)」**を選択してから、**「ツールの設定 (Tools Settings)」**を選択して、オブジェクトをフィルター処理することができます。「コンテンツ・ペインのフィルター (Contents Pane filter)」ダイアログで、必ず**「フィルターを使用可能にする (Enable Filter)」** チェック・ボックスを選択してください。
- コンテンツ・ペイン・ツールバー。このツールバーは、コンテンツ・ペインの下部にあります。これによって、コンテンツ・ペインの情報を調整できます。このツールバーは、プロダクト全体でほとんどの詳細表示の下部または横に表示される、共通制御です。

コントロール・センターで作業を行っているときに、一部のフィールドが太い赤色の枠で強調表示されている場合があるかもしれません。この枠は、そのフィールドが入力の必要な必須フィールドであることを示しています。値を選択または入力すると、この赤色の枠は消えます。

DB2 (OS/390 版) でカスタマイズされたコントロール・センターを使用する

DB2 (OS/390 版) プラットフォームのカスタマイズされたコントロール・センターを、独自の定義済み管理点として使用して、サブシステム、データベース、または表、視点、データベース・ユーザーなどのデータベース・オブジェクトを管理します。このカスタマイズされたコントロール・センターを使用して、定義する DB2 (OS/390 版) オブジェクトにアクセスすることができます。

カスタマイズされたコントロール・センターの主なエレメントは、デフォルトのコントロール・センターで列挙されたものと同じです。カスタマイズされたコントロール・センターによって、個別設定されたコントロール・センターに含めたいオブジェクトを指定することができます。このユーザー定義のツリーは、DB2 オブジェクトを管理するために、保管し、起動することができます。これは、すべてのユーザーのデフォルトであるコントロール・センターのツリーに置き換わるものではありませんが、コントロール・センターが起動されるたびに同じようにオブジェクトのセットにアクセスしたい場合は役に立ちます。 必要なだけカスタマイズされたツリーを作成することができ、それぞれのツリーに、オブジェクトの異なるセットを含め、選択によりどんな順序にもすることができます。

カスタマイズされたツリーを使用すると、 DB2 オブジェクトの固定階層を介して経由するための労力が削減され、また関連オブジェクトをグループ化する手段も提供されます。たとえば、給与計算情報がある表だけを含むツリーを定義することができます。

管理可能なシステム

コントロール・センターから、OS/2、Windows、および UNIX プラットフォームのユニバーサル・データベース・ファミリー用のデータベース・オブジェクトを管理できます。各プラットフォーム固有のインストール情報およびセットアップ情報については、*概説およびインストール* を参照してください。

DB2 (AS/400 版)、DB2 (VSE および VM 版) システム、および DB2 (OS/390 版) から、 DB2 ユニバーサル・データベース・ファミリーの製品にデータを複製することもできます。製品間の複製については、 *レプリケーションの手引きおよび解説書* を参照してください。

管理可能なオブジェクト

コントロール・センターからオブジェクトを管理したい場合、それらのオブジェクトをオブジェクト・ツリーに追加する必要があります。データベースを除去するか、またはコントロール・センターの外部にアンカタログし、コントロール・センターを使ってそのデータベースでタスクを実行したい場合、データベースをオブジェクト・ツリーに追加する必要があります。

コントロール・センターから管理できる DB2 ユニバーサル・データベース・オブジェクトは、次のとおりです。

- システム
- インスタンス
- 表
- 視点
- 索引
- トリガー
- ユーザー定義タイプ
- ユーザー定義関数
- パッケージ
- 別名
- 複製オブジェクト
- ユーザーおよびグループ

コントロール・センターから管理できる DB2 (OS/390 版) バージョン 5 オブジェクトは、次のとおりです。

- バッファー・プール

- 視点
- カタログ表
- ストレージ・グループ
- 別名
- 同義語
- DB2 ユーザー
- ロケーション
- アプリケーション・オブジェクト (コレクション、パッケージ、プラン、プロシージャー)
- データベース
- 表
- 表スペース
- 索引
- 複製ソース
- 複製サブスクリプション

DB2 (OS/390 版) バージョン 6 では、上記のオブジェクトすべてに加えて、次のオブジェクトもコントロール・センターから管理できます。

- スキーマ
- トリガー
- ユーザー定義関数
- 特殊タイプ

これらの各オブジェクトで実行されるアクションを見るには、オブジェクト・ペインでオブジェクトを選択し、右マウス・ボタンをクリックしてください。機能をリストしたポップアップ・ウィンドウが表示されます。

コントロール・センターでのシステム表示

システムでカタログ化されており、 DB2 をインストールしてあるすべてのシステムを表示する方法は次のとおりです。

1. **「システム (Systems)」**の隣の正符号 (+) をクリックして、オブジェクト・ツリーを展開します。ローカル・マシンおよびリモート・マシンがあればそのアイコンが表示されます。ローカル・システムは、「ローカル (Local)」アイコンによって表示されます。ローカル・マシンが DB2 サーバーである場合のみ表示されます。「ローカル (Local)」アイコンで右マウス・ボタンをクリックすると、ポップアップ・メニューのオプションのうちの 1 つ、**「管理サーバーに接続 (Attach to administration server)」**が呼び出されます。管理サーバーによって、パフォーマンス・モニター、およびスケジューリングなどの機能を利用できます。これは DB2 サービス要求を満たすため、 DB2 管理ツールによって使用され、自動的に作成されて開始します。

DB2 管理サーバーのデフォルト名は、プラットフォームによって異なります。たとえば、Windows プラットフォームおよび OS/2 プラットフォームの場合、『DB2DAS00』 が使用されます。 AIX プラットフォームの場合は、『db2as』 が使用されます。

2. 「ローカル (Local)」アイコンを展開します。ローカル・マシン上の DB2 のインスタンスが、ツリー構造に表示されます。

   OS/2、Windows およびサポートされる DB2 UNIX ベースのシステムでは、データベース・マネージャー・コードの各コピーを、マシン上のディレクトリーに保管される個別のインスタンス と考えることができます。 DB2 (OS/390 版) では、インスタンスはサブシステムとして参照されます。デフォルトのローカル・インスタンスが、DB2 のインストール時に作成されます。 1 つのシステムに、複数のインスタンスを持つことができます。これらのインスタンスを使用して、実稼働環境から開発環境を分離したり、機密情報を特定の人々のグループに制限したりすることができます。また、特定の環境に合わせてインスタンスを調整することもできます。

3. **「インスタンス (Instances)」**アイコンを展開します。存在する各データベースごとに、アイコンと名前が表示されます。

DB2 (OS/390 版) オブジェクトの管理

コントロール・センターを使って、既存の DB2 (OS/390 版) バージョン 5、および DB2 UDB (OS/390 版) バージョン 6 製品の多くの機能、たとえばオブジェクトの作成、変更、除去などを実行し、さらにデータの再編成やロードを行うユーティリティーを実行することができます。しかし、コントロール・センターから DB2 (OS/390 版) サブシステムを管理する前に、まずサブシステムへの接続を構成してからオブジェクト・ツリーに追加することが必要です。

DB2 (OS/390 版) サブシステムの追加

クライアント構成アシスタントをインストールした場合、それを使って DB2 (OS/390 版) システムへの接続を簡単に構成できます。クライアント構成アシスタントをインストールしていない場合、コマンド行プロセッサー (CLP) を使用して DB2 (OS/390 版) システムへの接続を手動で構成する必要があります。

クライアント構成アシスタントは、クライアントに対して LAN 上で使用可能な、すべての DB2 (OS/390 版) システムをネットワークで検索するために使用します。 DB2 (OS/390 版) システムの 1 つを追加したい場合、**データベースの追加**ウィザードを使用してシステムを追加するか、プロファイルを使用して接続をインポートするか、または手動で接続を追加することができます。

ネットワークの検索を選択する場合、システムに接続を定義したネットワーク上に DB2 コネクト製品がなければなりません。アクセス・プロファイルの使用を選択する場合、追加したいシステムを表す DB2 コネクト・サーバー接続をプロファイルから選択する必要があります。手動で接続を構成することにした場合、システム名、通信プロトコル、およびホスト名および TCP/IP のポート番号、または SNA の記号宛先名などの通

信プロトコル・パラメーターを知っておく必要があります。 DB2 (OS/390 版) システムが追加されると、 DB2 コネクト・サーバー接続を表すオブジェクトが、コントロール・センターのローカル・システムに表示されます。

DB2 (OS/390 版) バージョン 5 以降のシステムを追加すると、そのシステムはコントロール・センターのオブジェクト・ツリー内にある独自のセクションに表示されます。特定のシステムにある DB2 (OS/390 版) および他のデータベース・オブジェクトを表示するには、その DB2 (OS/390 版) システムを表す DB2 (OS/390 版) システム・アイコンから、オブジェクト・ツリーを展開します。

特定のオブジェクトで実行できるアクションのリストを表示するには、オブジェクト・ツリーで表示されるオブジェクトを選択し、右マウス・ボタンをクリックしてください。ポップアップ・メニューに、そのオブジェクトで実行できる使用可能なアクションが表示されます。たとえば、視点の作成、変更、除去、さらにその内容の表示、その特権の修正、および関連する他のオブジェクトのリスト表示を行うことができます。実行できる機能についての詳細は、 DB2 (OS/390 版) オブジェクトのオンライン・ヘルプを参照してください。

ゲートウェイ接続の管理

DB2 コネクト・サーバーがカタログ化されると、ローカル・システムのインスタンス・オブジェクトにあるコントロール・センターのオブジェクト・ツリーに「ゲートウェイ接続 (Gateway Connections)」フォルダーが表示されます。「ゲートウェイ接続 (Gateway Connections)」フォルダーには、ローカルにカタログ化されるホストおよび AS/400 データベースへの接続を管理するのに使用される、オブジェクトの階層が入っています。これらの接続管理オブジェクトに関連したアクションを使用して、ホストおよび AS/400 データベース接続のリスト、強制実行、およびモニターを行えます。

「ゲートウェイ接続 (Gateway Connections)」フォルダーにあるオブジェクト・ツリーは、ホストおよび AS/400 データベースへの接続の管理に使用されますが、データベース管理タスクには使用されません。しかし、ローカル・システム上でホストまたは AS/400 データベースを追加、変更、または除去する必要がある場合は、クライアント構成アシスタントを使用します。

コントロール・センターから実行できる機能

コントロール・センターから、次のことを実行できます。

- データベース・オブジェクトの管理。データベース、表スペース、表、視点、索引、トリガー、およびスキーマを作成、変更、除去することができます。 ユーザーを管理することもできます。
- データの管理。データをロード、インポート、エクスポート、再編成したり、統計を収集したりできます。
- ジョブのスケジュール。ジョブには、スクリプトの実行の保留、実行、完了などがあります。 ジョブをスケジュールして特定の時刻に開始することができます。

- データベースまたは表スペースのバックアップと復元による予防保守の実行。
- パフォーマンスのモニターおよびトラブルシューティングの実行。
- データの複製。
- インスタンスおよびデータベースの構成および調整。
- DB2 コネクト・サーバーおよびサブシステムなどのデータベース接続の管理。アプリケーションの管理。
- アクセス・プランを調べるための Explain SQL を使用した照会の分析。
- コントロール・センター内にあるメニューやテキストの表示に使用されているフォントの変更。フォント、フォントのサイズ、色などを変更できます。変更を有効にするためにはコントロール・センターを再始動する必要があります。
- 他のツールの立ち上げ。たとえば、サテライト管理センターまたはコマンド・センターを立ち上げることができます。

オブジェクトで実行できるすべてのアクションを表示するには、オブジェクト・ツリー・ペインまたはコンテンツ・ペインからオブジェクトを選択し、右マウス・ボタンをクリックしてください。オブジェクトのそのタイプで実行できるすべての機能がポップアップ・メニューに表示されます。たとえば、表フォルダーを選択した場合、ウィザードのヘルプがあってもなくても、新しい表の作成、表のパフォーマンスのモニター、コンテンツ・ペインに表示される表のフィルター処理などを実行できます。 実行できる機能は、選択するオブジェクトによって異なります。

コンテンツ・ペインのオブジェクトで右マウス・ボタンをクリックし、特定のオブジェクトの付加的な機能を実行します。たとえば、コンテンツ・ペインで表の 1 つを選択して右マウス・ボタンをクリックした場合、ポップアップ・ウィンドウがその表で使用できる機能を表示します。

新しいオブジェクトの作成

新しいオブジェクトの作成方法は次のとおりです。

1. データベース・フォルダーを展開します。オブジェクト・タイプがフォルダー・アイコンとして表示されます。
2. オブジェクトのフォルダー・アイコンで右マウス・ボタンをクリックします。たとえば、**「表 (Tables)」**アイコンをクリックします。ポップアップ・メニューが表示されます。いくつかのオブジェクトについては、機能を実行するのに 2 つのオプションがあります。 1 つのオプションは、ウィザードを使用することです。ウィザードは、実行できるすべての機能に存在するわけではありません。
3. **「作成 (Create)」**を選択します。表を作成するためのウィザードがあるので、オプションは 2 つです。そのうち 1 つはウィザードを使った表の作成です。ウィザード・オプションを選択すると、情報を入力するようにというプロンプトが出され、選択肢についての提案が表示されます。ウィザードは、経験の浅いユーザーや、データベース・オブジェクトを頻繁に作成しない人に特に役立ちます。

既存のオブジェクトの処理

オブジェクト・ツリー・ペインの表フォルダーなどのオブジェクトをクリックすると、すでに存在するすべての表がコンテンツ・ペインに表示されます。その後、処理したい表を選択して右マウス・ボタンをクリックし、その特定の表で実行したい機能を起動します。

コントロール・センターの使用の詳細については、**「ヘルプ (Help)」**メニューで使用可能な、またはコントロール・センターのどこででも F1 を押して、オンライン・ヘルプに移動してください。

オブジェクトの位置指定 (DB2 (OS/390 版) のみ)

「位置指定 (Locate)」ノートブックを使用して、簡単にデータベースまたはサブシステム・オブジェクトを検索できます。これによって、以下のことが可能です。

- コントロール・センターのツリー構造を経由してナビゲートしなくても、オブジェクトを検索できます。オブジェクトは 1 つのデータベースまたはサブシステム、表スペースにあることも、複数のデータベース、表、およびサポートするオブジェクトにわたって存在することもあります。
- サブシステム内の複数のデータベースで、オブジェクト (表スペース、表、および索引) を見つけることができます。

「位置指定 (Locate)」ノートブックの「位置指定 (Locate)」ページを使用して、検索基準を指定します。「位置指定 (Locate)」ノートブックの「拡張機能 (Advanced)」ページを使用して、検索をさらにカスタマイズします。「拡張機能 (Advanced)」ページで提供されるテキストを編集し、検索基準を追加または変更します。

データベースまたは DB2 (OS/390 版) サブシステム内で定義されたオブジェクトを見つける方法は、以下のとおりです。

1. コントロール・センターのオブジェクト上で右マウス・ボタンをクリックします。ポップアップ・メニューから**「位置指定 (Locate)」**を選択します。「位置指定 (Locate)」ノートブックがオープンします。
2. 「オブジェクト・タイプ (Object type)」フィールドから、検索するデータベース・オブジェクトのタイプを選択します。使用できるターゲット・オブジェクトのリストは、検索を開始するオブジェクトによって異なります。
3. 「位置指定 (Locate)」ページで、検索基準を入力します。最低 1 つの検索基準を入力することが必要です。また、検索の助けとしてワイルドカードを使用することができます。有効な区切り文字を使用して小文字または拡張文字セットを囲まない限り、文字は英大文字に変換されます。
4. 「位置指定 (Locate)」ページで、そのページのフィールドで選択されたすべての条件に合うか、または条件のうち少なくとも 1 つに合えばよいかを指定するラジオ・ボタンを選択します。

5. **「OK」**をクリックして、検索基準を使用します。検索結果が「位置指定結果 (Locate Result)」ウィンドウに表示されます。出力表の形式は、検索したオブジェクトのタイプによって異なります。
6. 同じ、または異なる基準で検索を繰り返すには、**「適用 (APPLY)」**をクリックします。
7. 「位置指定結果 (Locate Result)」ウィンドウに表示される行を選択し、その行で右マウス・ボタンをクリックして、実行できる付加的なアクションを表示するポップアップ・メニューを見ることができます。

サテライト管理センター

サテライト管理センターは、 DB2 コントロール・センターから使用可能なツールのセットです。これによって、集中管理点から DB2 サーバーの集合のセットアップ、および管理を実行できます。グループに所属する各 DB2 サーバーは、サテライトと呼ばれます。集中管理点からサテライトを管理することは、 DB2 が DB2 サテライトを使用する人からは隠れているため、ユーザーがデータベース管理について学ぶ必要がないことを意味します。

グループを使用して、特性を共用している DB2 サーバー、たとえばそれらで実行されるアプリケーションや、アプリケーションをサポートするデータベース構成などについて DB2 サーバーを編成できます。それらの DB2 サーバーは、データベース構成、使用法、および目的の点で類似しています。

DB2 サーバーを一緒にグループにすることによって、各 DB2 サーバーを個々に管理するのではなく、DB2 サーバーのグループを管理できます。 DB2 サーバーの既存のグループと同じ役割を果たす付加的な DB2 サーバーを獲得した場合、サテライト管理センターを使ってそれらをそのグループに追加することができます。

サテライト管理センターから、グループ、サテライト、アプリケーション・バージョン、バッチ、および認証権限を作成できます。 また、正常なコード・セットを定義し、サテライト環境の管理に関連した他の機能を実行することもできます。サテライト環境についての情報は、サテライト制御データベースと呼ばれる中央データベースに保管されています。このデータベースは、他のものの中で、どのサテライトが環境に存在するか、各サテライトがどのグループに所属するか、および、どのバージョンのエンド・ユーザー・アプリケーションをサテライトが実行しているかを記録します。このデータベースは、DB2 制御サーバーと呼ばれる DB2 サーバーにあります。

サテライト管理センターの機能性を使用可能にする前に、コントロール・センターで、サテライト制御データベース (SATCTLDB) をまずカタログ化する必要があります。使用可能にすると、サテライト・コントロール・センターを使って、サテライトがアプリケーション・バージョンに合わせて同期化するときに実行する、サテライト、グループ、およびバッチをセットアップおよび維持することができます。

データベース構成をセットアップおよび維持するために、各サテライトはサテライト制御データベースに接続し、エンド・ユーザー・アプリケーションのバージョンに対応するバッチをダウンロードします。サテライトはこれらのバッチをローカルに実行してから、サテライト制御データベースに結果を報告します。バッチのダウンロード、実行、およびバッチ実行の結果のプロセスは、同期化と呼ばれます。サテライトは、グループに所属し、エンド・ユーザー・アプリケーションの同じバージョンを実行している他のサテライトとの一貫性を保つために、同期化を行います。

コマンド・センター

ツールバーの「コマンド・センター (Command Center)」アイコンをクリックすることによって、コントロール・センターからコマンド・センターを開始できます。

コマンド・センターによって次のことが実行できます。

- 1 つ以上の SQL ステートメントおよび DB2 コマンドの結果出力を、結果ウィンドウで見ることができます。結果をスクロールしたり、レポートを生成したりできます。
- スクリプト・センターにコマンド・スクリプトを作成および保管できます。コマンド・スクリプトを編集して、新しいスクリプトを作成できます。次に、スクリプト・センターからコマンド・スクリプトを、指定するときにはいつでもジョブとして実行するようにスケジュールすることができます。
- SQL ステートメント、DB2 コマンド、およびオペレーティング・システム・コマンドを実行できます。コマンド・センターから DB2 コマンドを実行する場合、コマンドの最初に **DB2** を付ける必要はありません。 REXX など、サポートされるオペレーティング・システムのスクリプト言語であれば、最初に感嘆符 (!) を付けることによってオペレーティング・システム・コマンドを実行できます。コマンド・センターを使用してコマンドおよびステートメントを実行すると、それぞれのコマンドを個々に入力して実行しなくても、たくさんのコマンドを一度に発行できます。
- メイン・ツールバーから、コントロール・センターなどの DB2 管理ツールに、すばやくアクセスできます。
- 実行前に、SQL ステートメントに関連したアクセス・プラン、および統計を表示することができます。

スクリプト・センター

コントロール・センターのツールバーからアイコンを選択すると、スクリプト・センターを開始できます。スクリプト・センターは、必要なときにいつでも実行用にスケジュールできる、コマンドおよびステートメントのセットを作成することによって、スクリプトを作成するためのツールです。以前作成したスクリプト、またはコマンド・センターに保管したスクリプトをインポートすることができます。保管されるスクリプトのセットからスクリプトを選択したり、既存のスクリプトを編集して新しいスクリプトを作成したり、スクリプトをコピーまたは除去したりすることができます。

スクリプト・センターの内部でも、または独自のエディターを使ってスクリプト・センターの外部で、スクリプトを編集することができます。スクリプト・センター内からスクリプトを実行する場合、ジャーナルで結果がログに記録されるという利点もあります。

スクリプト・センターのスクリプトからオペレーティング・システム・コマンドを実行する方法は、次のとおりです。

1. **「スクリプト -> 新規 (Script —> New)」**を選択します。「新規コマンド・スクリプト (New Command Script)」ウィンドウがオープンします。
2. **スクリプト・タイプ**について、**「OS コマンド (OS command)」**ラジオ・ボタンを選択します。
3. スクリプト名、説明、および作業ディレクトリーを入力します。
4. コマンドを入力します。
5. **「OK」**をクリックします。

スクリプト・センターから、システムに通知されるすべてのコマンド・スクリプトについての記述やスクリプト・タイプなどの情報を表示でき、また以下のタスクを実行することができます。

- DB2 およびオペレーティング・システム・コマンドを含む、コマンド・スクリプトを作成する。
- 保管されたコマンド・スクリプトを即時に実行する。
- 後の日付または定まった間隔で実行するようにスクリプトをスケジュールする。たとえば、複数の表の統計を収集するスクリプトを作成したい場合があるかもしれません。その場合、ジョブが夜間に実行されるようにスケジュールできます。ジョブを実行したい時間、日付、週、月、一週間に複数回、または一月に複数回を指定することによって、スケジュールされた間隔で、それらを自動で実行するようにジョブをスケジュールできます。ジョブは、スクリプトをスケジュールするか、またはスクリプトをすぐに実行するときに作成されます。
- ツールバーからジャーナルにアクセスし、特定のスクリプトを使用するジョブ、およびすべてのスケジュールされたジョブの状況を表示する。
- 保管されたコマンド・スクリプトを編集する。

スクリプト・センターで既存のスクリプトを使用する

スクリプト・センターから作成したのではない、既存のスクリプトでスクリプト・センターを使用する方法は次のとおりです。

1. **コントロール・センター**のツールバーから、**「スクリプト・センター (Script Center)」**アイコンをクリックします。スクリプト・センターがオープンします。
2. **「スクリプト -> インポート (Script —> Import)」**を選択します。「ファイル・ブラウザー (File Browser)」ウィンドウがオープンします。

3. 既存のスクリプト・ファイルを選択し、**「OK」**をクリックします。「新規コマンド・スクリプト (New Command Script)」ウィンドウがオープンします。スクリプト・エディターのウィンドウの下の部分に、スクリプトが表示されます。**「インスタンス (Instance)」**、**「スクリプト名 (Script name)」**、**「スクリプト記述 (Script description)」**、および**「作業ディレクトリー (Working directory)」**フィールドに入力し、**「スクリプト・タイプ (Script type)」**を選択します。
4. **「OK」**をクリックします。スクリプト・センターにスクリプトが作成されます。

保管されたコマンド・スクリプトを実行するためのスケジュール

スクリプトをスケジュールする方法は次のとおりです。

1. コントロール・センターのツールバーで、**「スクリプト・センター (Script Center)」**アイコンをクリックします。スクリプト・センターがオープンします。
2. 実行をスケジュールするスクリプトで右マウス・ボタンをクリックし、ポップアップ・メニューから**「スケジュール (Schedule)」**を選択します。「スケジューラー (Scheduler)」ウィンドウがオープンします。
3. ジョブの頻度と、完了アクション (完了メッセージまたは立ち上げる別のコマンド・スクリプトなど) を選択します。
4. **「OK」**をクリックします。これで、ジャーナルで追跡できる保留中のジョブが開始します。

ジャーナル

コントロール・センターのツールバーからアイコンを選択すると、ジャーナルを開始できます。 ジャーナルを使って、ジョブのモニターおよび結果の検討が行えます。また、ジャーナルから、リカバリー・ヒストリーおよび DB2 メッセージを表示することもできます。ジャーナルによって、次のことが可能です。

- 保留中のジョブ、実行中のジョブ、ジョブ履歴のモニター
- 結果の検討
- リカバリー・ヒストリーとアラート・メッセージの表示
- DB2 メッセージのログの表示

ジョブの処理

ジャーナルを使ってジョブを処理します。ジャーナルをオープンする方法は次のとおりです。

1. スクリプト・センターから、**「ジャーナル (Journal)」**アイコンをクリックします。ジャーナルがオープンします。
2. 後で実行されるようにスケジュールされたジョブを表示するには、**「保留中のジョブ (Pending jobs)」**押しボタンをクリックします。 すると、保留中のジョブのリストにジョブが表示されます。また、ジョブについてのすべての情報も表示されます。保

留中のジョブでは、スケジュールの変更、関連するスクリプトの表示、または即時の実行などのアクションを実行できます。保管スクリプトが変更されると、そのスクリプトに従属するすべてのジョブが、新しく修正された動作を継承します。

また、ジャーナルから、現在実行中のジョブおよびジョブ履歴を表示することもできます。

「ジャーナル (Journal)」ウィンドウの他のページは次のとおりです。

- 「回復 (Recovery)」ページ。このページでは、リカバリー・ヒストリー (バックアップからの詳細、復元操作、およびロード操作) が表示され、回復ログを復元できます。
- 「アラート (Alerts)」ページ。このページはすべての警報を表示します。
- 「メッセージ (Messages)」ページ。このページは DB2 管理ツールを介して発行されたすべてのメッセージを表示します。

ジャーナルのオンライン・ヘルプでは、ジョブおよびログの処理について詳細なステップを説明しています。

ライセンス・センター

ライセンス・センターは、システム上にインストールされた DB2 プロダクトの、ライセンス状況および使用状況についての情報を表示します。また、ライセンス・センターでは、適切なライセンス・モニター用にシステムを構成することができます。ライセンス・センターによって、次のことが可能です。

- 新しいライセンスを追加する。
- プロダクトの試供ライセンスから、永続ライセンスにアップグレードする。
- ライセンスの詳細を表示する。

ライセンス情報の詳細を表示すると、次の事項が表示されます。

- プロダクト名
- バージョン情報
- 満了日付
- 登録ユーザー
- 並行ユーザー数
- 資格のあるユーザー数
- 並行ユーザー数
- 制約ポリシー
- プロセッサー数 (DB2 ユニバーサル・データベース エンタープライズ・エディションおよびエンタープライズ拡張エディションの場合)

アラート・センター

アラート・センターは、システムをモニターし、潜在的な問題について警告するツールです。アラート・センターを自動的にオープンするように設定し、限界値を超え、したがってアラームまたは警告の状態にある、モニターされたオブジェクトがあれば表示することができます。コントロール・センターから起動できるパフォーマンス・モニターを使って、限界値を設定します。アイコンの色は、警告の重大度を示します。赤いアイコンはアラームを示しています。黄色いアイコンは警告を示しています。

クライアント構成アシスタント

基本的にクライアント構成アシスタントは、クライアントをローカルまたはリモート DB2 サーバーにセットアップするのに役立つウィザードが含まれたツールです。しかし、このツールを使用して DB2 コネクト・サーバーを簡単に構成することもできます。

クライアント構成アシスタントでは、アプリケーションが接続できるデータベースのリストを保守します。クライアント構成アシスタントはノードやデータベースをカタログ化することにより、構成タスクにつきものの煩わしさから解放します。

クライアント構成アシスタントからは、次のタスクを実行できます。

- データベース接続項目の追加、変更、および削除。
- 選択したデータベースとの接続テスト。
- データベース・マネージャー構成パラメーターの構成。
- CLI/ODBC 設定の構成。
- 選択したデータベースへの DB2 ユーティリティーおよび他のアプリケーションのバインド。
- 構成情報のインポートおよびエクスポート。 これにより、すでに構成が完了しているマシンの既存の構成を使用して、新しいマシンを構成することが可能になります。
- 選択したデータベースへの接続に使用しているユーザー ID 用のパスワードの変更。

クライアント構成アシスタントでは、以下の方法により、新しいデータベース接続項目を簡単に追加できるようになっています。

- プロファイルの使用。構成済みのマシンからプロファイルをエクスポートすることにより、それを新しいマシンの構成にも使用できます。サーバー・プロファイルはコントロール・センターから、クライアントまたはサーバー・プロファイルは CCA からエクスポートできます。
- ネットワークの検索。 CCA はネットワークを検索して、管理サーバーが実行されている DB2 システムを探し出すことができます。検索ディスカバリー・モードと既知 (または指示) ディスカバリー・モードがあります。検索ディスカバリー・モードの場合は、ネットワーク構成に制限が課されます。 (一般的なネットワーク・ルーターは、検索ディスカバリー要求の送信を許可しません。) 既知ディスカバリーの場合

は、ある程度の情報だけで必要としているサーバー・システムを探し出すことができます。すでにゲートウェイに定義されているホストまたは AS/400 システムを探し出すこともできます。

- データベースへの接続の手動構成。すべての情報を入力する必要がありますが、ウィザードが必要な作業のお手伝いをします。

パフォーマンス・モニター

パフォーマンス・モニターは、DB2 ユニバーサル・データベース、およびそれが制御するデータの状態についての情報を提供します。これは、データベース環境に合わせてカスタマイズ可能なグラフィカル・ユーティリティーです。パフォーマンス・モニターが収集する値が受け入れ可能な範囲内にない場合に、警告またはアラームの引き金となる限界値またはゾーンを定義することができます。

オブジェクト・ツリー・ペインまたはコンテンツ・ペインでオブジェクトを選択し、右マウス・ボタンをクリックすることによって、インスタンス、データベース、表、表スペース、および接続などの DB2 オブジェクトをモニターできます。ここから、モニター活動の開始を選択できます。

オブジェクトがモニター中の場合、アイコンは緑、黄色、赤で表示され、モニターの状況を示します。色は、すでに設定した限界値によって定義されたとおり、問題の重大度を表します。緑は、モニターが実行中で、すべて正常であることを表します。黄色は警告で、モニター中のオブジェクトが限界値に達しつつあることを示します。赤はアラームで、モニター中のオブジェクトが限界値に達したことを示します。 DB2 に含まれる事前定義モニターを使用するか、または独自のモニターを作成することができます。

パフォーマンス・モニターが収集している情報を表示するには、オブジェクトで右マウス・ボタンをクリックし、ポップアップ・ウィンドウの「**モニター活動の表示 (Show Monitor Activity)**」を選択します。

パフォーマンス・モニターからの情報は、以下の目的で使用します。

- パフォーマンス上の問題の検出
- 最適パフォーマンスを得るためのデータベースの調整
- パフォーマンス傾向の分析
- データベース・アプリケーションのパフォーマンスの分析
- 問題発生の予防

パフォーマンス・モニターを使用すると、ディスク活動、バッファー・プール使用状況、事前取り出しの量、ロックの使用状況、および特定の間隔でのレコード・ブロックなど、データベース情報のビジュアルな表示を作成することによって、傾向を分析することができます。

既存の問題のモニターが必要なとき、またはシステムのパフォーマンスを監視したい場合、このツールを使用します。これによってデータベース活動のスナップショット、および指定時刻でのパフォーマンス・データをとることができます。これらのスナップショットは、時系列の比較に使用されます。グラフのそれぞれの点は、データ値を表します。スナップショットをとるためのステップは、30ページの『一時点でのパフォーマンスのモニター』で説明されています。この情報は、潜在的な問題を識別および分析したり、設定した限界値に基づく例外条件を識別したりするのに役立ちます。データベース・マネージャーとそのデータベース・アプリケーションのパフォーマンスを、一時点について、また時系列的に傾向を調査する必要がある場合に、パフォーマンス・ツールを使用します。また、アラーム状態にある要素の概要を得るためにも使用します。これは、調整の必要なパラメーターを識別するのに役立ちます。その後、その要素に設定されていたパラメーターを詳しく調査し、パフォーマンスを改善するために変更することができます。

イベント・モニター

一時点のスナップショットをとるのとは対照的に、イベント・モニターはある一定の期間でのデータベース活動について情報を収集します。この収集された情報は、特定のデータベース・イベント、たとえばデータベース接続や SQL ステートメントの活動記録の分かりやすい要約を提供します。イベント・モニターは、特定のイベントが発生したときにデータベースの状態を記録します。これを使って、データベースの活動記録のトレースを獲得できます。イベント・モニターのレコードは保管され、データが取り込まれた後で分析されます。イベント・モニターは、トランザクションにかかる時間や、たとえば SQL ステートメントが CPU をどれだけ使ったかなどを知る必要がある場合に使用します。その後、イベント・アナライザーを使って、イベント・モニターから記録されたデータを読み取ります。

各データベース接続ごとに、1 つのイベント・レコードが作成されます。その接続で実行されるステートメントごとに、ステートメント・レコードが作成されます。各接続イベント・レコードは、イベント・アナライザーの「接続ビュー (Connections View)」ウィンドウの 1 行に対応します。このウィンドウは、モニターされた期間中に接続された、各アプリケーションごとの情報を表示するもので、以下の情報が含まれます。

- アプリケーション名
- 実行 ID
- 接続時刻
- 合計 CPU 時間
- ロック待機時間
- 合計ソート時間
- デッドロック
- 切断時刻
- アプリケーション ID

各ステートメント・イベント・レコードは、イベント・アナライザーの「ステートメント・ビュー (Statements View)」ウィンドウの 1 行に対応します。

モニター・ツールの使用

パフォーマンス・モニターおよびイベント・アナライザーには、以下の利点があります。

- 広範囲かつ柔軟なデータ収集。バッファー・プールおよび入出力、ロックおよびデッドロック、ソート、通信、エージェント、情報のログを含めた、 200 以上のパフォーマンス変数がサポートされています。データは、データベース・マネージャー、データベース、表スペース、表、バッファー・プール、接続、トランザクション、および SQL ステートメントについて表示されます。
- 使いやすく、直感的な表示。データは、読みやすいグラフ、または便利に論理グループに編成されたテキスト・ビューを使用して、リアルタイムで表示できます。詳細ビューと要約ビューの両方が提供され、さらに詳細な情報にアクセスすることもできます。
- 強力なアラート機能。どんなパフォーマンス測定にも、限界値を指定することによって例外条件を定義することができます。限界値は、パフォーマンス・グラフ上の特定のゾーンで測定を行うことによって、パフォーマンス測定に達したか、または限界値を超えたときを視覚的に識別するために使用されます。限界値に達したときに以下のアクションの一部または全部が起きるように指定できます。
  - アラート・センターを介して通知される。
  - 音響警報を受け取る。
  - プログラムが実行される。
  - メッセージが表示される。

  または、通知なしを選択することもできます。

29ページの図2 は、各種のモニターが協働する様子を示しています。

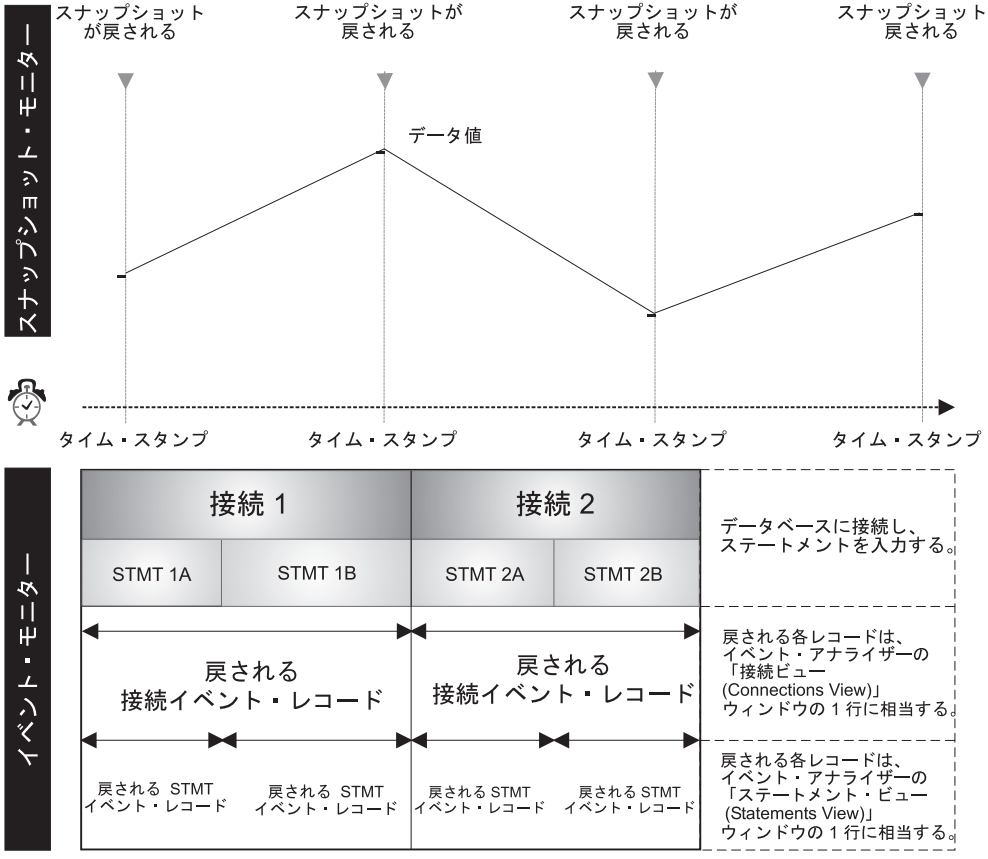


図 2. 比較: スナップショットの入手とイベントのモニター. (イベント・モニター、イベント・アナライザー)

データベースのモニターおよび調整に関する考慮事項

データベースのモニターおよび調整を開始する前に、以下のことを実行してください。

- 目標を定義する。たとえば、データベースの並行性が、特別なアプリケーションの開始時に低下するかどうか検査できるように、特定の時刻にインスタンス・レベルでアプリケーションがリソースを使用する様子を知りたい場合があります。または、たとえば特定のアプリケーションの実行中に、全体のパフォーマンスがよくない場合、アプリケーションの実行中に発生するインスタンス・レベルのイベントを知りたい場合もあります。
- 分析する情報を決定する。たとえば、ボトルネックがハードウェア関連かどうかを確認するために、スナップショットを取ってデータベース接続活動、または表スペース、バッファー・プール、および入出力活動をモニターする必要があるかもしれません。ボトルネックが環境に関連しているかどうか確認するには、イベント・アナライザーを使用して、以下のことをモニターします。

- ピーク時間にスケジュールされているデータベース・タスクが多すぎる
- ユーザー接続の数が多い
- データベース区画 (ハードウェア負荷平衡) がうまく最適化されていない
- サーバーが、データベース・サーバー以外にも使用されている

見える影響には、たとえば次のようなものがあります。

- 照会 / 応答が遅い
- スケジュールされたタスクが時間どおりに完了しない
- アプリケーションがタイムアウトになる

• DB2 で使用可能な事前定義モニターを使用するか、または独自のモニターを作成するか、決定してください。

次の節では、スナップショットの取り方、およびアラート・センターを使ってパフォーマンス関連の問題を追跡する方法を説明します。

一時点でのパフォーマンスのモニター

複雑なデータ・コレクションを実行し、データを分析して潜在的な問題の位置を正確に示したい場合、パフォーマンス・モニターを使用してシステムのスナップショットを取り、パフォーマンス・データの変化を、ある時間にわたって監視します。

このツールで以下のことを実行できます。

- パフォーマンス情報のグラフ化
- パフォーマンス・スナップショットの取り込み頻度の設定
- パフォーマンス計算の結果の表示
- 限界値および限界値アクションの定義
- アラートの生成および保管
- 要約情報の表示 (たとえば、すべてのデータベース)

次のタイプの情報が取り込まれます。

- 長期間にわたる活動についての情報 (アプリケーションの完了にかかる時間が長すぎる場合のデータベース活動など)
- 活動の現在のレベルについての情報を追跡するカウンター (データベース用のオープン・カーソルの数など)
- データベース活動についての累積情報 (データベース・インスタンスがアクティブである間に確立された接続の最大数、または特定のデータベースに対して実行された SQL ステートメントの合計数など)

事前定義された間隔でスナップショットを取ると、データベース・マネージャーおよびそのアプリケーションにおける活動の現在の状態のピクチャーが提供されます。この情報は、以下の目的で使用されます。

- パフォーマンス上の問題の検出
- パフォーマンス傾向の分析
- データベース・マネージャーおよびデータベース構成パラメーターの調整
- データベース・アプリケーションのパフォーマンスの分析

パフォーマンス情報は、以下のデータベース・オブジェクトで使用可能です。

- インスタンス
- データベース
- 表
- 表スペース
- データベース接続

それぞれのオブジェクトごとに、さまざまなパフォーマンス変数をモニターできます。どのスナップショット・モニターのウィンドウにもある、**「ヘルプ (Help)」**メニューから使用可能なパフォーマンス変数参照ヘルプは、すべてのパフォーマンス変数を説明しています。これらの変数は、カテゴリーごとに編成されます。以下のカテゴリーがあります。

- インスタンス: エージェント、接続、ソート
- データベース: ロックおよびデッドロック、バッファー・プールおよび入出力、接続、ソート、SQL ステートメント活動
- 表: 表
- 表スペース: バッファー・プールおよび入出力
- データベース接続: バッファー・プールおよび入出力、ロックおよびデッドロック、ソート、SQL ステートメント活動

スナップショットの生成に関する詳細は、オンライン・ヘルプを参照してください。

事前定義モニター

DB2 パフォーマンス・モニターには、事前定義モニターのセットがあり、そのまま使用するか、または自分の要件に合わせてコピーして修正することができます。これらは、パフォーマンス計算の包括的なセットを提供します。 IBM 提供のパフォーマンス・モニターの名前、等式、またはテキスト記述を変更することはできません。ただし、限界値およびアラート・アクションは変更できます。事前定義モニターを使用して、パフォーマンス・モニターについて学び、事前定義モニターをコピーしたり、自分のコピーからパフォーマンス変数を追加または除去したりすることによって、独自のモニターを作成します。

DB2 が備えている事前定義モニターは次のとおりです。

- 容量のモニター。システム容量についての情報を入手するのに使用します。このモニターは、ある期間にわたるシステムの全体的な使用状況を調べるために、定期的に検査することができます。
- ソート。ソート・ヒープおよびソート・ヒープしきい値パラメーターが、正しく設定されているかどうか確認するために使用します。このモニターは、活動のピーク期間に、またはアプリケーション変更の際に、最初にシステムを開始するときに実行する必要があります。
- ロッキング。このモニターは、システムでどれだけロッキングが発生するか、およびロック・リスト・パラメーターが適切に設定されているか判別するために使用します。
- キャッシュ。このモニターは、キャッシュの使用を最適化するために使用します。これらの値をピーク期間にモニターすることによって、キャッシュのサイズを増やす必要があるかどうか判別できます。
- バッファー・プール。このモニターは、小さな表で、独自のバッファー・プールが必要かどうかを判別するのに使用します。
- デッドロック。このモニターは、アプリケーションがデッドロックになっているかどうかを判別するのに使用します。
- 高速コミュニケーション・マネージャー。このモニターは、ノード間の情報の伝達に使用されるメモリーのパーセンテージを表示するのに使用します。
- 事前取り出し。このモニターは、システムに十分な事前取り出しを定義したかどうかを判別するのに使用します。
- ディスク・パフォーマンス。このモニターは、入出力を監視するのに使用します。データベースおよび表スペース・レベルでのディスク・パフォーマンスに焦点を当てたパフォーマンス変数が含まれます。
- グローバル・メモリー。このモニターは、アプリケーション・メモリーの使用を監視するのに使用します。
- 長時間実行メモリー。このモニターは、照会の完了に長い時間がかかっている理由を判別するのに役立ちます。
- ゲートウェイ接続。このモニターは、DB2 コネクト・サーバー接続を監視するのに使用します。

事前定義モニターの使用法の例については、パフォーマンス・モニターに提供されるオンライン・ヘルプを参照してください。

コントロール・センターから、使用可能なモニターのリストを表示するには、**「システム (Systems)」**フォルダーで右マウス・ボタンをクリックし、ポップアップ・メニューから**「モニターのリスト (List Monitors)」**を選択します。「モニターのリスト (List Monitors)」ウィンドウがオープンします。ここには、現在接続中の JDBC サーバーに保管されているモニターがリストされます。各モニターごとに、モニターの名前、説明、状況、デフォルト・モニターかどうか、およびモニターの作成者が表示されます。『モニターの状況』は、ローカル・システム上のモニター状況を示します。 JDBC サーバー

上の状況ではありません。 各レベルの『デフォルト』は、インスタンス、データベース、表、表スペース、または接続レベルでのデフォルト・モニターを示します。事前定義モニターについては、『作成 (Created by)』列に **NULLID** が含まれます。 ウィンドウの右側には、モニター上のさまざまなタスクを実行するための押しボタンがあります。 JDBC サーバーについて詳しくは、50ページの『コントロール・センターを Java アプレットとして実行する』を参照してください。

オブジェクトのデフォルト・モニターとして、開始するモニターを選択することができます。

パフォーマンス・モニターを開始すると、ツールバーの「アラート・センター (Alert Center)」ボタンをクリックして、モニター中で、限界値のいずれかに達したためにアラート状態であるオブジェクトの状況を表示できます。これらは、限界値を超えた期間だけ表示されます。

モニターされているオブジェクトを細かく監視したい場合、アラート・センターをオープンしたままにしておくか、または**「モニターの表示 (Show Monitor)」**ウィンドウを要約ページでオープンしたままにしておき、赤または黄色の警告がないかどうか監視することができます。また、コントロール・センターの設定を修正して、新しい警告またはアラームが追加された場合に自動的にアラート・センターがオープンするように修正することもできます。アラート・センターから、モニターが続行している間、一時的にアラートを延期することもできます。

オブジェクトがアラート・センターに表示される場合に必要なアクション

アラート・センターを自動的にオープンするように設定し、アラームまたは警告の状態にある (つまり、限界値を超えた)、モニターされたオブジェクトがあれば表示することができます。このデフォルトの変更は、「ツール設定 (Tools Settings)」ウィンドウから行えます。

アラート・センターにオブジェクトが表示されたら、そのオブジェクトで右マウス・ボタンをクリックし、**「パフォーマンス・モニター -> モニターの表示 (Performance Monitor -—> Show Monitor)」**を選択して、そのデータベース・オブジェクトのパフォーマンスの詳細を表示します。

一定期間のイベントの分析

イベント・アナライザーは、別の DB2 パフォーマンス・ツールです。このツールは、発生したイベントの診断情報が必要な場合に使用します。イベント・アナライザーは、イベント・モニターと組み合わせて使用します。たとえば、イベント・モニターを使って、データベースがアクティブな間に、接続、トランザクション、ステートメント、およびデッドロックなどのデータベース活動を追跡できます。また、イベント・モニターは、データベースからアプリケーションが切断されたときにログに記録される、累積パ

フォーマンス・データを記録することもできます。イベント・モニターがイベント・モニター・ファイルを作成した後、イベント・アナライザーを使用してパフォーマンス情報を見ることができます。

イベント・モニター・ツールによって、以下のことを行えます。

- イベント・モニターを作成し、自分が興味のあるデータベース・イベントのタイプをモニターする。
- イベント・モニターを活動化してイベント・データの収集を開始する。データはファイルに保管されます。
- イベント・データの収集から、イベント・モニターを停止する。
- イベント・モニターが作成するトレース・タイプの要約情報を表示する。
- イベント・モニターが必要なくなったときに除去する。トレース・ファイルをクリーンアップするオプションもあります。
- データベースに関連したイベント・モニターのリストを表示する。
- イベント・モニターの定義を表示する。

以下のイベント・タイプについてイベント・モニターが生成するデータを、イベント・アナライザーによって表示できます。

- データベース接続活動 (接続と切断の間の期間)
- トランザクション (作業単位)
- SQL ステートメントの実行
- デッドロック活動の検出

イベント・アナライザー

以下のイベント・タイプ用に、イベント・モニターを作成してから、イベント・アナライザーを使って収集された情報を表示できます。ただし、db2evmon 実行可能ファイル (*コマンド解説書* およびシステム・モニター *手引きおよび解説書* で説明) を使って、以下のイベント・タイプについて生成されたデータを表示します。

- デッドロック
- データベース活動
- 表スペース活動
- 表活動
- ステートメント活動

イベント・モニターおよびイベント・アナライザーを使用してイベント・データを解析するには、以下のステップを実行します。以下のステップは、接続およびステートメント・イベントについて、イベント・モニターを作成する方法の一例に過ぎません。イベント・モニターを作成するには、以下のことを行ってください。

1. コマンド・センターのコマンド行から、 **db2emcrt** とタイプします。「イベント・モニター (Event Monitor)」ウィンドウがオープンします。
2. **「イベント・モニター (Event Monitor)」**をクリックし、メニューから「作成 (Create)」を選択します。「イベント・モニターの作成 (Create Event Monitor)」ウィンドウがオープンします。
3. フィールドで、作成しているイベント・モニターの名前を指定します。この新しいイベント・モニターを、既存のモニターと同じ名前にすることはできません。名前にブランクのスペースを入れることはできません。
4. DB2 ユニバーサル・データベース エンタープライズ拡張エディション製品の場合に限って、「ノード上 (On Node)」ドロップダウン・リストから、イベント・モニター・ファイルが入れられるノードを選択します。
5. DB2 ユニバーサル・データベース エンタープライズ拡張エディション製品の場合に限って、イベント・モニターが入れられる効力範囲を選択します。 デフォルトでは、効力範囲はグローバルです。
6. モニターしたいイベントのタイプを示すチェック・ボックスを 1 つ以上選択します。デッドロック・イベント・タイプがデフォルトで選択されていることに注意してください。
7. このモニターを開始するときを示します。「今開始する (Start now)」がデフォルトで選択されていることに注意してください。
8. これらのレベルでモニターを制御する接続、ステートメント、またはトランザクションに 1 つ以上の条件を定義します。
9. モニターがイベント・データ・ファイルを作成するパス (ディレクトリー名) を確認します。
10. **「オプション (Options)」**をクリックして、**「イベント・モニター・ファイルの指定 (Specifying Event Monitor File)」**オプションのウィンドウをオープンします。これらのオプションは、モニター出力が扱われる方法、およびイベント・モニターのパフォーマンスへの影響を決定します。
11. 「OK」をクリックしてモニターを作成するか、または「取消 (Cancel)」をクリックしてモニターを作成しないで終了します。
12. イベント・モニターをオフにするには、イベント・モニターで右マウス・ボタンをクリックし、ポップアップ・メニューから**「イベント・モニターの停止 (Stop Event Monitoring)」**を選択します。

    これによってイベント・モニターは強制的にトレース・ファイルに書き込まれます。モニターがオフになっていないと、情報はバッファーがいっぱいになったとき、またはすべての接続が終了するときにディスクに書き込まれるだけです。「イベント・モニター (Event Monitors)」ウィンドウから、作成したイベント・モニターで右マウス・ボタンをクリックし、ポップアップ・メニューから**「イベント・モニター・ファイルの表示 (View Event Monitor Files)」**を選択することによって、結果のイベント・データを表示することができます。「モニター期間表示 (Monitored Periods View)」ウィンドウがオープンします。

イベント・アナライザーからイベント・データにアクセスする方法は次のとおりです。

1. コマンド・センターのコマンド行から、 **db2eva** とタイプしてイベント・アナライザーを開始します。「イベント・アナライザー (Event Analyzer)」ウィンドウがオープンします。
2. 「パス (Path)」フィールドで、データ・ファイルが保管されるパス (ディレクトリー名) を確認します。ファイルが移動されていない場合、イベント・モニターの作成時に指定されたパスです。ファイルが移動された場合、そのディレクトリーを指定してください。 ... をクリックすると、既存のディレクトリーがリストされます。

   注: データ・ファイルがリモートに保管される場合、それらを表示するには、ローカル・マシンに FTP でファイル転送する必要があります。ファイルのサイズによっては、この転送にはかなり時間がかかる場合があります。ファイルは、どのローカル・パスにも転送できます。ファイルが作成されたときに使用されたのと同じパスを選択する必要はありません。
3. **OK** をクリックして、ディレクトリーに含まれるデータ・ファイルにアクセスするか、または**「取消 (Cancel)」** をクリックして終了します。 「モニター期間表示 (Monitored Periods View)」ウィンドウがオープンします。
4. モニターされる期間で右マウス・ボタンをクリックし、ポップアップ・メニューから**「オープン -> 接続 (Open as —> Connections)」**を選択します。「接続表示 (Connections View)」ウィンドウがオープンします。このウィンドウは、イベント・モニター・セッション中に作成された接続のリストを表示します。 (リストされる接続は複数個あるかもしれません。今問題の接続がリストの最初にあるとは限りません。)
5. 接続で右マウス・ボタンをクリックし、ポップアップ・メニューから**「オープン -> ステートメント (Open as —> Statements)」**を選択します。「SQL ステートメント表示 (SQL Statements View)」ウィンドウがオープンします。選択された接続についての、すべてのステートメントを表示します。情報の列が各ステートメントごとに提供されます。以下の情報が含まれます。
   - **操作**
   - **パッケージ名**
   - **作成者**
   - **開始時刻**
   - **経過時間**
   - **合計 CPU 時間**
   - **テキスト**

イベント・モニターおよびイベント・アナライザーのオンライン・ヘルプでは、イベント・モニターの作成および結果のイベント・データの表示について詳細に説明しています。

SQL ステートメントの解析

Explained SQL ステートメントのアクセス・プランをグラフとして表示し、パフォーマンスを向上させるためにこの情報を使って SQL 照会をチューニングできます。

アクセス・プラン・グラフは、以下の詳細を表示します。

- 表 (さらに関連する列) および索引
- 演算子 (表走査、分類、および結合など)
- 表スペースおよび関数

バージョン 6 の前は、アクセス・プランを表示するのに、 Visual Explain というツールを使用していました。現在では、コマンド行から別個のツールとして Visual Explain を起動することはできません。しかし、コントロール・センター、およびコマンド・センターにあるさまざまなデータベース・オブジェクトから、 Visual Explain *機能* を呼び出すことができます。この節では、この機能を説明するのに *Visual Explain 機能* という用語を使用します。

Visual Explain 機能を使用して、以下のことを行えます。

- 最適化を行う際に使用した統計の表示。次に、それらの統計を現行のカタログ統計と比較すると、パッケージの再バインドがパフォーマンスを改善するかどうかを判別することができます。
- 表のアクセスに索引を使用したかどうかの判別。索引を使用しなかった場合、Visual Explain 機能は、索引化を行うとどの列の役に立つかを判別するのに役に立ちます。
- さまざまな調整技法を実行した効果の表示。アクセス・プラン・グラフの照会前のバージョンと照会後のバージョンを比較することにより、行います。
- アクセス・プランでの各操作に関する情報の獲得。これにはコストの見積合計や検索される行数 (カーディナリティー) が含まれます。

照会のパフォーマンスの向上

Visual Explain 機能を使用して、SQL ステートメントを解析およびチューニングします。 Explain が実行された SQL ステートメントについて、アクセス・プランのグラフィカルなビューを提供します。表および索引、さらにそこで行われる操作がノードとして表され、データの流れがノード間のリンクによって表されます。このグラフの使用可能な情報を使用して、より良いパフォーマンスが得られるように SQL 照会をチューニングする方法を見つけることができます。

Visual Explain 機能は、 SQL ステートメントがコンパイルされる方法についての情報を取り込みます。この情報を使用して、SQL ステートメントのプランや、潜在的な実行のパフォーマンスを理解することができます。

この情報は次のような場合に役に立ちます。

- アプリケーション・プログラムの設計。

- データベースの設計。
- 2 つの表が結合される方法。使用される結合方法、表が結合される順序、ソートのオカレンスおよびソートのタイプ。
- SQL ステートメントのパフォーマンスを改善する方法を判別する (たとえば、新しい索引を作成することによって)。
- 最適化を行う際に使用した統計の表示。次に、それらの統計を現行のカタログ統計と比較すると、パッケージの再バインドがパフォーマンスを改善するかどうかを判別することができます。また、収集している統計がパフォーマンスを改善するかどうか判別することもできます。
- 表のアクセスに索引を使用したかどうかの判別。索引を使用しなかった場合、Visual Explain 機能は、照会パフォーマンスを改善するために、どの列を索引に含められるかを判別するのに役立ちます。
- パフォーマンスを向上させる目的でさまざまな調整技法を実行した効果を表示する。アクセス・プラン・グラフの照会前のバージョンと照会後のバージョンを比較することにより、これを行います。
- アクセス・プランでの各操作に関する情報の獲得。これにはコストの見積合計や検索される行数が含まれます。

Explained SQL ステートメントのアクセス・プランを Visual Explain 機能を使用して把握すれば、索引を使用することが照会のパフォーマンス向上につながるかどうかを判断できる場合があります。その照会用の推奨索引を受け取るには、索引ウィザードを使用してください。あるいは、RECOMMENDED_INDEXES EXPLAIN モードを使用することもできます。索引ウィザードについての詳細は、コントロール・センターからインフォメーション・センターにアクセスしてください。

RECOMMENDED_INDEXES EXPLAIN モードの詳細については、 *管理の手引き: パフォーマンス* を参照してください。

単純な動的 SQL ステートメントの解析

この節では、動的 SQL 照会の解析を開始する、簡単な例を示します。

1. コントロール・センターから、SAMPLE データベースで右マウス・ボタンをクリックし、ポップアップ・メニューから**「SQL の Explain (Explain SQL)」**を選択します。「SQL ステートメントの Explain (Explain SQL Statement)」ウィンドウがオープンします。
2. **「SQL テキスト (SQL text)」**フィールドに、次の SQL ステートメントを入力します。

   ```
   select * from staff order by name
   ```

3. **「OK」**をクリックします。「アクセス・プラン・グラフ (Access Plan Graph)」ウィンドウがオープンします。グラフは、最適化プログラムが照会の結果を適用するために最も効果的であるとして選択したパスを表します。

4. 任意選択: いずれかのノードをダブルクリックします (たとえば、RETURN 操作ノード)。「演算子詳細 (Operator Details)」ウィンドウがオープンし、その演算子の詳細を表示します。

Explained SQL ステートメントは自動的に保管されます。これを後で表示する方法は次のとおりです。

1. コントロール・センターから、SAMPLE データベースで右マウス・ボタンをクリックし、ポップアップ・メニューから**「Explain 済みステートメント履歴の表示 (Show explained statements history)」**を選択します。「Explain 済みステートメント履歴 (Explained Statements History)」ウィンドウがオープンします。
2. 必要な項目を見つけます。**「SQL テキスト (SQL text)」**列で、前に Explained SQL ステートメントを見つけることができます。
3. その項目で右マウス・ボタンをクリックし、ポップアップ・メニューから**「アクセス・プランの表示 (Show access plan)」**を選択します。「アクセス・プラン・グラフ (Access Plan Graph)」ウィンドウがオープンします。

Visual Explain 機能のオンライン・ヘルプ (**「ヘルプ (Help)」**メニューからアクセス可能) は、 SQL ステートメントのパフォーマンスを改善するために、「アクセス・プラン・グラフ (Access Plan Graph)」ウィンドウを解釈する方法を詳しく説明します。 また、オンライン・ヘルプには、Visual Explain の使用法を学ぶのに役立つ、詳細な例も含まれています。

リモート・データベースの管理

ここでは、次のことを実行する方法を示します。

- リモート・システムの追加
- そのシステムに関して処理したいインスタンスの追加
- そのインスタンスの下で処理したいデータベースの追加

DB2 はまず、ノード・ディレクトリー (データベース・クライアントが接続できるすべてのサーバーのエントリー、および接続で使用される通信プロトコルを含む) を検査し、リモート・システムがすでに認識されているかどうかを確認します。リモート・システムがリモート・システム上のシステム、インスタンス、またはデータベースに認識されていない場合、クライアントとしてリモート・システムに自分自身を設定する必要があります。

DB2 のインストール後、クライアント構成アシスタントを使って、システム、インスタンス、およびデータベースを検索し、それらに通信を構成することができます。次に、カタログ化することによってリモート・システムを追加します。これによって、ノード・ディレクトリーにシステムのエントリーが作成され、インスタンスとデータベースを認識させることができます。次に、インスタンスとデータベースをカタログ化することによって、システムにそれらを追加し、それぞれノード・ディレクトリーとデータベ

ース・ディレクトリーにエントリーを作成する必要があります。これによって、それぞれノード・ディレクトリーおよびデータベース・ディレクトリーにエントリーが作成されます。構成が完了すると、リモート・システムがコントロール・センターに表示され、それらを対象に作業することができます。

リモート・システムを追加する方法は次のとおりです。

1. コントロール・センターから、**「システム (Systems)」**オブジェクトで右マウス・ボタンをクリックし、ポップアップ・メニューから**「追加 (Add)」**を選択します。「システムの追加 (Add System)」ウィンドウがオープンします。
2. **「システム名 (System name)」**フィールドに、システム名を入力します。

   インスタンスの **Discover** 構成パラメーターが、 **search** に設定されており、**discover comm** 構成パラメーターがブランクでない場合、**「最新表示 (Refresh)」**を選択して、リモート・システムのリストを入手できます。次に、**「システム名 (System name)」**フィールドの下のリストから、システムのうち 1 つを選択することができます。
3. **「リモート・インスタンス名 (Remote instance name)」**フィールドに、リモート・インスタンス名を入力します。
4. **「オペレーティング・システム (Operating system)」**リストから、リモート・システムに合ったオペレーティング・システムのタイプを選択します。
5. リモート・ロケーションでの通信に使用される、必要なプロトコルを選択します。ローカル・システムについては、**「ローカル (Local)」**が唯一の有効なプロトコルなので、自動的に選択されます。リモート・システムの場合の可能なプロトコルは次のとおりです。
   - APPC
   - IPX/SPX
   - NetBIOS
   - TCP/IP
   - 名前付きパイプ (Windows NT および Windows 9x オペレーティング・システム専用)

   コンピューターが現在セットアップされているプロトコルだけが、リスト・ボックスに表示されます。
6. 適切なプロトコル・パラメーターを入力します。
7. システムに関連したコメントを入力します。
8. **「適用 (Apply)」**をクリックして、ノード・ディレクトリーにシステムを追加します。

次に、そのシステムに関して処理したいインスタンスを追加します。

1. コントロール・センターから、今追加したシステムに所属する**「インスタンス (Instances)」**オブジェクトで右マウス・ボタンをクリックします。

2. ポップアップ・メニューから、**「追加 (Add)」**を選択します。「インスタンス追加 (Add Instance)」ウィンドウがオープンします。
3. フィールドに必要な値を入力します。
4. **「最新表示 (Refresh)」**押しボタンをクリックし、既存のインスタンスのリストを表示します。
5. 処理したいインスタンスを選択します。
6. **「適用 (Apply)」**押しボタンをクリックしてから、**「クローズ (Close)」**押しボタンをクリックします。

最後に、そのインスタンスの下で処理したいデータベースを追加します。

1. コントロール・センターから、**「データベース (Databases)」**オブジェクト上で右マウス・ボタンをクリックします。
2. ポップアップ・メニューから、**「追加 (Add)」**を選択します。「データベースの追加 (Add Database)」ウィンドウがオープンします。
3. データベース名、通信プロトコルのタイプ、および任意選択で、別名をタイプします。この場合の別名は、データベースを識別するのに使用される代替名です。
4. **「最新表示 (Refresh)」**押しボタンをクリックし、そのインスタンスでの既存のデータベースのリストを表示します。
5. データベースを選択します。
6. **「適用 (Apply)」**押しボタンをクリックしてから、**「クローズ (Close)」**押しボタンをクリックします。

ユーザーの管理

データベース管理者として、場合によってはデータにアクセスする人のタイプを制御したり、データの表示を制限したりする必要があります。以下の情報は、管理ツールを使用してデータベース権限およびデータベース・オブジェクトの特権を使用する方法を説明します。

データベース*権限* には、データベース全体に対するアクションが関係しています。データベース作成時に、いくつかの権限が、データベースにアクセスするすべての人に自動的に付与されます。たとえば、CONNECT、CREATETAB、BINDADD および IMPLICIT_SCHEMA 権限は、すべてのユーザーに付与されます。データベース*特権* には、データベース内の特定のオブジェクトに対するアクションが関係しています。データベース作成時に、いくつかの特権が、データベースにアクセスするすべての人に自動的に付与されます。たとえば、SELECT 特権がカタログ視点で付与され、正常にバインドされた各ユーティリティーでは、 EXECUTE および BIND 特権がすべてのユーザーに付与されます。

特権と権限は共に、インスタンスおよびそのデータベース・オブジェクトへのアクセスを制御するために働きます。ユーザーがアクセスできるのは、該当する許可、つまり、必須の特権や権限を持っているオブジェクトに限られます。

権限と特権の授与および取り消し

DB2 管理ツールを使用して、データベース、表スペース、表、視点、およびスキーマについて、ユーザーおよびグループに対して、特権を授与したり取り消したりすることができます。

1. コントロール・センターから、特権を授与または取り消したいデータベース、表、視点、スキーマ、または索引で右マウス・ボタンをクリックします。ポップアップ・メニューから、**「権限 (Authorities)」**または**「特権 (Privileges)」**を選択します。「権限 (Authorities)」ウィンドウまたは「特権 (Privileges)」ウィンドウがオープンします。
2. **「ユーザー (User)」**ページを選択してユーザー権限または特権を扱うか、または**「グループ (Group)」**ページを選択してグループ権限または特権を扱います。
3. 1 つ以上のユーザーまたはグループを選択します。ユーザーまたはグループをリストに追加するには、**「ユーザーの追加 (Add User)」**、または**「グループの追加 (Add Group)」**押しボタンをクリックしてください。
4. ウィンドウの下部に沿って、**「はい (Yes)」**、**「いいえ (No)」**、または**「授与 (Grant)」**を個々の権限または特権ごとに選択します。**「授与 (Grant)」**は、有効なオプションがあるオブジェクトにのみ表示されます。
5. 終了したら、**「適用 (Apply)」**押しボタンをクリックします。

特定ユーザーが権限を持つオブジェクトを検討または変更したい場合、ユーザーを選択して右マウス・ボタンをクリックしてから、オブジェクトに権限を追加または変更するか、または権限を除去します。

データの移動

DB2 は、データを表から既存のソースに移動するのに役立つ、インポート、エクスポートおよびロード・ユーティリティーを備えています。この節で説明される情報は、データ移動の簡単な概説です。データの移動の詳細については、 *データ移動ユーティリティー 手引きおよび解説書* を参照してください。

インポート・ユーティリティーは、入力ファイルからデータを取り、表または視点に挿入するためのものです。 この場合、入力ファイルには、ロータス 1-2-3 ファイルまたは ASCII ファイルなどの既存のデータのソースから抽出されたデータが含まれます。また、インポート・ユーティリティーを使用して、エクスポート・ユーティリティーで保管された表または視点を再作成することもできます。 以下の情報は、データのインポート方法を説明しています。

サポートされる形式で入力ファイルを使用可能にしてから、インポート・ノートブックを使用して、ファイルから既存の表にデータを挿入します。表にすでにデータが入っている場合は、既存データを置き換えるか、ファイル内のデータに追加するかのどちらかを行うことができます。

また、インポート・ノートブックは、新しい表を作成して、入力ファイルからデータを読み込んだり、選択された表の既存の行を削除し、入力ファイルからのデータを再度読み込んだりするために使用することもできます。

ファイルを既存の表にインポートする方法は次のとおりです。

1. インポート・ノートブックの「ファイル (File)」ページをオープンします。
2. 任意選択。インポート・ノートブックを指定します。
3. 任意選択。ラージ・オブジェクトを検索します。
4. 任意選択。列インポート・オプションを指定します。
5. **「OK」**をクリックします。

インポート・ノートブックの「ファイル (File)」ページをオープンする方法は次のとおりです。

1. コントロール・センターからオブジェクト・ツリーを展開し、**「表 (Table)」**フォルダーを見つけます。
2. **「表 (Tables)」**フォルダーをクリックします。既存の表がコンテンツ・ペインに表示されます。
3. コンテンツ・ペインの表で右マウス・ボタンをクリックし、ポップアップ・メニューから**「インポート (Import)」**を選択します。インポート・ノートブックが表示され、「ファイル (File)」ページが表示されます。

ファイル・オプションを指定する方法は次のとおりです。

1. 「ファイル (File)」ページの**「ファイルのインポート (Import file)」**で、インポートしたいデータが入っているファイル名を入力します。
2. 次のうちどれかを選択して、インポートするファイルのタイプを指定します。
   - 区切りなし ASCII 形式 (ASC)

     区切りなし ASCII データは、列で位置合わせされるデータです。
   - 区切り付き ASCII 形式 (DEL)

     区切り付き ASCII データは、列値がユーザー定義の区切り文字 (たとえばコンマなど) で分離されているようなデータを保管する場合に使用される一般的な方法です。
   - ワークシート形式 (WSF)
   - 統合交換フォーマット (IXF)

PC/IXF は、データベース表または視点の構造化された記述です。 PC/IXF 形式でエクスポートされたデータは、別の DB2 ユニバーサル・データベースの製品データベースにインポート、またはロードすることができます。

サポートされる特定の製品およびリリースについては、オンライン・ヘルプを参照してください。

3. 任意選択: 対応する**「オプション (Options)」**押しボタンをクリックして、ファイル・タイプ修飾子を指定します。その形式の「オプション (Options)」ウィンドウがオープンします。
4. **「インポート・モード (Import mode)」**を選択します。使用可能なインポート・モードは、選択したファイル・タイプによって異なります。
5. 任意選択: **「レコードのコミット (Commit records)」**フィールドで、変更がコミットされる前にインポートするレコード数を入力します。
6. 任意選択: **「再始動 (Restart)」**フィールドで、インポート・アクションが開始する前にスキップするファイルのレコード数を入力します。
7. 任意選択: **「複合 (Compound)」**フィールドで、 (実行可能ブロックで) 実行される SQL ステートメント数を指定する数を入力します。
8. 任意選択: **「10 進数データで暗黙の 10 進小数点を挿入する (Insert an implied decimal point on decimal data) (IMPLIEDDECIMALPOINT)」**チェック・ボックスを選択します。
9. **「メッセージ・ファイル (Message file)」**フィールドで、インポート中に発生する警告およびエラー・メッセージを入れるファイル名を入力します。

別のファイルからラージ・オブジェクトを検索するには、インポート・ノートブックの「ラージ・オブジェクト (Large Objects)」ページを使用して、 LOB ファイルを保管するパス (複数の場合もある) からラージ・オブジェクト (LOB) を検索します。

1. **「LOBs (Retrieve large objects) in separate files (大規模オブジェクト (LOB) を分離ファイルで検索する) (LOBSINFILE)」**チェック・ボックスをクリックして、「ラージ・オブジェクト (Large Objects)」ページを使用可能にします。
2. **「追加 (Add)」**押しボタンをクリックして、 LOB パス・リスト・ボックス内の別の LOB ファイルの位置を指定します。これらのパスで、 (**「LOB パス (LOB paths)」** リスト・ボックスで表示される順序で) 入力ファイルの LOB 列で指定された LOB ファイルが検索されます。
3. **「OK」**をクリックして他のノートブックのデフォルトを受け入れ、インポート・プロセスを開始します。

列インポート・オプションを指定します。**インポート**・ノートブックの**「列 (Columns)」**ページを使って、列インポート・オプションを指定します。

1. **「列の組み込み元 (Include columns by)」**ボックスのラジオ・ボタンから 1 つをクリックし、表にデータ・ファイル列をインポートするのに使用される列方式を指定します。使用可能な方式は、「ファイル (File)」ページで選択したファイル・タイプとモードによって異なります。
2. 任意選択: **「変更 (Change)」**押しボタンをクリックして、インポート・ファイルの列属性を指定または変更します。

   このオプションは、**「デフォルト (D 方式) (Default (method D))」**ラジオ・ボタンを選択した場合は使用できません。

ストレージの管理

データベース管理者として、表および索引のサイズを見積もったり、既存の表がいっぱいになったときにスペースをさらに追加して、表スペースで使用可能なスペースの量を検査したりすることが必要です。

この節では、次のことを行う方法を説明します。

- 表および索引のサイズの見積もり
- 表スペースで使用可能なスペースの量の検査
- 既存の表スペースがいっぱいになり始めたときにスペースを追加する

表および索引サイズの見積もり

新規の、または既存の表または索引に必要なストレージの量は、「サイズの見積もり (Estimate Size)」ダイアログを呼び出すことによって見積もることができます。このダイアログは、個々の表および索引を選択してそこで右マウス・ボタンをクリックして呼び出すか、または「表の作成 (Create Table)」および「索引の作成 (Create Index)」ウィンドウから、**「サイズの見積もり (Estimate Size)」**を選択します。サイズは、特定の表の定義およびその従属索引について見積もられます。見積もりは、表に行数が指定されている場合に使用される、計画された量です。可変長フィールドの最小および最大サイズに基づいて、最小および最大のスペースも見積もられます。表または索引で起動されると、**「サイズの見積もり (Estimate Size)」**ダイアログには、表の仕様が事前に記入され、表およびその表に従属するすべての索引に関連する数が入ります。**「最新表示 (Refresh)」**押しボタンをクリックすると、見積サイズ、最小サイズ、および最大サイズが、**「行の新規合計数 (New total number of rows)」**、および**「新規の平均の行の長さ (New average row length)」**フィールドで入力した数に基づいて更新されます。

表または索引のサイズの見積もりは、以下のことを実行したい場合に役立ちます。

- 新しい表を作成するのに、表スペースの大きさを知りたい場合。
- 既存の表のサイズの見積もりに基づいて新しい表を作成する場合。
- システムのストレージがなくなり始めているので、表スペース中の異なる表および索引オブジェクトが使用するスペースを知りたい場合。
- データのロードに先立って、表の計画サイズを見積もりたい場合。

注: DB2 ユニバーサル・データベース エンタープライズ拡張エディション製品でサイズの見積もりを使用する場合、サイズの見積もりはデータベース区画ではなく、表内のデータの論理サイズに基づきます。

ある期間表の統計を更新していない場合、**「統計の実行 (Run statistics)」**押しボタンをクリックして、選択された表の統計を更新することができます。 索引を選択してから**「統計の実行 (Run statistics)」**ボタンをクリックすると、統計は関連した表で実行されます。

表のサイズを見積もる方法は次のとおりです。

- 「サイズの見積もり (Estimate Size)」ウィンドウをオープンします。
- **「新規合計行数 (New total number of rows)」**に異なる値を選択するか、またはデフォルトを受け入れます。
- **「最新表示 (Refresh)」**をクリックして、新しい値のサイズの見積もりを表示します。
- **「新規の平均の行の長さ (New average row length)」**に異なる値を選択するか、またはデフォルトを受け入れます。
- **「最新表示 (Refresh)」**をクリックして、新しい値のサイズの見積もりを表示します。

表スペースで使用可能なスペースの検査

DMS 表スペースで使用可能なスペースの量を検査する方法は次のとおりです。

1. コントロール・センターから、**「表スペース (Table Spaces)」**アイコンをダブルクリックします。 すべての表スペースのリストがコンテンツ・ペインに表示されます。
2. **「割り振りサイズ (Allocated size)」**、**「使用サイズ (Size used)」**、および**「使用パーセント (Percentage used)」**というタイトルの列にスクロールし、表スペースで使用可能なスペースの量に関連する詳細を表示します。 スペースは、1 ページが 4 KB のページ数で測定されます。

列の順序および表示される列を、コンテンツ・ペインの下部にある、「列のカスタマイズ (Customize Columns)」アイコンを使用してカスタマイズできます。

SMS 表スペースで使用できるスペースを検査するには、オペレーティング・システムによって提供される機能を使ってスペース使用量をモニターし、表スペース用のディレクトリーにある使用可能なスペースがなくならないようにします。

表スペースにスペースを追加する

DMS 表スペースの容量は、表スペースに割り当てられたコンテナーの合計サイズです。 DMS 表スペースが容量に達した場合 (表スペースの使用量によっては、 90% が可能な限界値)、スペースを追加することが必要です。データベース・マネージャーは、

使用可能なコンテナー全体で、自動的に DMS 表スペース中の表のバランスを取り直します。再バランス中、表スペースのデータは引き続きアクセス可能です。

容量に達した DMS 表スペースについては、別のコンテナーを追加できます。

1. コントロール・センターから、コンテナーに追加したい表スペースをコンテンツ・ペインで探して右マウス・ボタンをクリックし、ポップアップ・メニューから**「更新 (Alter)」**を選択します。「表スペースの更新 (Alter Table Space)」ウィンドウをオープンします。
2. **「追加 (Add)」**をクリックします。「コンテナーの追加 (Add Container)」ウィンドウがオープンします。
3. **「ファイル (File)」**または**「ロー・デバイス (Raw device)」**ラジオ・ボタンを選択し、フィールドを完成させます。詳細は、オンライン・ヘルプを参照してください。
4. **「OK」**をクリックします。

一般的には、SMS 表スペースのサイズはあまり容易に拡張することはできません。これは、SMS 容量が、ファイル・システムで使用可能なスペース、およびオペレーティング・システムがサポートするファイルの最大サイズに依存しているためです。しかし、オペレーティング・システムによっては、オペレーティング・システム機能を使ってファイル・システムのサイズを増やすことができる場合があります。 UNIX ベースのシステムの SMS 表スペースの場合は、適切な UNIX ベースのシステム・コマンドを使用して、表のサイズを大きくすることができます。実行中の UNIX ベースのシステムのマニュアルを参照してください。 SMS 表スペースを含むファイル・システムに非 DB2 ファイルも含まれる場合、これらのファイルを別のファイル・システムに移動して、 DB2 が使用するためのファイル・システムに使用可能なスペースを広げることができます。また、表スペースをバックアップ元よりも多くのコンテナーに復元することを意味する、リダイレクト復元を実行することもできます。リダイレクト復元は、データベース復元ノートブックから実行できます。復元したいデータベースから、ポップアップ・メニューで**「復元 -> データベース (Restore —> Database)」**を選択します。

トラブルシューティング

DB2 には、DB2 サーバーおよびクライアントのテクニカル・サポートのためのトラブルシューティング・マニュアルが付属しています。次の場合に役立ちます。

- 問題またはエラーを簡潔な方法で識別する
- 症状に基づいて問題を解決する
- 使用可能な診断ツールを使用する
- 日常の DB2 操作に合わせてトラブルシューティング戦略を開発する

問題判別の手引き では、これらの基本的なトラブルシューティングに関するトピックを掲載しています。

- トラブルシューティングの良い方法

- サーバーでのトラブルシューティング
- クライアントでのトラブルシューティング
- ホスト通信に関するトラブルシューティング
- アプリケーションに関するトラブルシューティング
- トラブルシューティングと問題判別

問題判別の手引き では、これらの高度なトラブルシューティングに関するトピックを掲載しています。

- DB2 プロセス・モデル
- ログ情報の使用
- トレースの取り方
- Unix ベース、OS/2、および Microsoft Windows オペレーティング・システムの診断ツール

最新情報およびテクニカル資料については、WWW のアドレス
http://www.software.ibm.com/data/db2/library/ で使用可能です。

IBM 社との連絡方法の詳細については、本書の最後の節を参照してください。

データの複製

複製は、ソース・サーバーでデータベース・ログに保管される変更を使用し、それをターゲット・サーバーに適用するプロセスです。複製を使用して、企業内のデータについて、コピー操作を定義し、同期を取り、自動化し、管理することができます。自動的にホスト・システムからターゲット・サイトにデータを送達することができます。たとえば、データおよびアプリケーションを事業所、小売店、また販売担当者のラップトップにまでコピーすることができます。

複製で使用する 2 つの操作可能なコンポーネントは、Capture (取り込み) と Apply (適用) です。 Capture (取り込み) コンポーネントは、複製に定義されているソース表のデータに加えられた変更を、データベース・ログを読み取ることによって取り込みます。Apply (適用) コンポーネントは、変更データ表で以前に取り込まれ保管されたデータを読み取り、それをターゲット表に適用します。

コントロール・センターを使用して、**「複製ソースとして定義 (Define as replication source)」**、および**「加入の定義 (Define subscription)」**アクションを使って、複製に必要なセットアップを実行できます。複製のコンポーネントである Capture (取り込み) と Apply (適用) は、 DB2 管理ツールの外部で実行します。

複製管理者は、コントロール・センターから以下のアクションを実行できます。

- 複製ソースの定義
- 複製サブスクリプションの定義

- 適用プロセス中に SQL を指定してデータを拡張する

データの複製の高水準のステップは、以下のとおりです。詳細は、*レプリケーションの手引きおよび解説書* を参照してください。

1. 複製シナリオを設計します (ソースおよびターゲット表をマップします)。
2. 複製ソースを定義します (これは取り込みアクションに関連します)。

複製ソースを定義する方法は次のとおりです。

1. 取り込むソース列を指定します。
2. 複製オプションを選択します。
3. 複製サブスクリプションを定義します (これは適用アクションに関連します)。
4. ソース表を、データ取り込み変更オプションを使って代替します。
5. Capture (取り込み) を開始し、データ変更を読み取って保管します。
6. Apply (適用) を開始し、ターゲット表への変更を複製します。

複製サブスクリプションを定義する方法は次のとおりです。

1. サブスクリプション・セットに名前を付けます。
2. データベースおよびターゲット表を指定します。
3. ターゲット列を指定します。
4. 行選択を指定します。
5. 実行時処理用の SQL を指定します。
6. 加入タイミングを設定します。

Lightweight Directory Access Protocol (LDAP) の使用

クライアント構成アシスタント (CCA) を使用すれば、 LDAP サーバーにエントリーを追加したり、 LDAP サーバーからエントリーを削除したりできます。 LDAP サーバーに登録されているすべてのデータベース・インスタンスは、クライアント上に自動的にカタログ (キャッシュ) されます。これらのデータベース・インスタンスは、正規のノードとしてナビゲーター・ツリーに表示されます。これらのデータベースは、マシン上でカタログ化された他のデータベースと同じ方法で管理できます (ADD DATABASE オプションはこのリリースではまだ使用可能になっていないので例外)。

LDAP データベースを管理するには、データベースを選択して右マウス・ボタンをクリックします。ポップアップ・ウィンドウに、実行できる機能がリストされます。 LDAP に関する詳細については、421ページの『付録J. Lightweight Directory Access Protocol (LDAP) ディレクトリー・サービス』を参照してください。

Java コントロール・センターの使用

コントロール・センターを Java アプリケーションとして、または Web サーバーを介して Java アプレットとして実行することができます。どちらの場合でも、コントロール・センターを実行するには、 Java 仮想マシン (JVM) をマシン上にインストールすることが必要です。コントロール・センターを Java アプリケーションとして実行するには、正しい Java Runtime Environment (JRE) をインストールすることも必要です。 Java 仮想マシンは、アプリケーションを実行するための Java Runtime Environment (JRE)、またはアプレットを実行するための Java 使用可能ブラウザーのどちらかです。

Java **アプリケーション**は、正しい JRE がインストールされていれば、マシン上の他のアプリケーションと同じように実行されます。

コントロール・センターを Java アプレットとして実行する

Java **アプレット**は、 Java 使用可能ブラウザー内で実行されるプログラムです。コントロール・センターのアプレット・コードは、リモート・マシンに存在でき、 Web サーバーを介してクライアントのブラウザーで使用されます。コントロール・センターを Java アプレットとして実行する場合、 Windows 32 ビットまたは OS/2 オペレーティング・システムで実行する、サポートされる Java 使用可能ブラウザーを使用する必要があります。現在、UNIX オペレーティング・システムでサポートされるブラウザーはありません。

コントロール・センター JDBC アプレット・サーバーは、アプレット・サーバーが存在するマシン上に管理者権限を持つユーザー・アカウントで開始しなければなりません。コントロール・センター JDBC アプレット・サーバーを、起動時に自動的に開始するように設定できます。

コントロール・センターを Java アプレットとして実行するには、コントロール・センターのアプレット・コードを持つマシンと、コントロール・センター JDBC アプレット・サーバー上に、 Web サーバーをセットアップしておく必要があります。 Web サーバーは、`sqllib` ディレクトリーへのアクセスを許可する必要があります。仮想ディレクトリーの使用を選択する場合、このディレクトリーをホーム・ディレクトリーと置き換えてください。たとえば、仮想ディレクトリーの名前を `temp` にした場合、`sqllib/temp` というディレクトリーを使用する必要があります。 DB2 は、OS/2 の FAT ドライブではコントロール・センターのインストールをサポートしていません。これは、OS/2 FAT ドライブが、Java で必要な長いファイル名をサポートしていないためです。コントロール・センターを Java アプリケーションまたは Java アプレットとしてインストールおよび構成することについての詳細は、各プラットフォームに合った版の *概説およびインストール* を参照してください。

Java ツールを管理に使用する

バージョン 6 では、DB2 にはコントロール・センターの機能を拡張する、 Java インターフェースのセットが含まれています。 Java インターフェースによって、次のことが可能です。

- オブジェクトの処理中、メニュー・リストに付加的な項目を追加する。
- コントロール・センターのツールバーに、「追加 (Add)」ボタンを追加する。

この機能を使用するには、正しいレベルの Java ソフトウェアをインストールする必要があります。この機能の詳細については、455ページの『付録K. コントロール・センターの拡張』を参照してください。

第2部 設計のインプリメント

第2章 データベースを作成する前に

データベースの設計が決定したら、データベースとその中のオブジェクトを作成しなければなりません。オブジェクトには、スキーマ、ノード・グループ、表スペース、表、視点、ラッパー、サーバー、ニックネーム、タイプのマッピング、関数のマッピング、別名、ユーザー定義タイプ (UDT)、ユーザー定義関数 (UDF)、トリガー、制約、索引、およびパッケージがあります。これらのオブジェクトは、コントロール・センターから、またはアプリケーションの中の API を通して、コマンド行プロセッサーの中の SQL ステートメントを使用して作成することができます。

SQL ステートメントについての詳細は、*SQL 解説書* を参照してください。コマンド行プロセッサーについての詳細は、*コマンド解説書* を参照してください。 API についての詳細は、*管理 API 解説書* を参照してください。

注: プラットフォームで、データベース・オブジェクトを作成できるユーザー・インターフェースをサポートしている場合があります。このインターフェースを、SQL ステートメント、コマンド行プロセッサー、または API の代わりに使用することができます。 この機能があるかどうかを判別するために、使用しているプラットフォーム用の*概説およびインストール* を参照してください。

この章では、コントロール・センターを使用してタスクを完了する方法を、囲み線の中に表記して強調しています。同じことをコマンド行から行う方法を、そのすぐ後ろに示します。例が併記されている場合もあります。また、一方の方法しか示されていないタスクもあります。コントロール・センターで作業を行う際には、ヘルプを使用して、この章に記載されている概要よりも詳しい情報を参照できます。

この章では、データベースとそのすべてのオブジェクトを作成する前に知っている必要がある情報を中心に説明します。前提条件となる概念とトピック、およびデータベースの作成前に済ませておくべきタスクを示します。

次の章には、データベース設計のインプリメンテーションを構成する、さまざまなオブジェクトに関する短い説明があります。

この部の最後の章には、データベースを更新する前に考慮しなければならないトピックと、データベース・オブジェクトの更新方法や除去方法について説明されています。

この章やその後の章で説明するトピックの一部は、 DB2 ユニバーサル・データベースがオペレーティング・システムと対話を行う部分で、オペレーティング・システム固有の違いがある場合があります。 DB2 UDB によって提供されるものではなく、ネイティ

ブのオペレーティング・システムの機能および違いを利用することができます。相違点を正確に知るには、該当の*概説およびインストール* および特定のオペレーティング・システムの説明書を参照してください。

1 つの例として、Windows NT は、『サービス』と呼ばれるアプリケーション・タイプをサポートします。 DB2 (Windows NT 版) は、サービスとして定義された DB2 インスタンスを持つことができます。サービスはシステム・ブート時に自動的に開始できますが、これは、サービス制御パネル・アプレットを通してユーザーによって行われるか、または Microsoft Win32 アプリケーション・プログラミング・インターフェース (API) に含まれるサービス機能を使用する Win32 ベースのアプリケーションによって行われます。サービスは、システムにログオンしているユーザーがなくても実行できます。

データベースを作成する前の前提条件

データベースを実現する前に、以下の前提条件タスクについて理解している必要があります。

- 『DB2 の開始』
- 57ページの『Windows NT での DB2 UDB の開始』
- 57ページの『データベース・マネージャーの複数インスタンスを使用する』
- 59ページの『スキーマによるオブジェクトの編成とグループ化』
- 60ページの『並列化の使用可能化』
- 61ページの『データ区分化の使用可能化』
- 64ページの『DB2 の停止』

DB2 の開始

通常の業務を行っている最中に DB2 を開始または停止する必要がある場合があります。たとえば、以下のタスクを実行する前に、インスタンスを開始しなければなりません。

- インスタンスのデータベースに接続する。
- アプリケーションをプリコンパイルする。
- データベースにパッケージをバインドする。
- ホスト・データベースにアクセスする。

システムで DB2 インスタンスを開始するには、次のようにします。

1. インスタンスに対する SYSADM、SYSCTRL、または SYSMAINT 権限を持つユーザー ID またはユーザー名を使ってログインします。あるいは、インスタンス所有者としてログインします。
2. UNIX オペレーティング・システムでは、以下のように開始スクリプトを実行します。

```
. INSTHOME/sqllib/db2profile      (Bourne または Korn シェルの場合)
source INSTHOME/sqllib/db2cshrc   (C シェルの場合)
```

INSTHOME は、使用するインスタンスのホーム・ディレクトリーです。

3. インスタンスを開始するには、以下の 2 つの方法のいずれかを使用します。
   a. コントロール・センターを使用してインスタンスを開始するには、以下のようにします。

1) オブジェクト・ツリーを順に展開し、**「インスタンス (Instances)」**フォルダーを表示します。 2) 開始するインスタンスを右クリックして、ポップアップ・メニューから**「開始 (start)」**を選択します。

   b. コマンド行を使用してインスタンスを開始するには、以下のように入力します。

```
db2start
```

注: db2start コマンドを実行すると、 73ページの『現行インスタンスの設定』にある規則にしたがってインスタンスが開始されます。

Windows NT での DB2 UDB の開始

db2start コマンドを使用すると、DB2 は NT サービスとして立ち上がります。DB2START を呼び出すときにスイッチ ″/D″ を指定すると、 Windows NT での DB2 をプロセスとして実行することができます。「コントロール パネル」または ″NET START″ コマンドを使用して、 DB2 をサービスとして開始することもできます。

DB2START から DB2 をサービスとして正常に立ち上げるには、そのユーザー・アカウントで NT サービスを開始するために Windows NT オペレーティング・システムで定義された、正しい特権が必要です。ユーザー・アカウントは、管理者、サーバー・オペレーター、またはパワー・ユーザーのいずれかのグループのメンバーです。

区分データベースで実行しているときには、各データベース区画サーバーは NT サービスとして開始されます。

データベース・マネージャーの複数インスタンスを使用する

1 つのサーバーにデータベース・マネージャーの複数インスタンスを作成することができます。これはつまり、物理的に 1 つのマシンに同じ製品のインスタンスを複数作成し、それらを並行して稼働させることができるということです。これにより、複数の環境を柔軟に設定できます。

次の環境を作成するために、複数のインスタンスが必要になる場合があります。

- 開発環境を実稼動環境から分離する。
- インスタンスがサービスする特定のアプリケーションのそれぞれを別個に調整する。

- 機密情報を管理担当者から保護する。たとえば、給与計算データベースをそのインスタンスに関して保護し、他のインスタンスの所有者が給与計算データを見ることができないようにすることができます。

DB2 プログラム・ファイルは、物理的には特定のマシンの 1 つのロケーションに保管されます。作成される各インスタンスは、このロケーションを指しているので、作成されるインスタンスごとにプログラム・ファイルが複写されるわけではありません。いくつかの関連するデータベースを、単一のインスタンス内に置くことができます。

インスタンスはノード・ディレクトリーにローカルまたはリモートのいずれかとしてカタログされます。デフォルト・インスタンスは DB2INSTANCE 環境変数で定義されます。データベースの作成、アプリケーションの強制終了、データベースのモニター、またはデータベース・マネージャー構成の更新などの、インスタンス・レベルでしか行えない保守およびユーティリティー・タスクを実行するために、他のインスタンスに*接続*することができます。デフォルト・インスタンスにないインスタンスに接続しようとすると、そのインスタンスとの通信方法を判別するためにノード・ディレクトリーが使用されます。

コマンド解説書 では、各コマンドを実行するために必要な接続のタイプについての情報を提供します。

複数のインスタンスに対する DB2 サポートは、オペレーティング・システムごとに異なります。 1 つのマシン上での複数の DB2 インスタンスの定義については、使用しているプラットフォームに適した*概説およびインストール* を参照してください。

リモートにある別のインスタンスを接続するには、 *コマンド解説書* に説明されているように、ATTACH コマンドを使用します。

コントロール・センターを使用するには、以下のようにします。

1. オブジェクト・ツリーを順に展開し、**「インスタンス (Instances)」**フォルダーを表示します。
2. 接続したいインスタンスをクリックします。
3. 選択したインスタンス名を右クリックします。
4. 「接続 - DB2 (Attach-DB2)」ウィンドウで、ユーザー ID とパスワードを入力し、**「OK」**をクリックします。

コマンド行を使用してインスタンスに接続するには、以下のように入力します。

```
db2 attach to <instance name>
```

たとえば、以下のようにコマンドを出すと、以前にノード・ディレクトリーにカタログ化された testdb2 というインスタンスに接続します。

```
db2 attach to testdb2
```

testdb2 インスタンスの保守に関連した作業を実行した後で、次のコマンドを実行すると、そのインスタンスから*切り離す* ことができます。

```
db2 detach
```

スキーマによるオブジェクトの編成とグループ化

データベース・オブジェクトは、1 つの識別子で構成されているか、 2 つの識別子で構成されるスキーマ*修飾オブジェクト* です。スキーマ修飾オブジェクトのスキーマまたは高位部分は、データベースの中のオブジェクトを分類またはグループ化するための手段を提供します。 表、視点、別名、特殊タイプ、関数、索引、パッケージ、またはトリガーが作成されると、それがスキーマに割り当てられます。この割り当ては、明示的または暗黙的のいずれかで行われます。

ステートメント中のオブジェクトを参照するときに 2 部から成るオブジェクト名の高位部分を使用する場合は、スキーマを明示的に使用することになります。たとえば、USER A がスキーマ C の CREATE TABLE ステートメントを次のように発行します。

```
CREATE TABLE C.X (COL1 INT)
```

2 部から成るオブジェクト名の高位部分を使用する場合は、スキーマを暗黙的に使用することになります。スキーマを暗黙的に使用すると、オブジェクト名の高位部分を構成するのに使用されるスキーマ名を識別するために、 CURRENT SCHEMA 特殊レジスターが使用されます。 CURRENT SCHEMA の初期値は、現行セッション・ユーザーの許可 ID です。これを現行セッション中に変更したい場合は、 SET SCHEMA ステートメントを使用して特殊レジスターを別のスキーマ名に設定することができます。詳しくは、*SQL 解説書* を参照してください。

111ページの『システム・カタログ表の定義』に記載されているとおり、一部のオブジェクトは、データベースが作成されると、特定のスキーマ内に作成されます。

動的 SQL ステートメントでは、スキーマ修飾オブジェクト名は、修飾なしオブジェクト名参照として CURRENT SCHEMA 特殊レジスター値を暗黙的に使用します。静的 SQL ステートメントでは、QUALIFIER プリコンパイル / バインド・オプションは、修飾なしデータベース・オブジェクト名の修飾子を暗黙的に指定します。

独自のオブジェクトを作成する前に、それらをデフォルトのスキーマの中に作成したいか、または論理的にオブジェクトをグループ化する別のスキーマを使用するかを検討しなければなりません。共用されるオブジェクトを作成している場合は、別のスキーマ名を使用すると、非常に便利です。明示的にスキーマを作成する方法について詳しくは、124ページの『スキーマの作成』を参照してください。

並列化の使用可能化

データベース区画内または非区分データベース内で並列化を利用するためには、構成パラメーターを修正しなければなりません。たとえば、区画内並列化を使用して、対称マルチプロセッサー (SMP) マシン上の複数のプロセッサーを利用することができます。

区画内並列化の使用可能化

コントロール・センターを使用して、特定のデータベースまたはデータベース・マネージャー構成ファイルの中の個々の項目の値を調べたり修正したりできます。

また、 GET DATABASE CONFIGURATION コマンドおよび GET DATABASE MANAGER CONFIGURATION コマンドを使用して、特定のデータベースまたはデータベース・マネージャー構成ファイルの中の個々の項目の値を調べることもできます。特定のデータベースまたはデータベース・マネージャー構成ファイルの中の個々の項目を修正するためには、それぞれ UPDATE DATABASE CONFIGURATION コマンドと UPDATE DATABASE MANAGER CONFIGURATION コマンドを使用します。

区画内並列化に影響を与える構成パラメーターには、 *max_querydegree* と *intra_parallel* のデータベース・マネージャー・パラメーター、および *dft_degree* データベース・パラメーターがあります。構成パラメーターについての詳細は、*管理の手引き: パフォーマンス* を参照してください。

区画内照会並列化の使用可能化

区画内照会並列化を行わせるためには、以下のデータベース構成パラメーターとデータベース・マネージャー構成パラメーターを修正する必要があります。

INTRA_PARALLEL
データベース・マネージャー構成パラメーター。このパラメーターの詳細は、*管理の手引き: パフォーマンス* を参照してください。

DFT_DEGREE
データベース構成パラメーター。 DEGREE バインド・オプションおよび CURRENT DEGREE 特殊レジスターに対するデフォルトを提供します。このパラメーターの詳細は、*管理の手引き: パフォーマンス* を参照してください。

DEGREE
静的 SQL に対するプリコンパイルまたはバインドのオプション。詳しくは、*コマンド解説書* を参照してください。

CURRENT DEGREE
動的 SQL に対する特殊レジスター。詳しくは、*SQL 解説書* を参照してください。

構成パラメーターの設定について、およびアプリケーションを並列で処理可能にする方法については、 *管理の手引き: パフォーマンス* の『DB2 の構成』を参照してください。

区画間照会並列化の使用可能化

区画間並列化は、データベース区画の数およびこれらの区画にわたるデータの分配に基づいて、自動的に行われます。

ユーティリティー並列化の使用可能化

この節では、以下のユーティリティーについて、区画内並列化を可能にする方法の概要について説明します。

- ロード
- 索引作成
- データベース / 表スペースのバックアップ
- データベース / 表スペースの復元

ユーティリティーに対する区画間並列化は、データベース区画の数に基づいて自動的に行われます。

ロード: Load ユーティリティーは自動的に並列化を使用できるようにします。または、LOAD コマンドで以下のパラメーターを使うことができます。

- CPU_PARALLELISM
- DISK_PARALLELISM

LOAD コマンドに関する詳細は、 *データ移動ユーティリティー手引きおよび解説書* を参照してください。

AutoLoader: `autoloader.cfg` ファイルで、 LOAD 指定に MODIFIED BY ANYORDER パラメーターを指定することによって、 AutoLoader が複数分割処理を行えるようにすることができます。詳細については、*データ移動ユーティリティー手引きおよび解説書* を参照してください。

索引作成: 索引の作成中に並列化を使用可能にするには、以下のとおりでなければなりません。

- INTRA_PARALLEL データベース・マネージャー構成パラメーターは ON でなければなりません。
- 表は、並列化の益が得られる十分な大きさでなければなりません。
- 1 つの SMP マシンで、複数プロセッサーが使用可能でなければなりません。

CREATE INDEX ステートメントの詳細は、 *SQL 解説書* を参照してください。

データ区分化の使用可能化

区分データベース環境で実行している場合、 CREATE DATABASE コマンドまたは sqlecrea() アプリケーション・プログラミング・インターフェース (API) を使用して、`db2nodes.cfg` ファイル内に存在するどのノードからでもデータベースを作成することができます。 詳しくは、*コマンド解説書* および*管理 API 解説書* を参照してください。

区分データベースを作成する前に、データベースが作成されるインスタンスに対して、ローカル・クライアントとして接続するのか、あるいはリモート・クライアントとして接続するのかを決めなければなりません。次に、そのインスタンスに接続しなければなりません。また、どのデータベース区画がそのデータベースに対するカタログ・ノードになるのかを選択しなければなりません。接続して CREATE DATABASE コマンドを実行するデータベース区画は、その特定のデータベースに対するカタログ・ノード になります。

カタログ・ノードは、すべてのシステム・カタログ表が保管されるデータベース区画です。システム表に対するすべてのアクセスは、このデータベース区画を通して行わなければなりません。連合データベース・オブジェクト (ラッパー、サーバー、ニックネームなど) はすべて、このノードのシステム・カタログに保管されます。

可能であれば、各データベースを別個のインスタンスの中に作成すべきです。これが可能でない場合 (つまり、1 インスタンス当たり複数のデータベースを作成しなければならない場合)、カタログ・ノードを使用可能なデータベース区画に分散させる必要があります。これを行うと、単一データベース区画におけるカタログ情報の競合が削減されます。

注: 他のデータでバックアップに必要な時間が増えてしまうため、定期的にカタログ・ノードのバックアップをとり、 (可能ならば) そこにユーザー・データを書き込むのを避けるべきです。

データベースを作成すると、 db2nodes.cfg ファイルに定義されたすべてのデータベース区画にわたって自動的に作成されます。

システムに最初のデータベースが作成されると、システム・データベース・ディレクトリーが作成されます。これは、作成した他のデータベースについての情報と一緒に追加されます。システム・データベース・ディレクトリーは sqldbdir であり、ホーム・ディレクトリーの下の sqllib ディレクトリーに配置されます。このディレクトリーは、区分データベースを形成するすべてのデータベース区画に対する唯一のシステム・データベース・ディレクトリーであるため、共用のファイル・システム (たとえば、UNIX プラットフォーム上の NFS) に常駐しなければなりません。

また、sqldbdir ディレクトリーに常駐するのは、システム・インテンション・ファイルです。これは sqldbins と呼ばれ、データベース区画が同期を維持できるようにするものです。このファイルも、すべてのデータベース区画にわたって 1 つのディレクトリーしかないため、共用のファイル・システムに常駐しなければなりません。このファイルは、データベースを形成するすべての区画で共用されます。

データ区分化を利用するためには、構成パラメーターを修正しなければなりません。GET DATABASE CONFIGURATION コマンドおよび GET DATABASE MANAGER CONFIGURATION コマンドを使用して、特定のデータベースまたはデータベース・マネージャー構成ファイルの中の個々の項目の値を調べることができます。特定のデータ

ベースまたはデータベース・マネージャー構成ファイルの個々の項目を修正するためには、それぞれ UPDATE DATABASE CONFIGURATION コマンドと UPDATE DATABASE MANAGER CONFIGURATION コマンドを使用します。

区分データベースに影響を与えるデータベース・マネージャー構成パラメーターには、*conn_elapse*、 *fcm_num_anchors*、 *fcm_num_buffers*、 *fcm_num_connect*、 *fcm_num_rqb*、 *max_connretries*、 *max_coordagents*、 *max_time_diff*、 *num_poolagents*、および *stop_start_time* があります。

構成パラメーターについての詳細は、*管理の手引き: パフォーマンス* を参照してください。

データベース / 表スペースのバックアップ

データベースまたは表スペースのバックアップ中に入出力並列化を使用可能にするには、以下のようにします。

- 複数のターゲット媒体を使用します。
- 並列入出力用に表スペースを構成します。
- BACKUP コマンドで PARALLELISM パラメーターを使用し、並列化の度合いを指定します。
- BACKUP コマンドで WITH num-buffers BUFFERS パラメーターを使用し、並列度に見合う十分なバッファーを使用できるようにします。バッファーの数は、宛先の媒体、選択した並列度、そして少しの余分を合計した数に等しいものにします。

  また以下のような、バックアップ・バッファー・サイズを使用します。

  – 可能な限り大きなサイズにする。 4 MB か 8 MB (1024 か 2048ページ) が良いようです。
  – 少なくとも、バックアップする表スペースの可能な最大数を含められるだけの大きさにする (extentsize * コンテナー数)。

BACKUP DATABASE コマンドの詳細は、*コマンド解説書* を参照してください。

データベース / 表スペースの復元

データベースまたは表スペースの復元中に入出力並列化を使用可能にするには、以下のようにします。

- 複数のソース媒体を使用します。
- 並列入出力用に表スペースを構成します。
- RESTORE コマンドで PARALLELISM パラメーターを使用し、並列化の度合いを指定します。
- RESTORE コマンドで WITH num-buffers BUFFERS パラメーターを使用し、並列度に見合う十分なバッファーを使用できるようにします。バッファーの数は、宛先の媒体、選択した並列度、そして少しの余分を合計した数に等しいものにします。

  また以下のような、復元バッファー・サイズを使用します。

– 可能な限り大きなサイズにする。 4 MB か 8 MB (1024 か 2048ページ) が良いようです。
– 少なくとも、復元する表スペースの可能な最大数を含められるだけの大きさにする (extentsize * コンテナー数)。
– バックアップ・バッファー・サイズと同じか、その倍数である。

RESTORE DATABASE コマンドの詳細は、*コマンド解説書* を参照してください。

DB2 の停止

db2stop コマンドはサーバーでのみ実行できます。このコマンドの実行中はデータベースの接続は許されません。接続されたインスタンスがあると、DB2 が停止する前に強制的にオフにされます。

注: コマンド行プロセッサーのセッションがインスタンスに接続されたら、**terminate** コマンドを実行し、各セッションを終了してから、 **db2stop** コマンドを実行します。 **db2stop** コマンドを実行すると、 DB2INSTANCE 環境変数で定義されたインスタンスが停止します。

システムで DB2 インスタンスを停止するには、以下のことを行わなければなりません。

1. インスタンスに対する SYSADM、SYSCTRL、または SYSMAINT 権限を持つユーザー ID またはユーザー名を使ってインスタンスにログインする、つまり接続します。あるいは、インスタンス所有者としてログインします。
2. 停止したい特定のデータベースに接続された、すべてのアプリケーションおよびユーザーを表示します。重要なアプリケーションまたはクリティカルなアプリケーションが実行されていないことを確認するために、アプリケーションをリストします。リストを表示するには、SYSADM、SYSCTRL、または SYSMAINT 権限が必要です。
3. すべてのアプリケーションおよびユーザーにデータベースの使用を中断させます。ユーザーに強制するためには、SYSADM または SYSCTRL 権限が必要です。
4. UNIX オペレーティング・システムでは、以下のように開始スクリプトを実行します。

   ```
   . INSTHOME/sqllib/db2profile      (Bourne または Korn シェルの場合)
   source INSTHOME/sqllib/db2cshrc   (C シェルの場合)
   ```

   INSTHOME は、使用するインスタンスのホーム・ディレクトリーです。
5. インスタンスを停止するには、以下のいずれかの方法を使用します。

a. オブジェクト・ツリーを順に展開し、**「インスタンス (Instances)」**フォルダーを表示します。
b. 停止したいインスタンスを 1 つずつクリックします。
c. 選択したすべてのインスタンスを右クリックして、ポップアップ・メニューから**「停止 (stop)」**を選択します。
d. 「停止の確認 (Confirm stop)」ウィンドウで、**「OK」**をクリックします。

コマンド行を使用してインスタンスを停止するには、以下のように入力します。

```
db2stop
```

データベースの作成に関する詳細

データベースを作成する前に、以下の作業を考慮または実行する必要があります。

- 論理および物理データベースの特性の設計
- インスタンスの作成
- 環境変数およびプロファイル・レジストリーの確立
- DB2 管理サーバー (DAS) の作成
- ノード構成ファイルの作成
- データベース構成ファイルの作成
- 応答ファイルを使った構成情報の複製
- FCM 通信の使用可能化

論理および物理データベースの特性の設計

データベースを作成する前に、論理的または物理的なデータベースの設計を決定しなければなりません。論理データベースと物理データベースの設計についての詳細は、 *管理の手引き: 計画* を参照してください。

インスタンスの作成

インスタンスとは、データベースをカタログし、構成パラメーターを設定するための、論理データベース・マネージャー環境です。インスタンスは、必要に応じて複数作成できます。複数インスタンスを使用すると、次のことを行えます。

- 1 つのインスタンスを開発環境用に使用し、別のインスタンスを実稼働環境用に使用する。
- 特定の環境用にインスタンスを調整する。
- 機密情報へのアクセスを制限する。
- それぞれのインスタンスごとに SYSADM、SYSCTRL、および SYSMAINT 権限の割り当てを制御する。

- インスタンスごとにデータベース・マネージャーの構成を最適化する。
- インスタンスの失敗による影響を制限する。インスタンスが失敗した場合、1 つのインスタンスだけが影響を受けます。他のインスタンスは正常に機能し続けます。

注意点として、複数インスタンスには以下のように多少の難点があります。

- インスタンスごとに、追加のシステム・リソース (仮想メモリーとディスク・スペース) が必要になる。
- 追加インスタンスを管理するために余分の管理が必要になる。

インスタンス・ディレクトリーには、データベース・インスタンスに関連するすべての情報が保管されます。インスタンス・ディレクトリーの位置を作成後に変更することはできません。インスタンス・ディレクトリーの内容は、以下のとおりです。

- データベース・マネージャー構成ファイル
- システム・データベース・ディレクトリー
- ノード・ディレクトリー
- DB2 診断ファイル (db2diag.log)
- ノード構成ファイル (db2nodes.cfg)
- デバッグ情報 (例外 / レジスターのダンプや DB2 プロセス用の呼び出しスタックなど) が入った他のファイル。

UNIX オペレーティング・システムでは、インスタンス・ディレクトリーは INSTHOME/sqllib ディレクトリーにあります。ただし、INSTHOME は、インスタンス所有者のホーム・ディレクトリーです。

区分データベース・システムにおけるインスタンス・ディレクトリーは、インスタンスに属するすべてのデータベース区画サーバー間で共用されます。したがって、インスタンス・ディレクトリーは、インスタンス内のすべてのマシンがアクセスできるネットワーク共用ドライブ上に作成しなければなりません。

インストール手順の一部として、『DB2』という DB2 の初期インスタンスを作成します。 UNIX では、命名規則の指針にはずれない範囲で初期インスタンスの名前を付けることができます。インスタンス名は、ディレクトリー構造を設定するために使用します。

このインスタンスをすぐに使えるようにするために、インストール中に次の設定がなされます。

- 環境変数 DB2INSTANCE が『DB2』に設定される。
- DB2 レジストリー変数 DB2INSTDEF が『DB2』に設定される。

UNIX では、命名規則の指針にはずれない範囲でデフォルトの名前を付けることができます。

これらの設定により、『DB2』がデフォルト・インスタンスとして確立されます。デフォルトで使用されるインスタンスを変更することはできますが、最初に、追加インスタンスを作成する必要があります。

DB2 を使用する前に、各ユーザーのデータベース環境を更新して、インスタンスにアクセスし、DB2 プログラムを実行できるようにします。このことはすべてのユーザー (管理ユーザーも含む) に当てはまります。

UNIX オペレーティング・システムでは、データベース環境の設定に役立つサンプル・スクリプト・ファイルが提供されます。サンプル・ファイルは、Bourne または Korn シェルの場合は db2profile、 C シェルの場合は db2cshrc です。これらのスクリプトは、インスタンス所有者のホーム・ディレクトリー下の sqllib サブディレクトリーに配置されます。インスタンス所有者またはインスタンスの SYSADM グループに属するユーザーはだれでも、インスタンスの全ユーザーのスクリプトをカスタマイズできます。また、スクリプトをユーザーごとにコピーして、カスタマイズすることもできます。

サンプル・スクリプトには、次の目的を持つステートメントが入っています。

- 既存の検索パスに以下のディレクトリーを追加して、ユーザーの「パス」を更新します。すなわち、インスタンス所有者のホーム・ディレクトリーの sqllib サブディレクトリー下にある bin、 adm、misc サブディレクトリー。
- DB2INSTANCE 環境変数をインスタンス名に設定します。

DB2 環境の自動設定

注: 以下の説明は、UNIX オペレーティング・システム環境にのみ適用されます。

デフォルトのスクリプトは、現行セッションの期間中に限ってユーザー環境に影響を与えます。 .profile ファイルを変更すれば、ユーザーが Bourne または Korn シェルを使ってログオンするときに、 db2profile スクリプトを自動的に実行できようになります。 C シェルのユーザーであれば、 .login ファイルを変更して db2shrc スクリプト・ファイルを実行できるようになります。

.profile または .login スクリプト・ファイルに、以下に示すいずれかのステートメントを追加します。

- 1 つのバージョンのスクリプトを共用するユーザーの場合は、次のように追加します。

```
. INSTHOME/sqllib/db2profile      (Bourne または Korn シェルの場合)
source INSTHOME/sqllib/db2cshrc   (C シェルの場合)
```

  この場合の INSTHOME は、使用するインスタンスのホーム・ディレクトリーです。
- ホーム・ディレクトリーにカスタマイズしたバージョンのスクリプトがあるユーザーの場合は、次のように追加します。

```
. USERHOME/db2profile          (Bourne または Korn シェルの場合)
source USERHOME/db2cshrc       (C シェルの場合)
```

USERHOME は、ユーザーのホーム・ディレクトリーです。

DB2 環境の手動設定

注: 以下の説明は、UNIX オペレーティング・システム環境にのみ適用されます。

使用したいインスタンスを選択するには、コマンド・プロンプトで次のいずれかのステートメントを入力します。 ピリオド (.) とスペースは必須です。

- 1 つのバージョンのスクリプトを共用するユーザーの場合は、次のように追加します。

  ```
  . INSTHOME/sqllib/db2profile      (Bourne または Korn シェルの場合)
  source INSTHOME/sqllib/db2cshrc   (C シェルの場合)
  ```

  この場合の INSTHOME は、使用するインスタンスのホーム・ディレクトリーです。
- ホーム・ディレクトリーにカスタマイズしたバージョンのスクリプトがあるユーザーの場合は、次のように追加します。

  ```
  . USERHOME/db2profile          (Bourne または Korn シェルの場合)
  source USERHOME/db2cshrc       (C シェルの場合)
  ```

  USERHOME は、ユーザーのホーム・ディレクトリーです。

同時に複数のインスタンスを処理したい場合は、使用する各インスタンスのスクリプトを別々のウィンドウで実行します。たとえば、test および prod という 2 つのインスタンスがあり、そのホーム・ディレクトリーが /u/test および /u/prod であるとします。

ウィンドウ 1 では、次のようにします。

- Bourne または Korn シェルでは、次のように入力します。

  ```
  . /u/test/sqllib/db2profile
  ```
- C シェルでは、次のように入力します。

  ```
  source /u/test/sqllib/db2cshrc
  ```

ウィンドウ 2 では、次のようにします。

- Bourne または Korn シェルでは、次のように入力します。

  ```
  . /u/prod/sqllib/db2profile
  ```
- C シェルでは、次のように入力します。

  ```
  source /u/prod/sqllib/db2cshrc
  ```

ウィンドウ 1 は test インスタンスを処理するため、ウィンドウ 2 は prod インスタンスを処理するために使用します。

注: **which db2** コマンドを入力し、検索パスが正確に設定されていることを確かめます。このコマンドを実行すると、DB2 CLP 実行可能モジュールの絶対パスが戻されます。その位置が、インスタンスの sqllib ディレクトリーにあることを確かめてください。

1 つのシステム上の複数インスタンス

1 つのシステム上で複数のインスタンスを持つことが可能です。しかし、一度に 1 つの DB2 インスタンス内の作業だけが可能です。

インスタンス所有者およびシステム管理 (SYSADM) グループであるグループは、個々のインスタンスと関連付けられます。インスタンス所有者および SYSADM グループは、インスタンスの作成プロセス中に割り当てられます。 1 つのインスタンスに限って 1 つのユーザー ID またはユーザー名を使用することができます。そのユーザー ID またはユーザー名は、*インスタンス所有者* とも呼ばれます。

各インスタンス所有者は固有のホーム・ディレクトリーを持つ必要があります。インスタンスの実行に必要なファイルすべては、インスタンス所有者のユーザー ID またはユーザー名のホーム・ディレクトリーに作成されます。インスタンス所有者のユーザー ID またはユーザー名をシステムから除去する必要が生じた場合、インスタンスに関連付けられたファイルと、そのインスタンスに保管されたデータへのアクセスを失うおそれがあります。このため、インスタンス所有者のユーザー ID またはユーザー名は、 DB2 の実行だけのために使用することをお勧めします。

インスタンス所有者の 1 次グループも重要です。この 1 次グループは自動的にインスタンスのシステム管理グループになり、インスタンスに対する SYSADM 権限を取得します。インスタンス所有者の 1 次グループのメンバーである他のユーザー ID またはユーザー名も、このレベルの権限を取得します。この理由から、インスタンス所有者のユーザー ID またはユーザー名は、インスタンスの管理用に確保した 1 次グループに割り当てたいと思うかもしれません。 (また、1 次グループは必ずインスタンス所有者のユーザー ID またはユーザー名に割り当てるようにします。割り当てなければ、システム・デフォルトの 1 次グループが使用されます。)

インスタンスのシステム管理グループにしたいグループがすでにある場合は、インスタンス所有者ユーザー ID またはユーザー名の作成時に、そのグループを 1 次グループとして割り当てるだけで済みます。インスタンスに対する管理権限を他のユーザーに付与するには、システム管理グループとして割り当てられたグループに該当するユーザーを追加します。

インスタンス間で SYSADM 権限を分離するには、インスタンス所有者ユーザー ID またはユーザー名ごとに異なる 1 次グループを使用します。ただし、複数インスタンスで共通の SYSADM 権限を持つことにした場合は、複数インスタンスに対して同じ 1 次グループを使用することができます。

インスタンスの追加

OS/2 での管理権限を持っている場合、または Windows NT 上の管理グループに属している場合、追加の DB2 インスタンスを追加できます。インスタンスの追加先のマシンは、インスタンス所有マシン (ノード 0) になります。インスタンスは必ず、管理サーバーが常駐するマシン上で追加してください。

別のインスタンスを追加するには、以下のステップを実行します。

1. 管理権限を持っているか、またはローカル管理者グループに属するユーザー ID またはユーザー名でログオンします。
2. インスタンスを追加するには、次の方法のいずれかを使用してください。

   コントロール・センターを使用するには、以下のようにします。

   a. オブジェクト・ツリーを順に展開し、使用するシステムの**「インスタンス (Instances)」**フォルダーを表示します。
   b. インスタンスのフォルダーを右クリックして、ポップアップ・メニューから**「追加 (Add)」**を選択します。
   c. 情報をすべて入力し、**「適用 (Apply)」**をクリックします。

   コマンド行を使用してインスタンスを追加するには、以下のように入力します。

   ```
   db2icrt <instance_name>
   ```

3. 管理サーバーを作成します。

db2icrt コマンドを使用して別の DB2 インスタンスを追加するときには、インスタンス所有者のログイン名を提供する必要があり、任意選択で、インスタンスの認証タイプを指定します。認証タイプは、そのインスタンスの下で作成されたすべてのデータベースに適用されます。認証タイプとは、ユーザーの認証が行われる場所を示すものです。認証についての詳細は、231ページの『第5章 データベース・アクセスの制御』を参照してください。

注: db2iupdt コマンドを使って、インスタンス構成を更新することができます。

インスタンス・ディレクトリーの位置は、 DB2INSTPROF 環境変数を使って DB2PATH から変更することができます。その場合は、インスタンス・ディレクトリーの書き込みアクセスが必要です。 DB2PATH 以外のパスにディレクトリーを作成したければ、 **db2icrt** コマンドを入力する**前に**、 DB2INSTPROF を設定しなければなりません。

DB2 エンタープライズ拡張エディション上でインスタンスを追加する際の詳細: DB2 ユニバーサル・データベース エンタープライズ拡張エディションで作業する場合は、区分データベース・システムである新規インスタンスを追加することを宣言する必要もあります。この宣言は、コマンド行で `-s eee` を使って行います。

UNIX 上でインスタンスを作成する際の詳細: UNIX オペレーティング・システムで作業する場合、 **db2icrt** コマンドには、以下の任意指定パラメーターがあります。

- -h または -?

  このパラメーターは、コマンドのヘルプ・メニューを表示する場合に使用します。

- -d

  このパラメーターは、問題判別中に使用するデバッグ・モードを設定します。

- -a AuthType

  このパラメーターは、インスタンスの認証タイプを指定します。有効な認証タイプは、SERVER、CLIENT、DCS、または DCE です。指定しない場合、DB2 サーバーがインストールされていれば、デフォルトは SERVER に設定されます。それ以外の場合は、CLIENT に設定されます。

  注:

  1. インスタンスの認証タイプは、インスタンスが所有するすべてのデータベースに適用されます。
  2. UNIX オペレーティング・システムでは、認証タイプ DCE の選択は無効です。

- -u FencedID

  このパラメーターは、分離ユーザー定義関数 (UDF) とストアード・プロシージャーを実行するユーザーです。 DB2 クライアントまたは DB2 アプリケーション開発クライアントをインストールする場合、これは必須ではありません。他の DB2 製品の場合、これは必須パラメーターです。

  注: FencedID は、『root』 または 『bin』 ではありません。

- -p PortName

  このパラメーターは、使用予定の TCP/IP サービス名またはポート番号を指定します。その際、この値はインスタンス内のすべてのデータベースについて、そのインスタンスのデータベース構成ファイルで設定されます。

- -s InstType

  異なるタイプのインスタンスを追加できます。有効なインスタンス・タイプは、`ee`、`eee`、および `client` です。

次に例を示します。

- DB2 サーバー用のインスタンスを追加するには、以下のコマンドを使用できます。

  ```
  db2icrt -u db2fenc1 db2inst1
  ```

- DB2 コネクト エンタープライズ・エディションだけをインストールした場合は、インスタンス名を分離 ID としても使用できます。

  ```
  db2icrt -u db2inst1 db2inst1
  ```

- DB2 クライアント用のインスタンスを追加するには、以下のコマンドを使用できます。

```
db2icrt db2inst1 -s client -u fencedID
```

DB2 クライアント・インスタンスは、ワークステーションから他のデータベース・サーバーへ接続したい場合に作成します。作成すると、そのワークステーションにローカル・データベースは必要なくなります。

Windows NT 上でインスタンスを作成する際の詳細: Windows NT オペレーティング・システムで作業する場合、 **db2icrt** コマンドには、以下の任意指定パラメーターがあります。

- -s InstType

  異なるタイプのインスタンスを作成できます。有効なインスタンス・タイプは、ee、eee、および client です。

- /p:InstProf_Path

  これは、別のインスタンス・プロファイル・パスを指定する任意指定パラメーターです。パスを指定しない場合、インスタンス・ディレクトリーは SQLLIB ディレクトリー内に作成され、インスタンス名に連結された共用名 DB2 を指定されます。読み取りおよび書き込み許可は、ドメイン内の各人に自動的に与えられます。許可は、ディレクトリーへのアクセスを制限するために変更することができます。

  異なるインスタンス・プロファイル・パスを実際に指定する場合は、共用ドライブまたはディレクトリーを作成する必要があります。これにより、許可が変更されていなければ、ドメイン内のすべての人がインスタンス・ディレクトリーにアクセスできるようになります。

- /u:username,password

  区分データベース環境を作成する場合、 DB2 サービスのログオンおよびアカウント名とパスワードを宣言しなければなりません。

- /r:base_port,end_port

  これは、高速コミュニケーション・マネージャー (FCM) の TCP/IP ポート範囲を指定する任意指定パラメーターです。 TCP/IP ポート範囲を指定する場合は、区分データベース・システムのすべてのマシンで、そのポート範囲を使用できることを確認しなければなりません。

たとえば、DB2 (Windows NT 版) エンタープライズ拡張エディションでは、以下の例を利用できます。

```
db2icrt inst1 -s eee
   /p:¥¥machineA¥db2mpp
   /u:yourname,yourpwd /r:9010,9015
```

注: db2icrt コマンドにより、インスタンスの作成に使用するユーザー名に以下の権利が与えられます。

- オペレーティング・システムの一部として活動する。
- トークン・オブジェクトを作成する。

- 割り当て量を増やす。
- サービスとしてログオンする。
- 処理レベル・トークンを置換する。

インスタンスは、共用ドライブにアクセスし、ユーザー・アカウントを認証し、DB2 を Windows NT サービスとして実行するために、これらのユーザー権を必要とします。

インスタンスのリスト

コントロール・センターを使用して、システムで使用可能なインスタンスすべてをリストするには、次のようにします。

1. オブジェクト・ツリーを順に展開し、システムの**「インスタンス (Instances)」**フォルダーを表示します。
2. インスタンスのフォルダーを右クリックして、ポップアップ・メニューから**「追加 (Add)」**を選択します。
3. 「データベースの追加 (Add Database)」ウィンドウで、**「最新表示 (Refresh)」**をクリックします。
4. ドロップダウン矢印をクリックして、データベース・インスタンスのリストを表示します。
5. **「取消 (Cancel)」**をクリックして、ウィンドウを終了します。

コマンド行を使用して、システムで使用可能なインスタンスすべてをリストするには、次のようにします。

```
db2ilist
```

現行セッションに適用されるインスタンスを判別するには、次のように入力します。

```
set db2instance
```

注: UNIX オペレーティング・システムでは、次のように入力します。

```
db2 get instance
```

現行インスタンスの設定

インスタンスのデータベース・マネージャーを開始または停止するためのコマンドを実行すると、 DB2 はそのコマンドを現行インスタンスに適用します。 DB2 は以下のように現行インスタンスを判別します。

- 現行セッションに DB2INSTANCE 環境変数が設定されていれば、その値が現行インスタンスです。 DB2INSTANCE 環境を設定するには、次のように入力します。

  ```
  set db2instance=<new_instance_name>
  ```

- DB2INSTANCE 環境変数が現行セッション用に設定されていない場合、 DB2 はシステム環境変数から DB2INSTANCE 環境変数用の設定値を使用します。 Windows NT

では、システム環境変数はシステム環境内で設定されます。 Windows 95 では、autoexec.bat ファイルで設定されます。 OS/2 では、config.sys ファイルで設定されます。

- DB2INSTANCE 環境変数がまったく設定されていなければ、 DB2 はレジストリー変数 DB2INSTDEF を使用します。

  DB2INSTDEF レジストリー変数をレジストリーのグローバル・レベルで設定するには、次のように入力します。

  ```
  db2set db2instdef=<new_instance_name> -g
  ```

自動始動インスタンス

UNIX オペレーティング・システムで、各システムの再始動後にインスタンスを自動開始できるようにするには、次のコマンドを入力します。

```
db2iauto -on InstName
```

InstName はインスタンスのログイン名です。

UNIX オペレーティング・システムで、各システムの再始動後にインスタンスを自動開始できないようにするには、次のコマンドを入力します。

```
db2iauto -off InstName
```

InstName はインスタンスのログイン名です。

複数インスタンスの並列実行

DB2 インスタンスは同じレベルのコードを使用している限り、複数のインスタンスを開始できます。

コントロール・センターを使用して複数インスタンスを並列実行するには、以下のようにします。

1. オブジェクト・ツリーを順に展開し、**「データベース (Databases)」**フォルダーを表示します。
2. インスタンスを右クリックして、ポップアップ・メニューから**「開始 (Start)」**を選択します。
3. ステップ 2 を繰り返し実行し、並行実行したいインスタンスをすべて開始します。

コマンド行を使用して複数インスタンスを並列実行するには、以下のようにします。

1. 次のように入力して、開始する他のインスタンスの名前に DB2INSTANCE 変数を設定します。

   ```
   set db2instance=<another_instName>
   ```

2. **db2start** コマンドを入力してインスタンスを開始します。

ライセンス管理

DB2 製品のライセンス管理は主に、製品のオンライン・インターフェースのコントロール・センター内でライセンス・センターを使って行います。ライセンス・センターでは、インストールした個々の製品に関して、ライセンス情報、統計、登録済みユーザー、および現行ユーザーを調べることができます。

環境変数およびプロファイル・レジストリーの確立

環境変数およびレジストリー変数は、データベース環境を制御します。

DB2 プロファイル・レジストリーを導入する前に、 (たとえば) Windows または OS/2 ワークステーションの環境変数を変更するには、環境変数を変更してリブートする必要がありました。現在、環境は、いくつかの例外はありますが、 DB2 プロファイル・レジストリーに保管されているレジストリー変数によって制御されます。特定インスタンスのシステム管理 (SYSADM) 権限を持つユーザーは、そのインスタンスのレジストリー値を更新することができます。リブートせずにレジストリー変数を更新するには、**db2set** コマンドを使用します。この情報は、即時にプロファイル・レジストリーの中に保管されます。 DB2 レジストリーは、変更を行った後に開始した DB2 サーバー・インスタンスと DB2 アプリケーションに更新情報を適用します。

レジストリーを更新する場合、現在実行中の DB2 アプリケーションまたはユーザーに影響はありません。更新後に開始されたアプリケーションは新しい値を使用します。

注: DB2 環境変数 DB2INSTANCE、DB2NODE、 DB2PATH、および DB2INSTPROF は、オペレーティング・システムにもよりますが、 DB2 プロファイル・レジストリーに保管されない場合があります。これらの環境変数を更新するためには、 **set** コマンドを使用しなければなりません。変更は、次回システムをリブートするまで有効です。 UNIX プラットフォームでは、 **set** コマンドの代わりに **export** コマンドを使用できます。

プロファイル・レジストリーを使用することによって、環境変数の中央制御が可能になります。 *管理の手引き: パフォーマンス* の『DB2 レジストリーおよび環境変数』には、環境変数とレジストリー変数の多くがリストされています。さまざまなレベルのサポートが、さまざまな環境プロファイルを通して提供されるようになっています。環境変数のリモートでの管理も、DB2 管理サーバーを使用すれば可能です。

以下の 4 つのプロファイル・レジストリーがあります。

- DB2 インスタンス・レベル・プロファイル・レジストリー。 DB2 環境変数の大多数は、このレジストリーの中に置かれます。特定のインスタンスについての環境変数の設定値が、このレジストリーに保持されます。このレベルに定義された値は、グローバル・レベルの設定値をオーバーライドします。
- DB2 グローバル・レベル・プロファイル・レジストリー。特定のインスタンスごとに 1 つの環境変数が設定されるのではない場合、このレジストリーが使用されます。こ

のレジストリーは、マシン全体にわたる環境変数設定値を持ちます。 DB2 UDB EEE では、マシンごとに 1 つのグローバル・レベル・プロファイルが存在します。
- DB2 インスタンス・ノード・レベル・プロファイル・レジストリー。異なる複数のデータベース区画にデータベースが分割されているシステムでは、このレジストリーが各ノード (つまり、マシン) ごとに常駐し、そのノードにデータを保管するすべてのインスタンスについての環境変数設定値が含まれます。このレベルで定義された値は、インスタンスおよびグローバル・レベルの同様の設定値をオーバーライドします。
- DB2 インスタンス・プロファイル・レジストリー。このレジストリーには、このシステムによって認識されるすべてのインスタンス名のリストが含まれます。

ユーザーは、**set** コマンド (UNIX プラットフォームでは export コマンド) を使用して、セッション環境変数の設定値を変更することによって、ユーザーのセッションに対する DB2 インスタンス・プロファイル・レジストリー環境変数の設定値をオーバーライドすることができます。

DB2 は、レジストリー値と環境変数を検査し、それらを以下の順序で解決することによって、操作環境を構成します。
1. `set` コマンドを使用して設定された環境変数。 (あるいは、 UNIX プラットフォームでは `export` コマンド。)
2. インスタンス・ノード・レベル・プロファイルを使用して設定されたレジストリー値 (以下に示すノード番号を使用した `db2set -i` コマンドを使用)。
3. インスタンス・プロファイルを使用して設定されたレジストリー値 (以下に示す `db2set -i` コマンドを使用)。
4. グローバル・プロファイルを使用して設定されたレジストリー値 (以下に示す `db2set -g` コマンドを使用)。

db2set コマンドの使用

db2set コマンドは、レジストリー変数 (および環境変数) のローカルでの宣言をサポートします。

コマンドに対するヘルプ情報を表示するには、以下を使用します。

```
db2set ?
```

サポートされるすべてのレジストリー変数の完全なセットをリストするには、以下を使用します。

```
db2set -lr
```

現在のインスタンスまたはデフォルトのインスタンスに定義されているすべてのレジストリー変数をリストするには、以下を使用します。

```
db2set
```

プロファイル・レジストリーに定義されているすべてのレジストリー変数をリストするには、以下を使用します。

```
db2set -all
```

現在のインスタンスまたはデフォルトのインスタンス内のレジストリー変数の値を表示するには、以下を使用します。

```
db2set registry_variable_name
```

全レベルのレジストリー変数の値を表示するには、以下を使用します。

```
db2set registry_variable_name -all
```

指定レベルの変数値を削除するには、同じコマンド構文を使って変数を設定できますが、変数値には何も指定しません。たとえば、ノード・レベルの変数設定値を削除するには、次のように入力します。

```
db2set registry_variable_name= -i instance_name
node_number
```

変数値を削除してその使用を制限する場合、その値が上位レベルで定義されていれば、次のように入力します。

```
db2set registry_variable_name= -null instance_name
```

このコマンドを使用すると、指定するパラメーターの設定値が削除され、上位レベルのプロファイルでこの変数の値 (この場合は DB2 グローバル・レベル・プロファイル) を変更できなくなります。ただし、指定する変数を下位レベルのプロファイル (この場合は DB2 ノード・レベル・プロファイル) で引き続き設定することはできます。

現在のインスタンスまたはデフォルトのインスタンス内のレジストリー変数の値を変更するには、以下を使用します。

```
db2set registry_variable_name=new_value
```

インスタンス内のすべてのデータベースについて、レジストリー変数のデフォルトを変更するには、以下を使用します。

```
db2set registry_variable_name=new_value
   -i instance_name
```

このシステム内のすべてのインスタンス内のレジストリー変数のデフォルトを変更するには、以下を使用します。

```
db2set registry_variable_name=new_value -g
```

レジストリー値をユーザー・レベルで設定するには、以下を使用します。

```
db2set -ul
```

レジストリー値を特定ユーザーのユーザー・レベルで設定するには、以下を使用します。

```
db2set -ul user_name
```

注:

1. パラメーター "-i"、"-g"、"-ul" は、同じコマンド内では同時に使用できません。
2. デフォルトとして常にグローバル・レベル・プロファイルに設定されるパラメーターもあります。そのようなパラメーターをインスタンスまたはノード・レベル・プロファイルで設定することはできません。たとえば、db2system および db2instdef があります。
3. UNIX では、インスタンスのレジストリー値を変更するためには、システム管理 (SYSADM) 権限を持っていなければなりません。グローバル・レベル・レジストリー中のパラメーターを変更できるのは、 root 権限を持つユーザーだけです。

LDAP 環境で実行している場合、ある DB2 レジストリー変数値については、LDAP 内で設定することができます (つまり、その効力範囲が、ディレクトリー区画または Windows NT ドメインに属するすべてのマシンとすべてのユーザーに渡る、グローバルなものとなります)。現時点で、LDAP グローバル・レベルで設定できる DB2 レジストリー変数は、 DB2LDAP_SEARCH_SCOPE だけです。

この変数を LDAP グローバル・レベルで設定するには、 db2set コマンドの -gl オプションを使用してください。

注: このオプションは -g オプションとは違います。 -g オプションは、マシン・グローバル・レベルで DB2 レジストリー変数を設定するのに使用するオプションです。それに対し、-gl は LDAP グローバル・レベルに固有のオプションです。また、LDAP 内での DB2 レジストリー変数の設定をサポートするのは、 Windows プラットフォームだけです。

LDAP でグローバル・レベルの検索有効範囲値を設定するには、以下を使用します。

```
db2set -gl db2ldap_search_scope = value
```

value には 『local』、『domain』、または 『global』 を指定できます。

このインスタンス内の特定のノードのレジストリー変数のデフォルトを変更するには、以下を使用します。

```
db2set registry_variable_name=new_value
   -i instance_name node_number
```

1 つのインスタンスの 1 つのレジストリー変数をリセットして、グローバル・プロファイル・レジストリーの中のデフォルトに戻すには、以下を使用します。

```
db2set -r registry_variable_name
```

1 つのインスタンス内のノードの 1 つのレジストリー変数をリセットして、グローバル・プロファイル・レジストリーの中のデフォルトに戻すには、以下を使用します。

```
db2set -r registry_variable_name node_number
```

OS/2 上の環境変数の設定

DB2 固有のレジストリー値はすべて、 DB2 プロファイル・レジストリーで定義することをぜひお勧めします。 DB2 変数がレジストリー以外で設定されていれば、その変数をリモート管理することはできません。変数値を有効にするには、ワークステーションをリブートしなければなりません。

OS/2 上では、DB2PATH および DB2INSTPROF とは別に、 config.sys 内に環境変数を定義してはなりません。すべての変数は、**db2set** コマンドを使用してプロファイル・レジストリーの中に定義する必要があります (実際の環境変数として残っている値を除きます)。

DB2INSTANCE も実際の環境変数として残りますが、 DB2INSTDEF レジストリー変数を使用する場合は必要ありません。このレジストリー変数は、
DB2INSTANCE が設定されない場合に使用されるデフォルトのインスタンス名を定義します。

DB2INSTANCE および DB2PATH は DB2 をインストールするときに設定されます。DB2INSTPROF はインストール後に設定できます。環境変数 DB2PATH は設定する必要があります。この環境変数はインストール中に設定され、変更すべきではありません。 DB2INSTANCE および DB2INSTPROF 環境変数の設定は任意選択です。

環境変数の設定値を判別するには、次のように入力します。

```
set variable
```

環境変数の設定値を変更するには、次のコマンドを入力します。

```
set variable=value
```

システム環境変数を設定するためには、config.sys ファイルを編集し、変更を有効にするためにシステムをリブートします。

異なるプロファイル・レジストリーが、以下にしたがって配置されます。

- DB2 インスタンス・レベル・プロファイル・レジストリー・ファイルは、以下のもとに配置されます。

  ```
  %DB2INSTPROF%¥instance_name¥PROFILE.ENV
  ```

- DB2 グローバル・レベル・プロファイルは、以下のもとに配置されます。

  ```
  %DB2INSTPROF%¥DEFAULT.ENV
  ```

- DB2 インスタンス・プロファイル・レジストリーは、以下のもとに配置されます。

  ```
  %DB2INSTPROF%¥PROFILES.REG
  ```

Windows NT および Windows 95 上での環境変数の設定

DB2 固有のレジストリー値はすべて、 DB2 プロファイル・レジストリーで定義することをぜひお勧めします。 DB2 変数がレジストリー以外で設定されていれば、その変数をリモート管理することはできません。変数値を有効にするには、ワークステーションをリブートしなければなりません。

Windows 32 ビット・オペレーティング・システムには 1 つのシステム環境変数 DB2INSTANCE があり、この値はプロファイル・レジストリーの外部でしか設定できません。ただし、DB2INSTANCE の設定は必須ではありません。 DB2 プロファイル・レジストリー変数 DB2INSTDEF をグローバル・レベル・プロファイルで設定すれば、DB2INSTANCE が定義されていない場合に使用するインスタンス名を指定することができます。

Windows NT 上の DB2 エンタープライズ拡張エディションには、 2 つのシステム環境変数 DB2INSTANCE および DB2NODE があり、この値はプロファイル・レジストリーの外部でしか設定できません。 DB2INSTANCE の設定は必須ではありません。 DB2 プロファイル・レジストリー変数 DB2INSTDEF をグローバル・レベル・プロファイルで設定すれば、 DB2INSTANCE が定義されていない場合に使用するインスタンス名を指定することができます。

DB2NODE 環境変数は、マシン内のターゲット論理ノードへの要求を経路指定するために使用します。この環境変数は、DB2 プロファイル・レジストリーの中ではなく、アプリケーションまたはコマンドが発行されるセッションで設定しなければなりません。この変数を指定しない場合、ターゲット論理ノードはデフォルトとして、マシン上のポートをゼロ (0) に定義された論理ノードに設定されます。

環境変数の設定値を判別するためには、**echo** コマンドを使用します。たとえば、DB2PATH 環境変数の値を検査するには、以下のように入力します。

```
echo %db2path%
```

システム環境変数を設定するには、以下のことを行います。

Windows 95 および Windows 98 の場合: *autoexec.bat* ファイルを編集し、変更を有効にするためにシステムをリブートします。

Windows NT 4.x の場合: 以下のように、 DB2 環境変数 DB2INSTANCE、DB2PATH、DB2INSTPROF を設定することができます。

- **「スタート」**、**「設定」**、**「コントロール パネル」**を選択します。
- **「システム」**アイコンをダブルクリックします。
- システム・コントロール・パネルのシステム環境変数セクションで、以下のことを行います。
  1. DB2INSTANCE 変数が存在しない場合は、以下のようにします。
     a. いずれかのシステム環境変数を選択します。

b. 「*変数*」フィールドの中の名前を DB2INSTANCE に変更します。

c. 「*値*」フィールドをインスタンス名 (たとえば、db2inst) に変更します。

2. DB2INSTANCE 変数がすでに存在している場合は、以下のようにして新しい値を追加します。

   a. DB2INSTANCE 環境変数を選択します。

   b. 「*値*」フィールドをインスタンス名 (たとえば、db2inst) に変更します。

3. 「設定」を選択します。
4. 「OK」を選択します。
5. これらの変更が有効になるよう、システムをリブートします。

注: 環境変数 DB2INSTANCE もセッション (プロセス) レベルで設定できます。たとえば、TEST という 2 番目の DB2 インスタンスを開始したければ、コマンド・ウィンドウで以下のコマンドを発行します。

```
set db2instance=TEST
db2start
```

プロファイル・レジストリーは、以下のように配置されます。

- Windows NT オペレーティング・システム・レジストリーの中の DB2 インスタンス・レベル・プロファイル・レジストリーは、以下のパスを使用して配置されます。

  ¥HKEY_LOCAL_MACHINE¥SOFTWARE¥IBM¥DB2¥PROFILES¥*instance_name*

  注: *instance_name* は、作業を行っているデータベース区画に固有のものです。

- Windows NT レジストリーの中の DB2 グローバル・レベル・プロファイル・レジストリーは、以下のパスを使用して配置されます。

  ¥HKEY_LOCAL_MACHINE¥SOFTWARE¥IBM¥DB2¥GLOBAL_PROFILE

- Windows NT レジストリーの中の DB2 インスタンス・ノード・レベル・プロファイル・レジストリーは、以下のパスを使用して配置されます。

  ...¥SOFTWARE¥IBM¥DB2¥PROFILES¥*instance_name*¥NODES¥*node_number*

  注: *instance_name* と *node_number* は、作業しているデータベース区画に固有のものです。

DB2 UDB には、リモート・マシン上のインスタンス・レベルの DB2 UDB レジストリー変数にアクセスする機能があります。現在、DB2 UDB レジストリー変数は、マシンまたはグローバル・レベル、インスタンス・レベル、およびノード・レベルという 3 つの異なるレベルに保管されています。インスタンス・レベル (ノード・レベルも含む) に保管されているレジストリー変数は、 DB2REMOTEPREG を使用して別のマシンにリダイレクトすることができます。 DB2REMOTEPREG が設定されると、DB2 UDB は、DB2REMOTEPREG が指すマシンから DB2 UDB レジストリー変数にアクセスします。たとえば、次のようにします。

```
db2set DB2REMOTEPREG=rmtwkstn
```

ただし、*rmtwkstn* はリモート・ワークステーション名です。

注: すべての DB2 インスタンス・プロファイルとインスタンス・リストが、指定されたリモート・マシン名上に置かれるので、このオプションの設定には注意が必要です。

この機能は、レジストリーが含まれる同じマシン上のリモート LAN 装置を指すために、 DBINSTPROF の設定と組み合わせて使用することができます。

UNIX システム上の環境変数の設定

DB2 固有のレジストリー値はすべて、 DB2 プロファイル・レジストリーで定義することをぜひお勧めします。 DB2 変数がレジストリー以外で設定されていれば、その変数をリモート管理することはできません。

UNIX オペレーティング・システムでは、システム環境変数 DB2INSTANCE を設定しなければなりません。

db2profile (Korn シェルの場合) および **db2cshrc** (Bourne シェルまたは C シェルの場合) というスクリプトが、データベース環境のセットアップを援助するために例として提供されています。これらのファイルは `insthome/sqllib` の中にあります (ただし、`insthome` はインスタンス所有者のホーム・ディレクトリーです)。

これらのスクリプトには、以下のためのステートメントが入っています。

- 以下のディレクトリーにユーザーのパスを更新します。
  - `insthome/sqllib/bin`
  - `insthome/sqllib/adm`
  - `insthome/sqllib/misc`
- 実行のために、 `DB2INSTANCE` をデフォルトのローカル `instance_name` に設定します。

注: PATH および DB2INSTANCE を除いて、 DB2 でサポートされる変数は DB2 プロファイル・レジストリーで設定しなければなりません。 DB2 で設定されない変数を設定するには、スクリプト・ファイル **db2profile** および **db2cshrc** で定義する必要があります。

インスタンス所有者または SYSADM ユーザーは、インスタンスのすべてのユーザーのためにこれらのスクリプトをカスタマイズすることができます。あるいは、ユーザーがスクリプトをコピーしてカスタマイズした後、スクリプトを直接呼び出すか、または自分の .profile または .login ファイルに追加することができます。

現行セッションについての環境変数を変更するには、以下のようなコマンドを出します。

- Korn シェルの場合、

```
db2instance=inst1
export db2instance
```

- Bourne シェルまたは C シェルの場合、

```
set db2instance inst1
```

DB2 プロファイル・レジストリーが正しく管理されるようにするためには、 UNIX オペレーティング・システム上で、以下のファイル所有権規則に従わなければなりません。 (DB2 管理サーバー (DAS) については、84ページの『DB2 管理サーバー (DAS) の作成』を参照してください。)

- DB2 インスタンス・レベル・プロファイル・レジストリー・ファイルは、以下のもとに配置されます。

```
INSTHOME/sqllib/profile.env
```

このファイルのアクセス許可と所有権は、以下のようである必要があります。

```
-rw-r--r-- Instance_Owner DAS_Instance_Group profile.env
```

INSTHOME は、インスタンス所有者のホーム・パスです。

- DB2 グローバル・レベル・プロファイルは、以下のもとに配置されます。
  - AIX、Solaris、SINIX、および NUMA-Q(Sequent) オペレーティング・システムの場合は、 */var/db2/<version_id>/default.env* (<version_id> は現行バージョン)。
  - HP-UX オペレーティング・システムの場合は、 */var/opt/db2/<version_id>/default.env* (<version_id> は現行バージョン)。

このファイルのアクセス許可と所有権は、以下のようである必要があります。

```
-rw-r--r-- DAS_Instance_Owner DAS_Instance_Group default.env
```

グローバル・レジストリー変数を修正する場合、ユーザーはルートまたは DAS インスタンス所有者としてログオンしなければなりません。 DB2 管理サーバーの詳細は、84ページの『DB2 管理サーバー (DAS) の作成』を参照してください。

- DB2 インスタンス・ノード・レベル・プロファイル・レジストリーは、以下のもとに配置されます。

```
INSTHOME/sqllib/nodes/node_number.env
```

ディレクトリーとこのファイルのアクセス許可と所有権は、以下のようである必要があります。

```
drwxrwxr-x Instance_Owner DAS_Instance_Group nodes

-rw-r--r-- Instance_Owner DAS_Instance_Group node_number.env
```

注: *Instance_Owner* と *DAS_Instance_Owner* は、両方とも *DAS_Instance_Group* のメンバーである必要があります。

INSTHOME は、インスタンス所有者のホーム・パスです。

- DB2 インスタンス・プロファイル・レジストリーは、以下のもとに配置されます。
  - AIX、Solaris、SINIX、および NUMA-Q(Sequent) オペレーティング・システムの場合は、 `/var/db2/<version_id>/profiles.reg` (<version_id> は現行バージョン)。
  - HP-UX オペレーティング・システムの場合は、`/var/opt/db2/<version_id>/profiles.reg` (<version_id> は現行バージョン)。

このファイルのアクセス許可と所有権は、以下のようである必要があります。

```
-rw-r--r-- root system profiles.reg
```

DB2 管理サーバー (DAS) の作成

DB2 管理サーバー (DAS) は、他の DB2 サーバー上での管理作業を支援するためだけに使用される DB2 管理制御点です。クライアント構成アシスタントまたはコントロール・センターを使用したい場合は、 DAS を実行しておく必要があります。 DAS は、以下の管理タスクを処理するときにコントロール・センターおよびクライアント構成アシスタントを援助します。

- DB2 サーバーをリモートに管理できる。
- DB2 とオペレーティング・システム・コマンド・スクリプトの両方の実行をスケジュールする機能も含めた、ジョブ管理用の機能を提供する。これらのコマンド・スクリプトはユーザーが定義します。コントロール・センターは、ジョブのスケジュールを定義するため、完了したジョブの結果を表示するため、また DAS にとってリモートまたはローカルに置かれているジョブに対して他の管理タスクを実行するために使用されます。
- DB2 ディスカバリー・ユーティリティーと共に、 DB2 インスタンス、データベースおよび他の DB2 管理サーバーの構成についての情報を検出するための手段を提供する。この情報は、DB2 データベースへのクライアント接続の構成を単純化して自動化するために、クライアント構成アシスタントとコントロール・センターが使用します。

1 つのマシン上には 1 つの DAS しか持つことができません。オペレーティング・システムのブート時に、 DAS を開始するために DAS の構成がインストール中に行われます。

DAS は、ホスト・システムでリモート・タスクを実行するために、コントロール・センターまたはクライアント構成アシスタントからのクライアント要求の代わりに使用されます。 DAS にアクセスするには、SYSADM 権限のあるクライアントが必要です。これらのクライアントのすべてが、 SYSADM_GROUP 構成パラメーターを構成するものとなっていても構いません。

これらの要求されたタスクを実行するのに、特定の権限が必要となる場合があります。DAS は、特定のユーザーの識別子の下で実行されます。ユーザーに授与される特権は、

管理者が実行するタスクやオペレーションに関連したコマンドだけに限られていなければなりません。一般的に、必要なタスクまたはオペレーションには次のものがあります。

- オペレーティング・システム (OS) 構成情報を照会する。
- OS を照会してユーザーおよびグループ情報を得る。
- 他の DB2 インスタンスを開始または停止するために、DB2 インスタンスに作用する。
- スケジュールされたジョブを実行する。
- コネクティビティーおよびプロトコル構成を検索する。

DAS の通信の設定についての詳細は、使用しているプラットフォーム用の*概説およびインストール* を参照してください。

DAS の作成

一般にセットアップ・プログラムは、 DB2 インストール中にインスタンス所有マシン上に DAS を作成します。しかし、セットアップ・プログラムが DAS の作成に失敗した場合は、 DAS を手動で作成することができます。

インストール処理の間に DAS に関連して生じることの概要については、以下のことを考慮してください。

- OS/2 または Windows NT プラットフォームでは、以下のようにします。

  ローカル管理者権限を持つアカウントを使って、DAS を作成したいマシンにログオンします。特定のユーザーを識別する場合は、ローカル管理者権限を持つユーザーを作成します。 `db2admin create` を入力します。特定のユーザー・アカウントを使用する場合は、`db2admin create` を出すときに、 『/USER:』と『/PASSWORD:』を使用する必要があります。

  DAS を作成するときに、ユーザー・アカウント名とユーザー・パスワードを指定することができます (指定は任意です)。有効であれば、ユーザー・アカウント名とパスワードは、DAS の所有者を識別します。 DAS 用に作成したユーザー ID またはアカウント名は、ユーザー・アカウントとして使用しないでください。アカウント名のパスワードを『Password Never Expires』 (無期限のパスワード) に設定します。
  DAS を作成した後で、**db2admin setid** コマンドを使用してユーザー・アカウント名とユーザー・パスワードを提供することによって、その所有権を確立または修正することができます。このコマンドの詳細については、*コマンド解説書* を参照してください。

  DB2 UDB (Windows NT 版) エンタープライズ拡張エディションでは、クライアント構成アシスタントまたはコントロール・センターを使って DB2 サーバーへの接続構成を自動化する場合、 DAS として同一マシン上にあるデータベース区画サーバーは調整プログラム・ノードになります。つまり、クライアントからデータベースへの物理接続はすべて、他のデータベース区画サーバーに経路指定される前に、インスタンス所有マシン上のデータベース区画サーバーに向けられます。

DB2 UDB (Windows NT 版) エンタープライズ拡張エディションでは、他のマシン上の追加管理サーバーを追加すると、クライアント構成アシスタントまたはコントロール・センターで DB2 ディスカバリーを使って他のシステムを調整プログラム・ノードとして構成することができます。そのためには、以下のことを実行します。

1. ローカル管理者権限を持つアカウントを使ってマシンにログオンします。
2. DAS が使用するローカル管理者権限を持つ Windows NT アカウントを作成します。アカウントのユーザー名が DB2 命名規則に従ったものであることを確かめてください。 DAS 用のアカウントを作成する場合は、次の点に注意します。
   - DAS 用のアカウントは、ユーザー・アカウントとして使用しないでください。
   - アカウントのパスワードを Password Never Expires (無期限のパスワード) に設定します。
3. 次のコマンドを実行します。

   ```
   db2admin create /user:username
      /password:passwrd
   ```

   ここで、username と passwrd は DAS 用のユーザー名とパスワードです。

- UNIX プラットフォームでは、以下のようにします。
  1. root 権限を持っていることを確認します。
  2. コマンド・プロンプトで、 DB2 ユニバーサル・データベース・インスタンスのパスの instance サブディレクトリーから、以下のコマンドを出します。

     ```
     dasicrt ASName
     ```

     - AIX では、次のようにします。

       ```
       /usr/lpp/db2_nn_00&/instance/
          dasicrt ASname
       ```

     - HP-UX、NUMA-Q(Sequent)、または Solaris では、次のようにします。

       ```
       /opt/IBMdb2/<version_id>/instance/
          dasicrt ASname
       ```

     - Linux では、次のようにします。

       ```
       /usr/IBMdb2/<version_id>/instance/
          dasicrt ASname
       ```

     ただし、*ASName* は管理サーバーのインスタンス名、 db2_nn_00& または <version_id> は現行バージョンの識別子です。

注: NIS および NIS+ を実行する場合は、次のような仕方でユーザー名とグループ名をセットアップします。

- DAS の 1 次グループは、全インスタンスの 2 次グループに入っていなければなりません。
- DAS の 2 次グループは、全インスタンスの 1 次グループに入っていなければなりません。

2 次グループのリストは、システム上で NIS と NIS+ が実行されている場合に限って、自動的に変更されます。

ユーザー ID が所有できるインスタンスは 1 つだけなので、作成する個々の DB2 管理サーバー (DAS) を所有するには、別々のユーザー ID を持たなければなりません。

管理サーバーを作成した後、それを使用してディレクトリー構造およびアクセス許可を確立する必要があります。

DAS の開始と停止

DAS を手動で開始または停止するには、最初に、ローカル管理者権限を持つアカウントまたはユーザー ID を使ってマシンにログオンしなければなりません。

DB2 (OS/2 版) または DB2 (Windows NT 版) で作業する場合は、以下のことを行う必要があります。

- DAS を開始するには、`db2admin start` を入力します。
- DAS を停止するには、`db2admin stop` を入力します。

注: Windows NT では、上記の両方の場合に、これらのコマンドを使用する人は SYSADM、SYSCTRL、または SYSMAINT 権限を持っている必要があります。

DB2 (任意の UNIX オペレーティング・システム用) で作業する場合は、以下のことを行う必要があります。

- DAS を開始するには、以下のようにします。
  1. DAS 所有者としてログインします。
  2. 次のいずれか 1 つを使用してスクリプトのセットアップを実行します。

     ```
     . INSTHOME/sqllib/db2profile      (Bourne または Korn シェルの場合)
     source INSTHOME/sqllib/db2cshrc   (C シェルの場合)
     ```

     ただし、INSTHOME はインスタンスのホーム・ディレクトリーです。
  3. 次のように **db2admin** コマンドを使って DAS を開始します。

     ```
     db2admin start
     ```

  注: DAS は各システムのリブート後に自動的に開始されます。
- DAS を停止するには、以下のようにします。
  1. DAS 所有者としてログインします。
  2. 次のいずれか 1 つを使用してスクリプトのセットアップを実行します。

     ```
     . INSTHOME/sqllib/db2profile      (Bourne または Korn シェルの場合)
     source INSTHOME/sqllib/db2cshrc   (C シェルの場合)
     ```

     ただし、INSTHOME はインスタンスのホーム・ディレクトリーです。

3. 次のように **db2admin** コマンドを使って DAS を停止します。

```
db2admin stop
```

注: UNIX のもとでは、上記の両方の場合に、これらのコマンドを使用する人は、 DAS 所有者の許可 ID を使用してログオンされていなければなりません。

DAS のリスト

マシン上の DAS インスタンスの名前を取得するには、次のように入力します。

```
db2admin
```

このコマンドは、DAS の開始と停止、新しいユーザーとパスワードの作成、DAS インスタンスの除去、 DAS インスタンスに関連したユーザー・アカウントの確立や修正にも使用します。

DAS の構成

DAS に関係するこれらの管理構成パラメーターの現行値を調べるには、以下を入力します。

```
db2 get admin cfg
```

このコマンドを実行すると、製品のインストール中にデフォルトとして与えられた現行値、または構成パラメーターを以前構成したときに与えられた現行値が表示されます。

DAS に関係するデータベース・マネージャー構成ファイルの中の個々の項目を更新するには、以下を入力します。

```
db2 update admin cfg using ...
```

どのデータベース・マネージャー構成パラメーターを変更できるかについての詳細は、*コマンド解説書* を参照してください。

構成パラメーターを推奨されるデータベース・マネージャーのデフォルトにリセットするには、以下を入力します。

```
db2 reset admin cfg
```

データベース・マネージャー構成ファイルに対する変更は、それらがメモリーの中にロードされた後 (つまり、 `db2admin stop` の後に `db2admin start` が続くとき) にのみ有効になります。

DAS の通信プロトコルをセットアップするには、使用しているプラットフォーム用の*概説およびインストール* を参照してください。

DAS についての機密保護の考慮事項

最初に、ローカル管理者権限を持つアカウントまたはユーザー ID を使ってマシンにログオンしなければなりません。

注: Windows NT では、ログオン・アカウント用に設定されない必須アクセス権があるので、**「コントロール パネル」**の**「サービス」**ユーティリティーを使って DAS 用のログオン・アカウントを変更しないでください。 DB2 管理サーバー (DAS) 用のログオン・アカウントを設定または変更する場合は必ず、 **db2admin** コマンドを使用します。

DAS を作成したら、次のように **db2admin** コマンドを使って、ログオン・アカウントを設定または変更できます。

```
db2admin setid username password
```

ここで、*username* と *password* は、ローカル管理者権限を持つアカウントのユーザー名とパスワードです。

環境内のサーバーのそれぞれで、ユーザー ID またはユーザー名が SYSADM 権限を持つことが推奨されます。こうすれば、必要に応じて、他のインスタンスを開始または停止することができます。

DAS の更新

UNIX オペレーティング・システムでは、 DB2 をプログラム一時修正 (PTF) またはコード・パッチによって更新する場合、すべての DB2 管理サーバー (DAS) とすべての既存インスタンスを更新する必要があります。 DAS を更新するには、インストールした DB2 バージョンとリリースに固有のサブディレクトリー下の instance サブディレクトリーにある **dasiupdt** コマンドを使用します。

最初に、『root』として (UNIX の場合)、またはローカル管理者権限を持つアカウントかユーザー ID を使って、マシンにログオンしなければなりません。

コマンドは、次のように使用します。

```
dasiupdt InstName
```

InstName はインスタンス所有者のログイン名です。このコマンドには次のような任意指定パラメーターがあり、 InstName の前に配置してスペースで区切ります。

- -h または -?

  このコマンドのヘルプ・メニューを表示します。

- -d

  デバッグ・モードを設定し、問題分析に使用します。

DAS の除去

最初に、『root』として (UNIX の場合)、またはローカル管理者権限を持つアカウントかユーザー ID を使って、マシンにログオンしなければなりません。

DAS を除去するには、以下のようにします。

- OS/2 または Windows NT オペレーティング・システムでは、以下のようにします。

1. db2admin stop を使用して DAS を停止します。
2. sqllib サブディレクトリー下の db2das00 サブディレクトリーにあるすべてのファイル (必要に応じて) をバックアップします。インスタンス・ディレクトリーは *DB2INSTPROF* レジストリー変数で指示されます。

   注: この例では、除去する DAS の名前を db2das00 とします。
3. db2admin drop を使用して DAS を除去します。

注: Windows NT では、このコマンドを使用する人は SYSADM、SYSCTRL、または SYSMAINT 権限を持っている必要があります。

- UNIX オペレーティング・システムでは、次のようにします。
  1. DAS 所有者としてログインします。
  2. 次のいずれか 1 つを使用してスクリプトのセットアップを実行します。

     ```
     . INSTHOME/sqllib/db2profile      (Bourne または Korn シェルの場合)
     source INSTHOME/sqllib/db2cshrc   (C シェルの場合)
     ```

     ただし、INSTHOME はインスタンスのホーム・ディレクトリーです。
  3. 次のように **db2admin** コマンドを使って DAS を停止します。

     ```
     db2admin stop
     ```
  4. DAS のホーム・ディレクトリー下の sqllib サブディレクトリーにあるすべてのファイル (必要に応じて) をバックアップします。インスタンス・ディレクトリーは *DB2INSTPROF* レジストリー変数で指示されます。
  5. ログオフします。
  6. ルートとしてログインし、次のように **dasidrop** コマンドを使って DAS を除去します。

     ```
     dasidrop ASName
     ```

     ただし、ASName は、管理サーバーのインスタンス名です。このコマンドは、インストールした DB2 バージョンおよびリリースに固有のサブディレクトリー下の instance サブディレクトリーにあります。

     注: dasidrop コマンドを実行すると、 DB2 管理サーバー (DAS) のホーム・ディレクトリーにある sqllib ディレクトリーが除去されます。

EEE システムを使用した DAS のセットアップ

以下では、コントロール・センターを使用して、 DB2 EEE サーバー (Solaris、NT、Sequent、HP-UX、および AIX) をリモート管理用に構成するのに必要なステップについて説明します。

インストール中、セットアップ・プログラムはインスタンス所有マシン上に単一の DAS を作成します。他のマシン上に追加の DAS を作成して、コントロール・センターまたはクライアント構成アシスタントから、他の調整プログラム・ノードにアクセスするこ

ともできます。そうすれば、調整プログラム・ノードとして作業する場合のオーバーヘッドを、インスタンス内の複数ノードに分散できます。

調整プログラム機能を分散するには、次のようにします。

1. 区分データベース・システム内で選択した追加マシンに新しい DAS を作成します。
2. コントロール・センターまたはクライアント構成アシスタント内で個々の DAS を別個のシステムとしてカタログ化します。
3. 個々の新しいシステム下の同じインスタンスをカタログ化し、その都度、DAS をカタログ化するために使用した同じマシン名を指定します。

構成には 2 つの局面があります。つまり、DB2 管理サーバー (DAS) で必要なことと、ターゲットである管理される DB2 インスタンスで推奨されていることです。以下の 3 つのセクションのうち、 2 つの構成トピックの説明にそれぞれ 1 つのセクションが割り振られています。 2 つの構成トピックの説明の前に、前提となる環境について説明しているセクションがあります。

環境の例:

製品 / バージョン:
DB2 UDB EEE V7.1

インストール・パス:
install_path

TCP サービス・ファイル:
tcp_services_file

DB2 インスタンス:

名前: db2inst

所有者 ID:
db2inst

インスタンス・パス:
instance_path

ノード: 3 つのノード、db2nodes.cfg:

- 0 hostA 0 hostA0switch
- 1 hostA 1 hostA1switch
- 2 hostB 0 hostBswitch

DB 名: db2instDB

DAS:

名前: db2as

所有者 / ユーザー ID:
db2as

インスタンス・パス:
das_path

インストール / 実行ホスト:
hostA

ノード間通信ポート:
16000 (hostA および hostB 用の未使用ポート)

注: 上記のフィールドをサイト特有の値で置き換えてください。たとえば、次の表にはサポートされているそれぞれの EEE プラットフォーム用のパス名の例が含まれています。

表 1. サポートされている EEE プラットフォーム用のパス名の例

パス	DB2 UDB EEE (AIX 版)	DB2 UDB EEE (Solaris 版)	DB2 UDB EEE (Windows NT 版)
install_path	/usr/lpp/<v_r_ID>	/opt/IBMdb2/<v_r_ID>	C:¥sqllib
instance_path	/home/db2inst/sqllib	/home/db2inst/sqllib	C:¥profiles¥db2inst
das_path	/home/db2as/sqllib	/home/db2as/sqllib	C:¥profiles¥db2as
tcp_services_file	/etc/services	/etc/services	C:¥winnt¥system32¥drivers¥etc¥services

この表で、 <v_r_ID> はプラットフォーム固有のバージョンとリリースの識別子です。たとえば、DB2 UDB (AIX 版) EEE バージョン 5.2 の場合は、 <v_r_ID> は db2_05_00 になります。

DB2 UDB EEE のインストール時に、セットアップ・プログラムはインスタンス所有マシン上に DAS を作成します。データベース区画サーバーは DAS と同じマシン上に常駐し、インスタンス用の接続点となります。つまり、このデータベース区画サーバーは、コントロール・センターまたはクライアント構成アシスタントからインスタンスに対して発行される要求の調整プログラム・ノードです。

DAS 構成: DAS は、コントロール・センターの代わりに特定のタスクを実行する管理制御点です。物理的なマシン 1 台につき設けることのできる DAS は多くて 1 つです。いくつかのマシンで構成される EEE インスタンスの場合は、コントロール・センターが EEE インスタンスを実行できるよう、少なくとも 1 つのマシンで DAS を実行している必要があります。この DAS (db2as) は、ターゲット DB2 インスタンス (db2inst) の親としてコントロール・センターのナビゲーター・ツリーに存在するシステムを「表して」います。

たとえば、 db2inst は 2 つの物理マシンまたはホストにわたって分散している 3 つのノードから構成されています。 hostA **か** hostB のいずれかで db2das を実行することによって、最小要件を満たすことができます。

注:

1. hostA に存在する区画の数は、 hostA で実行できる DAS の数については何の関係もありません。ホストに複数の論理ノード (MLN) 構成が存在するとしても、 hostA で実行できる db2as のコピーは 1 つだけです。
2. すべてのホストで DAS ID、db2as を作成する必要はありません。むしろ、DAS ID は DAS が実行されるホスト上にのみ存在していなければなりません。また、DAS ID のホーム・ディレクトリーをすべてのホストにマウントする必要もありません。特にこの例では、ID db2as は hostA に存在していなければならず、hostB では必要ありません。また、db2as のホーム・ディレクトリーは hostB にマウントされている必要はありません。

DAS とのコントロール・センター通信: サービス・ポート: コントロール・センターは、TCP サービス・ポートを使用して DAS と通信します。このポートは DB2 UDB 専用として予約されているので、 *tcp_services_file* に新しいエントリーを挿入する必要はありません。

ノード間管理通信: サービス・ポート: いくつかの管理タスクについて、DAS はすべてのノードとの通信を確立する必要があります。そうするためには、インスタンスに参加している各ホストの *tcp_services_file* に、名前付き TCP ポートを定義する必要があります。

注: Windows NT EEE は、 TCP ポート・エントリーを *tcp_services_file* に追加しようとします。

たとえば、db2inst が 2 つのホスト hostA と hostB にわたって定義されています。 91 ページの『環境の例』で指定したとおり、ポート 16000 は両方のホストで未使用です。したがって、次の行を hostA と hostB の両方の *tcp_services_file* に挿入する必要があります。

```
db2ccmsrv  16000/tcp
```

db2ccmsrv ポート名が上記のとおりのつづりで存在していなければならず、選択された同じポート番号がすべてのホストで使用される必要があります。

ノード間管理通信: UNIX DB2 EEE サーバー: hostA と hostB 上で TCP ポート行を *tcp_services_file* に挿入すると、そのインスタンスに参加しているすべてのホストで、管理 listener 処理またはデーモンである db2cclst を開始する必要があります。これは、コマンド行から手動で行うこともできますし、あるいはシステムがブートするたびに db2cclst を自動的に呼び出すようシステムを構成することによっても行えます。

手動: 管理したいインスタンスの ID である db2inst で、次のコマンドを hostA または hostB のいずれかから呼び出します。

```
rah '<install_path>/bin/db2cclst'
```

たとえば、AIX では、このコマンド呼び出しは次のようになります。

```
rah '/usr/lpp/<v_r_ID>/bin/db2cclst'
```

rah コマンドは、バージョンおよびリリース・サブディレクトリーの instance サブディレクトリーにあります。バージョンおよびリリース・サブディレクトリーの厳密な名前は、オペレーティング・システムによって異なります。instance は、使用するインスタンスのホーム・ディレクトリーです。

この例で、 <v_r_ID> はプラットフォーム固有のバージョンとリリースの識別子です。たとえば、DB2 UDB (AIX 版) EEE バージョン 5.2 の場合は、<v_r_ID> は db2_05_00 になります。

自動: root として、次のファイルに適切な項目を追加します。

```
/etc/inittab
```

たとえば、AIX では、このコマンド呼び出しは hostA と hostB で次のように実行されます。

```
mkitab "db2cclst::once:su - db2inst -c
   /usr/lpp/<v_r_ID>/bin/db2cclst"
```

いずれかのマシンがブートするたびに、ユーザー介入なしで db2cclst が呼び出されます。

この表で、 <v_r_ID> はプラットフォーム固有のバージョンとリリースの識別子です。たとえば、DB2 UDB (AIX 版) EEE バージョン 5.2 の場合は、<v_r_ID> は db2_05_00 になります。

各ホストで listener デーモンがアクティブになっていることを確認するには、次のコマンドをインスタンス ID である db2inst で呼び出します。

```
rah 'ps -ef | grep db2cclst'
```

db2cclst 処理が各ホストで実行されていない場合は、各ホストで次の行を /etc/syslog.conf に追加することによって、詳細な診断情報を得ることができます。

```
*.info  /tmp/db2/user.info
```

ここで、ファイル /tmp/db2/user.info は、よりふさわしいファイルで置き換えることができます。

注: このファイルは存在していなければならず、変更が加えられた後に、 SYSLOG デーモンが自らの構成ファイルを再読み取りするようになっている必要があります。

```
kill -1 <syslogd PID>
```

ここで、syslogd PID は次のコマンドを実行することによって得ることができます。

```
ps -ef | grep syslogd
```

上記のようにして手動で listener を呼び出した後、障害の発生したホストの syslog ファイル /tmp/db2/user.info で、 db2cclst によって生成されたエラー・メッセージを見ることができます。

ノード間管理通信: Windows NT DB2 EEE サーバー: DB2 リモート・コマンド・サービス (db2rcmd.exe) は、ノード間管理通信を自動的に処理します。万一障害が発生する場合は、Windows NT レジストリーに診断情報が入れられます。

機密保護: インスタンスに対して DAS が管理タスクを実行するためには、 DAS に十分の権限がなければなりません。特に、DAS はターゲット (管理されるインスタンス) のシステム管理者 (SYSADM) でなければなりません。

DAS には、 DAS が管理するすべての DB2 インスタンスについての同様の権限を授与する必要があります。候補となるインスタンスは、DAS と同じマシンにインストールされているインスタンスです。 DB2 EEE インスタンスについては、上記の説明に適合するよう、少なくとも 1 つのデータベース区画サーバーが、DAS と同じマシンに存在している必要があります。

たとえば、UNIX では、db2inst を管理するのに必要な権限を db2as に授与する方法の 1 つは、 db2inst と db2as の 1 次グループを同一にすることです。あるいは、db2inst の 1 次グループを db2as の 2 次グループとし、 db2as の 1 次グループを db2inst の 2 次グループとするだけでも十分です。最後に、別のオプションとして、 db2inst の SYSADM_GROUP データベース管理構成パラメーターを db2as の 1 次グループに設定する方法があります。

Windows NT では、 db2as は hostA と hostB のローカル管理者グループのメンバーとなっている必要があります。 db2as ID を作成して両方のホストのローカル管理者グループに追加するというオプションに加えて、 db2as のドメイン ID を作成し、この ID を各ホストのローカル管理者グループに追加することもできます。

環境: DAS のインストールでは、正常な操作に必要な特定のレジストリー変数を構成しなければなりません。これらの変数の現行値を確認するには、 DB2 インスタンス ID である db2inst か DAS ID である db2das のいずれかで、次のコマンドを実行します。

```
db2set -g
```

少なくとも以下のパラメーターは、以下に示す値を指定して定義する必要があります。

```
DB2SYSTEM=hostA
DB2ADMINSERVER=db2as
```

同様に、コントロール・センターから DAS と通信するためには、 DB2COMM レジストリー変数が『TCPIP』に設定されていなければなりません。この設定を確認するには、DAS ID である db2as で、次のコマンドを実行して、グローバル (-g) とインスタンス (-i) レベル (設定する必要があるのは 1 つのみ) をチェックします。

```
db2set -all
```

DB2 インスタンスがコントロール・センターと db2inst の通信を確立できるようにするために、 db2inst ID で次のコマンドを出すことによって、 DB2COMM パラメーターが『TCPIP』 に設定されていることを確認します。

```
db2set -all
```

DAS のこのパラメーターを変更する場合は、変更を有効にするために DAS を再始動させる必要があります。このパラメーターが DB2 インスタンス用に変更されると、 DB2 インスタンスの再始動も必要になります。 db2inst については、*db2stop* に続けて *db2start* を出しますが、 *db2admin stop* と *db2admin start* は DAS について出します。

管理サーバー、インスタンス、およびデータベースのディスカバリー: 既知ディスカバリー (Known Discovery) を使用すると、クライアントに認識されているシステム上のインスタンスとデータベースを検出することができます。また、新しいシステムを追加して、そのインスタンスとデータベースを検出することもできます。検索ディスカバリー (Search Discovery) を使用すると、既知ディスカバリーの全機能が提供され、他の DB2 サーバーのためのローカル・ネットワークを検索できるオプションが追加されます。

サーバーで既知ディスカバリーをサポートするには、 DAS 構成ファイル中の *discover* パラメーターを KNOWN に設定します。検索ディスカバリーをサポートするには、このパラメーターを SEARCH に設定します。サーバーとそのインスタンスおよびデータベースの検出を行わないようにするには、このパラメーターを DISABLE に設定します。

注: 検索ディスカバリーによってクライアントに戻される TCP/IP ホスト名は、**hostname** コマンドの入力時に DB2 サーバー・システムによって戻されるのと同じホスト名です。クライアント側では、このホスト名によってマップされる IP アドレスは、クライアント・マシンで構成される TCP/IP ドメイン・ネーム・サーバー (DNS) か、あるいは、DNS が構成されていない場合はクライアントの hosts ファイル中のマッピング・エントリーによって判別されます。 DB2 サーバー・システムに複数のアダプター・カードを構成してある場合は、 TCP/IP が正確なホスト名を戻すようにサーバー上で構成されており、 DNS またはローカル・クライアントの hosts ファイルが任意の IP アドレスにホスト名をマップすることを確かめます。

クライアント側では、*discover* パラメーターを使ってディスカバリー機能も使用可能にします。ただし、この場合の *discover* パラメーターは次のように、クライアント・インスタンス (またはクライアントとして動作するサーバー) で設定します。

- **KNOWN**

  クライアント構成アシスタントを使って既知リスト内のシステムを最新表示できます。また、**「システムの追加 (Add Systems)」**押しボタンを使って、新しいシステムをリストに追加することもできます。 *discover* パラメーターを KNOWN に設定すると、クライアント構成アシスタントはネットワークを検索できなくなります。

- **SEARCH**

  既知ディスカバリーの全機能を使用可能にし、ネットワークを検索することができます。

  「他のシステム (ネットワークの検索) (Other Systems (Search the network))」アイコンは、この選択項目を選択した場合だけ表示されます。これがデフォルト設定です。

- **DISABLE**

  ディスカバリーを使用不能にします。この例では、「データベースの追加ウィザード (Add Database Wizard)」で、**「ネットワークの検索 (Search the network)」**オプションを使用することができません。

注: *discover* パラメーターはデフォルトにより、すべてのクライアントおよびサーバー・インスタンスに SEARCH が設定されます。 *discover* パラメーターはデフォルトにより、すべての DB2 管理サーバー (DAS) に SEARCH が設定されます。ただし、UNIX エンタープライズ拡張エディション環境にインストールされた DAS の場合は、 *discover* のデフォルトとして KNOWN が設定されます。

検索ディスカバリーの追加設定: 検索ディスカバリーでは、 *discover_comm* パラメーターをサーバー (DB2 管理サーバーの構成ファイル内) にも、クライアント (データベース・マネージャーの構成ファイル内) にも設定しなければなりません。

discover_comm パラメーターは、サーバーがクライアントからの検索要求を聴取し、クライアントが検索要求を送出するために使用する通信プロトコルを制御するために使用します。 *discover_comm* パラメーターは TCP/IP または NetBIOS に設定することができます。現在サポートされているプロトコルは TCP/IP と NetBIOS だけです。

DAS では、*discover_comm* に指定された値は、 DB2COMM に設定された値と同等か、そのサブセットでなければなりません。

注: コントロール・センターおよびクライアント構成アシスタントでの問題を回避するには、 **db2set** コマンドを使って、 DB2COMM レジストリー変数が DB2 レジストリーで設定されていることを確かめます。 DB2COMM レジストリー変数の設定に他の方法を使用することはお勧めできません。

サーバーでは、*discover_comm* パラメーターを DAS 構成ファイルで設定します。クライアント (またはクライアントとして動作するサーバー) では、 *discover_comm* をデータベース・マネージャー構成ファイルで設定します。

注: 検索ディスカバリーを使用する場合、 *discover_comm* パラメーターによってクライアントに指定された少なくとも 1 つのプロトコルが、 *discover_comm* パラメーターによって DAS に指定されたプロトコルと一致していなければなりません。一致するものがなければ、サーバーはクライアントの要求に応答しません。

DB2COMM レジストリー変数の設定値を調べるには、次のように入力します。

```
db2set db2comm
```

さらに、 2 つの DB2 プロファイル・レジストリー変数 DB2DISCOVERYTIME と DB2NBDISCOVERYRECVBUFS を使用して、クライアント上で NetBIOS を使って検索ディスカバリーを調整できます。たいていの場合、これらのレジストリー変数のデフォルトが適切な値になります。

DB2DISCOVERYTIME および DB2NBDISCOVERRCVBUFS プロファイル・レジストリー値は、クライアント・インスタンス (またはクライアントとして動作するサーバー) で設定されます。次のようにレジストリー値を設定します。

- DB2DISCOVERYTIME レジストリー値を 60 秒に設定するには、次のコマンドを入力します。

  ```
  db2set db2discoverytime=60
  ```

  このコマンドにより、検索ディスカバリーはサーバーからの応答を 60 秒待つように指示されます。

- DB2NBDISCOVERRCVBUFS レジストリー値を 20 秒に設定するには、次のように入力します。

  ```
  db2set db2nbdiscoverrcvbufs=20
  ```

  このコマンドにより、検出されたサーバーからの並行応答メッセージ用に割り当てられる NetBIOS バッファーの数が指定されます。

ディスカバリーからサーバー・インスタンスおよびデータベースを隠す: サーバー上に複数のインスタンスがあり、それらのインスタンス内に複数のデータベースがあるかもしれません。そのようなインスタンスやデータベースをディスカバリー・プロセスから隠したいと思うことがあります。

クライアントがシステム上のサーバー・インスタンスを検出できるようにするには、システム上の各サーバー・インスタンスの *discover_inst* データベース・マネージャー構成パラメーターを ENABLE (デフォルト値) に設定します。このパラメーターを DISABLE に設定すると、このインスタンスとデータベースをディスカバリーから隠すことができます。

クライアントからデータベースを検出できるようにするには、 *discover_db* データベース構成パラメーターを ENABLE (デフォルト値) に設定します。このパラメーターを DISABLE に設定すると、データベースをディスカバリーから隠すことができます。

ディスカバリー・パラメーターの設定: *discover* および *discover_comm* パラメーターは、サーバー・システム上の DAS 構成ファイル、およびクライアント上のデータベース・マネージャー構成ファイルで設定されます。パラメーターの設定方法は、次のとおりです。

- DAS 上:

  次のコマンド・プロセスを使って DAS 構成ファイルを更新します。

  ```
  update admin cfg using discover [ DISABLE | KNOWN |
     SEARCH ]
  update admin cfg using discover_comm [ NETBIOS | TCPIP ]
  ```

  次のコマンドを入力して DAS を停止し、再始動します。

  ```
  db2admin stop
  db2admin start
  ```

  注: 検索ディスカバリーは NetBIOS および TCP/IP でのみ動作します。

- コントロール・センターを使用する場合は、以下のようにします。

1. クライアント構成アシスタントを開始します。
2. **「クライアント設定 (Client Settings)」**押しボタンをクリックします。
3. **「通信 (Communications)」**タブを選択します。
4. **「パラメーター (Parameters)」**ウィンドウから、修正したいパラメーターを選択します。
5. **「値 (Value)」**ボックスから、修正したいパラメーターの値を選択します。
6. **「了解 (OK)」**押しボタンをクリックして、**「クライアント設定 (Client Settings)」**ウィンドウをクローズします。 DB2 メッセージ・ウィンドウがオープンします。
7. **「了解 (OK)」**押しボタンをクリックし、変更を有効にするためにアプリケーションを再始動します。

注: *discover_comm* に NETBIOS が組み込まれている場合、ワークステーション名 (*nname*) パラメーターがクライアントと DAS の両方に設定されていることを確かめてください。また、使用したいアダプター番号に DB2NBADAPTERS レジストリー値が設定されていることも確かめる必要があります。

コントロール・センターを使って、以下のように *discover_inst* および *discover_db* パラメーターを設定します。

1. オブジェクト・ツリーを順に展開し、**「インスタンス (Instances)」**フォルダーを表示します。
2. インスタンスを右クリックして、ポップアップ・メニューから**「構成 (Configure)」**を選択します。
3. 「環境 (Environment)」ページで、 *discover_inst* パラメーターを選択します。
4. クライアントからサーバー・インスタンスを検出できるようにするには、**「使用可能化 (Enable)」**を選択して、**「OK」**をクリックします。
5. オブジェクト・ツリー内のデータベースを右クリックして、ポップアップ・メニューから**「構成 (Configure)」**を選択します。
6. 「環境 (Environment)」ページで、 *discover_db* パラメーターを選択します。
7. クライアントからデータベースを検出できるようにするには、**「使用可能化 (Enable)」**を選択して、**「OK」**をクリックします。

クライアント構成アシスタントおよびコントロール・センターを使用するための DAS の設定

ネットワーク上のシステムについての情報を検索するには、 DB2 ディスカバリーを構成しなければなりません。 DB2 ディスカバリーは、クライアント構成アシスタントとコントロール・センターが使用する機能です。この機能の構成では、DB2 ディスカバリーによって正確な情報が検索されるよう、インスタンス・リストと DB2 管理サーバー (DAS) の構成を更新しなければならない場合があります。

インスタンス・リストの更新: DB2 管理サーバー (DAS) が区分データベース・システム内のすべてのインスタンスを認識する必要はありません。インスタンスが作成されるとき、最初はインスタンス所有マシン上の DAS だけがそのインスタンスを認識するからです。

DAS を持たないマシン上にインスタンスを作成した場合、そのマシン上に DAS を作成してインスタンスを認識させることができます。

複数の DAS を作成し、区分データベース・システム内のインスタンスすべてを個々の DAS に認識させるには、以下のステップを実行します。

1. **それぞれの DAS ごとに**

   管理サーバー・マシンで **db2ilist** コマンドを実行し、この DAS に認識されているインスタンスのリストを表示します。

   注: インスタンスのリストが完全であれば、残りのステップを実行しなくても次のセクションに進むことができます。

2. **上記のステップのインスタンス・リストで個々のインスタンスが欠落している場合**

インスタンス所有マシンで、**db2nlist** コマンドを実行して、 DAS を持つマシン用のエントリーがあるかどうかを調べます。エントリーがなければ、**db2ncrt** コマンドを実行して、インスタンスにこのマシンを追加します。

注: DAS マシンでは、インスタンス用にネットワーク共用ドライブを使用できなければなりません。

DAS 構成の更新

デフォルトにより、セットアップ・プログラムは DB2SYSTEM レジストリー変数を Windows NT のコンピューター名に設定します。ディスカバリーによって検索されるシステム名は、 DB2 管理サーバー (DAS) が常駐するシステムです。接続が確立されると、ディスカバリーは検索したシステムを調整プログラム・ノードとして使用します。

DAS 構成を更新するには、次の 2 つの方法があります。

- DB2 システムから調整プログラム・ノードを選択できるようにするには、個々の DB2 管理サーバーの構成ファイルで DISCOVER=SEARCH (デフォルト値) を設定します。

  複数の DAS があると、クライアント構成アシスタントまたはコントロール・センターのインターフェース上の複数のシステムに同じインスタンスが表示される場合がありますが、各システムにはインスタンスへの異なる通信アクセス・パスが指定されます。ユーザーは、通信用の調整プログラム・ノードとして、異なる DB2 システムを選択することができます。
- ユーザーが調整プログラム・ノードを選択できないようにしたい場合は、すべての DAS に対して DISCOVER=KNOWN を設定します。ただし、DAS 構成内の 1 つの DAS だけに対しては DISCOVER=SEARCH を設定します。ディスカバリーは、接続が確立されるときに DAS が調整プログラム・ノードとして常駐するデータベース区画サーバーを使用します。

ノード構成ファイルの作成

データベースを区分データベース環境で操作する場合、 db2nodes.cfg というノード構成ファイルを作成する必要があります。このファイルは、並列機能を持ったデータベース・マネージャーを複数区画にわたって開始する前に、そのインスタンスに対するホーム・ディレクトリーの sqllib サブディレクトリーの中に配置されていなければなりません。このファイルには、1 つのインスタンスの中のすべてのデータベース区画の構成情報が含まれ、そのインスタンスのすべてのデータベース区画によって共用されます。

Windows NT に関する考慮事項: DB2 エンタープライズ拡張エディションを Windows NT で使用している場合は、インスタンスを作成したときにノード構成ファイルが作成されます。ノード構成ファイルは手動で作成したり変更したりしないでください。

注: インスタンスが削除された場合にデータの消失を避けるため、 sqllib サブディレクトリーには、 DB2 によって作成されたもの以外のファイルまたはディレクトリーを作成しないでください。ただし、以下の 2 つの例外があります。システムがストアード・プロシージャーをサポートしている場合は、ストアード・プロシージャー・アプリケーションを sqllib サブディレクトリーの下の function サブディレクトリーに入れます。 (ストアード・プロシージャーについての詳細は、 *管理の手引き: パフォーマンス* の『ストアード・プロシージャー』を参照してください。) もう 1 つの例外は、ユーザー定義特殊関数 (UDF) が作成される場合です。 UDF の実行可能コードは、同じディレクトリーに入れることが許されます。

ファイルには、1 つのインスタンスに属する各データベース区画ごとに 1 行が含まれます。それぞれの行は、以下の形式になっています。

```
nodenum hostname [logical-port [netname]]
```

トークンはブランクで区切られます。変数は、以下のとおりです。

nodenum

ノードを固有に定義するノード番号 (0 ～ 999 が可能)。ノード番号は、昇順でなければなりません。間の番号が抜けていてもかまいません。

いったんノード番号が割り当てられると、それを変更することはできません。(変更すると、データを区分する方法を指定する区分化マップの中の情報が信用できないものになります。)

ノードを除去した場合、そのノード番号は、追加する任意の新しいノードに再使用することができます。

ノード番号は、データベース・ディレクトリーの中にノード名を生成するために使用されます。ノード名は、以下の形式になります。

```
NODEnnnn
```

nnnn はノード番号で、左側はゼロで埋められます。このノード番号は、CREATE DATABASE コマンドおよび DROP DATABASE コマンドによっても使用されます。

hostname

区画間通信のための IP アドレスのホスト名。 (netname が指定された場合は、例外です。この場合、netname がほとんどの通信で使用され、 hostname は DB2START、DB2STOP、および db2_all でのみ使用されます。)

logical-port

このパラメーターの指定は任意であり、ノードの論理ポート番号を指定します。この番号は、データベース・マネージャーのインスタンス名と一緒に使用され、 etc/services ファイルの中の TCP/IP サービス名項目を識別します。

IP アドレスと論理ポートの組み合わせは、既知のアドレスとして使用され、ノード間の通信接続をサポートするために、すべてのアプリケーション内で固有なものでなければなりません。

各 *hostname* について、 1 つの *logical-port* は、 0 かまたはブランク (0 がデフォルト) でなければなりません。この *logical-port* に関連付けられるノードは、クライアントが接続するホスト上のデフォルトのノードです。これは、 db2profile スクリプト内の DB2NODE 環境変数か、または sqlesetc() API を使用してオーバーライドすることができます。

同じホスト上に複数のノードがある場合 (つまり、 1 つのホストに対して複数の *nodenum* がある場合)、 *logical-port* 番号を論理ノードに、0 から昇順に、間の番号を抜かさずに割り当てる必要があります。

netname

このパラメーターの指定は任意であり、それぞれが独自のホスト名を持つ、複数の活動状態の TCP/IP インターフェースを持ったホストをサポートするために使用されます。

以下の例は、RS/6000 SP システムのための可能なノード構成ファイルを示したものであり、 SP2EN1 が複数の TCP/IP インターフェースと 2 つの論理ノードを持ち、 DB2 ユニバーサル・データベースのインターフェースとして SP2SW1 を使用しています。この例は、ノード番号が 1 (0 ではなく) から開始しており、 *nodenum* の順番は間の番号が抜けていることも示しています。

```
nodenum   hostname   logical-port   netname
1         SP2EN1     0              SP2SW1
2         SP2EN1     1              SP2SW1
4         SP2EN2     0
5         SP2EN3
```

好みのエディターを使用して、db2nodes.cfg ファイルを更新することができます。 (例外: Windows NT ではエディターは使用しないでください。) ただし、データ区分化ではノード番号を変更してはならないため、ファイル内の情報の保全性を注意して保護する必要があります。ノード構成ファイルは、DB2START を出したときにロックされ、DB2STOP がデータベース・マネージャーを停止させた後でロックが解除されます。ファイルがロックされている場合、DB2START コマンドで、必要に応じて、ファイルを更新することができます。たとえば、RESTART オプションまたは ADDNODE オプションを指定して DB2START を出すことができます。

注: DB2STOP コマンドが失敗し、ノード構成ファイルがロック解除されない場合、ロック解除するために DB2STOP FORCE を出してください。

データベース構成ファイルの作成

データベースごとにデータベース構成ファイル も作成されます。このファイルの作成は管理者のために行われます。このファイルには、データベースの使用に影響を与える、次のような様々な構成パラメーター の値が入れられます。

- データベースの作成時に指定または使用されるパラメーター (データベース・コード・ページ、照合順序、DB2 リリース・レベルなど)
- データベースの現行の状態を示すパラメーター (バックアップ保留フラグ、データベース一貫性フラグ、ロールフォワード操作保留フラグなど)
- データベースの操作時に使用されるシステム・リソースの量を定義するパラメーター (バッファー・プール・サイズ、データベース・ログ、分類メモリー・サイズなど)

これらのパラメーターについては、 *管理の手引き: パフォーマンス* の『DB2 の構成』に詳しく記されています。

構成ファイル内のパラメーターは手動で変更しないでください。サポートされているインターフェースだけを使用する必要があります。

パフォーマンス上のヒント: 構成パラメーターの多くはデフォルト値が提供されていますが、データベースの最適なパフォーマンスを達成するためには構成パラメーターの更新が必要な場合があります。

複数区画の場合: 複数の区画にわたって区分化されたデータベースを持っている場合、構成ファイルは、すべてのデータベース区画で同じものである必要があります。 SQL コンパイラーは、ローカル・ノードの構成ファイルの情報に基づいて分散 SQL ステートメントをコンパイルし、 SQL ステートメントのニーズを満足させるためのアクセス・プランを作成するので、これらの構成ファイルには整合性が必要です。データベース区画ごとに異なる構成ファイルを維持していると、どのデータベース区画でステートメントが準備されたかによって、異なるアクセス・プランが作成される可能性があります。 **db2_all** を使用して、すべてのデータベース区画で構成ファイルの同期を保ってください。

応答ファイルを使った構成情報の複製

db2rspgn という応答ファイル生成ユーティリティーを使用すれば、システムを再インストールするとき、または現行システムのレジストリー値、データベース・マネージャー構成パラメーター、およびデータベース管理構成パラメーターを、同一のシステムに複製したいときに使える応答ファイルを作成できます。

あるシステムに 1 つまたは複数の DB2 製品をインストールしたら、 *db2rspgn* を使って必要な値を応答ファイルに生成できます。その後、応答ファイルは同一のシステムを再作成するために使用できます。

コマンド行構文では、応答ファイルおよびサポート・ファイルがある場合に、その宛先ディレクトリーを宣言します。さらに、コピーしたいインスタンスを指定することもで

きます。また、管理インスタンスまたは DataLink サーバー・インスタンス、あるいはその両方を使用不能にすることもできます。大規模な展開については、*サテライト管理手引きおよび解説書* を参照してください。

このユーティリティーの構文についての詳細、および生成された応答ファイルの使用方法についての説明は、該当する*概説およびインストール* を参照してください。

FCM 通信の使用可能化

区分データベース環境では、データベース区画間のほとんどの通信は、高速コミュニケーション・マネージャー (FCM) によって処理されます。データベース区画で FCM を使用可能にし、他のデータベース区画との通信ができるようにするには、下記に示すように、区画の `etc` ディレクトリーの `services` ファイル内にサービス項目を作成する必要があります。 FCM は、指定されたポートを使用して通信を行います。同じホスト上に複数の区画を定義している場合、以下に示すように、ある範囲のポートを定義しなければなりません。

Windows NT に関する考慮事項

DB2 エンタープライズ拡張エディションを Windows NT 環境で使用している場合、 TCP/IP のポート範囲は次のものによって自動的にサービス・ファイルに追加されます。

- インストール・プログラムがインスタンスを作成したり新しいノードを追加したりするときに、インストール・プログラムによって
- db2icrt ユーティリティーが新しいインスタンスを作成するときに、 db2icrt ユーティリティーによって
- db2ncrt ユーティリティーがマシンに最初のノードを追加したときに、db2ncrt ユーティリティーによって

詳細については、*DB2 エンタープライズ拡張エディション (Windows 版) 概説およびインストール* を参照してください。

サービス項目の構文は、以下のとおりです。

```
DB2_instance port/tcp #comment
```

DB2_*instance*

instance の値は、データベース・マネージャーのインスタンスの名前です。名前の中のすべての文字は小文字でなければなりません。 `db2puser` というインスタンス名であるとすれば、 `DB2_db2puser` というように指定します。

port/**tcp**

データベース区画のために予約したい TCP/IP ポート。

#*comment*

この項目と関連付けたい任意の注釈。注釈の前には、ポンド記号 (#) を付けなければなりません。

/etc/services ファイルが共用されている場合、ファイル内に割り当てられるポートの数は、そのインスタンス内の複数のデータベース区画の最大数と等しいかそれより大きくなるようにしなければなりません。ポートを割り当てる場合には、バックアップとして使用できるプロセッサーもその数の中に入れるようにしなければなりません。

/etc/services ファイルが共用されていない場合は、同じ考慮事項が適用されます。ただし、DB2 インスタンスのために定義された項目が、すべての /etc/services ファイルで同じになるようにしなければならない (区分データベースに適用されない項目は、同じである必要はありません) という追加の考慮事項があります。

1 つのインスタンス内で同じホスト上に複数のデータベース区画がある場合、使用する FCM のために複数のポートを定義しなければなりません。このためには、etc/services ファイルの中に 2 行を組み込んで、割り当てるポートの範囲を示します。最初の行は最初のポートを指定し、2 番目の行は複数のポート・ブロックの終わりを示します。以下の例では、sales というインスタンスに 5 つのポートが割り当てられます。これは、そのインスタンスには、 5 つを超えるデータベース区画を持つプロセッサーはないことを意味します。

```
DB2_sales         9000/tcp
DB2_sales_END     9004/tcp
```

注: END は、大文字でのみ指定しなければなりません。また、両方の下線 (_) 文字も含めるようにしなければなりません。

第3章 データベースの作成

この章では、データベース設計のインプリメンテーションを構成する、さまざまなオブジェクトを個々に簡潔に説明しています。

前の章では、データベースを作成する前に知っている必要がある情報を中心に説明しました。また、いくつかのトピックや、データベースの作成前に済ませておくべきタスクも記載されていました。

この部の最後の章では、データベースを更新する前に考慮しなければならないトピックが示されています。また、データベース・オブジェクトの更新方法や除去方法について説明されています。

データベースの作成時には、以下のタスクがそれぞれ実行されます。

- データベースで必要になるすべてのシステム・カタログ表を設定する。
- データベースの回復ログを割り当てる。
- データベースの構成ファイルを作成し、デフォルト値を設定する。
- データベースのユーティリティーをデータベースにバインドする。

システム・カタログ視点上の CREATETAB、BINDADD、CONNECT、IMPLICIT_SCHEMA、および SELECT の各データベース特権は、 PUBLIC に付与されます。

コントロール・センターを使用してデータベースを作成するには、以下のようにします。

1. オブジェクト・ツリーを順に展開し、**「データベース (Databases)」**フォルダーを表示します。
2. **「データベース (Databases)」**フォルダーを右クリックして、ポップアップ・メニューから**「作成 (Create)」—>「ウィザードを使用してデータベースを (Database Using Wizard)」**を選択します。
3. ステップに従ってこのタスクを完了します。

以下のコマンド行プロセッサー・コマンドは、デフォルトの位置に `personl` と呼ばれるデータベースを、 ″Personnel DB for BSchiefer Co″ という関連する注釈を付けて作成します。

```
create database personl
   with "Personnel DB for BSchiefer Co"
```

データベースをデータベース・マネージャーの別のインスタンス (おそらくは、リモート) に作成する場合は、 57ページの『データベース・マネージャーの複数インスタンスを使用する』を参照してください。このトピックでは、さらに、デフォルト・インスタンス以外のインスタンス (リモート・インスタンスを含む) に対してインスタンス・レベルの管理を実行する場合に使用しなければならないコマンドについて紹介しています。

注: デフォルトのデータベース位置、および CREATE DATABASE コマンドを使用して異なる位置を指定する方法については、 *コマンド解説書* を参照してください。

データベース作成時に自分で、あるいはデータベース・マネージャーが行うタスクについて、次の部分で説明します。

- 109ページの『最初のノード・グループの定義』
- 109ページの『最初の表スペースの定義』
- 111ページの『システム・カタログ表の定義』
- 111ページの『データベース・ディレクトリーの定義』
- 113ページの『DCE ディレクトリー・サービス』
- 114ページの『Lightweight Directory Access Protocol (LDAP) ディレクトリー・サービス』
- 115ページの『データベース回復ログの定義』
- 116ページの『データベースへのユーティリティーのバインド』
- 116ページの『データベースのカタログ化』
- 114ページの『ノード・グループの作成』
- 117ページの『表スペースの作成』
- 124ページの『スキーマの作成』
- 126ページの『表の作成とデータの読み込み』
- 143ページの『トリガーの作成』
- 145ページの『ユーザー定義関数 (UDF) または方式の作成』
- 149ページの『ユーザー定義タイプ (UDT) の作成』
- 151ページの『視点の作成』
- 154ページの『要約表の作成』
- 156ページの『別名の作成』
- 158ページの『ラッパーの作成』
- 159ページの『サーバーの作成』
- 167ページの『ニックネームの作成』
- 169ページの『索引の作成、索引拡張子、または索引の指定』

データベースの物理的な実現に関する追加情報については、 *管理の手引き: 計画* を参照してください。

最初のノード・グループの定義

データベースを最初に作成するときに、 db2nodes.cfg ファイルに指定されたすべての区画について、データベース区画が作成されます。その他の区画は、 ADD NODE コマンドおよび DROP NODE コマンドを使用して追加または除去することができます。

以下の 3 つのノード・グループが定義されます。

- SYSCATSPACE 表スペース用の IBMCATGROUP (システム・カタログ表を保持します)
- TEMPSPACE1 表スペース用の IBMTEMPGROUP (データベース処理の間に作成された一時表を保持します)
- USERSPACE1 表スペース用の IBMDEFAULTGROUP (デフォルトで、ユーザー表と索引を保持します)

最初の表スペースの定義

データベースを作成するとき、以下の 3 つの表スペースが定義されます。

- システム・カタログ表用の SYSCATSPACE (111ページの『システム・カタログ表の定義』を参照)
- データベース処理中に作成されたシステム一時表用の TEMPSPACE1
- ユーザー定義の表および索引用の USERSPACE1

注: データベースを初めて作成する時点では、ユーザー一時表スペースは作成されません。

CREATE DATABASE コマンドを使用してどの表スペース・パラメーターも指定していない場合は、システム管理ストレージ (SMS) のディレクトリー・コンテナーを使用して、データベース・マネージャーがこれらの表スペースを作成します。ディレクトリー・コンテナーはデータベースのサブディレクトリーに作成されます (データベースの物理ディレクトリーについて詳しくは、 *管理の手引き: 計画* を参照)。これらの表スペースのエクステント・サイズはデフォルトに設定されます。

コントロール・センターを使用して初期表スペースを定義するには、以下のようにします。

1. オブジェクト・ツリーを順に展開し、**「データベース (Databases)」**フォルダーを表示します。
2. **「データベース (Databases)」**フォルダーを右クリックして、ポップアップ・メニューから**「作成 (Create)」—>「ウィザードを使用してデータベースを (Database Using Wizard)」**を選択します。
3. ステップに従ってこのタスクを完了します。

コマンド行を使用して初期表スペースを定義するには、以下のように入力します。

```
CREATE DATABASE <name>
  CATALOG TABLESPACE
    MANAGED BY SYSTEM USING ('<path>')
    EXTENTSIZE <value> PREFETCHSIZE <value>
  USER TABLESPACE
    MANAGED BY DATABASE USING (FILE'<path>' 5000,
                               FILE'<path>' 5000)
    EXTENTSIZE <value> PREFETCHSIZE <value>
  TEMPORARY TABLESPACE
    MANAGED BY SYSTEM USING ('<path>')
  WITH "<comment>"
```

表スペースにデフォルト定義を使用したくない場合は、 CREATE DATABASE コマンドでこれらの特性を指定することができます。たとえば、次のコマンドは OS/2 上にデータベースを作成するために使用するものです。

```
CREATE DATABASE PERSONL
  CATALOG TABLESPACE
    MANAGED BY SYSTEM USING ('d:¥pcatalog','e:¥pcatalog')
    EXTENTSIZE 16 PREFETCHSIZE 32
  USER TABLESPACE
    MANAGED BY DATABASE USING (FILE'd:¥db2data¥personl' 5000,
                               FILE'd:¥db2data¥personl' 5000)
    EXTENTSIZE 32 PREFETCHSIZE 64
  TEMPORARY TABLESPACE
    MANAGED BY SYSTEM USING ('f:¥db2temp¥personl')
  WITH "Personnel DB for BSchiefer Co"
```

この例では、最初の表スペースのそれぞれの定義が明示的に提供されています。デフォルト定義を使用したくない表スペースの表スペース定義だけを指定するだけで済みます。

CREATE DATABASE コマンドの MANAGED BY 句をコーディングした後に、CREATE TABLESPACE コマンドの MANAGED BY 句と同じ形式が続きます。その他の例については、117ページの『表スペースの作成』を参照してください。

データベースを作成する前に、 *管理の手引き: 計画* および表スペースの設計と選択に関する情報を参照してください。

システム・カタログ表の定義

1 組のシステム・カタログ表が、それぞれのデータベースごとに作成され維持されます。これらの表には、データベース・オブジェクト (たとえば、表、視点、索引、およびパッケージ) の定義についての情報と、これらのオブジェクトに対してユーザーが持っているアクセスのタイプについての機密保護の情報が含まれています。これらの表は、SYSCATSPACE 表スペースに保管されます。

これらの表は、表作成時など、データベースの操作中に更新されます。これらの表を明示的に作成したり除去したりすることはできませんが、内容の照会や表示は可能です。データベースが作成されると、システム・カタログ表オブジェクトに加えて、次のデータベース・オブジェクトがシステム・カタログで定義されます。

- 一連のユーザー定義関数 (UDF) が SYSFUN スキーマに作成されます。システムで作成される関数についての詳細は、*SQL 解説書* を参照してください。
- システム・カタログ表の一連の読み取り専用の視点が SYSCAT スキーマに作成されます。これらの視点については、*SQL 解説書* の『カタログ視点』を参照してください。
- 一連の更新可能なカタログ視点が SYSSTAT スキーマに作成されます。更新可能な視点を使用すると、特定の統計情報を使用して、仮定データベースのパフォーマンスを調査したり、 RUNSTATS ユーティリティーを使用しないで統計を更新したりすることができます。これらの視点については、 *SQL 解説書* の『更新可能なカタログ視点』を参照してください。

データベースが作成された後で、282ページの『システム・カタログ視点の機密保護』に記載されているように、システム・カタログ視点のアクセスを制限することができます。

データベース・ディレクトリーの定義

以下の 3 つのディレクトリーが、新しいデータベースの確立またはセットアップのときに使用されます。

- ローカル・データベース・ディレクトリー
- システム・データベース・ディレクトリー
- ノード・ディレクトリー

ローカル・データベース・ディレクトリー

データベースが定義されているパス (オペレーティング・システムによっては「ドライブ」と呼ばれる) ごとに、*ローカル・データベース・ディレクトリー* ・ファイルが 1 つずつあります。このディレクトリーには、そこからアクセスできるデータベースごとに 1 つの項目が入っています。各項目には次のものが含まれています。

- CREATE DATABASE コマンドによって提供されたデータベース名

- データベースの別名 (別名が指定されない場合は、データベース名と同じ)
- データベースを説明する注釈 (CREATE DATABASE コマンドで提供されたもの)
- データベースのためのルート・ディレクトリーの名前
- その他のシステム情報

特定のデータベースについてこのファイルの内容を見るためには、以下のコマンドを出します (ただし、*location* はデータベースの位置を指定します)。

```
LIST DATABASE DIRECTORY ON location
```

システム・データベース・ディレクトリー

システム・データベース・ディレクトリー・ファイルは、データベース・マネージャーの各インスタンスごとに存在し、このインスタンスについてカタログされているそれぞれのデータベースごとに 1 つの項目が含まれています。データベースは CREATE DATABASE コマンドの発行時に暗黙のうちにカタログ化されますが、 CATALOG DATABASE コマンドによって明示的にカタログ化することもできます。データベースのカタログ化については、116ページの『データベースのカタログ化』を参照してください。

作成されたそれぞれのデータベースごとに 1 つの項目がディレクトリーに追加されますが、これには以下の情報が含まれます。

- CREATE DATABASE コマンドによって提供されたデータベース名
- データベースの別名 (別名が指定されない場合は、データベース名と同じ)
- CREATE DATABASE コマンドによって提供されたデータベース注釈
- *ローカル・データベース・ディレクトリー* の位置
- データベースが*間接* データベースであることを示す標識。これは、データベースがシステム・データベース・ディレクトリーのファイルとして同じマシン上にあるという意味です。
- その他のシステム情報

このファイルの内容を見るためには、データベース・ディレクトリー・ファイルの位置を*指定せずに* LIST DATABASE DIRECTORY コマンドを出します。

区分データベース環境では、すべてのデータベース区画が、同じシステム・データベース・ディレクトリー・ファイル (そのインスタンスのホーム・ディレクトリーの `sqldbdir` サブディレクトリーの中にある `sqldbdir`) を常にアクセスするようにしなければなりません。同じ `sqldbdir` サブディレクトリーの中の、システム・データベース・ディレクトリーまたはシステム・インテンション・ファイル `sqldbins` のいずれかが、共用ファイル・システム上にある別のファイルに対する記号リンクである場合、予期しないエラーが発生する可能性があります。これらのファイルについては、61ページの『データ区分化の使用可能化』で説明します。

ノード・ディレクトリー

データベース・マネージャーは、最初のデータベース区画がカタログされるときにノード・ディレクトリーを作成します。データベース区画をカタログするためには、CATALOG NODE コマンドを使用します。ローカルのノード・ディレクトリーの内容をリストするには、 LIST NODE DIRECTORY コマンドを使用します。ノード・ディレクトリーは、各データベース・クライアントごとに作成され維持されます。ディレクトリーには、そのクライアントがアクセスできる 1 つ以上のデータベースを持っている各リモート・ワークステーションごとに 1 つの項目が入っています。 DB2 クライアントは、データベース接続またはインスタンス接続が要求されると、ノード・ディレクトリーの中の通信エンドポイント情報を使用します。

ディレクトリーの中の項目には、クライアントからリモート・データベース区画に通信するために使用される、通信プロトコルのタイプについての情報も含まれます。ローカル・データベース区画をカタログすることによって、同じマシン上に常駐するインスタンスに対する別名が作成されます。ユーザーのクライアントからアクセスする同じワークステーションに複数のインスタンスがある場合には、ローカル・ノードをカタログする必要があります。

DCE ディレクトリー・サービス

DCE は、オープン・ソフトウェア・ファウンデーション (OSF) アーキテクチャーの 1 つであり、分散異種コンピューティング環境におけるアプリケーションの作成、使用、および保守をサポートするためのツールとサービスを提供します。これは、オペレーティング・システム、ネットワーク、および、クライアント・アプリケーションからリモート・サーバーへのアクセスを可能にする分散アプリケーションとの間にある層です。

ローカル・ディレクトリーの場合、ターゲット・データベースの物理的な位置は、クライアント・ワークステーションごとに、データベース・ディレクトリーとノード・ディレクトリーとに個別に格納されます。したがってデータベース管理者は、これらのディレクトリーの更新と変更にかなりの時間を費やすことになります。 DCE ディレクトリー・サービスには、ローカル・ディレクトリーの代わりになる中央ディレクトリーが用意されています。これによって、データベースまたはデータベース・マネージャーのインスタンスの情報を中央位置に記録しておき、変更や更新はその位置で行うことができるようになります。

DCE は DB2 を実行するための前提条件ではありませんが、 DCE 環境での操作を行う場合は、337ページの『付録B. 分散コンピューティング環境 (DCE) ディレクトリー・サービスの使用』を参照してください。

Lightweight Directory Access Protocol (LDAP) ディレクトリー・サービス

Lightweight Directory Access Protocol (LDAP) は、ディレクトリー・サービスに対する業界標準のアクセス方式です。ディレクトリー・サービスとは、分散環境内にある複数のシステムおよびサービスについてのリソース情報を集めたリポジトリーです。クライアントとサーバーはディレクトリー・サービスを使用して、それらのリソースにアクセスします。各データベース・サーバーのインスタンスは自らの存在を LDAP サーバーに知らせるとともに、データベースの作成時にはデータベース情報を LDAP ディレクトリーへ送信します。クライアントがデータベースに接続すると、 LDAP ディレクトリーからそのサーバーのカタログ情報を取り出せます。各クライアントは、それぞれのマシンでローカルにカタログ情報を保管する必要はなくなります。クライアント・アプリケーションは、LDAP ディレクトリーの中で、データベースへ接続するのに必要な情報を探します。

LDAP は DB2 を実行するための前提条件ではありませんが、 LDAP 環境での操作を行う場合は、421ページの『付録J. Lightweight Directory Access Protocol (LDAP) ディレクトリー・サービス』を参照してください。

ノード・グループの作成

CREATE NODEGROUP ステートメントを使用してノード・グループを作成します。このステートメントは、表スペース・コンテナーおよび表データが常駐するノードのセットを指定します。このステートメントは、以下のことも行います。

- ノード・グループの区分化マップの作成。区分化マップについての詳細は、 *管理の手引き: 計画* を参照してください。
- 区分化マップ ID の生成。
- 以下のカタログ表へのレコードの挿入。
  - SYSCAT.NODEGROUPS
  - SYSCAT.PARTITIONMAPS
  - SYSCAT.NODEGROUPDEF

コントロール・センターを使用してノードグループを作成するには、以下のようにします。

1. オブジェクト・ツリーを順に展開し、**「ノードグループ (Nodegroups)」**フォルダーを表示します。
2. **「ノードグループ (Nodegroups)」**フォルダーを右クリックして、ポップアップ・メニューから**「作成 (Create)」**を選択します。
3. 「ノードグループの作成 (Create Nodegroups)」ウィンドウで、情報をすべて入力し、矢印を使用してノードを**「使用可能ノード (Available nodes)」**ボックスから**「選択ノード (Selected nodes)」**ボックスに移動して、**「Ok」**をクリックします。

コマンド行を使用してノードグループを作成するには、以下のように入力します。

```
CREATE NODEGROUP <name> ON NODES (<value>,<value>)
```

データベース内のデータベース区画のサブセット上にいくつかの表をロードしたいとします。以下のコマンドを使用して、少なくとも 3 つのノード (0 ～ 2) からなるデータベース内に、 2 つのノード (1 と 2) のノード・グループを作成します。

```
CREATE NODEGROUP mixng12 ON NODES (1,2)
```

ノード・グループの作成についての詳細は、*SQL 解説書* を参照してください。

CREATE DATABASE コマンドまたは sqlecrea() API は、デフォルトのシステム・ノード・グループである、 IBMDEFAULTGROUP、IBMCATGROUP、および IBMTEMPGROUP も作成します。 (ノード・グループについての詳細は、*管理の手引き : 計画* を参照してください。)

データベース回復ログの定義

データベース回復ログ は、新しい表の追加または既存の表に対する更新を含む、データベースに対して行われたすべての変更の記録を保持します。このログはいくつかの*ログ・エクステント* からなり、それぞれのログ・エクステントは、*ログ・ファイル* と呼ばれる別個のファイルに入っています。

データベース回復ログを使用して、障害 (たとえば、システム電源異常またはアプリケーション・エラー) によって、データベースが矛盾した状態のままにならないようにすることができます。障害が発生した場合、すでに入力はされたがコミットされていない変更事項はロールバックされ、コミット済みのすべてのトランザクション (ディスクに物理的に書き込まれてはいないかもしれない) は再実行されます。こうしたアクションによって、データベースの保全性が保たれます。

詳細については、*データ移動ユーティリティー手引きおよび解説書* を参照してください。

データベースへのユーティリティーのバインド

データベースの作成時に、データベース・マネージャーによって db2ubind.lst 内のユーティリティーをデータベースにバインドすることが試みられます。このファイルは、sqllib ディレクトリーの bnd サブディレクトリーに格納されています。

ユーティリティーをバインドすると、パッケージ が作成されます。これは、1 つのソース・ファイルからの特定の SQL ステートメントを処理するのに必要な情報がすべて入れられているオブジェクトです。

注: クライアントからこれらのユーティリティーを使用する場合は、ユーティリティーを明示的にバインドする必要があります。詳細については、使用しているプラットフォーム用の*概説およびインストール* を参照してください。

何らかの理由で、データベースにユーティリティーをバインドまたは再バインドする必要がある場合、コマンド行プロセッサーを使用して、以下のコマンドを出します。

```
connect to sample
bind @db2ubind.lst
```

注: sample データベースにパッケージを作成するには、これらのファイルが入っているディレクトリーを使用しなければなりません。バインド・ファイルは、 SQLLIB ディレクトリーの BND サブディレクトリーの中にあります。この例では、sample がデータベースの名前です。

データベースのカタログ化

新しいデータベースを作成すると、システム・データベース・ディレクトリーのファイルに自動的にカタログ化されます。また、CATALOG DATABASE コマンドを使って、システム・データベース・ディレクトリーのファイルにデータベースを明示的にカタログ化することもできます。 CATALOG DATABASE コマンドを使えば、違う別名でデータベースをカタログ化したり、 UNCATALOG DATABASE コマンドによって以前に削除したデータベース項目をカタログ化したりすることが可能になります。

次のコマンド行プロセッサー・コマンドを使うと、 personl データベースが humanres としてカタログ化されます。

```
catalog database personl as humanres
   with "Human Resources Database"
```

この場合、システム・データベース・ディレクトリー項目のデータベース別名は humanres になります。これは、データベース名 (personl) とは違うものです。

デフォルト以外のインスタンスにデータベースをカタログ化することもできます。次の例では、データベース B への接続は、 INSTANCE_C に対して行われます。

```
catalog database b as b at node instance_c
```

注: クライアント・ノードで CATALOG DATABASE コマンドを使うと、データベース・サーバー・マシンにあるデータベースをカタログ化することもできます。詳しくは、使用しているプラットフォーム用の*概説およびインストール* を参照してください。

分散コンピューティング環境 (DCE) のセル・ディレクトリーについては、113ページの『DCE ディレクトリー・サービス』と 337ページの『付録B. 分散コンピューティング環境 (DCE) ディレクトリー・サービスの使用』を参照してください。

注: パフォーマンスを向上させるために、ディレクトリー・ファイル (データベース・ディレクトリーを含む) をメモリーにキャッシュすることがあります。 (ディレクトリー・キャッシュを使用可能にする方法については、 *管理の手引き: パフォーマンス* の『ディレクトリー・キャッシュ・サポート (dir_cache)』を参照してください。) ディレクトリー・キャッシュが使用可能な場合は、別のアプリケーションがディレクトリーを変更しても (たとえば、 CATALOG DATABASE または UNCATALOG DATABASE コマンドを使用して)、ユーザーのアプリケーションが再始動されるまで、その変更は効力を持ちません。コマンド行プロセッサー・セッションが使用するディレクトリー・キャッシュを最新表示するには、 `db2 terminate` コマンドを出してください。

アプリケーション・レベルのキャッシュに加えて、データベース・マネージャー・レベルのキャッシュも、内部的なデータベース・マネージャーの索引に使用されます。この『共用』キャッシュを最新表示するには、 `db2stop` および `db2start` コマンドを出してください。

ディレクトリー・キャッシュについての詳細は、 *管理の手引き: パフォーマンス* の『ディレクトリー・キャッシュ・サポート (dir_cache)』を参照してください。

表スペースの作成

データベースの中に表スペースを作成すると、その表スペースにコンテナーが割り当てられ、その定義と属性がデータベース・システム・カタログに記録されます。このようにして、その表スペースに表を作成できるようになります。

表スペースの設計情報については、*管理の手引き: 計画* を参照してください。

CREATE TABLESPACE ステートメントの構文については、 *SQL 解説書* で詳細に説明されています。 SMS と DMS の表スペースについては、*管理の手引き: 計画* を参照してください。

コントロール・センターを使用して表スペースを作成するには、以下のようにします。

1. オブジェクト・ツリーを順に展開し、**「表スペース (Table spaces)」**フォルダーを表示します。
2. **「表スペース (Table spaces)」**フォルダーを右クリックして、ポップアップ・メニューから**「作成 (Create)」—>「ウィザードを使用して表スペースを (Table Spaces Using Wizard)」**を選択します。
3. このウィザードのステップに従って、タスクをすべて実行します。

コマンド行を使用して SMS 表スペースを作成するには、以下のように入力します。

```
CREATE TABLESPACE <NAME>
   MANAGED BY SYSTEM
   USING ('<path>')
```

コマンド行を使用して DMS 表スペースを作成するには、以下のように入力します。

```
CREATE TABLESPACE <NAME>
   MANAGED BY DATABASE
   USING (FILE'<path>' <size>)
```

次の SQL ステートメントは、別個の 3 つのドライブの 3 つのディレクトリーを使用して、 OS/2 または Windows NT 上で SMS 表スペースを作成するものです。

```
CREATE TABLESPACE RESOURCE
   MANAGED BY SYSTEM
   USING ('d:¥acc_tbsp', 'e:¥acc_tbsp', 'f:¥acc_tbsp')
```

次の SQL ステートメントは、それぞれ 5,000 ページの各ファイル・コンテナーを使用して、 OS/2 上に DMS 表スペースを作成します。

```
CREATE TABLESPACE RESOURCE
   MANAGED BY DATABASE
   USING (FILE'd:¥db2data¥acc_tbsp' 5000,
          FILE'e:¥db2data¥acc_tbsp' 5000)
```

上記の 2 つの例では、コンテナーに明示的な名前が提供されました。しかし、相対コンテナー名を指定する場合、コンテナーはデータベース用に作成されたサブディレクトリーの中に作成されます (データベースの物理ディレクトリーについて詳しくは、 *管理の手引き: 計画* を参照)。

さらに、指定されたパス名の一部が存在しない場合は、データベース・マネージャーが作成します。サブディレクトリーがデータベース・マネージャーに作成される場合は、表スペースが除去されると、そのサブディレクトリーもデータベース・マネージャーによって削除されます。

上記の例は、表スペースが特定のノード・グループに関連付けられていないことを前提としています。デフォルトのノード・グループである IBMDEFAULTGROUP は、以下のパラメーターがステートメント内に指定されていない場合に使用されます。

```
IN nodegroup
```

以下の SQL ステートメントは、それぞれ 10 000 ページの 3 つの論理ボリュームを使用して UNIX ベースのシステム上に DMS 表スペースを作成し、それらの入出力特性を指定します。

```
CREATE TABLESPACE RESOURCE
   MANAGED BY DATABASE
   USING (DEVICE '/dev/rdblv6' 10000,
          DEVICE '/dev/rdblv7' 10000,
          DEVICE '/dev/rdblv8' 10000)
   OVERHEAD 24.1
   TRANSFERRATE 0.9
```

この SQL ステートメントで指定している UNIX デバイスは、すでに存在しているものであり、インスタンス所有者および SYSADM グループが書き込みを行えるようでなければなりません。

以下の例は、UNIX 区分データベース内の ODDNODEGROUP と呼ばれるノード・グループ上に DMS 表スペースを作成します。 ODDNODEGROUP は、CREATE NODEGROUP ステートメントを使用してあらかじめ作成されていなければなりません。この場合、ODDNODEGROUP ノード・グループは、1、3、および 5 の番号が付いたデータベース区画から成っていると想定されています。すべてのデータベース区画上で、10 000 の 4 KB ページについて、デバイス /dev/hdisk0 を使用します。さらに、40 000 の 4 KB ページから成るデータベース区画ごとに 1 つのデバイスを宣言します。

```
CREATE TABLESPACE PLANS
   MANAGED BY DATABASE
   USING (DEVICE '/dev/HDISK0' 10000, DEVICE '/dev/n1hd01' 40000) ON NODE 1
         (DEVICE '/dev/HDISK0' 10000, DEVICE '/dev/n3hd03' 40000) ON NODE 3
         (DEVICE '/dev/HDISK0' 10000, DEVICE '/dev/n5hd05' 40000) ON NODE 5
```

UNIX デバイスは次の 2 つの区分に分類されます。それは、文字シリアル・デバイスとブロック構造デバイスです。すべてのファイル・システム・デバイスの場合、各ブロック・デバイス (つまりクックド・デバイス) ごとに、対応する文字シリアル・デバイス (つまりロー・デバイス) を持っているのが普通です。ブロック構造デバイスは、普通、『hd0』 または 『fd0』 のような名前で指定されます。文字シリアル・デバイスは、普通、『rhd0』、『rfd0』、または 『rmt0』 のような名前で指定されます。これらの文字シリアル・デバイスは、ブロック・デバイスよりも速いアクセス速度を持っています。文字シリアル・デバイス名は、CREATE TABLESPACE コマンド上で使用する必要があり、ブロック・デバイス名は使用できません。

オーバーヘッドと転送速度は、SQL ステートメントのコンパイル時に使用する最善のアクセス・パスを決定するための目安になります。 OVERHEAD パラメーターと TRANSFERRATE パラメーターについては、 *管理の手引き: パフォーマンス* の『アプリケーション・パフォーマンスのチューニング』を参照してください。

DB2 で順次事前取り出し機能を使用すると、並列入出力が使われ、順次入出力のパフォーマンスを大幅に向上させることができます。この機能についての詳細は、 *管理の手引き: パフォーマンス* の『順次事前取り出しについて』を参照してください。

デフォルトの 4 KB サイズより大きいページ・サイズを使用する表スペースを作成することもできます。次の SQL ステートメントは、UNIX ベースのシステムで 8 KB の SMS 表スペースを作成するものです。

```
CREATE TABLESPACE SMS8K
   PAGESIZE 8192
   MANAGED BY SYSTEM
   USING ('FSMS_8K_1')
   BUFFERPOOL BUFFPOOL8K
```

関連したバッファー・プールのページ・サイズも、同じ 8 KB でなければならないことに注意してください。

作成された表スペースは、それが参照するバッファー・プールが活動化されるまで使用できません。

ALTER TABLESPACE SQL ステートメントを使うと、 DMS 表スペースにコンテナーを追加したり、表スペースの PREFETCHSIZE、OVERHEAD、および TRANSFERRATE の設定値を変更したりすることができます。表スペース・ステートメントを発行するトランザクションは、システム・カタログの競合を避けるために、できるだけ早くコミットする必要があります。

注: PREFETCHSIZE は、EXTENTSIZE の整数倍でなければなりません。たとえば、EXTENTSIZE が 10 なら PREFETCHSIZE は 20 や 30 などにする必要があります。詳しくは、*管理の手引き: パフォーマンス* の『順次事前取り出しについて』を参照してください。

システム一時表スペースの作成

システム一時表スペースは、システム一時表を格納するのに使用されます。データベースを作成すると、定義される 3 つのデフォルト表スペースのうち 1 つが、『TEMPSPACE1』 というシステム一時表スペースになります。

注: システム一時表を格納できるのはシステム一時表スペースだけなので、データベースは常に少なくとも 1 つのシステム一時表スペースを持っていなければなりません。

CREATE TABLESPACE ステートメントを使用して、別のシステム一時表スペースを作成することもできます。たとえば、次のようにします。

```
CREATE SYSTEM TEMPORARY TABLESPACE tmp_tbsp
   MANAGED BY SYSTEM
   USING ('d:¥tmp_tbsp','e:¥tmp_tbsp')
```

システム一時表スペースの作成時に指定できるノードグループは、 IBMTEMPGROUP だけです。

ユーザー一時表スペースの作成

ユーザー一時表スペースは、宣言された一時表を格納するのに使用されます。

CREATE TABLESPACE ステートメントを使用して、ユーザー一時表スペースを作成できます。

```
CREATE USER TEMPORARY TABLESPACE usr_tbsp
   MANAGED BY DATABASE
   USING (FILE 'd:¥db2data¥user_tbsp' 5000,
          FILE 'e:¥db2data¥user_tbsp' 5000)
```

正規表スペースと同様、ユーザー一時表スペースは IBMTEMPGROUP 以外のノードグループに作成できます。ユーザー一時表スペースの作成時には、 IBMDEFAULTGROUP がデフォルトのノードグループとして使用されます。

DECLARE GLOBAL TEMPORARY TABLE ステートメントは、宣言された一時表を、ユーザー一時表スペース内で使用できるように定義します。

ノード・グループ内の表スペースの作成

複数データベース区画ノード・グループの中に表スペースを置くことによって、その表スペース内のすべての表が、そのノード・グループ内の各データベース区画にわたって分割、つまり区分化されます。表スペースはノード・グループ内に作成されます。いったん 1 つのノード・グループ内に入ると、表スペースは、そこにとどまらなければならず、別のノード・グループに変更することはできません。 CREATE TABLESPACE ステートメントは、表スペースとノード・グループを関連付けるために使用されます。

ロー I/O

DB2 ユニバーサル・データベースは、直接ディスク・アクセス (ロー I/O) をサポートしています。したがって、直接ディスク・アクセス (ロー) デバイスを DB2 ユニバーサル・データベース・システムに接続することができます。 (Windows 95、および Windows 98 オペレーティング・システムだけは例外です。) 以下のリストでは、この種類のデバイスを識別するための物理的また論理的方法が示されています。

- Windows で物理ハード・ディスクを指定するには、次の構文を使用します。

  ¥¥.¥PhysicalDriveN

ここで N は、システムにある物理ドライブのいずれかを表します。この場合、N を 0、1、2、または他の正の整数に置き換えることができます。

¥¥.¥PhysicalDisk5

- Windows で論理生区画 (つまり未フォーマットの区画) を指定するには、次の構文を使用します。

  ¥¥.¥N:

  ここで N: は、システムにある論理ドライブ名を表します。たとえば、N: を E: または他のドライブ名で置き換えることができます。
- **注:** デバイスにログを書き込むには、 Windows NT バージョン 4.0 (サービス・パック 3 適用済み) をインストールしておく必要があります。
- UNIX ベースのプラットフォームでは、文字のシリアル・デバイス名 (/dev/rhd0 など) を使用します。

Linux でのロー I/O の使用

Linux には、ブロック・デバイスにバインドしなければロー入出力を実行できない、ロー・デバイス・ノードのプールがあります。ロー・デバイスをブロック・デバイスにバインドするための情報の中央リポジトリーとして機能するロー・デバイス・コントローラーがあります。バインドは、一般的に Linux ディストリビューターによって提供されているユーティリティー raw を使用して実行されます。

Linux でロー I/O をセットアップするには、以下が必要になります。

- 1 つ以上の IDE または SCSI 空きディスク区分
- Linux カーネル 2.4.0 以降 (ただし、Linux 配布版によってはカーネル 2.2 でもロー入出力が提供されている場合があります。)
- ロー・デバイス・コントローラー /dev/rawctl または /dev/raw。 これらのコントローラーがない場合、次のようにしてシンボリック・リンクを作成してください。

  ```
  # ln -s /dev/your_raw_dev_ctrl /dev/rawctl
  ```
- 通常、Linux 配布版で提供される raw ユーティリティー
- DB2 バージョン 7.1 フィックスパック 3 以降

注: ロー・デバイス・ノード名は、現在ロー入出力をサポートしている配布版によって異なります。

配布版	ロー・デバイス・ノード	ロー・デバイス・コントローラー
RedHat 6.2	/dev/raw/raw1 to 255	/dev/rawctl
SuSE 7.0	/dev/raw1 to 63	/dev/raw

DB2 は、これら 2 つのロー・デバイス・コントローラー、およびロー・デバイス・ノードのその他ほとんどの名前をサポートしています。ロー・デバイスは、Linux/390 での DB2 ではサポートされていません。

Linux でロー入出力を構成するには:

この例では、使用されるロー区分は /dev/sda5 です。この区分には、重要なデータは含まれていないと想定します。

ステップ 1. この区分の 4 096 バイト・ページの数を計算します。端数が出た場合は切り捨てます。たとえば、次のようにします。

```
# fdisk /dev/sda
Command (m for help): p

Disk /dev/sda: 255 heads, 63 sectors, 1106 cylinders
Units = cylinders of 16065 * 512 bytes

   Device Boot   Start    End    Blocks   Id  System
/dev/sda1            1    523   4200997   83  Linux
/dev/sda2          524   1106   4682947+   5  Extended
/dev/sda5          524   1106   4682947   83  Linux

Command (m for help): q
#
```

/dev/sda5 のページ数:

```
num_pages = floor( ((1106-524+1)*16065*512)/4096 )
num_pages = 11170736
```

ステップ 2. 未使用のロー・デバイス・ノードをこの区分にバインドします。マシンをリブートするたびに行う必要があり、ルート・アクセスが必要です。 raw -a を使用して、どのロー・デバイス・ノードが使用されているかを調べます。

```
# raw /dev/raw/raw1 /dev/sda5
/dev/raw/raw1: bound to major 8, minor 5
```

ステップ 3. ロー・デバイス・コントローラーおよびディスク区分にグローバル読み取りアクセス権を設定します。ロー・デバイスには、グローバル読み取り / 書き込みアクセス権を設定します。

```
# chmod a+r /dev/rawctl
# chmod a+r /dev/sdb1
# chmod a+rw /dev/raw/raw1
```

ステップ 4. ディスク区分ではなくロー・デバイスを指定して、DB2 に表スペースを作成します。たとえば、次のようにします。

```
CREATE TABLESPACE dms1
MANAGED BY DATABASE
USING (DEVICE '/dev/raw/raw1' 11170736)
```

ロー・デバイスの表スペースは、DB2 でサポートされているその他すべてのページ・サイズでもサポートされています。

スキーマの作成

データを表に編成すれば、表 (他の関連オブジェクトも) を一緒にグループ化する利点もあります。これは、CREATE SCHEMA ステートメントを使用して、スキーマを定義することによって行うことができます。スキーマについての情報は、接続するデータベースのシステム・カタログ表に保持されます。他のオブジェクトが作成されると、それらのオブジェクトをこのスキーマ内に置くことができます。

CREATE SCHEMA ステートメントの構文についての詳細は、 *SQL 解説書* に説明があります。新しいスキーマ名は、すでにシステム・カタログに存在する名前にすることはできず、 ″SYS″ で始めることはできません。

ユーザーが SYSADM 権限または DBADM 権限を持っている場合、そのユーザーは、任意の有効な名前でスキーマを作成することができます。データベースが作成されるときに、IMPLICIT_SCHEMA 権限が PUBLIC (つまり、すべてのユーザー) に付与されます。

CREATE SCHEMA ステートメントの一部として作成されたどのオブジェクトの定義者も、スキーマの所有者になります。この所有者は、他のユーザーに対して、スキーマ特権の GRANT および REVOKE を行うことができます。

このステートメントは、DBADM 権限を持つユーザーが出されなければなりません。

スキーマは、ユーザーが IMPLICIT_SCHEMA 権限を持ったときに、暗黙に作成されることもあります。この権限を使用して、ユーザーは、すでに存在していないスキーマ名を持つオブジェクトを作成するときに、常に暗黙にスキーマを作成します。

ユーザーが IMPLICIT_SCHEMA 権限を持っていない場合、ユーザー自身の許可 ID と同じ名前のスキーマだけを作成することができます。

スキーマはデータベース内での固有性を強制するために使用されるので、スキーマ内のオブジェクトに対する直接アクセスは許可されません。 2 人のユーザーが 2 つの表 (または他のオブジェクト) を同時に作成できるかどうかを考慮する際に、このことが明らかになります。スキーマを使用して固有性を強制しないと、 3 人めのユーザーが表を照会しようとする場合にあいまいさが生じます。もっと詳しく制限されていなければ、どの表を使用すべきかを判別できません。

表名に対する制限の一部としてスキーマ名を入力しなくても、別のユーザーが表にアクセスできるようにするには、そのユーザーについて視点を設定する必要があります。視点の定義では、完全修飾表名 (ユーザーのスキーマを含む) を定義します。そうすれ

ば、ユーザーは視点名を使用して照会するだけで済みます。視点は、ユーザーのスキーマによって視点定義の一部として完全修飾されています。

コントロール・センターを使用してスキーマを作成するには、以下のようにします。

1. オブジェクト・ツリーを順に展開し、**「スキーマ (Schema)」**フォルダーを表示します。
2. **「スキーマ (Schema)」**フォルダーを右クリックして、ポップアップ・メニューから**「作成 (Create)」**を選択します。
3. 新しいスキーマに関する情報をすべて入力し、**「Ok」**をクリックします。

コマンド行を使用してスキーマを作成するには、以下のように入力します。

```
CREATE SCHEMA <name> AUTHORIZATION <name>
```

以下は、許可 ID ″joe″ を持つ個人ユーザーについてのスキーマを作成する、 CREATE SCHEMA ステートメントの例です。

```
CREATE SCHEMA joeschma AUTHORIZATION joe
```

スキーマの設定

特定の DB2 接続内から発行された動的 SQL ステートメント中の修飾されていないオブジェクト参照が使用できるよう、デフォルトのスキーマを設定するとします。これは、デフォルトとして使用したいスキーマに、特殊レジスター CURRENT SCHEMA を設定することによって行えます。この特殊レジスターは、どのユーザーでも設定することができます。設定するための許可は必要ありません。

SET SCHEMA ステートメントの構文についての詳細は、 *SQL 解説書* に説明があります。

以下に CURRENT SCHEMA 特殊レジスターの設定方法の例を示します。

```
SET CURRENT SCHEMA = 'SCHEMA01'
```

このステートメントは、アプリケーション・プログラム内から使用するか、あるいは対話式に発行することができます。一度設定されると、CURRENT SCHEMA 特殊レジスターの値は、データベース・オブジェクトへの修飾されていない参照が存在する CREATE SCHEMA ステートメント以外の動的 SQL ステートメント用の修飾子 (スキーマ) として使用されます。

CURRENT SCHEMA 特殊レジスターの初期値は、現行セッション・ユーザーの許可 ID と同じになります。

表の作成とデータの読み込み

表内でのデータの編成方法を決めたなら、次の段階は、CREATE TABLE ステートメントを使って表を作成することです。表の説明は、接続先のデータベースのシステム・カタログの中に格納されます。

CREATE TABLE ステートメントの構文についての詳細は、 *SQL 解説書* に説明があります。要約表を作成することについては、 154ページの『要約表の作成』を参照してください。表、列、その他のデータベース・オブジェクトの命名については、 329ページの『付録A. 命名規則』を参照してください。

CREATE TABLE ステートメントを使うと、表の名前として修飾付きまたは修飾なしの識別子が付けられ、表の各列の定義が与えられます。表ごとに別々の表スペースに保管して、 1 つの表スペースには 1 つの表しか含まれないようにすることができます。表の除去や作成を頻繁に行う場合は、表をそれぞれ別の表スペースに格納し、表ではなく表スペースを除去するほうが効率的です。 1 つの表スペース内に多数の表を保管することもできます。区分データベース環境では、選択された表スペースは、表データが保管されるノード・グループおよびデータベース区画も定義します。

表には、当初、何もデータが入っていません。表にデータ行を加えるには、次のどちらかを使用します。

- INSERT ステートメント (*SQL 解説書* に説明があります)
- LOAD または IMPORT コマンド (*コマンド解説書* に説明があります)
- 区分データベース環境で作業している場合は、オートローダー・ユーティリティー (*データ移動ユーティリティー手引きおよび解説書* に説明があります)

表データの出し入れについての詳細は、*データ移動ユーティリティー手引きおよび解説書* に記されています。

変更内容をログに記録せずに、表にデータを追加することができます。 CREATE TABLE ステートメントの NOT LOGGED INITIALLY 文節を使用すると、変更内容は表にログ記録されません。表が作成されたのと同じ作業単位内の INSERT、DELETE、UPDATE、CREATE INDEX、DROP INDEX、または ALTER TABLE の操作によって表に対して行われた変更は、いずれもログに記録されません。ログ記録は、後続の作業単位で開始します。

表には、1 つまたは複数の列定義が含まれています。 1 つの表について、最大 500 列を定義することができます。列は、エンティティーの属性を表します。 1 つの列の値は、すべて同じタイプの情報です。詳しくは、*SQL 解説書* を参照してください。

注: 4 KB のページ・サイズを使用しているときには、最大で 500 列です。ページ・サイズが 8 KB、16 KB、または 32 KB のときは、最大で 1012 列です。

列定義には、*列名*、*データ・タイプ*、およびもし必要なら*ヌル値属性* またはデフォルト値 (ユーザーがオプションで選択します) が含まれます。

列名は列に含まれる情報について記述したもので、簡単に識別できるものにすべきです。これはその表内では固有なものでなければなりませんが、他の表では同じ名前を使用することができます。命名規則については、329ページの『付録A. 命名規則』を参照してください。

列のデータ・タイプは、列の値の長さと有効なデータの種類を示すものです。データベース・マネージャーで使うデータ・タイプには、文字ストリング、数値、日付、時間、ラージ・オブジェクトがあります。漢字ストリング・データ・タイプは、マルチバイト文字セットを使用するデータベース環境だけで利用できます。また、ユーザー定義特殊タイプで列を定義することもできます。この点については、149ページの『ユーザー定義タイプ (UDT) の作成』を参照してください。

デフォルト属性の指定は、値が指定されていない場合にどの値を使用するかを指定するものです。デフォルト値を指定するか、またはシステム定義のデフォルト値を使用することができます。デフォルト値は、ヌル値属性を指定した列でも指定しない列でも指定することができます。

ヌル値属性仕様は、列にヌル値を含めることができるかどうかを示します。

コントロール・センターを使用して表を作成するには、以下のようにします。

1. オブジェクト・ツリーを順に展開し、**「表 (Tables)」**フォルダーを表示します。
2. **「表 (Tables)」**フォルダーを右クリックして、ポップアップ・メニューから**「作成 (Create)」—>「ウィザードを使用して表を (Tables Using Wizard)」**を選択します。
3. このウィザードのステップに従って、タスクをすべて実行します。

コマンド行を使用して表を作成するには、以下のように入力します。

```
CREATE TABLE <NAME>
   (<column_name>  <data_type>  <null_attribute>)
   IN <TABLE_SPACE_NAME)
```

RESOURCE 表スペースに EMPLOYEE 表を作成するための CREATE TABLE ステートメントの例を次に示します。この表はサンプル・データベースの中で定義されています。

```
CREATE TABLE EMPLOYEE
   (EMPNO     CHAR(6)     NOT NULL PRIMARY KEY,
    FIRSTNME  VARCHAR(12) NOT NULL,
    MIDINIT   CHAR(1)     NOT NULL WITH DEFAULT,
    LASTNAME  VARCHAR(15) NOT NULL,
```

```
    WORKDEPT  CHAR(3),
    PHONENO   CHAR(4),
    PHOTO     BLOB(10M)   NOT NULL)
IN RESOURCE
```

表を作成するときは、表の列を構造型の属性に基づいたものとなるよう選ぶことができます。そのような表を「タイプ付き表」といいます。

タイプ付き表は、別のタイプ付き表から列の一部を継承するように定義できます。そのような表を「副表」といい、副表から継承された表を「スーパー表」といいます。タイプ付き表とすべての副表を組み合わせたものを「表階層」といいます。表階層内の最上部にある表 (副表を持たない表) を階層の「ルート表」といいます。

以下の部分では、この例を基にして、考慮すべき他の項目を取り上げます。

- 『ラージ・オブジェクト (LOB) 列の考慮事項』
- 130ページの『固有制約の定義』
- 135ページの『新しい表に生成列を定義する』
- 136ページの『ユーザー定義一時表の作成』
- 137ページの『識別列を新しい表に定義する』
- 138ページの『シーケンスの作成』
- 139ページの『IDENTITY 列とシーケンスの比較』
- 140ページの『タイプ付き表の作成』
- 140ページの『タイプ付き表へのデータの読み込み』
- 140ページの『階層表』
- 141ページの『複数の表スペースへの表の作成』
- 142ページの『区分データベースへの表の作成』

照会の結果に基づいて定義される表を作成することもできます。この種の表は、*要約表*と呼ばれます。詳しくは、154ページの『要約表の作成』を参照してください。

ラージ・オブジェクト (LOB) 列の考慮事項

ラージ・オブジェクト列が入っている表を作成する前に、次の事柄を決定する必要があります。

1. LOB 列に対する変更をログに記録したいか。

   これらの変更をログに記録したくない場合は、表の作成時に NOT LOGGED 文節を指定して、ログ記録をオフにする必要があります。

```
CREATE TABLE EMPLOYEE
   (EMPNO     CHAR(6)     NOT NULL PRIMARY KEY,
    FIRSTNME  VARCHAR(12) NOT NULL,
    MIDINIT   CHAR(1)     NOT NULL WITH DEFAULT,
    LASTNAME  VARCHAR(15) NOT NULL,
```

```
    WORKDEPT  CHAR(3),
    PHONENO   CHAR(4),
    PHOTO     BLOB(10M)   NOT NULL  NOT LOGGED)
IN RESOURCE
```

LOB 列の大きさが 1 GB を超える場合は、ログ記録をオフにしなければなりません。 (目安として、大きさが 10 MB を超える LOB 列はログに記録することができません。) 列定義に指定した他のオプションと同様、ログ記録オプションを変更するための唯一の方法は、表を再作成することです。

変更をログに記録することを選択しなくても、 LOB 列にはシャドー が作成されて、ロールバックがシステム生成エラーの結果であるか、アプリケーションの要求であるかによって、変更をロールバックすることができます。シャドーの作成は、現行のストレージの内容が上書きされない場所では、回復技法となります。つまり、古くなった未修正ページは 『シャドー』・コピーとして保持されます。これらのコピーは、トランザクション・ロールバックのサポートに必要ではなくなると、破棄されます。

注: RESTORE および ROLLFORWARD コマンドを使用してデータベースを回復するとき、『NOT LOGGED』 になっており、最後のバックアップで書き込まれた LOB データは *2 進ゼロで置き換えられます*。

2. LOB 列のために必要なスペースを最小化したいか。

   CREATE TABLE ステートメントの COMPACT 文節を使用すると、 LOB 列をできる限り小さくすることができます。たとえば、次のようにします。

```
CREATE TABLE EMPLOYEE
   (EMPNO     CHAR(6)     NOT NULL PRIMARY KEY,
    FIRSTNME  VARCHAR(12) NOT NULL,
    MIDINIT   CHAR(1)     NOT NULL WITH DEFAULT,
    LASTNAME  VARCHAR(15) NOT NULL,
    WORKDEPT  CHAR(3),
    PHONENO   CHAR(4),
    PHOTO     BLOB(10M)   NOT NULL  NOT LOGGED  COMPACT)
IN RESOURCE
```

   特に LOB 値のサイズが (行わなければならないストレージ調整のために) 増えた場合には、 LOB 列を小さくした表を付加するときに、*パフォーマンス上のコスト* がかかります。

   予備ファイルの割り振りがサポートされておらず、 LOB が SMS 表スペースに置かれている、OS/2 のようなプラットフォームでは、 COMPACT 文節を使用することを検討してください。予備ファイルの割り振りは、オペレーティング・システムが物理ディスク・スペースを使用する方法と関係があります。予備ファイルの割り振りをサポートするオペレーティング・システムは、予備ファイルの割り振りをサポートしないオペレーティング・システムに比べ、 LOB の保管にそれほどの物理ディスク容量を使用しません。 COMPACT オプションを指定すると、予備ファイルの割り振りがサポートされているかどうかに関係なく、大量の物理ディスク容量を 『節約』 す

ることができます。 COMPACT を使用した場合には物理ディスク容量が節約できるので、オペレーティング・システムが予備ファイルの割り振りをサポートしていない場合には、 COMPACT を使用することを検討すべきです。

注: DB2 システム・カタログは LOB 列を使用しており、以前のバージョンよりも多くのスペースを使用します。

3. DB2 システム・カタログ内の LOB 列も含む LOB 列について、パフォーマンスの向上を望むか。

   カタログ表の中にラージ・オブジェクト (LOB) 列があります。 LOB データは他のデータと一緒にバッファー・プールの中には保持されず、必要になるたびにディスクから読み取られます。ディスクからの読み取りによって、カタログの LOB 列が含まれている場合には、 DB2 のパフォーマンスが低下します。ファイル・システムは通常データを保管 (またはキャッシュ) するための独自の場所を持っているので、SMS 表スペースを使用するか、ファイル・コンテナー上に作成された DMS 表スペースを使用して、 LOB が以前に参照されている場合に入出力が発生する可能性を避けるようにしてください。

制約の定義

この節では、制約の定義方法について説明します。

- 『固有制約の定義』
- 131ページの『参照制約の定義』
- 134ページの『表検査制約の定義』

制約についての詳細は、 *管理の手引き: 計画* の制約の計画に関する節と、 *SQL 解説書* を参照してください。

固有制約の定義

固有制約 は、指定されたキー内のそれぞれの値が固有なものになるようにします。 1 つの表は、最大 1 つの固有制約を基本キーとして定義して、任意の数の固有制約を持つことができます。

CREATE TABLE または ALTER TABLE ステートメントの UNIQUE 文節を使用して固有制約を定義します。固有キーは複数の列で構成できます。 1 つの表上で、複数の固有制約が許されます。ただし、副表に固有制約を定義することはできません。

いったん確立されると、 INSERT または UPDATE ステートメントが表内のデータを修正するときに、データベース・マネージャーによって、その固有制約が自動的に施行されます。固有制約は、固有索引を通して施行されます。

固有制約が ALTER TABLE ステートメントに定義され、その固有キーの同じ列のセット上に索引が存在する場合、その索引は固有索引となり、制約によって使用されます。

任意の固有制約を 1 つとって、それを基本キー として使用することができます。基本キーは、(他の固有制約と一緒に) 参照制約の中の親キーとして使用することができます。 1 つの表当たり 1 つの基本キーだけが可能です。基本キーを定義するには、CREATE TABLE または ALTER TABLE ステートメントで PRIMARY KEY 文節を使います。基本キーには、複数の列を含めることができます。

1 次索引は、基本キーの値が固有なものとなるように施行します。基本キーの指定された表を作成すると、データベース・マネージャーはその基本キーについての 1 次索引を作成します。

固有制約として使用される索引に対するパフォーマンス上のヒントには、以下のものがあります。

- 索引のある空の表の初期ロードを実行するときは、 IMPORT よりも LOAD のほうがパフォーマンスはよくなります。これは、LOAD の INSERT または REPLACE のいずれのモードを使用していても同じです。
- 索引のある既存の表に、大量のデータを追加するときには (IMPORT INSERT または LOAD INSERT を使用)、 IMPORT よりも LOAD のほうがパフォーマンスは多少よくなります。
- IMPORT コマンドを使用して最初の大量のデータのロードを行う場合は、データがインポートされた後で固有キーを作成してください。こうすれば、表のロード時に索引保守のためのオーバーヘッドを避けることができます。さらに、索引が使用する記憶域を最小限にすることができます。
- REPLACE モードで ロード・ユーティリティーを使用している場合は、データをロードする前に固有キーを作成してください。 この場合、ロード中に索引を作成するほうが、ロードの後で CREATE INDEX ステートメントを使用するより効率的です。

参照制約の定義

表定義と列定義に参照制約を追加すると、参照保全が課せられます。参照制約は、CREATE TABLE または ALTER TABLE ステートメントの FOREIGN KEY 文節および REFERENCES 文節を使用して確立されます。タイプ付き表に対する参照制約や、タイプ付き表である親表に対する参照制約の効果についての詳細は、 *SQL 解説書* を参照してください。

外部キーの指定によって、1 つの表の行内または 2 つの表の行間の値について、制約が施行されます。データベース・マネージャーは、表定義で指定された制約を検査し、それに応じて関係を維持します。その目標は、1 つのデータベース・オブジェクトが別のオブジェクトを参照するときに、いつでも整合性が保たれているようにすることです。

たとえば、基本キーと外部キーは、それぞれ部署番号の列を持ちます。 EMPLOYEE 表の場合、列名は WORKDEPT であり、DEPARTMENT 表の場合、列名は DEPTNO です。この 2 つの表の間の関係は、次の制約によって定義されています。

- EMPLOYEE 表の各従業員の部署番号は 1 つだけであり、その番号は DEPARTMENT 表の中にも存在しています。
- EMPLOYEE 表の各行は、DEPARTMENT 表の 1 つの行だけと関連があります。表同士の間には、一意の関連があります。
- EMPLOYEE 表の行のうち WORKDEPT の値がヌル値でないものは、それぞれ DEPARTMENT 表の DEPTNO 列の 1 つの行と関連しています。
- DEPARTMENT 表は親表であり、EMPLOYEE 表は従属表です。

親表である DEPARTMENT を定義する SQL ステートメントは、次のとおりです。

```
CREATE TABLE DEPARTMENT
   (DEPTNO    CHAR(3)     NOT NULL,
    DEPTNAME  VARCHAR(29) NOT NULL,
    MGRNO     CHAR(6),
    ADMRDEPT  CHAR(3)     NOT NULL,
    LOCATION  CHAR(16),
       PRIMARY KEY (DEPTNO))
IN RESOURCE
```

従属表である EMPLOYEE を定義する SQL ステートメントは、次のとおりです。

```
CREATE TABLE EMPLOYEE
   (EMPNO     CHAR(6)     NOT NULL PRIMARY KEY,
    FIRSTNME   VARCHAR(12)  NOT NULL,
    LASTNAME   VARCHAR(15)  NOT NULL,
    WORKDEPT   CHAR(3),
    PHONENO    CHAR(4),
    PHOTO     BLOB(10m)   NOT NULL,
       FOREIGN KEY DEPT (WORKDEPT)
       REFERENCES DEPARTMENT ON DELETE NO ACTION)
IN RESOURCE
```

DEPTNO 列を DEPARTMENT 表の外部キーとして指定し、 WORKDEPT を EMPLOYEE 表の外部キーとして指定すると、 WORKDEPT 値についての参照制約を定義することになります。この制約は、2 つの表の間での値の参照保全を施行するものとなります。この場合、EMPLOYEE 表に追加される従業員の部署番号は、DEPARTMENT 表の中にあるものでなければなりません。

EMPLOYEE 表の参照制約の削除規則は、NO ACTION です。つまり、DEPARTMENT 表から部署を削除しようとしても、その部署に従業員がいれば削除はできません。

この例では CREATE TABLE ステートメントを使って参照制約を追加していますが、ALTER TABLE ステートメントを使うこともできます。 198ページの『構造と内容の両方における表の修正』を参照してください。

別の例: 前の例の中で使用されたのと同じ表定義が使用されます。また、DEPARTMENT 表は、EMPLOYEE 表より前に作成されます。各部署にはマネージャー

がおり、そのマネージャーは EMPLOYEE 表にリストされています。
DEPARTMENT 表の MGRNO 番号は、実際には EMPLOYEE 表の外部キーになっています。この参照サイクルのために、この制約にはわずかに問題があります。外部キーは後で追加することができます (202ページの『基本キーと外部キーの追加』を参照)。また、CREATE SCHEMA ステートメントを使用して、EMPLOYEE 表と
DEPARTMENT 表の両方を同時に作成することもできます (*SQL 解説書* の例を参照)。

FOREIGN KEY 文節: 外部キーは、同じ表または別の表内の基本キーまたは固有キーを参照します。外部キーを割り当てると、指定した参照制約にしたがって参照保全が維持されます。外部キーを定義するには、 CREATE TABLE または ALTER TABLE ステートメントで FOREIGN KEY 文節を使います。

外部キーの中の列の数は、親表の対応する基本制約または固有制約 (親キーと呼ばれる) の中の列の数と同じでなければなりません。さらに、キー列定義の対応する部分は、それぞれ同じデータ・タイプ、同じ長さでなければなりません。外部キーには、*制約名* を割り当てることができます。名前は、割り当てなくても自動的に割り当てられます。使いやすさのためには、自分で*制約名* を割り当て、システムが生成した名前は使用しないようにすることをお勧めします。

複合外部キーの値は、外部キーの各列の値が親キーの対応する列の値と**等しければ**、親キーの値と一致します。ヌル値が含まれる外部キーは、親キーが定義上ヌル値を持つことができないため、親キーの値と一致することはありません。しかし、外部キーのヌル値は、ヌル値ではないどの部分の値とも無関係に常に有効です。

外部キー定義に適用される規則は、次のとおりです。

- 1 つの表に複数の外部キーが可能です。
- 外部キーのいずれかの部分がヌル値可なら、その外部キーはヌル値可です。
- 外部キーのいずれかの部分がヌルならば、その外部キーの値はヌルです。

REFERENCES 文節: REFERENCES 文節は、関係の中の親表を指定し、必要な制約を定義するためのものです。この文節は列定義の中に含めることもできますし、
CREATE TABLE または ALTER TABLE ステートメントの中に
FOREIGN KEY 文節を伴う別個の文節として含めることもできます。

REFERENCES 文節を列制約として指定する場合、暗黙の列リストは、指定する列名で構成されることになります。複数の列にそれぞれ別個の REFERENCES 文節を使うことも可能ですし、 1 つの列で複数の文節を使うことも可能です。

REFERENCES 文節には削除規則が含まれています。この例の中で使用される規則は、ON DELETE NO ACTION です。つまり、部署に従業員が配属されているなら、その部署は削除できません。削除規則としては、このほかに ON DELETE CASCADE、ON DELETE SET NULL、および ON DELETE RESTRICT があります。参照保全を実装する際の DELETE 規則について詳しくは、 *管理の手引き: 計画* を参照してください。

ユーティリティー操作に及ぼす影響: ロード・ユーティリティーは自己参照と従属表の制約検査をオフにして、これらの表を検査保留状態にします。 ロード・ユーティリティーが完了してから、オフになっていたすべての表の制約検査をオンにする必要があります。 たとえば、DEPARTMENT 表と EMPLOYEE 表だけが検査保留状態になっている場合には、以下のコマンドを実行することができます。

```
SET INTEGRITY FOR DEPARTMENT, EMPLOYEE IMMEDIATE CHECKED
```

インポート・ユーティリティーは、参照保全によって次のような影響を受けます。

- オブジェクト表にそれ自体以外の従属表がある場合、 REPLACE および REPLACE CREATE 関数は使用できません。

  これらの関数を使用する場合は、まずその表が親表となっている外部キーをすべて除去してください。インポートが終了したら、ALTER TABLE ステートメントで外部キーを再作成してください。
- 自己参照制約が入っている表へのインポートが成功するかどうかは、行をインポートする順番によります。

表検査制約の定義

表の検査制約は、表検査制約が定義されている表の行ごとに適用される検索条件を指定するものです。表に表検査制約を作成するには、表の作成時または変更時に検査制約定義を表に対応付けます。この制約は、 INSERT または UPDATE ステートメントで表の中のデータを変更する時に自動的に活動化されます。表の検査制約は、DELETE または SELECT ステートメントに影響を及ぼしません。検査制約はタイプ付き表に関連付けることができます。

制約名は、同じ CREATE TABLE ステートメント内に指定されている他の制約と同じものにすることはできません。制約名を指定しない場合は、その制約の 18 文字の固有識別子がシステムによって生成されます。

表検査制約は、キーの固有性でカバーできないデータ保全規則、または参照保全制約を施行するために使用されます。場合によっては、定義域検査を実施するために、表の検査制約を使用することもできます。 CREATE TABLE ステートメントで発行された次の制約は、すべての活動の開始日付が同じ活動の終了日付より後になっていないことを確認します。

```
CREATE TABLE EMP_ACT
   (EMPNO        CHAR(6)      NOT NULL,
    PROJNO     CHAR(6)      NOT NULL,
    ACTNO      SMALLINT     NOT NULL,
    EMPTIME    DECIMAL(5,2),
    EMSTDATE   DATE,
    EMENDATE   DATE,
    CONSTRAINT ACTDATES CHECK(EMSTDATE <= EMENDATE) )
IN RESOURCE
```

この例では CREATE TABLE ステートメントを使って表検査制約を追加していますが、 ALTER TABLE ステートメントを使うこともできます。 198ページの『構造と内容の両方における表の修正』を参照してください。

新しい表に生成列を定義する

生成列は基礎表に定義されます。生成列に格納される値は、式を使って計算された値です。挿入操作や更新操作で指定された値ではありません。作成する表で、特定の式や述部を頻繁に使用することが分かっている場合、その表に 1 つまたは複数の生成列を追加することができます。生成列を使用すると、表データを照会する際のパフォーマンスを向上させることができます。

たとえば、パフォーマンスが重要な場合、式の評価の費用を高める恐れのある要因が 2 つあります。

1. 照会中に式の評価を何度も行わなければならないこと。
2. 計算が複雑であること。

照会のパフォーマンスを向上させるには、式の結果を入れるために、列をもう 1 つ定義することができます。そうすれば、同じ式を含んだ照会を発行するときに、生成列を直接に使用できます。あるいは、最適化プログラムの照会書き直しコンポーネントで、その式を生成列に置き換えることもできます。

生成列には、非固有の索引を作成することもできます。

照会時に複数の表のデータを結合する場合、生成列を追加すると、最適化プログラムがより良い結合の方針を選択できるかもしれません。

以下に、CREATE TABLE ステートメントを使って生成列を定義する方法の例を示します。

```
CREATE TABLE t1 (c1 INT,
                 c2 DOUBLE,
                 c3 DOUBLE GENERATED ALWAYS AS (c1 + c2)
                 c4 GENERATED ALWAYS AS
                    (CASE WHEN c1 > c2 THEN 1 ELSE NULL END))
```

この表を作成した後で、生成列を使用して索引を作成できます。たとえば、次のようにします。

```
CREATE INDEX i1 ON t1(c4)
```

生成列を照会に利用できます。たとえば、次のようにします。

```
SELECT COUNT(*) FROM t1 WHERE c1 > c2
```

これは、以下のように書き換えることができます。

```
SELECT COUNT(*) FROM t1 WHERE c4 IS NOT NULL
```

別の例を以下に示します。

```
SELECT c1 + c2 FROM t1 WHERE (c1 + c2) * c1 > 100
```

これは、以下のように書き換えることができます。

```
SELECT c3 FROM t1 WHERE c3 * c1 > 100
```

生成列は、照会のパフォーマンスを向上させるために使用します。したがって、恐らく、生成列を追加するのは、表を作成したり、表にデータを読み込んだ後になるでしょう。詳細については、126ページの『表の作成とデータの読み込み』を参照してください。

ユーザー定義一時表の作成

一時表を定義するには、DECLARE GLOBAL TEMPORARY TABLE ステートメントを使用します。このステートメントは 1 つのアプリケーションの中から使用されます。ユーザー定義一時表が保持されるのは、アプリケーションがデータベースから切断されるまでの間だけです。

この表の記述は、システム・カタログには現れません。したがって、この表を他のアプリケーションのために保持したり、他のアプリケーションと共用したりすることはできません。

この表を使用するアプリケーションが終了したりデータベースから切断されたりすると、表の中のデータはすべて削除され、表は暗黙的に除去されます。

一時表を定義する方法の例を以下に示します。

```
DECLARE GLOBAL TEMPORARY TABLE gbl_temp
   LIKE empltabl
   ON COMMIT DELETE ROWS
   NOT LOGGED
   IN usr_tbsp
```

このステートメントにより、 gbl_temp というユーザー一時表が作成されます。このユーザー一時表の列は、名前と記述が empltabl の列と完全に一致するように定義されています。暗黙定義には、列名、データ・タイプ、ヌル可能特性、および列のデフォルト値の属性だけが含まれます。他のすべての列属性 (固有制約、外部キー制約、トリガー、索引を含む) は、定義されていません。 COMMIT 操作を実行すると、表で WITH HOLD カーソルがオープンしていなければ、表の中のデータはすべて削除されます。ユーザー一時表に対する変更内容はログに記録されません。ユーザー一時表は、指定されたユーザー一時表スペースに置かれます。この表スペースがないと、この表の宣言は失敗します。

DECLARE GLOBAL TEMPORARY TABLE ステートメントの詳細については、 *SQL 解説書* を参照してください。

注: ユーザー定義一時表は、以下のものをサポートしません。

- LOB タイプの列 (または LOB に基づく特殊タイプ列)
- ユーザー定義タイプの列
- LONG VARCHAR 列
- DATALINK 列

識別列を新しい表に定義する

識別列 を使用すると、表に追加する個々の行に対し、それぞれ固有であることが保証された数値を DB2 が自動的に生成します。表に追加する個々の行を固有に識別する必要があることが分かっている場合、その表を作成する際に、識別列を追加できます。

いったん作成した後で、表の記述を更新して、識別列を組み込むことはできません。

CREATE TABLE ステートメントで AS IDENTITY 文節を使用すると、識別列を指定できます。

以下に、CREATE TABLE ステートメントを使って識別列を定義する方法の例を示します。

```
CREATE TABLE table (col1 INT,
                    col2 DOUBLE,
                    col3 INT NOT NULL GENERATED ALWAYS AS IDENTITY
                                      (START WITH 100, INCREMENT BY 5))
```

この例では、3 番目の列が識別列です。この列で使用される値を指定して、追加される個々の行を固有に識別することもできます。この例の場合、最初に入力される行については、値 『100』 が識別列に入れられます。この表に行が追加されるごとに、値は 5 ずつ増えます。

識別列を使用する他の例としては、注文番号、従業員番号、在庫番号、事例番号などがあります。 ALWAYS または BY DEFAULT を指定して、 DB2 によって識別列の値が生成されるようにすることができます。

GENERATED ALWAYS として定義された識別列は、固有になることが保証されます。使用される値は、常に DB2 によって生成されます。アプリケーションが明示的に値を指定することはできません。識別列を GENERATED BY DEFAULT として定義すると、アプリケーションが明示的に識別列の値を指定できます。アプリケーションが値を指定しないと、 DB2 が値を生成します。アプリケーションが値を制御するので、値が固有であることを DB2 が保証することはできません。 GENERATED BY DEFAULT 文節は、既存の表の内容をコピーする目的でデータ伝搬を行う場合や、表のアンロードや再ロードを行う場合に使用します。

注: 現在、識別列は、区分データベース環境ではサポートされていません。

明示的な識別列値が指定されている表に行を挿入する場合、内部的に生成される次の値は更新されないため、表の中の既存の値との間で競合が起こることがあります。重複値があると、エラー・メッセージが生成されます。

識別列を新しい表に定義することに関する追加情報は、 *SQL 解説書* を参照してください。

シーケンスの作成

シーケンス とは、値の自動生成を可能にするデータベース・オブジェクトです。シーケンスは、固有キー値を生成するタスクに最も適しています。アプリケーションはシーケンスを使用し、データベースの外部に固有カウンターを生成したことによって発生する可能性のある、並列性およびパフォーマンスの問題を回避することができます。

識別列属性とは異なり、シーケンスは特定の表列に関連付けられたり、固有の表列にバインドされることはなく、その表列からのみアクセス可能です。

以下の方法のいずれかでシーケンスが値を生成するよう、シーケンスを作成または変更することができます。

- バインドなしで単調増分または減分する
- ユーザー定義の制限まで単調増分または減分して終了する
- ユーザー定義の制限まで単調増分または減分し、先頭に戻って循環する

以下に、シーケンス・オブジェクト作成の例を示します。

```
CREATE SEQUENCE order_seq
   START WITH 1
   INCREMENT BY 1
   NOMAXVALUE
   NOCYCLE
   CACHE 24
```

この例で、シーケンスは order_seq です。 1 から始まり、上限なしで 1 ずつ増えていきます。上限が割り当てられていないため、先頭に戻って循環することはありません。CACHE パラメーターに関連する数値は、データベース・マネージャーが事前割り当てし、メモリーに保管するシーケンス値の最大数を指定します。

生成されるシーケンス番号のプロパティー

- 値は位取りがゼロの数値データ・タイプになります。このようなデータ・タイプは SMALLINT、BIGINT、INTEGER、および DECIMAL です。
- 連続値は、指定される整数増分によって異なります。デフォルト増分値は 1 です。
- カウンター値は回復可能です。カウンター値は、リカバリが要求されたときにログから再構成されます。
- パフォーマンスを上げるため、値をキャッシュに入れることができます。値を事前割り振りしてキャッシュに保管しておくと、シーケンスのために値を生成するとき、ロ

グへの非同期入出力が少なくなります。システム障害イベントが発生した場合、コミットされていないキャッシュ値はすべて使用されなくなり、失われたものと見なされます。 CACHE に指定された値は、失われる可能性のあるシーケンス値の最大数です。

シーケンスを含むデータベースを以前の状態にリカバリすると、いくつかのシーケンスで値が重複する場合があります。値の重複を回避するため、シーケンスを含むデータベースを以前の状態にリカバリしないでください。

シーケンスは単一ノード・データベースでのみサポートされています。

シーケンスで使用される式には、以下の 2 つがあります。

PREVVAL 式は、現行セッション内の直前のステートメントに指定されたシーケンスについて最後に生成された値を返します。

NEXTVAL 式は、指定されたシーケンスの次の値を返します。 NEXTVAL 式がシーケンスの名前を指定していれば、新しいシーケンス番号が生成されます。ただし、照会の中に同じシーケンス名を指定している NEXTVAL 式のインスタンスが複数ある場合、シーケンスのカウンターは結果の行ごとに 1 つずつ増えていきます。

同じシーケンス番号は、先頭の表の NEXTVAL 式およびその他の表の PREVVAL 式を使用してシーケンス番号を参照することによって、2 つの異なる表内の固有キー値として使用することができます。

たとえば、次のようにします。

```
INSERT INTO order (orderno, custno)
   VALUES (NEXTVAL FOR order_seq, 123456);
INSERT INTO line_item (orderno, partno, quantity)
   VALUES (PREVVAL FOR order_seq, 987654, 1)
```

NEXTVAL または PREVVAL 式は、以下の項目で使用することができます。

- INSERT ステートメント、VALUES 文節
- SELECT ステートメント、SELECT リスト
- SET 割り当てステートメント
- UPDATE ステートメント、SET 文節
- VALUES または VALUES INTO ステートメント

IDENTITY 列とシーケンスの比較

IDENTITY 列とシーケンスは類似していますが、異なる点もあります。それぞれの特性は、データベースおよびアプリケーションの設計時に利用することができます。

識別列の特性

- 識別列は、表の作成時にのみ、表の一部として定義することができます。表を作成した後、変更して識別列を追加することはできません。 (ただし、既存の識別列の特性を変更することはできます。)
- 識別列は、1 つの表の値を自動的に生成します。
- 識別列が GENERATED ALWAYS として定義されている場合、使用される値は常にデータベース・マネージャーによって生成されます。表の内容の変更中にアプリケーションで独自の値を指定することはできません。

シーケンス・オブジェクトの特性

- シーケンス・オブジェクトとは、どの表にも関連付けられていないデータベース・オブジェクトです。
- シーケンス・オブジェクトは、SQL ステートメントで使用可能な順次値を生成します。
- シーケンス・オブジェクトはどのアプリケーションでも使用できるため、指定されたシーケンス内の次の値、およびステートメントの実行前に生成された値の検索を制御するための 2 つの式があります。 PREVVAL 式は、現行セッション内の直前のステートメントに指定されたシーケンスについて最後に生成された値を返します。 NEXTVAL 式は、指定されたシーケンスの次の値を返します。これらの式を使用すると、複数の表内の複数の SQL ステートメントで同じ値を使用できるようになります。

以上がこれら 2 つの特性のすべてではありませんが、このような特性を考慮することによって、データベース設計やデータベースで使用するアプリケーションに応じてどちらを使用すればよいか判断するために役立ちます。

タイプ付き表の作成

CREATE TABLE ステートメントの変種を使用して、タイプ付き表を作成できます。タイプ付き表に関する必要なすべての情報については、 *アプリケーション開発の手引き* を参照してください。

タイプ付き表へのデータの読み込み

構造型を作成して、対応する表および副表を作成したら、タイプ付き表にデータを読み込むことができます。タイプ付き表に関する必要なすべての情報については、 *アプリケーション開発の手引き* を参照してください。

階層表

階層表とは、タイプ付き表階層の実装に関連付けられた表であり、階層のルート表と同時に作成されます。階層表に関する必要なすべての情報については、 *アプリケーション開発の手引き* を参照してください。

複数の表スペースへの表の作成

表データは、表の索引や、表に関連した長形式列データと同じ表スペースに格納することができます。また索引や長形式の列データを、それ以外の表データとは別の表スペースに入れることもできます。 CREATE TABLE ステートメントを実行する前に、すべての表スペースが存在していなければなりません。表の一部を分離することができるのは、 DMS 表スペースを使用している場合だけです。

コントロール・センターを使用して複数の表スペースに 1 つの表を作成するには、以下のようにします。

1. オブジェクト・ツリーを順に展開し、**「表 (Tables)」**フォルダーを表示します。
2. **「表 (Tables)」**フォルダーを右クリックして、ポップアップ・メニューから**「作成 (Create)」—>「ウィザードを使用して表を (Tables Using Wizard)」**を選択します。
3. 表名を入力して、**「次へ (Next)」**をクリックします。
4. 表の列を選択します。
5. **「表スペース (Table space)」**ページで、**「別の索引スペースを使用 (Use separate index space)」**と**「別の長形式スペースを使用 (Use separate long space)」**をクリックし、情報を指定して、**「終了 (Finish)」**をクリックします。

コマンド行を使用して複数の表スペースに 1 つの表を作成するには、以下のように入力します。

```
CREATE TABLE <name>
   (<column_name>  <data_type>  <null_attribute>)
    IN <table_space_name>
    INDEX IN <index_space_name>
    LONG  IN <long_space_name>
```

次の例では、別の表スペースに表の別の部分を格納するために、 EMP_PHOTO 表が作成される方法を示します。

```
CREATE TABLE EMP_PHOTO
   (EMPNO        CHAR(6)      NOT NULL,
    PHOTO_FORMAT VARCHAR(10)  NOT NULL,
    PICTURE      BLOB(100K) )
IN RESOURCE
INDEX IN RESOURCE_INDEXES
LONG  IN RESOURCE_PHOTO
```

この例では、EMP_PHOTO データが次のように作成されます。

- EMP_PHOTO 表について作成された索引が RESOURCES_INDEXES 表スペースに格納される。
- PICTURE 列のデータが RESOURCE_PHOTO 表スペースに格納される。

- EMPNO および PHOTO_FORMAT 列のデータが RESOURCE 表スペースに格納される。

1 つの表に複数の DMS 表スペースを使用することについては、 *管理の手引き: 計画* を参照して、さらに考慮してください。

CREATE TABLE ステートメントの詳細については、*SQL 解説書* を参照してください。

区分データベースへの表の作成

物理的に分割つまり区分化される表を作成する前に、以下のことを考慮する必要があります。

- 表スペースは、複数のデータベース区画にわたって展開することができます。展開する区画の数は、ノード・グループの中の区画の数によって決まります。
- 表は、同じ表スペースに置かれるか、または、最初の表スペースに加えて、同じノード・グループに関連する別の表スペースの中に置かれることによって、併置を行うことができます。詳細については、*管理の手引き: 計画* を参照してください。

区分データベース環境で表を作成する場合、*区分化キー* という、追加のオプションが存在します。区分化キーは、表の定義の一部であるキーです。このキーは、各データ行が保管される区画を判別します。

*後から変更することはできない*ため、適切な区分化キーを選択することが重要です。さらに、何らかの固有索引を (したがって、固有キーまたは基本キーも)、区分化キーのスーパーセットとして定義しなければなりません。つまり、区分化キーが定義された場合、固有キーおよび基本キーは、区分化キーと同じ列をすべて含まなければなりません (固有キーと基本キーには、それ以上の列が含まれる場合があります)。

区分化キーを明示して指定しない場合、以下のデフォルトが使用されます。*デフォルトの区分化キーが適切であることを確認*してください。

- 基本キーが CREATE TABLE ステートメントに指定された場合、基本キーの最初の列が区分化キーとして使用されます。
- 基本キーがない場合、ロング・フィールドでない最初の列が使用されます。
- デフォルトの区分化キーの要件を満たす列がない場合、表は区分化キーなしで作成されます (これは、単一区画のノード・グループでのみ許されます)。

以下はその例です。

```
CREATE TABLE MIXREC (MIX_CNTL INTEGER NOT NULL,
                     MIX_DESC CHAR(20) NOT NULL,
                     MIX_CHR  CHAR(9) NOT NULL,
                     MIX_INT INTEGER NOT NULL,
                     MIX_INTS SMALLINT NOT NULL,
                     MIX_DEC DECIMAL NOT NULL,
                     MIX_FLT FLOAT NOT NULL,
```

```
MIX_DATE DATE NOT NULL,
MIX_TIME TIME NOT NULL,
MIX_TMSTMP TIMESTAMP NOT NULL)
IN MIXTS12
PARTITIONING KEY (MIX_INT) USING HASHING
```

上記の例で、表スペースは MIXTS12 であり、区分化キーは MIX_INT です。区分化キーが明示して指定されない場合、区分化キーは MIX_CNTL になります。 (基本キーが指定されず、区分化キーが定義されない場合、区分化キーは、リスト内の最初のロング列以外の列になります。)

1 つの表の 1 つの行、およびその行に関するすべての情報は、常に同じデータベース区画上に常駐します。

1 つの表の 1 区画についてのサイズの限界は、 64 GB または使用可能ディスク・スペースのうち、いずれか小さいほうになります。 (表スペースに 4 KB のページ・サイズを想定しています。) 表のサイズは、データベース区画の数の 64 GB (または使用可能ディスク・スペース) 倍までの大きさにすることができます。表スペースのページ・サイズが 8 KB の場合、表のサイズは、データベース区画の数の 128 GB (または使用可能ディスク・スペース) 倍までの大きさにすることができます。表スペースのページ・サイズが 16 KB の場合、表のサイズは、データベース区画の数の 256 GB (または使用可能ディスク・スペース) 倍までの大きさにすることができます。表スペースのページ・サイズが 32 KB の場合、表のサイズは、データベース区画の数の 512 GB (または使用可能ディスク・スペース) 倍までの大きさにすることができます。

トリガーの作成

トリガーは、指定した基礎表またはタイプ付き表に対する INSERT、UPDATE、DELETE 文節と一緒に実行される一連のアクション、またはそれらの文節によってトリガー起動される一連のアクションを定義するものです。トリガーは、たとえば次のような目的で使います。

- 入力データの妥当性検査
- 新しく挿入された行の値を生成する
- 相互参照のために他の表から読み込む
- 監査証跡のために他の表に書き込む

トリガーにニックネームを使用することはできません。

トリガーを使えば、一般的な保全規則や業務規則をサポートできます。たとえば、トリガーによって、注文に応じる前に顧客のクレジット限度を調べたり、要約データ表を更新したりできます。

トリガーを使うことの利点は、次のとおりです。

- アプリケーション開発がより速くなる: トリガーはデータベースの中に保管されるため、各アプリケーションの中にアクションをコーディングする必要がありません。
- 保守が簡単: 一度トリガーを定義すると、トリガーを作成した表にアクセスするたびに、そのトリガーが自動的に呼び出されます。
- 業務規則がグローバルに適用される: 業務方針が変わった場合、各アプリケーション・プログラムを変更しなくても、トリガーを変更するだけで済みます。

コントロール・センターを使用してトリガーを作成するには、以下のようにします。

1. オブジェクト・ツリーを順に展開し、**「トリガー (Triggers)」**フォルダーを表示します。
2. **「トリガー (Triggers)」**フォルダーを右クリックして、ポップアップ・メニューから**「作成 (Create)」**を選択します。
3. トリガーに関する情報を指定します。
4. トリガーによって呼び出すアクションを指定し、**「Ok」**をクリックします。

コマンド行を使用してトリガーを作成するには、以下のように入力します。

```
CREATE TRIGGER <name>
   <action> ON <table_name>
   <operation>
   <triggered_action>
```

次の SQL ステートメントは、新人が採用されるたびに従業員の数を増やすトリガーが作成されます。これによって、EMPLOYEE 表に行が追加されるたびに、COMPANY_STATS 表の従業員数 (NBEMP) 列に 1 が加算されます。

```
CREATE TRIGGER NEW_HIRED
   AFTER INSERT ON EMPLOYEE
   FOR EACH ROW MODE DB2SQL
   UPDATE COMPANY_STATS SET NBEMP = NBEMP+1;
```

トリガー本体には、INSERT、探索 UPDATE、探索 DELETE、全選択、SET 変位変数、および SIGNAL SQLSTATE のうちの 1 つ以上の SQL ステートメントを含めることができます。トリガーは、それが参照する INSERT、 UPDATE、または DELETE ステートメントの前または後に起動できます。 CREATE TRIGGER ステートメントの構文に関する完全な情報については、 *SQL 解説書* を参照してください。トリガーの作成および使用についての詳細は、*アプリケーション開発の手引き* を参照してください。

注: BEFORE トリガーの場合は、識別列以外の生成列の列名を、トリガー・アクションによって指定することはできません。したがって、生成される識別値は BEFORE トリガーに認識されます。

アトミック・トリガーを作成する際には、ステートメント終了文字に注意する必要があります。データベース・マネージャーは、デフォルトではステートメント終了文字を

『;』 と見なします。 『;』 以外の文字を使用するには、スクリプト内でステートメント終了文字を手動で編集して、アトミック・トリガーを作成しなければなりません。たとえば、『;』 を 『#』 などの別の特殊文字で置き換えます。

次に、以下のいずれかを行う必要があります。

- コマンド・センターで選択したスクリプト・タブを使用して「ツール (tools)」—> 「ツール設定 (tools settings)」メニューから区切り文字を変更し、スクリプトを実行します。または、
- コマンド行プロセッサーから、以下を使用します。

```
db2 -td <delimiter> -vf <script>
```

ここで、delimiter は代わりのステートメント終了文字で、 <script> はその中の新しい区切り文字を使って変更したスクリプトです。

トリガーの従属関係

他のオブジェクトに対するトリガーの従属関係は、すべて SYSCAT.TRIGDEP カタログに記録されます。トリガーはさまざまなオブジェクトに従属する可能性があります。これらのオブジェクトと従属のトリガーについては、 *SQL 解説書* の DROP ステートメントの説明で詳細に説明されています。

これらのオブジェクトのどれかを除去すると、トリガーは機能しなくなりますが、その定義はカタログ内に残ります。そのトリガーを再び有効にするには、カタログからその定義を取り出し、新しい CREATE TRIGGER ステートメントを実行依頼してください。

トリガーを除去すると、その定義は SYSCAT.TRIGGERS カタログ視点から削除され、その従属関係もすべて SYSCAT.TRIGDEP カタログ視点から削除されます。トリガーに UPDATE、INSERT、または DELETE 従属性のあるパッケージは、すべて無効になります。

従属オブジェクトが視点で、それが作動不能になっている場合は、トリガーにも作動不能のマークが付けられます。作動不能のマークが付けられたトリガーに従属するパッケージがあれば、すべて無効にされます。 (詳細については、225ページの『オブジェクトを変更する場合のステートメントの従属関係』を参照してください。)

ユーザー定義関数 (UDF) または方式の作成

ユーザー定義関数 (UDF) は、SQL の組み込み関数によるサポートを拡張および追加するものであり、組み込み関数が使用できるところであればどこでも使用することができます。 UDF を、以下のいずれかとして作成することができます。

- 外部関数。プログラム言語で作成されたもの。
- ソース関数。それを実現したものは、既存の他の関数から継承されます。

以下の 3 つのタイプの UDF があります。

スカラー

呼び出されるたびに、単一値の応答を戻します。たとえば、組み込み関数 SUBSTR() はスカラー関数です。スカラー UDF は、外部関数またはソース関数のいずれも可能です。

列

1 組みの類似値 (1 つの列) から、単一値の応答を戻します。 DB2 では、総計関数と呼ばれる場合もあります。列関数の 1 つの例は、組み込み関数 AVG() です。外部列 UDF を DB2 に定義することはできませんが、組み込み列関数の 1 つのソース関数となる列 UDF は定義することができます。これは、特殊タイプの場合に便利です。

たとえば、基本タイプ INTEGER で定義された特殊タイプ SHOESIZE がある場合、組み込み関数 AVG(INTEGER) のソース関数となる UDF AVG(SHOESIZE) を定義することができ、それは列関数になります。

表

それを参照する SQL ステートメントに対して 1 つの表を戻します。表関数は、SELECT ステートメントの FROM 文節の中でのみ参照できます。このような関数は、DB2 データでないデータのため、またはそのようなデータを DB2 表に変換するために、 SQL 言語処理能力を利用するために使用できます。

たとえば、表関数は、ファイルを取り出してそれを表に変換するか、ワールド・ワイド・ウェブ (WWW) からのサンプル・データを表の形にするか、あるいは Lotus Notes データベースにアクセスして、メール・メッセージの日付、送信者、テキストなどの情報を戻します。この情報は、データベース内の他の表と結合することができます。

表関数は、外部関数のみが可能です。ソース関数にはできません。

既存の UDF についての情報は、 SYSCAT.FUNCTIONS および SYSCAT.FUNCPARMS カタログ視点の中に記録されます。システム・カタログには UDF の実行可能コードは含まれません。 (このため、バックアップおよび回復の計画を作成する場合には、 UDF 実行可能コードをどのように管理するかを考慮する必要があります。)

UDF のパフォーマンスに関する統計は、 SQL ステートメントをコンパイルするときに重要です。システム・カタログ内の UDF 統計の更新方法については、 *管理の手引き: パフォーマンス* の『ユーザー定義関数の統計の更新』を参照してください。

特定のアプリケーションに合わせて UDF を作成するための CREATE FUNCTION ステートメントの使用についての詳細は、 *アプリケーション開発の手引き* を参照してください。 UDF の構文の詳細は、*SQL 解説書* を参照してください。

関数マッピングの作成

連合データベースでは、 1 つまたは複数のデータ・ソースを持つローカル関数またはローカル関数テンプレート (『関数テンプレートの作成』 に説明) をマップする必要があるときに、関数マッピングを作成します。多くのデータ・ソース関数で、デフォルトの関数マッピングが提供されています。

以下の場合に、関数マッピングが便利です。

- 新規の組み込み関数がデータ・ソースで使用可能になるとき。
- データ・ソースでのユーザー定義関数をローカル関数にマップする必要があるとき。
- アプリケーションで、デフォルト・マッピングで提供されるものとは異なるデフォルトの動作が求められるとき。

CREATE FUNCTION MAPPING ステートメントで定義された関数マッピングは、統合データ・ソースに保管されます。

関数 (関数テンプレート) には、データ・ソース関数と同数の入力パラメーターを指定する必要があります。さらに、統合される側の入力パラメーターのデータ・タイプと、データ・ソース側の入力パラメーターのデータ・タイプには、互換性がなければなりません。これらの要件は戻り値にも適用されます。

CREATE FUNCTION MAPPING ステートメントを、関数マッピングを作成するために使用します。たとえば、サーバー ORACLE1 で Oracle の AVGNEW 関数と DB2 の同等関数との関数マッピングを作成するには、次のようにします。

```
CREATE FUNCTION MAPPING ORAVGNEW FOR SYSIBM.AVG(INT) SERVER ORACLE1
OPTIONS (REMOTE_NAME 'AVGNEW')
```

このステートメントを使用するには、連合データベースで SYSADM または DBADM 権限のいずれかを持っている必要があります。関数マッピングの属性は SYSCAT.FUNCMAPPINGS に保管されます。

統合サーバーは入力ホスト変数をバインドしたり、LOB、 LONG VARCHAR/VARGRAPHIC、 DATALINK、特殊および構造型タイプの結果を検索したりしません。入力パラメーターまたは戻り値に上記のいずれかのタイプが含まれていると、関数マッピングは作成できません。

関数マッピングの使用および作成についての詳細は、 *アプリケーション開発の手引き* を参照してください。 CREATE FUNCTION MAPPING 構文についての詳細は、 *SQL 解説書* を参照してください。

関数テンプレートの作成

連合システムでは、関数テンプレートによって関数マッピング用の『アンカー』が提供されます。アンカーは、該当する DB2 関数が統合サーバーに存在しない場合にデー

タ・ソース関数のマッピングを可能にするために使用します。関数マッピングでは、関数テンプレートが存在するか、または DB2 に同様の関数が備わっていなければなりません。

テンプレートとは関数シェル、すなわち、名前、入力パラメーター、および戻り値にほかなりません。関数にはローカルの実行可能コードはありません。

関数用のローカル実行可能コードがないため、データ・ソースに使用可能な関数があっても、関数テンプレートの呼び出しが失敗するという可能性があります。たとえば、次のような照会を考えてみます。

```
SELECT myfunc(C1)
   FROM nick1
   WHERE C2 < 'A'
```

DB2 および nick1 が参照するオブジェクトの入ったデータ・ソースの照合順序が同じではない場合、関数がデータ・ソースにある間に比較を行う必要があるため、照会は失敗します。照合順序が同じであれば、比較操作は、 myfunc が参照する基礎関数を持つデータ・ソースで行うことができます。

関数 (関数テンプレート) には、データ・ソース関数と同数の入力パラメーターを指定する必要があります。統合される側の入力パラメーターのデータ・タイプと、データ・ソース側の入力パラメーターのデータ・タイプには、互換性がなければなりません。これらの要件は戻り値にも適用されます。

関数テンプレートは、 CREATE FUNCTION ステートメントに AS TEMPLATE キーワードを指定して作成します。テンプレートを作成したら、 CREATE FUNCTION MAPPING ステートメントを使ってテンプレートをデータベースにマップします。

たとえば、サーバー S1 に関数 MYS1FUNC 用の関数テンプレートと関数マッピングを作成するには、次のようにします。

```
CREATE FUNCTION MYFUNC(INT) RETURNS INT AS TEMPLATE

CREATE FUNCTION MAPPING S1_MYFUNC FOR MYFUNC(INT) SERVER S1 OPTIONS
(REMOTE_NAME 'MYS1FUNC')
```

関数テンプレートの使用および作成についての詳細は、*アプリケーション開発の手引き* を参照してください。 CREATE FUNCTION 構文についての詳細は、*SQL 解説書* を参照してください。

ユーザー定義タイプ (UDT) の作成

ユーザー定義タイプ (UDT) は、データベース内でユーザーによって作成される名前付きデータ・タイプです。 UDT は、それぞれがタイプを持つ名前付き属性の順序がある組み込みデータ・タイプまたは構造型と、共通表示を共用する特殊タイプとなることができます。構造型はタイプ階層を定義しますが、別の構造型 (スーパータイプと呼ばれる) のサブタイプとなることができます。

UDT は、強い型指定をサポートしています。これは、UDT が他のタイプと同じ表示を共用していても、指定された UDT の値が同じタイプ階層の同じ UDT (複数も可) の値とのみ互換性のあるものと見なされるという意味です。

SYSCAT.DATATYPES カタログ視点を使用して、データベース用に定義された UDT を見ることができます。このカタログ視点では、データベースの作成時にデータベース・マネージャーが定義したデータ・タイプも示しています。すべてのデータ・タイプのリストについては、*SQL 解説書* を参照してください。

UDT は、ほとんどのシステム提供関数 (組み込み関数) には引き数として使用することはできません。これらの演算子や他の演算子を使用するには、ユーザー定義関数を作成する必要があります。

以下の場合にのみ、UDT を除去することができます。

- 既存の表の列定義の中で使用されていない場合。
- 既存のタイプ付き表またはタイプ付き視点のタイプとして使用されていない場合。
- 除去できない UDF 関数の中で使用されていない場合。ある UDF に視点、トリガー、表検査制約、または別の UDF が従属している場合、その UDF は除去できません。

UDT を除去すると、それに従属する関数があれば、それも除去されます。

ユーザー定義特殊タイプの作成

ユーザー定義特殊タイプは、整数、10 進数、または文字タイプなど、既存のタイプから導出されたタイプです。特殊タイプは、CREATE DISTINCT TYPE ステートメントを使用して作成することができます。

次の SQL ステートメントは、特殊タイプ `t_educ` を smallint として作成するものです。

```
CREATE DISTINCT TYPE T_EDUC AS SMALLINT WITH COMPARISONS
```

例にあるように、 WITH COMPARISONS 文節が CREATE DISTINCT TYPE ステートメントで指定された場合には、同じ特殊タイプのインスタンスを相互に比較することが

できます。 WITH COMPARISONS 文節は、ソース・データがラージ・オブジェクト、DATALINK、LONG VARCHAR、または LONG VARGRAPHIC タイプである場合は、指定できません。

特殊タイプのインスタンスは、ソース・タイプに定義された関数の引き数または操作のオペランドとして使用することはできません。同様に、ソース・タイプは、特殊タイプを使用するよう定義された引き数またはオペランドの中で使用することはできません。

特殊タイプを作成したなら、それを使用して、 CREATE TABLE ステートメントに列を定義することができます。

```
CREATE TABLE EMPLOYEE
   (EMPNO        CHAR(6)      NOT NULL,
    FIRSTNME   VARCHAR(12)  NOT NULL,
    LASTNAME   VARCHAR(15)  NOT NULL,
    WORKDEPT   CHAR(3),
    PHONENO    CHAR(4),
    PHOTO      BLOB(10M)    NOT NULL,
    EDLEVEL    T_EDUC)
IN RESOURCE
```

特殊タイプを作成すると、特殊タイプとソース・タイプ間のキャストもサポートされるようになります。したがって、タイプ T_EDUC の値を SMALLINT 値にキャストすることができ、 SMALLINT 値を T_EDUC 値にキャストすることができます。

CREATE DISTINCT TYPE ステートメントの構文に関する完全な情報については、*SQL 解説書* を参照してください。特殊タイプの作成および使用についての詳細は、*アプリケーション開発の手引き* を参照してください。

変換を利用して、UDT を基本データ・タイプに、基本データ・タイプを UDT に変換することができます。変換関数は CREATE TRANSFORM ステートメントによって作成します。

変換のサポートは、CREATE METHOD ステートメントおよび CREATE FUNCTION ステートメントの拡張形式の中にも見られます。このサポートについての詳細は、*SQL 解説書* を参照してください。

ユーザー定義構造型の作成

構造型は、1 つまたは複数の属性が入ったユーザー定義タイプで、個々のタイプは独自の名前とデータ・タイプを持っています。構造型は表のタイプの役割を果たし、その表の各列は、構造型の属性の 1 つから名前とデータ・タイプを取得します。構造型に関する必要なすべての情報については、 *アプリケーション開発の手引き* を参照してください。

タイプ・マッピングの作成

連合システムでは、タイプ・マッピングにより、データ・ソース内の特定データ・タイプと DB2 特殊データ・タイプの視点をマップできます。タイプ・マッピングは、1 つのデータ・ソース、またはデータ・ソースのある範囲 (タイプ、バージョン) に適用できます。

組み込みデータ・ソース・タイプおよび組み込み DB2 タイプには、デフォルトのデータ・タイプ・マッピングが用意されています。新規データ・タイプ・マッピング (ユーザーが作成するもの) は、 SYSCAT.TYPEMAPPINGS 視点にリストされます。

タイプ・マッピングは、CREATE TYPE MAPPING ステートメントを使って作成します。このステートメントを使用するには、連合データベースで SYSADM または DBADM 権限のいずれかを持っている必要があります。

タイプ・マッピング・ステートメントの例は、次のとおりです。

```
CREATE TYPE MAPPING MY_ORACLE_DEC FROM SYSIBM.DECIMAL(10,2)
TO SERVER ORACLE1 TYPE NUMBER([10..38],2)
```

LOB、LONG VARCHAR/VARGRAPHIC、DATALINK、構造型、または特殊タイプ用のタイプ・マッピングを作成することはできません。

タイプ・マッピングの使用および作成についての詳細は、 *アプリケーション開発の手引き* を参照してください。 CREATE TYPE MAPPING 構文についての詳細は、*SQL 解説書* を参照してください。

視点の作成

視点は 1 つまたは複数の基礎表、ニックネーム、または視点から導出されるもので、データの検索時には基礎表と交換可能なものとして使用されます。視点の中に表示されるデータに変更が加えられると、表そのもののデータも変更されます。

視点を作成することによって、重要データについてはアクセスを限定し、その他のデータについては一般にアクセスを許可することができます。

視点定義の SELECT リストに、基礎表の識別列の名前が直接または間接的に含まれている場合、その視点に挿入を実行するときには、 INSERT ステートメントが基礎表の識別列を直接参照している場合と同じ規則が適用されます。 INSERT ステートメントの詳細については、*SQL 解説書* を参照してください。

上記のような視点の使用に加えて、次のような目的で視点を使用することができます。

- アプリケーション・プログラムに影響を及ぼさずに表を変更する。このことは、基礎表に基づいて視点を作成することによって生じ得ます。基礎表を使用するアプリケー

ションは、新しい視点の作成によって影響を受けることはありません。新しいアプリケーションは、基礎表を使用するアプリケーションではなく、異なる目的のために作成された視点を使用できます。

- 列の中の値を合計する、最大値を選択する、または、それらの値を平均する。
- 1 つまたは複数のデータ・ソースの中の情報へのアクセスを提供する。 CREATE VIEW ステートメント内のニックネームを参照し、複数ロケーション / グローバル視点 (視点は異なるシステム上にある複数データ・ソース内の情報を結合できる) を作成することができます。

  標準の CREATE VIEW 構文を使ってニックネームを参照する視点を作成すると、基礎オブジェクトへのアクセスに視点作成者認証 ID ではなく視点ユーザーの認証 ID が使用されるという事実に注意を促す警告が表示されます。この警告を表示しないようにするには、FEDERATED キーワードを使用します。

視点を作成する代わりにネストした表式または共通表式を使うこともできます。その場合、カタログ参照が少なくなり、パフォーマンスは高くなります。共通表式についての詳細は、*SQL 解説書* を参照してください。

コントロール・センターを使用して視点を作成するには、以下のようにします。

1. オブジェクト・ツリーを順に展開し、**「視点 (Views)」**フォルダーを表示します。
2. **「視点 (Views)」**フォルダーを右クリックして、ポップアップ・メニューから**「作成 (Create)」**を選択します。
3. 情報をすべて入力し、**「Ok」**をクリックします。

コマンド行を使用して視点を作成するには、以下のように入力します。

```
CREATE VIEW <name> (<column>, <column>, <column>)
   SELECT <column_names> FROM <table_name>
   WITH CHECK OPTION
```

たとえば、EMPLOYEE 表には給与に関する情報が入っているかもしれませんが、そうした情報はだれもが利用できるようにすべきではありません。しかし、従業員の電話番号はだれでもアクセスできるようにすべきです。このような場合は、LASTNAME 列と PHONENO 列だけからなる視点を作成できます。視点へのアクセスは PUBLIC に与えるようにし、EMPLOYEE 表全体へのアクセスは、給与情報を見る権限のある人だけに制限するようにします。*読み取り専用* 視点についての詳細は、 *SQL 解説書* を参照してください。

視点を使うと、表データのうちアプリケーション・プログラムで利用できるサブセットを作成し、挿入または更新するデータの妥当性検査を実行することができます。視点の列名は、元の表の対応する列名と違うものにすることができます。

視点を使うと、プログラムやエンド・ユーザー照会で表データを見る方法の点で柔軟性が高くなります。

以下の SQL ステートメントは、EMPLOYEE 表に 1 つの視点を作成します。この視点は部門 A00 のすべての従業員を、従業員番号と電話番号の情報とともにリストします。

```
CREATE VIEW EMP_VIEW (DA00NAME, DA00NUM, PHONENO)
   AS SELECT LASTNAME, EMPNO, PHONENO FROM EMPLOYEE
   WHERE WORKDEPT = 'A00'
   WITH CHECK OPTION
```

このステートメントの最初の行では、視点の名前を指定し、その列を定義しています。EMP_VIEW という名前は、 SYSCAT.TABLES の中のそのスキーマ内で固有なものでなければなりません。視点名は表名の 1 つとして表示されますが、データは含まれていません。この視点の列は DA00NAME、 DA00NUM、および PHONENO の 3 つであり、それぞれ、 EMPLOYEE 表の LASTNAME、EMPNO、および PHONENO の列に対応するものです。最初の行に指定する列名は、SELECT ステートメントの選択リストに 1 対 1 に対応します。列名を指定しないなら、視点の列名には、 SELECT ステートメントの結果表の列と同じ名前が使われます。

第 2 行は、データベースから選択する値について記述する SELECT ステートメントです。これには、ALL、DISTINCT、FROM、WHERE、GROUP BY、および HAVING という文節を含めることができます。視点のための列を選択する元のデータ・オブジェクトの名前を、 FROM 文節の後に指定します。

WITH CHECK OPTION 文節は、視点に対して更新する行または挿入する行を視点定義に照らして検査し、定義に従っていない場合には拒否することを指定するものです。これによってデータ保全性は向上しますが、余分な処理が必要になります。この文節を省略すると、挿入する行または更新する行が視点定義に照らして検査されることはありません。

次の SQL ステートメントは、EMPLOYEE 表に基づく同じ視点を、 SELECT AS 文節を使って作成するものです。

```
CREATE VIEW EMP_VIEW
   SELECT LASTNAME AS DA00NAME,
          EMPNO AS DA00NUM,
          PHONENO
   FROM EMPLOYEE
   WHERE WORKDEPT = 'A00'
   WITH CHECK OPTION
```

定義の中で UDF を使う視点を作成することもできます。ただし、この視点を最新の関数が含まれるように更新するためには、その視点を除去してから再作成しなければなりません。視点が UDF に依存している場合、その関数は除去できません。

次の SQL ステートメントは、定義内に関数の含まれる視点を作成するものです。

```
CREATE VIEW EMPLOYEE_PENSION (NAME, PENSION)
  AS SELECT NAME, PENSION(HIREDATE,BIRTHDATE,SALARY,BONUS)
  FROM EMPLOYEE
```

UDF 関数の PENSION によって、従業員が現時点で受け取ることのできる年金が計算されます。その計算には、HIREDATE、BIRTHDATE、SALARY、および BONUS が使われます。

タイプ付き視点の作成

CREATE VIEW ステートメントを使用してタイプ付き視点を作成できます。タイプ付き視点に関する必要なすべての情報については、 *アプリケーション開発の手引き* を参照してください。

要約表の作成

要約表 は、その定義が照会の結果に基づいている表です。したがって、通常、要約表には、要約表の定義の基礎となる表 (複数も可) に存在するデータに基づいた、事前に計算された結果が含まれています。 SQL コンパイラーが、基礎表に対するよりも、要約表に対する照会のほうが効果的に実行できると判断する場合は、照会は要約表に対して実行され、基礎表に対する照会よりも速く結果を得られます。

区分データベース環境のすべてのノードにわたって表を複製するために、複製オプションを使用して要約表を作成できます。これらは、「複製要約表」と呼ばれています。この種の表の概要については、 *管理の手引き: 計画* を参照してください。

注: 要約表は静的 SQL やニックネームでは使用されません。

要約表または複製要約表の分離レベルが照会の分離レベルと等しいか、同レベルを上回る場合、照会の最適化には普通、要約表または複製要約表が使用されます。たとえば、照会がカーソル固定 (CS) 分離レベル下で実行される場合、最適化には CS または上位の分離レベルで定義された要約表と複製要約表だけが使用されます。

要約表を作成するには、AS *fullselect* 文節と、 IMMEDIATE または REFRESH DEFERRED オプションを指定した CREATE SUMMARY TABLE ステートメントを使用します。

要約表の列の名前は、一意的に識別することができます。列名のリストには、全選択の結果表の列と同じ数だけ名前がなければなりません。全選択の結果表に重複列があったり、無名列があったりする場合には、列名のリストを指定することが必要です。無名列は、選択リストの AS 文節に名前が指定されていない定数、関数、式、またはセット演算のために生じます。列名のリストを指定しないと、表の列は、全選択の結果セットの列の名前を受け継ぎます。

要約表の作成時には、基礎表の変更時に要約表を自動的に最新表示するかどうか、または REFRESH TABLE ステートメントを使って要約表を最新表示するかどうかを指定するオプションがあります。基礎表 (複数も可) に変更が加えられたときに要約表を自動的に最新表示するには、 REFRESH IMMEDIATE キーワードを指定します。以下の場合には、即時最新表示が便利です。

- 基礎表への実行時に、完了に時間を要する照会を行う
- 基礎表の変更頻度が低い
- 最新表示に多くの費用がかからない

このような状況の場合、要約表は事前計算した結果を提供できます。要約表の最新表示を据え置きたければ、REFRESH DEFERRED キーワードを指定します。 REFRESH DEFERRED を使用して指定された要約表は、基礎表に対する変更を反映**しません**。要約表は使用するべきですが、必ずそうしなければならないという意味ではありません。たとえば、DSS を実行する場合、既存のデータを入れるために要約表を使用します。

要約表が次の条件を満たす場合は、照会の代わりに REFRESH DEFERRED を使用して定義された要約表が使用されます。

- 次の場合以外は、即時最新表示要約表の全選択の制限に準拠している。
  - COUNT(*) または COUNT_BIG(*) を組み込むために SELECT リストが必要ではない。
  - SELECT リストには、MAX および MIN 列関数を組み込むことができる。
  - HAVING 文節を指定できる。

SQL 特殊レジスター CURRENT REFRESH AGE SQL は、 ANY または 99999999999999 という値に設定されます。この 9 の連続した値は、この特殊レジスターで指定できる最大値であり、 DECIMAL(20,6) というデータ・タイプのタイム・スタンプ期間値です。

注: REFRESH DEFERRED を使って定義した要約表は、静的 SQL を最適化するためには使用されません。

CURRENT REFRESH AGE 特殊レジスターを使用して、据え置き最新表示の要約表が、最新表示されなければならなくなる前に動的照会に使用できる時間を指定します。CURRENT REFRESH AGE 特殊レジスターの値を設定するには、 SET CURRENT REFRESH AGE ステートメントを使用します。 CURRENT REFRESH AGE 特殊レジスター、および SET CURRENT REFRESH AGE ステートメントについて詳しくは、*SQL 解説書* を参照してください。

REFRESH IMMEDIATE を使って定義した要約表は静的照会にも動的照会にも適用可能で、 CURRENT REFRESH AGE 特殊レジスターを使用する必要はありません。

注: CURRENT REFRESH AGE 特殊レジスターをゼロ以外の値に設定するに際しては、注意が必要です。照会の処理を最適化するために、基礎表の値を表さない要約表を

使用できるようにすると、照会の結果は基礎表のデータを正確には表しません。基礎表のデータが変更されていないことを知っている、またはデータについての知識に基づいて結果のエラーの程度を受け入れるつもりであるなら、これは無理のない設定といえるでしょう。

ソース・データに影響する活動については、そのうち要約表に正確なデータが含まることはなくなります。 REFRESH TABLE ステートメントを使用する必要があるでしょう。詳しくは、*SQL 解説書* を参照してください。

有効な *fullselect* に基づく新しい基礎表を作成したい場合は、表の作成時に DEFINITION ONLY キーワードを指定します。表作成操作が完了すると、新しい表は基礎表としてではなく要約表として扱われます。たとえば、次のようにして LOAD と SET INTEGRITY で使用される例外表を作成できます。

```
CREATE TABLE XT AS
   (SELECT T.*, CURRENT TIMESTAMP AS TIMESTAMP,CLOB(",32K)
   AS MSG FROM T) DEFINITION ONLY
```

以下は、要約表に関連する重要な制約事項です。

1. 要約表は更新できない。
2. 基礎表に要約表がある場合、基礎表の列の長さは更新できない。
3. 要約表にデータをインポートできない。
4. 要約表に固有索引を作成できない。
5. 1 つまたは複数のニックネームを参照する結果に基づく要約表は作成できない。

要約表の制約事項に関する完全な説明については、 *SQL 解説書* を参照してください。

別名の作成

別名は、表、ニックネーム、または視点を間接的に参照して、 SQL ステートメントを表や視点の修飾名とは無関係なものにするためのものです。表名や視点名を変更しても、別名の定義を変えるだけで済みます。別名は他の別名に対して作成することもできます。別名は、視点やトリガーの定義、また SQL ステートメントの中で使用できます。ただし、既存の表名や視点名を参照する表検査制約では使用できません。

別名は既存の表名が使用できる所ならどこにでも使用できます。また、別名連鎖に循環参照または反復参照がない限り他の別名を参照することもできます。

既存の表、視点、別名と同じ別名を作成することはできません。また、別名は同じデータベース内の表しか参照できません。 CREATE TABLE ステートメントまたは CREATE VIEW ステートメントでは、同じスキーマ内の別名と同じ表名や視点名は使用できません。

別名の作成には特別な権限は必要ありません。ただし、自分の現在の許可 ID が所有するスキーマ以外のスキーマに別名を作成する場合は、 DBADM 権限が必要です。

別名は、定義時に存在していない表、視点、または別名に対しても定義できます。しかし、別名の含まれる SQL ステートメントのコンパイル時には、存在していなければなりません。

別名または別名が参照するオブジェクトが除去されると、その別名に依存するすべてのパッケージは無効のマークが付けられ、その別名に依存するすべての視点およびトリガーは作動不能のマークが付けられます。

コントロール・センターを使用して別名を作成するには、以下のようにします。

1. オブジェクト・ツリーを順に展開し、**「別名 (Aliases)」**フォルダーを表示します。
2. **「別名 (Aliases)」**フォルダーを右クリックして、ポップアップ・メニューから**「作成 (Create)」**を選択します。
3. 情報をすべて入力し、**「Ok」**をクリックします。

コマンド行を使用して別名を作成するには、以下のように入力します。

```
CREATE ALIAS <alias_name> FOR <table_name>
```

別名は、ステートメントのコンパイル時に表名や視点名に置き換えられます。別名または別名連鎖が表名や視点名に置換できないと、エラーになります。たとえば、WORKERS を EMPLOYEE の別名にした場合、コンパイル時には、

```
SELECT * FROM WORKERS
```

は、実際には次のものになります。

```
SELECT * FROM EMPLOYEE
```

次の SQL ステートメントは、EMPLOYEE 表に WORKERS という別名を作成するものです。

```
CREATE ALIAS WORKERS FOR EMPLOYEE
```

注: DB2 (MVS/ESA 版) は、ALIAS と SYNONYM という、別名についての 2 つの異なる概念を採用しています。これらの 2 つの概念は、DB2 ユニバーサル・データベースと以下の点で異なります。

- DB2 (MVS/ESA 版) の ALIAS
  - 作成者が特殊権限または特権を有していなければならない。
  - 他の別名を参照できない。
- DB2 (MVS/ESA 版) の SYNONYM

– 作成者だけしか使用できない。
– 常に修飾なしである。
– 参照テーブルが除去されると、除去される。
– ネーム・スペースを表または視点と共用しない。

ラッパーの作成

連合データベースでは、CREATE WRAPPER ステートメントはラッパーを登録します。このステートメントは、統合サーバーが特定のデータ・ソース区分との相互作用に使用するメカニズムを定義します。

特定のデータ・ソース・タイプ、バージョン、通信プロトコル、およびオペレーティング・システムには、特定のライブラリーを使用しなければなりません。たとえば、AS/400 および DB2 (OS/390 版) データ・ソースには、 APPC 通信を使用する Windows NT オペレーティング・システム上で運用する連合データベース用の `libdrda.dll` ライブラリーを使ってアクセスします。

CREATE WRAPPER ステートメントを使用するには、連合データベースで SYSADM または DBADM 権限を持っている必要があります。

コントロール・センターかコマンド行プロセッサーから作成されたラッパーは、連合データベースに登録されます。

コントロール・センターを使用してラッパーを作成するには、以下のようにします。

1. オブジェクト・ツリーを順に展開し、**「連合データベース・オブジェクト (Federated Database Objects)」**フォルダーを表示します。
2. **「連合データベース・オブジェクト (Federated Database Objects)」**フォルダーを右クリックして、ポップアップ・メニューから**「ラッパーの作成 (Create wrapper)」**を選択します。
3. 情報をすべて入力し、**「Ok」**をクリックします。

コマンド行を使用してラッパーを作成するには、以下のように入力します。

```
CREATE WRAPPER <wrapper_name> LIBRARY '<library_name>'
```

以下の SQL ステートメントは、 Windows NT オペレーティング・システム上でラッパー ORACLE8 を登録します。

```
CREATE WRAPPER ORACLE8 LIBRARY 'libnet8.dll'
```

CREATE WRAPPER ステートメントの使用についての詳細は、 *SQL 解説書* を参照してください。

サーバーの作成

連合データベースでは、サーバーを作成して DB2 にデータベースを定義し、その特性、すなわち、名前、ラッパー、バージョン、位置、およびオプションを記述します。この情報は、特定のデータ管理システムにニックネームをマップし、 DB2 最適化プログラムに情報を提供するために使用します。サーバー情報は SYSCAT.SERVERS および SYSCAT.SERVEROPTIONS カタログ視点にあります。

注: この節では、サーバーは DRDA サーバーや DB2 サーバーではなく、データ・ソースを表します。他のデータ・ソース (Oracle など) にアクセスするには、 DB2 コネクトが必須です。

ラッパーを作成した場合に限り、サーバー・オブジェクトを作成できます。

このステートメントを使用するには、連合データベースで SYSADM または DBADM 権限を持っている必要があります。

ユーザー・マッピングを作成すれば、 DB2 とデータ・ソースのサーバー間の認証処理における差異を管理できます。ユーザー・マッピングについての詳細は、249ページの『ユーザー・マッピング』に記されています。

サーバーを除去すると、そのサーバーに従属するすべてのオブジェクト (ユーザー・マッピング、ニックネーム、関数マッピング、およびプランなど) が除去されます。

サーバーの作成時には、サーバー・オプションを指定します。これらのオプションには、サーバーに関する必要な詳細 (ノード名など) が含まれます。サーバー・オプションでは、特定のパフォーマンスおよび機密保護値も設定できます。

サーバーは、コントロール・センターまたはコマンド行プロセッサーから作成できます。

コントロール・センターを使用してサーバーを作成するには、以下のようにします。

1. オブジェクト・ツリーを展開し、**「連合データベース・オブジェクト (Federated Database Objects)」**フォルダーの下に、**「サーバー (Servers)」**を表示します。
2. **「サーバー (Servers)」**フォルダーを右クリックして、ポップアップ・メニューから**「サーバーの作成 (Create server)」**を選択します。
3. 情報をすべて入力し、**「Ok」**をクリックします。

コマンド行を使用してサーバーを作成するには、以下のように入力します。

```
CREATE SERVER <server_name> TYPE <server_type>
   VERSION <server_version> WRAPPER <wrapper_name>
   OPTIONS (<server_option_name> <string_constant>)
```

以下のサンプル SQL ステートメントでは、Oracle サーバー ORA8 を作成します。

```
CREATE SERVER ORA8 TYPE ORACLE VERSION 8 WRAPPER ORACLE8 OPTIONS
(NODE 'ONODE')
```

以下のサンプル SQL ステートメントでは、DB2 サーバー DB2TEST を作成します。

```
CREATE SERVER DB2TEST TYPE DB2 VERSION 6.1 WRAPPER DB2UDB OPTIONS
(NODE 'DB2TEST', DBNAME 'TEST1')
```

SERVER SQL ステートメント中の NODE の定義は、データ・ソースによって異なります。データ・ソースが DB2 DBMS であれば、値は 1 つまたは複数のデータ・ソースを持つ DB2 のインスタンスを参照します。上記の例では、DBNAME オプションでデータベース名を指定している点に注意してください。データ・ソースが DB2 (OS/390 版) DBMS であれば、値は DB2 (OS/390 版) システムの LOCATION 名を参照します。データ・ソースが Oracle DBMS であれば、 Oracle インスタンスに含まれるデータベースは 1 つだけなので、 DBNAME オプションは不要です。

CREATE SERVER ステートメント構文についての詳細は、 *SQL 解説書* を参照してください。 CREATE SERVER ステートメントの使用についての詳細は、 *インストールおよび構成 補足* を参照してください。

データ・ソースの定義に役立ち、認証処理を容易にするサーバー・オプションの使用

統合サーバーからデータ・ソースにアクセスする方法に影響する値に対して、サーバー・オプション という変数を設定することができます。この節の内容は以下のとおりです。

- サーバー・オプションの目的。
- サーバー・オプションを指定するために使用する SQL ステートメントの説明。
- サーバー・オプションとその設定値。

サーバー・オプションの目的

サーバー・オプションは普通、次の目的で使用します。

- データ・ソースについての情報を提供し、更新する。サーバー参照には、データ・ソースについての基本情報 (たとえば、データ・ソースの名前) と、時間の経過に伴って変更される可能性がある情報の両方が含まれます。変更可能な情報の中には、サーバー・オプションに割り当てられた値によって伝えられる情報もあります。たとえば、cpu_ratio オプションに割り当てられた値は、データ・ソースの CPU が DB2 システムの CPU よりも高速であるか低速であるかを示します。 DB2 システムで 1 つまたは複数のプロセッサーがアップグレードされると、この値は変更されることになります。
- 認証を容易にする。サーバー・オプションによっては、データ・ソースにユーザー ID とパスワードを適切なケースで送信するように設定することができます。たとえば、fold_id オプションを設定すると、統合サーバーはユーザー ID をデータ・ソース

に送る前に、データ・ソースに必要なケース (大文字か小文字) に名前を変換することができます。また、ユーザー ID を必要なケースで統合サーバーに定義する場合、fold_id オプションを設定すれば、サーバーはケースを変更しようとせず、プロセス中にオーバーヘッドを消費しなくなります。

- 照会を最適化する。サーバー・オプションとその値によって、最適化を簡単に設定できる場合があります。たとえば、CREATE SERVER ステートメントでは、特定のパフォーマンス統計をオプション値として指定できます。特に、cpu_ratio オプションは、データ・ソースおよび統合サーバーの CPU の相対速度を指示する値に設定できます。また、io_ratio オプションは、データ・ソースおよび統合サーバーの入出力装置の相対速度を指示する値に設定できます。 CREATE SERVER を実行すると、これらの統計はカタログ視点 SYSCAT.SERVEROPTIONS に追加され、最適化プログラムはデータ・ソースのアクセス・プランを立てるときにそれらの統計を使用します。(たとえばデータ・ソース CPU がアップグレードされたときなどに) 統計が変更されると、 ALTER SERVER ステートメントを使用して、この変更を反映するよう SYSCAT.SERVEROPTIONS を更新できます。その後、最適化プログラムは更新された情報を使用して、データ・ソースの次のアクセス・プランを立てます。

サーバー・オプション用の SQL

サーバー・オプションに値を割り当てられる 3 つの SQL ステートメント、すなわち、CREATE SERVER、ALTER SERVER、SET SERVER OPTION があります。

CREATE SERVER ステートメントは、データ・ソースへの複数接続でいつまでも持続する値にオプションを設定する場合に使用します。このステートメントを使用すると、デフォルト以外の値にオプションを設定したり、オプションにデフォルト値がない場合にオプションを初期値に設定することができます。

ALTER SERVER ステートメントは、 CREATE SERVER ステートメントに指定した値にサーバー・オプションを設定した後、オプションを複数接続でも持続する値に設定したい場合に使用します。

SET SERVER OPTION ステートメントは、データベースへの単一接続の期間中に、一時的にサーバー・オプション値を変更する場合に使用します。このステートメントは、データ・ソースへの接続後にまず、最初の作業単位内で発行しなければなりません。

たとえば、Oracle サーバー ORASEB1 でプラン・ヒントを一時的に使用したい場合は、次のステートメントを発行します。

```
SET SERVER OPTION plan_hints TO 'Y' FOR SERVER ORASEB1
```

サーバー・オプションとその設定値

以下の表には、サーバー・オプションとそのオプションに有効な設定値が記されています。特に指定がなければ、すべてのサーバー・オプション値は単一引用符で囲む必要があります。

表 2. サーバー・オプションとその設定値

オプション	有効な設定値	デフォルト設定
collating_sequence	データ・ソースが連合データベースと同じデフォルト照合順序を使用しているかどうかを、コード・セットと国別情報に基づいて指定する。データ・ソースの照合順序が DB2 の照合順序と異なっている場合、 DB2 の照合順序に依存しているほとんどの操作は、データ・ソースにおいてリモートに評価できません。たとえば、照合順序が異なるデータ・ソースにおいて、ニックネーム文字列に対して MAX 列関数を実行する場合がそうです。 MAX 関数がリモート・データ・ソースにおいて評価される場合は結果が異なるので、 DB2 は集約操作と MAX 関数をローカルに実行します。 照会に等号が含まれている場合は、照合順序が異なっていても ('N' に設定されている)、照会のその部分を後入れ先出しすることができます。たとえば、述部 C1 = 'A' はデータ・ソースに後入れ先出しすることができます。もちろん、データ・ソースでの照合順序で大文字小文字が区別されない場合、そのような照会を後入れ先出しすることはできません。データ・ソースが大文字小文字を区別しない場合、 C1= 'A' と C1 = 'a' の結果は同じです。これは大文字小文字の区別を行う環境 (DB2) では受け入れられません。 管理者は、データ・ソースの照合順序と一致する特定の照合順序の連合データベースを作成できます。この方法を用いると、すべてのデータ・ソースが同じ照合順序を使用する場合、または、ほとんどあるいはすべての列関数が同じ照合順序を使用するデータ・ソースに向けられている場合は、パフォーマンスが高速になります。 'Y' データ・ソースの照合順序は、連合データベースの照合順序と同じ。 'N' データ・ソースの照合順序は、連合データベースの照合順序と異なる。 'I' データ・ソースの照合順序は、連合データベースの照合順序と異なっており、大文字小文字を区別しない (たとえば、'TOLLESON' と 'TolLESon' は同じものと見なされる)。	'N'
comm_rate	統合サーバーとその関連データ・ソース間の通信速度を指定する。秒当たりの MB 単位で表されます。	'2.0'

表 2. サーバー・オプションとその設定値 (続き)

オプション	有効な設定値	デフォルト設定
connectstring	OLE DB Provider への接続に必要な初期化プロパティーを指定する。接続ストリングの完全な構文と意味については、*Microsoft OLE DB 2.0 Programmer's Reference and Data Access SDK, Microsoft Press, 1998* の『Data Link API of the OLE DB Core Components』を参照してください。	なし
cpu_ratio	データ・ソースの CPU が統合サーバーの CPU より、どれほど速いまたは遅いかを表す。値 『1.0』 は、データ・ソースの CPU と連合データベース・サーバーの CPU の間の速度が等しいことを示します。値 <1.0 は、データ・ソース CPU の速度が遅いことを示します。値 >1.0 は、データ・ソース CPU の速度が速いことを示します。	'1.0'
dbname	統合サーバーがアクセスするデータ・ソース・データベースの名前。 DB2 ファミリー・データ・ソースでは必須ですが、Oracle データ・ソースには適用されません。	なし
fold_id (この表の最後にある注 1 および 4 を参照。)	統合サーバーが認証のためにデータ・ソースに送信するユーザー ID に適用される。有効な値は次のとおりです。 'U' 統合サーバーは、ユーザー ID をデータ・ソースに送信する前に、ユーザー ID を大文字に変換します。これは、 DB2 ファミリーおよび Oracle** データ・ソースについて論理的に選択できます (この表の最後にある注 2 を参照。) 'N' 統合サーバーは、ユーザー ID をデータ・ソースに送信する前に、ユーザー ID に対して何の処理も行いません。 (この表の最後の注 2 を参照。) 'L' 統合サーバーは、ユーザー ID をデータ・ソースに送信する前に、ユーザー ID を小文字に変換します。 これらの設定値のいずれも使用しない場合は、統合サーバーはユーザー ID を大文字にしてデータ・ソースに送信しようとします。そのユーザー ID を正常に送信できない場合は、サーバーはユーザー ID を小文字で送信しようとします。	なし

表 2. サーバー・オプションとその設定値 (続き)

オプション	有効な設定値	デフォルト設定
fold_pw (この表の最後にある注 1、3 および 4 を参照。)	統合サーバーが認証のためにデータ・ソースに送信するパスワードに適用される。有効な値は次のとおりです。 'U' 統合サーバーは、パスワードをデータ・ソースに送信する前に、パスワードを大文字に変換します。これは、DB2 ファミリーおよび Oracle データ・ソースについて論理的に選択できます。 'N' 統合サーバーは、パスワードをデータ・ソースに送信する前に、パスワードに対して何の処理も行いません。 'L' 統合サーバーは、パスワードをデータ・ソースに送信する前に、パスワードを小文字に変換します。 これらの設定値のいずれも使用しない場合は、統合サーバーはパスワードを大文字にしてデータ・ソースに送信しようとします。そのパスワードを正常に送信できない場合は、サーバーはパスワードを小文字で送信しようとします。	なし
io_ratio	データ・ソースの入出力システムが統合サーバーの入出力システムより、どれほど速いまたは遅いかを表す。値 『1.0』 は、データ・ソースの CPU と連合データベース・サーバーの CPU の間の速度が等しいことを示します。値 <1.0 は、データ・ソース CPU の速度が遅いことを示します。値 >1.0 は、データ・ソース CPU の速度が速いことを示します。	'1.0'
node	データ・ソースを RDBMS へのインスタンスとして定義するための名前。すべてのデータ・ソースについて必須です。 DB2 ファミリー・データ・ソースの場合、この名前は連合データベースの `DB2 node` ディレクトリーで指定されているノードです。このディレクトリーを表示するには、 **db2 list node directory** コマンドを発行します。 Oracle データ・ソースの場合、この名前は Oracle `tnsnames.ora` ファイルで指定されているサーバー名です。Windows NT プラットフォームでこの名前にアクセスするには、 Oracle SQL Net Easy Configuration ツールの「**View Configuration Information**」オプションを指定します。	なし

表 2. サーバー・オプションとその設定値 (続き)

オプション	有効な設定値	デフォルト設定
password	パスワードがデータ・ソースに送信されるかどうかを指定する。 'Y' パスワードは常にデータ・ソースに送信されて妥当性検査されます。これがデフォルト値です。 'N' パスワードはデータ・ソースに送信されず (ユーザー・マッピングに関係なく)、妥当性検査されません。 'ENCRYPTION' パスワードは常に暗号化された形式でデータ・ソースに送信されて妥当性検査されます。暗号化されたパスワードをサポートする DB2 ファミリー・データ・ソースについてのみ有効。	'Y'
plan_hints	プラン・ヒント を使用可能にするかどうかを指定する。プラン・ヒントはステートメントの一部分であり、データ・ソース最適化プログラムについての追加情報を提供します。特定の照会タイプについてこの情報を利用すれば、照会パフォーマンスを改善することができます。プラン・ヒントは、データ・ソース最適化プログラムが索引を使用するかどうか、どの索引を使用するか、またはどの表結合順序を使うかを判別するのに役立ちます。 'Y' データ・ソースがプラン・ヒントをサポートしている場合は、プラン・ヒントが使用可能になります。 'N' プラン・ヒントはデータ・ソースで使用可能になりません。	'N'
pushdown	'Y' DB2 はデータ・ソースに操作を評価させることを考慮します。 'N' DB2 はリモート・データ・ソースからの列しか検索せず、データ・ソースに結合などのその他の操作を評価させません。	'Y'

表 2. サーバー・オプションとその設定値 (続き)

オプション	有効な設定値	デフォルト設定
varchar_no_trailing_blanks	このデータ・ソースが、ブランクが埋め込まれていない VARCHAR 比較セマンティクスを使用するかどうかを指定する。後書きブランクを含んでいない可変長文字ストリングについて、一部の DBMS のブランク埋め込みなしの比較セマンティクスでは、 DB2 の比較セマンティクスと同じ結果が戻されます。データ・ソースにあるすべての VARCHAR 表 / 視点の列に後書きブランクが含まれていないことが確かである場合は、データ・ソースについてこのサーバー・オプションを 'Y' に設定することを考慮してください。このオプションは、Oracle データ・ソースでしばしば使用されます。ニックネームを持つ可能性のあるすべてのオブジェクトを考慮に入れてください。 'Y' このデータ・ソースのブランク埋め込みなしの比較セマンティクスは、DB2 と同じです。 'N' このデータ・ソースのブランク埋め込みなしの比較セマンティクスは、 DB2 と同じではありません。	'N'

この表に関する**注記**:

1. このフィールドは、認証に指定される値に関係なく適用されます。
2. DB2 はユーザー ID を大文字で格納するので、値 'N' と 'U' は論理的に互いに同等です。
3. パスワードの設定が 'N' である場合は、 fold_pw を設定しても効果はありません。パスワードが送信されないので、大文字小文字の区別は意味をなしません。
4. いずれかのオプションについてヌル値設定を使用することは避けてください。 DB2 がユーザー ID とパスワードの解決を複数回試行することになるので、ヌル値設定は有用に思えますが、パフォーマンスが低下する場合があります (DB2 が、データ・ソース認証に成功するまでにユーザー ID とパスワードを 4 回送信するということもあり得ます)。

サーバーでのパススルー・セッションの使用

パススルー・セッションにより、アプリケーションはサーバーの固有クライアント・アクセス方式と固有の SQL ダイアレクトを使ってサーバーと直接通信できます。

パススルー・セッションは、次の場合に便利です。

- アプリケーションがデータ・ソースでオブジェクトを作成したり、 INSERT、UPDATE、または DELETE 操作を実行したりする必要がある。
- DB2 が固有のデータ・ソース操作をサポートしない。

パススルー・セッションでオブジェクトを参照するときは、オブジェクト (ニックネームではない) の真の名前を使用します。

SET PASSTHRU ステートメントは、パススルー・セッションを開始し、サーバーへ直接アクセスするために使用します。このステートメントは動的に発行しなければなりません。このステートメントの例を以下に示します。

```
SET PASSTHRU BACKEND
```

このステートメントは、データ・ソース BACKEND へのパススルー・セッションをオープンします。

パススルー・セッションでの SET PASSTHRU および SQL 処理についての詳細は、*SQL 解説書* を参照してください。

ニックネームの作成

統合データ・ソースでは、ニックネームはデータ・ソース表、別名、および視点の識別子です。分散要求は普通、データ・ソース表または視点ではなく、ニックネームを参照します。

ニックネームは、DB2 が位置の透過性を提供するための手段の一部です。ニックネームはサーバー定義に依存してデータ・ソースの位置情報を検出し、データ・ソースへ効率的にアクセスします。たとえば、ALTER SERVER ステートメントは、すべてのユーザーおよびアプリケーションに関してサーバー・パフォーマンス・データとバージョン情報を透過的に更新し、新しいニックネームやアプリケーション・コードへの変更を必要としません。

ニックネームはコントロール・センターで作成することも、コマンド行プロセッサーから作成することもできます。同じデータ・ソース表または視点には、複数のニックネームを定義することができます。

ニックネームを静的 SQL ステートメントで使用することはできません。

ニックネームを作成する前に、データ・ソースで RUNSTATS コマンドに相当するコマンドを実行し、データ・ソース・オブジェクト用の統計を更新します。統計情報は、連合データベース・カタログにニックネームが作成され保管されるときに、データ・ソースから収集されます。このカタログ・データには、表と列の定義、また可能な場合には、索引の定義と統計が含まれます。

次の SQL ステートメントは CUSTOMER というニックネームを作成します。

```
CREATE NICKNAME CUSTOMER for OS390A.SHAWNB.CUSTLIST
```

このステートメントを使用するには、連合データベースで SYSADM または DBADM 権限の 1 つか、あるいは、データベース特権 IMPLICIT_SCHEMA とスキーマ特権 CREATEIN (現行スキーマ用) のいずれかを持っていなければなりません。

CREATE NICKNAME ステートメントの使用に関するさらに詳しい説明は、 *SQL 解説書* を参照してください。

ニックネームおよびデータ・ソース・オブジェクトの参照

データ・ソース・オブジェクトの参照では普通、定義されたニックネームを使用します。 1 つの例外として、パススルー・セッション内での参照 (詳しくは 166ページの『サーバーでのパススルー・セッションの使用』を参照) があります。たとえば、データ・ソース表 DB2MVS1.PERSON.DEPT にニックネーム DEPT を定義すると、ステートメント SELECT * FROM DEPT を使用できますが、ステートメント SELECT * FROM DB2MVS1.PERSON.DEPT は使用できません。

ニックネームおよびデータ・ソース・オブジェクトの処理

| 大半のユーティリティー・コマンド (LOAD、IMPORT、EXPORT、REORGCHK、
| REORGANIZE TABLE) はニックネームをサポートします。

COMMENT ON はサポートされ、連合データベースでシステム・カタログを更新します。

| Insert、update、および delete 操作は、ニックネームに対してはサポートされません。

既存のニックネームとデータ・ソースの識別

いくつかのニックネームを作成したら、以下の情報を使用して、特定のニックネームが対応するデータ・ソース、あるいは特定データ・ソースのニックネームすべてを識別したいと思うかもしれません。

ニックネームとそのデータ・ソースの識別

この例では、ニックネーム (*PAYROLL*) とその作成者 (*ACCTG*) は分かるものの、データ・ソースに関する追加情報が必要であると仮定します。以下の SQL ステートメントを使って、最初に、そのデータ・ソース (SERVER) で *PAYROLL* として知られているものに関する情報を取得します。

```
select option, setting
  from syscat.taboptions
  where tabname = 'PAYROLL'
     and tabschema = 'ACCTG'
     and option in ('SERVER','REMOTE_SCHEMA','REMOTE_TABLE');
```

このステートメントからの応答セットは DB2_MVS, FINANCE, DEPTJ35_PAYROLL です。 *PAYROLL* は、 DB2_MVS という名前のサーバーで FINANCE が所有する DEPTJ35_PAYROLL という表のニックネームであることが分かっています。この情報を後続の SELECT ステートメントで次のように使用することができます。

```
select option,setting
    from syscat.serveroptions
    where servername = 'DB2_MVS'
       and option in ('NODE','DBNAME');
```

このステートメントからの応答セットは REGIONW および DB2MVSDB3 です。表 DEPTJ35_PAYROLL は、 REGIONW というノード上の DB2MVSDB3 という名前のデータベースにあることが分かりました。

この情報があれば、LIST NODE DIRECTORY コマンドを使って、使用される通信プロトコルと機密保護タイプなど、 REGIONW ノードについての情報を取得することができます。ノードが DB2 ファミリー以外のデータ・ソース用のものであれば、そのデータ・ソースの構成ファイルを調べて同様の情報を検索する必要があります。たとえば、ノードが Oracle データ・ソースであれば、 Oracle tnsnames.ora ファイルから同様の情報を取得します。

システム・カタログ視点についての詳細は、*SQL 解説書* を参照してください。

DB2 に認識されているすべてのニックネームの識別

以下の SQL ステートメントは、統合データ・ソースに認識されているニックネームすべてのリストを提供します。その中には、ニックネームごとのスキーマ名とリモート・サーバーが含まれます。

```
select tabname,tabschema, setting as remote_server
   from syscat.taboptions
   where option = 'SERVER';
```

索引の作成、索引拡張子、または索引の指定

索引は、行の位置のリストを、指定した 1 つまたは複数の列の内容によって分類したものです。索引は、通常、表へのアクセスを高速にするために使用されます。しかし索引は、論理データ設計の点でも役立ちます。たとえば、*固有索引* を使うと、列に重複値を入力できなくなるため、表内に同じ行ができることはありません。また、列の値を昇順にするか降順にするかを指定するためにも、索引を作成することができます。

索引の拡張とは、構造型または特殊タイプの列の索引と一緒に使用される索引オブジェクトです。

索引の指定はメタデータ構成です。それは、ニックネームが参照するデータ・ソース・オブジェクト (表または視点) の索引があることを、最適化プログラムに通知します。索引の指定は行位置のリストを含んでおらず、索引を記述したものにすぎません。最適化プログラムは索引の指定を使用して、ニックネームが示すオブジェクトへのアクセスを向上させます。 ニックネームが初めて作成される場合、 DB2 が認識できる形式の基礎表用の索引がデータ・ソースにあれば、索引の指定が生成されます。

注: 視点が 1 つの表に対するものであれば、必要に応じて表のニックネームや視点のニックネームに対する索引の指定を作成します。

次の場合は、索引または索引の指定を手動で作成します。

- その結果として、パフォーマンスが向上する場合。たとえば、最適化プログラムで、ネストされたループ結合の内部表として特定の表やニックネームを使うようにする場合、索引が存在しなければ、結合列に対する索引の指定を作成します。索引または索引の指定を必要とする場合の詳細については、 *管理の手引き: パフォーマンス* を参照してください。
- 基礎表の索引が、その表のニックネームが作成された後に追加された場合。

索引の指定は、基礎表の索引が存在しない場合に作成できます (DB2 は、 CREATE INDEX ステートメントの発行時にリモート索引があるかどうかを調べません)。また、UNIQUE キーワードを指定したときでも行を固有のものにする必要はありません。

DB2 索引アドバイザーは、最適の索引セットを選ぶのに役立つウィザードです。このウィザードは、コントロール・センターからアクセスできるウィザードです。互換性のあるユーティリティーとして、*db2advis* があります。

索引は、列ごとに基礎表の中に定義されます。索引は、表の作成者、または特定の列では直接アクセスが必要であることについて知っているユーザーが作成できます。 1 次索引キーは、ユーザー定義の索引がすでに存在しているのでない限り、基本キーに対して自動的に作成されます。

1 つの基礎表に対して、索引はいくつでも作成でき、そのようにして照会のパフォーマンスを高めることができます。しかし、索引の数が多ければ多いほど、更新、削除、挿入の操作時にデータベース・マネージャーの実行する修正作業は多くなります。更新事項の多い表に対してたくさんの索引を作成すると、要求の処理が遅くなってしまう可能性があります。したがって、索引の使用は、頻繁にアクセスするための利点があるということが明らかな場合だけにしてください。

索引キーの一部となる列はどれも、255 バイトまでに制限されています。

注: DB2_INDEX_2BYTEVARLEN レジストリー変数を使用すると、 255 バイトを超える長さの列を索引キーの一部として指定することができます。

1 つの索引あたりの列の最大数は 16 です。タイプ付き表の索引を作成する場合、列の最大数は 15 です。索引キーの最大長は 1024 バイトです。前述のように、 1 つの表に対する索引キーが多くなると、要求の処理速度が低下することがあります。同様に、索引キーが大きくなっても、要求の処理速度が低下することがあります。

索引キー は 1 つの列または複数の列の集合のことであり、そこに索引が定義され、索引の有効性が判別されます。索引キーを構成する列の順序は、索引キーの作成に影響を与えることはありませんが、索引を使用するかどうかを決定するときに、最適化プログラムに影響を与えます。

索引を作成する表が空であっても、索引は作成されますが、表がロードされるか行が挿入される時点まで索引項目は作成されません。表が空でない場合、CREATE INDEX ステートメントの処理中に索引項目が作成されます。

クラスター化索引 では、新しい行がキー値の近い既存の行と物理的に近い位置に挿入されます。このことによって、データ・ページへのアクセス・パターンの線形化が進み、プリフェッチの効果性が上がるので、照会の間のパフォーマンスが向上します。

基本キーをクラスター化索引にしたい場合は、 CREATE TABLE で基本キーを指定しないでください。基本キーが作成されると、関連する索引を変更することはできなくなります。基本キー文節を指定しないで CREATE TABLE を実行します。その後、クラスター化属性を指定して CREATE INDEX を発行します。最後に、ALTER TABLE ステートメントを使用して、作成されたばかりの索引に対応する基本キーを追加します。この索引が、基本キー索引として使用されます。

一般的に、クラスター化索引が一意的なものであれば、クラスター化の保守はより効果的に行えます。

固有索引キーの一部ではないものの、その索引で保管され保守される列データは、*組み込み 列*と呼ばれます。組み込み列を指定できるのは、固有索引についてだけです。組み込み列のある索引を作成しているときは、固有キー列だけが保管され、固有なものであると見なされます。組み込み列を使用すると、索引へのアクセスが関係しているときには、データ検索のパフォーマンスが向上します。

データベース・マネージャーは、最下レベルが**葉ノード**で構成される場合の索引の保管に B+ 木構造を使用します。葉ノードや葉ページとは、実際の索引キー値が保管される場所です。索引の作成時には、上記の索引葉ページを使用可能にしてオンラインでマージしたり、再編成することができます。オンラインでの索引の再編成は、大幅な削除および更新活動の後、多数の索引葉ページにわずかの索引キーだけが残されるという状況を避けるために使用します。そのような状況でオンライン再編成を行わなければ、データと索引のオフライン再編成によってスペースを再利用できるだけです。作成する索引に、索引ページをオンラインで再編成する機能が必要かどうかを判断するには、次の質問を考慮してください。すなわち、キーを削除するたびにマージ可能なスペースの有無を検査するという追加のパフォーマンス上の費用、および (スペースが十分にある場合に) マージを完了するための実費用とを費やすことは、索引用スペースの使用効率の改善という利点に勝るか、またスペースを再利用するためにオフラインで再編成を実行するという、必要性の小さい作業に見合う価値があるか、ということです。

注: オンライン再編成によるマージ後に解放されたページは、同じ表内の他の索引に再利用することだけに使用できます。全再編成を使用すると、解放されたページを他のオブジェクト (データベース管理ストレージでの作業時)、またはディスク・スペース (システム管理ストレージでの作業時) に使用できます。また、オンライン再編

成では、索引の非葉ページは解放されませんが、全再編成では、索引を可能な限り最小化して索引を最小化することにより、非葉ページと葉ページに加えて索引のレベル数も削減されます。

オンラインで再編成する索引を実現させる方法についての詳細は、 173ページの『CREATE INDEX ステートメントの使用』を参照してください。

1 つの区分データベース内の表に対する索引は、同じ CREATE INDEX ステートメントを使用して作成されます。これらの索引は、その表の区分化キーに基づいて区分化されます。表上の索引は、ノード・グループ内の各ノード上のその表内のローカル索引から作られます。複数区画環境で定義された固有索引は、区分化キーのスーパーセットでなければならないことに注意してください。

パフォーマンス上のヒント: 以下の一連の作業を実行しようとしている場合には、

1. 表の作成
2. 表のロード
3. 索引の作成
4. RUNSTATS の実行

作業の実行順序を、以下のようにすることを考慮する必要があります。

1. 表を作成する
2. 索引を作成する
3. `statistics yes` オプションを要求して表をロードする

LOAD のパフォーマンス向上について詳しくは、 *データ移動ユーティリティー手引きおよび解説書* を参照してください。

索引作成後も、索引は常に維持されていきます。その結果、アプリケーション・プログラムが、表の中の行をランダムにアクセスおよび処理するためにキー値を使用したときに、そのキー値に基づく索引を使用して、行を直接アクセスすることができます。基礎表の中の行の物理的なストレージが順番に並んでいるわけではないので、このことは重要です。行を挿入すると、クラスター化索引の定義をしなければ、その行が単に最も便利な保管場所に入れられるだけです。特定の選択条件に一致する表の行を探索しているときで、表に索引がない場合、表全体が走査されます。索引を使用すれば、時間のかかる順次探索を実行することなく、データ検索を効率的に行えます。

索引のデータは、表データと同じ表スペース内か、または索引データが入った別個の表スペース内に保管することができます。索引データの保管に使用する表スペースは、表の作成時に決められます (141ページの『複数の表スペースへの表の作成』を参照してください)。

コントロール・センターを使用して索引を作成するには、以下のようにします。

1. オブジェクト・ツリーを順に展開し、**「索引 (Indexes)」**フォルダーを表示します。
2. **「索引 (Indexes)」**フォルダーを右クリックして、ポップアップ・メニューから**「作成 (Create)」—>「ウィザードを使用して索引を (Index Using Wizard)」**を選択します。
3. このウィザードのステップに従って、タスクをすべて実行します。

コマンド行を使用して索引を作成するには、以下のように入力します。

```
CREATE INDEX <name> ON <table_name> (<column_name>)
```

索引の作成については、次の 『索引の使用』と 『CREATE INDEX ステートメントの使用』の 2 つの部分でさらに詳しく説明します。

索引の使用

索引がアプリケーション・プログラムによって直接使用されることはありません。索引を使用するかどうか、および使用できる可能性がある索引はどれかを判断するのは、最適化プログラムの仕事です。

表上での最適な索引は、以下のものです。

- 高速ディスクを使用するもの
- 高クラスター化されたもの
- 狭い範囲だけの列からなるもの

索引の利点に関する詳しい説明については、 *管理の手引き: パフォーマンス* の『索引走査の概念』を参照してください。

CREATE INDEX ステートメントの使用

重複値の許容される索引 (非固有索引) を作成することができます。それによって、基本キー以外の列による効率的な検索が可能になり、索引列に重複値を入れることができるようになります。

以下の SQL ステートメントは、EMPLOYEE 表の LASTNAME 列を昇順にソートしたものから、 LNAME という非固有索引を作成します。

```
CREATE INDEX LNAME ON EMPLOYEE (LASTNAME ASC)
```

次の SQL ステートメントは、電話番号列に基づく固有索引を作成するものです。

```
CREATE UNIQUE INDEX PH ON EMPLOYEE (PHONENO DESC)
```

固有索引を使う場合は、索引列に重複値を入れることができなくなります。行を更新するかまたは新しい行を挿入する SQL ステートメントの終わりに、制約が施行されます。 1 つ以上の列のセットがすでに重複値をもっている場合には、このタイプの索引は作成できません。

ASC というキーワードは、索引項目を列ごとの昇順にします。また、DESC は、列ごとに降順にします。デフォルトは昇順です。

固有索引を 2 つの列に作成することができます。そのうちの 1 つは組み込み列です。基本キーは、組み込み列ではない方の列で定義されます。どちらの列も同じ表での基本キーとして、カタログ内に表示されます。通常、1 つの表当たり 1 つの基本キーだけです。

INCLUDE 文節では、一連の索引キー列に追加列を付加することを指定します。この文節に組み込まれる列は、固有性を強制するためには使用されません。組み込み列は、索引のみのアクセスを行うことにより、照会のパフォーマンスを向上させることがあります。これらの列は、固有性を強制するために使用される列とは異なっていなければなりません (そうしないと、エラー・メッセージ SQLSTATE 42711 が出されます)。列の数および長さ属性の合計の限界は、固有キー内と索引内のすべての列に適用されます。

既存の索引が基本キー定義と一致しているかどうか判別するために、検査が実行されます (索引内の INCLUDE 列はすべて無視されます)。索引の定義が、列の順序または方向 (昇順か降順のいずれか) の指定に関係なく、同じ列のセットを識別する場合は、索引の定義が一致します。一致する索引定義が見つかった場合、システムが求めれば、索引の記述はそれが基本索引であることを示すように変更されます。また、それが非固有索引であれば (固有性を確認した後で) 固有に変更されます。

なぜなら、カタログ内で示された同じ表に複数の基本キーを持っている可能性があるからです。

構造型を処理する際、構造型がユーザー定義索引タイプの作成に必要になることがあります。その場合は、索引の保守、索引の検索、および索引活用機能を定義する手段が必要になります。索引タイプの作成要件についての詳細は、*SQL 解説書* を参照してください。

次の SQL ステートメントは、 EMPLOYEE 表の LASTNAME 列に INDEX1 というクラスター化索引を作成するものです。

```
CREATE INDEX INDEX1 ON EMPLOYEE (LASTNAME) CLUSTER
```

データベースの内部記憶域を効率的に使用するには、 ALTER TABLE ステートメントの PCTFREE パラメーターでクラスター索引を使用します。このようにすると、新しいデータを正しいページに挿入できます。データが正しいページに挿入されると、クラスター化の順序が崩れることがありません。通常、表に対する INSERT アクティビティーが多ければ多いほど、クラスター化を保守するために必要な (その表での) PCTFREE 値

は大きくなります。この索引は、データが物理ページに置かれる順序を決定するので、特定の表について定義できるクラスター化索引は 1 つのみです。

一方、これらの新しい行の索引キー値が、たとえば常に高いキー値である場合は、表のクラスター化属性はそのキー値を表の最後に置こうとします。他のページに空きスペースがあっても、クラスター化を保つことについてはほとんど意味がありません。この場合、表を追加モードにすることは、クラスター化索引および PCTFREE 値を大きくするという方法よりもよい方法です。 ALTER TABLE APPEND ON を出して、表を追加モードにすることができます。 ALTER TABLE に関する概説については、211ページの『表属性の変更』を参照してください。 ALTER TABLE に関する詳細については、*SQL 解説書* を参照してください。

上記の事柄は、UPDATE によって生成される、行のサイズを増加させる新しい「オーバーフロー」行についても当てはまります。

CREATE INDEX ステートメントの MINPCTUSED 文節は、索引葉ページ上で使用される最小スペースの限界値を指定します。この文節が使用されると、この索引用にオンラインの索引再編成が使用可能になります。それが使用可能になったら、以下の考慮事項を使用して、オンライン再編成を行うかどうかを決定します。すなわち、この索引の葉ページからキーが削除され、ページ上の使用済みスペースのパーセンテージが指定限界値を下回ったら、近隣の索引葉ページを調べて、 2 つの葉ページ上のキーを単一の索引葉ページにマージできるかどうかを判別します。

たとえば、次の SQL ステートメントは、オンライン索引再編成が使用可能な索引を作成します。

```
CREATE INDEX LASTN ON EMPLOYEE (LASTNAME) MINPCTUSED=20
```

この索引からキーが削除されるとき、索引ページに残っているキーが索引ページの 20 パーセント以下を占めていれば、この索引ページのキーと近隣の索引ページのキーをマージして索引ページを削除しようとします。結合したキーが単一ページにすべて収まる場合、このマージは実行され、索引ページの 1 つが削除されます。

CREATE INDEX ステートメント PCTFREE 文節は、索引が作成されるときに空きスペースとして残す各索引ページのパーセンテージを指定します。索引ページに多くの空きスペースを残すと、分割されるページが少なくなります。このようにすると、順次索引ページを回復するために表を再編成する必要性が少なくなりますが、この必要性はプリフェッチを増大させるものです。プリフェッチは、パフォーマンスを向上させる 1 つの重要なコンポーネントです。また、キー値が常に高い場合は、 CREATE INDEX ステートメントの PCTFREE 文節の値を低くすることを考慮したいでしょう。そうすることにより、各索引ページ上で予約されるむだなスペースは限られたものになります。

複製要約表がある場合は、その基礎表 (複数も可) には固有索引がなければならず、複製要約表を定義する照会で索引キー列が使用される必要があります。複製要約表についての詳細は、*管理の手引き: 計画* を参照してください。

区画内並列化の場合、索引作成のパフォーマンスは、索引作成の間に実行されるスキャンとソートに複数のプロセッサーを使用することによって向上します。複数プロセッサーの使用は、 *intra_parallel* を YES(1) または ANY(-1) に設定することによって可能になります。索引作成の間に使用されるプロセッサーの数はシステムによって決定され、構成パラメーターの *dft_degree* または *max_querydegree*、アプリケーションの実行時の程度、または SQL ステートメントのコンパイルの程度によって影響を受けることはありません。データベース構成パラメーターの *indexsort* が NO である場合、索引作成は複数のプロセッサーを使用しません。

複数区分データベースでは、固有索引を区分化キーのスーパーセットとして定義しなければなりません。

ユーザー定義拡張索引タイプの作成

DB2 ユニバーサル・データベースでは、ユーザー定義の索引タイプをサポートするため、索引がどのように機能するかを制御する主なコンポーネントの論理を、独自に作成して適用できます。この種の置換可能なコンポーネントには、以下のものが含まれます。

- 索引の保守。これは、索引列の内容を索引キーにマップすることを可能にします。この種のマッピングは、ユーザー定義のマッピング関数を使用して行われます。 1 つの拡張索引に関与できる構造型列は 1 つだけです。拡張索引は、通常の索引とは異なり、 1 行あたり複数の索引項目を持つことができます。行ごとに複数の索引項目を使用することにより、たとえば、テキスト文書を 1 つのオブジェクトとして格納し、文書中の各キーワードを別々の索引項目とすることができます。
- 索引活用。アプリケーション設計者は、これを使用すると、フィルター処理条件 (範囲述部) をユーザー定義関数 (UDF) と関連付けることができます。そうしないと、UDF は最適化プログラムから認識されません。これにより、DB2 は行ごとに別々に UDF 呼び出しを行わずに済むので、クライアントとサーバーの間でコンテキストを切り替える必要がなくなり、パフォーマンスが大幅に向上します。

注: ユーザー定義関数の定義は決定的でなければならず、外部アクションを許可して、最適化プログラムから活用できるようにしてはなりません。

オプションでデータ・フィルター関数も指定できます。最適化プログラムは、取り出されたタプルに対してフィルターを使用してから、ユーザー定義関数を評価します。

構造型または特殊タイプの列に限り、索引の拡張を使用して、これらのオブジェクトに対してユーザー定義の拡張型タイプを作成できます。ユーザー定義の拡張索引タイプには以下のことは行えません。

- クラスター索引と共に定義すること。
- INCLUDE 列を持つこと。

索引の保守に関する詳細

CREATE INDEX EXTENSION ステートメントを使用して、索引の操作を制御する 2 つのコンポーネントを定義できます。

索引の保守とは、索引列の内容 (またはソース・キー) をターゲットの索引キーに変換するプロセスのことです。変換プロセスは、データベースに事前定義された表関数を使用して定義されます。

FROM SOURCE KEY 文節は、この索引の拡張でサポートされるソース・キー列のタイプを、構造型データ・タイプまたは特殊タイプとして指定します。パラメーター名とデータ・タイプを 1 つずつ指定して、それらをソース・キー列と関連付けます。

GENERATE KEY USING 文節は、索引キーの生成に使用されるユーザー定義表関数を指定します。この関数からの出力の指定は、 TARGET KEY 文節の指定の中で行わなければなりません。 この関数からの出力を、 FILTER USING 文節で指定される索引フィルター関数の入力として使用することもできます。

索引の検索に関する詳細

索引の検索は、検索引き数を検索範囲にマップします。

CREATE INDEX EXTENSION ステートメントの WITH TARGET KEY 文節は、ターゲット・キー・パラメーターを指定します。このパラメーターは、GENERATE KEY USING 文節に指定されるユーザー定義表関数の出力です。パラメーター名とデータ・タイプを 1 つずつ指定して、それらをターゲット・キー列と関連付けます。このパラメーターは、 GENERATE KEY USING 文節のユーザー定義表関数の RETURNS 表の列に対応します。

SEARCH METHODS 文節は、索引の検索の方式を 1 つまたは複数定義します。検索の方式はそれぞれ、方式名、検索引き数、範囲作成関数、およびオプションの索引フィルター関数から成ります。検索の方式はそれぞれ、ユーザー定義表関数が、基礎となるユーザー定義索引の索引検索範囲をどのように作成するかを定義します。さらに、検索の方式はそれぞれ、特定の検索範囲にある索引項目が、単一値を戻すためにユーザー定義のスカラー関数によってさらに限定される方法を定義します。

- WHEN 文節は、ラベルを検索の方式と関連付けます。ラベルとは、SQL 識別子の一種で、索引活用規則 (ユーザー定義関数の PREDICATES 述部にある) で指定された方式名に関連しています。範囲関数および索引フィルター関数 (またはその一方) の引き数として使用するため、 1 つまたは複数のパラメーター名とデータ・タイプを指定します。 WHEN 文節は、 CREATE FUNCTION ステートメントの PREDICATES 文節と着信する照会とが合致した場合に、最適化プログラムによって取られるアクションを指定します。
- RANGE THROUGH 文節は、索引キーの範囲を作成するユーザー定義の外部表関数を指定します。この文節を使用すると、索引キーがキー範囲外にある場合に、関連する UDF を最適化プログラムが呼び出さないようにできます。

- FILTER USING 文節は、範囲作成関数から戻される索引項目をフィルター処理する、ユーザー定義の外部表関数または case 式を指定します (オプション)。索引フィルター関数または case 式から値 1 が戻された場合は、索引項目に対応する行が表から取り出されます。 1 以外の値が戻された場合、索引項目は廃棄されます。この機能が有用なのは、 2 次フィルターの費用が元の方式の評価の費用に比べて低く、かつ 2 次フィルターの選択率が比較的低い場合です。

索引活用に関する詳細

索引活用は、検索の方式の評価時に生じます。

CREATE FUNCTION (外部スカラー) ステートメントは、索引の拡張のために定義された検索の方式と共に使用される、ユーザー定義の述部を作成します。

PREDICATES 文節は、この関数が使用されている述部、つまり索引の拡張を活用する可能性のある (および検索条件のためにオプションの SELECTIVITY 文節を使用する可能性のある) 述部を識別します。 PREDICATES 述部を指定する場合は、NO EXTERNAL ACTION を指定して、関数を DETERMINISTIC として定義しなければなりません。

- WHEN 文節の後ろには、述部の中で比較演算子 (=、>、 < など) および定数または式 (EXPRESSION AS 文節を使用) と共に定義される関数の、特定の使用法が続きます。述部でこの関数が、同じ比較演算子、および所定の定数または式と共に使用される場合、フィルター処理や索引活用を使用できます。定数は、主に結果タイプ 1 または 0 のブール式を処理するために使用します。それ以外の場合はいずれも、EXPRESSION AS 文節を選択する方が賢明です。
- FILTER USING 文節は、結果表に追加のフィルター処理を実行するために使用できるフィルター関数を識別します。これは、定義される関数 (述部で使用される関数) に代わる、より高速な関数です。適格かどうかを判断するためにユーザー定義の述部を実行しなければならない行数を削減します。索引によって作成された結果が、ユーザー定義の述部が予期する結果と近い場合、このフィルター関数の適用は余分なものとなる可能性があります。
- オプションで、索引を活用するために、索引の拡張の検索の方式ごとに一連の規則を定義することができます。また、索引の拡張に検索の方式を定義し、検索のターゲット、検索引き数、および索引の検索を実行する際の使用法を記述することもできます。
  - SEARCH BY INDEX EXTENSION 文節は、索引の拡張を識別します。
  - オプションの EXACT 文節は、述部の評価の点で、索引の検索が厳密に行われることを示します。この文節は、索引の検索の後、元のユーザー指定の述部関数またはフィルター関数を適用しないよう、データベースに指示します。索引の検索を使用しない場合は、元の述部とフィルター関数を適用する必要があります。 EXACT 文節を使用しないと、索引の検索の後で、元のユーザー指定の述部が適用されます。 EXACT 述部は、索引の検索により戻される結果が、述部と同じである場合に効果的です。この述部により、照会の実行時に、索引検索から得られる結果にユ

ーザー定義の述部が適用されないようにすることができます。索引が述部に近似しているに過ぎないと予想される場合は、 EXACT 文節を指定しないでください、

– WHEN KEY 文節は、検索のターゲットを定義します。キー当たり 1 つだけ検索ターゲットを指定できます。 WHEN KEY 文節の後に指定する値は、定義しようとしている関数のパラメーター名を識別します。指定されたパラメータの値が、指定された索引の拡張に基づく索引に含まれる列である場合、この文節は真として評価されます。

– USE 文節は、検索引き数を定義します。検索引き数は、索引の拡張で定義されている方式のうちどれを使用するかを識別します。ここに指定する方式名は、索引の拡張に定義されている方式と一致していなければなりません。 1 つまたは複数のパラメーター値により、定義する関数のパラメーター名を識別します。この値は、検索ターゲットに指定されているパラメーター名とは別の値でなければなりません。パラメーター値の数とその個々のデータ・タイプは、索引の拡張中に定義されている方式のパラメーターと一致していなければなりません。組み込みデータ・タイプと特殊データ・タイプの場合、一致は厳密でなければならず、同じ構造型に属していなければなりません。

索引の拡張の定義のシナリオ

索引の拡張の定義のシナリオを以下に示します。

1. 構造型 (形状) を定義します。 CREATE TYPE ステートメントを使用してタイプ階層を定義します。形状はスーパータイプで、形状なし、点、線、および多角形はサブタイプです。これらの構造型は、立体的な実体をモデル化します。たとえば、店の場所は点、河川の流域は線、営業上の地区の境界は多角形になります。最低限の境界を持つ長方形 (minimum bounded rectangle: mbr) は 1 つの属性です。 gtype 属性は、関連した実体が点、線、または多角形のいずれであるかを識別します。地理的な境界は、 numpart、numpoint、および geometry 属性によってモデル化されます。これら以外の属性は、このシナリオとは関係ないので無視します。
2. 索引の拡張を作成します。
   - CREATE FUNCTION ステートメントを使用して、キー変換 (gridentry)、範囲の作成 (gridrange)、および索引のフィルター処理 (checkduplicate と mbroverlap) に使用する関数を作成します。
   - CREATE INDEX EXTENSION ステートメントを使用して、索引のコンポーネントとして必要なもののうち、残りのものを作成します。
3. 索引の保守コンポーネントに対応するキー変換を作成します。

```
CREATE INDEX EXTENSION iename (parm_name datatype, ...)
   FROM SOURCE KEY (parm_name datatype)
   GENERATE KEY USING table_function_invocation
   ...
```

FROM SOURCE KEY 文節は、キー変換のパラメーターとデータ・タイプを識別します。 GENERATE KEY USING 文節は、自分が生成した値とソース・キーとをマップする関数を識別します。

4. 索引の検索コンポーネントに対応する、範囲作成関数と索引フィルター関数を定義します。

```
CREATE INDEX EXTENSION iename (parm_name datatype, ...)
   ...
   WITH TARGET KEY
      WHEN method_name (parm_name datatype, ...)
      RANGE THROUGH range_producing_function_invocation
      FILTER USING index_filtering_function_invocation
```

WITH TARGET KEY 文節は、検索の方式の定義を識別します。 WHEN 文節は、方式名を識別します。 RANGE THROUGH 文節は、使用される索引の効力範囲を制限するのに使用される関数を識別します。 FILTER USING 文節は、結果の索引値から不要な項目を除去するのに使用される関数を識別します。

注: FILTER USING 文節は、索引フィルター処理関数の代わりに case 式を識別することもできます。

5. 索引の拡張を活用するための述部を定義します。

```
CREATE FUNCTION within (x shape, y shape)
   RETURNS INTEGER
   ...
   PREDICATES
      WHEN = 1
         FILTER USING mbrWithin (x..mbr..xmin, ...)
         SEARCH BY INDEX EXTENSION grid_extension
         WHEN KEY (parm_name) USE method_name(parm_name)
```

PREDICATES 文節の後ろには、1 つまたは複数の述部が続きます。それぞれの述部は、WHEN 文節で始まります。 WHEN 文節は、比較演算子とその後に続く定数または EXPRESSION AS 文節で始まり、その後に述部の指定が続きます。 FILTER USING 文節は、結果表に追加のフィルター処理を実行するために使用できるフィルター関数を識別します。これは、定義される関数 (述部で使用される関数) に代わる、より費用の低い関数です。適格な行を判別するためにユーザー定義の述部を実行しなければならない行数を削減します。 SEARCH BY INDEX EXTENSION 文節は、索引活用を実行する場所を指定します。索引活用は、索引を活用するために使用できる索引の拡張の検索の方式を使用して、一連の規則を定義します。 WHEN KEY 文節は、活用の規則を指定します。活用の規則として、検索ターゲット、検索引き数、および検索の方式を使用して索引検索を実行する際の使用法を記述できます。

6. フィルター関数を定義します。

```
CREATE FUNCTION mbrWithin (...)
```

ここで定義されている関数は、索引の拡張の述部で使用するために作成されます。

照会のパフォーマンスを向上するために作成した索引が、照会最適化プログラムにより正常に活用されるようにするため、関数呼び出しに SELECTIVITY オプションを使用できます。述部によって戻される行のパーセンテージを指定したい場合は、関数呼び出しに SELECTIVITY オプションを使用して、 DB2 最適化プログラムが有効なアクセス・パスを選択するようにすることができます。

以下の例で、 within ユーザー定義関数は中心と半径を (中心は 1 つ目のパラメーターに基づいて、半径は 2 つ目のパラメーターに基づいて) 計算し、該当する選択率でステートメントのストリングを構築します。

```
SELECT * FROM customer
   WHERE within(loc, circle(100, 100, 10)) = 1 SELECTIVITY .05
```

この例に示されている述部 (SELECTIVITY .05) は、 customer 表の行のうち 95 パーセントをフィルター処理します。

第4章 データベースの変更

この章では、データベースを変更する前に考慮しなければならない点、およびデータベース・オブジェクトの変更方法と除去方法について説明します。

データベースを変更する前に

データベース設計が実現した後に、データベース設計の変更が必要になる場合があります。以前の設計での主要な設計上の論点を再考慮する必要が生じます。以下の点に特別な注意を払う必要があります。

- 『論理的および物理的な設計特性の変更』
- 『ライセンス情報の変更』
- 『インスタンスの変更』
- 187ページの『環境変数およびプロファイル・レジストリー変数の変更』
- 188ページの『ノード構成ファイルの変更』
- 188ページの『データベース構成の変更』

論理的および物理的な設計特性の変更

データベース全体に影響を与える変更を行う前に、すべての論理的および物理的な設計の決定事項について見直す必要があります。たとえば、1 つの表スペースを変更する場合、SMS または DMS のストレージ・タイプの使用に関して、設計上の決定事項の見直しを行う必要があります。 (詳細は、*管理の手引き: 計画* を参照してください。)

ライセンス情報の変更

DB2 製品のライセンス管理の一環として、ライセンスの数を増やす必要があるかもしれません。コントロール・センターの中のライセンス・センターを使用すれば、インストールした製品の使用状況を調べ、その使用状況に基づいてライセンスの数を増やすことができます。

インスタンスの変更

インスタンスは、後ほど製品をインストールしたり削除したりしても、できるだけ影響を受けないように設定されています。

既存のインスタンスはたいてい、インストールまたは削除する製品の機能を自動的に継承するか、アクセスを失います。しかし、特定の実行可能コードやコンポーネントがインストールまたは削除された場合、既存のインスタンスは自動的に新しいシステムの構

成パラメーターを継承したり、すべての追加機能へのアクセスを取得したりするわけではありません。そのインスタンスは更新しなければなりません。

プログラム一時修正 (PTF) またはパッチを使って DB2 を更新する場合は、 **db2iupdt** コマンドを使って既存のすべての DB2 インスタンスを更新する必要があります。また、**dasiupdt** コマンドを使って管理サーバー (DAS) を更新する必要もあります。

インスタンスを変更または削除する前には、インスタンス内にあるインスタンス・サーバーとデータベース区画サーバーを理解しておく必要があります。

インスタンスのリスト

コントロール・センターを使用して、システムで使用可能なインスタンスすべてをリストするには、次のようにします。

1. オブジェクト・ツリーを順に展開し、**「データベース (Databases)」**フォルダーを表示します。
2. インスタンスをリストするデータベースを右クリックして、ポップアップ・メニューから**「追加 (Add)」**を選択します。
3. **「最新表示 (Refresh)」**をクリックし、**「データベース名 (Database name)」**フィールドをクリックして、インスタンスのリストを表示します。
4. **「取消 (Cancel)」**を押します。

コマンド行を使用して、システムで使用可能なインスタンスすべてをリストするには、次のようにします。

```
db2ilist
```

(OS/2 またはサポートされている Windows プラットフォーム上で) 現行セッションに適用されるインスタンスを判別するには、次のコマンドを使用します。

```
set db2instance
```

インスタンスの構成の更新

db2iupdt コマンドを実行すると、以下の事柄を実行して指定インスタンスが更新されます。

- インスタンス所有者のホーム・ディレクトリー下の `sqllib` サブディレクトリーにあるファイルを置き換える。
- ノード・タイプが変更されていれば、新しいデータベース・マネージャー構成ファイルが作成される。この作成は、既存のデータベース・マネージャー構成ファイルから関連する値を、新しいノード・タイプ用のデフォルトのデータベース・マネージャー構成ファイルとマージして行います。新しいデータベース・マネージャー構成ファイ

ルが作成されると、古いファイルは、インスタンス所有者のホーム・ディレクトリーにある sqllib サブディレクトリーの backup サブディレクトリーにバックアップされます。

db2iupdt コマンドは、バージョンおよびリリース・サブディレクトリー (厳密な名前はオペレーティング・システムによって異なる) の instance サブディレクトリーにあります。

コマンドは、次のように使用します。

```
db2iupdt InstName
```

InstName はインスタンス所有者のログイン名です。

このコマンドに関連して、他にも次の任意指定パラメーターがあります。

- -h または -?

  このコマンドのヘルプ・メニューを表示します。

- -d

  問題判別中に使用するデバッグ・モードを設定します。

- -a AuthType

  インスタンスの認証タイプを指定します。有効な認証タイプは、SERVER、CLIENT、DCS、または DCE です。指定しない場合、DB2 サーバーがインストールされていれば、デフォルトは SERVER に設定されます。それ以外の場合は、CLIENT に設定されます。インスタンスの認証タイプは、インスタンスが所有するすべてのデータベースに適用されます。

  UNIX オペレーティング・システムの場合、DCE は有効な認証タイプではありません。

- -e

  既存の各インスタンスを更新できます。存在するものは **db2ilist** を使用して表示できます。

- -u FencedID

  分離ユーザー定義関数 (UDF) とストアード・プロシージャーを実行するユーザーの名前を指定します。 DB2 クライアントまたは DB2 ソフトウェア開発者キットをインストールする場合、これは必須ではありません。 他の DB2 製品の場合、これは必須パラメーターです。

  注: FencedID は、『root』 または 『bin』 ではありません。

- -k

  このパラメーターは、現在のインスタンス・タイプを保存します。このパラメーターを指定しない場合、現行インスタンスは次の順序で、使用可能な上位のインスタンス・タイプにアップグレードされます。

– ローカルおよびリモート・クライアントを持つ区分データベース・サーバー (DB2 エンタープライズ拡張エディションのデフォルト・インスタンス・タイプ)
– ローカルおよびリモート・クライアントを持つデータベース・サーバー (DB2 ユニバーサル・データベース エンタープライズ・エディションのデフォルト・インスタンス・タイプ)
– クライアント (DB2 クライアントのデフォルト・インスタンス・タイプ)

次に例を示します。

- インスタンスの作成後に DB2 ユニバーサル・データベース ワークグループ・エディションまたは DB2 ユニバーサル・データベース エンタープライズ・エディションをインストールした場合は、次のコマンドを入力してインスタンスを更新してください。

  ```
  db2iupdt -u db2fenc1 db2inst1
  ```

- DB2 コネクト エンタープライズ・エディションをインストールした場合は、インスタンス名を分離 ID としても使用できます。

  ```
  db2iupdt -u db2inst1 db2inst1
  ```

- クライアント・インスタンスを更新するには、次のコマンドを使用することができます。

  ```
  db2iupdt db2inst1
  ```

インスタンスの除去

コントロール・センターを使用してインスタンスを除去するには、以下のようにします。

1. オブジェクト・ツリーを順に展開し、除去したいインスタンスを表示します。
2. インスタンス名を右クリックして、ポップアップ・メニューから**「除去 (Remove)」**を選択します。
3. **「確認 (Confirmation)」**ボックスにチェックを付け、**「Ok」**をクリックします。

コマンド行を使用してインスタンスを除去するには、以下のように入力します。

```
db2idrop <instance_name>
```

コマンド行を使用してインスタンスを除去する場合の、準備事項と詳細は以下のとおりです。

1. インスタンスを現在使用しているアプリケーションをすべて停止します。
2. それぞれの DB2 コマンド・ウィンドウで **db2 terminate** コマンドを実行して、コマンド行プロセッサーを停止します。
3. **db2stop** コマンドを実行してインスタンスを停止します。

4. DB2INSTPROF レジストリー変数で指示されたインスタンス・ディレクトリーをバックアップします。 UNIX オペレーティング・システムでは、 INSTHOME/sqllib ディレクトリー (ただし、INSTHOME は、インスタンス所有者のホーム・ディレクトリー) にファイルをバックアップすることを考慮します。たとえば、データベース・マネージャー構成ファイル db2systm、 db2nodes.cfg ファイル、ユーザー定義関数 (UDF)、または分離したストアード・プロシージャー・アプリケーションを保管したいと思うかもしれません。
5. (UNIX オペレーティング・システム上のみ) インスタンス所有者としてログオフします。
6. (UNIX オペレーティング・システム上のみ) root 権限を持つユーザーとしてログインします。
7. 次のように **db2idrop** コマンドを発行します。

   ```
   db2idrop InstName
   ```

   ここで、InstName は、除去されるインスタンスの名前です。

   このコマンドを使うと、インスタンスのリストからインスタンス項目が除去され、インスタンス・ディレクトリーが除去されます。
8. (UNIX オペレーティング・システム上のみ) root 権限を持つユーザーとして、このインスタンス所有者のユーザー ID およびグループ (そのインスタンスだけに使用している場合) を除去することもできます。インスタンスの再作成を計画している場合は、上記の除去は行わないでください。

   インスタンス所有者およびインスタンス所有者グループは他の目的で使用することもあるので、このステップは任意選択です。

db2idrop コマンドを使うと、インスタンスのリストからインスタンス項目が除去され、インスタンス所有者のホーム・ディレクトリーにある sqllib サブディレクトリーが除去されます。

環境変数およびプロファイル・レジストリー変数の変更

どの環境変数 (存在する場合) を特定のオペレーティング・システム上で変更する必要があるかを考慮しなければなりません。何らかの環境変数が変更され、ユーザーが UNIX プラットフォーム上にいない場合、新しい環境変数を有効にするためにシステムを再始動する必要があります。データベースを変更する前に、グローバル・プロファイル・レジストリーの中のプロファイル・レジストリー変数をリセットする必要があるかどうかを見直してください。その後で、プロファイル・レジストリー変数を、新しいデータベース環境にもっとも適したものに再設定することができます。プロファイル・レジストリー変数だけが変更されている場合、システムの再始動は必要ありません。

ノード構成ファイルの変更

ノードグループに対する変更 (ノードの追加または削除、あるいは既存のノードの移動) を計画している場合は、行うべきことに関する詳細について、 *管理の手引き: パフォーマンス* の『プロセッサーの追加による構成のスケーリング』を参照してください。

データベース構成の変更

データベースに対する変更を計画している場合、構成パラメーターの値を見直す必要があります。一部の値は、データベースに対して行われる現在進行中の変更の一部として、データベースが使用されている方法に基づいて時々調整することができます。

データベース構成を変更するには、コントロール・センター内のパフォーマンス構成ウィザードを使用します。このウィザードは、どの構成パラメーターを修正したらよいかを提案し、それらに推奨値を提供することによって、インスタンスごとの単一データベースのパフォーマンスのチューニングとメモリー所要量のバランスを行うのを手助けします。

注: パラメーターを修正しても、以下の時点まで値は更新されません。

- データベース・パラメーターの場合、すべてのアプリケーションが切断された後で、そのデータベースに対する最初の新しい接続が行われるまで。
- データベース・マネージャー・パラメーターの場合、次回、そのインスタンスを停止して開始するまで。

ほとんどのケースで、パフォーマンス構成ウィザードによって推奨された値は、デフォルト値よりもパフォーマンスがよくなります。これは、ユーザーの作業負荷とユーザー自身の固有のサーバーについての情報に基づいた値であるためです。ただし、この値は、指定されたデータベース・システムのパフォーマンスを改善するよう設計されたものであり、必ずしも最適なものではないことに注意してください。これらの値は、最適なパフォーマンスを獲得するために、さらに調整を行うための出発点と考えてください。

コントロール・センターを使用してデータベース構成に変更を加えるには、以下のようにします。

1. オブジェクト・ツリーを順に展開し、**「データベース (Databases)」**フォルダーを表示します。
2. 変更を加えたいインスタンスまたはデータベースを右クリックして、ポップアップ・メニューから**「ウィザードを使用してパフォーマンスを構成する (Configure Performance Using Wizard)」**を選択します。
3. 個々のページをクリックし、必要に応じて情報に変更を加えます。
4. **「結果 (Results)」**ページをクリックして、作業内容を検討し、推奨されている構成パラメーターを適用します。
5. 更新を適用し終えたら、**「終了 (Finish)」**をクリックします。

コマンド行を使用してデータベース・マネージャー構成に変更を加えるには、以下のようにします。

```
UPDATE DBM CFG FOR <database_alias>
   USING <config_keyword>=<value>
```

1 つのコマンドで 1 つまたは複数の <config_keyword>=<value> の組み合わせを更新できます。データベース・マネージャー構成ファイルに対する変更内容は、そのほとんどが、メモリーへのロード後にはじめて有効になります。サーバー構成パラメーターの場合は、 START DATABASE MANAGER コマンドの実行時に有効になります。クライアント構成パラメーターの場合は、アプリケーションの再始動時に有効になります。

現行のデータベース・マネージャー構成パラメーターを表示したり印刷したりするには、 GET DATABASE MANAGER CONFIGURATION コマンドを使用してください。

ベンチマークによってシステムを改善する方法、およびシステムを構成する方法についての詳細は、 *管理の手引き: パフォーマンス* の『ベンチマーク・テスト』および『DB2 の構成』を参照してください。

複数区画の場合: 複数の区画にわたって区分化されたデータベースを持っている場合、データベース構成ファイルは、すべてのデータベース区画で同じものである必要があります。 SQL コンパイラーは、ノードの構成ファイルの情報に基づいて分散 SQL ステートメントをコンパイルし、 SQL ステートメントのニーズを満足させるためのアクセス・プランを作成するので、これらの構成ファイルには整合性が必要です。 データベース区画ごとに異なる構成ファイルを維持していると、どのデータベース区画でステートメントが準備されたかによって、異なるアクセス・プランが作成される可能性があります。 **db2_all** を使用して、すべてのデータベース区画で構成ファイルを保守してください。

データベースの変更

データベースを変更するときのタスクは、データベースを作成するときのタスクと同じほど多くあります。これらのタスクは、以前に作成されたデータベースのある性質を更新したり除去したりします。タスクには以下のものがあります。

- 『データベースの除去』
- 191ページの『ノード・グループの変更』
- 191ページの『表スペースの変更』
- 197ページの『スキーマの除去』
- 198ページの『構造と内容の両方における表の修正』
- 214ページの『ユーザー定義構造型の変更』
- 214ページの『タイプ付き表の行の削除および更新』
- 214ページの『既存の表の名前変更』
- 215ページの『表の除去』
- 217ページの『トリガーの除去』
- 218ページの『ユーザー定義関数 (UDF)、タイプ・マッピング、方式の除去』
- 218ページの『ユーザー定義タイプ (UDT) またはタイプ・マッピングの除去』
- 219ページの『視点の変更または除去』
- 221ページの『要約表の除去』
- 222ページの『サーバーの変更または除去』
- 223ページの『ニックネームの変更または除去』
- 224ページの『索引、索引の拡張、または索引の指定の除去』
- 225ページの『オブジェクトを変更する場合のステートメントの従属関係』

データベースの除去

データベース内の一部のオブジェクトは変更可能ですが、データベースそのものを変更することはできません。変更するためには、データベースを除去して再作成しなければなりません。データベースを除去することは、データベース内のすべてのオブジェクト、コンテナー、関連ファイルが削除されるため、広範囲に及ぶ影響があります。データベースを除去すると、そのデータベースはデータベース・ディレクトリーから除去 (アンカタログ) されます。

コントロール・センターを使用してデータベースを除去するには、以下のようにします。

1. オブジェクト・ツリーを順に展開し、**「データベース (Databases)」**フォルダーを表示します。
2. 除去したいデータベースを右クリックして、ポップアップ・メニューから**「除去 (Drop)」**を選択します。
3. **「確認 (Confirmation)」**ボックスをクリックして、**「Ok」**をクリックします。

コマンド行を使用してデータベースを除去するには、以下のように入力します。

```
DROP DATABASE <name>
```

次のコマンドは、SAMPLE というデータベースを削除するものです。

```
DROP DATABASE SAMPLE
```

注: SAMPLE データベースについての実験を続けるつもりであるなら、このデータベースは削除しないでください。 SAMPLE データベースを削除した後で再び必要であることが分かった場合には、これを再作成できます。

ノード・グループの変更

ノードグループの変更についての詳細は、 *管理の手引き: パフォーマンス* の『プロセッサーの追加による構成のスケーリング』を参照してください。

ノードを追加または除去したら、ノード・グループ内の新しいノードのセットにわたって、現行データを再分配しなければなりません。これを行うためには、REDISTRIBUTE NODEGROUP コマンドを使用します。詳しくは、*管理の手引き: パフォーマンス* の『データベース区画間でのデータの再配分』、および*コマンド解説書* を参照してください。

表スペースの変更

データベースを作成するときに、少なくとも 3 つの表スペースを作成します。それらの表スペースは、カタログ表スペース (SYSCATSPACE)、ユーザー表スペース (デフォルトの名前は USERSPACE1)、およびシステム一時表スペース (デフォルトの名前は TEMPSPACE1) です。これらの表スペースのそれぞれのうちの少なくとも 1 つは保持しなければなりません。必要なら、さらにユーザー表スペースと一時表スペースを余分に追加することができます。

注: カタログ表スペース SYSCATSPACE は除去できませんし、もう 1 つ作成することもできません。また、常に少なくとも 1 つのシステム一時表スペースが必要です。その他のシステム一時表スペースは作成できます。表スペースのページ・サイズやエクステント・サイズを、作成後に変更することもできません。

この節では、以下のように、表スペースの変更方法について説明します。

表スペースの設計情報については、*管理の手引き: 計画* を参照してください。

DMS 表スペースへのコンテナーの追加

DMS 表スペース (つまり、MANAGED BY DATABASE 文節を使用して作成されたもの) のサイズは、 1 つ以上のコンテナーを表スペースに追加することによって増やすことができます。

すべてのコンテナーにわたって表スペースの内容のバランスが再調整されます。バランスの再調整中も、表スペースへのアクセスは制限されません。複数のコンテナーを追加する必要がある場合は、それらを同時に追加しなければなりません。

コントロール・センターを使用して DMS 表スペースをコンテナーに追加するには、以下のようにします。

1. オブジェクト・ツリーを順に展開し、**「表スペース (Table Spaces)」**フォルダーを表示します。
2. コンテナーに追加したい表スペースを右クリックし、ポップアップ・メニューから**「更新 (Alter)」**を選択します。
3. **「追加 (Add)」**をクリックし、情報をすべて入力して、**「Ok」**をクリックします。
4. 表スペースが区分データベース環境にある場合に、表スペースのパフォーマンス・パラメーターを変更する必要があれば、**「拡張機能 (Advanced)」**をクリックします。
5. **「Ok」**をクリックします。

コマンド行を使用して DMS 表スペースをコンテナーに追加するには、以下のようにします。

```
ALTER TABLESPACE <name>
   ADD (DEVICE '<path>' <size>)
```

以下の例は、UNIX ベースのシステム上にある表スペースに対して、 2 つの新しいデバイス・コンテナー (それぞれが 10 000 ページのもの) を追加する方法を示したものです。

```
ALTER TABLESPACE RESOURCE
  ADD (DEVICE '/dev/rhd9'  10000,
       DEVICE  '/dev/rhd10' 10000)
```

ALTER TABLESPACE ステートメントを使用すると、パフォーマンスに影響を及ぼす可能性のある、表スペースの他の特性を変更することができます。詳しくは、*管理の手引き: パフォーマンス* の『照会最適化に対する表スペースの影響』を参照してください。

DMS 表スペース内のコンテナーの変更

DMS 表スペース中のコンテナー (つまり、 MANAGED BY DATABASE 文節を使用して作成されたもの) のサイズは、 1 つ以上のコンテナーをサイズ変更したり、表スペースに関連した 1 つ以上のコンテナーを拡張することによって増やすことができます。コンテナーのサイズの新しい上限がわかっている場合には、サイズ変更方式を考慮する必要があります。コンテナーの現在のサイズがわからない場合 (および検討していない場合) には、拡張方式を考慮する必要があります。

コマンド行を使用して DMS 表スペース中の 1 つ以上のコンテナーをサイズ変更するには、以下のようにします。

```
ALTER TABLESPACE <name>
   RESIZE (DEVICE '<path>' <size>)
```

以下の例は、UNIX ベースのシステム上の表スペース中にある、 2 つのデバイス・コンテナー (それぞれが 1 000 ページのもの) のサイズを増やす方法を示したものです。

```
ALTER TABLESPACE HISTORY
  RESIZE (DEVICE '/dev/rhd7'  2000,
          DEVICE  '/dev/rhd8' 2000)
```

このアクションを実行すると、 2 つのデバイスは 1 000 ページから 2 000 ページにサイズが増えます。新しいコンテナーを追加する場合と同様に、すべてのコンテナーにわたって表スペースの内容のバランスが再調整されます。バランスの再調整中も、表スペースへのアクセスは制限されません。

コマンド行を使用して DMS 表スペース中の 1 つ以上のコンテナーを拡張するには、以下のようにします。

```
ALTER TABLESPACE <name>
   EXTEND (DEVICE '<path>' <size>)
```

以下の例は、UNIX ベースのシステム上の表スペース中にある、 2 つのデバイス・コンテナー (それぞれが 1 000 ページのもの) のサイズを増やす方法を示したものです。

```
ALTER TABLESPACE HISTORY
  EXTEND (DEVICE '/dev/rhd11'  1000,
          DEVICE  '/dev/rhd12' 1000)
```

このアクションを実行すると、 2 つのデバイスは 1 000 ページから 2 000 ページにサイズが増えます。新しいコンテナーを追加する場合と同様に、すべてのコンテナーにわたって表スペースの内容のバランスが再調整されます。バランスの再調整中も、表スペースへのアクセスは制限されません。

表スペースの作成中または作成後に追加される DMS コンテナー (ファイルとロー・デバイス・コンテナーの両方)、あるいは表スペースの作成中または作成後に拡張される DMS コンテナーは、プリフェッチャーを使用して並列で実行されます。このような作成またはサイズ変更コンテナー操作の並列性を高めるために、システムで実行されるプ

リフェッチャーの数を増やすことができます。並列で行われないプロセスは、これらのアクションのロギングと、コンテナーの作成の場合は、コンテナーのタグ付けだけです。

注: CREATE TABLESPACE または ALTER TABLESPACE ステートメントの並列性 (新しいコンテナーの既存の表スペースへの追加に関して) を最大限にするには、プリフェッチャーの数が、追加されるコンテナーの数と等しいかそれ以上であることを確かめてください。

注: コンテナーのサイズを小さくすることはできません。

ALTER TABLESPACE ステートメントを使用すると、パフォーマンスに影響を及ぼす可能性のある、表スペースの他の特性を変更することができます。詳しくは、*管理の手引き: パフォーマンス* の『照会最適化に対する表スペースの影響』を参照してください。

区分内の SMS 表スペースへのコンテナーの追加

コマンド行を使用して、SMS 表スペースにコンテナーを追加するには、以下のように入力します。

```
ALTER TABLESPACE <name>
   ADD ('<path>')
   ON NODE (<partition_number>)
```

番号で指定された区分、および区分の範囲内のすべての区分 (もしくはノード) は、表スペースが定義されているノード・グループに存在していなくてはなりません。partition_number は、ステートメントのただ 1 つの *on-nodes-clause* で、明示的にのみ、または範囲で表される場合もあります。

以下の例では、UNIX ベースのオペレーティング・システム上で、表スペース『plans』が使用しているノード・グループの 3 番区分にどのように新規のコンテナーを追加するかを示しています。

```
ALTER TABLESPACE plans
   ADD ('/dev/rhdisk0')
   ON NODE (3)
```

表スペースの名前変更

既存の表スペースに、その中の個々のオブジェクトとは関係のない名前を新たに付けることができます。表スペースの名前を変更すると、その表スペースを参照するカタログ・レコードもすべて変更されます。

SYSCATSPACE 表スペースの名前を変更することはできません。

「ロールフォワード保留」または「ロールフォワード進行中」状態の表スペースの名前を変更することはできません。

バックアップを取った後に名前変更した表スペースを復元する際には、 RESTORE DATABASE コマンド中で新しい表スペースの名前を使用しなければなりません。変更前の表スペース名を使用しても、検出されません。同様に、 ROLLFORWARD DATABASE コマンドを使用して表スペースをロールフォワードする場合も、必ず新しい名前を使用するようにしてください。変更前の表スペース名を使用しても、検出されません。

表スペース状況の切り換え

表スペースに関連付けられたコンテナーがすでにアクセス可能になっている場合、ALTER TABLESPACE ステートメントの SWITCH ONLINE 文節を使用して、表スペースから OFFLINE 状態を除去できます。表スペースは、データベースがまだ起動し作動している間に、OFFLINE 状態を除去します。

この文節を使用する代わりに、データベースからすべてのアプリケーションを切断し、再びアプリケーションをデータベースに接続しなおすこともできます。これで、表スペースから OFFLINE 状態が除去されます。

コマンド行を使用して表スペースから OFFLINE 状態を除去するには、以下のように入力します。

```
ALTER TABLESPACE <name>
   SWITCH ONLINE
```

ユーザー表スペースの除去

ユーザー表スペースを除去するときには、その表スペース内のすべてのデータを削除し、コンテナーを解放し、カタログ項目を除去し、そして、その表スペースの中に定義されたすべてのオブジェクトを除去するか無効のマークを付けます。

表スペースを除去することによって、空の表スペース内のコンテナーを再使用することができますが、コンテナーを再使用する前に、 DROP TABLESPACE コマンドを COMMIT しなければなりません。

該当する単一ユーザー表スペース内の索引および LOB データを含むすべての表データが入ったユーザー表スペースを除去できます。また、いくつかの表スペースにわたる表を持つようなユーザー表スペースも除去できます。つまり、1 つの表スペースに表データ、別の表スペースに索引、 3 番目の表スペースに LOB を持つ場合があります。あるいは、単一のステートメントで 3 つの表スペースすべてを同時に除去する必要があります。 スパンされる表が入った表スペースはすべて、この単一ステートメントの一部にする必要があります。さもなければ、除去要求は失敗します。スパンされる表データが入った表スペースを除去する方法についての詳細は、 *SQL 解説書* を参照してください。

コントロール・センターを使用してユーザー表スペースを除去するには、以下のようにします。

1. オブジェクト・ツリーを順に展開し、**「表スペース (Table Spaces)」**フォルダーを表示します。
2. 除去したい表スペースを右クリックして、ポップアップ・メニューから**「除去 (Drop)」**を選択します。
3. **「確認 (Confirmation)」**ボックスにチェックを付け、**「Ok」**をクリックします。

コマンド行を使用してユーザー表スペースを除去するには、以下のように入力します。

```
DROP TABLESPACE <name>
```

以下の SQL ステートメントは、表スペース ACCOUNTING を除去します。

```
DROP TABLESPACE ACCOUNTING
```

システム一時表スペースの除去

データベースは常に少なくとも 1 つのシステム一時表スペースを持っていなければならないため、まずシステム一時表スペースをもう 1 つ作成してからでなければ、システム一時表スペースを除去することはできません。たとえば、SMS 一時表スペースにコンテナーを追加したければ、まず新しいシステム一時表スペースを追加してから、古いシステム一時表スペースを除去しなければなりません。

コントロール・センターを使用してシステム表スペースを除去するには、以下のようにします。

1. オブジェクト・ツリーを順に展開し、**「表スペース (Table Spaces)」**フォルダーを表示します。
2. 他にシステム一時表スペースが 1 つしかない場合は、**「表スペース (Table Spaces)」**フォルダーを右クリックして、ポップアップ・メニューから**「作成 (Create)」—>「ウィザードを使用して表スペースを (Table Spaces Using Wizard)」**を選択します。それ以外の場合は、スキップ 4 にスキップします。
3. このウィザードのステップに従って、必要に応じて新しいシステム一時表スペースを作成します。
4. **「表スペース (Table Spaces)」**フォルダーを再度クリックし、ウィンドウ (「内容 (Contents)」ペイン) の右側に表スペースのリストを表示します。
5. 除去したいシステム一時表スペースを右クリックして、ポップアップ・メニューから**「除去 (Drop)」**をクリックします。
6. **「確認 (Confirmation)」**ボックスにチェックを付け、**「Ok」**をクリックします。

システム一時表スペースが 1 つしかない場合は、その表スペースを削除する前に、システム一時表スペースをもう 1 つ作成しなければなりません。これは、コマンド行を使用し、以下のように入力すると実行できます。

```
CREATE SYSTEM TEMPORARY TABLESPACE <name>
   MANAGED BY SYSTEM USING ('<device>')
```

続いて、コマンド行を使用してシステム表スペースを除去するには、以下のように入力します。

```
DROP TABLESPACE <name>
```

以下の SQL は、 TEMPSPACE2 と呼ばれる新しいシステム一時表スペースを作成します。

```
CREATE SYSTEM TEMPORARY TABLESPACE TEMPSPACE2
   MANAGED BY SYSTEM USING ('d')
```

TEMPSPACE2 が作成されれば、以下のコマンドを使用して、元のシステム一時表スペース TEMPSPACE1 を除去することができます。

```
DROP TABLESPACE TEMPSPACE1
```

表スペースを除去することによって、空の表スペース内のコンテナーを再使用することができますが、コンテナーを再使用する前に、 DROP TABLESPACE コマンドを COMMIT しなければなりません。

ユーザー一時表スペースの除去

ユーザー一時表スペースを除去できるのは、その表スペースに定義されている、宣言済みの現行の表がない場合に限ります。表スペースを除去するときに、その表スペース内の宣言済み一時表の除去が試みられることはまったくありません。

注: 一時表を宣言しているアプリケーションがデータベースから切断されると、宣言された一時表は暗黙的に除去されます。

スキーマの除去

スキーマを除去する前に、そのスキーマの中にあったオブジェクト自体をすべて除去するか、別のスキーマに移動しなければなりません。 DROP ステートメントを試みているときには、スキーマ名がカタログの中になければなりません。そうでないと、エラーが戻されます。

コントロール・センターを使用してスキーマを除去するには、以下のようにします。

1. オブジェクト・ツリーを順に展開し、**「スキーマ (Schemas)」**フォルダーを表示します。
2. 除去したいスキーマを右クリックして、ポップアップ・メニューから**「除去 (Drop)」**を選択します。
3. **「確認 (Confirmation)」**ボックスにチェックを付け、**「Ok」**をクリックします。

コマンド行を使用してスキーマを除去するには、以下のように入力します。

```
DROP SCHEMA <name>
```

以下の例では、"joeschma" というスキーマが除去されます。

```
DROP SCHEMA joeschma RESTRICT
```

RESTRICT キーワードは、データベースから削除するよう指定されたスキーマにはオブジェクトを定義できないという規則を、強制的に実行します。

構造と内容の両方における表の修正

表の構造と内容を修正するために必要なタスクには、次の事柄が含まれます。

- 『既存の表への列の追加』
- 199ページの『列定義の修正』
- 200ページの『表または視点からの行の除去』
- 201ページの『制約の変更』
- 206ページの『既存の表に生成列を定義する』
- 209ページの『表の揮発性を宣言する』
- 210ページの『区分化キーの変更』
- 211ページの『表属性の変更』
- 214ページの『要約表のデータの最新表示』

表に対するトリガーは変更できないことに注意してください。適切でなくなったトリガーを除去して (217ページの『トリガーの除去』を参照)、その代わりのものを追加 (143ページの『トリガーの作成』を参照) しなければなりません。

既存の表への列の追加

列定義には、列名、データ・タイプ、および必要な制約が含まれます。

表に列が追加されると、列は右端の既存の列定義の右側に論理的に配置されます。既存の表に新しい列を追加する場合、システム・カタログの表記述を修正するだけなので、表へのアクセス時間はすぐには影響を受けません。 UPDATE ステートメントによって修正される時点まで、既存のレコードが物理的に変更されることはありません。表から既存の行を取り出すと、新しい列には、列の定義にしたがってヌル値かデフォルト値かが入れられます。表が作成された後に追加された列は、NOT NULL として定義することはできません。 NOT NULL WITH DEFAULT として、またはヌル値可能としてのいずれかで定義しなければなりません。

コントロール・センターを使用して既存の表に列を追加するには、以下のようにします。

1. オブジェクト・ツリーを順に展開し、**「表 (Tables)」**フォルダーを表示します。
2. 列を追加したい表を右クリックし、ポップアップ・メニューから**「更新 (Alter)」**を選択します。
3. **「列 (Columns)」**ページを検査して、列に関する情報をすべて入力し、**「Ok」**をクリックします。

コマンド行を使用して既存の表に列を追加するには、以下のようにします。

```
ALTER TABLE <table_name>
   ADD <column_name> <data_type> <null_attribute>
```

列の追加は、SQL ステートメントを使ってもできます。次のステートメントは、ALTER TABLE ステートメントを使って、 EMPLOYEE 表に 3 つの列を追加するものです。

```
ALTER TABLE EMPLOYEE
   ADD MIDINIT CHAR(1)   NOT NULL WITH DEFAULT
   ADD HIREDATE DATE
   ADD WORKDEPT CHAR(3)
```

列定義の修正

既存の VARCHAR 列の長さを長くすることによって、列の特性を修正できます。文字数は、使用されるページ・サイズに従属する値まで増やすことができます。

コントロール・センターを使用して既存の表の列を修正するには、以下のようにします。

1. オブジェクト・ツリーを順に展開し、**「表 (Tables)」**フォルダーを表示します。
2. 右側のペイン内にある表のリストから、修正したい列を含む表を右クリックし、ポップアップ・メニューから**「更新 (Alter)」**を選択します。
3. **「列 (Columns)」**ページを検査して、列を選択し、**「変更 (Change)」**をクリックします。
4. 列の新しいバイト・カウントを**「長さ (Length)」**に入力し、**「Ok」**をクリックします。

コマンド行を使用して既存の表の列を修正するには、以下のように入力します。

```
ALTER TABLE ALTER COLUMN
   <column_name> <modification_type>
```

たとえば、4000 文字まで増やすには、次のような形式で指定します。

```
ALTER TABLE ALTER COLUMN
   COLNAM1 SET DATA TYPE VARCHAR(4000)
```

タイプ付き表の列を変更することはできません。しかし、効力範囲がまだ定義されていない既存の参照タイプ列に、効力範囲を追加することは可能です。たとえば、次のようにします。

```
ALTER TABLE ALTER COLUMN
   COLNAMT1 ADD SCOPE TYPTAB1
```

ALTER TABLE ステートメントについての詳細は、*SQL 解説書* を参照してください。

表または視点からの行の除去

行を削除することにより、表または視点の内容を変更できます。視点から行を削除すると、視点の基本となっている表から行が削除されます。 DELETE ステートメントを使用して、次のことを行います。

- 検索条件によりオプションで判別されている 1 つ以上の行を削除する。これを*検索済み DELETE* と言います。
- カーソルの現在位置により判別されている 1 行だけを削除する。これを*位置決め DELETE* と言います。

DELETE 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ができ、また動的 SQL ステートメントとして出すことができます。

変更される表が参照制約によって他の表に関連している場合、行の削除を実行する際の考慮事項があります。識別された表、または識別された視点の基礎表が親である場合、削除するように選択した行は RESTRICT 削除規則に関連した従属関係を持っていてはなりません。さらに、DELETE は、RESTRICT 削除規則に関連した従属関係を持つ下層行にカスケードしてはなりません。

削除操作が RESTRICT 削除規則によって妨げられない場合、選択した行が削除されます。選択した行の下層行に発生する事柄についてのさらに詳しい情報は、 *SQL 解説書* を参照してください。

たとえば、表 (DEPARTMENT) から部門 (DEPTNO) 『D11』 を削除するには、以下を使用します。

```
DELETE FROM department WHERE deptno='D11'
```

複数行の DELETE の実行中にエラーが発生する場合、表は変更されません。エラーの発生により、検索条件に一致するすべての行を削除できず、既存の参照制約で必要とされるすべての操作が妨げられる場合、表は変更されません。

適切なロックが存在していない場合、正常な DELETE ステートメントの実行時に、 1 つ以上の排他ロックが獲得されます。 COMMIT または ROLLBACK ステートメントの後に、ロックが解放されます。ロックにより、他のアプリケーションは表に対して操作を実行できなくなります。

識別列定義の変更

表を再作成した後でインポートまたはロード操作が実行される場合、および表に IDENTITY 列がある場合、表の内容を再作成した後で、 IDENTITY 値を 1 から生成するようにリセットされます。再作成された表に新規行を挿入する際に、 IDENTITY 列を再び 1 から始めたり、 IDENTITY 列に重複値を含めたりしないで済むようにするには、以下のようにしてください。

1. 表を再作成します。
2. MODIFIED BY IDENTITYOVERRIDE 文節を使用してデータを表にロードします。データは表にロードされますが、行の識別値は生成されません。
3. 次の照会を実行して、IDENTITY 列の最新のカウンター値を入手します。

   ```
   SELECT MAX(<IDENTITY column>)
   ```

   これで、表の IDENTITY 列値であったものと等しい値が戻されます。
4. ALTER TABLE ステートメントの RESTART 文節を使用します。

   ```
   ALTER TABLE <table name> ALTER COLUMN <IDENTITY column>
      RESTART WITH <last counter value>
   ```
5. 新規行を表に挿入します。 RESTART WITH 文節で指定された値に基づいて、IDENTITY 列値が生成されます。

制約の変更

制約は、それらを除去した後、代わりに新しいものを追加することによってのみ変更できます。詳細については、以下の部分を参照してください。

- 『制約の追加』
- 204ページの『制約の除去』

制約について詳しくは、130ページの『制約の定義』を参照してください。

制約の追加

ALTER TABLE ステートメントを使用して制約を追加します。このステートメント (構文を含む) についての詳細は、 *SQL 解説書* を参照してください。

制約について詳しくは、130ページの『制約の定義』を参照してください。

固有制約の追加: 固有制約を既存の表に追加することができます。制約名は、ALTER TABLE ステートメント内に指定した他のどの制約とも同じにすることはできず、その表内で固有なものでなければなりません (これには、定義されたすべての参照保全制約の名前も含まれます)。ステートメントの完了前に、既存のデータが新しい条件に照らして検査されます。

以下の SQL ステートメントは、表内の従業員を固有なものとして識別するための新しい方法を表す、 EMPLOYEE 表に対する固有制約を追加します。

```
ALTER TABLE EMPLOYEE
   ADD CONSTRAINT NEWID UNIQUE(EMPNO,HIREDATE)
```

基本キーと外部キーの追加: 大きな表に制約を追加する場合は、まず表を検査保留状態にし、制約を追加してから、表を検査して違反行の統合リストを求めるのが効率的です。明示的に検査保留状態を設定するには、SET INTEGRITY ステートメントを使います。表が親表の場合は、すべての従属表および子孫表に対しても暗黙のうちに検査保留が設定されます。

コントロール・センターを使用して基本キーを追加するには、以下のようにします。

1. オブジェクト・ツリーを順に展開し、**「表 (Tables)」**フォルダーを表示します。
2. 修正したい表を右クリックし、ポップアップ・メニューから**「更新 (Alter)」**を選択します。
3. **「基本キー (Primary Key)」**ページで、 1 つまたは複数の列を基本キーとして選択し、矢印をクリックして移動します。
4. オプション: 基本キーの制約名を入力します。
5. **「Ok」**をクリックします。

コマンド行を使用して基本キーを追加するには、以下のように入力します。

```
ALTER TABLE <name>
   ADD CONSTRAINT <column_name>
   PRIMARY KEY <column_name>
```

外部キーが表に追加される場合、以下のステートメントが含まれるパッケージまたはキャッシュに入った動的 SQL は、無効のマークが付けられます。

- 外部キーが入っている表の挿入または更新を行うステートメント
- 親表の更新または削除を行うステートメント

詳しくは、225ページの『オブジェクトを変更する場合のステートメントの従属関係』を参照してください。

コントロール・センターを使用して外部キーを追加するには、以下のようにします。

1. オブジェクト・ツリーを順に展開し、**「表 (Tables)」**フォルダーを表示します。
2. 修正したい表を右クリックし、ポップアップ・メニューから**「更新 (Alter)」**を選択します。
3. **「外部キー (Foreign Keys)」**ページで、**「追加 (Add)」**をクリックします。
4. **「外部キーの追加 (Add Foreign Keys)」**ウィンドウで、親表の情報を指定します。
5. 1 つまたは複数の列を外部キーとして選択し、矢印をクリックして移動します。
6. 親表の行が削除されたり更新されたりした場合に従属表に対して行うアクションを指定します。外部キーの制約名を追加することもできます。
7. **「Ok」**をクリックします。

コマンド行を使用して外部キーを追加するには、以下のように入力します。

```
ALTER TABLE <name>
   ADD CONSTRAINT <column_name>
    FOREIGN KEY <column_name>
    ON DELETE <action_type>
    ON UPDATE <action_type>
```

以下の例は、表に基本キーと外部キーを追加するための ALTER TABLE ステートメントを示しています。

```
ALTER TABLE PROJECT
  ADD CONSTRAINT PROJECT_KEY
      PRIMARY KEY (PROJNO)
ALTER TABLE EMP_ACT
  ADD CONSTRAINT ACTIVITY_KEY
      PRIMARY KEY (EMPNO, PROJNO, ACTNO)
  ADD CONSTRAINT ACT_EMP_REF
      FOREIGN KEY (EMPNO)
      REFERENCES EMPLOYEE
      ON DELETE  RESTRICT
  ADD CONSTRAINT ACT_PROJ_REF
      FOREIGN KEY (PROJNO)
      REFERENCES PROJECT
      ON DELETE CASCADE
```

表検査制約の追加: 検査制約は、ALTER TABLE ステートメントを使用して既存の表に追加することができます。制約名は、ALTER TABLE ステートメント内に指定した他のどの制約とも同じにすることはできず、その表内で固有なものでなければなりません (これには、定義されたすべての参照保全制約の名前も含まれます)。ステートメントの完了前に、既存のデータが新しい条件に照らして検査されます。

大きな表に制約を追加するには、表を検査保留状態にし、制約を追加してから、表を検査して制約に違反する行の統合リストを求めるのが効率的です。明示して検査保留状態

を設定するには、SET INTEGRITY ステートメントを使用します。表が親表である場合、検査保留は、すべての従属表と子孫表にも暗黙に設定されます。

表検査制約が追加された場合、表の挿入または更新を行うパッケージおよびキャッシュに入った動的 SQL は、無効のマークが付けられます。詳しくは、225ページの『オブジェクトを変更する場合のステートメントの従属関係』を参照してください。

コントロール・センターを使用して表検査制約を追加するには、以下のようにします。

1. オブジェクト・ツリーを順に展開し、**「表 (Tables)」**フォルダーを表示します。
2. 修正したい表を右クリックし、ポップアップ・メニューから**「更新 (Alter)」**を選択します。
3. **「検査制約 (Check Constraints)」**ページで、**「追加 (Add)」**をクリックします。
4. **「検査制約の追加 (Add Check Constraint)」**ウィンドウで、情報をすべて入力して、**「Ok」**をクリックします。
5. **「検査制約 (Check Constraints)」**ページで、**「Ok」**をクリックします。

コマンド行を使用して表検査制約を追加するには、以下のように入力します。

```
ALTER TABLE <name>
   ADD CONSTRAINT <name> (<constraint>)
```

次の SQL ステートメントは、各従業員の給与と歩合給の合計が $25,000 を超えなければならないという制約を EMPLOYEE 表に追加するものです。

```
ALTER TABLE EMPLOYEE
   ADD CONSTRAINT REVENUE CHECK (SALARY + COMM > 25000)
```

制約の除去

ALTER TABLE ステートメントを使用して制約を除去します。このステートメント (構文を含む) についての詳細は、 *SQL 解説書* を参照してください。

制約について詳しくは、130ページの『制約の定義』を参照してください。

固有制約の除去: ALTER TABLE ステートメントを使用して、明示的に固有制約を除去することができます。表上のすべての固有制約の名前は、SYSCAT.INDEXES システム・カタログ視点の中にあります。

以下の SQL ステートメントは、EMPLOYEE 表から固有制約 NEWID を除去します。

```
ALTER TABLE EMPLOYEE
   DROP UNIQUE NEWID
```

この固有制約を除去すると、その制約を使用しているすべてのパッケージまたはキャッシュに入った動的 SQL を無効にします。

基本キーと外部キーの除去: コントロール・センターを使用して基本キーを除去するには、以下のようにします。

1. オブジェクト・ツリーを順に展開し、**「表 (Tables)」**フォルダーを表示します。
2. 修正したい表を右クリックし、ポップアップ・メニューから**「更新 (Alter)」**を選択します。
3. **「基本キー (Primary Key)」**ページの右側で、除去する基本キーを選択し、矢印をクリックして左側の**「使用可能な列 (Available columns)」**ボックスに移動します。
4. **「Ok」**をクリックします。

コマンド行を使用して基本キーを除去するには、以下のように入力します。

```
ALTER TABLE <name>
  DROP PRIMARY KEY
```

外部キーが除去されると、以下のものを含むパッケージまたはキャッシュに入った動的 SQL ステートメントは、無効のマークが付けられます。

- 外部キーが入っている表の挿入または更新を行うステートメント
- 親表の更新または削除を行うステートメント

詳細は、225ページの『オブジェクトを変更する場合のステートメントの従属関係』を参照してください。

コントロール・センターを使用して外部キーを除去するには、以下のようにします。

1. オブジェクト・ツリーを順に展開し、**「表 (Tables)」**フォルダーを表示します。
2. 修正したい表を右クリックし、ポップアップ・メニューから**「更新 (Alter)」**を選択します。
3. **「外部キー (Foreign Keys)」**ページで、**「追加 (Add)」**をクリックします。
4. 右側で除去する外部キーを選択し、矢印をクリックして左側の**「使用可能な列 (Available columns)」**ボックスに移動します。
5. **「外部キー (Foreign Keys)」**ページで、**「Ok」**をクリックします。

コマンド行を使用して外部キーを除去するには、以下のように入力します。

```
ALTER TABLE <name>
   DROP FOREIGN KEY <foreign_key_name>
```

以下の例は、 ALTER TABLE ステートメントの DROP PRIMARY KEY および DROP FOREIGN KEY 文節を使用して、表の基本キーと外部キーを除去します。

```
ALTER TABLE EMP_ACT
  DROP PRIMARY KEY
  DROP FOREIGN KEY ACT_EMP_REF
  DROP FOREIGN KEY ACT_PROJ_REF
ALTER TABLE PROJECT
  DROP PRIMARY KEY
```

ALTER TABLE ステートメントについての詳細は、*SQL 解説書* を参照してください。

表検査制約の除去: 表検査制約を明示的に除去または変更するには、ALTER TABLE ステートメントを使います。また、DROP TABLE ステートメントを使うと、暗黙のうちに検査制約が除去されます。

表検査制約を除去すると、その表上で INSERT または UPDATE 依存関係を持つすべてのパッケージおよびキャッシュに入った動的 SQL ステートメントは、無効にされます。 (詳しくは、225ページの『オブジェクトを変更する場合のステートメントの従属関係』を参照してください。) 表内のすべての検査制約の名前は、SYSCAT.CHECKS カタログ視点の中にあります。名前がシステム生成である表検査制約の除去を試行する前に、 SYSCAT.CHECKS カタログ視点でその名前を探してください。

コントロール・センターを使用して表検査制約を除去するには、以下のようにします。

1. オブジェクト・ツリーを順に展開し、**「表 (Tables)」**フォルダーを表示します。
2. 修正したい表を右クリックし、ポップアップ・メニューから**「更新 (Alter)」**を選択します。
3. **「検査制約 (Check Constraints)」**ページで、除去する検査制約を選択し、**「除去 (Remove)」**をクリックして、**「Ok」**をクリックします。

コマンド行を使用して表検査制約を除去するには、以下のようにします。

```
ALTER TABLE <table_name>
   DROP CHECK <check_constraint_name>
```

次の SQL ステートメントは、 EMPLOYEE 表から REVENUE という表検査制約を除去するものです。

```
ALTER TABLE EMPLOYEE
   DROP CHECK REVENUE
```

既存の表に生成列を定義する

生成列は基礎表に定義されます。生成列に格納される値は、式を使って計算された値です。挿入操作や更新操作で指定された値ではありません。生成列は、表を作成する時、または既存の表を修正する時に作成できます。

生成列を定義するには、以下のステップを実行します。

1. 表を検査保留状態にします。

```
SET INTEGRITY FOR t1 OFF
```

2. 表を更新して、1 つまたは複数の生成列を追加します。

```
ALTER TABLE t1 ADD COLUMN c3 DOUBLE GENERATED ALWAYS AS (c1 + c2),
               ADD COLUMN c4 GENERATED ALWAYS AS
                  (CASE WHEN c1 > c3 THEN 1 ELSE NULL END))
```

3. この時点で、このタスクを完了するための方法は、表に対して実行する作業に応じて複数あります。
   - 表が非常に大きく、タスクを完了できるだけの大きさのログ・スペースがあるかどうか分からない場合。データをロードしてから、保全性検査を有効に戻すまでの間に、以下のコマンドを実行する必要があります。

     ```
     COMMIT
     ```

     続いて、db2gncol ユーティリティーを使用し、生成列の値を設定する必要があります。このユーティリティーは、 sqllib ディレクトリー中の bin サブディレクトリーにあります。以下のようにこのユーティリティーを使用します。

     ```
     db2gncol -d <dbname> -s <schema> -t <table_name>
              -c <commitcount>
     ```

     dbname は、表のあるデータベースの別名を指定します。 schema は、表のスキーマ名を指定し、大文字小文字の区別があります。 table_name は表を指定します。その表の列のために、新しい値が式で計算して生成されます。 schema と table_name は両方とも大文字小文字の区別があります。 commitcount は、ログをクリーンアップするための内部コミットから、次の内部コミットまでの間に処理される行数です。このパラメーターは、列値の生成を実行するのに必要なログ・スペースのサイズに影響します。

     前述の例には示されていないオプション・パラメーターが 2 つあります。それは -u <username> と -p <password> で、ユーザーとパスワードを識別します。ユーザーには、SYSADM または DBADM 権限が必要です。ユーザーとパスワードを識別しないと、現行のユーザー ID が使用されます。

     このユーティリティーに関するヘルプ情報が必要な場合は、以下のように入力します。

     ```
     db2gncol -h
     ```

     ヘルプ・パラメーターを使用すると、他のパラメーターはすべて無視されます。

     表は、検査保留状態にある場合でも、プロセス全体のためにロックされます。その理由は、検査保留状態の表にアクセスできる他のユーティリティーがあるからです。ロックにより、この種のユーティリティーと競合しないようにされます。
   - 生成列を更新するためのログ・スペースが、 SET INTEGRITY を実行するのに十分の大きさであると予想される場合。通常はこれがあてはまります。データをロードした後に、以下を使用して生成列の値を再度計算して割り当てます。

```
SET INTEGRITY FOR t1 IMMEDIATE CHECKED
   FORCE GENERATED
```

注: この時点で例外表を使用できます。

- 表が非常に大きく、タスクを完了できるだけの大きさのログ・スペースがあるかどうか分からないものの、最初に挙げた方式を選択しない場合。データをロードしてから、保全性検査を有効に戻すまでの間に、以下のコマンドを実行する必要があります。
  a. 表の排他ロックを取得します。そうすると、非コミット読み取り以外のトランザクションは表にアクセスできなくなります。

     ```
     LOCK TABLE t1
     ```

  b. データを検査せずに、表をオンライン状態にします。

     ```
     SET INTEGRITY FOR t1 ALL IMMEDIATE UNCHECKED
     ```

  c. ログが満杯にならないよう、非再現性コミットと述部を使用して生成列を更新します。

     ```
     UPDATE t1 SET (c3, c4) = (DEFAULT, DEFAULT) WHERE <predicate>
     ```

  d. 表をオンラインにして、保全性の検査を行います。

     ```
     SET INTEGRITY FOR t1 OFF
     SET INTEGRITY FOR t1 IMMEDIATE CHECKED
     ```

  e. コミット・ステートメントを使用してトランザクションを完了し、表をロック解除します。

     ```
     COMMIT
     ```

- NOT LOGGED INITIALLY オプションを指定して表が作成されたことが分かっている場合。この場合、表に関するログ記録は無効になっています。これには、生成列値の処理中に、通常の影響やリスクが伴います。
  a. NOT LOGGED INITIALLY オプションをアクティブにします。

     ```
     ALTER TABLE t1 ACTIVATE NOT LOGGED INITIALLY
     ```

  b. 値を生成します。

     ```
     SET INTEGRITY FOR t1 IMMEDIATE CHECKED FORCE GENERATED
     ```

  c. トランザクションをコミットして、 NOT LOGGED INITIALLY オプションを再度無効にします。

     ```
     COMMIT
     ```

等価性の検査制約のように、式を適用することによって、生成列の値に単純な検査を加えることもできます。

```
SET INTEGRITY FOR t1 IMMEDIATE CHECKED
```

生成列に (たとえば LOAD を使用して) 値を入れ、かつその値が、生成された式に一致することが分かっている場合、値の検査または割り当てを行わずに、表を検査保留状態から解除することができます。

```
SET INTEGRITY FOR t1 GENERATED COLUMN IMMEDIATE UNCHECKED
```

生成列を定義できるのは、等価性の比較が定義されているデータ・タイプだけです。生成列を定義できないデータ・タイプとしては、構造型、LOB、CLOB、DBCLOB、LONG VARCHAR、LONG VARGRAPHIC、およびこれらのデータ・タイプを使用して定義したユーザー定義のタイプがあります。

制約、固有索引、参照制約、基本キー、およびグローバル一時表には生成列を使用できません。 LIKE を使用して作成した表と、具体化した視点は、生成列の特性を継承しません。

キーワード DEFAULT を指定しないと、生成列を挿入したり更新したりできません。挿入の際に DEFAULT を使用すると、列リストに列を列挙する必要がなくなります。代わりに値リストの中で生成列を DEFAULT に設定できます。更新の際に DEFAULT を使用すると、 SET INTEGRITY によって検査抜きでオンラインで配置された生成列を再計算できます。

トリガー処理の順序の関係上、BEFORE トリガーのヘッダー (更新前) または本体が、生成列を参照するようになっていてはなりません。処理の順序としては、生成列は BEFORE トリガーの後に処理されます。

db2look ユーティリティーは、生成列によって生成される検査制約を考慮に入れません。

複製を使用する際には、ターゲット表のマッピングに生成列を使用してはなりません。複製する際には 2 つの選択肢があります。

- ターゲット表に、正規の (つまり、生成列ではない) 列として生成列が定義されていなければならない。
- ターゲット表のマッピングから生成列を省略しなければならない。

生成列を処理する際には複数の制約事項があります。

- 生成列同士は従属関係にあってはならない。
- 生成列の作成に使用する式に、副照会を含めることはできない。これには、READS SQL DATA を実行する関数を使用する式も含まれます。
- 生成列には検査制約は使用できない。
- 生成列には固有索引は使用できない。これには、固有制約と基本キーが含まれます。

表の揮発性を宣言する

揮発性 表は、実行時にその内容が空であるものから内容が非常に大きなものに至るまで、さまざまな内容を持つ表として定義されます。この種の表には揮発性、つまり極端

| な変更の可能性があるため、 RUNSTATS が収集する統計の信頼性が不確かなものになります。 統計は一時点で収集され、表面的なものにすぎません。揮発性表を使用するアクセス・プランを生成しても、実行プランが不正確または貧弱なものになる可能性があります。 たとえば、揮発性表が空のときに統計を収集すると、最適化プログラムの傾向として、索引走査ではなく表走査を使って揮発性表にアクセスするほうが優先されます。

この傾向を回避するには、ALTER TABLE ステートメントを使って、表を揮発性として宣言することを考慮してください。表を揮発性として宣言すれば、最適化プログラムは表走査ではなく索引走査の使用を考慮します。宣言された揮発性表を使用するアクセス・プランは、該当する表の既存統計には依存しません。

コントロール・センターを使用して表の揮発性を宣言するには、以下のようにします。

1. オブジェクト・ツリーを順に展開し、**「表 (Tables)」**フォルダーを表示します。
2. 修正したい表を右クリックし、ポップアップ・メニューから**「更新 (Alter)」**を選択します。
3. **「表 (Table)」**ページで、**「実行時にカーディナリティーが大幅に変わる (Cardinality varies significantly at run time)」**チェック・ボックスを選択し、**「Ok」**をクリックします。

| コマンド行を使用して、表を 『揮発性』 として宣言するには、以下のように入力します。

```
ALTER TABLE <table_name>
    VOLATILE CARDINALITY
```

区分化キーの変更

単一区画ノード・グループ内の表の区分化キーのみを変更することができます。まず既存の区分化キーを除去してから、別の区分化キーを作成してください。

以下の SQL ステートメントは、MIXREC 表から区分化キー MIX_INT を除去します。

```
ALTER TABLE MIXREC
   DROP PARTITIONING KEY
```

詳しくは、*SQL 解説書* の ALTER TABLE ステートメントの説明を参照してください。

| 複数区画ノード・グループの中の表の区分化キーは、変更できません。この区分化キーを除去しようとすると、エラーが戻されます。

| 複数区画ノード・グループの区分化キーを変更するには、以下のどちらかを実行します。

- すべてのデータを単一区画ノード・グループにエクスポートしてから、上記の指示に従う。
- すべてのデータをエクスポートして、表を除去し、区分化キーを再定義して表を再作成してから、すべてのデータをインポートする。

これらの方法はいずれも、大きなデータベースの場合は現実的ではありません。このため、大きなデータベースの設計を実現する前に、適切な区分化キーを定義しておくことがもっとも重要です。

表属性の変更

データ取り込みオプション、各ページでの空きスペースのパーセンテージ (PCTFREE)、ロック・サイズ、または追加モードなどの表属性を変更する必要があるかもしれません。

表の各ページに残す空きスペースの量は、PCTFREE によって指定しますが、これはクラスター索引を効果的に使用するための重要な考慮事項です。指定する量は、既存のデータと予想される将来のデータの性質によって決まります。 PCTFREE の指定は、LOAD と REORG については有効ですが、挿入、更新、およびインポート活動では無視されます。

PCTFREE を大きな値に設定すると、クラスター化を長い期間保つことができますが、ディスク・スペースもより多く必要となります。

LOCKSIZE パラメーターを使用して、表がアクセスされるときに使用されるロックのサイズ (細分性) を指定できます。表が作成されるとき、デフォルトでは行レベルのロックが定義されます。表レベルのロックを使用すると、獲得し解放する必要のあるロックの数を限定することによって、照会のパフォーマンスが向上します。

APPEND ON を指定すると、全体の表のパフォーマンスを向上させることができます。これによって、より速い挿入が可能となり、一方では空きスペースについての情報が除去されます。

クラスター化索引のある表は、追加モードをオンにするように変更することはできません。同様に、追加モードの表でクラスター化索引を作成することはできません。

識別列の変更

ALTER TABLE ステートメントで、既存の識別列の属性を変更します。このステートメントの詳細 (構文など) については、*SQL 解説書* を参照してください。

シーケンスの特性を持つように識別列を変更するためには、いくつかの方法があります。

ALTER TABLE および識別列には、以下のような固有のタスクがあります。

- RESTART は、識別列に関連付けられたシーケンスを、識別列が最初に作成されたときに明示的または暗黙的に指定された値にリセットします。
- RESTART WITH <numeric-constant> は、識別列に関連付けられたシーケンスを数値定数にリセットします。数値定数は、識別列に割り当てることができる、小数点以下に非ゼロ桁がない正または負の値です。

シーケンスの変更

ALTER SEQUENCE ステートメントで、既存のシーケンスの属性を変更します。このステートメントの詳細 (構文など) については、*SQL 解説書* を参照してください。

変更可能なシーケンスの属性

- 今後の値の間の増分を変更
- 新しい最小値または最大値を確立
- キャッシュ済みシーケンス番号の数を変更
- シーケンスが循環するかどうかを変更
- 要求の順序でシーケンス番号が生成されるかどうかを変更
- シーケンスを再始動

シーケンス作成の一部ではないタスクが 2 つあります。追加される値は以下の通りです。

- RESTART。シーケンスを、そのシーケンスの作成時に開始値として明示的または暗黙的に指定された値にリセットします。
- RESTART WITH numeric-constant。シーケンスを数値定数にリセットします。数値定数は、小数点以下に非ゼロ桁がない正または負の値です。

シーケンスを再始動、または CYCLE に変更した後、重複するシーケンス番号が生成される可能性があります。今後のシーケンス番号だけが ALTER SEQUENCE ステートメントによって影響を受けます。

シーケンスのデータ・タイプは変更できません。その代わり、現行シーケンスをドロップして、新しいデータ・タイプを指定した新しいシーケンスを作成する必要があります。

シーケンスが変更されると、DB2 で使用されていないキャッシュ済みの値はすべて失われます。

シーケンスのドロップ

シーケンスを削除するには、DROP ステートメントを使用します。このステートメントの詳細 (構文など) については、*SQL 解説書* を参照してください。

以下を実行すると、特定のシーケンスをドロップすることができます。

```
DROP SEQUENCE sequence_name
```

sequence_name はドロップするシーケンス名で、ここには、既存のシーケンスを正しく識別するための暗黙的または明示的なスキーマ名が入ります。

IDENTITY 列のためにシステム作成されたシーケンスは、DROP SEQUENCE ステートメントを使用してドロップすることはできません。

シーケンスをドロップすると、そのシーケンスの特権もすべてドロップされます。

要約表の特性の変更

多少の制約事項はありますが、要約表を正規表に変更したり、正規表を要約表に変更したりできます。他の表タイプは変更できません。変更できるのは正規表と要約表だけです。たとえば、複製要約表から正規表に (あるいはその逆に) 変更することはできません。

正規表を要約表に変更すると、その表は検査保留状態になります。このように変更した場合、要約表の定義を全選択したものは変更前の表の定義と一致していなければなりません。つまり、以下の点を満たしていなければなりません。

- 列の数が同じでなければならない。
- 列名と位置が一致しなければならない。
- データ・タイプが同一でなければならない。

要約表が元の表で定義されている場合、元の表自体を変更して要約表にすることはできません。元の表にトリガー、検査制約、参照制約、または定義済みの固有索引がある場合、その表を要約表に変更することはできません。要約表を定義するために表の特性を変更する場合、同じ ALTER TABLE ステートメントを使って、別の方法で表を変更することはできません。

正規表を要約表に変更する場合、要約表の定義の全選択では、元の表を直接、あるいは視点、別名、または要約表を介して間接的に参照することはできません。

要約表を正規表に変更するには、以下のようにします。

```
ALTER TABLE sumtable
   SET SUMMARY AS DEFINITION ONLY
```

正規表を要約表に変更するには、以下のようにします。

```
ALTER TABLE regtable
   SET SUMMARY AS <fullselect>
```

正規表を要約表に変更する際の全選択に関する制約事項は、 CREATE SUMMARY TABLE ステートメントを使用して要約表を作成する際の制約事項に大変類似しています。

CREATE SUMMARY TABLE ステートメントの詳細については、 *SQL 解説書* を参照してください。

要約表のデータの最新表示

REFRESH TABLE ステートメントを使用して、 1 つまたは複数の要約表のデータを最新表示できます。このステートメントは、アプリケーション・プログラムに組み込むこともできますし、動的に出すこともできます。このステートメントを使用するには、SYSADM または DBADM 権限か、更新する表に対する CONTROL 特権が必要です。

以下の例は、要約表のデータを最新表示する方法を示したものです。

```
REFRESH TABLE SUMTAB1
```

REFRESH TABLE ステートメントの詳細については、*SQL 解説書* を参照してください。

ユーザー定義構造型の変更

構造型の作成後に、その構造型に関連した属性を追加または削除しなければならないことに気付くことがあるでしょう。これは、ALTER TYPE (構造化) ステートメントを使用して行われます。構造型に関する必要なすべての情報については、 *アプリケーション開発の手引き* を参照してください。

タイプ付き表の行の削除および更新

検索済み DELETE ステートメントか、位置決め済み DELETE ステートメントのいずれかを使用して、行をタイプ付き表から削除できます。検索済み UPDATE ステートメントか、位置決め済み UPDATE ステートメントのいずれかを使用して、タイプ付き表の行を更新できます。タイプ付き表に関する必要なすべての情報については、 *アプリケーション開発の手引き* を参照してください。

既存の表の名前変更

スキーマ内の既存の表に新しい名前を与え、オリジナルの表に作成されていた許可と索引はそのまま維持することができます。

名前変更する既存の表は、表を識別する別名であっても構いません。名前変更する既存の表は、カタログ表、要約表、タイプ付き表、または表か別名以外のオブジェクトの名前であってはなりません。

既存の表は、以下のいずれでも参照できません。

- 視点
- トリガー
- 参照制約
- 要約表
- 既存の参照列の効力範囲

また、表内に検査制約があったり、識別列以外の生成列があったりしてはなりません。オリジナルの表に依存するパッケージまたはキャッシュに入った動的 SQL ステートメントは、すべて無効にされます。オリジナルの表を参照している別名は修正されません。

名前変更している表が、これらの制約事項のいずれの影響も受けないようにするために、該当のシステム・カタログ表を検査することを考慮する必要があります。

名前が変更されたばかりの表をパッケージが参照する場合は、そのパッケージを再バインドしなければなりません。以下のいずれかの場合には、パッケージは暗黙に再バインドすることができます。

- 別の表が、その表のオリジナルの名前を使用して名前変更されている。
- その表のオリジナルの名前を使用して別名または視点が作成されている。

暗黙または明示の再バインドを試みる前に、上記の 2 つのいずれかを選択していなければなりません。どちらも選択していないと、再バインドは失敗します。

コントロール・センターを使用して既存の表の名前を変更するには、以下のようにします。

1. オブジェクト・ツリーを順に展開し、**「表 (Tables)」**フォルダーを表示します。
2. 名前変更したい表を右クリックし、ポップアップ・メニューから**「名前変更 (Rename)」**を選択します。
3. 新しい表名を入力して、**「Ok」**をクリックします。

コマンド行を使用して既存の表の名前を変更するには、以下のように入力します。

```
RENAME TABLE <schema_name>.<table_name> TO <new_name>
```

下記の SQL ステートメントは、 COMPANY スキーマ内の EMPLOYEE 表の名前を EMPL に変更します。

```
RENAME TABLE COMPANY.EMPLOYEE TO EMPL
```

RENAME TABLE ステートメントの詳細については、*SQL 解説書* を参照してください。

表の除去

表を除去するには、DROP TABLE SQL ステートメントを使います。

表が除去されると、その表についての情報が含まれる SYSCAT.TABLES カタログの中の行が除去され、その表に依存する他のオブジェクトがあれば、影響を受けます。たとえば、次のようになります。

- すべての列名は除去されます。

- その表の列について作成された索引は除去されます。
- その表に基づくすべての視点には作動不能のマークが付けられます。 (詳しくは、220ページの『作動不能視点の回復』を参照してください。)
- 除去された表と従属視点に対するすべての特権が暗黙のうちに取り消されます。
- その表が親表または従属表となっている参照制約がすべて除去されます。
- 除去された表に依存するすべてのパッケージおよびキャッシュに入った動的 SQL ステートメントは、無効のマークが付けられ、従属オブジェクトが再作成されるまで、そのままの状態になります。これには、除去される階層内の副表の上にあるスーパー表に依存しているパッケージが含まれます。 (詳しくは、225ページの『オブジェクトを変更する場合のステートメントの従属関係』を参照してください。)
- 参照の効力範囲として、除去された表を定義しているすべての参照列は、「効力範囲解除」されます。
- その表の別名定義は影響を受けません。別名は定義し直すことができません。
- 除去された表に依存しているすべてのトリガーには、作動不能のマークが付けられます。
- DATALINK 列によってリンクされているすべてのファイルは、リンク解除されます。リンク解除操作は非同期に実行されますので、それらのファイルは他の操作にすぐに使用することはできません。

コントロール・センターを使用して表を除去するには、以下のようにします。

1. オブジェクト・ツリーを順に展開し、**「表 (Tables)」**フォルダーを表示します。
2. 除去したい表を右クリックして、ポップアップ・メニューから**「除去 (Drop)」**を選択します。
3. **「確認 (Confirmation)」**ボックスにチェックを付け、**「Ok」**をクリックします。

コマンド行を使用して表を除去するには、以下のように入力します。

```
DROP TABLE <table_name>
```

次のステートメントは、DEPARTMENT という表を除去するものです。

```
DROP TABLE DEPARTMENT
```

個々の表に副表がある場合、その表は除去できません。ただし、次の例に示すとおり、表階層内の表はすべて、単一の DROP TABLE HIERARCHY ステートメントを使って除去できます。

```
DROP TABLE HIERARCHY person
```

DROP TABLE HIERARCHY ステートメントでは、除去する階層のルート表を指定しなければなりません。

表階層の除去と特定表の除去を比較した場合、次のような相違があります。

- DROP TABLE HIERARCHY は、個々の DROP 表ステートメントで活動化される削除トリガーを活動化しない。たとえば、個々の副表を除去すると、そのスーパー表に対する削除トリガーが活動化されます。
- DROP TABLE HIERARCHY は、除去された表の個々の行についてログ項目を作成しない。代わりに、階層の除去は単一のイベントとしてログ記録されます。

DROP ステートメントの詳細については、*SQL 解説書* を参照してください。

ユーザー定義一時表の除去

ユーザー定義一時表 (DECLARE GLOBAL TEMPORARY TABLE ステートメントを使用して作成した表) を除去する際には、いくつかの考慮事項に注意する必要があります。

この種の表を除去する際には、表名がスキーマ名 SESSION で修飾されていなければならず、その表を作成したアプリケーション中になければなりません。

パッケージはこのタイプの表に従属できないので、この種の表を除去しても無効になりません。

活動状態の作業単位や保管ポイントより前に作成したユーザー定義一時表を除去すると、その表は機能的に除去され、アプリケーションはその表にアクセスできなくなります。しかし、その表のスペースは依然として表スペース中に予約されているため、その作業単位がコミットされるか保管ポイントが終了するまで、ユーザー一時表スペースは除去できません。

DROP ステートメントの詳細については、*SQL 解説書* を参照してください。

トリガーの除去

トリガー・オブジェクトは DROP ステートメントを使用して除去できますが、この手順によって、以下のように従属パッケージに無効のマークが付けられます。

- 明示列リストなしの更新トリガーが除去されると、ターゲット表上での更新使用を持つパッケージは無効にされます。
- 列リストつきの更新トリガーが除去されると、ターゲット表上での更新使用を持つパッケージは、そのパッケージが CREATE TRIGGER ステートメントの `column-name` リストの中の少なくとも 1 つの列上での更新使用も持っている場合にのみ無効にされます。
- 挿入トリガーが除去されると、ターゲット表上での挿入使用を持つパッケージが無効にされます。
- 削除トリガーが除去されると、ターゲット表上での削除使用を持つパッケージが無効にされます。

パッケージは、アプリケーション・プログラムが明示的にバインドまたは再バインドされるか、あるいは、そのパッケージが実行され、データベース・マネージャーが自動的に再バインドを行うかするまで、無効のままとなります。

ユーザー定義関数 (UDF)、タイプ・マッピング、方式の除去

ユーザー定義関数 (UDF)、関数テンプレート、または関数マッピングは、 DROP ステートメントを使って除去することができます。

マッピング・オプション DISABLE を指定すれば、関数マッピングを使用不可にすることができます。実行方法についての詳細は、*SQL 解説書* を参照してください。

ある UDF に視点、トリガー、表検査制約、または別の UDF が従属している場合、その UDF は除去できません。 CREATE DISTINCT TYPE ステートメントによって暗黙的に生成された関数は除去できません。 SYSIBM スキーマまたは SYSFUN スキーマのいずれかの中にある関数は、除去できません。

他のオブジェクトを 1 つの関数または関数テンプレートに従属させることができます。関数を除去する前に、そのような従属関係 (関数マッピングを含む) を除去しておかなければなりません。ただし、作動不能のマークが付いているパッケージは例外です。このようなパッケージは、暗黙には再バインドされません。パッケージは、BIND または REBIND コマンドを使用して再バインドされるか、あるいは、PREP コマンドを使用して準備されていなければなりません。上記のコマンドについての詳細は、*コマンド解説書* を参照してください。 UDF を除去すると、それを使用していたパッケージまたはキャッシュに入った動的 SQL ステートメントをすべて無効にします。

関数マッピングを除去すると、パッケージに無効のマークが付けられます。自動再バインドが実行され、最適化プログラムはローカル関数を使用しようとします。ローカル関数がテンプレートである場合、暗黙的な再バインドは失敗します。

(詳細については、225ページの『オブジェクトを変更する場合のステートメントの従属関係』を参照してください。)

ユーザー定義タイプ (UDT) またはタイプ・マッピングの除去

ユーザー定義タイプ (UDT) またはタイプ・マッピングは、 DROP ステートメントを使って除去することができます。以下のように UDT が使用されている場合には UDT を除去できません。

- 既存の表や視点 (特殊タイプ) の列定義内で
- 既存のタイプ付き表またはタイプ付き視点のタイプとして使用されている場合 (構造型)
- 別の構造型のスーパータイプとして

デフォルトのタイプ・マッピングは除去できません。別のタイプ・マッピングを作成して指定変更することしかできません。

データベース・マネージャーはこの特殊タイプに依存する関数をすべて除去しようとします。 UDF を除去できなければ、UDT を除去できません。ある UDF に視点、トリガー、表検査制約、または別の UDF が従属している場合、その UDF は除去できません。 UDT を除去すると、それを使用していたパッケージまたはキャッシュに入った動的 SQL ステートメントをすべて無効にします。

UDT 用の変換を作成し、UDT の除去を計画している場合は、変換を除去する必要があるかどうかを考慮してください。これは、DROP TRANSFORM ステートメントを使って実行されます。このステートメントの詳細については、*SQL 解説書* を参照してください。管理者または他のアプリケーション開発者によって定義された変換だけを除去できる点に注意してください。組み込み変換とその関連グループの定義を除去することはできません。

ユーザー定義タイプについての詳細は、 *SQL 解説書* および *アプリケーション開発の手引き* を参照してください。

視点の変更または除去

ALTER VIEW ステートメントは、効力範囲を追加するよう参照タイプ列を変更することによって、既存の視点を変更します。視点に加えるその他の変更では、視点を除去した後に再作成する必要があります。

視点を変更するときには、効力範囲がまだ定義されていない既存の参照タイプ列に、効力範囲を追加することが必要です。さらに、列はスーパー表から継承したものであってはなりません。

ALTER VIEW ステートメントの列名のデータ・タイプは、 REF (タイプ付き表名またはタイプ付き視点名のタイプ) でなければなりません。

パッケージ、またはキャッシュに入った動的ステートメントに無効のマークが付けられても、表および索引のような他のデータベース・オブジェクトは影響されません。詳しくは、225ページの『オブジェクトを変更する場合のステートメントの従属関係』を参照してください。

ALTER VIEW ステートメントの詳細については、*SQL 解説書* を参照してください。

コントロール・センターを使用して視点を変更するには、以下のようにします。

1. オブジェクト・ツリーを順に展開し、**「視点 (Views)」**フォルダーを表示します。
2. 修正したい視点を右クリックし、ポップアップ・メニューから**「更新 (Alter)」**を選択します。
3. **「視点の更新 (Alter view)」**ウィンドウで、注釈を入力するか修正し、**「Ok」**をクリックします。

コマンド行を使用して視点を変更するには、以下のように入力します。

```
ALTER VIEW <view_name> ALTER <column name>
   ADD SCOPE <typed table or view name>
```

コントロール・センターを使用して視点を除去するには、以下のようにします。

1. オブジェクト・ツリーを順に展開し、**「視点 (Views)」**フォルダーを表示します。
2. 除去したい視点を右クリックして、ポップアップ・メニューから**「除去 (Drop)」**を選択します。
3. **「確認 (Confirmation)」**ボックスにチェックを付け、**「Ok」**をクリックします。

コマンド行を使用して視点を除去するには、以下のように入力します。

```
DROP VIEW <view_name>
```

以下の例は、EMP_VIEW を除去する方法を示したものです。

```
DROP VIEW EMP_VIEW
```

除去する視点に依存する視点があれば、それらは作動不能になります。 (詳しくは、『作動不能視点の回復』を参照してください。)

表階層の場合と同様、階層のルート視点を指定しても、次の例に示すとおり、1 つのステートメント内で視点階層全体を除去することはできません。

```
DROP VIEW HIERARCHY VPerson
```

視点の除去と作成についての詳細は、*SQL 解説書* を参照してください。

作動不能視点の回復

視点は、次のような場合に*作動不能* になることがあります。

- 基礎表での特権が取り消された結果
- 表、別名、または関数が除去される場合
- スーパービューが作動不能になる場合
- 従属する視点が除去される場合

次のステップは、作動不能視点を回復するのに役に立ちます。

1. 視点を作成するために最初に使用した SQL ステートメントを判別する。この情報は SYSCAT.VIEW カタログ視点の TEXT 列から獲得することができます。
2. CREATE VIEW ステートメントおよび同じ名前と同じ定義を使用して、視点を再作成する。
3. GRANT ステートメントを使用して、視点に以前に付与されていたすべての特権を再度付与する。 (作動不能視点に付与されていたすべての特権は取り消されていることに注意してください。)

作動不能視点を回復したくない場合は、 DROP VIEW ステートメントを使用してその視点を明示的に除去するか、または同じ名前と別の定義を使用して新規の視点を作成することができます。

作動不能視点は、SYSCAT.TABLES および SYSCAT.VIEWS カタログ視点にしか項目がありません。 SYSCAT.VIEWDEP、SYSCAT.TABAUTH、SYSCAT.COLUMNS、および SYSCAT.COLAUTH カタログ視点のすべての項目が除去されます。

要約表の除去

要約表は変更できませんが、除去することはできます。

この表を参照しているすべての索引、基本キー、外部キー、および検査制約が除去されます。この表を参照するすべての視点およびトリガーは、作動不能になります。除去されたオブジェクトまたは作動不能とマークされたオブジェクトに依存するすべてのパッケージは、無効になります。パッケージの従属関係の詳細については、225ページの『オブジェクトを変更する場合のステートメントの従属関係』を参照してください。

コントロール・センターを使用して要約表を除去するには、以下のようにします。

1. オブジェクト・ツリーを順に展開し、**「表 (Tables)」**フォルダーを表示します。
2. 除去したい要約表を右クリックして、ポップアップ・メニューから**「除去 (Drop)」**を選択します。
3. **「確認 (Confirmation)」**ボックスにチェックを付け、**「Ok」**をクリックします。

コマンド行を使用して要約表を除去するには、以下のように入力します。

```
DROP TABLE <table_name>
```

次の SQL ステートメントは、要約表 XT を除去するものです。

```
DROP TABLE XT
```

作動不能要約表の回復

要約表は、基礎表での SELECT 特権を取り消されると、*作動不能* になります。

次のステップは、作動不能要約表を回復するのに役に立ちます。

- 要約表を作成するために最初に使用した SQL ステートメントを判別する。この情報は SYSCAT.VIEW カタログ視点の TEXT 列から獲得することができます。
- CREATE SUMMARY TABLE ステートメントおよび同じ名前と同じ定義を使用して、要約表を再作成する。
- GRANT ステートメントを使用して、要約表に以前に付与されていたすべての特権を再度付与する。 (作動不能要約表に付与されていたすべての特権は取り消されていることに注意してください。)

作動不能要約表を回復したくない場合は、 DROP TABLE ステートメントを使用してその要約表を明示的に除去するか、または同じ名前と別の定義を使用して新規の要約表を作成することができます。

作動不能要約表は、SYSCAT.TABLES および SYSCAT.VIEWS カタログ視点にしか項目がありません。 SYSCAT.VIEWDEP、SYSCAT.TABAUTH、 SYSCAT.COLUMNS、および SYSCAT.COLAUTH カタログ視点のすべての項目が除去されます。

ラッパーの除去

DROP ステートメントを使うと、データベースからラッパーを除去することができます。以下の例は、DRDA ラッパーを除去する方法を示したものです。

```
DROP WRAPPER DRDA
```

ラッパーに従属するすべてのサーバー定義、ユーザー定義関数マッピング、およびユーザー定義データ・タイプ・マッピングは除去されます。除去されたサーバー定義に従属するすべてのユーザー定義マッピング、ニックネーム、ユーザー定義タイプ・マッピング、およびユーザー・マッピングも除去されます。除去されたニックネームに従属する索引の指定は除去され、それらのニックネームに従属する視点は作動不能としてマークされます。除去されたオブジェクトに従属するすべてのパッケージと作動不能視点は無効にされます。

DROP ラッパーに対する SYSADM または DBADM 権限のいずれかを持っている必要があります。

ラッパーの除去についての詳細は、*SQL 解説書* を参照してください。

サーバーの変更または除去

ALTER SERVER ステートメントは、連合データベース・カタログ内の既存サーバー定義を修正します。このステートメントは、次の目的で使用します。

- 特定データ・ソースの定義を修正する。
- 特定タイプまたはバージョンの複数データ・ソースの定義を修正する。
- 特定データ・ソースの構成に変更を加える。たとえば、特定サーバーによって識別された DBMS がより高速のプロセッサーを持つ、新しいワークステーションに移行される場合は、 cpu_ratio サーバー・オプションを更新する必要があります。

このステートメントを使って、 *dbname* または *node* サーバー・オプションを修正することはできません。

次の例は、ORA1 サーバーの変更方法を示しています。

```
ALTER SERVER ORA1 OPTIONS (SET CPU_RATIO '5.0')
```

サーバーは連合データベースから除去することができます。 次の例は、ORALOC01 サーバーを除去する方法を示しています。

```
DROP SERVER ORALOC01
```

データ・ソースにある表および視点のすべてのニックネームは除去されます。それらのニックネームに従属する索引の指定は除去されます。除去されたサーバー定義に従属するすべてのユーザー定義関数マッピング、ユーザー定義タイプ・マッピング、およびユーザー・マッピングも除去されます。除去されたサーバー定義に従属するすべてのパッケージ、関数マッピング、ニックネーム、および索引の指定は無効にされます。

ALTER または DROP サーバーに対する SYSADM または DBADM 権限のいずれかを持っている必要があります。

サーバーの除去および変更についての詳細は、*SQL 解説書* を参照してください。

ニックネームの変更または除去

ALTER NICKNAME を使うと、データ・ソース表または視点に関してローカルに保管された情報を更新することができます。たとえば、このステートメントを使って列のローカル名を変更したり、列データ・タイプを別のデータ・タイプにマップしたりすることができます。このステートメントは列オプションを追加する場合にも使用できます。ALTER NICKNAME 構文についての詳細は、*SQL 解説書* を参照してください。

ニックネームが除去されると、そのニックネームに関して作成された視点には作動不能のマークが付けられます。ニックネームが視点内で参照されているときにニックネームの列名やデータ・タイプを変更することはできません。

SYSADM または DBADM 権限の 1 つか、ニックネームに対する CONTROL および ALL データベース特権のいずれか、または ALTERIN (現行スキーマの場合) スキーマ特権を持っているか、あるいはニックネームの定義者でなければ、連合データベースでこのステートメントを使用することはできません。

ニックネーム列の変更およびニックネームの除去

次の例は、ニックネーム TESTNN を変更して、 COL1 から列のローカル名を NEWCOL に変更する方法を示しています。

```
ALTER NICKNAME TESTNN ALTER COLUMN COL1 LOCAL NAME NEWCOL
```

次の例は、ニックネーム TESTNN を除去する方法を示しています。

```
DROP NICKNAME TESTNN
```

ニックネーム列オプションの変更

列オプション というパラメーターに割り当てる値の形式で列情報を指定します。これらの値は、大文字でも小文字でも指定できます。以下の表では、各値の説明と追加情報を記します。

表 3. 列オプションとその設定値

オプション	有効な設定値	デフォルト設定
numeric_string	'Y' はい - この列には数値データのストリングだけが含まれます。重要: この列に、後書きブランクを付けられた数値ストリングだけが含まれる場合、 'Y' を指定することはお勧めできません。 'N' いいえ - この列は数値データのストリングに限定されていません。 列の numeric_string を 'Y' に設定すると、列データの分類に干渉するブランクがこの列には含まれないことを、最適化プログラムに知らせることになります。このオプションは、データ・ソースの照合順序が DB2 の照合順序とは異なる場合に役立ちます。このオプションでマークされた列は、照合順序が異なるためにローカルな (データ・ソースの) 評価から除かれるということはありません。	'N'
varchar_no_trailing_blanks	特定の VARCHAR 列に後書きブランクがないかどうかを示します。 'Y' はい - この VARCHAR 列に後書きブランクはありません。 'N' いいえ - この VARCHAR 列には後書きブランクがあります。 データ・ソース VARCHAR 列に埋め込みブランクがない場合、最適化プログラムがその列にアクセスするときの戦略は、列に後書きブランクが含まれているかどうかによって異なる部分があります。デフォルトでは、最適化プログラムは後書きブランクを実際に含んでいるものと「想定」しています。この想定に基づき、これらの列から返された値が希望する値になるように照会を変更するアクセス戦略が開発されます。ただし、VARCHAR 列に後書きブランクがなく、そのことを最適化プログラムに知らせてある場合、さらに効果的なアクセス戦略を開発することができます。特定の列に後書きブランクがないことを最適化プログラムに知らせるには、ALTER NICKNAME ステートメントでその列を指定してください (構文については、 *SQL 解説書* を参照してください)。	'N'

索引、索引の拡張、または索引の指定の除去

索引定義、索引の拡張、または索引の指定の文節は変更できません。索引や索引の拡張を除去してから再び作成してください。 (索引または索引の指定を除去しても、他のオブジェクトが除去されることはありません。しかし、場合によっては一部のパッケージが無効になることがあります。)

コントロール・センターを使用して索引、索引の拡張、または索引の指定を除去するには、以下のようにします。

1. オブジェクト・ツリーを順に展開し、**「索引 (Indexes)」**フォルダーを表示します。
2. 除去したい索引を右クリックして、ポップアップ・メニューから**「除去 (Drop)」**を選択します。
3. **「確認 (Confirmation)」**ボックスにチェックを付け、**「Ok」**をクリックします。

コマンド行を使用して索引、索引の拡張、または索引の指定を除去するには、以下のように入力します。

```
DROP INDEX <index_name>
```

次の SQL ステートメントは、PH という索引を除去するものです。

```
DROP INDEX PH
```

次の SQL ステートメントは、IX_MAP という索引の拡張を除去するものです。

```
DROP INDEX EXTENSION ix_map RESTRICT
```

索引の拡張の名前は、カタログ中に記述されている索引の拡張を識別するものでなければなりません。 RESTRICT 文節は、索引の拡張の定義に従属する索引は定義できないという規則を施行します。基礎となる索引がこの索引の拡張に従属していると、除去は失敗します。

基本キーまたは固有キーの索引 (索引の指定でない場合に限る) は、明示的に除去することはできません。索引を除去するには、次の方法のいずれかを使用してください。

- 基本キーまたは固有キーに対して、 1 次索引または固有制約が自動的に作成されていた場合、基本キーまたは固有キーを除去すると、索引が除去されることになります。除去は、ALTER TABLE ステートメントによって行われます。
- 1 次索引または固有制約がユーザー定義であった場合、ALTER TABLE を使用して、まず基本キーまたは固有キーを除去しなければなりません。基本キーまたは固有キーが除去された後は、索引は 1 次索引または固有索引とは考えられなくなるため、明示して除去することができます。

その索引に依存するパッケージ、およびキャッシュに入った動的 SQL ステートメントには、無効のマークが付けられます。詳しくは、『オブジェクトを変更する場合のステートメントの従属関係』を参照してください。アプリケーション・プログラムは、索引の追加や除去による変更事項には影響されません。

オブジェクトを変更する場合のステートメントの従属関係

ステートメントの従属関係には、パッケージ、およびキャッシュに入った動的 SQL ステートメントが含まれます。*パッケージ* とは、データベース・オブジェクトの 1 つで、データベース・マネージャーが、特定のアプリケーション・プログラムにとって最も効率的な方法でデータにアクセスするのに必要な情報が入れられたものです。*バインド* とは、データベース・マネージャーが、アプリケーションの実行時にデータベースにアクセスするのに必要なパッケージを作成するプロセスです。 *アプリケーション開発の手引き* に、パッケージの作成方法についての詳細な説明があります。

パッケージおよびキャッシュに入った動的 SQL ステートメントは、さまざまなタイプのオブジェクトに従属することができます。これらのオブジェクトの全リストについては、*SQL 解説書* を参照してください。

そうしたオブジェクトは、明示的に参照できます。 SQL SELECT ステートメントに含める表やユーザー定義関数などはその例です。また、暗示的に参照できるオブジェクトもあります。たとえば、親表の行の削除時に、参照制約の違反がないかどうかの検査が必要な従属表がそうです。パッケージはさらに、パッケージ作成者に付与される特権にも依存しています。

パッケージ、またはキャッシュに入った動的 SQL ステートメントがオブジェクトに依存しており、そのオブジェクトが除去された場合は、そのパッケージまたはキャッシュに入った動的 SQL ステートメントは、「無効」状態になります。パッケージがユーザー定義関数に従属し、その関数が除去されると、パッケージは「作動不能」状態になります。

キャッシュに入れられた動的 SQL ステートメント (無効状態) は再び、次の使用に関して自動的に最適化されます。ステートメントに必要なオブジェクトが除去されている場合に動的 SQL ステートメントを実行すると、失敗してエラー・メッセージが表示されることがあります。

無効状態にあるパッケージは、次の使用に関して暗黙的に再バインドされます。そのようなパッケージは明示的に再バインドすることもできます。トリガーが除去されたためにパッケージに無効のマークが付けられた場合、再バインド・パッケージはトリガーを呼び出さなくなります。

無効状態のパッケージは、明示的に再バインドした後でなければ使用できません。パッケージのバインドおよび再バインドについての詳細は、 *アプリケーション開発の手引き* を参照してください。

連合データベースのオブジェクトには、同様の従属関係があります。たとえば、サーバーを除去すると、そのサーバーに関連付けられたニックネームを参照するパッケージ、またはキャッシュに入れられた動的 SQL は無効になります。

ある場合には、パッケージを再バインドできないことがあります。たとえば、表が除去されたのに再作成されない場合は、パッケージを再バインドできません。この場合、オブジェクトを再作成するか、除去されたオブジェクトをアプリケーションが使用しないようにアプリケーションを変更しなければなりません。

その他のほとんどの場合、たとえば、制約の一部が除去された場合は、パッケージを再バインドすることが可能です。

以下のシステム・カタログ視点は、パッケージの状態およびパッケージの従属関係を判別するのに役立ちます。

- SYSCAT.PACKAGEAUTH
- SYSCAT.PACKAGEDEP
- SYSCAT.PACKAGES

オブジェクトの従属関係について詳しくは、 *SQL 解説書* の DROP ステートメントの説明を参照してください。

第3部 データベースの機密保護

第5章 データベース・アクセスの制御

データベース管理者の重要な責務の 1 つに、データベースの機密保護があります。データベースの機密保護には、いくつかの活動が含まれます。

- 装置またはシステムの誤動作によるデータまたはデータ保全性の損失を防ぐこと。
- 貴重なデータへの無許可アクセスを防ぐこと。重要な情報については、『知る必要』のない人からはアクセスできないようにしなければなりません。
- 無許可ユーザーが、誤ってデータの削除や変更を行わないように保護すること。
- 285ページの『第6章 DB2 アクティビティーの監査』で説明されているユーザーによるデータのアクセスをモニターすること。

この部分では、次の点について説明します。

- 『導入システムのためのユーザー ID およびグループの選択』
- 235ページの『サーバーに対する認証方式の選択』
- 240ページの『リモート・クライアントでの認証についての考慮事項』
- 241ページの『区分データベースの考慮事項』
- 241ページの『ユーザーの認証のための DCE 機密保護サービスの使用』
- 247ページの『連合データベースの認証処理』
- 253ページの『特権、権限、および許可』
- 267ページの『データベース・オブジェクトに対するアクセスの制御』
- 278ページの『タスクとそれに必要な権限許可』
- 279ページの『システム・カタログの使用』

機密保護のための計画: データベース・アクセス制御の目標を定義し、だれに、何を、どのような状況でアクセスさせるかを指定することから始めてください。このような計画では、データベース機能、他のプログラムの機能、および管理手順を使用することによって、これらの目的を実現する方法についても記述する必要があります。

導入システムのためのユーザー ID およびグループの選択

機密保護の問題は、製品がインストールされたときから、DB2 管理者にとって重要なことです。プラットフォーム固有のそれぞれの*概説およびインストール* では、 DB2 の計画、インストール、および構成に必要なすべての情報を提示しています。

DB2 のインストールを完了させるには、ユーザー名、グループ名、およびパスワードが必要となります。インストール時に、管理者はこれらの要件ごとにデフォルト値を使用します。一度、DB2 のインストール時にデフォルトが使用されると、データベースが常

駐するインスタンスを作成する前に、管理者は新しいユーザー名、グループ名、およびパスワードを作成するよう強く勧められます。新しいユーザー名、グループ名、およびパスワードを使用することにより、デフォルトを知り、それからインスタンスおよびデータベース内で不適切な方式でそれらを使用する、管理者以外のユーザーのリスクを最小限にします。

ユーザーを認証する際に、パスワードは非常に重要です。認証要件がオペレーティング・システム・レベルで設定されていないときに、データベースがオペレーティング・システムを使用してユーザーを認証する場合、ユーザーは接続を許可されます。たとえば、UNIX オペレーティング・システムでは、未定義のパスワードは NULL として扱われます。また、定義されたパスワードを持つユーザーは、NULL パスワードを持っているものとして扱われます。オペレーティング・システムの観点では、これが一致であり、ユーザーの妥当性検査が行われ、データベースに接続することができます。オペレーティング・システムがデータベースに対するユーザーの認証を行う場合は、オペレーティング・システム・レベルのパスワードが必要です。

DB2 のインストールの後に続く別の機密保護勧告は、ユーザーに授与されたデフォルト特権の変更です。インストール処理の間、システム管理 (SYSADM) の特権は、各オペレーティング・システム上で下記のユーザーにデフォルトにより与えられます。

OS/2
ユーザー・プロファイル管理 (UPM) 管理者グループまたはローカル管理者グループに属する有効な DB2 のユーザー ID。

Windows 95 または Windows 98
あらゆる Windows 95 または Windows 98 ユーザー。

Windows NT または Windows 2000
管理者グループに属する有効な DB2 のユーザー名。

UNIX
インスタンス所有者のユーザー ID の 1 次グループに属する有効な DB2 のユーザー名。

SYSADM 特権は、DB2 の中の使用可能な特権の最も強力なセットです。 (特権については、この章の後の方で説明されています。) その結果、これらのユーザーのすべてがデフォルトによって、 SYSADM 特権を持つことを望むことはできません。 DB2 は、グループおよび個々のユーザー ID に特権を与えたり、取り消したりする能力を管理者に付与しています。

グループおよびユーザー ID を作成し、そして割り当てるプラットフォーム固有の情報は、それぞれの*概説およびインストール* に見いだせます。データベース・マネージャーの構成パラメーター SYSADM_GROUP を更新することにより、管理者はどのグループをシステム管理者特権を持つシステム管理グループとして定義するかを制御することが

できます。 DB2 インストールとその後のインスタンスだけでなく、さらにデータベース作成の機密保護要件を完成するために、下記のガイドラインに従わなければなりません。

システム管理グループと定義される (SYSADM_GROUP を更新することにより) グループが、少なくとも 1 つは存在しなければなりません。このグループの名前を使用すれば、インスタンス所有者のために作成されたグループのように簡易識別することができます。このグループに属するユーザー ID およびグループは、それぞれのインスタンスに対してシステム管理者権限を持っています。

特定インスタンスに関連付けられていると容易に認識される、インスタンス所有者ユーザー ID を作成することを考慮する必要があります。このユーザー ID は、そのグループの名前の 1 つとして上記で作成された SYSADM グループの名前を持つ必要があります。別の勧告は、インスタンス所有者のユーザー ID をインスタンス所有者グループの一員としてだけに使用し、他のグループでは使用しないようにすることです。このようにすることによって、インスタンス環境を変更できるユーザー ID および グループが増えるのを制御できます。

作成されたユーザー ID は、インスタンス内のデータおよびデータベースへの入力前に認証を可能にするため、 1 つのパスワードと必ず関連付けてください。パスワード作成時の勧告は、編成のパスワード命名ガイドラインに従うことです。

Windows NT プラットフォームについての考慮事項

エンタープライズ拡張エディション (Windows NT 版) で作業する場合、システム管理 (SYSADM) 権限は、 DB2 ユーザー・アカウントが定義されているマシンのローカル管理者グループに属している、有効なあらゆる DB2 ユーザー・アカウントに付与されます。

たとえば、ユーザーがあるドメイン・アカウントにログオンし、 DB2 データベースへのアクセスを試みる場合、 DB2 はドメイン・コントローラーを調べて、グループ (管理者のグループも含む) を列挙します。この動作は、以下のいずれかの方法で変更できます。

1. レジストリー変数 DB2_GRP_LOOKUP = local を設定して、ドメイン・アカウント (またはグローバル・グループ) をローカル管理者グループに追加します。
2. データベース・マネージャー構成ファイルを更新して、新しいグループを指定します。そのグループがローカル・マシンで列挙されるようにするには、DB2_GRP_LOOKUP レジストリー変数も設定しなければなりません。

Windows NT ドメイン環境のデフォルトでは、インスタンスに対する SYSADM 権限を付与されるのは、プライマリー・ドメイン・コントローラー (PDC) の管理者グループに属しているドメイン・ユーザーだけです。 DB2 は必ずアカウントが定義されているマシンで許可を行うので、サーバーのローカル管理者グループにドメイン・ユーザーを追加しても、そのグループにはドメイン・ユーザー SYSADM 権限は付与されません。

PDC の管理者グループにドメイン・ユーザーが追加されないようにするには、グローバル・グループを作成し、 SYSADM 権限を付与するユーザー (ドメインとローカルの両方) を追加します。これを行うには、以下のコマンドを入力します。

```
DB2STOP
DB2 UPDATE DBM CFG USING SYSADM_GROUP global_group
DB2START
```

UNIX プラットフォームについての考慮事項

UNIX ベースのプラットフォームの場合、分離されたユーザー定義関数 (UDF) およびストアード・プロシージャーのグループを作成しなければならず、かつ分離されたユーザー定義関数またはストアード・プロシージャーを使用するすべてのユーザー ID は、このグループのメンバーでなければなりません。 SYSADM グループの場合もそうですが、分離 UDF またはストアード・プロシージャーのグループ名は簡易識別を可能にしてください。分離 UDF またはストアード・プロシージャー・グループに属するユーザー ID は、デフォルトとしてグループに関連しているどんな権限および特権でも所有しています。

機密保護の理由で、インスタンス名を分離 ID として使用しないことをお勧めします。ただし、分離 UDF またはストアード・プロシージャーを使用する計画がないならば、別のユーザー ID を作成する代わりに分離 ID をインスタンス名に設定することができます。

勧告は、このグループに関連付けられていると認識されるユーザー ID を作成することです。分離 UDF およびストアード・プロシージャーのユーザーは、インスタンス作成スクリプト (db2icrt ... -u <FencedID>) のパラメーターとして指定されます。 DB2 クライアントまたは DB2 ソフトウェア開発者キットをインストールする場合、これは必須ではありません。

一般的な規則

すべてのオブジェクトとユーザーの命名について規則があります。これらの規則には、作業しているプラットフォームに特有のものもあります。たとえば、名前に大文字と小文字を使用することに関連した規則があります。

- UNIX プラットフォームでは、名前は小文字でなければなりません。
- OS/2 では、名前は大文字でなければなりません。
- Windows プラットフォームでは、名前は大文字でも、小文字でも、大小混合でも構いません。

DB2 の命名規則については、329ページの『付録A. 命名規則』を参照してください。

db2icrt コマンドはインスタンス所有者のホーム・ディレクトリーの下に、メイン SQL ライブラリー (sqllib) ディレクトリーを作成します。

サーバーに対する認証方式の選択

インスタンスまたはデータベースにアクセスするためには、まず、そのユーザーが*認証*されていることが必要です。各インスタンスの*認証タイプ* によって、ユーザーを検査する方法と場所が決まります。認証タイプは、サーバーのデータベース・マネージャー構成ファイルに保管されます。認証タイプは、インスタンスの作成時に初期設定されます。*認証*データベース・マネージャー構成パラメーターについての詳細は、 *管理の手引き: パフォーマンス* の『DB2 の構成』を参照してください。 インスタンスごとに 1 つの認証タイプがあり、それが、そのデータベース・サーバーおよびその制御下のすべてのデータベースのアクセスをカバーしています。

連合データベースからデータ・ソースにアクセスしたい場合、データ・ソース認証処理および統合認証タイプの定義を考慮する必要があります。詳細については、247ページの『連合データベースの認証処理』を参照してください。

以下の認証タイプがあります。

SERVER

ローカルのオペレーティング・システムの機密保護を使用して、サーバー上で認証が行われることを指定します。接続が試みられているときにユーザー ID およびパスワードが指定されると、それらがサーバーにある有効なユーザー ID とパスワードの組み合わせと比較され、そのユーザーがそのインスタンスへのアクセスを許されているかどうかが判別されます。これがデフォルトの機密保護メカニズムです。

注: サーバー・コードは、接続がローカルなのかリモートなのかを検出します。ローカル接続の場合、認証が SERVER であると、ユーザー ID とパスワードは、認証の成功のためには必要とされません。

リモート・インスタンスがサーバー認証を持っている場合、次の 2 つの方法で認証が行われます。

- ユーザー ID とパスワードがユーザーによって提供される。
- ユーザー ID とパスワードが DB2 によって検索されてから、サーバーに渡されて妥当性検査が行われる。 (ユーザーはすでにローカル・マシンまたはドメインにログインされています。)

SERVER_ENCRYPT

サーバーが、暗号化された SERVER 認証スキーマを受け入れるように指定します。クライアント認証が指定されない場合、クライアントはサーバーで選択された方式を使用して認証されます。

クライアント認証が DCS または SERVER である場合、クライアントはユーザー ID およびパスワードをサーバーに渡すことによって認証されます。クラ

イアント認証が DCS_ENCRYPT または SERVER_ENCRYPT である場合、クライアントはユーザー ID および暗号化されたパスワードを渡すことによって認証されます。

SERVER_ENCRYPT がクライアントで指定され、SERVER がサーバーで指定されると、認証レベルの不一致のためにエラーが戻されます。

CLIENT

オペレーティング・システムの機密保護を使用して、アプリケーションが呼び出されたデータベース区画上で認証が行われることを指定します。接続が試みられているときにユーザー ID およびパスワードが指定されると、それらがサーバーにある有効なユーザー ID とパスワードの組み合わせと比較され、そのユーザー ID がそのインスタンスへのアクセスを許されているかどうかが判別されます。データベース・サーバーでは、それ以上の認証は行われません。

ユーザーがローカルまたはクライアントのログインを行った場合、そのユーザーは、そのローカルのクライアント・ワークステーションでのみ認識されます。

リモート・インスタンスが CLIENT 認証である場合、 *trust_allclnts* と *trust_clntauth* という他の 2 つのパラメーターが最終的な認証タイプを決定します。

TRUSTED クライアントのみに対する CLIENT レベル機密保護:

トラステッド・クライアントとは、信頼できるローカル機密保護システムをもつクライアントのことです。具体的には、Windows 95 および Windows 98 の各オペレーティング・システムを除く、すべてのクライアントがトラステッド・クライアントです。

CLIENT の認証タイプが選択されている場合、固有の機密保護を操作環境が持っていないクライアントに対する保護のために、追加のオプションを選択することができます。

機密保護のないクライアントに対する保護のために、管理者は、*trust_allclnts* パラメーターを NO に設定することによって、「トラステッド・クライアント認証」を選択することができます。これは、すべてのトラステッド・プラットフォームが、サーバーに代わってユーザーの認証ができることを意味します。非トラステッド・クライアントは、サーバー上で認証され、ユーザー ID とパスワードを提供しなければなりません。ユーザーは、クライアントを信頼するかどうかを示すために、 *trust_allclnts* 構成パラメーターを使用します。このパラメーターのデフォルトは YES です。

注: 一部のクライアントが認証のためのネイティブの安全な機密保護システムを持っていない場合であっても、すべてのクライアントをトラステッド・クライアント (*trust_allclnts* が YES) とすることは可能です。

トラステッド・クライアントの場合であっても、サーバー側で認証を完了させたい場合があります。トラステッド・クライアントをどこで妥当性検査するかを指示するために、 *trust_clntauth* 構成パラメーターを使用します。このパラメーターのデフォルトは CLIENT です。このパラメーターについての詳細は、*管理の手引き: パフォーマンス* の『DB2 の構成』を参照してください。

注: トラステッド・クライアントの場合のみ、 CONNECT または ATTACH を試みているときにユーザー ID またはパスワードが明示して提供されないと、ユーザーの妥当性検査は、そのクライアントで行われます。 *trust_clntauth* パラメーターは、 USER/USING 文節で提供された情報をどこで妥当性検査するかを判別するためだけに使用されます。

DRDA クライアントを除くすべてのクライアントに対して、DB2 (MVS 版)、DB2 (OS/390 版)、DB2 (VM および VSE 版)、および DB2 (OS/400 版) から保護するには、 *trust_allclnts* パラメーターを DRDAONLY に設定します。上記のクライアントだけを、クライアント側の確認を行うよう承認することができます。他のすべてのクライアントには、サーバーによって認証されているユーザー ID とパスワードが必要です。

trust_clntauth パラメーターは、上記のクライアントが認証される位置を判別するのに使用されます。 *trust_clntauth* が "client" である場合、認証はクライアントで行われます。 *trust_clntauth* を "server" に設定すると、認証は、クライアント (パスワードが指定されなかった場合) およびサーバー (パスワードが指定された場合) で行われます。

表 4. TRUST_ALLCLNTS および TRUST_CLNTAUTH パラメーターの組み合わせを使用した認証モード

TRUST_ ALLCLNTS	TRUST_ CLNTAUTH	非トラステッドである DRDA クライアント認証、パスワードなし	非トラステッドである DRDA クライアント認証、パスワードあり	トラステッドである DRDA クライアント認証、パスワードなし	トラステッドである DRDA クライアント認証、パスワードあり	DRDA クライアント認証、パスワードなし	DRDA クライアント認証、パスワードあり
YES	CLIENT	CLIENT	CLIENT	CLIENT	CLIENT	CLIENT	CLIENT
YES	SERVER	CLIENT	SERVER	CLIENT	SERVER	CLIENT	SERVER
NO	CLIENT	SERVER	SERVER	CLIENT	CLIENT	CLIENT	CLIENT
NO	SERVER	SERVER	SERVER	CLIENT	SERVER	CLIENT	SERVER
DRDAONLY	CLIENT	SERVER	SERVER	SERVER	SERVER	CLIENT	CLIENT
DRDAONLY	SERVER	SERVER	SERVER	SERVER	SERVER	CLIENT	SERVER

DCS 主として、DB2 コネクトを使用してアクセスされるデータベースをカタログ化するために使用されます。 (このトピックについてさらに詳しくは、 *DB2 コネクト 使用者の手引き* の機密保護に関する節を参照してください。) これがデータベース・マネージャー構成ファイルの中の 1 つのインスタンスに対する

認証タイプを指定するために使用された場合、このタイプは **SERVER** 認証の場合と同じ意味になります。ただし、拡張プログラム間通信 (APPC) プロトコルを使用して分散リレーショナル・データベース体系 (DRDA) アプリケーション・サーバー (AS) を介して、サーバーがアクセスされている場合を除きます。 この場合、**DCS** を使用すると、認証はサーバーで行われるが、 APPC 層内のみであることを示します。それ以上の認証は、DB2 コード内では行われません。この値は、接続に対する APPC SECURITY パラメーターが、 SAME または PROGRAM として指定されている場合にのみサポートされます。

DCS_ENCRYPT
DB2 コネクトが、暗号化された SERVER 認証スキーマを受け入れるように指定します。クライアント認証が指定されない場合、クライアントはサーバーで選択された方式を使用して認証されます。

クライアント認証が DCS または SERVER である場合、クライアントはユーザー ID およびパスワードを DB2 コネクトに渡すことによって認証されます。クライアント認証が DCS_ENCRYPT または SERVER_ENCRYPT である場合、クライアントはユーザー ID および暗号化されたパスワードを渡すことによって認証されます。

DCS_ENCRYPT がクライアントで指定され、DCS がサーバーで指定されると、認証レベルの不一致のためにエラーが戻されます。

DCE DCE 機密保護サービスを使用してユーザーが認証されることを指定します。DCE 機密保護について詳しくは、241ページの『ユーザーの認証のための DCE 機密保護サービスの使用』を参照してください。

DCE_SERVER_ENCRYPT
サーバーが、DCE 認証または暗号化された SERVER 認証スキーマを受け入れるように指定します。クライアント認証が DCE であるか、または指定されない場合、クライアントは DCE 機密保護サービスを使用して認証されます。DCE 機密保護について詳しくは、241ページの『ユーザーの認証のための DCE 機密保護サービスの使用』を参照してください。

クライアント認証が SERVER または DCS である場合、クライアントはユーザー ID およびパスワードをサーバーに渡すことによって認証されます。クライアント認証が SERVER_ENCRYPT または DCS_ENCRYPT である場合、クライアントはユーザー ID および暗号化されたパスワードを渡すことによって認証されます。クライアントの認証タイプを DCE_SERVER_ENCRYPT として指定することはできません。インスタンスの認証タイプが DCE_SERVER_ENCRYPT として指定されると、すべてのローカル・アプリケーションは DCE を認証スキーマとして使用します。これは、データベース接続またはインスタンス接続を必要としないユーティリティー・コマンドにも当てはまります。

DCE と SERVER_ENCRYPT 認証タイプの混合を許可することに加えて、DCE_SERVER_ENCRYPT 認証タイプも、DCE 内のグループを使用するとき

に、制限のうち 1 つを緩和します。認証タイプが DCE_SERVER_ENCRYPT に設定される場合、前提事項は認証時以外で要求されているグループ・リストであり、 DCE からではなく基本オペレーティング・システムからとられます。その後、管理者として、認証時にサポートされるグループ・リスト・サポートを外部で提供するために、サーバー上でユーザーがショート DCE 名と一致するように設定することができます。

KERBEROS

DB2 クライアントとサーバーが両方とも、ケルベロス機密保護プロトコルをサポートしているオペレーティング・システム上で実行されている場合に使用します。ケルベロス機密保護プロトコルは、従来の暗号を使用して共有秘密鍵を作成することにより、サード・パーティーの認証サービスとして認証を実行します。この鍵がユーザーの証明書になり、ローカルまたはネットワーク・サービスが要求されるたびに、ユーザーのアイデンティティーの確認に使用されます。この鍵を使用することにより、ネットワークを介して生のテキストでユーザー名およびパスワードを渡す必要がなくなります。ケルベロス機密保護プロトコルにより、リモート DB2 サーバーへの単一のサインオンを行えるようになります。

KRB_SERVER_ENCRYPT

サーバーが、KERBEROS 認証または暗号化された SERVER 認証スキーマを受け入れるように指定します。クライアント認証が KERBEROS である場合、クライアントはケルベロス機密保護システムを使用して認証されます。クライアント認証が KERBEROS でない場合、システム認証タイプは SERVER_ENCRYPT と同じになります。

注: ケルベロス認証タイプがサポートされているのは、 Windows 2000 を実行しているクライアントおよびサーバーだけです。

注:

1. 認証のタイプの選択が重要になるのは、データベースにアクセスするリモート・データベース・クライアントがある場合、または連合データベース機能の使用時だけです。ローカル・クライアントを通してデータベースにアクセスしているほとんどのユーザーは、データベースと同じマシン上で常に認証されます。 DCE 機密保護サービスが使用される場合、この例外がある場合があります。リモート・クライアントのサポートと使用については、使用しているプラットフォーム用の*概説およびインストール* を参照してください。
2. 構成ファイル自体へのアクセスは構成ファイル内の情報によって保護されているため、認証情報を変更しているときに、誤って自分自身を自分のインスタンスからロックアウトしてしまわないようにしてください。以下のデータベース・マネージャー構成ファイル・パラメーターは、インスタンスへのアクセスを制御します。
   - AUTHENTICATION *
   - SYSADM_GROUP *

- TRUST_ALLCLNTS
- TRUST_CLNTAUTH
- SYSCTRL_GROUP
- SYSMAINT_GROUP

* は、2 つの最も重要なパラメーターを示し、これらが最も問題を引き起こす可能性があります。

このようなことが起こらないようにするために、行えることがいくつかあります。誤って自分自身を DB2 システムからロックアウトしてしまった場合、すべてのプラットフォームで使用可能なフェールセーフのオプションがあります。これは、高い特権をもったローカルのオペレーティング・システムの機密保護ユーザーを使用して、通常の DB2 機密保護検査をオーバーライドしてデータベース・マネージャー構成ファイルを更新することです。このユーザーは、**常に**データベース・マネージャー構成ファイルを更新するための特権を持っており、それによって問題を訂正します。ただし、この機密保護上のう回は、データベース・マネージャー構成ファイルのローカルでの更新にのみ制限されています。フェールセーフのためのユーザーは、リモートで、または他の DB2 コマンドに対して使用することはできません。この特別のユーザーは、以下のように識別されます。

- UNIX プラットフォームの場合: インスタンス所有者
- NT プラットフォームの場合: ローカル 『管理者』グループに属している人
- OS/2 プラットフォームの場合: UPM 管理者
- その他のプラットフォームの場合: その他のプラットフォーム上ではローカル機密保護がないため、すべてのユーザーがローカル機密保護検査に合格します。

3. Windows NT の機密保護の詳細については、383ページの『付録E. DB2 (Windows NT 版) が Windows NT 機密保護を処理する方法』を参照してください。

リモート・クライアントでの認証についての考慮事項

リモート・アクセスのためにデータベースをカタログ化する場合、認証タイプをデータベース・ディレクトリー項目の中に指定することができます。

DB2 コネクトを使用してアクセスされるデータベースの場合、値が指定されないと、SERVER 認証が使用されます。

リモートにアクセスされるが、 DB2 コネクトを使用していないデータベースの場合、認証タイプは必要ありません。しかし、認証タイプが指定されていないと、クライアントは、認証のフローを開始する前に、まずサーバーにコンタクトして値を入手しなければなりません。認証タイプが指定されていると、指定された値がサーバーのものと一致すれば、認証を即時に開始することができます。サーバーとの不一致が検出された場

合、DB2 は回復を試み、値を一致させるためのさらに多くのフローが実行されるか、または DB2 が回復できなければエラーになります。 不一致がある場合は、サーバーにある値の方が正しいとみなされます。

区分データベースの考慮事項

区分データベースでは、データベースの各区画に、同じ組のユーザーとグループが定義されていなければなりません。定義が同じでないと、ユーザーは、異なる区画で異なることを実行できるように許可されてしまうことがあります。すべての区画にわたって一貫していることが推奨されます。

ユーザーの認証のための DCE 機密保護サービスの使用

分散データベース環境の機密保護を考慮する場合、分散コンピューティング環境 (DCE) 機密保護サービスを使用するのが優れた選択です。これは、DCE が以下のものを提供するためです。

- ユーザーおよびパスワードの管理を集中化する。
- 生のテキストでパスワードおよびユーザー ID を伝送することがない。
- ユーザーにとって単一のサインオンで済む。

DB2 は、DCE のデフォルト・ログイン・コンテキスト、接続ログイン・コンテキスト、および代行ログイン・コンテキストをサポートします。*デフォルト・ログイン・コンテキスト* は、ユーザーがクライアントで `dce_login` を行ったときに確立されます。後続の DB2 コマンドはこのコンテキストにアクセスし、ユーザーによるそれ以上の介入なしで (つまり、ユーザー ID またはパスワードを必要とせずに)、ユーザー認証を行うことができます。*接続ログイン・コンテキスト* は、 USER/USING 文節を使用した CONNECT または ATTACH で提供されるユーザー ID およびパスワードを使用して、DB2 セッションに対して確立されます。また、*代行ログイン・コンテキスト* は、 DB2 クライアントが DCE サーバー・アプリケーションの一部として使用された場合に行われます。 DCE サーバー・アプリケーション (これも 1 つの DB2 クライアントです) は、 DCE クライアント・アプリケーション (そのポイントから、ユーザーのオリジナルのアイデンティティーが出されます) から要求を受け取ります。 DCE サーバーが DCE クライアントを代行できるよう、 DCE クライアントと DCE サーバーが正しく構成されている場合、DB2 は、代行トークンを取得して、それを DB2 サーバーに転送します。これによって、DB2 サーバーは、DCE サーバーのアイデンティティーを使用するのではなく、 DCE クライアントのオリジナルのアイデンティティーを使用して、要求を処理することができます。代行ログイン・コンテキストを確立する方法については、使用しているプラットフォームの DCE の資料に説明があります。

注: DCE をサポートしているベンダー製品がいくつかあります。機密保護サービス域で IBM の DCE 製品を使用して作業できることを保証するために、 2 つの新しい DLL が提供されています。それは、`db2dces.ibm` と `db2dcec.ibm` です。 (これら

の DLL ファイルは、Windows NT でのみ使用できます。) 機密保護サービスのために IBM の DCE 製品を購入して使用する場合には、これら 2 つのファイルを db2dces.dll および db2dcec.dll にそれぞれコピーしてください。別のベンダーの DCE 製品を使用することを考慮している場合は、そのベンダーのサービス部門と DB2 UDB のサービス部門と連絡を取って、機密保護サービスのためにそのベンダーの DCE を実装することが、 DB2 UDB で効果があるかどうかを話し合う必要があります。

DCE 用に DB2 ユーザーをセットアップする方法

ユーザーは、DB2 で使用される前に、分散コンピューティング環境 (DCE) レジストリーに登録され、正しい属性を持っていなければなりません。 DCE プリンシパルを作成する方法については、該当のプラットフォーム固有の DCE の資料を参照してください。

DCE 認証サーバーを希望する各 DB2 ユーザーは、DCE プリンシパルとアカウントが、クライアント・フラグを使用可能にして DCE レジストリーの中に定義されていなければなりません。このプリンシパルはまた、拡張レジストリー属性 (ERA) セクションに、特定の DCE 認証サーバーに接続した場合にこのプリンシパルに対してどの許可名が使用されるかを示す項目を持っていなければなりません。

また、データベース内でグループ特権を使用するために、ユーザー・プリンシパルをグループのメンバーにしたい場合もあります。グループ ERA 内の類似の情報が、グループ名を DB2 許可名にマップします。許可名は 2 次許可名ですが、同じ制約事項が適用されます。グループの作成およびメンバーの追加の方法についての詳細は、DCE の資料を参照してください。

ERA の中の情報は、特定のサーバー DCE プリンシパル名について、ユーザーの DCE プリンシパルまたはグループ名を DB2 許可名にマップします。 ERA を使用するためには、この属性の形式を示す ERA スキーマを定義しなければなりません。これは、DCE セルごとに 1 回行う必要があり、以下のステップを行うことによって達成されます。

1. 有効な DCE 管理者として DCE にログインする。
2. dcecp を呼び出し、プロンプトに以下のとおり入力する。

```
> xattrschema create /.:/sec/xattrschema/db2map ¥
> -aclmgr {{principal r m r m } {group r m r m }} ¥
> -annotation {Schema entry for DB2 database access} ¥
> -encoding stringarray ¥
> -multivalued no ¥
> -uuid 1cbe84ca-9df3-11cf-84cd-02608c2cd17b
```

これは、拡張レジストリー属性 db2map を作成します。

このマッピングを表示するには、dcecp プロンプトで以下のコマンドを出します。

```
> xattrschema show /.:/sec/xattrschema/db2map
```

以下のように表示されます。

```
{axlmgr
 {{principal {{query r} {update m} {test r} {delete m}}}
  {group     {{query r} {update m} {test r} {delete m}}}}}
{annotation {Schema entry for DB2 database access}}
{applydefs no}
{intercell rejects}
{multivalued no}
{reserved no}
{scope {}}
{trigbind {}}
{trigtype none}
{unique no}
{uuid 1cbe84ca-9df3-11cf-84cd-02608c2cd17b}
```

注: ERA に記録された許可名の内容についての制約事項は、 DCE によって施行されません。 DCE プリンシパルまたはグループに無効な許可名が与えられた場合、ユーザーを認証する試みが DB2 によって行われたときにエラーになります。 (認証は、CONNECT、ATTACH、DB2START、または認証が必要な他のいずれかの操作で行われることに注意してください。) また、DCE プリンシパルに対する許可名の割り当ては、1 対 1 の対応を持ち、固有なものになるようにすることを強くお勧めします。 DCE は、これらの条件を検査しません。

DB2 クライアントが DB2 UDB サーバーにアクセスする場合、 DB2 クライアントが DCE プリンシパルとして登録されると、プリンシパル名から許可名へのマッピングを提供するために、 ERA 情報を追加しなければなりません。これは、各ユーザーまたはグループに対して 1 回行われますが、以下のステップを行うことによって達成されます。

- 有効な DCE 管理者として DCE にログインする。
- dcecp を呼び出し、プロンプトで以下のとおり入力する。

  ```
  > principal modify principal_name ¥
  > -add {db2map map_1 map_2...map_n}
  ```

  ここで、map_n は、以下の形式を使用します。

  ```
  DCE_server_principal,DB2_authid
  ```

  ただし、DCE_server_principal は DB2 UDB サーバーに対する有効な DCE プリンシパル名 (または、このマッピングが、別の map_n 項目にすでに指定されたものでない任意の DB2 サーバーに対して有効であることを示すワイルドカード *) であり、DB2_authid は有効な DB2 許可名です。

  DCE グループを DCE プリンシパルに使用するつもりである場合、DCE グループには、 SYSADM または SYSCTRL 権限などの適切な権限を持つ DB2 authid へのマッピングもなければなりません。

DCE プリンシパル名を DB2 authid にマップするのに使用される DCE スキーマで指定される許可識別子 (authid) は、英大文字で記入する必要がある ことにぜひ注意してください。小文字の使用または大文字小文字の混合は、エラーとなります。

DCE を使用するための DB2 サーバーのセットアップ方法

サーバーは、DB2 で使用される前に、分散コンピューティング環境 (DCE) レジストリーの登録済みプリンシパルでなければならず、正しい属性を持っていなければなりません。 DCE サーバー・プリンシパルを作成する方法については、該当のプラットフォーム固有の DCE の資料を参照してください。

DCE 機密保護クライアントの実行時コードはインストールされていて、サーバー・インスタンスによってアクセス可能でなければなりません。

認証メカニズムとして DCE を使用したい各 DB2 サーバーは、 DB2START を出したときに DCE に登録しなければなりません。これを手操作で行うのを避けるために、DCE は、サーバーが *keytab* ファイルと呼ばれる特殊なファイルの中に自身のユーザー ID およびパスワード (鍵) を維持する方式を提供します。 DB2START 時に、DB2 は、データベース・マネージャー構成ファイルを読み取り、インスタンスの認証タイプを獲得します。認証タイプが DCE であることがわかると、keytab ファイルから情報を獲得するために、 DCE 呼び出しが DB2 サーバーによって行われます。サーバーを DCE に登録するために使用されるのは、この情報です。この登録によって、サーバーは、DCE クライアントから DCE トークンを受け入れ、そのトークンをこれらのユーザーの認証に使用することができます。

インスタンス管理者は、DCE コマンドを使用して、インスタンスに対する keytab ファイルを作成しなければなりません。 keytab ファイルの作成方法についての詳細は、使用しているプラットフォーム用の DCE の資料に説明があります。その資料の中で、keytab ファイルと、 *dcecp keytab* コマンドまたは *rgy_edit* コマンドに関連する詳細説明を参照してください。 DB2 keytab ファイルは、*keytab.db2* という名前でなければならず、インスタンスの `sqllib` ディレクトリーの `security` サブディレクトリーに常駐していなければなりません。 (Intel ベースのオペレーティング・システムの場合、このファイルは、 `sqllib` ディレクトリーの `INSTANCENAME` サブディレクトリーの `security` サブディレクトリーの中に常駐していなければなりません。 `INSTANCENAME` は、作業しているインスタンスのインスタンス名です。) 指定されたインスタンスについて、サーバー・プリンシパルに対して 1 つだけの項目が含まれている必要があり、そうでないと、DB2START 時にエラーになります。 UNIX オペレーティング・システムのプラットフォームでは、このファイルは、インスタンス所有者の読み取り / 書き込みだけが許されるファイル許可によって保護されていなければなりません。

以下は、keytab ファイルを作成する例です。

- 有効な DCE ユーザーとして DCE にログインする。
- `rgy_edit` を呼び出し、プロンプトで以下のとおり入力する。

```
> ktadd -p principal_name -pw principal_password ¥
> -f keytab.db2
```

DCE 構成が完了してから DCE 認証を使用して DB2 を開始するためには、データベース・マネージャー構成ファイルを更新して認証タイプを 『DCE』 にすることによって、 DB2 に DCE 認証を使用することを通知しなければなりません。これは、以下の CLP コマンドを出すことによって行われます。

```
db2 update database manager configuration using authentication DCE
 sysadm_group DCE_group_name
```

次に、SYSADM 権限を持つ有効な DB2 DCE ユーザーに対して dce_login を実行し、DB2START を出します。

注: DCE 認証を使用して DB2 を開始する前に、そのインスタンスに対する SYSADM として使用される DCE ユーザー・プリンシパルを定義して、インスタンスの開始、停止、および管理を行える有効な DCE ユーザー ID を持てるようにしてください。これを行う方法についての指示は、242ページの『DCE 用に DB2 ユーザーをセットアップする方法』を参照してください。

これらの指示に加えて、作成されたプリンシパルが、そのインスタンスに対する SYSADM_GROUP のメンバーになるようにしてください。デフォルトでは、このグループ名は、グループが明示して指定されていない場合 (つまり、SYSADM_GROUP がヌル値である場合)、DCE 認証に対して DB2ADMIN になります。ただし、インスタンスの認証タイプを変更する前に、好みのグループ名 (許可名) に更新することができます。選択した DCE グループには、その DCE グループを、指定された SYSADM_GROUP 許可名にマップする ERA が定義されていなければなりません。

DB2 管理サーバーの関数の 1 つは、DB2 インスタンスを開始することです。AUTHENTICATION = DCE であるとき、このインスタンス用 DB2 keytab ファイルで使用される DCE プリンシパルは、有効な DCE プリンシパルを DB2 authid にマッピングさせる必要があります。このマッピングは、DB2 インスタンスを開始するため DB2 管理サーバーで必須となります。有効なマッピングによって、この ID はクライアントとしてもサーバーとしても機能できます。

DCE を使用するために DB2 クライアント・インスタンスをセットアップする方法

データベース・マネージャー構成ファイルを更新し、認証タイプを DCE に設定することによって、クライアントのみのインスタンスを、ローカル操作に対して DCE 認証を使用するように設定することができます。 DCE に登録する必要があるサーバーがないため、クライアントのみのインスタンスについては、keytab ファイルは必要ありません。一般に、クライアントのみの DB2 インスタンスが DCE 認証を使用することは推奨されません (または必須ではありません) が、サポートはされています。

DCE 機密保護を使用してリモート・データベースへのアクセスを希望するクライアントは、適切な DCE 機密保護製品にアクセスする必要があります。クライアントは、ターゲット・データベースの認証タイプをデータベース・ディレクトリーにカタログすることを選択することができます (この選択は任意です)。クライアントが DCE 認証を指定することを選択した場合、 DCE サーバー・プリンシパルの完全修飾名も指定しなければなりません。 DCE 認証がディレクトリーに指定されていない場合、認証およびプリンシパルの情報は、 CONNECT 時にサーバーから取得されます。

DCE 機密保護を使用した DB2 の制約事項

DCE 認証を使用する場合、DB2 によって提供され、グループ・サポートに関連する特定の SQL 機能について、いくつかの制約事項があります。以下の制約事項は、DCE 認証を使用している場合に存在します。

- GRANT または REVOKE ステートメントを使用している場合、 USER および GROUP というキーワードを、指定された許可名を修飾するために指定**しなければなりません**。そうでないと、エラーが出されます。
- CREATE SCHEMA ステートメントの AUTHORIZATION 文節を使用している場合、指定された許可名のグループ・メンバーシップは、この文節の後にあるステートメントを実行するために必要な許可を評価するときには考慮**されません**。これによって、CREATE SCHEMA ステートメントの実行中に許可の障害が発生することがあります。
- パッケージを最初にバインドしたユーザー以外のユーザーによってパッケージが再バインドされる場合、最初にバインドしたユーザーの特権が再評価されます。この場合、最初にバインドしたユーザーのグループ・メンバーシップは、特権の再評価のときには考慮されません。これによって、再バインド時に許可の障害が発生することがあります。

DB2 で実行されるような DCE 認証は、OSF DCE 汎用機密保護サービスのアプリケーション・プログラミング・インターフェース (GSSAPI) を使用して獲得された DCE チケットを発行します。したがって、DCE 機密保護のすべての認証はそのデータベースのプロトコル層で行われます。ある通信メカニズムでは、付加的な通信層の機密保護 (DCE に必ずしも内蔵されるとは限らない) を提供する場合があります。通信層の認証がデータベース・プロトコル層の認証から完全に独立した状態を維持できる場合には、どんな制約事項も実施されません。ただし、接続が正常に確立される前に、データベース・プロトコル層の認証と通信層の認証の両方の基準が満たされなければなりません。データベース・プロトコル層および通信層の認証メカニズムが相互作用する場合で、ある組み合わせが機密漏れをもたらすならば、それらの使用は制限されることがあります。

DCE 認証は TCPIP SOCKS サポートと共同して使用する場合がありますが、その 2 つの機密保護メカニズムは互いに独立して働きます。これは、ユーザーが有効な DCE ロ

グイン・コンテキストを備えるだけではなく、さらに SOCKS サーバーの基準に一致するローカルのオペレーティング・システムのユーザー ID にログオンする必要があることを示しています。

DCE 認証は NT 名前付きパイプと共同して使用する場合がありますが、その 2 つの機密保護メカニズムは互いに独立して働きます。ユーザーは有効な DCE ログイン・コンテキストを備えるだけではなく、さらに NT 名前付きパイプ・サポートの基準に一致するユーザー ID に NT ドメインをログオンする必要があります。

上記 2 つの例に示されているように、 DCE プリンシパルおよびローカル・オペレーティング・システムのユーザー ID の両方を認証のために使用する際に考えられる混乱に取り組むため、統合された DCE を用いることができます。この場合さらに、システムにログオンする際、ユーザーは適切な DCE プリンシパルに自動的に記録されます。この機能の使用方法に関する詳細については、それがサポートされている場合、ご使用のプラットフォームの DCE 資料をご覧ください。このアプローチを使う際、同一の名前が DCE プリンシパルおよびローカルのオペレーティング・システム ID に使用されることに注意してください。このことは、DCE の暗号化されたチケットに含まれる同一の値が、通信層の暗号化されていないワイヤーに発行されることもあることを示しています。

SECURITY パラメーターが NONE に設定されるとき、 DCE 認証は APPC 通信によってのみ使用することができます。これは、通信層における暗号化されないプリンシパルまたはパスワード、あるいはその両方の送信の可能性を避けるためです。一方、データベース・プロトコル層には同一プリンシパルの暗号化された DCE トークンを使用しています。 APPC 層の DCE 機密保護は、この時点では DB2 によりサポートされていません。

連合データベースの認証処理

配布された結合インストール機能をインストールしてあり、*統合された* データベース・マネージャー構成変数を 'YES' に設定してある場合、 DB2 システムは連合システムとして機能します。連合システム中のデータベース認証設定は、標準の DB2 定義と少し異なります。最も重要な点として、連合システムではデータ・ソースの認証要件を考慮する必要があります。一般には、データ・ソース (DB2、 Oracle、 DB2 (OS/390 版) など) は、必須認証にセットアップされます。つまり、ID およびパスワード (必須) が、必ずデータ・ソースに流れるようにする必要があります。ここでは DB2 が提供する、データ・ソースで認証をサポートするための方法をすべて説明します。

認証の設定

SERVER

DB2 に接続するクライアントが、 DB2 にアクセスするためのユーザー ID とパスワードを提供することを指定します。この場合、ユーザー ID とパスワードは、データ・ソースへの伝送に使用できます。実際にはサーバー・オプショ

ンとユーザー・マッピングを介してデータ・ソースに渡されるものを制御しますが、データ・ソースへの伝送には認証情報が使用可能です。

CLIENT

オペレーティング・システムの機密保護を使用して、アプリケーションが呼び出されたデータベース区画上で認証が行われることを指定します。データ・ソースへの直接の伝送に、パスワードは使用できません。この場合、データ・ソースに認証が必要であれば、 1 つ以上のユーザー・マッピングを作成する必要があります。また、サーバー・オプションが、確実に正しいユーザー ID およびパスワード情報をデータ・ソースに伝送するように設定する必要もあります。

CLIENT 認証の使用時には、細心の注意を払ってください。この形式の認証は、ネットワークの保護の場合のみ考慮します。次の条件が満たされる場合、ユーザーには連合データベースの SYSADM 権限があります。

- 認証が CLIENT に設定されている。
- ユーザーに、クライアントでのルート・ステータスがある。
- ユーザーが SYSADM の許可名を知っている。
- ユーザーが DB2 上の SYSADM と同じクライアントで許可名を定義する。

DCS 認証が、DB2 ではなくデータ・ソースで行われるように指定します。この場合、標準 DB2 認証処理はう回されます。ユーザー ID およびパスワードは、サーバー・オプション設定に基づいてデータ・ソースに直接渡されます。認証は、Oracle または DB2 ファミリーのデータ・ソースでのみ行われます。

認証が DCS に設定される場合、細心の注意を払ってください。認証は、クライアントでも DB2 でも行われません。 SYSADM 許可名を知っているユーザーは、統合サーバー用の SYSADM 権限を持っているとみなすことができます。

DCE 認証が DCE に設定されると、ユーザー ID だけがデータ・ソースへの伝送に使用可能になります。パスワードは使用できません。データ・ソースに認証処理 (ユーザー ID およびパスワード) が必要な場合、パスワード (およびおそらくユーザー ID も) をデータ・ソースに伝送する、ユーザー・マッピングを定義することが必要です。データ・ソースが DB2 接続を承認する場合、外部機密保護システムから受け取る ID をデータ・ソースに渡すことができるので、ユーザー・マッピングは必要ありません。

他の DB2 認証設定も可能であり、1 つ以上がデータ・ソースへの伝送用の DB2 で、パスワードが使用可能であるという結果になります。 DB2 およびクライアント認証設定が DB2 へのパスワードの伝送という結果になる場合、そのパスワードはデータ・ソースでの付加的な認証処理で使用可能です。詳細については、237ページの表4 を参照してください。

ユーザー ID とパスワードをデータ・ソースに渡す

データ・ソースへの認証情報の伝送を制御するには、4 つの方法があります。 DB2 認証設定、ユーザー・マッピング、サーバー・オプション、および APPC 機密保護設定です。

認証の設定

この節の目的は、認証の設定が連合システムのグローバル認証処理に、どのような影響を与えるかを分かりやすく説明することです (認証の設定の定義については、 247ページの『認証の設定』を参照してください)。たとえば、DB2 認証が SERVER または DCS である場合、ユーザー ID とパスワードが接続に必要です。したがってユーザー ID とパスワードは、データ・ソースへの伝送に使用できます。認証が DCE または CLIENT に設定され、認証が連合データベースを含む DB2 システムでは行われない場合、ユーザー ID だけが使用可能になります。データ・ソース認証処理にパスワード (または、場合によっては異なるユーザー ID とパスワード) が必要な場合、ユーザー・マッピングを作成する必要があります。認証が CLIENT に設定され、 *trust_clntauth* パラメーター設定が SERVER である場合、 DB2 にパスワードを送信し、データ・ソースへの伝送に使用可能にすることができます。

ユーザー・マッピング

DB2 は、DB2 への接続に使用する許可名、または DB2 で定義される許可名のどちらかを送信できます。ユーザー・マッピングは DB2 で定義された許可名を保管します。これらの名前は CREATE USER MAPPING ステートメントで作成されます。

ユーザー・マッピングは柔軟です。ID を新しい ID とパスワード、または単にパスワードだけにマップすることができます。これらを使って欠落している情報を提供したり、 ID およびパスワードをデータ・ソースが受け入れる値に変更したりすることができます。

ユーザー・マッピングを作成または変更するには、 SYSADM または DBADM 権限のうちの 1 つを保持しているか、または認証 ID がステートメントに指定される許可名と一致していなければなりません。

ユーザー・マッピングのステートメント例を次に示します。

```
CREATE USER MAPPING FOR "SHAWN" SERVER DB21 OPTIONS (REMOTE_AUTHID "SHAWNBCA",
REMOTE_PASSWORD "MAPLELEAF")
```

ここで、DB2 認証 ID (SHAWN) が、リモート ID SHAWNBCA と、 DB21 という名前のサーバーのリモート・パスワード MAPLELEAF にマッピングされます。

DB2 での許可名 (またはパスワード) とデータ・ソースでの許可名 (またはパスワード) の相違点が、渡されるストリングが大文字か小文字かというだけのものである場合、新しい ID とパスワードを作成するよりも、サーバー・オプションを使用して、必要な設定に文字を変換することを考慮してください。詳細については、250ページの『サーバー・オプション』を参照してください。

認証設定が DCE で、データ・ソースに認証処理 (パスワードが求められる) が必要な場合、ユーザー・マッピングを作成する必要があります。 DB2 は DCE ユーザー ID をデータ・ソースに渡すだけです。パスワードをそのユーザー ID にマップしてから、データ・ソースに送信することが必要です。

サーバー・オプション

サーバー・オプションは、全体的な認証サポートを提供するのに使用できます。サーバー・オプションを使って、パスワードがデータ・ソースに渡されるかどうか (通常は渡される)、およびユーザー ID とパスワードを大文字または小文字に変換する必要があるかどうかを示します。サーバー・オプションは、CREATE SERVER、ALTER SERVER、および SET SERVER OPTION ステートメントを使って設定します。

認証処理に特定のサーバー・オプションについては、この節の残りの部分で説明します。サーバー・オプションのより完全なリストは、160ページの『データ・ソースの定義に役立ち、認証処理を容易にするサーバー・オプションの使用』に掲載されています。

パスワード・サーバー・オプション: パスワードのデフォルト設定は 'Y' です (パスワードはデータ・ソースに送信されます)。データ・ソースが認証を実行し、暗号化されたパスワードを要請していない場合はすべて、このオプションを 'Y' のままに、または 'Y' に設定してください。

DB2 は、暗号化されたパスワードを伝送できます。パスワードが暗号化された形式で DB2 ファミリーのデータ・ソースに送信される必要がある場合、サーバー・オプション・パスワードを 'ENCRYPTION' に設定します。 DB2 での認証設定が DCS_ENCRYPT または SERVER_ENCRYPT である場合、パスワードを 'ENCRYPTION' に設定することをお勧めします。

ユーザー ID は常にデータ・ソースに送信されます。

ID およびパスワードの大文字への変換オプション: 許可名とパスワードには、場合によっては変更が必要です。異なるデータ・ソースの許可名およびパスワード要件では、 ID およびパスワードが (大文字または小文字の使用に関して) 異なることがあります。

DB2 は、名前の相違を解決するのに役立つ 2 つのオプションを備えています。オプション名は **fold_id** および **fold_pw** で、設定は次のとおりです。

'U' DB2 は、パスワードをデータ・ソースに送信する前に、許可名またはパスワードを大文字に変換します。

'N' DB2 は許可名またはパスワードの大文字小文字を変換しません。

'L' DB2 は、パスワードをデータ・ソースに送信する前に、許可名またはパスワードを小文字に変換します。

null DB2 は、まず許可名またはパスワードを大文字で送信します。失敗すると、DB2 はそれを小文字に変換し、再度送信します。

null 設定は、多くの可能性をカバーするので、好ましく思えるかもしれません。しかし、パフォーマンスの見通しから考えると、接続に対してなされる試みが 1 つだけになるようにこれらのオプションを設定するのが最善です。 fold_id および fold_pw オプションが null に設定される場合、 DB2 が 4 つの試みを行って許可名とパスワードを送信することが可能です。

1. 許可名とパスワードの両方を大文字にする。
2. 許可名を大文字、パスワードを小文字にする。
3. 許可名を小文字、パスワードを大文字にする。
4. 許可名とパスワードの両方を小文字にする。

APPC 機密保護設定

APPC を介してユーザー ID とパスワードが必要な DRDA データ・ソースへ接続している場合、または認証設定が DCS で DRDA データ・ソースで認証を行っている場合、 DB2 およびそのデータ・ソース間の APPC 機密保護設定を、必ず PROGRAM にしてください。

連合データベース認証の例

この節では、連合システムの認証および許可ステップの概要について説明します。連合データベースの認証および許可処理の概要については、 252ページの図3 を参照してください。

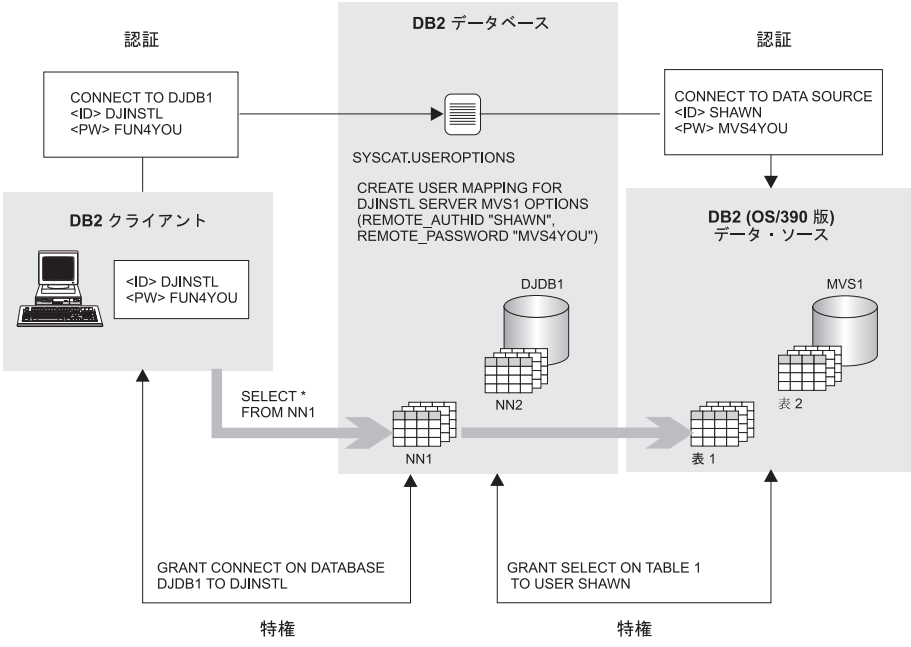


図 3. 連合データベースの認証および許可の処理

このシナリオのタスクは、ユーザー DJINSTL が 2 つのニックネーム (NN1 と NN2) に対して UNION 操作を実行できるようにすることです。ニックネームは、2 つの表を表します。 1 つのデータ・ソースは、DJINSTL が異なるユーザー ID と、 MVS1 というパスワード (図3 を参照) を持つ、DB2 (OS/390 版) です。 MVS1 で情報にアクセスするには、ユーザー・マッピングが必要です。他のデータ・ソースは、DJINSTL の ID とパスワードが同じである DB2 システムです。このデータ・ソース、DB21 で必要なのは、単にユーザー ID とパスワードを大文字で送信することだけです。

DB2 認証は SERVER に設定されます。 DJINSTL は TCP/IP 接続を介して Windows NT クライアントから DB2 にアクセスします。 DB2 から DB2 (OS/390 版) への接続も TCP/IP です。連合データベース名は DJDB1 です。

まず、DB2 がパスワードを求めていること、そしてパスワードが送信されていることを確認します。また、クライアントおよびサーバー認証のタイプは必ず一致させます。次のコマンドを発行して、DB2 サーバー認証タイプを検査します。

```
GET DATABASE MANAGER CONFIGURATION
```

この検査は DB2 サーバーから行います。次のコマンドを発行して、クライアント認証タイプを検査します。

```
LIST DATABASE DIRECTORY
```

この検査はクライアントから行います。どちらの場合も、認証は確実に SERVER に設定してください。クライアントの設定が DCS または CLIENT である場合、UNCATALOG DATABASE および CATALOG DATABASE コマンドを使用して設定を変更できます。

次に、パスワードを確実にデータ・ソースに送信します。連合データベース DJDB1 に接続した後、次のコマンドを発行します。

```
ALTER SERVER MVS1 OPTIONS (SET password 'Y')
ALTER SERVER DB21 OPTIONS (SET password 'Y')
```

次に、パスワードを適切な文字ケースで DB21 データ・ソースに送信します。

```
ALTER SERVER DB21 OPTIONS (ADD fold_id 'U')
ALTER SERVER DB21 OPTIONS (ADD fold_pw 'U')
```

次のステップは、連合データベース DJDB1 に接続し、ニックネームを選択するために、ユーザー DJINSTL を許可する特権を付与することです。

```
GRANT CONNECT ON DATABASE DJDB1 TO DJINSTL;
```

次に、DJINSTL の DB2 ID およびパスワードを、 MVS1 サーバーの正しいユーザー ID およびパスワードにマップします。

```
CREATE USER MAPPING FOR "DJINSTL" SERVER MVS1 OPTIONS (REMOTE_AUTHID "SHAWN",
REMOTE_PASSWORD "MVS4YOU")
```

この時点で、DB2 ユーザー ID DJINSTL は、データ・ソースに要求を送信できます。ニックネームが参照するデータ・ソース・オブジェクトにアクセスするには、さらに付加的なステップが必要になることがあります (ニックネームが参照する表、および視点には、通常は特権が必要です)。

特権、権限、および許可

特権 は、ユーザーがデータベース・リソースを作成したりデータベース・リソースにアクセスしたりすることを許可するためのものです。*権限レベル* によって、特権のグループ分けの方法、およびより高いレベルのデータベース・マネージャーの保守とユーティリティー操作が得られます。それらが一緒になって、データベース・マネージャーとそのデータベース・オブジェクトへのアクセスを制御する働きをします。ユーザーがアクセスできるのは、該当する許可、つまり、必須の特権や権限を持っているオブジェクトに限られます。

権限には、次の種類があります。

- 255ページの『システム管理権限 (SYSADM)』
- 256ページの『システム制御権限 (SYSCTRL)』
- 257ページの『システム保守権限 (SYSMAINT)』
- 258ページの『データベース管理権限 (DBADM)』

- 259ページの『LOAD 権限』

特権には次の種類があります。

- 259ページの『データベースの特権』
- 261ページの『スキーマの特権』
- 263ページの『表スペース特権』
- 263ページの『表および視点の特権』
- 265ページの『ニックネームの特権』
- 266ページの『サーバー特権』
- 266ページの『パッケージの特権』
- 267ページの『索引の特権』

図4 は、権限とその制御の範囲 (データベース、データベース・マネージャー) の間の関係を示します。

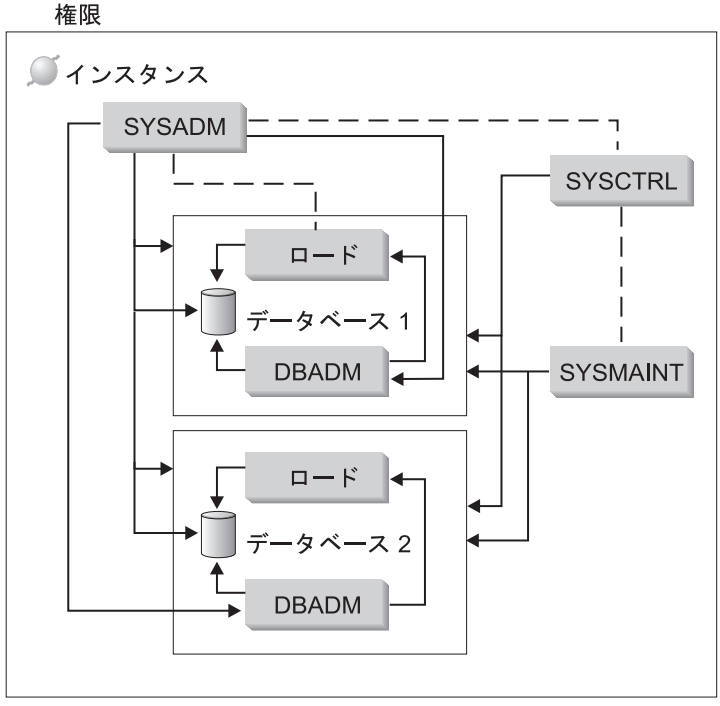


図 4. 権限の階層

ユーザーまたはグループは、次のレベルの権限許可を 1 つ以上持つことができます。

- 管理権限 (SYSADM または DBADM)。これは、一組のオブジェクトについての完全な特権です。

- システム権限 (SYSCTRL または SYSMAINT)。これは、システム管理のための完全な特権ですが、データへのアクセスはできません。
- LOAD 権限 (LOAD)。これは、データを表にロードする特権を LOAD ユーティリティーまたは AutoLoader ユーティリティーに与えます。
- 所有権特権 (場合によっては、 CONTROL 特権とも呼ばれる)。これは、特定のオブジェクトのための完全な特権です。
- 個別特権。これは、ユーザーが特定オブジェクトに対して特定機能を実行できるようにするためのものです。
- 暗黙特権。これは、パッケージを実行する特権を持つユーザーに与えられるものです。ユーザーがアプリケーションを実行できる場合でも、パッケージ内で使用されるデータ・オブジェクトに対する明示特権が必要であるとは限りません。詳しくは、272ページの『パッケージによる間接特権の許可』を参照してください。

管理権限 (SYSADM または DBADM) または所有権特権 (CONTROL) を持ったユーザーは、 GRANT および REVOKE ステートメントを使用して、他のユーザーとの間で特権の付与および取り消しを行うことができます。 (267ページの『データベース・オブジェクトに対するアクセスの制御』を参照。) 特権が WITH GRANT OPTION を使用して保持されているものである場合、別のユーザーに、表、視点、またはスキーマの特権を付与することもできます。ただし、WITH GRANT OPTION では、特権を付与する人が、いったん付与した特権を取り消すことは許されません。特権を取り消すためには、SYSADM 権限、DBADM 権限、または CONTROL 特権を持っていなければなりません。

1 つのユーザーまたはグループに対して、個々の特権または権限をいくつか組み合わせて許可することもできます。特権をリソースに関連付ける場合、そのリソースは存在していなければなりません。たとえば、表がそれ以前に作成されているのでなければ、その表についての SELECT 特権をユーザーに与えることはできません。

注: ある許可名が権限と特権を与えられ、しかもその許可名で作成されたユーザーがいない場合には、注意が必要です。後で、その許可名を使用してユーザーが作成され、その許可名に関連するすべての権限と特権を自動的に受け取る可能性があります。

特定のコマンド、API、または SQL コマンドについて、どんな許可が必要かについては、*コマンド解説書*、*管理 API 解説書*、または *SQL 解説書* を参照してください。

システム管理権限 (SYSADM)

SYSADM 権限は、最高レベルの管理権限です。 SYSADM 権限を与えられたユーザーは、ユーティリティーを実行したり、データベースおよびデータベース・マネージャーのコマンドを発行したり、データベース・マネージャー・インスタンス内のデータベースの表データにアクセスしたりできます。この権限は、インスタンス内のすべてのデータベース・オブジェクトを制御します。制御されるデータベース・オブジェクトには、

データベース、表、視点、索引、パッケージ、スキーマ、サーバー、別名、データ・タイプ、関数、プロシージャー、トリガー、表スペース、ノードグループ、バッファー・プール、およびイベント・モニターがあります。

SYSADM 権限は、 *sysadm_group* 構成パラメーター (*管理の手引き: パフォーマンス* の『DB2 の構成』を参照) によって指定されたグループに割り当てられます。このグループのメンバーシップは、データベース・マネージャーの外で、プラットフォームで使われている機密保護機能によって制御されます。 SYSADM 権限の作成、変更、または削除のためにシステム機密保護機能を使用する方法については、 *概説およびインストール* を参照してください。

SYSADM 権限を持つユーザーだけが実行できる機能は、次のとおりです。

- データベースの移行
- データベース・マネージャー構成ファイルの変更 (SYSCTRL または SYSMAINT 権限のあるグループを指定することを含む)
- DBADM 権限の付与

さらに、SYSADM 権限を持つユーザーは、次の権限を持つユーザーの機能も実行できます。

- 『システム制御権限 (SYSCTRL)』
- 257ページの『システム保守権限 (SYSMAINT)』
- 258ページの『データベース管理権限 (DBADM)』

注: SYSADM 権限を持つユーザーがデータベースを作成すると、そのユーザーには、そのデータベースに対する明示的な DBADM 権限が自動的に付与されます。データベースの作成者が SYSADM グループから除去され、そのデータベースに DBADM としてアクセスすることも防止したい場合は、この DBADM 権限を明示的に取り消さなければなりません。

システム制御権限 (SYSCTRL)

SYSCTRL 権限 は、最高レベルのシステム制御権限です。この権限があると、データベース・マネージャーのインスタンスとそのデータベースに対して、保守およびユーティリティー操作を実行することができます。これらの操作はシステム・リソースに影響を及ぼす場合がありますが、データベース内のデータに対するアクセスは認められていません。システム制御権限は、重要データの入ったデータベース・マネージャーのインスタンスを管理するユーザーを対象としたものです。

SYSCTRL 権限は、 *sysctrl_group* 構成パラメーター (*管理の手引き: パフォーマンス* の『DB2 の構成』を参照) によって指定されたグループに割り当てられます。グループを指定すると、そのグループに属するものは、プラットフォーム上で使用される機密保護機能によって、データベース・マネージャーの外で制御されます。

SYSCTRL 以上の権限を持つユーザーだけが実行できることは、次のとおりです。

- データベース、ノード、または分散接続サービス (DCS) ディレクトリーの更新
- システムからのユーザーの切断
- データベースの作成または除去
- 表スペースの除去、作成、または変更
- 新しいデータベースへの復元

さらに、SYSCTRL 権限を持つユーザーは、 『システム保守権限 (SYSMAINT)』 権限を持つユーザーの機能を実行できます。

SYSCTRL 権限を持つユーザーは、データベースへの接続に関する暗黙の特権も持っています。

注: SYSCTRL 権限を持つユーザーがデータベースを作成すると、そのユーザーには、そのデータベースに対する明示的な DBADM 権限が自動的に付与されます。データベースの作成者が SYSCTRL グループから除去され、そのデータベースに DBADM としてアクセスすることも防止したい場合は、この DBADM 権限を明示的に取り消さなければなりません。

システム保守権限 (SYSMAINT)

SYSMAINT 権限は、2 番目のレベルのシステム制御権限です。この権限があると、データベース・マネージャーのインスタンスとそのデータベースに対して、保守およびユーティリティー操作を実行することができます。これらの操作はシステム・リソースに影響を及ぼす場合がありますが、データベース内のデータに対するアクセスは認められていません。システム保守権限は、重要データの入ったデータベース・マネージャーのインスタンス内のデータベースを保守するユーザーを対象としています。

SYSMAINT 権限は、 *sysmaint_group* 構成パラメーター (*管理の手引き: パフォーマンス* の『DB2 の構成』を参照) によって指定されたグループに割り当てられます。グループを指定すると、そのグループに属するものは、プラットフォーム上で使用される機密保護機能によって、データベース・マネージャーの外で制御されます。

SYSMAINT 以上の権限を持つユーザーだけが実行できることは、次のとおりです。

- データベースの構成ファイルの更新
- データベースまたは表スペースのバックアップ
- 既存のデータベースへの復元
- 順方向回復の実行
- インスタンスの開始または停止
- 表スペースの復元
- トレースの実行

- データベース・マネージャー・インスタンスまたはそのデータベースのデータベース・システム・モニター・スナップショットの取得

SYSMAINT、DBADM、またはそれ以上の権限を持つユーザーは、次のことを実行できます。

- 表スペースの状態の照会
- ログ活動記録ファイルの更新
- 表スペースの静止
- 表の再編成
- RUNSTATS ユーティリティーを使用してのカタログ統計の収集

SYSMAINT 権限を持つユーザーは、データベースへの接続に関する暗黙の特権も持っています。

データベース管理権限 (DBADM)

DBADM 権限は、2 番目にレベルの高い管理権限です。この権限は特定のデータベースにのみ適用され、ユーザーは、特定のユーティリティーを実行し、データベース・コマンドを出し、そしてデータベース内のどの表のデータにもアクセスすることができます。 DBADM 権限が付与されると、BINDADD、CONNECT、CREATETAB、CREATE_NOT_FENCED、および IMPLICIT_SCHEMA 特権も付与されます。 SYSADM 権限を持つユーザーだけが DBADM 権限の付与または取り消しを行えます。 DBADM 権限を持つユーザーは、データベースについての特権を他のユーザーに付与できます。また、だれが特権を付与したかにかかわりなく、ユーザーの特権を取り消すこともできます。

DBADM 以上の権限を持つユーザーだけが実行できることは、次のとおりです。

- ログ・ファイルの読み取り
- イベント・モニターの作成、活動化、および除去

DBADM、SYSMAINT、またはそれ以上の権限を持つユーザーは、次のことを実行できます。

- 表スペースの状態の照会
- ログ活動記録ファイルの更新
- 表スペースの静止
- 表の再編成
- RUNSTATS ユーティリティーを使用してのカタログ統計の収集

注: DBADM によって上記の機能を実行できるのは、 DBADM 権限が与えられているデータベースに対してだけです。

LOAD 権限

表に対する INSERT 特権の他に、データベース・レベルでの LOAD 権限も持っているユーザーは、 LOAD コマンドまたは AutoLoader ユーティリティーを使用して、データを表にロードすることができます。

表に対する INSERT 特権の他に、データベース・レベルでの LOAD 権限も持っているユーザーは、直前のロード操作でデータを挿入するロードを行った場合に、 LOAD RESTART または LOAD TERMINATE を行うことができます。

直前のロード操作でロード置換を行った場合、ユーザーは、DELETE 特権が許可されていないと、 LOAD RESTART または LOAD TERMINATE を行うことができません。

LOAD の一部として例外表が使用される場合、ユーザーには、その例外表に対する INSERT 特権が必要です。

この権限を持っているユーザーは、 QUIESCE TABLESPACES FOR TABLE、RUNSTATS、および LIST TABLESPACES コマンドを実行することができます。

データベースの特権

図5 は、データベースの特権を示します。

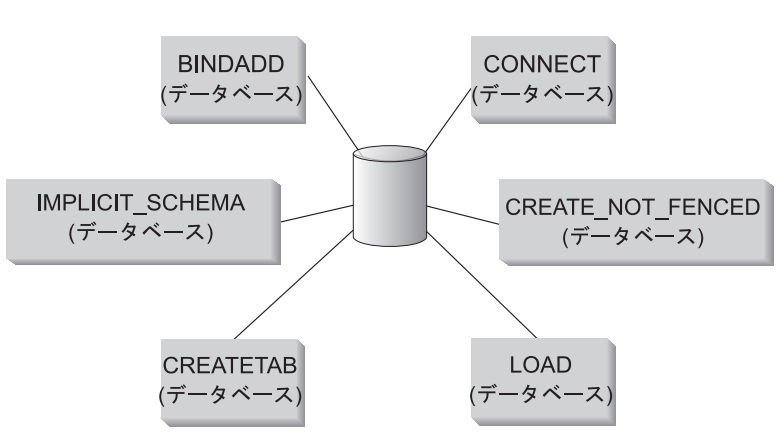


図 5. データベースの特権

データベースの特権には、データベース全体に対するアクションが関係しています。

- CONNECT: ユーザーはデータベースに接続できます。
- BINDADD: ユーザーはデータベースに新しいパッケージを作成できます。
- CREATETAB: ユーザーはデータベースに新しい表を作成できます。

- CREATE_NOT_FENCED: ユーザーは、『非分離』のユーザー定義関数 (UDF) またはプロシージャーを作成できます。『非分離』の UDF またはプロシージャーは、特に十分テストしなければなりません。これは、データベース・マネージャーはこれらの UDF またはプロシージャーの記憶域や制御ブロックを保護しないからです。 (このため、『非分離』での実行が許される UDF またはプロシージャーの作成およびテストが不十分であると、システムに重大な問題が起きることがあります。) (詳しくは、*アプリケーション開発の手引き* または *SQL 解説書* を参照してください。)
- IMPLICIT_SCHEMA: どのユーザーも、すでに存在していないスキーマ名を指定した CREATE ステートメントを使用してオブジェクトを作成することによって、暗黙にスキーマを作成することができます。 SYSIBM が暗黙に作成されたスキーマの所有者になり、 PUBLIC にこのスキーマ内にオブジェクトを作成するための特権が与えられます。
- LOAD: ユーザーは表にデータをロードできます。

他のユーザーに特権を付与したり他のユーザーから特権を取り消したりすることができるのは、 SYSADM または DBADM 権限を持つユーザーだけです。

注: データベース作成時に、以下の特権が自動的に PUBLIC に付与されます。

- CREATETAB
- BINDADD
- CONNECT
- IMPLICIT_SCHEMA
- USERSPACE1 表スペースに対する USE 特権
- システム・カタログ視点上での SELECT 特権

特権を除去するためには、 DBADM または SYSADM が明示的に PUBLIC から特権を取り消さなければなりません。

暗黙スキーマ権限 (IMPLICIT_SCHEMA) の考慮事項

新しいデータベースが作成されるとき、またはデータベースが以前のリリースから移行されるときに、 PUBLIC に IMPLICIT_SCHEMA データベース権限が与えられます。この権限を使用して、どのユーザーも、オブジェクトを作成し、すでに存在していないスキーマ名を指定することによって、スキーマを作成することができます。 SYSIBM が暗黙に作成されたスキーマの所有者になり、 PUBLIC にこのスキーマ内にオブジェクトを作成するための特権が与えられます。

だれが暗黙にスキーマ・オブジェクトを作成できるかを制御することがデータベースで必要な場合は、 IMPLICIT_SCHEMA データベース権限を PUBLIC から取り消す必要があります。いったんこれを行うと、スキーマ・オブジェクトが作成される方法は、以下の 3 つしかありません。

- どのユーザーも、CREATE SCHEMA ステートメントで自分自身の許可名を使用してスキーマを作成することができます。

- DBADM 権限を持つどのユーザーも、すでに存在していなければどのスキーマでも明示的に作成することができ、任意選択で、別のユーザーをそのスキーマの所有者として指定することができます。
- DBADM 権限をもつどのユーザーも (PUBLIC と独立して) IMPLICIT_SCHEMA データベース権限を持っているため、他のデータベース・オブジェクトを作成しているときに、任意の名前を持ったスキーマを暗黙に作成することができます。 SYSIBM が暗黙に作成されたスキーマの所有者になり、 PUBLIC がスキーマ内にオブジェクトを作成する特権を持ちます。

ユーザーは、自分自身の許可名を使用して、自分自身のスキーマを明示的に作成する能力を常に持っています。

スキーマの特権

スキーマ特権は、オブジェクト特権区分に入ります。オブジェクト特権は、
262ページの図6 に示されています。

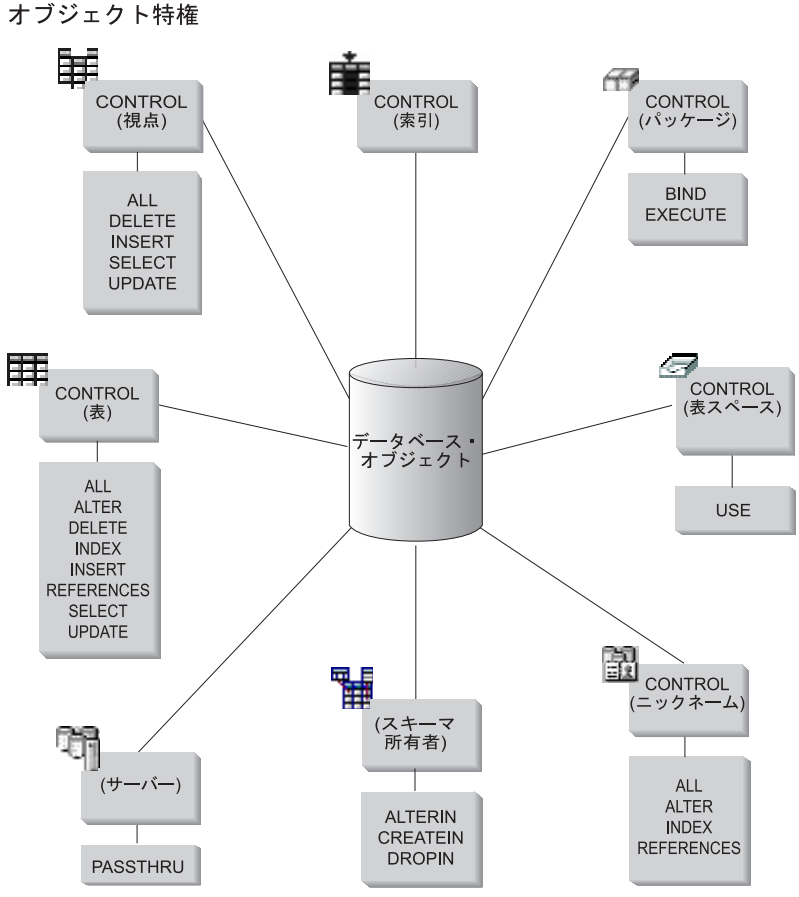


図 6. オブジェクト特権

スキーマの特権には、データベース内のスキーマ上でのアクションが含まれます。ユーザーには、以下の特権のどれでも付与することができます。

- CREATEIN により、ユーザーはスキーマ内にオブジェクトを作成できます。
- ALTERIN により、ユーザーはスキーマ内のオブジェクトを変更できます。
- DROPIN により、ユーザーはスキーマ内からオブジェクトを除去できます。

スキーマの所有者は、これらの特権をすべて持ち、その特権を他のユーザーに付与する能力を持ちます。スキーマ・オブジェクト内で操作されるオブジェクトには、表、視点、索引、パッケージ、データ・タイプ、関数、トリガー、プロシージャー、および別名があります。

表スペース特権

表スペース特権は、データベースの表スペースに対してアクションを実行することを可能にします。表スペースの USE 特権を付与されたユーザーは、その表スペース内で表を作成できます。

表スペースの所有者 (多くの場合、 SYSADM または SYSCTRL 権限を持っている作成者) は USE 特権を持っており、他のユーザーにこの特権を付与することができます。デフォルトでは、データベースの作成時に、表スペース USERSPACE1 に関する USE 特権が PUBLIC に付与されますが、この特権は取り消すこともできます。

USE 特権は、 SYSCATSPACE またはシステム一時表スペースでは使用できません。

表および視点の特権

表および視点の特権には、データベース内の表や視点に対するアクションが関係しています。次に挙げる特権のいずれかを使用するユーザーには、データベースについての CONNECT 特権が必要です。

- CONTROL: 表や視点についてすべての特権を持つことになり、表や視点の除去、表についての個別の特権の付与と取り消しなどができます。 CONTROL を付与するには、SYSADM または DBADM 権限が必要です。表の作成者は、自動的にその表の CONTROL 特権を受け取ります。視点の作成者は、視点定義の中で参照されているすべての表と視点に対する CONTROL 特権を持っているか、 SYSADM または DBADM 権限を持っている場合にのみ、自動的に CONTROL 特権を受け取ります。
- ALTER: ユーザーは、表への列の追加、表およびその列の注釈の追加または変更、基本キーまたは固有制約の追加、および表検査制約の作成または除去ができます。表のトリガーも作成できますが、その場合はトリガー中で参照されるすべてのオブジェクトに対する追加の権限が (トリガーが表の任意の列を参照する場合は表に対する SELECT も) 必要です。すべての子孫表に対する ALTER 特権を持つユーザーは、基本キーを除去することができます。表に対する ALTER 特権と親表に対する REFERENCES 特権を持つか、該当の列に対する REFERENCES 特権を持つユーザーは、外部キーを作成または除去することができます。 ALTER 特権を持つユーザーは、表に対する COMMENT ON も可能です。
- DELETE: ユーザーは、表や視点から行を削除できます。
- INDEX: ユーザーは、表について索引を作成できます。索引の作成者には、索引についての CONTROL 特権が自動的に与えられます。詳しくは、267ページの『索引の特権』を参照してください。
- INSERT: ユーザーは、表や視点に行を挿入したり、 IMPORT ユーティリティーを実行したりできます。
- REFERENCES: ユーザーは、表を関係の中の親として指定して、外部キーを作成および除去できます。ユーザーは、特定の列にのみこの特権を持つことができます。
- SELECT: ユーザーは、表や視点から行を取り出したり、表に基づく視点を作成したり、EXPORT ユーティリティーを実行したりできます。

- UPDATE: ユーザーは、表または視点の項目の変更、あるいは表または視点の 1 つ以上の特定の列の中の項目の変更ができます。ユーザーは、特定の列にのみこの特権を持つことができます。

GRANT ステートメントの WITH GRANT OPTION を使用して、これらの特権を他のユーザーに付与する特権を付与することもできます。

注: ユーザーまたはグループが、ある表の CONTROL 特権を付与された場合、その表に対する他のすべての特権は、自動的に WITH GRANT OPTION によって付与されます。その後、表に対する CONTROL 特権をユーザーから取り消しても、そのユーザーは自動的に付与された他の特権を依然として持っています。 CONTROL 特権と一緒に付与された特権をすべて取り消す場合は、特権を個別に明示的に取り消すか、または REVOKE ステートメントに ALL キーワードを指定しなければなりません。以下にその例を示します。

```
REVOKE ALL
   ON EMPLOYEE FROM USER HERON
```

タイプ付き表を処理しているときには、表および視点の特権に関連した含意があります。

注: 特権は、表階層の各レベルで別々に授与されます。その結果、タイプ付き表の階層内のスーパー表で特権を授与されたユーザーは、副表にも間接的に影響を与えることがあります。しかし、その副表で必要な特権が保持されている場合は、ユーザーは副表に対する操作を直接的にしか行えません。

表階層の表の間のスーパー表 / 副表関係は、SELECT、UPDATE、および DELETE などの操作が、操作のターゲット表とそのすべての副表 (もしあれば) の行に影響を与えることを意味します。この性質を『代替性』と呼ぶことができます。たとえば、タイプ Manager_t の副表マネージャーを使用して、タイプ Employee_t の Employee 表を作成したとします。構造型 Employee_t と Manager_t 間のタイプ / サブタイプ関係、また表 Employee と Manager 間の対応する表 / 副表関係によって示されているとおり、マネージャーはある種の (特殊な) 従業員です。この関係の結果、次の SQL 照会は、

```
SELECT * FROM Employee
```

従業員とマネージャー両方のオブジェクト識別子と Employee_t 属性を戻します。同様に、次の更新操作は、

```
UPDATE Employee SET Salary = Salary + 1000
```

マネージャーと従業員の給与を 1000 ドル引き上げます。

Employee で SELECT 特権を持つユーザーは、 Manager で明示的な SELECT 特権を持っていなくても、この SELECT 操作を実行できます。しかし、そのようなユーザーは、 Manager 副表に対して SELECT 操作を直接実行することは許可されませんので、Manager 表の継承されたのではない列にアクセスすることはできません。

同様に、Manager で UPDATE 特権を持つユーザーは、 Manager 表で明示的な UPDATE 特権がなくても、 Manager に対して UPDATE 操作を実行できるので、正規の従業員とマネージャー両方に影響を与えます。 しかし、そのようなユーザーは、Manager 副表に対して UPDATE 操作を直接実行することは許可されませんので、Manager 表の継承されたのではない列を更新することはできません。

特定のコマンド、API、または SQL ステートメントの実行に必要な権限許可については、次のマニュアルを参照してください。

- *SQL 解説書*
- *コマンド解説書*
- *管理 API 解説書.*

カタログ統計の更新に必要な許可については、*管理の手引き: パフォーマンス* を参照してください。

視点の特権が決定される方法については、 *SQL 解説書* の CREATE VIEW ステートメントの説明を参照してください。

ニックネームの特権

ニックネームの特権には、データベース内のニックネーム上でのアクションが含まれます。これらの特権は、ニックネームが参照するデータ・ソース・オブジェクト上の特権には影響しません。次に挙げる特権のいずれかを使用するユーザーには、データベースについての CONNECT 特権が必要です。

- CONTROL: ニックネームについてすべての特権を持つことになり、ニックネームの除去、ニックネームについての個別の特権の付与と取り消しなどができます。CONTROL を付与するには、SYSADM または DBADM 権限が必要です。ニックネームの作成者は、自動的にそのニックネームの CONTROL 特権を受け取ります。
- ALTER: ユーザーはニックネームの列名を変更したり、列のデータ・タイプがマップする DB2 タイプを追加または変更したり、ニックネーム列の列オプションを設定したりすることができます。
- INDEX: ユーザーは、ニックネームについて索引の指定を作成できます。ニックネームの作成者には、索引についての CONTROL 特権が自動的に与えられます。
- REFERENCES: ユーザーは、ニックネームを関係の中の親として指定して、外部キーを作成および除去できます。ユーザーは、特定の列にのみこの特権を持つことができます。

GRANT ステートメントの WITH GRANT OPTION を使用して、これらの特権を他のユーザーに付与する特権を付与することもできます。

注: ユーザーまたはグループが、あるニックネームの CONTROL 特権を付与された場合、そのニックネームに対する他のすべての特権は、自動的に WITH GRANT

OPTION によって付与されます。その後、ニックネームに対する CONTROL 特権をユーザーから取り消しても、そのユーザーは自動的に付与された他の特権を依然として持っています。

データ・ソースのデータにアクセスするには、ニックネームが参照するデータ・ソースにあるオブジェクトについての適切な許可もなければなりません。

ユーザーが 1 つ以上のニックネームを参照する視点にアクセスする場合、ユーザーにはデータ・ソースでニックネームが参照する視点とオブジェクトにアクセスする権限がなければなりません。

サーバー特権

PASSTHRU というサーバー特権があります。この特権は、 DDL および DML ステートメントを直接 (パススルー操作) データ・ソースに発行する、許可 ID を制御します。

DB2 は 2 つの SQL ステートメントを提供してパススルー操作を制御します。

- GRANT PASSTHRU: データ・ソースに対して SET PASSTHRU ステートメントを発行する権限を付与し、 DML と DDL ステートメントをそのデータ・ソースにパススルーします。
- REVOKE PASSTHRU: データ・ソースに対して SET PASSTHRU ステートメントを発行する権限を取り消し、 DML と DDL ステートメントをそのデータ・ソースにパススルーします。

パススルー権限をサーバー ORACLE1 のユーザー SHAWN に付与するサンプル・ステートメントを次に示します。

```
GRANT PASSTHRU ON SERVER ORACLE1 TO USER SHAWN
```

PASSTHRU ステートメントの構文に関する詳細は、*SQL 解説書* を参照してください。

パッケージの特権

パッケージとは、データベース・オブジェクトの 1 つで、データベース・マネージャーが、特定のアプリケーション・プログラムにとって最も効率的な方法でデータにアクセスするのに必要な情報が入れられたものです。パッケージの特権を与えられたユーザーは、パッケージの作成と操作を行えます。次に挙げる特権のいずれかを使用するユーザーには、データベースについての CONNECT 特権が必要です。

- CONTROL: ユーザーに、パッケージの再バインド、除去、または実行を行う能力、ならびに、これらの特権を他のユーザーに与える能力を提供します。パッケージの作成者には自動的にこの特権が与えられます。 CONTROL 特権を持つユーザーには、BIND 特権と EXECUTE 特権も付与されます。そのユーザーは、他のユーザーにも BIND 特権と EXECUTE 特権を付与できます。 CONTROL 特権を付与するためには、SYSADM または DBADM 権限が必要です。

- BIND: ユーザーは、既存のパッケージを再バインドできます。
- EXECUTE: ユーザーはパッケージを実行できます。

これらのパッケージ特権に加えて、BINDADD データベース特権によって、ユーザーは、データベース内に新しいパッケージを作成するか、または既存のパッケージを再バインドすることができます。

ニックネームを含むパッケージを実行する権限のあるユーザーには、パッケージ内で付加的な特権またはニックネームの権限レベルは必要ありません。ただし、ニックネームが参照するオブジェクトを含むデータ・ソースで、認証検査をパスする必要があります。さらに、パッケージ・ユーザーには、データ・ソースにあるデータ・ソース・オブジェクトについての適切な特権、または権限レベルが必要です。

DB2 ファミリーのデータ・ソースを使って通信するときに DB2 が動的 SQL を使用するため、ニックネームを含むパッケージが付加的な許可ステップを必要とする可能性があります。データ・ソースでパッケージを実行する許可 ID には、そのデータ・ソースでパッケージを動的に実行するための適切な権限が必要です。 DB2 が静的および動的 SQL を処理する方法についての詳細は、 *SQL 解説書* を参照してください。

索引の特権

索引または索引の指定の作成者には、索引についての CONTROL 特権が自動的に与えられます。索引の CONTROL 特権は、実際には、索引を除去するための能力です。ある索引の CONTROL 特権を付与するためには、そのユーザーには SYSADM または DBADM 権限が必要です。

表レベルの INDEX 特権を使うと、表に関する索引を作成できます (263ページの『表および視点の特権』を参照)。

シーケンス特権

シーケンスの作成者には、自動的に USAGE 特権が与えられます。 USAGE 特権は、シーケンスで NEXTVAL および PREVVAL 式を使用する場合に必要になります。他のユーザーに NEXTVAL および PREVVAL 式の使用を許可するには、シーケンス特権を PUBLIC に付与しなければなりません。これで、すべてのユーザーが指定されたシーケンスでこれらの式を使用できるようになります。

データベース・オブジェクトに対するアクセスの制御

データベース・アクセスを制御するには、直接および間接の特権、管理権限、およびパッケージについての理解が必要です。この部分ではそうした点について説明し、いくつかの例を挙げます。

直接付与される特権は、システム・カタログに格納されます。データベース・アクセス制御計画の実施を監査する方法については、 279ページの『システム・カタログの使用』を参照してください。

権限許可を制御する方法には、次の 3 通りのものがあります。

- 明示的な権限許可は、 GRANT および REVOKE ステートメントを使用して制御される特権によって制御されます。
- 暗黙の権限許可は、オブジェクトの作成と除去によって制御されます。
- 間接特権はパッケージに関連したものです。

この部分では、次の点について説明します。

- 『特権の付与』
- 269ページの『特権の取り消し』
- 271ページの『オブジェクトの作成と除去による暗黙許可の管理』
- 272ページの『パッケージによる間接特権の許可』
- 273ページの『視点によるデータへのアクセスの制御』
- 276ページの『監査機能を使用したデータ・アクセスのモニター』

特権の付与

GRANT ステートメントは、許可されたユーザーが特権を付与することができるようにするものです。特権は、1 つのステートメントで、1 つ以上の許可名に付与するか、あるいは、特権をすべてのユーザーが使用可能なようにする PUBLIC に付与することができます。許可名は、個別のユーザーかまたはグループのいずれかにすることができることに注意してください。

オペレーティング・システムに同じ名前のユーザーとグループがある場合、ユーザーとグループのどちらに特権を付与するのかを指定する必要があります。 GRANT および REVOKE ステートメントのどちらにおいても、 USER および GROUP というキーワードがサポートされています。これらの任意選択のキーワードが使用されないと、データベース・マネージャーはオペレーティング・システムの機密保護機能をチェックして、その許可名がユーザーであるかグループであるかを判別します。許可名がユーザーとグループの両方である可能性がある場合、エラーが戻されます。

次の例は、HERON というユーザーに対して、 EMPLOYEE 表についての SELECT 特権を付与するものです。

```
GRANT SELECT
   ON EMPLOYEE TO USER HERON
```

次の例は、HERON というグループに対して、 EMPLOYEE 表についての SELECT 特権を付与するものです。

```
GRANT SELECT
   ON EMPLOYEE TO GROUP HERON
```

ほとんどのデータベース・オブジェクトに対する特権を付与するには、ユーザーがそのオブジェクトに対する SYSADM 権限、DBADM 権限、または CONTROL 特権を持つか、またはそのユーザーが WITH GRANT OPTION 特権を保持していなければなりません。特権を付与できるのは既存のオブジェクトについてだけです。だれかに CONTROL 特権を付与するためには、SYSADM または DBADM 権限が必要です。DBADM 権限を付与するには、SYSADM 権限が必要です。

GRANT ステートメントの詳細については、*SQL 解説書* を参照してください。

特権の取り消し

REVOKE ステートメントは、許可されたユーザーが、他のユーザーに付与されている特権を取り消すことができるようにするためのものです。データベース・オブジェクトについての特権を取り消すには、DBADM 権限、 SYSADM 権限、またはそのオブジェクトについての CONTROL 特権が必要です。 WITH GRANT OPTION 特権を保持しているだけでは、その特権を取り消すには十分でないことに注意してください。他のユーザーの CONTROL 特権を取り消すには、SYSADM または DBADM 権限が必要です。DBADM 権限を取り消すには、SYSADM 権限が必要です。特権を取り消すことができるのは、既存のオブジェクトについてだけです。

注: 表または視点に対する DBADM 権限または CONTROL 特権を持っていないユーザーは、 WITH GRANT OPTION を使用して付与された特権を取り消すことはできません。また、取り消された人から付与された特権を受け取っているユーザーに対する取り消しには、連鎖はありません。特権の取り消しに必要な権限の詳細については、 *SQL 解説書* を参照してください。

同じ名前のユーザーとグループの両方に特権が付与されている場合、特権を取り消す時に、GROUP と USER キーワードのどちらかを指定する必要があります。次の例は、HERON というユーザーの EMPLOYEE 表についての SELECT 特権を取り消すものです。

```
REVOKE SELECT
   ON EMPLOYEE FROM USER HERON
```

次の例は、HERON というグループの EMPLOYEE 表についての SELECT 特権を取り消すものです。

```
REVOKE SELECT
   ON EMPLOYEE FROM GROUP HERON
```

1 つのグループから特権を取り消しても、そのグループに属するすべてのメンバーからその特権が取り消されるとは限らないことに注意してください。個別の名前が特権を直接付与されている場合は、その特権が直接取り消されるまで保持されます。

表特権がユーザーから取り消される場合、取り消された表特権に依存するそのユーザーによって作成されたすべての視点に対する特権も取り消されます。ただし、システムによって暗黙に付与された特権のみが取り消されます。視点についての特権が別のユーザーによって直接付与された場合、その特権は引き続き保持されます。

特権をグループに付与してから、そのグループの 1 人のメンバーだけから特権を取り消すという状況があります。エラー・メッセージ SQL0556N を受け取らないでこれを行うには、次の 2 つの方法だけを行ってください。

- グループからそのメンバーを除去します。あるいは、メンバーを減らして新規グループを作成し、その新規グループに特権を付与します。
- グループから特権を取り消してから、個々のユーザー (許可 ID) に特権を付与します。

明示的に付与された表 (または視点) についての特権が DBADM 権限によってユーザーから取り消される場合、その表に定義されている他の視点についての特権は取り消されることは**ありません**。視点についての特権は DBADM 権限によって使用可能になるものであり、基礎表の明示特権とは関係ないからです。

1 つ以上の基礎となる表または視点に基づいて視点を定義しており、それらの基礎となる表または視点の 1 つ以上に対する SELECT 特権を失った場合、その視点は使用できません。

注: 表または視点に対する CONTROL 特権がユーザーから取り消された場合でも、そのユーザーは、その特権を他のユーザーに付与する能力は持ち続けます。
CONTROL 特権が与えられると、そのユーザーは他の WITH GRANT OPTION 特権もすべて受け取ります。 CONTROL が取り消されても、他の特権のすべては、それらが明示して取り消されるまで、 WITH GRANT OPTION のまま残されます。

取り消された特権に依存しているすべてのパッケージは無効とみなされますが、十分な権限を持つユーザーによって再バインドされるなら再び有効になります。特権が後で再びアプリケーションをバインドしたユーザーに付与される場合、パッケージも再作成することができます。そのアプリケーションを実行すると、暗黙の再バインドが正常に実行されるトリガーとなります。特権が PUBLIC から取り消された場合、 PUBLIC 特権に基づいたバインドしかできないユーザーによってバインドされていたすべてのパッケージが無効にされます。ユーザーの持つ DBADM 権限が取り消されると、そのユーザーによってバインドされたパッケージはすべて無効になります。データベース・ユーティリティーに関連するパッケージも例外ではありません。無効のマークが付けられているパッケージを使用しようとすると、システムは、そのパッケージの再バインドを試みます。この再バインドの試みが失敗すると、エラー (SQLCODE -727) が発生します。この場合、それらのパッケージを明示的に再バインドするには、以下の権限が必要です。

- それらのパッケージを再バインドするための権限
- それらのパッケージ内で使われているオブジェクトについての該当する権限

そうしたパッケージの再バインドは、特権を取り消す時に行うべきです。REVOKE および REBIND PACKAGE ステートメントの詳細については、 *SQL 解説書* を参照してください。

1 つ以上の特権に基づいてトリガーを定義しており、それらの特権のうちの 1 つ以上を失った場合、そのトリガーは使用できません。

オブジェクトの作成と除去による暗黙許可の管理

データベース・マネージャーは、CREATE SCHEMA、CREATE TABLESPACE、CREATE TABLE、CREATE VIEW、または CREATE INDEX ステートメントを出すユーザー、あるいは PREP または BIND コマンドを使用して新しいパッケージを作成するユーザーに対して、暗黙に特定の特権を付与します。特権は、SYSADM または DBADM 権限を持つユーザーによってオブジェクトが作成されるときにも付与されます。同じように、オブジェクトを除去すると特権は除去されます。

作成されるオブジェクトが表スペース、表、索引、パッケージの場合、ユーザーにはそのオブジェクトについての CONTROL 特権が与えられます。オブジェクトが視点の場合、その視点についての CONTROL 特権が暗黙のうちに付与されるのは、その視点定義の中で参照されるすべての表と視点についての CONTROL 特権を持っている場合に限られます。

明示して作成されたオブジェクトがスキーマである場合、そのスキーマの所有者には、WITH GRANT OPTION によって ALTERIN、CREATEIN、および DROPIN 特権が与えられます。暗黙に作成されたスキーマは、PUBLIC に付与された CREATEIN 特権を持ちます。

視点の特権が決定される方法については、 *SQL 解説書* の CREATE VIEW ステートメントの説明を参照してください。

プランまたはパッケージの所有権の確立

BIND および PRECOMPILE コマンドは、アプリケーション・パッケージを作成または変更します。どちらのコマンドでも、 OWNER オプションを使って結果パッケージの所有者の名前を付けてください。パッケージの所有者の命名には、単純ルールがあります。

- どのユーザーでも、自分を所有者として命名できます。 OWNER オプションが指定されていない場合、これがデフォルトです。
- SYSADM または DBADM 権限を持つ ID は、OWNER オプションを使って、どの許可 ID でも所有者として命名できます。

DB2 データベース製品を使用するパッケージをバインドできる、すべてのオペレーティング・システムが OWNER オプションをサポートしているわけではありません。

BIND および PRECOMPILE コマンドの詳細については、*コマンド解説書* を参照してください。

パッケージによる間接特権の許可

データベース内のデータに対するアクセス要求は、アプリケーション・プログラムによって、また、対話式ワークステーション・セッションに関係しているユーザーによって行えます。パッケージに含まれているステートメントによって、ユーザーは多数のデータベース・オブジェクトに対して様々なアクションを実行できます。そうしたアクションにはそれぞれ特権が必要です。

パッケージをバインドしている個別ユーザーに付与された特権、および PUBLIC に付与された特権は、静的 SQL がバインドされるときの許可検査のために使用されます。グループを通して付与された特権は、静的 SQL がバインドされるときの許可検査のためには使用**されません**。有効な *authID* を持っており、パッケージをバインドするユーザーは、パッケージのバインド時に VALIDATE RUN が指定されている場合を除いて、そのパッケージ内の静的 SQL ステートメントを実行するために必要なすべての特権を明示的に付与されているか、あるいは PUBLIC を通して必要な特権が暗黙に付与されているかのいずれかでなければなりません。 BIND 時に VALIDATE RUN を指定すると、そのパッケージ内のどの静的 SQL ステートメントに許可の障害が発生したとしても、 BIND は失敗せず、その SQL ステートメントは実行時に再び有効になります。ユーザーがパッケージをバインドするための、適切な許可 (BIND または BINDADD 特権) を持っているようにするための検査を行う場合には、 PUBLIC、グループ、およびユーザーの特権が**すべて**使用されます。

パッケージには、静的 SQL および動的 SQL の両方を含めることができます。静的 SQL を含むパッケージを処理する場合、ユーザーに必要なのはパッケージについての EXECUTE 特権だけです。したがってそのユーザーは、パッケージ内の静的 SQL に対するパッケージ・バインダーの特権を間接的に取得することができます。ただし、これはパッケージによって課される制限の範囲内に限られます。

動的 SQL を含むパッケージを処理する場合、ユーザーは、そのパッケージに対する EXECUTE 特権を持っていなければなりません。ユーザーは、そのパッケージの EXECUTE 特権に加えて、そのパッケージ内の動的 SQL ステートメントを実行するためのすべての特権を必要とします。パッケージ内のすべての静的 SQL に対しては、バインダーの権限と特権が使用されます。

ニックネームを含むパッケージによる間接特権の許可

パッケージにニックネームへの参照が含まれる場合、パッケージ作成者およびパッケージ・ユーザーの許可処理はもう少し複雑です。パッケージ作成者が、ニックネームを含むパッケージを正常にバインドする場合、パッケージ作成者は、ニックネームがデータ・ソースで参照する表および視点に関して、許可検査または特権検査をパスする必要はありません。しかし、パッケージの実行者は、データ・ソースで認証および許可検査をパスすることが必要です。

たとえば、パッケージ作成者の .SQC ファイルに、複数の SQL ステートメントが入っているとします。 1 つの静的ステートメントはローカル表を参照します。別の動的ステ

ートメントはニックネームを参照します。パッケージがバインドされると、ローカル表の特権を検証するためにパッケージ作成者の許可 ID が使用されます。しかし、ニックネームが識別するデータ・ソース・オブジェクトに関しては何も検査されません。 別のユーザーが、そのパッケージ用の EXECUTE 特権があることを前提としてパッケージを実行する場合、そのユーザーは表を参照するステートメントへの付加的な特権検査をパスする必要はありません。しかし、ニックネームを参照するステートメントの場合は、パッケージを実行するユーザーはデータ・ソースで認証検査と特権検査をパスすることが必要です。

.SQC ファイルに動的 SQL ステートメントと、表とニックネームの参照が混ざったもののみが入っている場合、ローカル・オブジェクトとニックネームへの DB2 許可検査は同じです。 パッケージ・ユーザーは、ステートメント内のすべてのローカル・オブジェクト (表、視点) に関する特権検査をパスしなければならず、さらにニックネーム・オブジェクトの特権検査もパスする必要があります (パッケージ・ユーザーは、ニックネームが識別するオブジェクトを含むデータ・ソースで認証および特権検査をパスしなければなりません)。どちらの場合も、パッケージのユーザーには EXECUTE 特権が必要です。

パッケージの ID およびパスワードは、すべてのデータ・ソース認証、および特権処理に使用されます。この情報は、ユーザー・マッピングの作成によって変更できます。

注: ニックネームは、静的 SQL では指定できません。 DYNAMICRULES オプション (BIND に設定) を、ニックネームを含むパッケージで使用しないでください。

DB2 ファミリーのデータ・ソースを使って通信するときに DB2 が動的 SQL を使用するため、ニックネームを含むパッケージが付加的な許可ステップを必要とする可能性があります。データ・ソースでパッケージを実行する許可 ID には、そのデータ・ソースでパッケージを動的に実行するための適切な権限が必要です。 DB2 が静的および動的 SQL を処理する方法についての詳細は、 *SQL 解説書* を参照してください。

視点によるデータへのアクセスの制御

視点を使うと、次のようにして、表に対するアクセス制御や特権付与が行えます。

- 表内の指定した列だけにアクセスを制限する

  表内の特定の列だけに対するアクセスが必要なユーザーやアプリケーション・プログラムのためには、許可されたユーザーは、必要な列だけを指定した視点を作成できます。
- 表内の行のサブセットだけにアクセスを制限する

  許可されたユーザーは、視点定義の副照会の中に WHERE 文節を指定することにより、視点によってアクセスする行を限定できます。
- データ・ソース表または視点内の行のサブセットだけにアクセスを制限します。ニックネームによってデータ・ソースにアクセスしている場合、ニックネームを参照する

ローカル DB2 視点を作成することができます。これらの視点は、1 つ以上のデータ・ソースからニックネームを参照することができます。

注: 複数のデータ・ソースを参照するニックネームを含む視点を作成できるので、ユーザーは 1 つの視点から複数のデータ・ソースのデータにアクセスできます。これらの視点は、*マルチ・ロケーション視点* と呼ばれます。このような視点は、分散環境全体で重要な表の列の情報を結合する場合や、または個々のユーザーに、特定のオブジェクトに対してデータ・ソースで必要な特権がない場合に役立ちます。

視点を作成するには、SYSADM 権限、DBADM 権限、または、その視点定義の中で参照する各表または各視点についての CONTROL 特権あるいは SELECT 特権が必要です。ユーザーは、その視点用に指定されたスキーマ内に、オブジェクトを作成することもできなければなりません。つまり、スキーマがまだ存在していなければ、既存のスキーマに対する CREATEIN 特権、または、データベースに対する IMPLICIT_SCHEMA 権限が必要です。詳細は、 151ページの『視点の作成』を参照してください。

ニックネームを参照する視点を作成している場合、視点でニックネームが参照するデータ・ソース・オブジェクト (表と視点) で付加的な権限が必要になることはありません。しかし、ユーザーには視点へのアクセス時に基礎となるデータ・ソース・オブジェクト用に、 SELECT 権限または同等の権限レベルが必要です。

ユーザーに、基礎となるオブジェクト (表および視点) に対してデータ・ソースでの適切な権限がない場合、次のことを実行できます。

1. ユーザーのアクセスが許可されるデータ・ソース表で、これらの列に対してデータ・ソース視点を作成する
2. この視点の SELECT 特権をユーザーに付与する
3. 視点を参照するためのニックネームを作成する

その後、新しいニックネームを参照する SELECT ステートメントを発行することによって、列にアクセスすることができます。

以下のシナリオは、情報へのアクセスを制限するために、視点を使用する方法をより詳細に示した例です。

次のようなさまざまな理由から、STAFF 表の情報にアクセスする必要のある人が多いとします。たとえば、次のようなものがあります。

- 人事部では、表全体についての更新と参照ができなければなりません。

  この要件を満たすのに必要なことは、次のようにして、 PERSONNL グループに STAFF 表に対する SELECT 特権と UPDATE 特権を付与するだけです。

  ```
  GRANT SELECT,UPDATE ON TABLE STAFF TO GROUP PERSONNL
  ```

- 各部署のマネージャーは、部下の給与についての情報を参照する必要があります。

これは、各部署のマネージャーごとに、専用の視点を作成することによって解決できます。たとえば、51 番の部署のマネージャーに対しては、次のように視点を作成できます。

```
CREATE VIEW EMP051 AS
   SELECT NAME,SALARY,JOB FROM STAFF
   WHERE DEPT=51
GRANT SELECT ON TABLE EMP051 TO JANE
```

JANE という許可名を持つマネージャーは、STAFF 表と同じように EMP051 視点を照会します。このマネージャーが STAFF 表の EMP051 という視点にアクセスするとき、表示される情報は次のようになります。

NAME	SALARY	JOB
Fraye	45150.0	Mgr
Williams	37156.5	Sales
Smith	35654.5	Sales
Lundquist	26369.8	Clerk
Wheeler	22460.0	Clerk

- すべてのユーザーは他の従業員の場所を知る必要があります。この要件を満たすには、次のようにして、 STAFF 表の NAME 列と ORG 表の LOCATION 列についての視点を作成し、それぞれ DEPT 列と DEPTNUMB 列に基づいて 2 つの表を結合します。

  ```
  CREATE VIEW EMPLOCS AS
     SELECT NAME, LOCATION FROM STAFF, ORG
     WHERE STAFF.DEPT=ORG.DEPTNUMB
  GRANT SELECT ON TABLE EMPLOCS TO PUBLIC
  ```

  従業員の場所に関する視点にアクセスするユーザーは、次の情報を見ることになります。

NAME	LOCATION
Molinare	New York
Lu	New York
Daniels	New York
Jones	New York
Hanes	Boston
Rothman	Boston
Ngan	Boston
Kermisch	Boston
Sanders	Washington

NAME	LOCATION
Pernal	Washington
James	Washington
Sneider	Washington
Marenghi	Atlanta
O'Brien	Atlanta
Quigley	Atlanta
Naughton	Atlanta
Abrahams	Atlanta
Koonitz	Chicago
Plotz	Chicago
Yamaguchi	Chicago
Scoutten	Chicago
Fraye	Dallas
Williams	Dallas
Smith	Dallas
Lundquist	Dallas
Wheeler	Dallas
Lea	San Francisco
Wilson	San Francisco
Graham	San Francisco
Gonzales	San Francisco
Burke	San Francisco
Quill	Denver
Davis	Denver
Edwards	Denver
Gafney	Denver

監査機能を使用したデータ・アクセスのモニター

DB2 監査機能は、一連の事前定義データベースのイベントに対して監査証跡を生成し、かつその監査証跡を維持できるようにします。データへのアクセスを妨げる機能ではありませんが、監査機能はデータ・オブジェクトをアクセスまたは変更しようとした試みの記録をモニターかつ保持することができます。

SYSADM 権限は、監査機能の管理者ツール `db2audit` を使用するのに必須です。

DB2 監査機能の詳細記述については、285ページの『第6章 DB2 アクティビティーの監査』を参照してください。

データ暗号化

セキュリティ計画の一環として、データの暗号化があります。これを行うため、暗号化および暗号化解除組み込み関数である ENCRYPT、DECRYPT_BIN、DECRYPT_CHAR、および GETHINT を使用することができます。 これらの関数の詳細 (構文など) については、*SQL 解説書* を参照してください。

ENCRYPT 関数は、パスワード・ベースの暗号化メソッドでデータを暗号化します。これらの関数によって、パスワード・ヒントをカプセル化することもできます。パスワード・ヒントは、暗号化データに組み込まれています。暗号化したデータの暗号化を解除するには、正しいパスワードを使用する必要があります。これらの関数を使用する開発者は、パスワードを忘れた場合の管理や使用できないデータの管理について考慮しなければなりません。

ENCRYPT 関数の結果は、最初の引き数と同じデータ・タイプになります。

VARCHAR のみ暗号化することができます。

宣言された結果の長さは、以下のいずれかです。

- 任意選択のヒント・パラメーターが指定されている場合、データ引き数の長さプラス 42
- 任意選択のヒント・パラメーターが指定されていない場合、データ引き数の長さプラス 10

DECRYPT_BIN および DECRYPT_CHAR 関数は、パスワード・ベースの暗号化解除を使用して、データの暗号化を解除します。

DECRYPT_BIN および DECRYPT_CHAR 関数の結果は、最初の引き数と同じデータ・タイプになります。

宣言された結果の長さは、オリジナル・データの長さです。

GETHINT 関数は、カプセル化されたパスワード・ヒントを返します。パスワード・ヒントとは、データ所有者がパスワードを思い浮かべるために役立つフレーズです。たとえば、パスワード ″Pacific″ を思い浮かべるために、ワード「Ocean」をヒントとして使用することができます。

データの暗号化に使用するパスワードは、以下のいずれかの方法で決定されます。

- パスワード引き数。パスワードは、ENCRYPT 関数が呼び出されるときに明示的に渡されるストリングです。データは、与えられたパスワードで暗号化および暗号化解除されます。

- 特殊レジスター・パスワード。 SET ENCRYPTION PASSWORD ステートメントはパスワード値を暗号化し、その暗号化されたパスワードをデータベース・マネージャーに送信して、特殊レジスターに保管します。パスワード・パラメーターなしで呼び出された ENCRYPT、DECRYPT_BIN、および DECRYPT_CHAR 関数は、ENCRYPTION PASSWORD 特殊レジスターの値を使用します。

  特殊レジスターの初期値 (デフォルト値) は空ストリングです。

パスワードに有効な長さは 6 ～ 127 文字です。ヒントに有効な長さは 0 ～ 32 文字です。

ENCRYPTION PASSWORD 特殊レジスターがクライアントによって設定されている場合、パスワードはクライアント側で暗号化された後、データベース・サーバーに送信されて暗号化解除されます。パスワードを読めない状態に保つため、データベース・サーバーでも再暗号化されます。 DECRYPT_BIN および DECRYPT_CHAR 関数は、使用する前に特殊レジスターを暗号化解除する必要があります。 ENCRYPTION PASSWORD に見つかった値も読めない状態になっています。ゲートウェイ・セキュリティはサポートされていません。

タスクとそれに必要な権限許可

仕事の責任の分担方法は、それぞれの組織によって違います。 表5 に、他の一般的な仕事の種類、それらの仕事に通常伴うタスク、およびそれらのタスクを行うために必要な権限または特権を挙げます。

表 5. 一般的な仕事の種類、タスク、および必要な権限許可

仕事の種類	タスク	必要な許可
部署管理者	部署のシステムを監督する。データベースを作成する。	SYSCTRL 権限。部署に独自のインスタンスのある場合は SYSADM 権限。
機密保護管理者	権限許可と特権の一部または全部を他のユーザーに許可する。	SYSADM または DBADM 権限。
データベース管理者	データベースの設計、開発、操作、保全、保守を行う。	データベースについての DBADM および SYSMAINT 権限。場合によっては、SYSCTRL 権限。
システム操作員	データベースをモニターし、バックアップ機能を実行する。	SYSMAINT 権限。

表 5. 一般的な仕事の種類、タスク、および必要な権限許可 (続き)

仕事の種類	タスク	必要な許可
アプリケーション・プログラマー	データベース・マネージャーのアプリケーション・プログラムの開発およびテストを行う。テスト・データの表を作成することもある。	既存のパッケージに対する BINDADD、BIND 特権、 1 つ以上のデータベースの CONNECT および CREATETAB 特権、一部の特定のスキーマの特権、および一部の表についての一連の特権。
ユーザー・アナリスト	システム・カタログ視点を調べて、アプリケーション・プログラムのデータ要件を定義する。	カタログ視点の SELECT 特権。 1 つ以上のデータベースの CONNECT 特権。
プログラム・エンド・ユーザー	アプリケーション・プログラムを実行する。	パッケージの EXECUTE 特権。 1 つ以上のデータベースの CONNECT 特権。この表の下にある注記を参照してください。
インフォメーション・センター・コンサルタント	照会ユーザーについてのデータ要件を定義する。表と視点を作成し、データベース・オブジェクトへのアクセス権を付与することによって、データを提供する。	データベースについての DBADM 権限。
照会ユーザー	データの検索、追加、削除、または変更のために SQL ステートメントを出す。場合によっては、結果を表に保管する。	1 つ以上のデータベースの CONNECT 特権。作成する表および視点のスキーマの CREATEIN 特権。一部の表および視点の SELECT、 INSERT、 UPDATE、 DELETE 特権。

注: アプリケーション・プログラムに動的 SQL ステートメントが含まれている場合、プログラムのエンド・ユーザーには、 EXECUTE と CONNECT に加えてさらにほかの特権 (SELECT、INSERT、DELETE、および UPDATE など) が必要になる場合があります。

システム・カタログの使用

それぞれのデータベースについての情報は、システム・カタログと呼ばれる 1 組の視点 (データベースが生成されるときに作成されます) の中で自動的に維持されます。このシステム・カタログには、表、列、索引、プログラム、特権、その他のオブジェクトが含まれています。

これらの視点のうち、以下の 6 つには、ユーザーによって保持される特権と、それぞれの特権を付与するユーザーのアイデンティティーがリストされています。

SYSCAT.DBAUTH	データベースの特権のリスト
SYSCAT.TABAUTH	表と視点の特権のリスト
SYSCAT.COLAUTH	列の特権のリスト
SYSCAT.PACKAGEAUTH	パッケージの特権のリスト
SYSCAT.INDEXAUTH	索引の特権のリスト
SYSCAT.SCHEMAAUTH	スキーマの特権のリスト
SYSCAT.PASSTHRUAUTH	サーバーの特権のリスト

システムによってユーザーに付与される特権の付与者は、SYSIBM になります。SYSADM、SYSMAINT および SYSCTRL はシステム・カタログにリストされます。

CREATE ステートメントと GRANT ステートメントを使うと、システム・カタログの中に特権が入れられます。 SYSADM および DBADM 権限を持つユーザーは、システム・カタログ視点に対する SELECT 特権の付与と取り消しを行うことができます。次の例は、これらの SQL 照会を使って、特権に関する情報を抽出する方法を示すものです。

- 『付与された特権を持つ許可名の検索』
- 281ページの『DBADM 権限を持つすべての名前の検索』
- 281ページの『表へのアクセスを許可されている名前の検索』
- 282ページの『ユーザーに付与されたすべての特権の検索』
- 282ページの『システム・カタログ視点の機密保護』

付与された特権を持つ許可名の検索

どれか 1 つのシステム・カタログ視点に、すべての特権についての情報があるということはありません。以下のステートメントは、特権を持つすべての許可名を検索します。

```
SELECT DISTINCT GRANTEE, GRANTEETYPE, 'DATABASE' FROM SYSCAT.DBAUTH
UNION
SELECT DISTINCT GRANTEE, GRANTEETYPE, 'TABLE   ' FROM SYSCAT.TABAUTH
UNION
SELECT DISTINCT GRANTEE, GRANTEETYPE, 'PACKAGE ' FROM SYSCAT.PACKAGEAUTH
UNION
SELECT DISTINCT GRANTEE, GRANTEETYPE, 'INDEX   ' FROM SYSCAT.INDEXAUTH
UNION
SELECT DISTINCT GRANTEE, GRANTEETYPE, 'COLUMN  ' FROM SYSCAT.COLAUTH
UNION
SELECT DISTINCT GRANTEE, GRANTEETYPE, 'SCHEMA  ' FROM SYSCAT.SCHEMAAUTH
UNION
SELECT DISTINCT GRANTEE, GRANTEETYPE, 'SERVER  ' FROM SYSCAT.PASSTHRUAUTH
ORDER BY GRANTEE, GRANTEETYPE, 3
```

時折、このステートメントによって検索されたリストと、システムの機密保護機能で定義されているユーザー名とグループ名のリストとを比較してみるべきです。これによって、有効でなくなった許可名を識別することができます。

注: リモート・データベース・クライアントをサポートしている場合、許可名をリモート・クライアントだけに定義し、データベースのサーバー・マシンには定義しないということも可能です。

DBADM 権限を持つすべての名前の検索

以下のステートメントは、DBADM 権限が直接付与されている、すべての許可名を検索します。

```
SELECT DISTINCT GRANTEE FROM SYSCAT.DBAUTH
   WHERE DBADMAUTH = 'Y'
```

表へのアクセスを許可されている名前の検索

以下のステートメントは、修飾子 JAMES を持つ表 EMPLOYEE にアクセスすることが直接許可されている、すべての許可名を検索します。

```
SELECT DISTINCT GRANTEETYPE, GRANTEE FROM SYSCAT.TABAUTH
   WHERE TABNAME = 'EMPLOYEE'
     AND TABSCHEMA = 'JAMES'
UNION
SELECT DISTINCT GRANTEETYPE, GRANTEE FROM SYSCAT.COLAUTH
   WHERE TABNAME = 'EMPLOYEE'
     AND TABSCHEMA = 'JAMES'
```

だれが修飾子 JAMES を持つ表 EMPLOYEE を更新できるかを調べるためには、以下のステートメントを出します。

```
SELECT DISTINCT GRANTEETYPE, GRANTEE FROM SYSCAT.TABAUTH
   WHERE TABNAME = 'EMPLOYEE' AND TABSCHEMA = 'JAMES' AND
      (CONTROLAUTH  = 'Y' OR
       UPDATEAUTH   = 'Y' OR  UPDATEAUTH   = 'G')
UNION
SELECT DISTINCT GRANTEETYPE, GRANTEE FROM SYSCAT.DBAUTH
   WHERE DBADMAUTH = 'Y'
UNION
SELECT DISTINCT GRANTEETYPE, GRANTEE FROM SYSCAT.COLAUTH
   WHERE TABNAME = 'EMPLOYEE' AND TABSCHEMA = 'JAMES' AND
   PRIVTYPE = 'U'
```

これは、DBADM 権限をもつ許可名があればすべて検索し、さらに CONTROL または UPDATE 特権が直接付与されている許可名も検索します。ただし、SYSADM 権限だけを保持しているユーザーの許可名は戻しません。

一部の許可名は、個別のユーザーだけでなく、グループのものである場合もあることに注意してください。

ユーザーに付与されたすべての特権の検索

ユーザーは、システム・カタログ視点についての照会を行うことにより、自ら持っている特権のリストと、他のユーザーに付与した特権のリストを作成できます。たとえば、以下のステートメントは、個々の許可名に直接付与されているデータベース特権のリストを検索します。

```
SELECT * FROM SYSCAT.DBAUTH
   WHERE GRANTEE = USER AND GRANTEETYPE = 'U'
```

表の特権のうち特定のユーザーによって直接付与されたものを検索するには、次のようなステートメントを使います。

```
SELECT * FROM SYSCAT.TABAUTH
   WHERE GRANTOR  = USER
```

以下のステートメントは、特定のユーザーによって直接付与された、個別の列特権のリストを検索します。

```
SELECT * FROM SYSCAT.COLAUTH
   WHERE GRANTOR  = USER
```

このステートメントの中の USER というキーワードは、常にユーザーの許可名の値と等しくなります。 USER は読み取り専用の特殊レジスターです。特殊レジスターについて詳しくは、*SQL 解説書* を参照してください。

システム・カタログ視点の機密保護

データベースの生成時に、システム・カタログ視点に対する SELECT 特権が PUBLIC に付与されます。 (PUBLIC に自動的に付与される他の特権については、 259ページの『データベースの特権』を参照してください。) 多くの場合、このことによって機密保護上の問題が生じることはありません。しかし、これらの表にはデータベース内のすべてのオブジェクトが含まれているため、非常に重要なデータの場合は適切でないことがあります。このような場合には、PUBLIC から SELECT 特権を取り消してから、必要に応じて、特定のユーザーに対して SELECT 特権を付与することを考慮してください。システム・カタログ視点についての SELECT 特権の付与と取り消しは、他の視点の場合と同じ方法で行いますが、そのためには SYSADM または DBADM 権限が必要です。

少なくとも、次のカタログ視点に対するアクセスを制限することを考慮すべきです。

- SYSCAT.DBAUTH
- SYSCAT.TABAUTH
- SYSCAT.PACKAGEAUTH
- SYSCAT.INDEXAUTH
- SYSCAT.COLAUTH
- SYSCAT.PASSTHRUAUTH

- SYSCAT.SCHEMAAUTH

それによって、ユーザー特権についての情報がデータベースにアクセスできる人全員で利用できるような事態を避けることができます。この情報を使用して、不適切なユーザーがデータベースに対する無許可アクセスを取得する可能性があります。

各列にどの統計が集められているかも調べてください (*管理の手引き: パフォーマンス*の『カタログ統計』を参照)。システム・カタログに記録されている統計には、ご使用の環境では機密情報となりうるデータ値が含まれていることもあります。この統計に機密データが含まれている場合には、 SYSCAT.COLUMNS および SYSCAT.COLDIST カタログ視点についての SELECT 特権を、 PUBLIC から取り消すことができます。

システム・カタログ視点に対するアクセスを限定したい場合は、それぞれの許可名が自分自身の特権にかかわる情報だけを検索できるようにする視点を定義することができます。

たとえば、次の視点 MYSELECTS には、ユーザーの許可名に SELECT 特権が直接付与されている表の所有者と名前が含まれます。

```
CREATE VIEW MYSELECTS AS
   SELECT TABSCHEMA, TABNAME FROM SYSCAT.TABAUTH
   WHERE GRANTEETYPE = 'U'
     AND GRANTEE = USER
     AND SELECTAUTH = 'Y'
```

このステートメントの中の USER というキーワードは、常に、許可名の値と等しくなります。

以下のステートメントは、この視点をそれぞれの許可名から利用可能にするものです。

```
GRANT SELECT ON TABLE MYSELECTS TO PUBLIC
```

最後に、基礎表についての SELECT 特権を必ず取り消すようにしてください。

```
REVOKE SELECT ON TABLE SYSCAT.TABAUTH FROM PUBLIC
```

第6章 DB2 アクティビティーの監査

認証、権限、および、特権はデータへの既知のまたは予期されたアクセスを制御するために使用できますが、これらの方法はデータへの未知のまたは予期されないアクセスを防ぐには十分ではない可能性があります。後者のタイプのデータ・アクセスの検出を支援するために、 DB2 には監査機能があります。不必要なデータ・アクセスの正常モニターと以降の分析は、データ・アクセスの制御において、およびデータへの悪意のあるまたは不注意な無許可アクセスを最終的に防止し、改良につなげることができます。システム管理処置を含む、アプリケーションおよび個々のユーザー・アクセスのモニターは、データベース・システムのアクティビティーの履歴レコードを与えます。

DB2 監査機能は、一連の事前定義データベースのイベントに対して監査証跡を生成し、かつその監査証跡を維持できるようにします。この機能により生成されたレコードは、監査ログ・ファイルに保持されます。これらのレコードの分析により、システム誤用を識別することになる使用パターンを明らかにすることができます。一度識別されると、そのようなシステム誤用の削減または除去の処置がとられます。

監査機能はインスタンス・レベルで作用し、すべてのインスタンス・レベルのアクティビティーおよびデータベース・レベルのアクティビティーを記録します。

区分データベース環境において作動しているとき、多数の監査可能なイベントはユーザーがつながれる (調整プログラム・ノード) 区分で、またはカタログ・ノード (それらが同じパーティションではない場合) で発生します。この含意は、監査レコードが複数の区分で生成できることです。各監査レコードの一部は、調整プログラム・ノードおよび起点ノード識別子に関する情報を含んでいます。

監査ログ (db2audit.log) および監査構成ファイル (db2audit.cfg) は、インスタンスの `security` サブディレクトリーに位置しています。インスタンスを作成するときには、読み取り / 書き込み許可がオペレーティング・システムによりファイルの可能な場所にセットされます。デフォルトでは、この許可はインスタンス所有者専用の読み取り / 書き込みとなります。これらの許可を変更しないことが推奨されています。

監査機能の管理者ツール (db2audit) のユーザーは、 SYSADM の権限 / 特権を持っていなければなりません。

監査機能は、明示的に停止および開始される必要があります。開始するとき、監査機能は既存の監査構成情報を使用します。監査機能が DB2 サーバーから独立しているので、インスタンスが停止されるとしても、それは活動的な状態を維持します。実際、インスタンスが停止されるとき、監査レコードは監査ログにおいて生成される可能性があります。

監査機能の許可ユーザーは、監査機能の中で次の動作を制御できます。

- DB2 インスタンスの中で監査可能なイベントの記録を開始する。
- DB2 インスタンスの中で監査可能なイベントの記録を停止する。
- 記録される監査可能なイベントの区分の選択を含め、監査機能の動作を構成する。
- 現在の監査構成の記述を要求する。
- インスタンスからすべての保留している監査レコードをフラッシュし、それらを監査ログに書き込む。
- 監査ログから監査レコードをフラット・ファイルまたは ASCII 区切りファイルにフォーマットかつコピーすることにより抽出する。抽出は次の 2 つの理由のいずれかのために行われます。ログ・レコード分析の準備またはログ・レコードのプルーニングの準備のいずれかです。
- 現行監査ログにある監査レコードから余分なものを取り除く。

生成される監査レコードには様々な区分があります。監査するために使用可能なイベントの区分の記述 (下記参照) において、各区分の名前に続くのは区分タイプを識別するのに使用される 1 つの単語のキーワードであることに注目してください。監査のために使用可能なイベントの区分は次のようになります。

- 監査 (AUDIT)。監査設定が変更されるとき、または監査ログがアクセスされるときレコードを生成する。
- 許可検査 (CHECKING)。アクセスのための試みの許可検査の間にレコードを生成する、または DB2 オブジェクトか関数を操作する。
- オブジェクト保守 (OBJMAINT)。データ・オブジェクトを作成または除去するときレコードを生成する。
- 機密保護保守 (SECMAINT)。オブジェクトまたはデータベース特権、つまり DBADM 権限を付与あるいは取り消すとき、レコードを生成する。 データベース・マネージャーの機密保護構成パラメーター SYSADM_GROUP、SYSCTRL_GROUP、または SYSMAINT_GROUP が変更されるときにもまたレコードは生成される。
- システム管理 (SYSADMIN)。 SYSADM、SYSMAINT、または SYSCTRL 権限を必要とする操作が実行されるとき、レコードを生成する。
- ユーザー検証 (VALIDATE)。ユーザーを認証しているとき、またはシステムの機密保護情報を検索しているときにレコードを生成する。
- 操作コンテキスト (CONTEXT)。データベースの操作が実行されるとき、操作コンテキストを表示するレコードを生成する。この区分を使用すると、監査ログ・ファイルのより良い変換処理を可能にします。ログのイベント相関関係子フィールドも同時に使用する際、一群のイベントは 1 つのデータベース操作に戻って関連付けることができます。たとえば、動的 SQL の SQL ステートメント、静的 SQL のパッケージ識別子、つまり CONNECT のような実行されている操作タイプのインディケーターは、監査結果を分析しているときに必要なコンテキストを提供できます。

注: 操作コンテキストを与える SQL ステートメントは、かなり長くなる可能性があり、コンテキストのレコードの中にすべてが示されています。これは、CONTEXT レコードを非常に大きくすることになります。

- 失敗、成功、またはその両方を監査できます。

データベースの操作により幾らかのレコードが生成される場合があります。監査ログに生成かつ移動されるレコードの実際の数は、監査機能構成により指定された場合、記録されるイベントの区分の数によって決められます。成功、失敗、またはその両方が監査されているかどうかにもよります。この理由のため、監査するイベントの選択が重要となります。

監査機能の動作

監査機能は、データベースのインスタンスに作用を及ぼすものを含む監査可能なイベントを記録します。この理由で、監査機能は DB2 インスタンスが停止されるとしても動作可能な DB2 の独立した部分となっています。監査機能が活動状態である場合、そして停止されたインスタンスが開始されるとき、インスタンスにあるデータベース・イベントの監査が再開します。

監査ログへの監査レコードの書き込みタイミングは、インスタンスのデータベースのパフォーマンスに有効な影響を与えます。監査レコードの書き込みは、そのレコードの生成をもたらすイベントの出現により同期的にまたは非同期的に行われます。*audit_buf_sz* データベース・マネージャーの構成パラメーター値は、いつ監査レコードの書き込みが行われるかを決定します。

このパラメーターの値がゼロ (0) であると、書き込みは同期的に行われます。監査レコードを生成しているイベントは、そのレコードがディスクに書き込まれるまで待機します。それぞれのレコードに関連したこの待機は、DB2 のパフォーマンスを減少させます。

audit_buf_sz の値がゼロより大きい場合、レコードの書き込みは非同期で行われます。ゼロより大きいとき *audit_buf_sz* の値は、内部バッファーの作成に使用される 4 KB ページの数値となります。 その内部バッファーは、一群の監査レコードをディスクに書き込む前にいくつかの監査レコードを保持するために使用されます。監査イベントの結果として監査レコードを生成するステートメントは、レコードがディスクに書き込まれるまで待機することなく、その動作を継続できます。

非同期のケースでは、監査レコードを空があるバッファーにしばらくの間とどめておくこともできないわけではありません。これを拡張期間の間に起きることを防ぐには、データベース・マネージャーが監査レコードの書き込みを定期的に行わせるようにします。さらに監査機能の許可ユーザーの、明示的な要求により監査バッファーをフラッシュすることができます。

同期のレコード書き込みか、非同期のレコード書き込みかによって、エラーが発生したときに違いが出てきます。非同期モードでは、監査レコードがディスクに書き込まれる前にバッファーに入れられるので、いくつかのレコードが失われる可能性があります。同期モードでは、エラーが書き込みを防ぐことのできる監査レコードは多くて 1 つなので、レコードが 1 つ失われる可能性があります。

ERRORTYPE 監査機能パラメーターの設定により、どのように DB2 と監査機能の間でエラーを管理するかを制御します。監査機能が活動状態にあり、監査機能パラメーター ERRORTYPE の設定が AUDIT であるとき、その監査機能は DB2 の他の部分と同じように扱われます。監査レコードを、正常に考慮され、ステートメントに関連した監査イベント (同期モードではディスク、非同期モードでは監査バッファー) に書き込まなければなりません。このモードで実行時にエラーに遭遇したときはいつでも、 1 つの負の SQLCODE を、監査レコードを生成するステートメントのアプリケーションに戻します。エラー・タイプが NORMAL にセットされると、次に db2audit にあるエラーが無視され、その操作の SQLCODE が戻されます。 ERRORTYPE の監査機能パラメーター (および他の関連するパラメーター) に関する付加情報については、 289ページの『監査機能使用のシナリオ』を参照してください。

API または SQL ステートメントおよび DB2 インスタンスの監査設定に応じて、特定のイベントに対して監査レコードが何も生成されなかったり、 1 つまたはいくつかの監査レコードが生成されたりします。たとえば、SELECT 副照会による SQL UPDATE ステートメントは、結果として 1 つの監査レコードを生じ、そこには表の UPDATE 特権に対する許可検査の結果を含んでいます。さらに、別の監査レコードには、表の SELECT 特権に対する許可検査の結果が含まれています。

動的なデータ操作言語 (DML) のステートメントの場合、ステートメントが準備された時点ですべての許可検査の監査レコードが生成されています。同一ユーザーによるステートメントの再使用では、その時に許可検査が行われないため再度監査されることはありません。しかしながら、特権情報を含んでいるカタログ表のいずれかに変更が行われた場合には、次の作業単位において、キャッシュされた動的 SQL ステートメントのステートメント特権は検査され、 1 つ以上の新規監査レコードが作成されます。

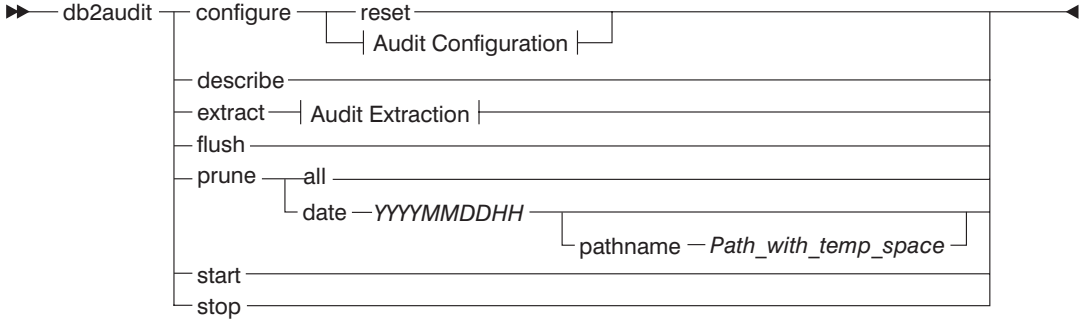
静的 DML ステートメントだけを含んでいるパッケージの場合、監査レコードを生成する可能性のある監査可能なイベントだけが、ユーザーの特権でそのパッケージを実行できるかどうかを調べるために許可検査となります。パッケージがプリコンパイルされるかまたはバインドされる時、パッケージにおける静的な SQL ステートメントのために必要とされる許可検査および考えられる監査レコードの作成は実行されます。パッケージ内部の静的 SQL ステートメントの実行は、監査することはできません。ユーザーにより明示的に、またはシステムにより暗黙的に 1 つのパッケージがバインドされるとき、静的 SQL ステートメントにより必要とされる許可検査のために複数の監査レコードが生成されます。

許可検査が実行時間に行われるステートメント (たとえば、データ定義言語 (DDL)、GRANT、および REVOKE ステートメント) の場合、これらのステートメントが使用されるときにはいつでも監査レコードが生成されます。

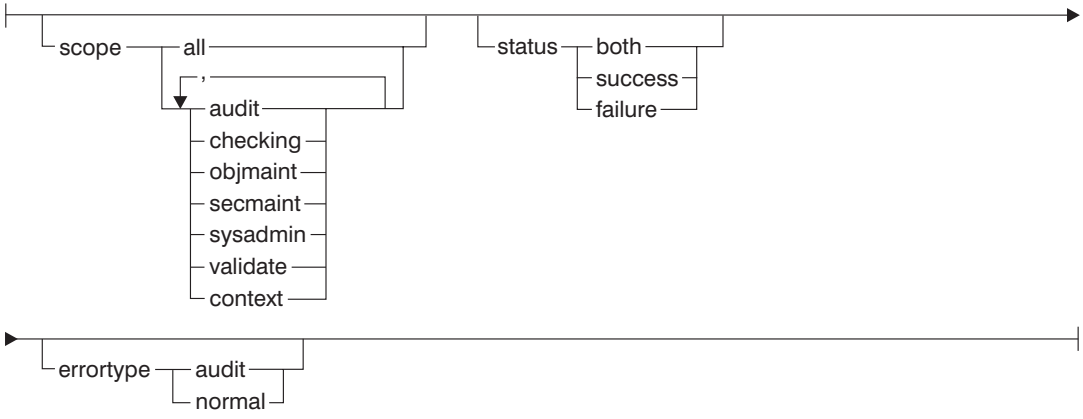
注: DDL を実行するとき、監査レコードのすべてのイベント (コンテキスト・イベントを除く) のために記録されたセクション数は、ステートメントの実際のセクション数がどんなものであってもゼロ (0) にリセットされます。

監査機能使用のシナリオ

次の構文図の各部分を検討することは、監査機能の使用方法を理解する助けになります。



Audit Configuration:



Audit Extraction:

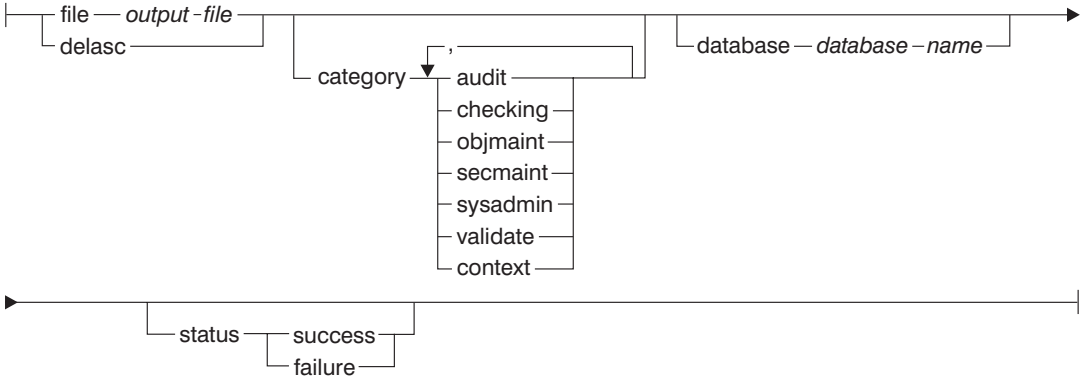


図 7. DB2AUDIT 構文

下記に、パラメーターの説明と暗黙の使用を示します。

configure
このパラメーターを使用すれば、インスタンスの `security` サブディレクトリ

ーにある db2audit.cfg 構成ファイルを変更することができます。このファイルへの変更は、インスタンスがシャットダウンしているときでさえも行うことができます。インスタンスが活動状態にある時に行われる更新は、区分の至るところで DB2 によりなされる監査に強力に影響します。監査機能が開始していて、かつ監査可能なイベントの *audit* 区分が監査されているならば、構成ファイルの configure 処置により監査レコードの作成が行われます。

下記に構成ファイルでの考えられる処置を示しています。

- RESET。この処置によって構成ファイルは初期構成に戻される (SCOPE は CONTEXT を除くすべての区分を、 STATUS は FAILURE、ERRORTYPE は NORMAL、そして監査機能は OFF を示している) 。 元の監査構成ファイルが脱落したか損傷したならば、これによって新しい監査構成ファイルを作成することになります。
- SCOPE。この処置により、監査されるがイベントのどの区分かまたは複数のどの区分かを指定する。さらに、これにより特定のフォーカスに絞って監査することを可能にし、ログの増加を抑えることができます。ログに記録されるイベントの数とタイプをできるだけ制限することをお勧めします。そうしないと、監査ログは急速に増加することになります。

  注: デフォルト SCOPE が CONTEXT を除いたすべての区分となり、そしてその結果として急速に生成されるレコードを生み出すかもしれないことにぜひ注意してください。モード (同期または非同期) とともに、区分の選択によっては、パフォーマンスがかなり縮小し、ディスク要件が著しく増加するという結果を生じさせる場合があります。
- STATUS。この操作により、正常終了のイベントまたは失敗したイベントのいずれかを記録するか、それだけでなく正常終了および失敗した両方のイベントを記録すべきかどうかを指定する。

  注: 操作の状況が知らされる前に、コンテキスト・イベントは発生します。それで、そのようなイベントはこのパラメーターに関連した値に関係なく記録されます。
- ERRORTYPE。この処置によって、監査エラーをユーザーに戻すかまたは無視するかどうかを指定する。このパラメーター値は、次のようになる可能性があります。
  - AUDIT。監査機能の中で発生するエラーを含むすべてのエラーは、 DB2 によって管理され、それから負の SQLCODE のすべては呼び出し元に折り返し報告される。
  - NORMAL。 db2audit によって生成されたすべてのエラーは無視され、実行されている操作に関連したエラーの SQLCODE だけがアプリケーションに戻される。

describe

このパラメーターは、標準出力に対して現行監査構成情報および状況を表示する。

extract このパラメーターを使用すると、監査ログから指示された宛先まで監査レコードの移動を可能にします。どんな任意選択の文節も指定されない場合、監査レコードのすべてが抽出され、フラットのレポート・ファイル内に置かれます。『extract』 パラメーターが指定されなければ、監査レコードは `security` ディレクトリーの `db2audit.out` と呼ばれるファイルに置かれます。 *output_file* がすでに存在する場合、エラー・メッセージは戻されます。

下記に、抽出する際に使用できる可能なオプションを示します。

- FILE。抽出された監査レコードを、ファイル (output_file) に置く。
- DELASC。抽出された監査レコードを、 DB2 リレーショナル表にロードするのに適切な区切り文字付き ASCII 形式で配置する。その出力は各区分ごとに 1 つ、独立したファイルに置かれます。ファイル名は次のとおりです。
  - audit.del
  - checking.del
  - objmaint.del
  - secmaint.del
  - sysadmin.del
  - validate.del
  - context.del

  さらに、DELASC 選択を使用すれば、監査ログから抽出するとき、デフォルトの監査文字ストリング区切り文字 (『0xff』) を指定変更することができます。それによって、DELASC DELIMITER の後に、監査レコードを保持する表にロードするための準備で使用したい新しい区切り文字を使用することができるようになります。この新規ロードの区切り文字は、単一文字 (たとえば、!) かまたは 16 進数で示される 4 バイト・ストリング (たとえば、0xff) のいずれかとなります。詳細については、312ページの『監査機能のヒントと技法』を参照してください。
- CATEGORY。監査イベントの指定された区分の監査レコードが抽出される。これが指定されない場合、すべての区分が抽出に対して適格となります。
- DATABASE。指定されたデータベースの監査レコードが抽出される。これが指定されない場合、すべてのデータベースが抽出に対して適格となります。
- STATUS。指定された状況の監査レコードが抽出される。これが指定されない場合、すべてのレコードが抽出に対して適格となります。

flush このパラメーターは、任意の保留監査レコードを監査ログに書き込むよう強制する。同様に、監査機能がエラー状態にある場合、監査状況はそのエンジンにおいて「ログ利用不可」から「ログ作動可能」の状況にリセットされます。

prune このパラメーターを使用すると、監査ログから監査レコードの削除を可能にする。監査機能が活動状態にあり、『audit』 区分のイベントが監査のために指定されている場合、その時には監査レコードは監査ログが簡潔にされた後に記録されます。

下記に、プルーニングする際に使用できる可能なオプションを示します。

- ALL。監査ログにあるすべての監査レコードが削除される。
- DATE yyyymmddhh。指定された日付 / 時間に、または指定された日付 / 時間以前に発生したすべての監査レコードを監査ログから削除するよう指定できる。ユーザーは、監査記録をプルーニングしているとき、監査機能が一時スペースとして使用する

  ```
  pathname
  ```

  を任意選択で指定することができます。この一時スペースを使用すると、監査ログのあるディスク容量がいっぱいで、プルーニング操作を行わせるのに十分なスペースがない時にも監査ログのプルーニングを行うことができるようにします。

start このパラメーターを使用すると、監査機能が db2audit.cfg ファイルの内容に基づくイベントの監査を始めることができます。パーティションされた DB2 インスタンスにおいて、この文節が指定されるとき、監査はすべてのパーティションで始まります。 『audit』 区分のイベントが監査のために指定されている場合、その監査機能が開始される際、そのとき 1 つの監査レコードが記録されます。

stop このパラメーターを使用すると、監査機能がイベントを監査するのを停止することができます。パーティションされた DB2 インスタンスにおいて、この文節が指定されるとき、監査はすべてのパーティションで停止します。 『audit』 区分のイベントが監査のために指定されている場合、その監査機能が停止される際、そのとき 1 つの監査レコードが記録されます。

監査機能のメッセージ

SQL1322N 監査ログ・ファイルへの書き込み時にエラーが発生しました。

説明: 監査イベントを監査ログ・ファイルに記録するために呼び出すとき、 DB2 監査機能はエラーに遭遇しました。監査ログがあるファイル・システムにはスペースがありません。

ユーザーの処置: システム管理者は、このファイル・システムのスペースを空けるか、または監査ログのサイズを削減するために余分なものを取り除くようにすべきです。

より大きなスペースが使用可能になるとき、db2audit を使用してメモリー内のすべてのデータをフラッシュアウトし、それから監査を作動可能状態にリセットします。適切な抽出が発生した、つまり削除済みレコードが回復可能でないとき、ログのプルーニングの前にログのコピーが行われたことを確認してください。

sqlcode: -1322

sqlstate: 50830

SQL1323N 監査構成ファイルにアクセスしているときエラーが発生しました。

説明: 監査構成ファイル (db2audit.cfg) を開くことができないか、または無効でした。このエラーについて考えられる理由は、db2audit.cfg ファイルが存在しないか、または損傷をしていたかのいずれかです。

ユーザーの処置: 下記の処置のいずれかを行ってください。

- 保管されたバージョンのファイルから復元する。
- 以下を発行して、監査機能の構成ファイルをリセットする。

```
db2audit reset
```

sqlcode: -1323

sqlstate: 57019

監査機能のレコード設計

監査レコードが DELASC 抽出オプションを使用して監査ログから抽出されるとき、それぞれのレコードは下記の表で示されるフォーマットのいずれかとなります。それぞれの表は、サンプル・レコードの内容を示すことによって始まります。レコードの各項目の記述は、関連する表において一度に 1 つの行で示されます。その項目が重要である場合、項目の名前は強調表示 (**太字**) されています。これらの項目には、ユーザーにとって非常に関心ある情報が含まれています。

注:

1. サンプル・レコードのすべてのフィールドが、必ずしも値を持っているとは限りません。
2. 中には、『Access Attempted』 のようにビットマップとして区切り文字付き ASCII 形式で保管されるフィールドもあります。ただし、現在のフラットなレポート・ファイルにおいて、それらのフィールドはビットマップ値を表す一連のストリングとして表示されます。

表 6. AUDIT イベントの監査レコード設計

timestamp=1998-06-24-11.54.05.151232;**category=AUDIT;audit event=START;**
event correlator=0;event status=0;
userid=boss;authid=BOSS;		

名前	**フォーマット**	**記述**
Timestamp	CHAR(26)	監査イベントの日付と時刻。
Category	CHAR(8)	監査イベントの区分。可能な値は以下のとおりです。 AUDIT
Audit Event	VARCHAR(32)	特定の監査イベント。 有効な値には、CONFIGURE、DB2AUD、EXTRACT、FLUSH、PRUNE、START、STOP、および UPDATE_ADMIN_CFG が含まれます。

表 6. AUDIT イベントの監査レコード設計 (続き)

timestamp=1998-06-24-11.54.05.151232;**category=AUDIT;audit event=START;**
event correlator=0;event status=0;
userid=boss;authid=BOSS;		

名前	**フォーマット**	**記述**
Event Correlator	INTEGER	監査している操作のための相関識別子。単一イベントに関連した監査レコードが何かを識別するために使用できます。
Event Status	INTEGER	監査イベントの状況、次のような 1 つの SQLCODE で表されます。 Successful event > = 0 Failed event < 0
User ID	VARCHAR(1024)	監査イベントの時刻でのユーザー ID。
Authorization ID	VARCHAR(128)	監査イベントの時刻での許可 ID。

表 7. CHECKING イベントの監査レコード設計

timestamp=1998-06-24-08.42.11.622984;**category=CHECKING;audit event=CHECKING_OBJECT;**
event correlator=2;**event status=0;**
database=FOO;userid=boss;authid=BOSS;
application id=*LOCAL.newton.980624124210;application name=testapp;
object name=FOO;object type=DATABASE;
access approval reason=DATABASE;access attempted=CONNECT;		

名前	**フォーマット**	**記述**
Timestamp	CHAR(26)	監査イベントの日付と時刻。
Category	CHAR(8)	監査イベントの区分。可能な値は以下のとおりです。 CHECKING
Audit Event	VARCHAR(32)	特定の監査イベント。 有効な値は、CHECKING_OBJECT および CHECKING_FUNCTION です。
Event Correlator	INTEGER	監査している操作のための相関識別子。単一イベントに関連した監査レコードが何かを識別するために使用できます。
Event Status	INTEGER	監査イベントの状況、次のような 1 つの SQLCODE で表されます。 Successful event > = 0 Failed event < 0
Database Name	CHAR(8)	どのイベントが生成されたかを示すデータベース名。インスタンス・レベルの監査イベントだった場合はブランクとなります。
User ID	VARCHAR(1024)	監査イベントの時刻でのユーザー ID。

表 7. CHECKING イベントの監査レコード設計 (続き)

```
timestamp=1998-06-24-08.42.11.622984;category=CHECKING;audit event=CHECKING_OBJECT;
  event correlator=2;event status=0;
  database=FOO;userid=boss;authid=BOSS;
  application id=*LOCAL.newton.980624124210;application name=testapp;
  object name=FOO;object type=DATABASE;
  access approval reason=DATABASE;access attempted=CONNECT;
```

名前	フォーマット	記述
Authorization ID	VARCHAR(128)	監査イベントの時刻での許可 ID。
Origin Node Number	SMALLINT	監査イベントが発生したところのノード番号。
Coordinator Node Number	SMALLINT	調整プログラムのノード番号。
Application ID	VARCHAR (255)	監査イベントが発生した時刻で使用していたアプリケーション ID。
Application Name	VARCHAR (1024)	監査イベントが発生した時刻で使用していたアプリケーション名。
Package Schema	VARCHAR (128)	監査イベントの時刻で使用していたパッケージのスキーマ。
Package Name	VARCHAR (128)	監査イベントが発生した時刻で使用していたパッケージ名。
Package Section Number	SMALLINT	監査イベントが発生した時刻で使用されていたパッケージのセクション番号。
Object Schema	VARCHAR (128)	どの監査イベントが生成されたかについてのオブジェクトのスキーマ。
Object Name	VARCHAR (128)	どの監査イベントが生成されたかについてのオブジェクト名。
Object Type	VARCHAR (32)	どの監査イベントが生成されたかについてのオブジェクト・タイプ。有効な値を次に示します。 TABLE、VIEW、ALIAS、FUNCTION、INDEX、PACKAGE、DATA_TYPE、NODEGROUP、SCHEMA、STORED_PROCEDURE、BUFFERPOOL、TABLESPACE、EVENT_MONITOR、TRIGGER、DATABASE、INSTANCE、FOREIGN_KEY、PRIMARY_KEY、UNIQUE_CONSTRAINT、CHECK_CONSTRAINT、WRAPPER、SERVER、NICKNAME、USER MAPPING、SERVER OPTION、TRANSFORM、TYPE MAPPING、FUNCTION MAPPING、SUMMARY TABLES、および NONE。
Access Approval Reason	CHAR(18)	なぜアクセスがこの監査イベントで承認されたのかその理由を示します。有効な値を次に示します。この表に続く最初のリストにそれらを示しています。

表 7. CHECKING イベントの監査レコード設計 (続き)

```
timestamp=1998-06-24-08.42.11.622984;category=CHECKING;audit event=CHECKING_OBJECT;
  event correlator=2;event status=0;
  database=FOO;userid=boss;authid=BOSS;
  application id=*LOCAL.newton.980624124210;application name=testapp;
  object name=FOO;object type=DATABASE;
  access approval reason=DATABASE;access attempted=CONNECT;
```

名前	フォーマット	記述
Access Attempted	CHAR(18)	試みられたアクセスのタイプを示します。有効な値を次に示します。この表に続く 2 番目のリストにそれらを示しています。

下記は、有効な CHECKING アクセス承認理由のリストです。

0x0000000000000001 ACCESS DENIED
アクセスは承認されません。それどころか、拒否されました。

0x0000000000000002 SYSADM
アクセスは承認されます。アプリケーション / ユーザーは SYSADM 権限を持ちます。

0x0000000000000004 SYSCTRL
アクセスは承認されます。アプリケーション / ユーザーは SYSCTRL 権限を持ちます。

0x0000000000000008 SYSMAINT
アクセスは承認されます。アプリケーション / ユーザーは SYSMAINT 権限を持ちます。

0x0000000000000010 DBADM
アクセスは承認されます。アプリケーション / ユーザーは DBADM 権限を持ちます。

0x0000000000000020 DATABASE PRIVILEGE
アクセスは承認されます。アプリケーション / ユーザーはこのデータベースに関して明示的な権限を持ちます。

0x0000000000000040 OBJECT PRIVILEGE
アクセスは承認されます。アプリケーション / ユーザーはオブジェクトまたは関数に関して明示的な権限を持ちます。

0x0000000000000080 DEFINER
アクセスは承認されます。アプリケーション / ユーザーは、オブジェクトまたは関数の定義をするものとなります。

0x0000000000000100 OWNER
アクセスは承認されます。アプリケーション / ユーザーは、オブジェクトまたは関数の所有者となります。

0x0000000000000200 CONTROL
アクセスは承認されます。アプリケーション / ユーザーは、オブジェクトまたは関数に関する CONTROL 権限を持ちます。

0x0000000000000400 BIND
アクセスは承認されます。アプリケーション / ユーザーは、パッケージに関するバインド権限を持ちます。

下記は、有効な CHECKING アクセス未遂のタイプのリストです。

0x0000000000000002 ALTER
オブジェクトを更新しようとします。

0x0000000000000004 DELETE
オブジェクトを削除しようとします。

0x0000000000000008 INDEX
索引を使用しようとします。

0x0000000000000010 INSERT
オブジェクトの中に挿入しようとします。

0x0000000000000020 SELECT
表または視点を照会しようとします。

0x0000000000000040 UPDATE
オブジェクトのデータを更新しようとします。

0x0000000000000080 REFERENCE
オブジェクト間の参照制約を確立しようとします。

0x0000000000000100 CREATE
オブジェクトを作成しようとします。

0x0000000000000200 DROP
オブジェクトを除去しようとします。

0x0000000000000400 CREATEIN
別のスキーマ内にオブジェクトを作成しようとします。

0x0000000000000800 DROPIN
別のスキーマ内に見いだされるオブジェクトを除去しようとします。

0x0000000000001000 ALTERIN
別のスキーマ内に見いだされるオブジェクトを更新または変更しようとします。

0x0000000000002000 EXECUTE
アプリケーションを実行または稼働しようとします。

0x0000000000004000 BIND
アプリケーションをバインドまたは準備しようとします。

0x0000000000008000 SET EVENT MONITOR
イベント・モニターのスイッチをセットしようとします。

0x0000000000010000 SET CONSTRAINTS
オブジェクトに関する制約をセットしようとします。

0x0000000000020000 COMMENT ON
オブジェクトに関する注釈を作成しようとします。

0x0000000000040000 GRANT
別のユーザー ID にオブジェクトに関する特権を付与しようとします。

0x0000000000080000 REVOKE
オブジェクトに関する特権をユーザー ID から取り消そうとします。

0x0000000000100000 LOCK
オブジェクトをロックしようとします。

0x0000000000200000 RENAME
オブジェクトを名前変更しようとします。

0x0000000000400000 CONNECT
オブジェクトに接続しようとします。

0x0000000000800000 Member of SYS Group
SYS グループのメンバーをアクセスまたは使用しようとします。

0x0000000001000000 Access All
保持されているオブジェクトに対して必要なすべての特権を使用して、ステートメントを実行しようとします (DBADM/SYSADM でのみ使用されます)。

0x0000000002000000 Drop All
複数のオブジェクトを除去しようとします。

0x0000000004000000 LOAD
表スペースに表をロードしようとします。

0x0000000008000000 USE
表スペースに表を作成しようとします。

表 8. OBJMAINT イベントの監査レコード設計

```
timestamp=1998-06-24-08.42.41.957524;category=OBJMAINT;audit event=CREATE_OBJECT;
  event correlator=3;event status=0;
  database=FOO;userid=boss;authid=BOSS;
  application id=*LOCAL.newton.980624124210;application name=testapp;
  package schema=NULLID;package name=SQLC28A1;
  package section=0;object schema=BOSS;object name=AUDIT;object type=TABLE;
```

名前	フォーマット	記述
Timestamp	CHAR(26)	監査イベントの日付と時刻。

表 8. OBJMAINT イベントの監査レコード設計 (続き)

```
timestamp=1998-06-24-08.42.41.957524;category=OBJMAINT;audit event=CREATE_OBJECT;
  event correlator=3;event status=0;
  database=FOO;userid=boss;authid=BOSS;
  application id=*LOCAL.newton.980624124210;application name=testapp;
  package schema=NULLID;package name=SQLC28A1;
  package section=0;object schema=BOSS;object name=AUDIT;object type=TABLE;
```

名前	フォーマット	記述
Category	CHAR(8)	監査イベントの区分。可能な値は以下のとおりです。 OBJMAINT
Audit Event	VARCHAR(32)	特定の監査イベント。 有効な値は、CREATE_OBJECT、RENAME_OBJECT、および DROP_OBJECT です。
Event Correlator	INTEGER	監査している操作のための相関識別子。単一イベントに関連した監査レコードが何かを識別するために使用できます。
Event Status	INTEGER	監査イベントの状況、次のような 1 つの SQLCODE で表されます。 Successful event > = 0 Failed event < 0
Database Name	CHAR(8)	どのイベントが生成されたかを示すデータベース名。インスタンス・レベルの監査イベントだった場合はブランクとなります。
User ID	VARCHAR(1024)	監査イベントの時刻でのユーザー ID。
Authorization ID	VARCHAR(128)	監査イベントの時刻での許可 ID。
Origin Node Number	SMALLINT	監査イベントが発生したところのノード番号。
Coordinator Node Number	SMALLINT	調整プログラムのノード番号。
Application ID	VARCHAR (255)	監査イベントが発生した時刻で使用していたアプリケーション ID。
Application Name	VARCHAR (1024)	監査イベントが発生した時刻で使用していたアプリケーション名。
Package Schema	VARCHAR (128)	監査イベントの時刻で使用していたパッケージのスキーマ。
Package Name	VARCHAR (128)	監査イベントが発生した時刻で使用していたパッケージ名。
Package Section Number	SMALLINT	監査イベントが発生した時刻で使用されていたパッケージのセクション番号。
Object Schema	VARCHAR (128)	どの監査イベントが生成されたかについてのオブジェクトのスキーマ。

表 8. OBJMAINT イベントの監査レコード設計 (続き)

```
timestamp=1998-06-24-08.42.41.957524;category=OBJMAINT;audit event=CREATE_OBJECT;
  event correlator=3;event status=0;
  database=FOO;userid=boss;authid=BOSS;
  application id=*LOCAL.newton.980624124210;application name=testapp;
  package schema=NULLID;package name=SQLC28A1;
  package section=0;object schema=BOSS;object name=AUDIT;object type=TABLE;
```

名前	フォーマット	記述
Object Name	VARCHAR (128)	どの監査イベントが生成されたかについてのオブジェクト名。
Object Type	VARCHAR (32)	どの監査イベントが生成されたかについてのオブジェクト・タイプ。有効な値を次に示します。 TABLE、VIEW、ALIAS、FUNCTION、INDEX、PACKAGE、DATA_TYPE、NODEGROUP、SCHEMA、STORED_PROCEDURE、BUFFERPOOL、TABLESPACE、EVENT_MONITOR、TRIGGER、DATABASE、INSTANCE、FOREIGN_KEY、PRIMARY_KEY、UNIQUE_CONSTRAINT、CHECK_CONSTRAINT、WRAPPER、SERVER、NICKNAME、USER MAPPING、SERVER OPTION、TRANSFORM、TYPE MAPPING、FUNCTION MAPPING、SUMMARY TABLES、および NONE。

表 9. SECMAINT イベントの監査レコード設計

```
timestamp=1998-06-24-11.57.45.188101;category=SECMAINT;audit event=GRANT;
  event correlator=4;event status=0;
  database=FOO;userid=boss;authid=BOSS;
  application id=*LOCAL.boss.980624155728;application name=db2bp;
  package schema=NULLID;package name=SQLC28A1;
  package section=0;object schema=BOSS;object name=T1;object type=TABLE;
  grantor=BOSS;grantee=WORKER;grantee type=USER;privilege=SELECT;
```

名前	フォーマット	記述
Timestamp	CHAR(26)	監査イベントの日付と時刻。
Category	CHAR(8)	監査イベントの区分。可能な値は以下のとおりです。 SECMAINT
Audit Event	VARCHAR(32)	特定の監査イベント。 有効な値は、GRANT、REVOKE、IMPLICIT_GRANT、IMPLICIT_REVOKE、および UPDATE_DBM_CFG です。
Event Correlator	INTEGER	監査している操作のための相関識別子。単一イベントに関連した監査レコードが何かを識別するために使用できます。

表 9. SECMAINT イベントの監査レコード設計 (続き)

```
timestamp=1998-06-24-11.57.45.188101;category=SECMAINT;audit event=GRANT;
  event correlator=4;event status=0;
  database=FOO;userid=boss;authid=BOSS;
  application id=*LOCAL.boss.980624155728;application name=db2bp;
  package schema=NULLID;package name=SQLC28A1;
  package section=0;object schema=BOSS;object name=T1;object type=TABLE;
  grantor=BOSS;grantee=WORKER;grantee type=USER;privilege=SELECT;
```

名前	フォーマット	記述
Event Status	INTEGER	監査イベントの状況、次のような 1 つの SQLCODE で表されます。 Successful event > = 0 Failed event < 0
Database Name	CHAR(8)	どのイベントが生成されたかを示すデータベース名。インスタンス・レベルの監査イベントだった場合はブランクとなります。
User ID	VARCHAR(1024)	監査イベントの時刻でのユーザー ID。
Authorization ID	VARCHAR(128)	監査イベントの時刻での許可 ID。
Origin Node Number	SMALLINT	監査イベントが発生したところのノード番号。
Coordinator Node Number	SMALLINT	調整プログラムのノード番号。
Application ID	VARCHAR (255)	監査イベントが発生した時刻で使用していたアプリケーション ID。
Application Name	VARCHAR (1024)	監査イベントが発生した時刻で使用していたアプリケーション名。
Package Schema	VARCHAR (128)	監査イベントの時刻で使用していたパッケージのスキーマ。
Package Name	VARCHAR (128)	監査イベントが発生した時刻で使用していたパッケージ名。
Package Section Number	SMALLINT	監査イベントが発生した時刻で使用されていたパッケージのセクション番号。
Object Schema	VARCHAR (128)	どの監査イベントが生成されたかについてのオブジェクトのスキーマ。
Object Name	VARCHAR (128)	どの監査イベントが生成されたかについてのオブジェクト名。

表 9. SECMAINT イベントの監査レコード設計 (続き)

```
timestamp=1998-06-24-11.57.45.188101;category=SECMAINT;audit event=GRANT;
  event correlator=4;event status=0;
  database=FOO;userid=boss;authid=BOSS;
  application id=*LOCAL.boss.980624155728;application name=db2bp;
  package schema=NULLID;package name=SQLC28A1;
  package section=0;object schema=BOSS;object name=T1;object type=TABLE;
  grantor=BOSS;grantee=WORKER;grantee type=USER;privilege=SELECT;
```

名前	フォーマット	記述
Object Type	VARCHAR (32)	どの監査イベントが生成されたかについてのオブジェクト・タイプ。有効な値を次に示します。 TABLE、VIEW、ALIAS、FUNCTION、INDEX、PACKAGE、DATA_TYPE、NODEGROUP、SCHEMA、STORED_PROCEDURE、BUFFERPOOL、TABLESPACE、EVENT_MONITOR、TRIGGER、DATABASE、INSTANCE、FOREIGN_KEY、PRIMARY_KEY、UNIQUE_CONSTRAINT、CHECK_CONSTRAINT、WRAPPER、SERVER、NICKNAME、USER MAPPING、SERVER OPTION、TRANSFORM、TYPE MAPPING、FUNCTION MAPPING、SUMMARY TABLES、および NONE。
Grantor	VARCHAR (128)	授与者の識別コード。
Grantee	VARCHAR (128)	特権または権限が付与されたか取り消された被授与者の識別コード。
Grantee Type	VARCHAR (32)	付与されたかまたは取り消された被授与者のタイプ。有効な値は、USER、GROUP、または BOTH です。
Privilege または Authority	CHAR(18)	付与されたか取り消された特権または権限のタイプを示します。有効な値を、この表に続くリストに示しています。

下記は、有効な SECMAINT 特権または権限のリストです。

0x0000000000000001 Control Table
表に関して付与または取り消された制御特権。

0x0000000000000002 ALTER TABLE
表を更新するために付与または取り消された特権。

0x0000000000000004 ALTER TABLE with GRANT
特権の授与が許可されている表を更新するために付与または取り消された特権。

0x0000000000000008 DELETE TABLE
表を除去するために付与または取り消された特権。

0x0000000000000010 DELETE TABLE with GRANT
特権の授与が許可されている表を除去するために付与または取り消された特権。

0x0000000000000020 Table Index
索引に関して付与または取り消された特権。

0x0000000000000040 Table Index with GRANT
特権の授与が許可されている索引に関して付与または取り消された特権。

0x0000000000000080 Table INSERT
表への挿入に関して付与または取り消された特権。

0x0000000000000100 Table INSERT with GRANT
特権の授与が許可されている表への挿入に関して付与または取り消された特権。

0x0000000000000200 Table SELECT
表での選択に関して付与または取り消された特権。

0x0000000000000400 Table SELECT with GRANT
特権の授与が許可されている表での選択に関して付与または取り消された特権。

0x0000000000000800 Table UPDATE
表への更新に関して付与または取り消された特権。

0x0000000000001000 Table UPDATE with GRANT
特権の授与が許可されている表への更新に関して付与または取り消された特権。

0x0000000000002000 Table REFERENCE
表への参照に関して付与または取り消された特権。

0x0000000000004000 Table REFERENCE with GRANT
特権の授与が許可されている表への参照に関して付与または取り消された特権。

0x0000000000008000 Package BIND
パッケージに関して付与または取り消されたバインド特権。

0x0000000000010000 Package EXECUTE
パッケージに関して付与または取り消された実行特権。

0x0000000000020000 CREATEIN Schema
スキーマに関して付与または取り消された作成特権。

0x0000000000040000 CREATEIN Schema with GRANT
特権の授与が許可されているスキーマに関して付与または取り消された作成特権。

0x0000000000080000 DROPIN Schema
スキーマに関して付与または取り消されたドロップイン特権。

0x0000000000100000 DROPIN Schema with GRANT
特権の授与が許可されているスキーマに関して付与または取り消されたドロップイン特権。

0x0000000000200000 ALTERIN Schema
スキーマに関して付与または取り消された ALTERIN 特権。

0x0000000000400000 ALTERIN Schema with GRANT
特権の授与が許可されているスキーマに関して付与または取り消された ALTERIN 特権。

0x0000000000800000 DBADM Authority
付与または取り消された DBADM 権限。

0x0000000001000000 CREATETAB Authority
付与または取り消された CREATETAB 権限。

0x0000000002000000 BINDADD Authority
付与または取り消された BINDADD 権限。

0x0000000004000000 CONNECT Authority
付与または取り消された CONNECT 権限。

0x0000000008000000 Create not fenced Authority
付与または取り消された非分離の作成権限。

0x0000000010000000 Implicit Schema Authority
付与または取り消された暗黙的スキーマ権限。

0x0000000020000000 Server PASSTHRU
このサーバー (連合データベースのデータ・ソース) でパススルー機能を使用するために、付与または取り消された特権。

0x0000000100000000 Table Space USE
表スペースに表を作成するために付与または取り消された特権。

0x0000000200000000 Table Space USE with GRANT
特権の授与が許可されている表スペースに表を作成するために付与または取り消された特権。

0x0000000400000000 Column UPDATE
表の 1 つ以上の特定の列への更新に関して付与または取り消された特権。

0x0000000800000000 Column UPDATE with GRANT
特権の授与が許可されている表の 1 つ以上の特定の列への更新に関して付与または取り消された特権。

0x0000001000000000 Column REFERENCE
表の 1 つ以上の特定の列への参照に関して付与または取り消された特権。

0x0000002000000000 Column REFERENCE with GRANT
特権の授与が許可されている表の 1 つ以上の特定の列への参照に関して付与または取り消された特権。

0x0000004000000000 LOAD Authority
付与または取り消された LOAD 権限。

表 10. SYSADMIN イベントの監査レコード設計

```
timestamp=1998-06-24-11.54.04.129923;category=SYSADMIN;audit event=DB2AUDIT;
  event correlator=1;event status=0;
  userid=boss;authid=BOSS;
  application id=*LOCAL.boss.980624155404;application name=db2audit;
```

名前	フォーマット	記述
Timestamp	CHAR(26)	監査イベントの日付と時刻。
Category	CHAR(8)	監査イベントの区分。可能な値は以下のとおりです。 SYSADMIN
Audit Event	VARCHAR(32)	特定の監査イベント。 有効な値を、この表に続くリストに示しています。
Event Correlator	INTEGER	監査している操作のための相関識別子。単一イベントに関連した監査レコードが何かを識別するために使用できます。
Event Status	INTEGER	監査イベントの状況、次のような 1 つの SQLCODE で表されます。 Successful event > = 0 Failed event < 0
Database Name	CHAR(8)	どのイベントが生成されたかを示すデータベース名。インスタンス・レベルの監査イベントだった場合はブランクとなります。
User ID	VARCHAR(1024)	監査イベントの時刻でのユーザー ID。
Authorization ID	VARCHAR(128)	監査イベントの時刻での許可 ID。
Origin Node Number	SMALLINT	監査イベントが発生したところのノード番号。
Coordinator Node Number	SMALLINT	調整プログラムのノード番号。
Application ID	VARCHAR (255)	監査イベントが発生した時刻で使用していたアプリケーション ID。
Application Name	VARCHAR (1024)	監査イベントが発生した時刻で使用していたアプリケーション名。
Package Schema	VARCHAR (128)	監査イベントの時刻で使用していたパッケージのスキーマ。
Package Name	VARCHAR (128)	監査イベントが発生した時刻で使用していたパッケージ名。

表 10. SYSADMIN イベントの監査レコード設計 (続き)

timestamp=1998-06-24-11.54.04.129923;**category=SYSADMIN;audit event=DB2AUDIT;**
event correlator=1;event status=0;
userid=boss;authid=BOSS;
application id=*LOCAL.boss.980624155404;application name=db2audit;		

名前	**フォーマット**	**記述**
Package Section Number	SMALLINT	監査イベントが発生した時刻で使用されていたパッケージのセクション番号。

下記は、有効な SYSADMIN 監査イベントのリストです。

表 11. SYSADMIN 監査イベント

START_DB2
STOP_DB2
CREATE_DATABASE
DROP_DATABASE
UPDATE_DBM_CFG
UPDATE_DB_CFG
CREATE_TABLESPACE
DROP_TABLESPACE
ALTER_TABLESPACE
RENAME_TABLESPACE
CREATE_NODEGROUP
DROP_NODEGROUP
ALTER_NODEGROUP
CREATE_BUFFERPOOL
DROP_BUFFERPOOL
ALTER_BUFFERPOOL
CREATE_EVENT_MONITOR
DROP_EVENT_MONITOR
ENABLE_MULTIPAGE
MIGRATE_DB_DIR
DB2TRC
DB2SET
ACTIVATE_DB
ADD_NODE
BACKUP_DB
CATALOG_NODE
CATALOG_DB
CATALOG_DCS_DB
CHANGE_DB_COMMENT
DEACTIVATE_DB
DROP_NODE_VERIFY
FORCE_APPLICATION
GET_SNAPSHOT
LIST_DRDA_INDOUBT_TRANSACTIONS
MIGRATE_DB
RESET_ADMIN_CFG
RESET_DB_CFG
RESET_DBM_CFG
RESET_MONITOR
RESTORE_DB
ROLLFORWARD_DB
SET_RUNTIME_DEGREE
SET_TABLESPACE_CONTAINERS
UNCATALOG_DB
UNCATALOG_DCS_DB
UNCATALOG_NODE
UPDATE_ADMIN_CFG
UPDATE_MON_SWITCHES
LOAD_TABLE
DB2AUDIT
SET_APPL_PRIORITY
CREATE_DB_AT_NODE
KILLDBM
MIGRATE_SYSTEM_DIRECTORY
DB2REMOT
DB2AUD
MERGE_DBM_CONFIG_FILE
UPDATE_CLI_CONFIGURATION
OPEN_TABLESPACE_QUERY
SINGLE_TABLESPACE_QUERY
CLOSE_TABLESPACE_QUERY
FETCH_TABLESPACE
OPEN_CONTAINER_QUERY
FETCH_CONTAINER_QUERY
CLOSE_CONTAINER_QUERY
GET_TABLESPACE_STATISTICS
DESCRIBE_DATABASE
ESTIMATE_SNAPSHOT_SIZE
READ_ASYNC_LOG_RECORD
PRUNE_RECOVERY_HISTORY
UPDATE_RECOVERY_HISTORY
QUIESCE_TABLESPACE
UNLOAD_TABLE
UPDATE_DATABASE_VERSION
CREATE_INSTANCE
DELETE_INSTANCE
SET_EVENT_MONITOR
GRANT_DBADM
REVOKE_DBADM
GRANT_DB_AUTHORITIES
REVOKE_DB_AUTHORITIES
REDIST_NODEGROUP

表 12. VALIDATE イベントの監査レコード設計

```
timestamp=1998-06-24-08.42.11.527490;category=VALIDATE;audit event=CHECK_GROUP_MEMBERSHIP;
  event correlator=2;event status=-1092;
  database=FOO;userid=boss;authid=BOSS;execution id=newton;
  application id=*LOCAL.newton.980624124210;application name=testapp;
  auth type=SERVER;
```

名前	フォーマット	記述
Timestamp	CHAR(26)	監査イベントの日付と時刻。
Category	CHAR(8)	監査イベントの区分。可能な値は以下のとおりです。 VALIDATE
Audit Event	VARCHAR(32)	特定の監査イベント。 有効な値は、GET_GROUPS、GET_USERID、AUTHENTICATE_PASSWORD、および VALIDATE_USER です。
Event Correlator	INTEGER	監査している操作のための相関識別子。単一イベントに関連した監査レコードが何かを識別するために使用できます。
Event Status	INTEGER	監査イベントの状況、次のような 1 つの SQLCODE で表されます。 Successful event > = 0 Failed event < 0
Database Name	CHAR(8)	どのイベントが生成されたかを示すデータベース名。インスタンス・レベルの監査イベントだった場合はブランクとなります。
User ID	VARCHAR(1024)	監査イベントの時刻でのユーザー ID。
Authorization ID	VARCHAR(128)	監査イベントの時刻での許可 ID。
Execution ID	VARCHAR(1024)	監査イベントの時刻で使用していた実行 ID。
Origin Node Number	SMALLINT	監査イベントが発生したところのノード番号。
Coordinator Node Number	SMALLINT	調整プログラムのノード番号。
Application ID	VARCHAR (255)	監査イベントが発生した時刻で使用していたアプリケーション ID。
Application Name	VARCHAR (1024)	監査イベントが発生した時刻で使用していたアプリケーション名。
Authentication Type	VARCHAR (32)	監査イベントの時刻での認証タイプ。
Package Schema	VARCHAR (128)	監査イベントの時刻で使用していたパッケージのスキーマ。
Package Name	VARCHAR (128)	監査イベントが発生した時刻で使用していたパッケージ名。

表 12. VALIDATE イベントの監査レコード設計 (続き)

```
timestamp=1998-06-24-08.42.11.527490;category=VALIDATE;audit event=CHECK_GROUP_MEMBERSHIP;
  event correlator=2;event status=-1092;
  database=FOO;userid=boss;authid=BOSS;execution id=newton;
  application id=*LOCAL.newton.980624124210;application name=testapp;
  auth type=SERVER;
```

名前	フォーマット	記述
Package Section Number	SMALLINT	監査イベントが発生した時刻で使用されていたパッケージのセクション番号。

表 13. CONTEXT イベントの監査レコード設計

```
timestamp=1998-06-24-08.42.41.476840;category=CONTEXT;audit event=EXECUTE_IMMEDIATE;
  event correlator=3;
  database=FOO;userid=boss;authid=BOSS;
  application id=*LOCAL.newton.980624124210;application name=testapp;
  package schema=NULLID;package name=SQLC28A1;
  package section=203;text=create table audit(c1 char(10), c2 integer);
```

名前	フォーマット	記述
Timestamp	CHAR(26)	監査イベントの日付と時刻。
Category	CHAR(8)	監査イベントの区分。可能な値は以下のとおりです。 CONTEXT
Audit Event	VARCHAR(32)	特定の監査イベント。 有効な値を、この表に続くリストに示しています。
Event Correlator	INTEGER	監査している操作のための相関識別子。単一イベントに関連した監査レコードが何かを識別するために使用できます。
Database Name	CHAR(8)	どのイベントが生成されたかを示すデータベース名。インスタンス・レベルの監査イベントだった場合はブランクとなります。
User ID	VARCHAR(1024)	監査イベントの時刻でのユーザー ID。
Authorization ID	VARCHAR(128)	監査イベントの時刻での許可 ID。
Origin Node Number	SMALLINT	監査イベントが発生したところのノード番号。
Coordinator Node Number	SMALLINT	調整プログラムのノード番号。
Application ID	VARCHAR (255)	監査イベントが発生した時刻で使用していたアプリケーション ID。
Application Name	VARCHAR (1024)	監査イベントが発生した時刻で使用していたアプリケーション名。
Package Schema	VARCHAR (128)	監査イベントの時刻で使用していたパッケージのスキーマ。

表 13. CONTEXT イベントの監査レコード設計 (続き)

timestamp=1998-06-24-08.42.41.476840;**category=CONTEXT;audit event=EXECUTE_IMMEDIATE;**
event correlator=3;
database=FOO;userid=boss;authid=BOSS;
application id=*LOCAL.newton.980624124210;application name=testapp;
package schema=NULLID;package name=SQLC28A1;
package section=203;**text=create table audit(c1 char(10), c2 integer);**		

名前	**フォーマット**	**記述**
Package Name	VARCHAR (128)	監査イベントが発生した時刻で使用していたパッケージ名。
Package Section Number	SMALLINT	監査イベントが発生した時刻で使用されていたパッケージのセクション番号。
Statement Text (ステートメント)	CLOB(32K)	適用できる場合には、SQL ステートメントのテキストです。 SQL ステートメントのテキストが使用可能でない場合、空白となります。

下記は、有効な CONTEXT 監査イベントのリストです。

表 14. CONTEXT 監査イベント

CONNECT	SET_APPL_PRIORITY
CONNECT_RESET	RESET_DB_CFG
ATTACH	GET_DB_CFG
DETACH	GET_DFLT_CFG
DARI_START	UPDATE_DBM_CFG
DARI_STOP	SET_MONITOR
BACKUP_DB	GET_SNAPSHOT
RESTORE_DB	ESTIMATE_SNAPSHOT_SIZE
ROLLFORWARD_DB	RESET_MONITOR
OPEN_TABLESPACE_QUERY	OPEN_HISTORY_FILE
FETCH_TABLESPACE	CLOSE_HISTORY_FILE
CLOSE_TABLESPACE_QUERY	FETCH_HISTORY_FILE
OPEN_CONTAINER_QUERY	SET_RUNTIME_DEGREE
CLOSE_CONTAINER_QUERY	UPDATE_AUDIT
FETCH_CONTAINER_QUERY	DBM_CFG_OPERATION
SET_TABLESPACE_CONTAINERS	DISCOVER
GET_TABLESPACE_STATISTIC	OPEN_CURSOR
READ_ASYNC_LOG_RECORD	CLOSE_CURSOR
QUIESCE_TABLESPACE	FETCH_CURSOR
LOAD_TABLE	EXECUTE
UNLOAD_TABLE	EXECUTE_IMMEDIATE
UPDATE_RECOVERY_HISTORY	PREPARE
PRUNE_RECOVERY_HISTORY	DESCRIBE
SINGLE_TABLESPACE_QUERY	BIND
LOAD_MSG_FILE	REBIND
UNQUIESCE_TABLESPACE	RUNSTATS
ENABLE_MULTIPAGE	REORG
DESCRIBE_DATABASE	REDISTRIBUTE
DROP_DATABASE	COMMIT
CREATE_DATABASE	ROLLBACK
ADD_NODE	REQUEST_ROLLBACK
FORCE_APPLICATION	IMPLICIT_REBIND

監査機能のヒントと技法

大部分のケースにおいて、CHECKING イベントを指定して作業をしているとき、監査レコードのオブジェクト・タイプのフィールドは、必要とされる特権または権限がオブジェクトにアクセスしようとするユーザー ID によって保持されているかどうかを調べるために、検査されるオブジェクトとなります。たとえば、ユーザーが 1 列を追加することによって 1 つの表へ ALTER しようとする場合、 CHECKING イベントの監査レ

コードは、試みられたアクセスが 『ALTER』 であり、検査されているオブジェクト・タイプが 『TABLE』 であったことを必ず表示します (注: 表特権であるため、検査される必要があるのは列でない)。

ただし、CREATE または BIND へのユーザー ID がオブジェクトであると認める、またはオブジェクトを削除するためのデータベース権限が存在するかどうかを確かめるための検査が必要なとき、データベースに対して検査があるがオブジェクト・タイプのフィールドは (データベース自身よりもむしろ) オブジェクトを作成し、バインドするか、または除去するかを指定します。

表に索引を作成するとき、索引の作成権が必須となります。それで、CHECKING イベントの監査レコードは「作成」よりもむしろ「索引」のタイプのアクセス試行を持ちます。

既に存在するパッケージをバインドしているとき、そのとき OBJMAINT イベントの監査レコードはパッケージの DROP に対して作成されます。そして、それから別の OBJMAINT イベントの監査レコードがパッケージの新しいコピーの CREATE に対して作成されます。

SQL データ定義言語 (DDL) は、正常にログに記録される OBJMAINT または SECMAINT イベントを生成する可能性があります。しかしながら、イベントのログ記録の後に続くことが可能です。後続のエラーが原因で、ROLLBACK を発生するかもしれません。これによって、作成されたのではないオブジェクト、つまり GRANT または REVOKE 処置を残すことになります。 CONTEXT イベントの使用は、この場合に重要となります。こうした CONTEXT イベントの監査レコード、特にイベントを終えるステートメントは、試みられた操作の完了の状態を表示します。

監査レコードを DB2 のリレーショナル表にロードするためにふさわしい区切り文字付き ASCII 形式に抽出しているとき、ステートメントのテキスト・フィールド内で使用される区切り文字に関して明瞭であるべきです。これは区切り文字付き ASCII ファイルを抽出しているときに行われることが可能であり、次のようにしてなされます。

```
db2audit extract delasc delimiter <load delimiter>
```

この *load delimiter* は、単一文字 (たとえば、") かまたは 16 進数で示される 4 バイト・ストリング (たとえば、 『0xff』) となります。有効なコマンドの例は次のとおりです。

```
db2audit extract delasc
db2audit extract delasc delimiter !
db2audit extract delasc delimiter 0xff
```

抽出しているとき、区切り文字としてデフォルトのロード区切り文字 (『"』) 以外の何かを使用した場合、 LOAD コマンドでは MODIFIED BY オプションを使ってください。区切り文字として使用される 『0xff』 を指定した LOAD コマンド例の一部分を次に示します。

```
db2 load from context.del of del modified by chardel0xff replace into ...
```

これにより、デフォルトのロード文字ストリング区切り文字、 『0xff』 が指定変更されます。

DB2 監査機能アクティビティーの制御

監査機能アクティビティーの制御の説明の一つとして、次のような簡単なシナリオを用いてみましょう。あるユーザー、*newton* は、表を接続しかつ作成する *testapp* と呼ばれるアプリケーションを実行します。この同じアプリケーションは、下記に説明されているそれぞれの例の中で使用されるものです。

次のような極端な例を提示することから始めます。それは、すべての成功するかつ不成功の監査イベントを監査することを決定した場合で、次のような方法で監査機能を構成します。

```
db2audit configure scope all status both
```

注: これは、すべての考えられる監査可能なイベントに対して監査レコードを作成します。結果として、多くの監査レコードが監査ログに書き込まれ、それはデータベース・マネージャーのパフォーマンスを減少させます。この極端なケースは、デモンストレーション目的のためだけにここで示されていて、上記に示すコマンドを指定して監査機能を構成することは勧められていません。

この構成 (『db2audit start』 を使用) を指定した監査機能を開始して、それから *testapp* アプリケーションを実行した後で、次のレコードが監査ログに生成され配置されます。このログから監査レコードを抽出することによって、下記のレコードがアプリケーションで実行される 2 つの処置のために生成されるのを確認するでしょう。

アクション
作成されるレコードのタイプ

CONNECT

```
timestamp=1998-06-24-08.42.10.555345;category=CONTEXT;
audit event=CONNECT;event correlator=2;database=FOO;
application id=*LOCAL.newton.980624124210;
application name=testapp;

timestamp=1998-06-24-08.42.10.944374;category=VALIDATE;
audit event=AUTHENTICATION;event correlator=2;event status=0;
database=FOO;userid=boss;authid=BOSS;execution id=newton;
application id=*LOCAL.newton.980624124210;application name=testapp;
auth type=SERVER;

timestamp=1998-06-24-08.42.11.527490;category=VALIDATE;
audit event=CHECK_GROUP_MEMBERSHIP;event correlator=2;
event status=-1092;database=FOO;userid=boss;authid=BOSS;
execution id=newton;application id=*LOCAL.newton.980624124210;
application name=testapp;auth type=SERVER;
```

```
timestamp=1998-06-24-08.42.11.561187;category=VALIDATE;
audit event=CHECK_GROUP_MEMBERSHIP;event correlator=2;
event status=-1092;database=FOO;userid=boss;authid=BOSS;
execution id=newton;application id=*LOCAL.newton.980624124210;
application name=testapp;auth type=SERVER;

timestamp=1998-06-24-08.42.11.594620;category=VALIDATE;
audit event=CHECK_GROUP_MEMBERSHIP;event correlator=2;
event status=-1092;database=FOO;userid=boss;authid=BOSS;
execution id=newton;application id=*LOCAL.newton.980624124210;
application name=testapp;auth type=SERVER;

timestamp=1998-06-24-08.42.11.622984;category=CHECKING;
audit event=CHECKING_OBJECT;event correlator=2;event status=0;
database=FOO;userid=boss;authid=BOSS;
application id=*LOCAL.newton.980624124210;application name=testapp;
object name=FOO;object type=DATABASE;access approval reason=DATABASE;
access attempted=CONNECT;

timestamp=1998-06-24-08.42.11.801554;category=CONTEXT;
audit event=COMMIT;event correlator=2;database=FOO;userid=boss;
authid=BOSS;application id=*LOCAL.newton.980624124210;
application name=testapp;

timestamp=1998-06-24-08.42.41.450975;category=CHECKING;
audit event=CHECKING_OBJECT;event correlator=2;event status=0;
database=FOO;userid=boss;authid=BOSS;
application id=*LOCAL.newton.980624124210;application name=testapp;
package schema=NULLID;package name=SQLC28A1;object schema=NULLID;
object name=SQLC28A1;object type=PACKAGE;
access approval reason=OBJECT;access attempted=EXECUTE;
```

CREATE TABLE

```
timestamp=1998-06-24-08.42.41.476840;category=CONTEXT;
audit event=EXECUTE_IMMEDIATE;event correlator=3;database=FOO;
userid=boss;authid=BOSS;application id=*LOCAL.newton.980624124210;
application name=testapp;package schema=NULLID;package name=SQLC28A1;
package section=203;text=create table audit(c1 char(10), c2 integer);

timestamp=1998-06-24-08.42.41.539692;category=CHECKING;
audit event=CHECKING_OBJECT;event correlator=3;event status=0;
database=FOO;userid=boss;authid=BOSS;
application id=*LOCAL.newton.980624124210;application name=testapp;
package schema=NULLID;package name=SQLC28A1;package section=0;
object schema=BOSS;object name=AUDIT;object type=TABLE;
access approval reason=DATABASE;access attempted=CREATE;

timestamp=1998-06-24-08.42.41.570876;category=CHECKING;
audit event=CHECKING_OBJECT;event correlator=3;event status=0;
database=FOO;userid=boss;authid=BOSS;
application id=*LOCAL.newton.980624124210;application name=testapp;
package schema=NULLID;package name=SQLC28A1;package section=0;
object name=BOSS;object type=SCHEMA;access approval reason=DATABASE;
access attempted=CREATE;
```

```
timestamp=1998-06-24-08.42.41.957524;category=OBJMAINT;
audit event=CREATE_OBJECT;event correlator=3;event status=0;
database=FOO;userid=boss;authid=BOSS;
application id=*LOCAL.newton.980624124210;application name=testapp;
package schema=NULLID;package name=SQLC28A1;package section=0;
object schema=BOSS;object name=AUDIT;object type=TABLE;

timestamp=1998-06-24-08.42.42.018900;category=CONTEXT;audit event=COMMIT;
event correlator=3;database=FOO;userid=boss;authid=BOSS;
application id=*LOCAL.newton.980624124210;application name=testapp;
package schema=NULLID;package name=SQLC28A1;
```

ご覧になってわかるように、すべての考えられる監査イベントおよびタイプの監査を要求する監査構成から生成される、有効な数の監査レコードがあります。

大部分のケースにおいて、監査することを望むイベントがより多くの制限付き、または焦点が合った視点での監査機能を形成します。たとえば、失敗するイベントの監査だけを望む場合もあります。この場合、監査機能は次のように構成されます。

```
db2audit configure scope audit,checking,objmaint,secmaint,sysadmin,
   validate status failure
```

注: この構成は、初期の監査構成かまたは監査構成がリセットされた時点に生じるものです。

この構成を指定した監査機能を開始して、それから *testapp* アプリケーションを実行した後で、次のレコードが監査ログに生成され配置されます。 (この場合、*testapp* がそれ以前に実行されていないことを想定しています。) このログから監査レコードを抽出することによって、下記のレコードがアプリケーションで実行される 2 つの処置のために生成されるのを確認するでしょう。

アクション
作成されるレコードのタイプ

CONNECT

```
timestamp=1998-06-24-08.42.11.527490;category=VALIDATE;
audit event=CHECK_GROUP_MEMBERSHIP;event correlator=2;
event status=-1092;database=FOO;userid=boss;authid=BOSS;
execution id=newton;application id=*LOCAL.newton.980624124210;
application name=testapp;auth type=SERVER;

timestamp=1998-06-24-08.42.11.561187;category=VALIDATE;
audit event=CHECK_GROUP_MEMBERSHIP;event correlator=2;
event status=-1092;database=FOO;userid=boss;authid=BOSS;
execution id=newton;application id=*LOCAL.newton.980624124210;
application name=testapp;auth type=SERVER;

timestamp=1998-06-24-08.42.11.594620;category=VALIDATE;
audit event=CHECK_GROUP_MEMBERSHIP;event correlator=2;
```

```
event status=-1092;database=FOO;userid=boss;authid=BOSS;
execution id=newton;application id=*LOCAL.newton.980624124210;
application name=testapp;auth type=SERVER;
```

CREATE TABLE

(なし)

すべての考えられる監査イベント (CONTEXT を除く) の監査を要求する監査構成から生成される監査レコードは、監査試行が失敗するとき以外だけではかなり少なくなります。監査構成を変更することによって、生成される監査レコードのタイプおよび性質を制御できます。

この監査機能を使用することにより、監査することを望むものがオブジェクト上で特権を首尾よく付与されたとき、監査レコードを作成することを可能にします。この場合において、次のような監査機能を構成することができます。

```
db2audit configure scope checking status success
```

この構成を指定した監査機能を開始して、それから *testapp* アプリケーションを実行した後で、次のレコードが監査ログに生成され配置されます。 (この場合、*testapp* がそれ以前に実行されていないことを想定しています。) このログから監査レコードを抽出することによって、下記のレコードがアプリケーションで実行される 2 つの処置のために生成されるのを確認するでしょう。

アクション

作成されるレコードのタイプ

CONNECT

```
timestamp=1998-06-24-08.42.11.622984;category=CHECKING;
audit event=CHECKING_OBJECT;event correlator=2;event status=0;
database=FOO;userid=boss;authid=BOSS;

timestamp=1998-06-24-08.42.41.450975;category=CHECKING;
audit event=CHECKING_OBJECT;event correlator=2;event status=0;
database=FOO;userid=boss;authid=BOSS;
application id=*LOCAL.newton.980624124210;application name=testapp;
package schema=NULLID;package name=SQLC28A1;object schema=NULLID;
object name=SQLC28A1;object type=PACKAGE;
access approval reason=OBJECT;access attempted=EXECUTE;

timestamp=1998-06-24-08.42.41.539692;category=CHECKING;
audit event=CHECKING_OBJECT;event correlator=3;event status=0;
database=FOO;userid=boss;authid=BOSS;
application id=*LOCAL.newton.980624124210;application name=testapp;
package schema=NULLID;package name=SQLC28A1;package section=0;
object schema=BOSS;object name=AUDIT;object type=TABLE;
access approval reason=DATABASE;access attempted=CREATE;

timestamp=1998-06-24-08.42.41.570876;category=CHECKING;
audit event=CHECKING_OBJECT;event correlator=3;event status=0;
database=FOO;userid=boss;authid=BOSS;
```

```
application id=*LOCAL.newton.980624124210;application name=testapp;
package schema=NULLID;package name=SQLC28A1;package section=0;
object name=BOSS;object type=SCHEMA;access approval reason=DATABASE;
access attempted=CREATE;
```

CREATE TABLE

(なし)

第4部 データの移動

第7章 データの移動に使用するユーティリティー

LOAD ユーティリティーは、表にデータを移動し、既存の索引を拡張し、統計を生成します。 LOAD は、データ量が多いときは、IMPORT ユーティリティーより高速にデータを移動します。 EXPORT ユーティリティーを使ってアンロードされたデータも、LOAD ユーティリティーでロードできます。

AutoLoader ユーティリティーは大量のデータを分割し、分割されたデータを区分データベースの異なった区分にロードします。

IMPORT および EXPORT ユーティリティーは、DB2 コネクトを使用して、表または視点と別のデータベースまたは表計算プログラムとの間、DB2 データベース間、および DB2 データベースとホスト・データベースとの間でデータを移動します。

データ・レプリケーション (以前は DataPropagator Relational (DPROPR)) は DB2 ユニバーサル・データベースのコンポーネントで、他の DB2 リレーショナル・データベースの他の表に表の更新項目を自動的にコピーするものです。

注: *コマンド解説書* および *管理 API 解説書* に掲載されているトピックおよび同様のトピックはすべて、 *データ移動ユーティリティー手引きおよび解説書* に統合されました。

このトピックについては、*データ移動ユーティリティー手引きおよび解説書* が主要かつ唯一の情報源になります。

レプリケーションの詳細については、*レプリケーションの手引きおよび解説書* を参照してください。

第5部 回復

第8章 データベースのリカバリー

問題が発生した場合には、データベースをリカバリーできなければなりません。問題には、電源障害、アプリケーションの障害、メディアやストレージの障害が関係しています。このような問題が発生したときにリカバリーできるようにするには、データベース全体またはそのデータベースを構成する表スペースのバックアップまたはコピーをとっておく必要があります。問題が発生した後に、これらのバックアップを使用して、データベースをリカバリーします。

問題発生後にデータベースを再構築することをリカバリーと言います。破損リカバリーは、自動的に障害後のデータベース・リカバリーを試行します。破損リカバリーにより、データベースは不整合な状態や使用不能の状態のままにならないように保護されます。データベースの障害が発生すると、データベースに対するトランザクションが不完全な状態になることがあります。破損リカバリーでは、不完全なトランザクションをロールバックするか、完了したトランザクションをコミットします。こうしたアクションによって、データベースの整合性が保たれ、使用可能になります。

データベースが損傷を受けて、データベースの内容が不確実な場合に、その他の 2 つのタイプのリカバリーがあります。損傷を受けたデータベースをリカバリーする方法は、バージョン・リカバリーとロールフォワード・リカバリーです。読み取り専用データベースで作業する場合、またはデータベースに記録されているトランザクションに関心がない場合には、バージョン・リカバリーをするだけで十分です。データベースのバックアップをとってある場合には、データベースのコピーを適用または復元できます。これをバージョン・リカバリーと言います。トランザクションを適用したデータベースで作業するとき、データベースにすべての変更が適用されていることを知る必要がある場合には、ロールフォワード・リカバリーを完了する必要があります。ロールフォワード・リカバリーには、データベースのバックアップの復元が関係しています。次に、トランザクションを記録しているログのレコードをデータベースに対して適用します。ログのアプリケーションはデータベースに対するすべてのアクティビティーを繰り返して、データベースが障害発生時の前の状態になるようにします。このリカバリー方式を使用すると、データベースに対する変更は失われません。ロギングはこのリカバリー方式のかぎです。

注: コマンド*解説書* および*管理 API 解説書* に掲載されているトピックおよび同様のトピックはすべて、 *データ回復と高可用性の手引きと解説書* に統合されました。

このトピックについては、*データ回復と高可用性の手引きと解説書* が主要かつ唯一の情報源になります。

第6部 付録

付録A. 命名規則

以下の中から、情報が必要な命名規則について説明する節に進んでください。

- 『一般的な命名規則』
- 『オブジェクトの命名規則』
- 334ページの『連合システムで大文字小文字を区別する値を保持する方法』

一般的な命名規則

特に指定されていなければ、すべての名前には以下の文字を含めることができます。

- A ～ Z。ほとんどの名前では、文字 A ～ Z は小文字から大文字に変換されます。
- 0 ～ 9
- @、#、$、および _ (下線)

名前は数値または下線文字で始めることはできません。

SQL 予約語を使用して表、視点、列、索引、または許可 ID を命名しないでください。SQL 予約語のリストについては、*SQL 解説書* を参照してください。

ご使用のオペレーティング・システムおよび DB2 での作業を行う場所に応じて、別個の処理を行うその他の特殊文字があります。ただし、それらは処理を行うかもしれませんが、必ず行うという保証はありません。データベース内のオブジェクトに名前を付ける際に、そのような特殊文字を使用することはお勧めできません。

オブジェクトの命名規則

すべてのオブジェクトは一般的な命名規則に従います。さらに、オブジェクトによっては、以下に示す追加の制約事項があります。

表 15. データベース、データベース別名、およびインスタンスの命名規則

オブジェクト	指針
• データベース • データベース別名 • インスタンス	• データベース名は、カタログされている場所で固有である必要があります。DB2 を UNIX ベースで実装している場合はこの場所はディレクトリー・パスですが、 Windows で実装している場合はドライブ名です。 • データベース別名は、システム・データベース・ディレクトリー内で固有である必要があります。新規のデータベースが作成されると、別名はデータベース名 (デフォルト) になります。結果として、データベース別名と同じ名前のデータベースがない場合でも、その名前を使用してデータベースを作成することはできません。 • データベース、データベース別名、およびインスタンス名は最高 8 バイトです。 • Windows NT および Windows 2000 システムでは、インスタンスにサービス名と同じ名前を付けることはできません。 **注:** 潜在的な問題を避けるには、データベースを通信環境で使用したい場合に、データベース名に特殊文字 @、#、および $ を使用しないでください。また、これらの文字はすべてのキーワードに共通ではないので、別の言語でそのデータベースを使用する計画がある場合は、これらの文字を使用しないでください。

表 16. データベース・オブジェクトの命名規則

オブジェクト	指針
• 別名 • バッファー・プール • 列 • イベント・モニター • 索引 • メソッド • ノードグループ • スキーマ • ストアード・プロシージャー • 表 • 表スペース • トリガー • UDF • UDT • 視点	最高 18 バイトまで含めることができます。ただし、以下の場合を除きます。 • テーブル名 (視点名、要約表名、別名、および相関名を含む)。最高 128 バイトまで含めることができます。 • 列名。最高 30 バイトまで含めることができます。 • スキーマ名、最高 30 バイトまで含めることができます。 • オブジェクト名には以下のものを含めることもできます。 – 有効なアクセント付き文字 (ö など) – マルチバイト・スペース以外のマルチバイト文字 (マルチバイト環境)

スキーマ名に関する追加情報

- 18 バイトより長いスキーマ名のある表は複製できません。
- ユーザーの定義したタイプ (UDT) には 8 バイトより長いスキーマ名を使用できません。
- 次のスキーマ名は予約語なので、使用してはなりません。
  SYSCAT、SYSFUN、SYSIBM、SYSSTAT。
- 今後の潜在的な移行の問題を避けるために、SYS で始まるスキーマ名を使用しないでください。データベース・マネージャーでは、SYS で始まるスキーマ名を使用して、トリガー、ユーザー定義タイプ、またはユーザー定義関数を作成することはできません。
- スキーマ名として SESSION を使用しないようにもお勧めします。 宣言済み一時表は、 SESSION により修飾されなければなりません。したがって、アプリケーションで宣言された一時表が持続表と同じ名前になり、アプリケーション論理が非常に複雑になる可能性があります。宣言済み一時表を処理する場合を除いては、スキーマ SESSION の使用は避けてください。

表 17. ユーザー、ユーザー ID、およびグループの命名規則

オブジェクト	指針
• グループ名 • ユーザー名 • ユーザー ID	• グループ名には最高 8 バイトを含めることができます。 • UNIX ベース・システムでのユーザー ID には最高 8 文字を含めることができます。 • Windows でのユーザー名には最高 30 文字を含めることができます。現在、Windows NT および Windows 2000 には 20 文字の実際的な制限があります。 • DCE 認証を使用する場合、ユーザー名には 8 文字の制限があります。 • DCE またはクライアント認証を使用しない場合、ユーザー名とパスワードが明示的に指定されていると、 8 文字を超えるユーザー名を持つ、Windows NT および Windows 2000 に接続している Windows 32 ビット以外のクライアントがサポートされます。 • 名前および ID は次のようにすることはできません。 – USERS、ADMINS、GUESTS、PUBLIC、LOCAL、または *SQL 解説書* にリストされている SQL 予約語。 – IBM、SQL、または SYS。 – アクセント付き文字を含める。 UNIX ベースのシステムでは、グループとユーザーの名前を同じにすることができます。 GRANT ステートメントの場合、グループまたはユーザーのいずれを参照するかを指定する必要があります。 REVOKE ステートメントの場合、許可カタログ表にさまざまな GRANTEETYPE 値の GRANTEE 行が複数あるかどうかに基づいて、ユーザーまたはグループを指定します。 Windows NT では、ローカル・グループ、グローバル・グループおよびユーザーの名前を同じにすることはできません。 OS/2 では、グループとユーザーの名前を同じにすることはできません。 **注:** 1. オペレーティング・システムによっては、大文字小文字の区別のあるユーザー ID およびパスワードを使用できます。ご使用のオペレーティング・システムではその区別があるかどうか、資料を見て調べてください。 2. CONNECT または ATTACH が成功した場合の許可 ID は、8 文字に切り捨てられます。省略符号 (...) が許可 ID に付加され、 SQLWARN フィールドに切り捨てを指示する警告が含まれます。詳細については、 *SQL 解説書* の CONNECT ステートメントを参照してください。

パスワードに関する追加情報

パスワードの保守作業を行うために必要となることがあります。このような作業がサーバーで必要となり、かつ多数のユーザーがそのサーバー環境で効率よくまたは快適に働けないため、この作業を実行するのは重大な障害となる可能性があります。 DB2 UDB には、サーバー上にいなくても、パスワードを更新し、検査するための手段が備わっています。たとえば、DB2 (OS/390 版) のバージョン 5 では、ユーザーのパスワードを変更するこの方式をサポートしています。エラー・メッセージ SQL1404N『パスワード失効』が出された場合には、下記のように CONNECT ステートメントを使用してパスワードを変更します。

```
CONNECT TO <database> USER <userid> USING <password>
   NEW <new_password> CONFIRM <new_password>
```

DB2 クライアント構成アシスタントの「パスワード変更 (Password change)」ダイアログもまた、パスワードを変更するのに使用される場合があります。 パスワード変更の方法についての詳細は、 *SQL 解説書*、および CCA オンライン・ヘルプを参照してください。

表 18. 連合データベース・オブジェクトの命名規則

オブジェクト	指針
• 機能マッピング • 索引仕様 • ニックネーム • サーバー • タイプ・マッピング • ユーザー・マッピング • ラッパー	• ニックネーム、マッピング、索引仕様、サーバー名、およびラッパー名は、128 バイトを超過することはできません。 • サーバー・オプション、ニックネーム・オプションおよびオプション設定は、255 バイトに制限されています。 • 連合データベース・オブジェクトの名前には以下のものを含めることができます。 – 有効なアクセント付き文字 (ö など) – マルチバイト・スペース以外のマルチバイト文字 (マルチバイト環境)

オブジェクト名での区切り ID の使用

キーワードを使用することができます。キーワードが SQL キーワードとして解釈される可能性もある文脈で使用される場合、区切り識別名として指定する必要があります。

区切り識別子を使用する際に、上記の命名規則に違反するオブジェクトを作成することもできます。しかしながら、そのオブジェクトを引き続き使用すると、エラーになることがあります。たとえば、名前に + または − (マイナス) 記号が含まれている列を作成した後に、その列を索引で使用すると、表を再編成しようとするときに問題が起きます。区切り ID の詳細については、*SQL 解説書* の『SQL ID』の節を参照してください。

連合システムで大文字小文字を区別する値を保持する方法

分散要求では、ID とパスワードを指定しなければならないことがあります。 ID とパスワードは、データ・ソースでは大文字小文字が区別されます。データ・ソースに渡されたときに大文字小文字が正しいことを確認するには、以下の指針に従ってください。

- ID とパスワードを要求されている文字で指定し、二重引用符で囲みます。
- ユーザー ID を指定しているのであれば、データ・ソースの *fold_id* サーバー・オプションを ″n″ (『大文字小文字を変更せず』) にセットします。パスワードを指定しているのであれば、データ・ソースの *fold_pw* サーバー・オプションを ″n″ にセットします。

  ユーザー ID およびパスワード指定のための別の方法があります。データ・ソースで小文字のユーザー ID を必要とする場合、任意の文字で指定してから *fold_id* サーバー・オプションを ″l″ (『この ID を小文字でデータ・ソースへ送る』) にセットできます。データ・ソースで大文字のユーザー ID を必要とする場合、任意の文字で指定してから *fold_id* を ″u″ (『この ID を大文字でデータ・ソースへ送る』) にセットできます。同様に、データ・ソースで小文字あるいは大文字のパスワードを必要としたとしても、 *fold_pw* サーバー・オプションを ″l″ か ″u″ にセットすれば、この要件を満たすことができます。

  サーバー・オプションについての詳細は、 160ページの『データ・ソースの定義に役立ち、認証処理を容易にするサーバー・オプションの使用』を参照してください。
- オペレーティング・システムのコマンド・プロンプトで、大文字小文字を区別する ID またはパスワードを二重引用符で囲む場合、システムがその二重引用符を正しく解析することを確認しなければなりません。そのためには、以下のようにします。
  - UNIX ベースのオペレーティング・システムでは、ステートメントを単一引用符で囲みます。
  - Windows NT オペレーティング・システムでは、各引用符の前に円記号を付けます。

  たとえば、DB2 のデータ・ソースにある多くの区切り識別子は、大文字小文字を区別します。 `NORBASE` というデータ・ソースに存在する DB2 for CS 視点 `"my_schema"."wkly_sal"` のために `NICK1` というニックネームを作成するとします。

  UNIX ベースのシステムのコマンド・プロンプトでは、次のように入力します。

  ```
  db2 'create nickname nick1 for norbase."my_schema"."wkly_sal"'
  ```

  Windows NT コマンド・プロンプトでは、次のように入力します。

  ```
  db2 create nickname nick1 for norbase.¥"my_schema¥".¥"wkly_sal¥"
  ```

  DB2 コマンド・プロンプト (対話機能モード) からステートメントを入力したり、アプリケーション・プログラムでステートメントを指定する場合、単一引用符や円記号は必要ありません。たとえば、UNIX ベースのシステムまたは Windows NT システムのいずれかの DB2 コマンド・プロンプトで、次のように入力します。

```
create nickname nick1 for norbase."my_schema"."wkly_sal"
```

付録B. 分散コンピューティング環境 (DCE) ディレクトリー・サービスの使用

DCE には、セル・ディレクトリー・サービス (CDS) とグローバル・ディレクトリー・サービス (GDS) があります。 DCE の概念とそのサービスに関する詳細ついては、*OSF DCE 入門* を参照してください。 DCE ディレクトリー・サービス用の DB2 関数は、CDS だけをサポートします。このサポートを使用すると、ユーザーは各データベース、ノード、および DCS データベースを単一のクライアントごとに作成する必要がなくなります。これらの情報については、DCE CDS にまとめられます。

以下の部分では、DCE ディレクトリー・サービスを使用して、データベースを設定しアクセスする方法について説明しています。

- ディレクトリー・オブジェクトの作成
- 各オブジェクト・クラスの属性
- ディレクトリー・サービス機密保護
- 構成パラメーターとレジストリー変数
- CATALOG コマンド、ATTACH コマンド、および CONNECT ステートメント
- クライアントがデータベースに接続する方法
- ディレクトリー探索の方法
- DCE ディレクトリー情報の一時的な指定変更
- ディレクトリー・サービスの作業
- ディレクトリー・サービスの制約事項

DCE ディレクトリー・サービスは、すべての DB2 クライアントにサポートされているとはかぎりません。 DCE ディレクトリー・サービスが DB2 クライアントにサポートされている場合は、 *概説およびインストール* にある追加情報をご覧ください。

ディレクトリー・オブジェクトの作成

データベース管理者が作成する必要のあるディレクトリー・オブジェクトには、次の 3 つの種類があります。

- 338ページの『データベース・オブジェクト』
- 339ページの『データベース・ロケーター・オブジェクト』
- 341ページの『経路指定情報オブジェクト』

各オブジェクトには、さまざまな属性が含まれています。属性の詳しい説明は、342ページの『各オブジェクト・クラスの属性』を参照してください。

データベース管理者がオブジェクトを作成する前に、 DCE 管理者がデータベース情報を CDS 表に追加し、データベース管理者に作成特権を付与しておかなければなりません。詳細については、359ページの『DCE 管理者の作業』を参照してください。

データベース・オブジェクト

データベース・オブジェクトは、ターゲット・データベースごとに必要です。オブジェクト名は、セル名にディレクトリー名とデータベース名が連結した形式で、たとえば、次のようになります。

```
/.../cell_name/dir_name1/dir_name2/OBJ_NAME
```

注: データベース名に関しては、以下のことをお勧めします。データベース名は 8 文字以下にし、大文字を使用します。データベース名に大文字と小文字が混ざっていたり、8 文字以上であったりすると、 CATALOG GLOBAL DATABASE コマンドを使用して別名を割り当てることが必要になります。このコマンドの詳細は、351ページの『CATALOG GLOBAL DATABASE コマンド』を参照してください。

以下に、データベース・オブジェクトの例を示します。 DCE ディレクトリーに保管されるオブジェクトには、タイム・スタンプのような他の情報があります。各属性の左にある文字は、属性が必須 (R)、任意選択 (O)、または注釈 (C) のいずれであるかを示しています。

```
   Object name:               /.../CELL_TORONTO/subsys/database/AIXDB1
R  DB_Object_Type:            D
C  DB_Product_Name:           DB2_for_AIX
C  DB_Product_Release:        V7R1M000
R  DB_Native_Database_Name:   AIXDBASE
R  DB_Database_Protocol:      DB2RA
R  DB_Authentication:         CLIENT
O  DB_Communication_Protocol:
O  DB_Database_Locator_Name: /.../CELL_TORONTO/subsys/database/AIX_INST
C  DB_Comment:                Test_database_on_AIX
```

データベースが、データベース・マネージャーのインスタンスに関連した多数のデータベースのうちの 1 つである場合、データベース・オブジェクトにはデータベース・ロケーター・オブジェクトの名前がある必要があり、通信プロトコルはブランクである必要があります。データベース・ロケーター・オブジェクトの名前は、データベース・マネージャーまたは DB2 コネクトのインスタンスの完全修飾名です。

次に示すのは、オブジェクトを作成する DCE コマンドの例です。オブジェクトを作成する前に、 DCE 管理者は 359ページの『DCE 管理者の作業』に説明されているステップを行う必要があります。

まず、*cdscp.inp* という名前のファイルに次のようにタイプします。

```
create object /.:/subsys/database/AIXDB1

add object /.:/subsys/database/AIXDB1 DB_Object_Type            = D
```

```
add object /.:/subsys/database/AIXDB1 DB_Product_Name           = DB2_for_AIX
add object /.:/subsys/database/AIXDB1 DB_Product_Release        = V7R1M000
add object /.:/subsys/database/AIXDB1 DB_Native_Database_Name   = AIXDBASE
add object /.:/subsys/database/AIXDB1 DB_Database_Protocol      = DB2RA
add object /.:/subsys/database/AIXDB1 DB_Authentication         = CLIENT
add object /.:/subsys/database/AIXDB1 DB_Database_Locator_Name  = /...
/CELL_TORONTO/subsys/database/AIX_INST
add object /.:/subsys/database/AIXDB1 DB_Comment      = Test_database_on_AIX
```

次に、以下のいずれかを行います。

- プリンシパル・パスワードで dcelogin を行う (OS/2 の場合)
- プリンシパル・パスワードで dce_login を行う (UNIX、 Windows オペレーティング・システム)

この後、次を行ってください。

- cdscp < cdscp.inp

オブジェクトを表示するには、次のコマンドを使用します。

```
cdscp show object /.:/subsys/database/AIXDB1
```

データベースが、データベース・マネージャーのインスタンスに関連したただ 1 つのデータベースである場合、データベース・オブジェクトには通信プロトコル属性の値を入れ、データベース・ロケーター・オブジェクトの名前はブランクである必要があります。たとえば、次のようにします。

```
   Object name:                 /.../CELL_TORONTO/subsys/database/MVSDB
R  DB_Object type:              D
C  DB_Product_Name:             DB2_for_MVS
C  DB_Product_Release:          V7R1M00
R  DB_Native_Database_Name:     MVSDBASE
R  DB_Database_Protocol:        DRDA
R  DB_Authentication:           SERVER
O  DB_Communication_Protocol:   APPC;NET1;TARGETLU1;DB2DRDA;MODE1;PROGRAM
O  DB_Database_Locator_Name:
C  DB_Comment:                  Test_database_on_MVS
```

データベース・ロケーター・オブジェクト

これらのオブジェクトには、データベース管理システム・インスタンスまたは DB2 コネクト・インスタンスが使用するすべての通信プロトコルの詳細が入っています。以下の各インスタンスには、データベース・ロケーター・オブジェクトが 1 つ必要です。

- データベース管理システムと DB2 コネクトの両方が指定されているインスタンス
- 複数のデータベースに関連付けられているが、 DB2 コネクトには関連付けられていないデータベース管理システム・インスタンス
- データベース管理システムには関連付けられていない DB2 コネクト・インスタンス

オブジェクト名は、セル名にディレクトリー名とデータベース・インスタンスの 1 パート名が連結した形式で、たとえば、次のようになります。

```
/.../cell_name/dir_name1/dir_name2/AIX_INST
```

注: インスタンスを ATTACH のターゲットとして使用する場合には、 1 パート名を 8 文字以下かつすべて大文字にしなければなりません。

以下に、データベース・ロケーターの例を示します。 DCE ディレクトリーに保管されるオブジェクトには、タイム・スタンプのような他の情報があります。各属性の左にある文字は、属性が必須 (R)、任意選択 (O)、または注釈 (C) のいずれであるかを示しています。

```
   Object name:                /.../CELL_TORONTO/subsys/database/AIX_INST
R  DB_Object_Type:             L
C  DB_Product_Name:            DB2_for_AIX
C  DB_Product_Release:         V7R1M00
R  DB_Communication_Protocol: TCPIP;HOSTNAME1;1234
R  DB_Communication_Protocol: APPC;NET1;TARGETLU1;TPN1;MODE;PROGRAM
C  DB_Comment:                 Test_instance_on_AIX
```

属性がデータベース・オブジェクトとデータベース・ロケーター・オブジェクトの両方で定義されるときには、データベース・オブジェクトの値が使用されます。

次に示すのは、オブジェクトを作成する DCE コマンドの例です。オブジェクトを作成する前に、 DCE 管理者は 359ページの『DCE 管理者の作業』に説明されているステップを行う必要があります。

まず、*cdscp.inp* という名前のファイルに次のようにタイプします。

```
create object /.:/subsys/database/AIX_INST

add object /.:/subsys/database/AIX_INST DB_Object_Type            = L
add object /.:/subsys/database/AIX_INST DB_Product_Name           = DB2_for_AIX
add object /.:/subsys/database/AIX_INST DB_Product_Release        = V7R1M00
add object /.:/subsys/database/AIX_INST DB_Communication_Protocol = TCPIP;
HOSTNAME1;1234
add object /.:/subsys/database/AIX_INST DB_Communication_Protocol = APPC;NET1;
TARGETLU;TPN1;MODE;PROGRAM
add object /.:/subsys/database/AIX_INST DB_Comment     = Test_instance_on_AIX
```

次に、以下のいずれかを行います。

- プリンシパル・パスワードで dcelogin を行う (OS/2 の場合)
- プリンシパル・パスワードで dce_login を行う (UNIX、 Windows オペレーティング・システム)

この後、次を行ってください。

- cdscp < cdscp.inp

オブジェクトを表示するには、次のコマンドを使用します。

```
cdscp show object /.:/subsys/database/AIX_INST
```

経路指定情報オブジェクト

ホスト・アクセスが必要な場合には、経路指定情報オブジェクトを作成する必要があります。クライアントで使用されるデータベース・プロトコルとターゲット・データベースで使用されるデータベース・プロトコルに不一致がある場合、経路指定オブジェクトは、どの DB2 コネクト・インスタンスが使用されるかをクライアントに知らせます。属性はターゲット・データベースごとにありますが、それには使用できるデータベース・プロトコルと、 DB2 コネクト・インスタンスのデータベース・ロケーター・オブジェクトの名前が含まれています。オブジェクト名は、セル名にディレクトリー名と固有 1 パート名が連結した形式で、たとえば、次のようになります。

```
/.../cell_name/dir_name1/dir_name2/ROUTE1
```

以下に、経路指定情報オブジェクトの例を示します。 DCE ディレクトリーに保管されるオブジェクトには、タイム・スタンプのような他の情報があります。各属性の左にある文字は、属性、および属性内の各トークンが必須 (R)、任意選択 (O)、または注釈 (C) のいずれであるかを示しています。

353ページの図8 では、クライアント・グループ 1 は、 Client_1、Client_2、および Client_3 です。

```
   Object name:    /.../CELL_TORONTO/subsys/database/ROUTE1
R  DB_Object_Type: R
C  DB_Comment:     Routing_for_client_group_1

R  DB_Target_Database_Info
   R  Database name                 = /.../CELL_TORONTO/subsys/database/MVSDB
   R  Outbound protocol from router = DRDA
   R  Inbound protocol to router    = DB2RA
   R  Authenticate at gateway       = 1
   O  Parameter string              = NOMAP,D,INTERRUPT_ENABLED
   R  DB_Database_Locator_Name      = /.../CELL_TORONTO/subsys/database/GW_INST

R  DB_Target_Database_Info
   R  Database name                 = *OTHERDBS
   R  Outbound protocol from router = DRDA
   R  Inbound protocol to router    = DB2RA
   R  Authenticate at gateway       = 0
   O  Parameter string              =
   R  DB_Database_Locator_Name      = /.../CELL_TORONTO/subsys/database/OTH_INST
```

データベース名 *OTHERDBS は、経路指定情報オブジェクトに明示的に定義されていないデータベースにアクセスする際に使用される共通ルーターを表す特殊値です。

次に示すのは、オブジェクトを作成する DCE コマンドの例です。円記号 (¥) は連結文字です。

オブジェクトを作成する前に、 DCE 管理者は 359ページの『DCE 管理者の作業』に説明されているステップを行う必要があります。

まず、*cdscp.inp* という名前のファイルに次のようにタイプします。

```
create object /.:/subsys/database/ROUTE1

add object /.:/subsys/database/ROUTE1 DB_Object_Type = R
add object /.:/subsys/database/ROUTE1 DB_Comment = Routing_for_client_group_1
add object /.:/subsys/database/ROUTE1 DB_Target_Database_Info = ¥
/.../CELL_TORONTO/subsys/database/MVSDB;¥
drda;db2ra;1;NOMAP,D,INTERRUPT_ENABLE;¥
/.../CELL_TORONTO/subsys/database/GW_INST
add object /.:/subsys/database/ROUTE1 DB_Target_Database_Info = ¥
*OTHERDBS;drda;db2ra;0;;¥
/.../CELL_TORONTO/subsys/database/OTH_INST
```

次に、以下のいずれかを行います。

- プリンシパル・パスワードで dcelogin を行う (OS/2 の場合)
- プリンシパル・パスワードで dce_login を行う (UNIX、 Windows オペレーティング・システム)

この後、次を行ってください。

- cdscp < cdscp.inp

オブジェクトを表示するには、次のコマンドを使用します。

```
cdscp show object /.:/subsys/database/ROUTE1
```

OSF DCE コマンドに関する詳細は、以下の DCE 資料を参照してください。

- *OSF DCE 管理者ガイド*
- *OSF DCE 管理者リファレンス*

各オブジェクト・クラスの属性

DCE 環境では、各オブジェクトおよびオブジェクト属性は、オブジェクト ID (OID) によって識別されます。各 OID は割り振り権限の階層から入手され、最高位の権限は国際標準化機構 (ISO) です。

表19 は、各オブジェクト・クラスの持つ属性を示し、また、343ページの表20 はオブジェクト・クラスの属性を示しています。

表 19. オブジェクト属性クラス

オブジェクト・クラス	オブジェクト ID (OID)	必須属性	任意選択属性
(DB) Database_Object	1.3.18.0.2.6.12	DAU、DOT、DDP、DNN	DCO、DPN、DRL、DLN、DCP、DPR

表 19. オブジェクト属性クラス (続き)

オブジェクト・クラス	オブジェクト ID (OID)	必須属性	任意選択属性
(DL) Database_Locator_Object	1.3.18.0.2.6.13	DOT、DCP	DCO、DPN、DRL
(RI) Routing_Information_Object	1.3.18.0.2.6.14	DOT、DTI	DCO、DPN、DRL

表 20. オブジェクト・クラス属性

属性名	OID	最小の長さ	最大の長さ	構文
(DAU) DB_Authentication	1.3.18.0.2.4.39	1	1024	Char
(DCO) DB_Comment	1.3.18.0.2.4.30	1	1024	Char
(DCP) DB_Communication_Protocol	1.3.18.0.2.4.31	1	1024	Char
(DDP) DB_Database_Protocol	1.3.18.0.2.4.32	1	1024	Char
(DLN) DB_Database_Locator_Name	1.3.18.0.2.4.33	1	1024	Char
(DNN) DB_Native_Database_Name	1.3.18.0.2.4.34	1	1024	Char
(DOT) DB_Object_Type	1.3.18.0.2.4.35	1	1	Char
(DPN) DB_Product_Name	1.3.18.0.2.4.36	1	1024	Char
(DRL) DB_Product_Release	1.3.18.0.2.4.37	1	1024	Char
(DTI) DB_Target_Database_Info	1.3.18.0.2.4.38	1	1024	Char
(DPR) DB_Principal	1.3.18.0.2.4.63	1	1024	Char
注: DCP、DDP、および DTI には複数の値を使用できます。他の属性には 1 つの値だけしか使用できません。				

各属性の詳細

次の節では、各属性を説明します。

注: DCE ディレクトリー・サービスは、項目が DB2 にとって有効であるか調べません。必ず、必須の属性を入力し、正確な値を入力するようにしてください。

DB_Authentication (DAU)

オブジェクトで必要な認証方式。この属性は、DB2 サーバーのデータベース・オブジェクトには必須です。この値は、CLIENT、SERVER または DCE にする必要があります。

DB_Principal (DPR)

認証方式が『DCE』である場合は、この属性には DCE プリンシパルを入力してください。

DB_Comment (DCO)
文書化のためだけに使用されます。

DB_Communication_Protocol (DCP)
各値がサポートされているネットワーク・プロトコルを記述するトークンから成る、複数値属性。ネットワーク・プロトコルの例としては、TCP/IP、APPC、IPX/SPX および NetBIOS があります。各トークンはセミコロンで区切られます。トークンとトークンの間にはスペースを入れないでください。

- TCP/IP のトークンは以下のとおりです。
  1. tcpip
  2. 宛先ノードのホスト名
  3. 着信 TCP/IP 接続要求を調べるためにオブジェクトで使用されるポート番号
  4. (任意選択) 機密保護は NONE にも SOCKS にもすることができる

  たとえば、`tcpip;HOSTNAME;1234` です。
- APPC のトークンは以下のとおりです。
  1. appc
  2. オブジェクトが属するターゲットのネットワーク ID
  3. ターゲットがある LU 名
  4. LU 内のオブジェクトを表すトランザクション・プログラム名 (TPN) (DB2 (MVS/ESA 版) の場合、TPN には DB2DRDA を使用してください。)
  5. モード名
  6. ターゲットで使用される機密保護のタイプ。値は次のとおりです。
     - NONE
     - PROGRAM
     - SAME

  たとえば、`appc;NETID;TARGETLU;TPNAME;MODE;PROGRAM` です。

  注: APPC の場合、クライアントはそのローカル制御点 (CP) を LU 名として使用します。
- (OS/2 およびサポートされている Windows オペレーティング・システムのみ) IPX/SPX のトークンは以下のとおりです。
  1. ipxspx
  2. ファイル・サーバーの名前
  3. オブジェクト名

  たとえば、`ipxspx;SVR_NAME;OBJ_NAME` です。

- (OS/2 およびサポートされている Windows オペレーティング・システムのみ) NetBIOS のトークンは以下のとおりです。

  1. netbios
  2. サーバーのノード名

  たとえば、クライアント・アダプター番号が登録値 *db2clientadpt* またはデータベース・マネージャー構成パラメーター *dft_client_adpt* のいずれかにあれば、 `netbios;SVR_NNME` となります。

- (サポートされている Windows オペレーティング・システムのみ) NPIPE のトークンは以下のとおりです。

  1. NPIPE
  2. サーバーのコンピューター名
  3. サーバーのインスタンス名

  たとえば、`npipe;computername;instance` です。

DB_Database_Protocol (DDP)
ターゲット・データベースでサポートされている 1 つまたは複数のデータベース・プロトコル。この値の例としては、DB2RA と DRDA があります。以下のコマンドは、 2 つのプロトコルを追加するための *cdscp* コマンドです。

```
add object /.:/subsys/database/AIXDB1 DB_Database_Protocol db2ra
add object /.:/subsys/database/AIXDB1 DB_Database_Protocol drda
```

DB_Database_Locator_Name (DLN)
データベース・ロケーター・オブジェクトの DCE 名。データベース・オブジェクト内では、これが DBMS インスタンスの名前です。経路指定情報オブジェクトでは、この名前は、DB2 コネクト・インスタンスの名前です。

たとえば、 `/.../CELL_TORONTO/subsys/database/AIX_INST` となります。

DB_Native_Database_Name (DNN)
データベースを含むインスタンス内で認識するのに使用するデータベース名または別名。これは、インスタンス上のローカル・アプリケーションがそのデータベースに接続するために使用する名前です。

DB2 ユニバーサル・データベースのデータベースの場合、この名前は最大で 8 文字です。それ以外のデータベースの場合は、名前の長さは異なることがあります。たとえば、DB2 (MVS/ESA 版) 上のデータベースの場合は、この名前を最大 18 文字にすることができます。

DB_Object_Type (DOT)
オブジェクトのタイプ。この属性はすべてのオブジェクトに必須で、以下の内のいずれかにすることができます。

D データベース・オブジェクト

L データベース・ロケーター・オブジェクト

R 経路指定情報オブジェクト

DB_Product_Name (DPN)
製品の識別子。文書化のためだけに使用されます。

DB_Product_Release (DRL)
製品のリリース・レベル。文書化のためだけに使用されます。

DB_Target_Database_Info (DTI)
各値が決まった数のトークンから成り、セミコロンで区切られている複数値属性。トークンとトークンの間にはスペースを入れないでください。トークンは、以下の順番にしてください。

1. データベース名。経路指定サービスが提供されるターゲット・データベースの DCE 名。値 *OTHERDBS は、経路指定情報オブジェクトで明示的には定義されないターゲット・データベースのデフォルト・ゲートウェイを指定します。
2. ルーターからのアウトバウンド・プロトコル。ターゲット・データベースによって使用されるデータベース・プロトコル、またはそのターゲット・データベースと通信するために経路指定 DB2 コネクト・インスタンスが使用するデータベース・プロトコル。たとえば、DRDA。
3. ルーターへのインバウンド・プロトコル。経路指定 DB2 コネクト・インスタンス・オブジェクトが受け入れるデータベース・プロトコル。たとえば、DB2RA。
4. ゲートウェイでの認証。有効な値は 0 または 1 です。詳細は、348ページの表21 を参照してください。
5. DB2 コネクトのゲートウェイに固有の情報を含むパラメーター・ストリング。このストリングには、トークンが以下に示す順序で入っていなければなりません。トークンは、コンマで区切ってください。トークンが指定されていない場合、デフォルトが使用されます。
   - マップ・ファイル名。デフォルトの SQLCODE マッピングを指定変更する SQLCODE マッピング・ファイルの完全修飾名。 SQLCODE マッピングをオフにするには、NOMAP を指定してください。
   - D。特定の SQLCODE が戻されると、アプリケーションは DRDA サーバー・データベースから切断します。 SQLCODE についての詳細は、*DB2 コネクト 使用者の手引き* を参照してください。
   - INTERRUPT_ENABLED。 DB2 コネクトは、クライアントが DRDA サーバーに接続しているときに割り込みを出すと、接続を除去し、その作業単位をロール・バックします。

   以下は、その例です。

```
NOMAP
/u/username/sqllib/map/dcs1new.map,D
/u/username/sqllib/map/dcs1new.map,D,INTERRUPT_ENABLED
```

デフォルトを使用する場合には、トークンの順番を保存するためにコンマを 1 つ使用してください。たとえば、以下のようにします。

```
,D
```

または

```
,,INTERRUPT_ENABLED
```

パラメーター・ストリングに関する詳細については、*DB2 コネクト 使用者の手引き* を参照してください。

6. 経路指定サービスを提供する DB2 コネクト・インスタンスの DCE 名。

以下は DB_Target_Database_Info の例です。

```
/.../CELL_TORONTO/subsys/database/MVSDB;¥
drda;db2ra;0;;¥
/.../CELL_TORONTO/subsys/database/GW_INST
```

注: 上記の例では、円記号 (¥) が行連結文字です。

ディレクトリー・サービス機密保護

DB2 コネクト・ゲートウェイのない環境で DCE ディレクトリー・サービスを使用している場合は、認証はデータベース・サーバーにアクセスする他のクライアントの場合と同様に用いられます。詳細については、235ページの『サーバーに対する認証方式の選択』を参照してください。

DB2 コネクト・ゲートウェイのある環境で DCE ディレクトリー・サービスを使用する場合は、 DB2 コネクト管理者がユーザー名とパスワードの妥当性検査を行う場所を決めます。 DCE ディレクトリーを用いて、以下を指定します。

- DB2 コネクト・ワークステーションを表すデータベース・ロケーター・オブジェクト内での通信プロトコルの機密保護タイプ。 (リモート・クライアントが APPC 接続によって DB2 コネクト拡張エディション・ゲートウェイに接続されている場合は、ゲートウェイの *DCE ロケーター・オブジェクト* に機密保護タイプ NONE を指定してください。)
- データベース・オブジェクト内での認証タイプ。
- データベース・オブジェクト (または関連するロケーター・オブジェクト) の通信プロトコルの機密保護タイプ。
- 経路指定情報オブジェクト内のゲートウェイ・トークンでの認証。

348ページの表21 には、これらの値の可能な組み合わせ、およびこの各組み合わせに対して妥当性検査を実行する場所が示してあります。次の表に示されている組み合わせは、 DCE ディレクトリー・サービスを持つ DB2 コネクトによってサポートされています。

表 21. APPC 接続を使用する DCE で有効な機密保護のシナリオ

	サーバーのデータベース・オブジェクト		経路指定オブジェクト	妥当性検査
事例	認証	機密保護	ゲートウェイでの認証	
1	CLIENT	SAME	0	リモート・クライアント (または DB2 コネクト・ワークステーション)
2	CLIENT	SAME	1	DB2 コネクト・ワークステーション
3	SERVER	PROGRAM	0	DRDA サーバー
4	SERVER	PROGRAM	1	DB2 コネクト・ワークステーションと DRDA サーバー
5	DCE	NONE	適用されない	DCE

表22 には、これらの値の可能な組み合わせ、およびこの各組み合わせに対して TCP/IP 接続を使用して妥当性検査を実行する場所が示してあります。次の表に示されている組み合わせは、 DCE ディレクトリー・サービスを持つ DB2 コネクトによってサポートされています。

表 22. TCP/IP 接続を使用する DCE で有効な機密保護シナリオ

事例	認証	ゲートウェイでの認証	妥当性検査
1	CLIENT	0	クライアント
2	CLIENT	1	DB2 コネクト・ワークステーション
3	SERVER	0	DRDA サーバー
4	適用されない	適用されない	なし
5	DCE	適用されない	DCE

どの組み合わせも APPC と TCP/IP の両方に適用されます。それぞれについて、以下にさらに詳しく説明します。

1. ユーザー名とパスワードは、リモート・クライアントでだけ妥当性検査が行われる。(ローカル・クライアントの場合は、ユーザー名とパスワードは DB2 コネクト・ワークステーションでだけ妥当性検査が行われる。)

   ユーザーは、最初にサインオンした場所で、認証されることになります。ユーザー ID はネットワークで送信されますが、パスワードがそのように送信されることはありません。すべてのワークステーションに十分な機密保護機能がある場合にのみ、このタイプの機密保護を使用してください。

2. ユーザー名とパスワードは、 DB2 コネクト・ワークステーションでだけ妥当性検査が行われる。パスワードはネットワークでリモート・クライアントから DB2 コネクト・ワークステーションに送信されますが、 DRDA サーバーに送信されることはありません。
3. ユーザー名とパスワードは、DRDA サーバーでだけ妥当性検査が行われる。パスワードはネットワークでリモート・クライアントから DB2 コネクト・ワークステーションに送信され、また、DB2 コネクト・ワークステーションから DRDA サーバーに送信されます。
4. ユーザー名とパスワードは、DB2 コネクト・ワークステーションと DRDA サーバーの両方で妥当性検査が行われる。パスワードはネットワークでリモート・クライアントから DB2 コネクト・ワークステーションに送信され、また、DB2 コネクト・ワークステーションから DRDA サーバーに送信されます。

   妥当性検査が 2 つの場所で実行されるため、 DB2 コネクト・ワークステーションと DRDA サーバーの両方で保持されているユーザー名とパスワードのセットは同じである必要があります。
5. DCE トークンは DCE 機密保護サーバーから入手される。

注:

1. AIX ベースのシステムの場合、機密保護タイプ SAME を使用するユーザーはすべて、 AIX システム・グループに属している必要があります。
2. リモート・クライアントを持つ AIX ベースのシステムの場合、 DB2 コネクト・ワークステーション上で実行する DB2 コネクト製品のインスタンスは AIX システム・グループに属している必要があります。
3. DRDA サーバーへのアクセスは、それ自身の機密保護機構やサブシステム (たとえば、仮想記憶通信アクセス方式 (VTAM) および資源アクセス管理機能 (RACF)) によって制御されます。保護されたデータベース・オブジェクトへのアクセスは、SQL の **GRANT** および **REVOKE** ステートメントによって制御されます。

構成パラメーターとレジストリー変数

以下の構成パラメーターは、DCE ディレクトリーで使用されます。それらの値の例を示します。詳細は、*管理の手引き: パフォーマンス* の『DB2 の構成』の章にある『配布サービス』を参照してください。

- *dir_obj_name* は、 *dir_path_name* と連結されるデータベース・インスタンス名です。インスタンス名が ATTACH コマンドの宛先として使用される場合は、この名前を 8 文字以下で、すべて大文字にしなければなりません。たとえば、次のようになります。

  ```
  AIX_INST
  ```

- *dir_type* は、 DCE ディレクトリー・サービスを使用するか否かを表します。 DCE ディレクトリー・サービスを使用可能にするには、このパラメーターを次のように設定します。

```
DCE
```

dir_type を NONE に設定すると、 DCE ディレクトリー・サービスの使用をサポートしないデータベース・クライアント上では、このパラメーターを更新できないので注意してください。

- *dir_path_name* は、 DCE 管理者によって提供されているディレクトリー・パス名です。たとえば、次のようになります。

  ```
  /.:/subsys/database/
  ```

- *route_obj_name* は、経路指定情報オブジェクトの DCE ディレクトリー・サービス名を提供する任意指定パラメーターです。この名前は、次の例のように完全修飾にすることもできます。

  ```
  /.:/subsys/database/ROUTE1
  ```

  または、次の例のように *dir_path_name* に連結される 1 パート名にすることもできます。

  ```
  ROUTE1
  ```

- *dft_client_comm* は、クライアントが使用する通信プロトコルを指定する任意指定 DCE パラメーターです。たとえば、次のようになります。

  ```
  TCPIP
  ```

  このパラメーターは、次の例のように複数のプロトコルを指定することもできます。

  ```
  TCPIP,APPC   (UNIX ベース・プラットフォームの場合)
  TCPIP,APPC,IPXSPX,NETBIOS    (OS/2 プラットフォームの場合)
  TCPIP,APPC,IPXSPX,NETBIOS,NPIPE    (サポートされている Windows
                                      オペレーティング・システムの場合)
  ```

- *dft_client_adpt* は、 OS/2 およびサポートされている Windows オペレーティング・システムで、 NetBIOS プロトコルのデフォルトのクライアント・アダプター番号を指定する任意指定 DCE パラメーターです。この番号の有効範囲は、0 ～ 15 です。このパラメーターに非数字が入っていると、値はゼロ (0) と見なされます。また、このパラメーターに許されている範囲外の値が入っている場合も、値はゼロ (0) と見なされます。

以下のパラメーターの場合、レジストリー変数はそれらのパラメーター値を指定変更することがあります。

構成パラメーター	レジストリー変数
dir_path_name	DB2DIRPATHNAME
route_obj_name	DB2ROUTE
dft_client_comm	DB2CLIENTCOMM
dft_client_adpt	DB2CLIENTADPT

これらのレジストリー変数の設定規則は、それぞれに対応する構成パラメーターと同じです。たとえば、DB2CLIENTCOMM は *dft_client_comm* と同様、次の例のようにそれぞれをコンマで区切って複数の値を持つことができます。

```
db2set DB2CLIENTCOMM=TCPIP,APPC
```

CATALOG コマンド、ATTACH コマンド、および CONNECT ステートメント

DCE 情報には以下のコマンドを指定することが必要です。

- CATALOG GLOBAL DATABASE コマンド
- CONNECT ステートメント
- ATTACH コマンド

CATALOG GLOBAL DATABASE コマンド

クライアントとサーバーのパス名が異なっているか、またはデータベース名が 8 文字以上か大文字と小文字が混ざっている場合には、 CATALOG GLOBAL DATABASE コマンドを使用してください。データベース管理者がデータベースとディレクトリー・タイプ DCE の DCE 名を入力します。

たとえば、次のようにします。

- クライアントとサーバーのパス名が異なっているとき、たとえば *dir_path_name = /.../CELL_TORONTO/subsys/database/* の場合。

  ```
  CATALOG GLOBAL DATABASE
  /.../CELL_VANCOUVER/subsys/database/VMDB AS VANVMDB
  USING DIRECTORY DCE WITH "comment-string"
  ```

- データベース名が 8 文字以上の場合 (たとえば、DB_LONGNAME)。

  ```
  CATALOG GLOBAL DATABASE
  /.../CELL_VANCOUVER/subsys/database/DB_LONGNAME AS VANVMDB
  USING DIRECTORY DCE WITH "comment-string"
  ```

CONNECT ステートメント

該当する DCE ディレクトリー・インスタンスを検索するには、クライアントはデータベースまたは DBMS インスタンスの完全修飾 DCE 名を知らなければなりません。CONNECT ステートメントに名前を指定するには、以下のいくつかの方法があります。

- 別名を入力する。たとえば、次のようにします。

  ```
  CONNECT TO VANVMDB
  ```

- 1 パート名を入力する。たとえば、次のようにします。

  ```
  CONNECT TO VMDB
  ```

この場合、クライアントで指定されているパス名が、サーバーで指定されているパス名と同じでなければなりません。 (パス名は、*dir_path_name* 構成パラメーターまたはそれに対応する登録値で指定されます。)

ATTACH コマンド

クライアントの有効なパス名は、ターゲット DBMS インスタンスのパス名と同じです。

dir_path_name がクライアントとサーバーで同じであり (たとえば、/.../CELL_TORONTO/subsys/database/)、かつ *dir_obj_name* がデータベース・サーバーでは AIX_INST である場合、このインスタンスへの接続コマンドは次のようになります。

```
ATTACH TO AIX_INST
```

クライアントがデータベースに接続する方法

353ページの図8 は、 2 つの DCE セルがあるデータベース・ネットワークの構成例を示しています。 /.../CELL_TORONTO および /.../CELL_VANCOUVER は、セルの名前です。 (これらのセルには /.:/subsys/database/ というディレクトリーが含まれており、表には図示されていませんが、他の例では使用されます。)

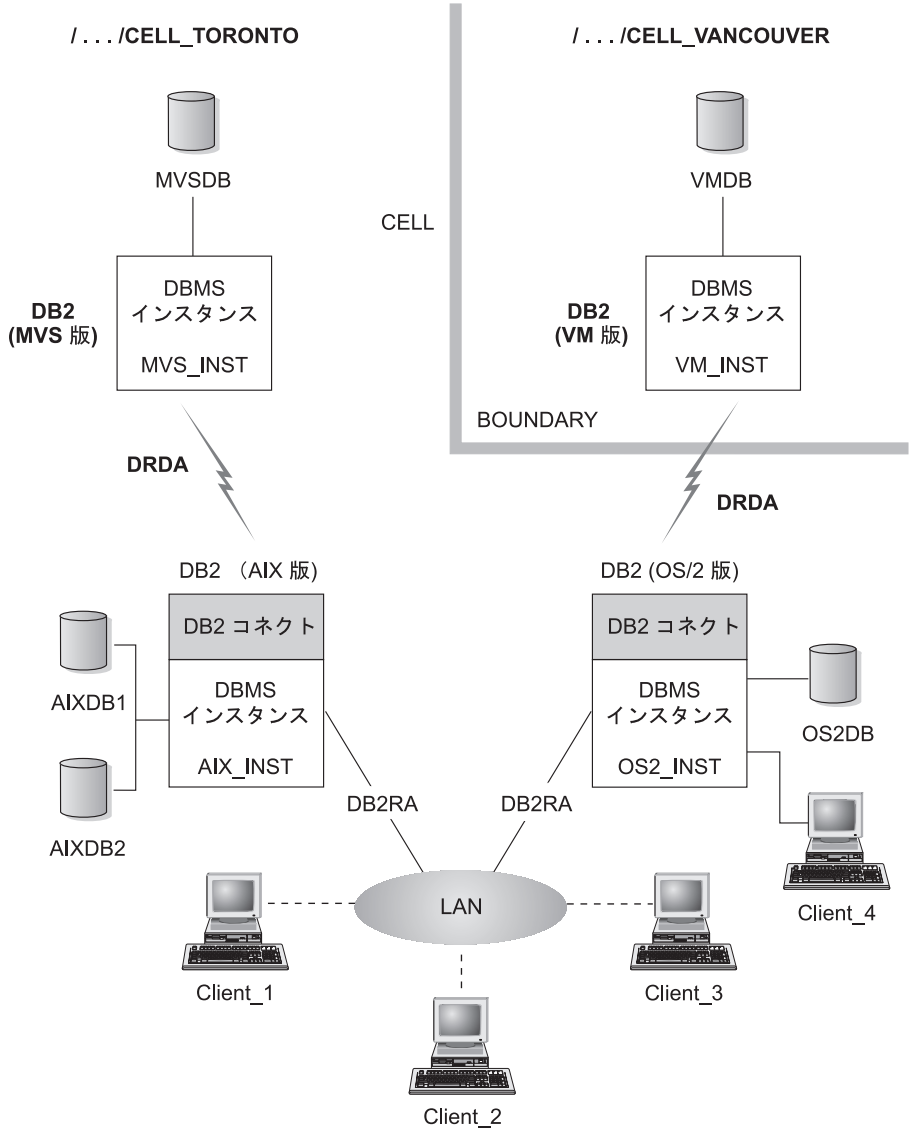


図 8. ネットワーク・データベースの構成

TORONTO セル内のクライアントが、 TORONTO と VANCOUVER の両方のセルにあるすべてのデータベースにアクセスできるようにするには、値をデータベース・マネージャー構成パラメーターに指定しなければなりません。そのように指定すると、以下のオブジェクトが作成されます。

- データベースごとのデータベース・オブジェクト。

- DB2 (AIX 版) および DB2 (OS/2 版) 用の 2 つのデータベース・サーバー上にあるターゲット・データベースのためのデータベース・ロケーター・オブジェクト。
- すべてのクライアントに知られている単一の経路指定情報オブジェクト。属性は、MVSDB および VMDB データベース用にどの DB2 コネクト・ノードを使用するかを指定します。

以下の部分では、クライアントがデータベースに接続する方法の例が示されています。

- 同一セル内の複数のデータベースへの接続
- 異なるセル内のデータベースへの接続.

これらの例には、指定しなければならないデータベース・マネージャー構成パラメーターも含まれています。

同一セル内の複数のデータベースへの接続

この節では、クライアントが同一セル内の複数のデータベースに接続する方法の例をいくつか示します。

1. Client_1 が AIXDB2 に接続します。データベースは、クライアントと同じディレクトリー・パス名を共用します。

   データベース管理者は、以下のことをする必要があります。

   - *dir_path_name* 構成パラメーター (または DB2DIRPATHNAME 登録値) に、ディレクトリー・パス名の値を指定します。
   - 構成パラメーター *dir_type* のディレクトリー・サービスの値を DCE に指定します。
   - 構成パラメーター *dft_client_comm* (または DB2CLIENTCOMM 登録値) に、通信プロトコルを指定します。

   ローカル・システム・データベース・ディレクトリーには AIXDB2 はないので、完全修飾名を使用して DCE ディレクトリーが検索されます。名前は、構成パラメーター *dir_path_name* (または DB2DIRPATHNAME 登録値) の値を AIXDB2 と連結して作成されます。

   イベントの順序は次のようになります。

   a. Client_1 が、データベース `/.../CELL_TORONTO/subsys/database/AIXDB2` の DCE 名を使用して、 AIXDB2 のデータベース・オブジェクトを入手する。
   b. このオブジェクトから、 Client_1 は AIXDB2 が DB プロトコル DB2RA (これは、 Client_1 が使用しているのと同じプロトコル) を使用していることを知る。
   c. DB プロトコルが一致しているので、 Client_1 は AIX_INST 用の データベース管理システム・ロケーター・オブジェクトを読み取り、使用するものと一致する通信プロトコル属性値を検索し、その情報を使用してその database インスタンスとの会話を開始します。

2. Client_3 が MVSDB に接続します。このデータベースは、クライアントと同じディレクトリー・パス名を共用しますが、クライアントとは異なるデータベース・プロトコルを使用します。

   データベース管理者は、以下のことをする必要があります。

   - *dir_path_name* 構成パラメーター (または DB2DIRPATHNAME 登録値) に、ディレクトリー・パス名の値を指定します。
   - 構成パラメーター *dir_type* のディレクトリー・サービスの値を DCE に指定します。
   - 構成パラメーター *dft_client_comm* (または DB2CLIENTCOMM 登録値) に、通信プロトコルを指定します。
   - 構成パラメーター *route_obj_name* (または DB2ROUTE 登録値) に、デフォルトの経路指定情報オブジェクトの DCE 名を指定します。

   イベントの順序は次のようになります。

   a. Client_3 が、データベース `/.../CELL_TORONTO/subsys/database/MVSDB` の DCE 名を使用して、 MVSDB のデータベース・オブジェクトを入手する。
   b. このオブジェクトから、 Client_3 は MVSDB が DB プロトコル DRDA (これは、 Client_3 は使用していないプロトコル) を使用していることを知る。
   c. ついで、Client_3 は *route_obj_name* 構成パラメーターまたは DB2ROUTE 登録値に定義されている名前を使用して、経路指定情報オブジェクトを入手する。クライアントは、MVSDB のターゲット・データベース情報を検索します。
   d. Client_3 は、 MVSDB ターゲット・データベースに関連したデータベース・ロケーター・オブジェクトを読み取り、通信プロトコルを検索し、SQL CONNECT 要求をルーターに送信する。
   e. すると、ルーターが MVSDB との APPC 接続を設定する。

異なるセル内のデータベースへの接続

この節では、データベース・プロトコルが異なるときに、クライアントが異なるセル内のデータベースに接続する方法の例について説明します。

1. Client_3 は以前に、以下の情報を使用して構成されています。
   - DCE ディレクトリー・サービス。 *dir_type* パラメーターに DCE を指定。
   - CELL_VANCOUVER 以外のセル。構成パラメーター *dir_path_name* で指定。たとえば、次のようになります。

     ```
     /.../CELL_TORONTO/subsys/database/
     ```

2. Client_3 を VMDB に接続するには、データベース管理者は以下のことをする必要があります。

- ローカル・システム・データベース・ディレクトリーに VMDB を明示的にカタログする。 VMDB の DCE 名をローカルで固有のデータベース別名と関連させ、その別名の値を使用して CONNECT ステートメントを発行します。たとえば、次のようにします。

  ```
  CATALOG GLOBAL DATABASE
  /.../CELL_VANCOUVER/subsys/database/VMDB AS VANVMDB
  USING DIRECTORY DCE WITH "comment-string"
  ```

  その後に、次のように続けます。

  ```
  CONNECT TO VANVMDB
  ```

- 構成パラメーター *dft_client_comm* (または DB2CLIENTCOMM 登録値) に、通信プロトコルを指定します。
- 構成パラメーター *route_obj_name* (または DB2ROUTE 登録値) に、デフォルトの経路指定情報オブジェクトの DCE 名を指定します。

イベントの順序は次のようになります。

a. Client_3 は VMDB のシステム・データベース・ディレクトリー内で VANVMDB の完全修飾 DCE 名を検索する。
b. Client_3 が、データベース /.../CELL_VANCOUVER/subsys/database/VMDB の DCE 名を使用して、 VMDB のデータベース・オブジェクトを入手する。
c. このオブジェクトから、Client_3 は VMDB が DB プロトコル DRDA (これは、Client_3 は使用していないプロトコル) を使用していることを知る。
d. ついで、Client_3 は *route_obj_name* 構成パラメーターまたは DB2ROUTE 登録値に定義されている名前を使用して、経路指定情報オブジェクトを入手する。クライアントは、VMDB のターゲット・データベース情報を検索します。
e. Client_3 は、VMDB ターゲット・データベースに関連したデータベース・ロケーター・オブジェクトを読み取り、通信プロトコルを検索し、SQL CONNECT 要求をルーターに送信する。
f. すると、ルーターが VMDB との APPC 接続を設定する。

ディレクトリー探索の方法

すべてのターゲット・データベースが同じディレクトリー・パス名を共用する環境で DCE ディレクトリーを使用する場合、クライアント上にはディレクトリー・パス名は必要ありません。

この節では、以下のものについてディレクトリーを探索する順序について説明します。

- ATTACH コマンド
- CONNECT ステートメント

ATTACH コマンド

図9 では、クライアントが ABC_INST という データベース管理システム・インスタンスに接続するときに、ディレクトリーが探索される方法を示しています。

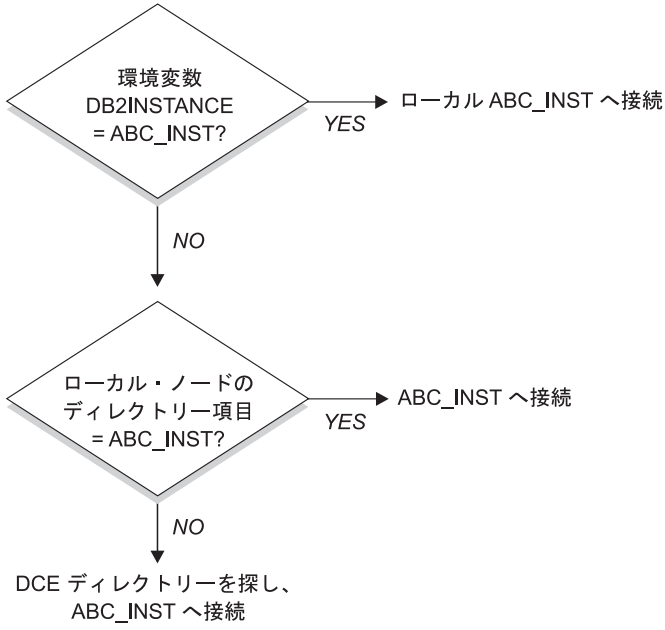


図 9. データベースに接続するためにディレクトリーが使用される方法

CONNECT ステートメント

358ページの図10 では、クライアントが DBTEST というデータベースに接続するときに、ディレクトリーが探索される方法が示されています。

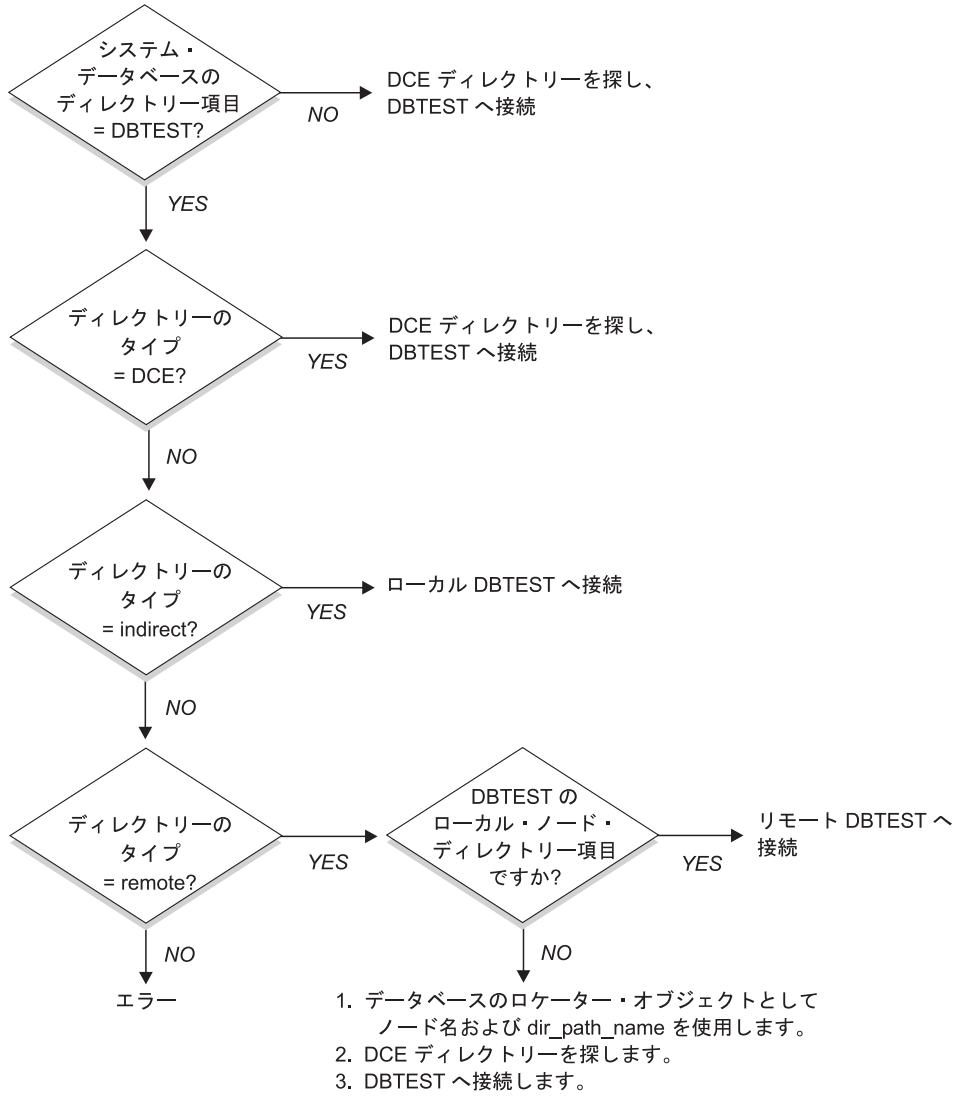


図 10. データベースに接続するためにディレクトリーが使用される方法

DCE ディレクトリー情報の一時的な指定変更

DCE ディレクトリー情報を指定変更するためにローカル・データベース・ディレクトリーを使用することができます。たとえば、DBTEST に接続する (CONNECT TO DBTEST を実行する) 場合 (ここで、 `/.:/subsys/database/DBTEST` は JAGUAR というホストがあるものと DCE ディレクトリーに定義されている)、 DBTEST を STORM

というホストにある別のデータベースに一時的に変更することができます。 DBTEST を、STORM を指すノード・ディレクトリー項目が指定されたリモート・データベースとして、ローカルにカタログ化してください。

データベースの別名は、クライアントのディレクトリー・パス名の後にそのデータベースの DCE 名を付けずに作成することができます。このコマンドの詳細は、351ページの『CATALOG GLOBAL DATABASE コマンド』を参照してください。

ディレクトリー・サービスの作業

DCE ディレクトリー・サービスを設定し、使用するためには、以下の作業を実行する必要があります。次の各節では、各作業を詳細に説明します。

- DCE 管理者の作業

  DCE 管理担当者は、DCE ディレクトリーを更新して、新規のデータベース・リソース情報を追加できるようにする必要があります。

- データベース管理者の作業

  データベース管理者は、DCE ディレクトリーを更新し、 DB2 のインストールと構成の情報を提供しなければなりません

- データベース・ユーザーの作業

  データベース・ユーザーは、DCE にログインし、ターゲット・データベース名を知っている必要があります。

加えて、ネットワーク管理者は各ユーザー・ノードのネットワーク・アクセスを設定します。詳細は、ネットワークの資料を参照してください。

DCE 管理者の作業

DCE 管理者は、ディレクトリー・オブジェクトを作成または読み取る前に、以下の作業を行わなければなりません。

- DB2 用のサブディレクトリーを割り当てる。(たとえば、/.:/subsys/database)
- ディレクトリー・オブジェクトを作成するよう、データベース管理担当者に特権を付与する。
- ディレクトリー・オブジェクトを読み取るよう、データベース・ユーザーに特権を付与する。
- *DCE 属性表* に、新規の DCE ディレクトリー・オブジェクト属性に関する情報を追加する。

  CDS 属性表 (UNIX プラットフォームでは */etc/dce/cds_attributes*、 OS/2 では *X:¥opt¥dcelocal¥etc¥cds_attr*、ただし "X" は該当するドライブ) を編集し、以下を追加する。

```
1.3.18.0.2.4.30 DB_Comment                char
1.3.18.0.2.4.31 DB_Communication_Protocol char
1.3.18.0.2.4.32 DB_Database_Protocol      char
1.3.18.0.2.4.33 DB_Database_Locator_Name  char
1.3.18.0.2.4.34 DB_Native_Database_Name   char
1.3.18.0.2.4.35 DB_Object_Type            char
1.3.18.0.2.4.36 DB_Product_Name           char
1.3.18.0.2.4.37 DB_Product_Release        char
1.3.18.0.2.4.38 DB_Target_Database_Info   char
1.3.18.0.2.4.39 DB_Authentication         char
1.3.18.0.2.4.63 DB_Principal              char
```

- ユーザーが DCE ディレクトリー・サービスを使用してデータベースにアクセスする必要があるときに、 DCE が確実に稼働するようにします。

詳細は、ご使用のプラットフォームの DCE 資料を参照してください。

データベース管理者の作業

データベース管理担当者は、次の作業を行う必要があります。

- データベース・リソースのディレクトリー・サブツリーを、DCE 管理担当者から入手する。たとえば、`/.:/subsys/database` とします。
- DB2 データベース・マネージャーのインストール時に、 DB2 で必要な新規の DCE ディレクトリー・オブジェクト属性を追加するよう DCE 管理担当者に要請する。
- DCE ディレクトリー・サブツリーの中の各データベース管理システム・インスタンスの固有名を割り当てる。たとえば、`/.:/subsys/database/AIX_INST` とします。
- 各データベース管理システム・インスタンスに、DCE のデータベース・マネージャー構成パラメーターを指定する。
  - *dir_type*
  - *dir_obj_name*
  - *dir_path_name*
  - *route_obj_name*
  - *dft_client_comm*
  - *dft_client_adpt*

  構成パラメーターの中には、クライアントで設定された登録変数によって、一時的に指定変更できるものがあります。 詳細は、349ページの『構成パラメーターとレジストリー変数』を参照してください。
- DCE ディレクトリー・サブツリーの中の各データベースの固有名を割り当てる。データベース構成ファイルの *dir_obj_name* パラメーターに名前を指定します。
- DCE の *cdscp* コマンドを使用して DCE ディレクトリー・サービスを作成し、オブジェクトを作成して表示する。オブジェクトは、データベース・マネージャー・インストール・プロセス、およびデータベース・マネージャー・インスタンス始動プロセスとは別に作成されます。

オブジェクトには 3 つのタイプがあります。

- データベース・オブジェクトは、ターゲット・データベースごとに必要です。
- データベース・ロケーター・オブジェクトは、DB2 コネクト・インスタンスごとに、および複数のデータベースに関連する データベース管理システム・インスタンス (DB2 コネクトのないもの) ごとに必要です。
- ホスト・データベースにアクセスするためには、経路指定情報オブジェクトを作成する必要があります。

- それぞれの環境に応じて、データベース管理担当者は次のことを判別する必要があります。
  - クライアントがアクセスするデータベースの種類、使用する通信プロトコルの種類を考慮して、クライアントを論理グループにグループ化する方法。
  - 経路指定情報オブジェクトがいくつ必要か。
  - どのターゲット・データベースが各経路指定情報オブジェクトに記録される必要があるか。
  - どの経路指定情報オブジェクトを、どのクライアント・グループに知らせる必要があるか。

  オブジェクトの詳細は、337ページの『ディレクトリー・オブジェクトの作成』を参照してください。

データベース・ユーザーの作業

データベース・ユーザーは、次の作業を行う必要があります。

- データベースの名前を、データベース管理担当者から入手します。この名前は、単に一部分の名前か、または完全修飾名のいずれでもかまいません。
- 必要に応じて、DCE ディレクトリー・サービスで必要な値を登録変数に指定します。 クライアントで登録変数を設定して、構成パラメーターを一時的に指定変更することができます。
  - ホスト・データベース・アクセスが必要な場合、経路指定情報オブジェクト項目の完全修飾 DCE 名を、データベース管理担当者から入手します。この名前が *route_obj_name* に指定されていないか、または別の名前が指定されている場合には、ホスト・データベースに接続する前に DB2ROUTE 登録変数にこの名前を指定してください。
  - 使用したい通信プロトコルが *dft_client_comm* に指定されていないか、または別のプロトコルが指定されている場合は、 DB2CLIENTCOMM 登録変数にクライアントの通信プロトコルを指定してください。 以下に UNIX の**いくつかの**例を示します。

    ```
    db2set DB2CLIENTCOMM=tcpip
    db2set DB2CLIENTCOMM=appc
    db2set DB2CLIENTCOMM=tcpip,appc
    db2set DB2CLIENTCOMM=appc,tcpip
    ```

以下に OS/2 のいくつかの例を示します。

```
db2set DB2CLIENTCOMM=ipxspx
db2set DB2CLIENTCOMM=netbios
db2set DB2CLIENTCOMM=tcpip,ipxspx,netbios
db2set DB2CLIENTCOMM=netbios,tcpip,ipxspx,appc
```

Windows オペレーティング・システムのいくつかの例を示します。

```
db2set DB2CLIENTCOMM=npipe
db2set DB2CLIENTCOMM=netbios
db2set DB2CLIENTCOMM=tcpip,ipxspx,netbios
db2set DB2CLIENTCOMM=netbios,tcpip,ipxspx,appc,npipe
```

複数の通信プロトコルがある場合は、最初に指定されているプロトコルが使用されます。

- *dir_path_name* 構成パラメーターに定義されているディレクトリー・パスの中、または、DB2DIRPATHNAME 登録変数の中にない DCE 名を持つデータベースがある場合は、 CATALOG GLOBAL DATABASE コマンドを使用してデータベースを明示的にカタログします。 詳細は、351ページの『CATALOG GLOBAL DATABASE コマンド』を参照してください。
- ターゲット・データベースに接続するか、またはデータベース・インスタンスに接続する前に、DCE にログインします。ログイン・コマンドに関する詳細は、 *OSF DCE 管理者ガイド* を参照してください。

ディレクトリー・サービスの制約事項

この節では、サポートされないものについて説明します。

- 全部のデータベース・クライアントがサポートされるわけではありません。 DCE ディレクトリー・サービスがご使用の DB2 クライアントでサポートされるか否かについては、 *概説およびインストール* を参照してください。現在サポートが提供されているのは、すべての UNIX、OS/2、およびサポートされている Windows オペレーティング・システム用の DB2 クライアントだけです。
- クライアントは、 DCE ディレクトリー・サービスを使用して DB2 for OS/2 バージョン 1 のサーバーに接続することはできません。
- TCP/IP、APPC、NetBIOS、IPX/SPX、または NPIPE プロトコルのいずれか、あるいはそれらのすべてを使用できるのは、サポートされている Windows オペレーティング・システムのクライアントだけです。 OS/2 クライアントだけが TCP/IP、APPC、NetBIOS、および IPX/SPX プロトコルのいずれかまたはすべてを使用することができます。サポートされるすべての UNIX クライアントは、 TCP/IP と APPC プロトコルだけを使用できます。
- LIST DATABASE (または NODE) DIRECTORY COMMANDS は、ローカル・ディレクトリーの項目だけをリストし、 DCE ディレクトリーの項目はリストしません。DCE 内では、*cdscp show object* コマンドを使用して、オブジェクトを表示することができます。

- 以下の条件すべてが存在している場合には、データベース・マネージャー・インスタンスの所有者は、データベース・マネージャーを始動する前に、(db2start コマンドを使用して) DCE にログインしなければなりません。
  - データベース・マネージャー・インスタンスが、 *dir_type* 構成パラメーターによって、 DCE ディレクトリー・サービスをサポートするように構成されている。
  - セル・ディレクトリー・サービスのオブジェクトは、 DCE に明示的にログインすることによってのみ読み取ることができる。
  - 以下のいずれかをサポートするには、 DCE ディレクトリーにアクセスしなければならない。
    - 別のインスタンスにある (*tm_database* 構成パラメーターによって指定されている) トランザクション・マネージャー・データベース
    - DCE ディレクトリー・サービスをサポートできない、または DCE ディレクトリー・サービスを使用するように構成されていないクライアント

  注: DCE のログインを実行するときには、チケット存続時間が長いプリンシパルを使用してください。
- DDCS バージョン 2.2 (またはそれ以前) のゲートウェイを使用して、 DCE ディレクトリー・サービスを使用するクライアントを DRDA サーバーに接続するときは、そのゲートウェイのローカル・ディレクトリーにデータベース別名をカタログする必要があります。このデータベース別名はクライアント上の別名と同じ名前でなければならず、同じデータベースを表していなければなりません。
- サポートされている Windows オペレーティング・システムのクライアントを使用している場合、 DB2DCE.DLL が使用されます。このファイルは *sqllib* サブディレクトリーの *bin* サブディレクトリーにあります。 DCE のプロバイダーが Gradient の場合は、デフォルトではファイル DB2DCE.GRD が DB2DCE.DLL と同じ働きをします。 DCE のプロバイダーが IBM の場合は、ファイル DB2DCE.IBM を DB2DCE.DLL にコピーしなければなりません。

付録C. データベース・リカバリー用のユーザー出口

ここでは、『データの移動』の章で扱われている内容と類似した概要を提供します。

注: *コマンド解説書* および *管理 API 解説書* に掲載されているトピックおよび同様のトピックはすべて、 *データ回復と高可用性の手引きと解説書* に統合されました。

このトピックについては、*データ回復と高可用性の手引きと解説書* が主要かつ唯一の情報源になります。

付録D. 複数のデータベース区画に対するコマンドの発行

区分データベース・システムでは、インスタンスにあるマシンで、あるいはデータベース区画サーバー (ノード) で実行するコマンドを発行したい場合があります。このような場合には、**rah** コマンドまたは **db2_all** コマンドを使用することができます。 **rah** コマンドは、ユーザーが、インスタンスにあるすべてのマシンで実行したいコマンドを発行できるようにするものです。インスタンスにあるデータベース区画サーバーでコマンドを実行したい場合は、 **db2_all** コマンドを実行します。この節では、これらのコマンドについての概要を説明します。以下の情報は、区分データベース・システムだけに適用されます。

注:

1. UNIX ベースのプラットフォームでは、ユーザーのログイン・シェルは、 Korn シェルまたはその他のシェルにすることができますが、特殊文字を含むコマンドをシェルが処理する方法は、シェルによってさまざまに異なります。
2. Windows NT で **rah** コマンドまたは **db2_all** コマンドを実行するには、管理者グループのメンバーになっているユーザー・アカウントでログオンしなければなりません。

コマンドの有効範囲を調べるには、*コマンド解説書* を参照してください。本書では、コマンドが 1 つのデータベース区画サーバーで実行しているのか、それともすべてのサーバーで実行しているのかを示しています。コマンドが 1 つのデータベース区画サーバーで実行している場合に、すべてのサーバーで実行させたければ、 **db2_all** を使用してください。 **db2trc** コマンドは例外で、 1 つのマシン上のすべての論理ノード (データベース区画サーバー) で実行します。すべてのマシン上のすべての論理ノードで **db2trc** を実行したい場合は、 **rah** を使用してください。

コマンド

あるデータベース区画サーバーで次々と順番にコマンドを実行することもできますし、複数のコマンドを並列に実行することもできます。 UNIX ベースのプラットフォームでは、コマンドを並列に実行するときは、出力をバッファーに送信して、表示用に収集する (デフォルトの処置) よう選択するか、またはコマンドが出されるマシンに出力を表示することができます。 Windows NT では、コマンドを並列に実行すると、そのコマンドが出されるマシンに出力が表示されます。

rah コマンドを使用するには、次のように入力してください。

```
rah command
```

db2_all コマンドを使用するには、次のように入力してください。

```
db2_all command
```

rah 構文に関するヘルプを表示するには、次のように入力してください。

```
rah "?"
```

コマンドは、対話式プロンプトで入力できるものであればほとんど何でもかまいません。たとえば、順に実行される複数のコマンドも可能です。 UNIX ベースのプラットフォームでは、複数のコマンドは、セミコロン (;) を用いて分離します。 Windows NT では、複数のコマンドは、& 記号 (&) を用いて分離します。最後のコマンドの後には、区切り記号を使用しないでください。

以下の例は、**db2_all** コマンドを使用して、ノード構成ファイルで指定したすべてのデータベース区画でデータベース構成を変更する方法を示したものです。 ; 文字が二重引用符の内側にあるので、要求は同時に実行されます。

```
db2_all ";UPDATE DB CFG FOR sample USING LOGFILSIZ=100"
```

コマンドの説明

以下のコマンドを使用できます。

コマンド	説明
rah	すべてのマシンでコマンドを実行します。
db2_all	指定したすべてのデータベース区画サーバーでコマンドを実行します。
db2_kill	複数のデータベース区画サーバーで実行されているすべてのプロセスを突然停止し、すべてのデータベース区画サーバーのすべてのリソースを終結処理します。このコマンドは、データベースを不整合にします。このコマンドは IBM サービスからの指示による以外は発行しないでください。
db2_call_stack	UNIX ベースのプラットフォームでは、すべてのデータベース区画サーバーで実行されているすべてのプロセスが、呼び出しトレースバックを syslog に書き出すようにします。 Windows NT では、すべてのデータベース区画サーバーで実行されているすべてのプロセスが、呼び出しトレースバックをインスタンス・ディレクトリー内の P*xxxx*.*nnn* ファイルに書き出すようにします (P*xxxx* はプロセス ID、*nnn* はノード番号)。

UNIX ベースのプラットフォームでは、これらのコマンドは、次のような特定の暗黙的な設定で **rah** を実行します。

- すべてのマシンで並列に実行する。

- コマンド出力を /tmp/$USER/db2_kill、 /tmp/$USER/db2_call_stack にそれぞれバッファーする。

Windows NT では、これらのコマンドは **rah** を実行して、すべてのマシンで並列に実行します。

実行するコマンドの指定

次のようにコマンドを指定することができます。

- パラメーターとして、コマンド行から。
- 何もパラメーターを指定しないときは、プロンプトに対する応答として。

コマンドに次のような特殊文字が入っている場合は、プロンプト方式を使用する必要があります。

```
| & ; < > ( ) { } [ ] unsubstituted $
```

コマンド行でパラメーターとしてコマンドを指定するときに、上にリストしたような特殊文字のいずれかが含まれている場合は、二重引用符で囲む必要があります。

注: UNIX ベースのプラットフォームでは、コマンドは、ユーザーがプロンプトで入力したのと同じように、ユーザーのコマンド履歴に追加されます。

コマンドの中の特殊文字はすべて、正常に入力することができます (¥ 以外は引用符で囲まずに)。コマンドに ¥ を入れる必要があるときは、円記号を 2 つ (¥¥) 入力しなければなりません。

注: UNIX ベースのプラットフォームでは、Korn シェルを使用していない場合は、コマンドの中の特殊文字はすべて正常に入力することができます (", ¥、置換不能文字 $、および単一引用符 (') 以外は、引用符に入れずに)。 コマンドにこれらの文字のいずれかを入れる必要があるときは、円記号を 3 つ (¥¥¥) 前に置かなければなりません。たとえば、コマンドに ¥ を入れる必要があるときは、円記号を 4 つ (¥¥¥¥) 入力しなければなりません。

コマンドに 2 重引用符 (") を入れる必要があるときは、円記号を 3 つ前に付けて、たとえば、¥¥¥" のように入力しなければなりません。

注:

1. UNIX ベースのプラットフォームでは、ユーザーのコマンド・シェルでとくに単一引用ストリングの内側に単一引用符を入れる方法を何か用意していないかぎり、コマンドに単一引用符 (') を含めることはできません。
2. Windows NT では、ユーザーのコマンド・ウィンドウでとくに単一引用ストリングの内側に単一引用符を入れる方法を何か用意していないかぎり、コマンドに単一引用符 (') を含めることはできません。

stdin からバックグラウンドで読み取りを行うロジックを含む Korn シェルのシェル・スクリプトを実行する場合、 stdin をソースに明示的にリダイレクトする必要があります。そうすれば、プロセスは端末上で停止されることなく読み取りを行うことができます (SIGTTIN メッセージ)。 stdin をリダイレクトするには、指定されている入力がなければ次の形式のスクリプトを実行します。

```
shell_script </dev/null &
```

同様に、db2_all をバックグラウンドで実行する際には、常に </dev/null を指定する必要があります。たとえば、次のようにします。

```
db2_all ";run_this_command" </dev/null &
```

これを行うことにより、端末上で停止させなくても stdin をリダイレクトすることができます。

この代わりの方法は、リモート・コマンドからの出力に関心がなければ、次のように db2_all 接頭部で 『daemonize』 オプションを使用することです。

```
db2_all ";daemonize_this_command" &
```

UNIX ベース・プラットフォームでのコマンドの並列の実行

注: この節の情報は UNIX ベースのプラットフォームだけに適用されます。

デフォルトでは、コマンドはそれぞれのマシンで順次的に実行されますが、特定の接頭部シーケンスをコマンドの前につけることによって、バックグラウンド rshells を用いて、コマンドを並列に実行するよう指定することができます。 rshell がバックグラウンドで実行されている場合は、それぞれのコマンドは、リモート・マシンにあるバッファー・ファイルに出力を入れます。このプロセスでは、出力は次のように 2 つに分けて取り出されます。

1. リモート・コマンドが完了した後。
2. 何らかのプロセスがまだ実行されている場合は、あとで実行される可能性のある rshell が終了した後。

デフォルトにより、バッファー・ファイルの名前は /tmp/$USER/rahout ですが、環境変数 $RAHBUFDIR/$RAHBUFNAME によって名前を指定することができます。

複数のコマンドを同時に実行したいことを指定するときは (デフォルトによって)、このスクリプトは、すべてのホストに送信されるコマンドに追加のコマンドを接頭部として付加して、 $RAHBUFDIR と $RAHBUFNAME をバッファー・ファイルで使用できるかどうかチェックすることができます。これは、$RAHBUFDIR を作成します。これを抑止するときは、環境変数 RAHCHECKBUF=no をエクスポートします。ディレクトリーが存在していて、使用可能であることがわかっている場合は、このようにすると時間を節約することができます。

rah を使用して複数のマシンでコマンドを同時に実行する前に、以下のことを行ってください。

- それぞれのマシンごとに、使用しているユーザー ID 用のディレクトリー /tmp/$USER が存在することを確認する。このディレクトリーがまだ存在していない場合は、それを作成するために以下のものを実行してください。

  ```
  rah ")mkdir /tmp/$USER"
  ```

- 以下の行をユーザーの .kshrc (Korn シェル構文の場合) または .profile に追加して、現行のセッションにそれを入力する。

  ```
  export RAHCHECKBUF=no
  ```

- リモート・コマンドを実行するそれぞれのマシン ID が、 **rah** を実行する ID に対する該当の .rhosts ファイルの中に項目をもっていること、および **rah**; を実行する ID が、リモート・コマンドを実行するそれぞれのマシン ID に対する項目を該当の .rhosts ファイルの中にもっていることを確認する。

UNIX ベース・プラットフォームでの rah プロセスのモニター

注: この節の情報は UNIX ベースのプラットフォームだけに適用されます。
リモート・コマンドがまだ実行しているか、またはバッファー出力がまだ累積されている間は、 rah によって開始されたプロセスは、活動をモニターして、次のことを行います。

- どのコマンドが実行されなかったかを示すメッセージを端末に書き出す。
- バッファー出力を検索する。

環境変数 RAHWAITTIME によって制御されるインターバルで、通知メッセージが書き出されます。この指定方法の詳細については、ヘルプ情報を参照してください。通知メッセージはすべて、 RAHWAITTIME=0 をエクスポートすることによって完全に抑止することができます。

1 次モニター・プロセスは、 **rahwaitfor** という名前のコマンド (ps コマンドで示される) です。 最初の通知メッセージで、このプロセスの pid (プロセス ID) が示されます。その他のすべてのモニター・プロセスは、 **rah** スクリプト (またはシンボリック・リンクの名前) を実行する **ksh** コマンドとして表示されます。必要であれば、次のコマンドによって、すべてのモニター・プロセスを停止することができます。

```
kill <pid>
```

ここで、<pid> は、1 次モニター・プロセスのプロセス ID です。シグナル番号を指定してはなりません。デフォルトである 15 のままにしてください。これは、リモート・コマンドにはまったく影響しませんが、バッファー出力の自動的な表示をしないようにします。 **rah** の 1 回の実行が生きている間に、 2 つ以上の異なるモニター・プロセスのセットが、異なる時点で実行される可能性があることに注意してください。しかし、どの時点でも現行のセットを停止すると、その後ではもう開始されません。

レギュラーのログイン・シェルが Korn シェルでない場合 (たとえば、 /bin/ksh) は、**rah** を使用することができますが、次のような特殊文字が含まれるコマンドを入力する方法に関する規則に、いくらか相違があります。

```
"  unsubstituted $ '
```

さらに詳しいことは、rah "?" を入力してください。また、UNIX ベースの環境では、リモート・コマンドを実行する ID にあるログイン・シェルが Korn シェルでないときは、 **rah** を実行する ID にあるログイン・シェルも Korn シェルであってはなりません。 (**rah** は、リモート ID のシェルが、ローカル ID にもとづく Korn シェルであるかどうかに関する判別を行います。) シェルは、単一引用符で囲まれたストリングについて、置換または特別な処理を行ってはなりません。現状のままにしておいてください。

その他の rah (Run All Hosts) 情報 (Solaris および AIX のみ)

パフォーマンスを向上させるために、 rah は大規模なシステムで tree_logic を使うように拡張されています。つまり、rah はリストに含まれるノード数を検査し、その数がしきい値を超過するのであれば、リストのサブセットを作成して、それ自体の再帰的呼び出しをそれぞれのノードに送信します。それぞれのノードでは、再帰的に呼び出された rah は前述の同じ論理に従います。これは、リストが十分に小さくなり、「リスト上のすべてのノードにコマンドを送信する」という標準的な論理 (ここでは ″ツリーのリーフ″ という論理) に従えるようになるまで続きます。このときのしきい値は、環境変数 RAHTREETHRESH で指定できます。これを指定しないと、デフォルトの 15 になります。

物理ノードに対して複数の論理ノードが存在するシステムの場合は、 db2_all は再帰的な呼び出しをそれぞれの物理ノードに送信してから、その同じ物理ノード上の他の論理ノードに rsh するので、物理ノード間の通信量も少なくなります。 (この点は、db2_all だけに当てはまるもので rah には当てはまりません。 rah は常にそれぞれの物理ノードだけに送信します。)

接頭部シーケンス

接頭部シーケンスは、1 桁以上の特殊文字です。コマンドの文字の直前に、ブランクをあけずに 1 つ以上の接頭部シーケンスを入力してください。シーケンスを 2 つ以上指定したい場合は、任意の順序でタイプすることができますが、複数文字のシーケンスの中にある文字は、順番に入力する必要があります。接頭部シーケンスを入力するときは、次の例に示すように、接頭部シーケンスも含めてコマンド全体を二重引用符で囲んでください。

- UNIX ベースのプラットフォームでは、以下のようにします。

  ```
  rah "};ps -F pid,ppid,etime,args -u $USER"
  ```

- Windows NT では、以下のようにします。

  ```
  rah "||db2 get db cfg for sample"
  ```

接頭部シーケンスは次のとおりです。

シーケンス	目的
|	バックグラウンドでコマンドを順に実行します。
|&	バックグラウンドで順番にコマンドを実行し、さらにすべてのリモート・コマンドが完了したあとで、まだいくつかのプロセスが実行中であっても、コマンドを終了します。たとえば、子プロセス (UNIX ベースのプラットフォームの場合) またはバックグラウンド・プロセス (Windows NT の場合) がまだ実行中であれば、もっと後になる可能性があります。このケースでは、コマンドは、別個のバックグラウンド・プロセスを開始して、コマンド終了後に生成されたリモート出力を検索し、もとのマシンに書き戻します。 **注:** UNIX ベースのプラットフォームでは、**&** を指定すると、さらに **rsh** コマンドが必要になるので、パフォーマンスが低下します。
||	バックグラウンドでコマンドを並列に実行します。
||&	バックグラウンドで並列にコマンドを実行し、さらにすべてのリモート・コマンドが完了した後で、上記の **|&** のケースで述べたようにしてコマンドを終了します。 **注:** UNIX ベースのプラットフォームでは、**&** を指定すると、さらに **rsh** コマンドが必要になるので、パフォーマンスが低下します。
;	上記の **||&** と同じ。これは、代替短縮形式です。 **注:** ; を指定すると、 **rsh** コマンドがさらに必要になるので、 **||** にくらべてパフォーマンスが低下します。
]	コマンドを実行する前に、ユーザーのプロファイルのドット実行 (dot-execution(..)) を付加します。 **注:** UNIX ベースのプラットフォームに限り使用できます。
}	コマンドを実行する前に、 $RAHENV (多分 **.kshrc**) で指名されたファイルのドット実行を付加します。 **注:** UNIX ベースのプラットフォームに限り使用できます。
]}	コマンドを実行する前に、ユーザーのプロファイルのドット実行を付加し、その後で、 $RAHENV (多分 **.kshrc**) で指名されたファイルを実行します。 **注:** UNIX ベースのプラットフォームに限り使用できます。
)	ユーザーのプロファイルおよび $RAHENV で指名されたファイルの実行を抑止します。

注: UNIX ベースのプラットフォームに限り使用できます。

' コマンドの起動をマシンにエコーします。

< このマシン以外のすべてのマシンに送信します。

<<-nnn< *nnn* 以外のすべてのデータベース区画サーバー (ノード番号 *nnn* を除いて db2nodes.cfg にあるすべてのデータベース区画サーバー。この表の最後の接頭部文字の後の最初の段落を参照してください) に送信します。

<<+nnn< データベース区画サーバー *nnn* (ノード番号が *nnn* の db2nodes.cfg にあるデータベース区画サーバー。この表の最後の接頭部文字の後の最初の段落を参照してください) だけに送信します。

(ブランク文字)

リモート・コマンドを、stdin、 stdout および stderr をすべてクローズしてバックグラウンドで実行します。このオプションは、バックグラウンドでコマンドを実行するときだけ、つまり ¥ または ; も含んでいる接頭部シーケンスの中でだけ有効です。このようにすると、コマンドは非常に早く完了することができます (リモート・コマンドが開始されるとすぐに)。この接頭部文字を **rah** コマンド行で指定する場合は、コマンドを単一引用符で囲むか、またはコマンドを 2 重引用符で囲んでから接頭部文字の前に ¥ を置きます。たとえば、次のようにします。

```
rah '; mydaemon'
```

または

```
rah ";¥ mydaemon"
```

rah コマンドは、バックグラウンド・プロセスとして実行されるとき、出力が戻されるのを待ちません。

> <> のオカレンスをマシン名と置換します。

" () のオカレンスをマシン索引と置換し、 ## のオカレンスをノード番号と置換します。

注:

1. マシン索引とは、データベース・システムのマシンに関連した番号のことです。複数の論理ノードを実行している場合は、マシンのマシン索引は、ノード構成ファイル内のそのマシンのノード番号に対応します。複数の論理ノードを実行しているマシンの場合は、項目数もそのノードの数になるので、このようなマシンのマシン索引を取得するには、その重複項目をカウントしないでください。たとえば MACH1 と MACH2 の両方とも 2 つの論理ノードを実行して

いる場合は、ノード構成ファイル内の MACH3 のノード番号は 5 になります。しかし、MACH3 のマシン索引は 3 になります。

Windows NT の場合、ノード構成ファイルを編集しないでください。マシン索引を取得するには、 **db2nlist** コマンドを使用してください。詳細は、*DB2 エンタープライズ拡張エディション (Windows 版) 概説およびインストール* の資料を参照してください。

2. " が指定されている場合は、重複はマシンのリストから除去されません。重複を除去するには、『マシン・リストからの重複項目の除去』を参照してください。

<<-nnn< と <<+nnn< 接頭部シーケンスを使用しているとき、 *nnn* は 1、2 または 3 桁の任意の区分番号にすることができますが、これは、 db2nodes.cfg ファイルの *nodenum* 値と一致している必要があります。

注: 接頭部シーケンスは、コマンドの一部とみなされます。接頭部シーケンスをコマンドの一部として指定するときは、接頭部シーケンスも含めてコマンド全体を 2 重引用符で囲んでください。

マシンのリストの指定

デフォルトにより、マシンのリストは、ノード構成ファイル db2nodes.cfg からとられます。以下のようにして指定変更することができます。

- 環境変数 RAHOSTFILE をエクスポート (UNIX ベースのプラットフォームの場合) または設定 (Windows NT の場合) することによって、マシンのリストが含まれるファイルのパス名を指定する。
- 環境変数 RAHOSTLIST をエクスポート (UNIX ベースのプラットフォームの場合) または設定 (Windows NT の場合) することによって、リストを明示的に、名前のストリングとしてスペースで区切って指定する。

  注: これらの環境変数が両方とも指定されている場合は、 RAHOSTLIST の方が優先します。

注: Windows NT の場合、ノード構成ファイル内で不整合が生じないように、このファイルを編集しないでください。インスタンスにあるマシンのリストを取得するには、 **db2nlist** コマンドを使用してください。詳細は、*DB2 エンタープライズ拡張エディション (Windows 版) 概説およびインストール* の資料を参照してください。

マシン・リストからの重複項目の除去

1 つのマシンで複数の論理ノード (データベース区画サーバー) を用いて、 DB2 エンタープライズ拡張エディションを実行している場合は、ユーザーの db2nodes.cfg にはそのマシンに対する複数項目が含まれることになります。この状態では、**rah** コマンド

は、ユーザーが、各マシンで 1 回だけ、あるいは、 db2nodes.cfg ファイルの中にリストされている各論理ノードについて 1 回、コマンドを実行したいかどうかを知る必要があります。 **rah** コマンドを用いて、マシンを指定します。 **db2_all** コマンドを使用して、論理ノードを指定します。

注: UNIX ベースのプラットフォームの場合、ユーザーがマシンを指定すると、 **rah** は通常、重複をマシン・リストから除去しますが、以下のものは例外です。論理ノードを指定している場合は、 **db2_all** は、ユーザーのコマンドに、以下の指定を前に付加します。

```
export DB2NODE=nnn  (Korn シェルの場合)
```

ここで、*nnn* は、目的のデータベース区画サーバーにコマンドが経路指定されるように、 db2nodes.cfg ファイルの中の対応する行からとられたノード番号です。

論理ノードを指定するときは、 <<-nnn< および <<+nnn< 接頭部シーケンスを用いて、1 つを除くすべての論理ノードが含まれるようにリストを限定するか、または、 1 つのデータベース区画サーバーだけを指定することができます。まずカタログ・ノードで最初にコマンドを実行して、それが完了したときに、他のすべてのデータベース区画サーバーで並列に同じコマンドを実行したいという場合に、上記のようにする必要があります。これは、通常、**db2 restart database** コマンドを実行するときに、必要になります。このために、カタログ・ノードのノード番号を知っている必要があります。接頭部シーケンスに関する情報は、372ページの『接頭部シーケンス』を参照してください。

rah コマンドを使用して **db2 restart database** を実行するときは、重複項目はマシンのリストから除去されます。しかし、" 接頭部を指定している場合は、" 接頭部を使用することが、各マシンではなく各データベース区画サーバーに送ることを意味しているため、重複は除去されません。

rah コマンドの制御

rah コマンドを制御するために以下の環境変数を使用することができます。

表 23.

名前	意味	デフォルト
$RAHBUFDIR **注:** UNIX ベースのプラットフォームに限り使用できます。	バッファーのディレクトリー	/tmp/$USER
$RAHBUFNAME **注:** UNIX ベースのプラットフォームに限り使用できます。	バッファーのファイル名	rahout

表 23. (続き)

名前	意味	デフォルト
$RAHOSTFILE (UNIX ベースのプラットフォームの場合)、RAHOSTFILE (Windows NT の場合)	ホストのリストが入っているファイル	db2nodes.cfg
$RAHOSTLIST (UNIX ベースのプラットフォームの場合)、RAHOSTLIST (Windows NT の場合)	ストリングとしてのホストのリスト	$RAHOSTFILE から抽出
$RAHCHECKBUF **注:** UNIX ベースのプラットフォームに限り使用できます。	″no″ に設定されていると、チェックはバイパスします。	設定されない
$RAHSLEEPTIME (UNIX ベースのプラットフォームの場合)、RAHSLEEPTIME (Windows NT の場合)	並列に実行されるコマンドからの最初の出力をこのスクリプトが待つ時間 (秒数)。	**db2_kill** の場合は 86400 秒、すべての他の場合 200 秒。

表 23. (続き)

名前	意味	デフォルト
$RAHWAITTIME (UNIX ベースのプラットフォームの場合)、RAHWAITTIME (Windows NT の場合)	Windows NT の場合、リモート・ジョブがまだ実行されていることを連続チェックする間隔の秒数 UNIX ベースのプラットフォームの場合、リモート・ジョブがまだ実行されていることを連続チェックしてから、`rah: waiting for <pid> ...` メッセージまでの間隔の秒数 プラットフォームに関係なく、正の整数を指定します。メッセージを抑止するには先行ゼロを値の前に付けます。たとえば、RAHWAITTIME=045 を指定してエクスポートします。 **rah** は、ジョブの完了を検知するためにこれらのチェックには依存しないので、低い値を指定する必要はありません。	45 秒
$RAHENV **注:** UNIX ベースのプラットフォームに限り使用できます。	$RAHDOTFILES=E または K または PE または B の場合に実行されるファイル名を指定します。	$ENV
$RAHUSER (UNIX ベースのプラットフォームの場合)、RAHUSER (Windows NT の場合)	UNIX ベースのプラットフォームの場合、リモート・コマンドがその下で実行されるユーザー ID Windows NT の場合、DB2 リモート・コマンド・サービスに関連したログオン・アカウント	$USER

注: UNIX ベースのプラットフォームの場合、リモート・シェルでセットされる値 (あれば) ではなく、 **rah** が実行される $RAHENV の値が使用されます。

UNIX ベース・プラットフォームの場合の $RAHDOTFILES

注: この節の情報は UNIX ベースのプラットフォームだけに適用されます。
接頭部シーケンスが指定されていない場合に実行される .files は、次のとおりです。

P .profile

E $RAHENV で指名されたファイル (多くの場合 .kshrc)

K E と同じ

PE $RAHENV で指名されたファイル (多くの場合 .kshrc) が後に続く .profile

B PE と同じ

N なし (またはどちらでもない)

注: ユーザーのログイン・シェルが Korn シェルでない場合は、実行を指定した任意のドット・ファイルは Korn シェル・プロセスで実行されるので、 Korn シェル構文にしたがっている必要があります。したがって、たとえば、ログイン・シェルが C シェルであるとすれば、 **rah** によって実行されるコマンド用にユーザーの .cshrc 環境をセットアップするために、その .cshrc に相応する Korn シェル INSTHOME/.profile を作成して、 INSTHOME/.cshrc の中で指定する必要があります。

```
setenv RAHDOTFILES P
```

あるいは、ユーザーの .cshrc に相応する Korn シェル INSTHOME/.kshrc を作成して、 INSTHOME/.cshrc の中で指定する必要があります。

```
setenv RAHDOTFILES E
setenv RAHENV INSTHOME/.kshrc
```

また、tty がないときは (**rsh** で起動されたときなど)、ユーザーの .cshrc は stdout に書き出さないことも重要です。 stdout に書き出す行を、たとえば、次のように囲むことによって、このことを確認することができます。

```
if { tty -s } then echo "executed .cshrc";
endif
```

Windows NT の場合のデフォルト環境プロファイルの設定

注: この節の情報は Windows NT だけに適用されます。

rah コマンドのデフォルト環境プロファイルを設定するには、 db2rah.env ファイルを使用します。これはインスタンス・ディレクトリー内に作成する必要があります。ファイルは以下の形式にする必要があります。

```
; This is a comment line
DB2INSTANCE=instancename
DB2DBDFT=database
; End of file
```

rah の環境を初期設定するのに必要な環境変数をすべて指定できます。

UNIX ベース・プラットフォームの場合の rah に関する問題の判別

注: この節の情報は UNIX ベースのプラットフォームだけに適用されます。
rah を実行しているときに検出される可能性がある問題の処理方法についての提案を、以下に示します。

1. **rah** がハングしている (または非常に長時間かかっている)。

   この問題は、下記のことが原因とみられます。

   - **rah** は、出力をバッファーする必要があると判断したが、ユーザーが、RAHCHECKBUF=no をエクスポートしなかったので、そのコマンドを実行する前に、**rah** は、コマンドをすべてのマシンに送って、バッファー・ディレクトリーの存在をチェックし、存在しなければバッファー・ディレクトリーを作成した。
   - ユーザーがコマンドを送っているマシンのうちのいくつかが、応答していない。**rsh** コマンドは最終的にタイムアウトになりますが、タイムアウトの間隔は極めて長く、通常は約 60 秒です。

2. ユーザーが次のようなメッセージを受け取った。

   - ログインの誤り
   - 許可が否定された

   マシンのいずれかが、その .hosts ファイルの中で正しく定義された **rah** を実行する ID をもっていないか、または **rah** を実行する ID が、 .rhosts で正しく定義されたホストのいずれかをもっていない、のどちらかです。

3. バックグラウンドの rshells を使用して並列にコマンドを実行しているときに、コマンドは、マシンで予期された経過時間内に実行され、完了するが、 **rah** が、このことに気付いて、シェル・プロンプトを準備するのに時間がかかっている。

   rah を実行している ID が、その .rhosts ファイルの中で正しく定義されたいずれかのマシンをもっていません。

4. **rah** は、シェル・コマンド行から実行されたときは正しく実行されますが、たとえば次のように、rsh をリモートで使用して **rah** を実行すると、

   ```
   rsh somewher -l $USER db2_kill
   ```

   rah は完了しません。

   これは正常です。 **rah** は、それが終了したあと実行を続ける、バックグラウンドのモニター・プロセスを開始します。これらのプロセスは、実行したコマンドに関連するすべてのプロセスが自ら終了するまで、正常に続けられます。 **db2_kill** の場合は、これは、すべてのデータベース・マネージャーの終了を意味します。そのコマンドが **rahwaitfor** および kill <process_id> であるようなプロセスを探すことによって、モニター・プロセスを終了することができます。 シグナル番号を指定してはなりません。代わりに、デフォルトの (15) にしておきます。

5. 同じ $RAHUSER の下で **rah** の複数のコマンドが出されたときに、 **rah** からの出力が正しく表示されないか、または **rah** が、 $RAHBUFNAME が存在しないことを誤って報告している。

   これは、複数の **rah** が同時に実行して、出力のバッファリング用として同じバッファー・ファイル ($RAHBUFDIR/$RAHBUFNAME など) を使用しようとしているためです。このような問題を避けるために、同時に実行される **rah** コマンドのそれぞれについて、別の $RAHBUFNAME を使用してください。たとえば、以下の ksh では、

   ```
   export RAHBUFNAME=rahout
   rah ";$command_1" &
   export RAHBUFNAME=rah2out
   rah ";$command_2" &
   ```

   のようにするか、または次のようにして、一意の名前をシェルに自動的に選択させるようにします。

   ```
   RAHBUFNAME=rahout.$$ db2_all ".....
   ```

   どのような方法を使用する場合も、ディスク・スペースに制約があるならば、何らかのポイントでバッファー・ファイルを確実に終結処理する必要があります。 **rah** は、実行の終わりにバッファー・ファイルを消去することはしませんが、次にユーザーが同じバッファー・ファイルを指定したときには既存のファイルを消去して再使用します。

6. 次のように入力して、

   ```
   rah '"print from ()'
   ```

   以下のメッセージを受け取った。

   ```
   ksh: syntax error at line 1 : (' unexpected
   ```

   () および ## の置換の前提条件は次のとおりです。

   - **rah** ではなく、 **db2_all** を使用する。
   - RAHOSTFILE をエクスポートするか、またはデフォルトである /sqllib/db2nodes.cfg ファイルにすることによって、 RAHOSTFILE が使用されていることを確認する。このような前提条件がない場合は、**rah** は、 () と ## を現状のままにします。コマンド **print from ()** は有効でないので、エラーになります。

   コマンドを並列実行している場合、パフォーマンスのヒントとして、 & から与えられる機能が本当に必要でない限り、|& よりは | を、 ||& または ; よりは || を使用してください。 & を指定すると、**rsh** コマンドが必要になり、パフォーマンスが低下するためです。

付録E. DB2 (Windows NT 版) が Windows NT 機密保護を処理する方法

Windows NT をインストールすると、2 つの管理者ユーザー名を作成できます。

- 1 つは、『Administrator』と呼ばれます。
- もう一方の名前は選択できます。この名前には、管理者権限がなければならず、DB2 の命名規則にも準拠していなければなりません。 DB2 の命名規則についての詳細は、329ページの『付録A. 命名規則』を参照してください。

ユーザーはローカル・マシンにログオンすることができます。あるいは、Windows NT ドメイン中にマシンをインストールしているならば、ユーザーはそのドメインにログオンできます。 DB2 (Windows NT 版) はこれら両方の機能をサポートします。ユーザーを認証するために、 DB2 は最初にローカル・マシンのリスト、次に現在のドメインのドメイン・コントローラー、最後にドメイン・コントローラーを認識する承認されたドメインをチェックします。

この動作の仕方を説明するために、DB2 インスタンスがサーバー認証を必要とすると仮定します。構成は、以下のとおりです。

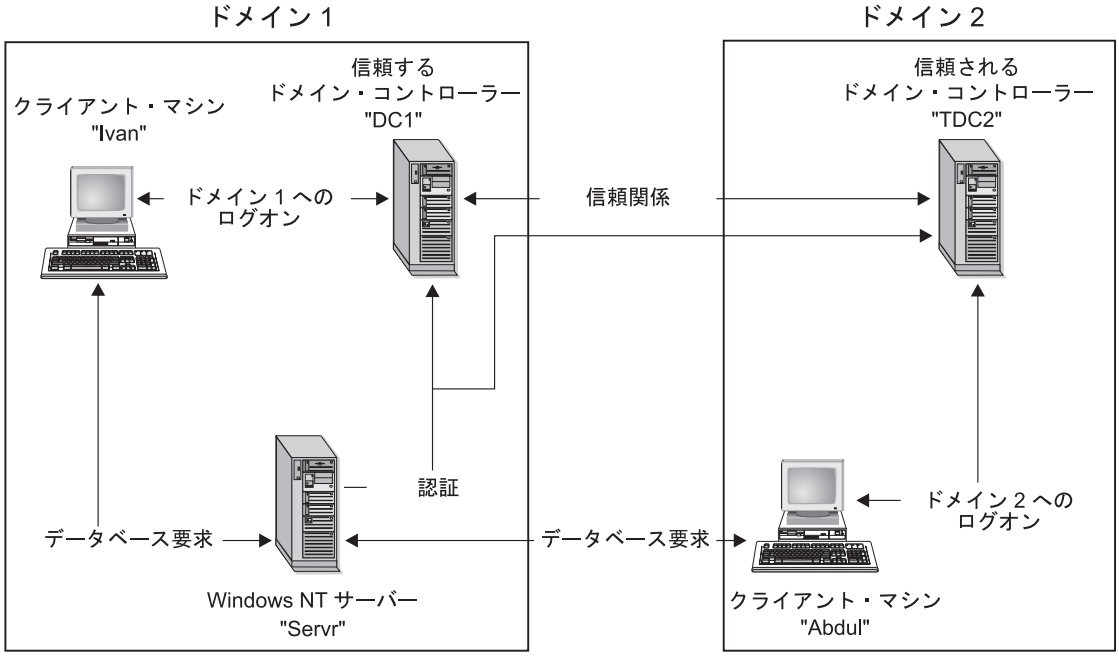


図 11. Windows NT のドメインを使用した認証

各マシンには、クライアント・マシンが Windows 9x を実行していない限り、機密保護データベース (機密保護アクセス管理 (SAM)) があります。 Windows 9x マシンには、SAM データベースはありません。 DC1 はドメイン・コントローラーで、そのクライアント・マシンの Ivan、 DB2 (Windows NT 版) のサーバーの Servr が登録されています。 TDC2 は DC1 に承認されたドメインで、クライアント・マシンの Abdul は TDC2 のドメインのメンバーです。

サーバー認証のサンプル事例

1. Abdul は、TDC2 ドメインにログオンします (つまり、 TDC2 SAM データベースに認識されています)。
2. 次に Abdul は、次のように入力して SRV3 上に常駐するためにカタログされた DB2 データベースに接続します。

   ```
   db2 connect to remotedb user Abdul using fredpw
   ```

3. SRV3 は、Abdul が認識されている場所を判別します。この情報を検出するのに使われる API は、最初にローカル・マシン (SRV3)、次にドメイン・コントローラー (DC1) を検索し、最後に承認されたドメインを検索しようとします。ユーザー名 Abdul が TDC2 上で検出されます。この検索順序には、ユーザーとグループに単一ネームスペースが必要です。
4. 次に SRV3 は、次のように実行します。

a. TDC2 を使ってユーザー名とパスワードの妥当性を検査します。
b. TDC2 に照会することによって、Abdul が管理者かどうかを検出します。
c. TDC2 に照会することによって、 Abdul のグループすべてを列挙します。

クライアント認証と Windows NT クライアント・マシンのサンプル事例

1. 管理者の Dale は、SRV3 にログオンし、クライアントに対するデータベース・インスタンスの認証を変更します。

```
db2 update dbm cfg using authentication client
db2stop myinst
db2start myinst
```

2. Windows クライアント・マシンで、Ivan は DC1 ドメインにログオンします (つまり、 DC1 SAM データベースに認識されています)。
3. 次に Ivan は、次のように入力して SRV3 上に常駐するためにカタログされた DB2 データベースに接続します。

```
DB2 CONNECT to remotedb user Ivan using johnpw
```

4. Ivan のマシンは、ユーザー名とパスワードの妥当性を検査します。この情報を検出するのに使われる API は、最初にローカル・マシン (Ivan)、次にドメイン・コントローラー (DC1) を検索し、最後に承認されたドメインを検索しようとします。ユーザー名 Ivan が DC1 上で検出されます。
5. 次に Ivan のマシンは、DC1 を使ってユーザー名とパスワードの妥当性を検査します。
6. 次に SRV3 は、次のように実行します。
   a. Ivan が認識された場所を判別します。
   b. DC1 に照会することによって、Ivan は管理者かどうかを検出します。
   c. DC1 に照会することによって、Ivan のすべてのグループを列挙します。

注: DB2 データベースに接続してみる前に、必ず DB2 機密保護サービスを始動してください。この機密保護サービスは DB2 によってインストールされ、 Windows NT のサービスとして実行できるように設定されます。ただし、自動的に始動することはできません。 DB2 機密保護サービスを始動するには、 `NET START DB2NTSECSERVER` コマンドを入力してください。

クライアント認証と Windows 95 クライアント・マシンのサンプル事例

1. 管理者の Dale は、SRV3 にログオンし、クライアントに対するデータベース・インスタンスの認証を変更します。

```
db2 update dbm cfg using authentication client
db2stop myinst
db2start myinst
```

2. Windows 95 マシンで、Ivan は DC1 ドメインにログオンします (つまり、 DC1 SAM データベースに認識されています)。

3. 次に Ivan は、次のように入力して SRV3 上に常駐するためにカタログされた DB2 データベースに接続します。

   ```
   db2 connect to remotedb user Ivan using johnpw
   ```

4. Ivan の Windows 95 マシンはユーザー名とパスワードの妥当性を検査することはできません。したがって、ユーザー名とパスワードは有効とみなされます。
5. 次に SRV3 は、次のように実行します。
   a. Ivan が認識された場所を判別します。
   b. DC1 に照会することによって、Ivan は管理者かどうかを検出します。
   c. DC1 に照会することによって、Ivan のすべてのグループを列挙します。

注: Windows 95 クライアントは、与えられたユーザー名とパスワードの妥当性を検査できないため、 Windows 95 でのクライアント認証は本質的に不確実です。 Windows 95 マシンが Windows NT 機密保護プロバイダーにアクセスした場合、妥当性検査されたパススルー・ログオンのために Windows 95 システムを構成することによって、機密保護のいくつかの基準を課することができます。この方法で Windows 95 システムを構成する方法についての詳細は、 Microsoft 社の Windows 95 の資料を参照してください。

DB2 はグローバル・グループもサポートします。グローバル・グループを使用するために、機密保護サーバー上にあるローカル・グループ内にグローバル・グループを組み込む必要があります。 DB2 が、ある人がメンバーになっているグループをすべて列挙するとき、 (1 つまたは複数のローカル・グループのメンバーになっているグローバル・グループ中に、そのグループ自体があるために) そのユーザーが間接的にメンバーになっているローカル・グループもリストします。

DB2 でのバックアップ・ドメイン・コントローラーの使用

また、 DB2 用に使用しているサーバーがバックアップ・ドメイン・コントローラーとしても動作する場合、 DB2 を構成してバックアップ・ドメイン・コントローラーを使用すれば、 DB2 のパフォーマンスを向上させ、ネットワーク通信量を削減することができます。

DB2DMNBCKCTLR レジストリー変数を設定することによって、 DB2 にバックアップ・ドメイン・コントローラーを指定します。

DB2 サーバーがバックアップ・ドメイン・コントローラーになっている場合、そのドメイン名が分かれば、次を使用します。

```
db2dmnbckctlr=DOMAIN_NAME
```

この場合、*DOMAIN_NAME* は大文字でなければなりません。

ローカル・マシンがバックアップ・ドメイン・コントローラーになっている場合、 DB2 がそのドメインを判別していれば、次を使用します。

```
DB2DMNBCKCTLR=?
```

注: デフォルトでは、DB2 はバックアップ・ドメイン・コントローラーを使用しません。バックアップ・ドメイン・コントローラーはプライマリー・ドメイン・コントローラーと同期しないことがあり、機密漏れを生じることがあるためです。プライマリー・ドメイン・コントローラーの機密保護データベースが更新されても、その変更内容がバックアップ・ドメイン・コントローラーに伝搬していない場合に、ドメイン・コントローラーが同期しなくなることがあります。この事態は、ネットワーク待ち時間が生じた場合や、コンピューターのブラウザー・サービスが作動可能でない場合に起こることがあります。

DB2 (Windows NT 版) での DB2 ユーザー認証

Windows NT ユーザーの場合、オペレーティング・システムの認証の方法が原因で、ユーザーの認証に問題の生じることがあります。この節では、 DB2 (Windows NT 版) でのユーザー認証に関する考慮事項について取り上げます。

- 『ユーザー名とグループ名の制約』
- 『DB2 (Windows NT 版) 機密保護サービス』
- 388ページの『DB2 のバックアップ・ドメイン・コントローラーへのインストール』
- 389ページの『グループおよびドメイン機密保護を使った認証』

ユーザー名とグループ名の制約

Windows NT 環境では、次のような制約があります。

- DB2 の内部では、ユーザー名は 30 文字以下に制限されています。グループ名は 8 文字以下に制限されています。
- Windows NT 環境では、ユーザー名に大文字小文字の区別はありません。ただし、パスワードには大文字小文字の区別があります。
- ユーザー名やグループ名には大文字と小文字の両方を含めることができます。ただし、DB2 内で使用されるときには大文字に変換されるのが普通です。たとえば、データベースに接続してから表 schema1.table1 を作成した場合、この表はデータベース内に SCHEMA1.TABLE1 として保管されます。 (小文字のオブジェクト名を使用したい場合は、コマンド行プロセッサーからコマンドを発行するときにオブジェクト名を引用符で囲むか、あるいはサード・パーティーの ODBC フロントエンド・ツールを使用します。)

DB2 (Windows NT 版) 機密保護サービス

DB2 ユニバーサル・データベースでは、ユーザー名とパスワードの認証が DB2 システム・コントローラーに統合されています。機密保護サービスが必要になるのは、認証 CLIENT 用に構成されたサーバーにクライアントが接続するときだけです。

DB2 のバックアップ・ドメイン・コントローラーへのインストール

Windows NT 環境では、ユーザーの認証をプライマリー・コントローラーとバックアップ・コントローラーのどちらでも行えます。この機能は、どのサイトにも 1 つの中央プライマリー・ドメイン・コントローラー (PDC) と、 1 つまたは複数のバックアップ・ドメイン・コントローラー (BDC) とが配置されているような大規模な分散 LAN では非常に重要です。ユーザーは認証のためにプライマリー・ドメイン・コントローラーを呼び出さなくても、現在のサイトにあるバックアップ・ドメイン・コントローラーで認証を行えます。

この場合、バックアップ・ドメイン・コントローラーを設けることの利点は、ユーザーの認証を短時間で行えることと、 BDC がない場合よりも LAN が混雑しないで済むということです。

以下の条件にあてはまる場合、認証は BDC で行うことができます。

- DB2 (Windows NT 版) サーバーがバックアップ・ドメイン・コントローラーにインストールされている場合。
- DB2DMNBCKCTLR プロファイル・レジストリー変数が正しく設定されている場合。

DB2DMNBCKCTLR プロファイル・レジストリー変数が設定されていない場合、あるいはブランクに設定されている場合、 DB2 (Windows NT 版) は認証をプライマリー・ドメイン・コントローラーで実行します。

DB2DMNBCKCTLR に有効な宣言設定は『?』またはドメイン名だけです。

DB2DMNBCKCTLR プロファイル・レジストリー変数が疑問符に設定されていて (DB2DMNBCKCTLR=?)、かつ以下の条件にあてはまる場合、 DB2 (Windows NT 版) は認証をバックアップ・ドメイン・コントローラーで実行します。

- cachedPrimaryDomain レジストリー値が、このマシンが属しているドメインの名前に設定されている。 (この設定を調べるには、**「HKEY_LOCAL_MACHINE-> ソフトウェア-> Microsoft-> Windows NT-> 現行バージョン-> WinLogon (HKEY_LOCAL_MACHINE—> Software—> Microsoft—> Windows NT—> Current Version—> WinLogon)」**を選択します。)
- サーバー・マネージャーによって、バックアップ・ドメイン・コントローラーが活動状態かつ使用可能であることが示されている。 (つまり、このマシンを表すアイコンがグレー表示にはなっていない。)
- DB2 Windows NT サーバーのレジストリーによって、そのシステムが指定されたドメイン上のバックアップ・ドメイン・コントローラーであることが示されている。

通常の状態であれば DB2DMNBCKCTLR=? 設定はたいてい正常に機能しますが、あらゆる環境で正常に機能することが保証されているわけではありません。ドメイン上のサーバーについて提供される情報は動的であるため、コンピューター・ブラウザーを実行して、この情報を常に正確かつ最新のものに保つ必要があります。大規模な LAN では

コンピューター・ブラウザーを実行できないことがあるため、サーバー・マネージャーの情報が最新のものではなくなる場合もあります。この場合、 DB2 (Windows NT 版) に認証をバックアップ・ドメイン・コントローラーで実行させる別の方法があります。それは、DB2DMNBCKCTLR=*xxx* を設定することです (*xxx* は DB2 サーバーの Windows NT ドメイン名)。この設定がなされている場合、以下の条件を満たしていれば、認証がバックアップ・ドメイン・コントローラーで行われます。

- cachedPrimaryDomain レジストリー値が、このマシンが属しているドメインの名前に設定されている。 (この設定を調べるには、**「HKEY_LOCAL_MACHINE-> ソフトウェア-> Microsoft-> Windows NT-> 現行バージョン-> WinLogon (HKEY_LOCAL_MACHINE—> Software—> Microsoft—> Windows NT—> Current Version—> WinLogon)」**を選択します。)
- マシンが、指定されたドメイン用のバックアップ・ドメイン・コントローラーとして構成されている。 (マシンがバックアップ・ドメイン・コントローラーとしてセットアップされていても、他のドメインのためのものである場合、この設定はエラーになります。)

グループおよびドメイン機密保護を使った認証

DB2 (Windows NT 版) は、以下のタイプのグループをサポートします。

- ローカル・グループ
- グローバル・グループ
- ローカル・グループのメンバーとしてのグローバル・グループ

  DB2 (Windows NT 版) は、ユーザーの情報が含まれているデータベースを使用して、そのユーザーがメンバーとなっているローカル・グループとグローバル・グループを列挙します。 DB2 ユニバーサル・データベースは、ユーザー・アカウントがどこで見つかったかに関係なく、確実に、 DB2 がインストールされているローカルの Windows NT サーバーでグループが列挙されるようにオーバーライドを行います。このオーバーライドを実現するには、以下のコマンドを使用します。

  – グローバル設定の場合:

  ```
  db2set -g DB2_GRP_LOOKUP=local
  ```

  – インスタンス設定の場合:

  ```
  db2set -i DB2_GRP_LOOKUP=local
  ```

  設定されているすべての DB2 プロファイル・レジストリー変数を表示するには、次のように入力します。

  ```
  db2set -all
  ```

DB2 (Windows NT 版) でドメイン機密保護を有効にするためには、権限と特権をローカル・グループに授与する必要があります。ローカル・グループ内のユーザー名とグローバル・グループ内のユーザー名は、ローカル・グループまたはグローバル・グループと同じドメインで定義しないと正常に認証を行えません。

DB2_GRP_LOOKUP プロファイル・レジストリー変数がローカルに設定されている場合、 DB2 はローカル・マシンでしかユーザーを探そうとしません。そのユーザーがローカル・マシンで見つからなかった場合や、そのユーザーがローカルまたはグローバル・グループのメンバーとして定義されていない場合、認証は失敗します。 DB2 は、同じドメインの他のマシンやドメイン・コントローラーからユーザーを探そうとは**しません**。

DB2_GRP_LOOKUP プロファイル・レジストリー変数が設定されていない場合、以下のことが行われます。

1. DB2 は最初に同じマシンからユーザーを探そうとします。
2. ユーザー名がローカルで定義されている場合、そのユーザーの認証はローカルで行われます。
3. ユーザーがローカルで見つからなかった場合、 DB2 は同じドメインからユーザー名を探そうとし、それでも見つからない場合は、信頼されている他のドメインから探そうとします。

以下の例は、 DB2 (Windows NT 版) がドメイン機密保護をどのようにサポートするかを示しています。最初の例では、ユーザー名とローカル・グループが同じドメイン上にあるため、接続は機能します。 2 番目の例では、ユーザー名とローカルまたはグローバル・グループが異なるドメイン上にあるため、接続は機能しません。

接続が成功する例: 以下のシナリオでは、ユーザー名とローカルまたはグローバル・グループが同じドメイン上にあるため、接続は機能します。

必ずしもユーザー名とローカルまたはグローバル・グループを、データベース・サーバーが実行されているドメインに定義する必要はありません。しかし、ユーザー名とローカルまたはグローバル・グループを同じドメインに定義する必要はあります。

表 24. ドメイン・コントローラーを使用した接続が成功する場合

Domain1	**Domain2**
Domain2 との間に信頼関係が存在している。	• Domain1 との間に信頼関係が存在している。 • ローカルまたはグローバル・グループ `grp2` が定義されている。 • ユーザー名 `id2` が定義されている。 • ユーザー名 `id2` が `grp2`のメンバーとなっている。

表 24. ドメイン・コントローラーを使用した接続が成功する場合 (続き)

Domain1	Domain2

DB2 サーバーがこのドメインで実行されている。以下の DB2 コマンドがこのサーバーから発行される。
`REVOKE CONNECT ON db FROM public`
`GRANT CONNECT ON db TO GROUP grp2`
`CONNECT TO db USER id2`	

ローカルまたはグローバル・ドメインがスキャンされるが、 id2 は見つからない。ドメイン機密保護がスキャンされる。	
	ユーザー名 id2 がこのドメインで見つかる。 DB2 は、このユーザー名についての追加情報 (つまり、このユーザー名がグループ grp2 のメンバーであるということ) を入手する。
ユーザー名とローカルまたはグローバル・グループが同じドメイン上にあるため、接続は機能する。	

接続が失敗する例: 以下のシナリオでは、ユーザー名がローカルまたはグローバル・グループとは異なるドメインで定義されているため、接続は機能しません。

表 25. ドメイン・コントローラーによって接続が確立されない例

Domain1	Domain2
Domain2 との間に信頼関係が存在している。	• Domain1 との間に信頼関係が存在している。 • ローカルまたはグローバル・グループ grp2 が定義されている。
• グローバル・グループ grp1 が定義されている。 • ユーザー名 id1 が定義されている。 • ユーザー名 id1 が grp1 のメンバーとなっている。	
	Domain1¥grp1 が grp2 のメンバーとなっている。

DB2 サーバーがこのドメインで実行されている。以下の DB2 コマンドがこのサーバーから発行される。
`REVOKE CONNECT ON db FROM public`
`GRANT CONNECT ON db TO GROUP grp2`
`CONNECT TO db USER id2`	

表 25. ドメイン・コントローラーによって接続が確立されない例 (続き)

Domain1	Domain2
ローカルまたはグローバルがスキャンされ、id1 が見つかる。 DB2 は、このユーザー名についての情報 (つまり、ユーザー名 id1 が grp1 のメンバーであり、グループ grp1 は Domain2¥grp2 のメンバーであるということ) を入手する。	
	グループ grp2 がこのドメインに存在している。
ローカルまたはグローバル・グループは Domain2 に存在しているが、実際のユーザー名は Domain1 で定義されているため、接続は機能しない。 GRANT CONNECT ON db TO GROUP grp1 コマンドが発行されていれば、接続は機能するはずである。	

付録F. Windows NT パフォーマンス・モニターの使用

DB2 (Windows NT 版) ユーザーが使えるパフォーマンス・モニターは、以下の 2 種類があります。

- **DB2 パフォーマンス・モニター**

  DB2 パフォーマンス・モニターは、 DB2 および DB2 コネクトだけに関連付けられたスナップショットとイベント・データを提供します。 (詳細は、「コントロール・センター」の**「ヘルプ (Help)」**押しボタンをクリックし、「はじめに (Getting Started)」オンライン・ヘルプを参照してください。)

- **Windows NT パフォーマンス・モニター**

  Windows NT パフォーマンス・モニターを使用すると、データベースとシステム・パフォーマンスの両方をモニターでき、そのシステムに登録されている任意のパフォーマンス・データ提供元から情報を取り出すことができます。 Windows NT では、以下のものを含めたマシン操作のすべてについても、パフォーマンス情報データが提供されます。

  – CPU 使用状況
  – メモリー使用状況
  – ディスクの活動状況
  – ネットワークの活動状況

Windows NT パフォーマンス・モニターへ DB2 を登録する

セットアップ・プログラムは、 DB2 を自動的に Windows NT パフォーマンス・モニターへ登録します。

Windows NT パフォーマンス・モニターを使用し、 DB2 および DB2 コネクトのパフォーマンス情報にアクセスできるようにするには、 DB2 (Windows NT 版) のパフォーマンス・カウンター用の DLL を登録する必要があります。またここで登録しておけば、 Win32 パフォーマンス API を使用してパフォーマンス・データを入手する Windows NT アプリケーションが他にあれば、そのアプリケーションも使用できるようになります。

DB2 (Windows NT 版) パフォーマンス・カウンターの DLL (DB2Perf.DLL) を、Windows NT パフォーマンス・モニターにインストールして登録するには、次のように入力します。

```
db2perfi -i
```

DLL を登録するならば、レジストリーのサービス・オプションで、新しいキーを作成することもできます。 1 つの項目には DLL の名前が示され、カウンターをサポートします。他の 3 つの項目には、その DLL に備えられている機能の名前が示されます。それらの機能は、以下のとおりです。

- **オープン**

  処理中に、DLL がシステムによって初めてロードされるときに呼び出されます。

- **収集**

  DLL からのパフォーマンス情報を要求するときに呼び出されます。

- **クローズ**

  DLL をアンロードするときに呼び出されます。

DB2 パフォーマンス情報にリモートでアクセスできるようにする

使用している DB2 (Windows NT 版) ワークステーションが、別の Windows NT マシンにネットワークで接続されている場合、この節で説明されているこの機能を使うことができます。

別の DB2 (Windows NT 版) マシンから Windows NT パフォーマンス・オブジェクトを見るには、 DB2 に管理者のユーザー名とパスワードを登録しなければなりません。(デフォルトの Windows NT パフォーマンス・モニターのユーザー名である **SYSTEM** は、 DB2 予約語なので使えません。) 名前を登録するには、次のように入力します。

```
db2perfr -r username password
```

注: 使用する `username` は、DB2 命名規則に適合していなければなりません。

ユーザー名とパスワードのデータは、レジストリー内のキーに置かれます。このときの機密保護は、管理者および SYSTEM アカウントだけがアクセスできるというものです。管理者のパスワードがレジストリーに格納されるなどの機密保護上の問題を防ぐため、データはエンコードされます。

注:

1. いったんユーザー名とパスワードの組み合わせを DB2 に登録してしまえば、パフォーマンス・モニターのローカル・インスタンスであっても、そのユーザー名とパスワードを使って明示的にログオンできます。つまり、DB2 に登録されたユーザー名情報が一致しなければ、パフォーマンス・モニターのローカル・セッションには、DB2 のパフォーマンス情報が示されないことになります。
2. ユーザー名とパスワードの組み合わせは、常に、 Windows NT の機密保護データベースに格納されているユーザー名とパスワードと一致している必要があります。Windows NT 機密保護データベース内のユーザー名かパスワードを変更した場合、リモートでのパフォーマンス・モニターに使うユーザー名とパスワードの組み合わせを再設定しなければなりません。
3. 登録するには、次のように入力します。

```
db2perfr -u <username> <password>
```

DB2 と DB2 コネクトのパフォーマンス値を表示する

パフォーマンス・モニターを使って DB2 および DB2 コネクトのパフォーマンス値を表示するには、**「追加先 (Add to)」**ボックスから、表示させる値を示すパフォーマンス・カウンターを選ぶだけです。このボックスには、パフォーマンス・データを提示するパフォーマンス・オブジェクトのリストが示されます。提供されているカウンターのリストを見るには、特定のオブジェクトを選択してください。

1 つのパフォーマンス・オブジェクトに、複数のインスタンスが存在することもあります。たとえば、LogicalDisk オブジェクトには、『% Disk Read Time』や『Disk Bytes/sec』などのカウンターが備えられています。さらに、マシン内の論理ドライブ (『C:』や『D:』など) ごとに、 1 つのインスタンスがあります。

Windows NT には、以下のパフォーマンス・オブジェクトがあります。

- **DB2 データベース・マネージャー**

  このオブジェクトは、1 つの Windows NT インスタンスのための、一般的な情報を提供します。モニターされる DB2 インスタンスは、オブジェクト・インスタンスとして表されます。

  実用ならびにパフォーマンス上の理由のため、パフォーマンス情報は、一度に 1 つの DB2 インスタンスだけから入手されます。パフォーマンス・モニターが示す DB2 インスタンスは、パフォーマンス・モニターの処理では、 `db2instance` レジストリー変数によって管理されます。同時に複数の DB2 インスタンスを実行していて、 2 つ以上のパフォーマンス情報を確認したい場合、パフォーマンス・モニターのセッションを個別に開始する必要があります、このとき、`db2instance` には、モニターする DB2 インスタンスごとに対応する値をセットするようにします。

  区分データベース・システムを実行している場合は、一度に 1 つのデータベース区画サーバー (ノード) からしかパフォーマンス情報を入手できません。デフォルトでは、デフォルト・ノード (論理ポートが 0 のノード) のパフォーマンス情報が表示されます。他のノードのパフォーマンス情報を表示するには、 DB2NODE 環境変数を、モニターしたいノードのノード番号に設定して、パフォーマンス・モニターの他のセッションを開始する必要があります。

- **DB2 データベース**

  このオブジェクトは、特定のデータベースの情報を提供します。現在アクティブなデータベースごとに、情報を利用できます。

- **DB2 アプリケーション**

  このオブジェクトは、特定の DB2 アプリケーションの情報を提供します。現在アクティブな DB2 アプリケーションごとに、情報を利用できます。

- **DB2 DCS データベース**

このオブジェクトは、特定の DCS データベースの情報を提供します。現在アクティブなデータベースごとに、情報を利用できます。

- **DB2 DCS アプリケーション**

  このオブジェクトは、特定の DB2 DCS アプリケーションの情報を提供します。現在アクティブな DB2 DCS アプリケーションごとに、情報を利用できます。

Windows NT パフォーマンス・モニターによってリストされるオブジェクトは、Windows NT マシンに何がインストールされているか、そしてどのアプリケーションがアクティブなのかによって異なります。たとえば、DB2 UDB をインストールしデータベース・マネージャーを開始していれば、 DB2 データベース・マネージャー・オブジェクトがリストされます。さらに、そのマシンで現在アクティブな DB2 データベースおよびアプリケーションがいくらかあれば、 DB2 データベースおよび DB2 アプリケーション・オブジェクトもリストされます。 Windows NT システムを DB2 コネクトのゲートウェイとして使っていて、現在アクティブな DCS データベースおよびアプリケーションがいくつかある場合、 DB2 DCS データベースおよび DB2 DCS アプリケーション・オブジェクトがリストされます。

リモート DB2 のパフォーマンス情報にアクセスする

DB2 パフォーマンス情報にリモートでアクセスできるようにする方法については、前に説明されています。**「追加先 (Add to)」**ボックスで、モニターする別のコンピューターを選択してください。これにより、そのコンピューター上で使用できるすべてのパフォーマンス・オブジェクトのリストが表示されます。

リモート・コンピューターで DB2 パフォーマンス・オブジェクトをモニターできるようにするには、そのコンピューターにインストールされている DB2 UDB または DB2 コネクト・コードのレベルが、バージョン 6 以上でなければなりません。

DB2 パフォーマンス値をリセットする

アプリケーションで DB2 モニター API を呼び出すと、戻される情報は、通常は DB2 サーバーを開始してからの累積値になります。しかし、以下のようにすると役立つことがあります。

- パフォーマンス値をリセットする
- テストを実行する
- その値をもう一度リセットする
- テストを再実行する

データベースのパフォーマンス値をリセットするには、 **db2perfc** プログラムを使用します。次のように入力してください。

```
db2perfc
```

デフォルトでは、これによりアクティブな DB2 データベースすべてのパフォーマンス値がリセットされます。ただし、リセットするするデータベースのリストを指定することも可能です。また -d オプションを使用して、 DCS データベースのパフォーマンス値をリセットするよう指定することもできます。たとえば、次のようにします。

```
db2perfc
db2perfc dbalias1 dbalias2 ... dbaliasn

db2perfc  -d
db2perfc  -d  dbalias1 dbalias2 ... dbaliasn
```

最初の例では、アクティブな DB2 データベースすべてのパフォーマンス値がリセットされます。次の例では、特定の DB2 データベースの値がリセットされます。 3 番目の例では、アクティブな DB2 DCS データベースすべてのパフォーマンス値がリセットされます。最後の例では、特定の DB2 DCS データベースの値がリセットされます。

db2perfc プログラムは、関係する DB2 サーバー・インスタンス (つまり、 **db2perfc** を実行するときのセッションで、 db2instance に入れられているインスタンス) のために、現在、データベース・パフォーマンス情報にアクセスしているすべてのプログラムの値をリセットします。

db2perfc を呼び出す場合、 **db2perfc** コマンドの実行中に DB2 パフォーマンス情報にリモートでアクセスしているユーザーがいれば、そのユーザーが見ることのできる値もリセットされます。

注: sqlmrset という DB2 API がありますが、これを使うと、アプリケーションからグローバルにではなくローカルに見ることのできる特定のデータベースの値を、アプリケーション側でリセットできます。詳細については、*管理 API 解説書* を参照してください。

付録G. Windows NT または Windows 2000 データベース区画サーバーを使った作業

Windows NT または Windows 2000 環境で構成の特性を変更する場合、この章で紹介するように特別な方法で行います。他のオペレーティング・システム環境では、 *管理の手引き: パフォーマンス* の『プロセッサーの追加による構成のスケーリング』という章に示されている方法で行います。

この章で紹介する方法は、以下のとおりです。

- 『インスタンス内のデータベース区画サーバーのリスト』
- 『インスタンスへのデータベース区画サーバーの追加』
- 401ページの『データベース区画の変更』
- 402ページの『インスタンスからデータベース区画を除去する』

インスタンス内のデータベース区画サーバーのリスト

Windows NT または Windows 2000 で **db2nlist** コマンドを使えば、インスタンスに関与するデータベース区画サーバーのリストを取得できます。

コマンドは、次のように使用します。

```
db2nlist
```

ここに示したコマンドを使用する場合、デフォルトのインスタンスは現行インスタンス (DB2INSTANCE 環境変数で設定される) です。特定のインスタンスを指定するには、次のコマンドを使ってインスタンスを指定できます。

```
db2nlist /i:instName
```

ここで、`instName` は、必要とする特定のインスタンス名です。

次のコマンドを使えば、各区画サーバーの状況を要求することもできます。

```
db2nlist /s
```

各データベース区画サーバーの状況は、開始中、実行中、停止中、または停止済みのいずれかになります。

インスタンスへのデータベース区画サーバーの追加

Windows NT または Windows 2000 で **db2ncrt** コマンドを使用すると、インスタンスにデータベース区画サーバー (ノード) を追加できます。

注: このインスタンスの中にデータベースがすでに含まれている場合は、 **db2ncrt** コマンドを使用しないでください。代わりに、**db2start addnode** コマンドを使用します。上記のコマンドにより、新しいデータベース区画サーバーにデータベースを正しく追加することができます。 db2nodes.cfg ファイルは**編集しないでください**。このファイルを変更すると、区分データベース・システムに不整合が生じる可能性があるからです。

このコマンドには、以下の必須パラメーターがあります。

```
db2ncrt /n:node_number
        /u:username,password
        /p:logical_port
```

- /n:

  データベース区画サーバーを識別するための固有ノード番号です。番号には、昇順で 1 ～ 999 までを指定できます。

- /u:

  DB2 サービスのログオン・アカウント名とパスワードです。

- /p:logical_port

  論理ポートがゼロ (0) でない場合に、データベース区画サーバーに使用する論理ポート番号。このパラメーターを指定しないと、論理ポート番号には 0 が割り当てられます。

論理ポート・パラメーターは、マシンに最初のノードを作成するときには、オプショナルに過ぎません。論理ノードを作成する場合は、このパラメーターを指定し、未使用の論理ポート番号を選択しなければなりません。このパラメーターの使用に関して、いくつかの制約事項があります。

- どのマシンにも、論理ポート 0 のデータベース区画サーバーが 1 つずつ存在しなければなりません。
- x:¥winnt¥system32¥drivers¥etc¥ ディレクトリーのサービス・ファイル内で、 FCM 通信のために予約されているポート範囲よりも大きいポート番号を選択することはできません。たとえば、現行インスタンスのために 4 つのポートの範囲を予約する場合、最大ポート番号は 3 になるはずです (ポート 1、2、3。ポート 0 はデフォルトの論理ノード)。ポート範囲は、 **db2icrt** を /r:base_port，end_port パラメーターと一緒に使用するときに定義します。

次のようないくつかの任意指定パラメーターもあります。

- /g:network_name

  データベース区画サーバーのネットワーク名を指定します。このパラメーターを指定しなかった場合、 DB2 はシステムで最初に検出した IP アドレスを使用します。

  マシン上に複数の IP アドレスがあり、データベース区画サーバーに特定の IP アドレスを割り当てたい場合に、このパラメーターを使用します。 ネットワーク名や IP アドレスを *network_name* パラメーターに入力することができます。

- /h:host_name

  TCP/IP ホスト名。ホスト名がローカル・ホスト名でない場合に FCM が内部通信用に使用します。このパラメーターが必要になるのは、データベース区画サーバーをリモート・マシンに追加する場合です。

- /i:instance_name

  インスタンス名。デフォルトは、現行インスタンスです。

- /m:machine_name

  ノードが常駐する Windows NT ワークステーションのコンピューター名。デフォルトの名前は、ローカル・マシンのコンピューター名です。

- /o:instance_owning_machine

  インスタンス所有マシンであるマシンのコンピューター名。デフォルトはローカル・マシンです。このパラメーターは、インスタンス所有マシンでないマシンで **db2ncrt** コマンドを呼び出すときに必須です。

たとえば、(複数の論理ノードを実行するために) 新しいデータベース区画サーバーを、インスタンス所有マシン MYMACHIN 上のインスタンス TESTMPP へ追加して、この新しいノードを論理ポート 1 を使用するノード 2 として認識されるようにするには、次のように入力します。

```
db2ncrt /n:2 /p:1 /u:my_id,my_pword /i:TESTMPP
   /M:TEST /o:MYMACHIN
```

データベース区画の変更

Windows NT または Windows 2000 で **db2nchg** コマンドを使用して、以下のことを行うことができます。

- データベース区画をあるマシンから別のマシンへと移動する。
- マシンの TCP/IP ホスト名を変更する。

  複数のネットワーク・アダプターを使用する予定がある場合には、このコマンドを使用して、 *db2nodes.cfg* ファイルの『netname』フィールドに TCP/IP アドレスを指定しなければなりません。
- 異なる論理ポート番号を使用する。
- データベース区画サーバー (ノード) に異なる名前を使用する。

このコマンドには、以下の必須パラメーターがあります。

```
db2nchg /n:node_number
```

パラメーター /n: は、構成を変更したいデータベース区画サーバーのノード番号です。このパラメーターは必須です。

オプション・パラメーターには、以下のものがあります。

- /i:instance_name

  このデータベース区画サーバーが参加しているインスタンスを指定します。このパラメーターを指定しなかった場合、デフォルトである現行インスタンスが使用されます。

- /u:username,password

  DB2 サービスのログオン・アカウント名とパスワードを変更します。このパラメーターを指定しなかった場合、ログオン・アカウント名とパスワードは変わりません。

- /p:logical_port

  データベース区画サーバーの論理ポートを変更します。データベース区画サーバーを異なるマシンへ移動させる場合、このパラメーターの指定は必須です。このパラメーターを指定しなかった場合、論理ポート番号は変わりません。

- /h:host_name

  FCM が内部通信のために使用する TCP/IP ホスト名を変更します。このパラメーターを指定しなかった場合、ホスト名は変わりません。

- /m:machine_name

  データベース区画サーバーを別のマシンへ移動させます。データベース区画サーバーを移動できるのは、インスタンス内にデータベースが 1 つもない場合だけです。

- /g:network_name

  データベース区画サーバーのネットワーク名を変更します。

  マシン上に複数の IP アドレスがあり、データベース区画サーバーに特定の IP アドレスを割り当てたい場合に、このパラメーターを使用します。ネットワーク名や IP アドレスを network_name に入力することができます。

たとえば、ノード 2 に割り当てられている論理ポート (インスタンス TESTMPP に参加している) が論理ポート 3 を使用するように変更したい場合は、次のコマンドを入力します。

```
db2nchg /n:2 /i:TESTMPP /p:3
```

インスタンスからデータベース区画を除去する

Windows NT または Windows 2000 で **db2ndrop** コマンドを使うと、データベースのないインスタンスからデータベース区画サーバー (ノード) を除去できます。データベース区画サーバーを除去する場合、そのノード番号は新しいデータベース区画サーバーに再使用することができます。

インスタンスからデータベース区画サーバーを除去する場合は、注意してください。インスタンス所有データベース区画サーバーのノードのゼロ (0) をインスタンスから除去すると、インスタンスは使用できなくなります。インスタンスを除去したい場合は、**db2idrop** コマンドを使用します。

注: このインスタンスの中にデータベースが含まれている場合は、 **db2ndrop** コマンドを使用しないでください。代わりに、**db2stop drop nodenum** コマンドを使用します。上記のコマンドにより、新しいデータベース区画からデータベースを正しく除去することができます。 db2nodes.cfg ファイルは**編集しないでください**。このファイルを変更すると、区分データベース・システムに不整合が生じる可能性があるからです。

複数の論理ノードが実行されているマシンから、論理ポート 0 に割り当てられているノードを除去する場合は、 0 以外の論理ポートに割り当てられているノードをすべて除去してからでないと、論理ポート 0 に割り当てられているノードを除去できません。どのデータベース区画サーバーにも、論理ポート 0 に割り当てられているノードが 1 つずつなければなりません。

このコマンドには、以下のパラメーターがあります。

```
db2ndrop /n:node_number /i:instance_name
```

- /n:

  データベース区画サーバーを識別するための固有ノード番号です。これは必須パラメーターです。番号には、昇順でゼロ (0) から 999 までを指定できます。ノードのゼロ (0) はインスタンス所有マシンを表すことに留意してください。

- /i:instance_name

  インスタンス名です。これはオプション・パラメーターです。このパラメーターを指定しない場合、デフォルトのインスタンスは現行インスタンス (DB2INSTANCE レジストリー変数で設定される) です。

付録H. 複数の論理ノードの構成

DB2 エンタープライズ拡張エディションでは、どのマシンにもデータベース区画サーバーが 1 つずつ割り当てられるように構成するのが一般的です。しかし、状況にもよりますが、複数のデータベース区画サーバーを同一のマシンで実行したほうが良い場合もあります。つまり、構成にマシンの数よりも多くのノードを含めることも可能です。その場合、それらのノードが*同じ* インスタンスに参加しているのであれば、そのマシンでは*複数の論理ノード* が実行されていると言います。それらのノードが異なるインスタンスに参加している場合には、そのマシンが複数の論理ノードをホストしていることにはなりません。

複数の論理ノードがサポートされているために、次のような 3 種類の構成を選択することが可能になります。

- 各マシンにデータベース区画サーバーが 1 つずつ用意されている、標準的な構成。
- 1 台のマシンに複数のデータベース区画サーバーがある、複数論理ノード構成。
- 複数のマシンのそれぞれで複数の論理ノードが実行される構成。

複数の論理ノードを使用する構成は、システムが、対称マルチプロセッサー (SMP) アーキテクチャーのマシン上で照会を実行するときに便利です。 1 つのマシンに複数の論理ノードを構成できるということは、いずれかのマシンで障害が発生した場合にも効果を発揮します。あるマシンで障害が発生しても (その結果、そのマシン上の 1 つまたは複数のデータベース区画サーバーが使用できなくなっても)、 DB2START NODENUM コマンドを使用すれば、他のマシンでそのデータベース区画サーバーを再始動できます。これにより、ユーザー・データを確実に引き続き利用できます。

他の利点は、複数の論理ノードがあれば SMP ハードウェア構成を十分に活用できるということです。さらに、データベース区画サーバーが小さくなるので、データベース区画や表スペースのバックアップおよび復元、索引の作成などといったタスクを実行するときのパフォーマンスが向上します。

次の 2 つの方法のいずれかで複数の論理ノードを構成できます。

- `db2nodes.cfg` ファイル内で論理ノード (データベース区画) を構成します。構成後、DB2START コマンドとその関連 API を使用して、すべての論理ノードとリモート・ノードを開始できます。

  注: Windows NT 環境では、システム内にデータベースが 1 つもない場合は、*db2ncrt* を使用してノードを追加する必要があります。 1 つ以上のデータベースがある場合は、 DB2START ADDNODE コマンドを使用します。 Windows NT 内では、 *db2nodes.cfg* ファイルを手動で編集することは絶対に避けてください。

- 他の論理データベース区画 (ノード) がすでに実行している、別のプロセッサー上で論理ノードを再始動します。この場合、 db2nodes.cfg 内で論理データベース区画として指定したホスト名とポート番号を指定変更できます。

db2nodes.cfg 内で論理データベース区画 (ノード) を構成するには、このファイル内に項目を作成して、ノードの論理ポート番号を割り当てなければなりません。次の構文を使用する必要があります。

```
nodenumber hostname logical-port netname
```

注: Windows NT 環境では、システム内にデータベースが 1 つもない場合は、 *db2ncrt* を使用してノードを追加する必要があります。 1 つ以上のデータベースがある場合は、 DB2START ADDNODE コマンドを使用します。 Windows NT 内では、 *db2nodes.cfg* ファイルを手動で編集することは絶対に避けてください。

Windows NT での *db2nodes.cfg* ファイルの形式は、 UNIX での同ファイルのそれとは異なります。 Windows NT での列形式は、次のとおりです。

```
nodenumber hostname computername logical_port netname
```

FCM 通信にとって十分な数のポートを etc ディレクトリーの services ファイルに定義しなければなりません。

付録I. 高速ノード間通信

DB2 ユニバーサル・データベース エンタープライズ拡張エディションをご使用の場合、作業環境の通信量は膨大なものに及ぶかもしれません。そのような環境下では、システム全体のスループットが業務に重大な影響を及ぼします。

区画環境に使用できるネットワークが 2 種類あります。 1 つは、公衆 LAN 上で TCP/IP を使用するネットワークです。もう 1 つは、専用の相互接続上で TCP/IP または仮想インターフェース (VI) アーキテクチャーを使用するネットワークです。

公衆相互接続は、既存の TCP/IP と協働します。 TCP/IP は、ほとんどすべての場所で有効な通信プロトコルです。これは、ローカル・エリア・ネットワーク (LAN) 環境です。この環境の利点は、専用のハードウェアおよびソフトウェアがなくても、クラスターにすぐに接続できることです。この環境の欠点は、クラスターのトラフィックが増えると LAN 全体のサービスの質が低下することです。たとえば、クラスター内のデータベース活動に影響を及ぼす通信『バースト』が発生すると、その LAN に関係する通信で障害が発生する場合があります。また、LAN 環境の他の部分で行われている通信により、クラスター内のデータベース処理のパフォーマンスを一貫した状態に保つことが難しくなります。

専用相互接続は、独立したネットワークとして機能します。このネットワークは、クラスター内で使用される唯一のネットワークとなる場合もあれば、 LAN 環境に追加される場合もあります。このネットワークは、クラスターのメンバー同士の通信専用に用いられます。このネットワークのことをシステム・エリア・ネットワーク (SAN) といいます。 (LAN 環境とは異なり) データベースのパフォーマンスが外部通信トラフィックの影響を受けることはありませんし、外部通信トラフィックがデータベースのパフォーマンスの影響を受けることもありません。この環境の欠点は、それぞれのネットワークを別々に管理しなければならない場合があることと、 LAN と SAN の両方に別々のハードウェア、ソフトウェア、プロトコルを追加しなければならないため、多額の費用が必要になることです。専用相互接続の例として、100 Mbps のイーサネットがあります。

今まで存在していた公衆 LAN 環境を残しつつも、 SAN 上で処理する転送データの量を (クラスター内で) 増やしたいとお考えになるかもしれません。そのような調整を行うことは、クラスター外へも通信アクセスを確立できるようにしたい場合には非常に好都合です。 Windows NT 操作環境内では、 NT ドメイン・コントローラーへの通信アクセス用に公衆 LAN を残しておく必要が場合もあります。 (ドメイン・コントローラーについては、 383ページの『付録E. DB2 (Windows NT 版) が Windows NT 機密保護を処理する方法』を参照してください。)

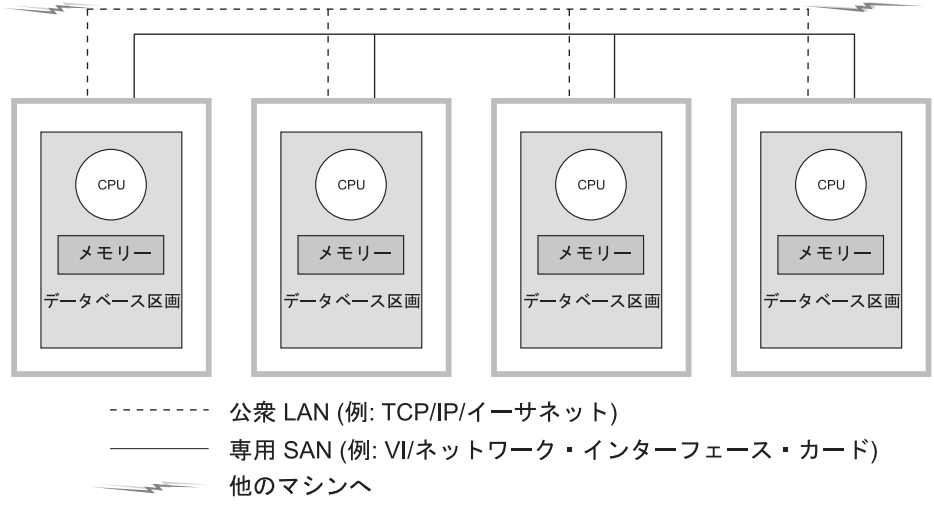


図 12. 専用 SAN と公衆 LAN の結合

続く節では、以下の点について説明します。

- 『TCP/IP を使用した高速相互接続』
- 409ページの『VI を使用した高速相互接続』

TCP/IP を使用した高速相互接続

以下に、 TCP/IP を使用するネットワーク・ハードウェアのセットアップに関連した前提条件の例を示します。

- 標準イーサネット

  特別なハードウェア、ソフトウェア、またはプロトコルは必要ありません。
- IBM Netfinity SP Switch

  このハードウェアの要件については、次の節で説明します。

IBM Netfinity SP Switch を使用するための前提条件

Netfinity の詳細を調べるには、 http://www.ibm.com/pc/us/netfinity の URL にアクセスしてください。

その他の資料とソフトウェアのアップグレードについては、 IBM Support Web (http://www.ibm.com/pc/support) にアクセスしてください。

1. **「サーバー (Servers)」**をクリックします。
2. 「ファミリー (family)」の下で、**「クラスター化 (Clustering)」**をクリックします。

3. 「テクニカル・インフォメーション (Technical Information)」の下で、ソフトウェア・アップグレードが必要な場合は**「ダウンロード可能なファイル (Downloadable files)」**を、その他の資料を参照したい場合には**「オンライン資料 (Online publications)」**をクリックします。

IBM Netfinity SP Switch についてのトピックを探し出し、必要なファイルをダウンロードします。

IBM Netfinity SP Switch のセットアップ手順

IBM Netfinity SP Switch のインストールに関する指示については、 *IBM Netfinity SP Switch Installation and User's Guide* を参照してください。

各種ハードウェア / ソフトウェア・コンポーネント (サーバー格納装置、ホスト・アダプター、SP スイッチ・ソフトウェアなど) をインストール、構成、およびテストする際には、付属のハードウェア / ソフトウェア・ガイドを参照してください。

DB2 は、一度インストールしたならば、追加の変更を加えなくても IBM Netfinity SP Switch を使用します。

VI を使用した高速相互接続

仮想インターフェース (VI) アーキテクチャーは、 Windows NT 大量並列処理 (MPP) 環境内での TCP/IP の代替のノード間通信プロトコルです。 VI は、Intel 社、Microsoft 社、および Compaq 社が合同で開発した、システム・エリア・ネットワーク (SAN) 上でのパフォーマンスを向上させる新しい通信アーキテクチャーです。このアーキテクチャーの詳細は、http://www.viarch.org を参照してください。

プロダクトは、DB2 ユニバーサル・データベースとは別売で、 VIA 使用可能化ネットワーク・インターフェース・カード (NIC)、スイッチ、およびソフトウェア・ドライバー実装が含まれます。独立ハードウェア・ベンダーの中には、そのような製品をリリースした、または今後リリースする予定のものがあります。

TCP/IP と比較すると、 VI アーキテクチャーには待ち時間が短い、帯域幅が広い、CPU の使用量が少なくて済むなどの特徴があります。通信集約的な環境では、 VI アーキテクチャーを使用すると総合的なシステムのスループットを向上させることができます。クラスター内のノード数が増え、データの通信量が増えると、 VI アーキテクチャーを使用することによる利益も大きくなってゆきます。

DB2 ユニバーサル・データベースでは、*Virtual Interface Architecture Specification, Version 1.0*、*Intel Virtual Interface (VI) Architecture Developers' Guide, Version 1.0* に準拠し、 『Virtual Interface Architecture Conformance Suite』をパスする VI アーキテクチャーの実装をサポートしています。この仕様書は、 Web の http://www.intel.com/design/servers/vi/the_spec/specification.htm にあります。
この Developer's Guide は、 Web の

http://www.intel.com/design/servers/vi/developer/ia_imp_guide.htm にあります。 Conformance Suite の情報も、この同じ URL にあります。

IBM では、 DB2 ユニバーサル・データベース EEE V5.2 での仮想インターフェース (VI) アーキテクチャーのサポートを表明しています。

VI アーキテクチャーを採用し DB2 ユニバーサル・データベース EEE によってサポートされている他のプロダクトについては、 http://www.software.ibm.com/data の DB2 ユニバーサル・データベース・サポート組織に連絡するか、 1-800-237-5511 (米国とカナダのみ) に連絡してください。

DB2 ユニバーサル・データベースでテスト済みのプロダクトは、以下のとおりです。

- GigaNet Interconnect。詳細は、411ページの『GigaNet Interconnect のセットアップ手順』を参照してください。
- Compaq ServerNet Interconnect。詳細は、413ページの『ServerNet Interconnect のセットアップ手順』を参照してください。
- Fujitsu Synfinity Interconnect。詳細は、417ページの『Synfinity Interconnect のセットアップ手順』を参照してください。

DB2 ユニバーサル・データベースで動作する他のプロダクトがあるかもしれません。そのプロダクトがサポートされているかどうかを確認するには、そのプロダクトのベンダーを調べ、次いで弊社のサービスとサポートを調べてください。

仮想インターフェース (VI) ハードウェアのセットアップ

以下に、 VI を使用するネットワーク・ハードウェアのセットアップに関連した前提条件の例を示します。

- GigaNet Interconnect

  411ページの『GigaNet Interconnect のセットアップ手順』には、このハードウェアを選択した場合に必要なハードウェア、ソフトウェア、およびプロトコルについて情報が示されています。

  GigaNet プロダクトの詳細を調べる、あるいは GigaNet のサービスおよびサポートに連絡を取るには、 http://www.giganet.com/ の URL にアクセスしてください。

- Compaq ServerNet Interconnect

  413ページの『ServerNet Interconnect のセットアップ手順』には、このハードウェアを選択した場合に必要なハードウェア、ソフトウェア、およびプロトコルについて情報が示されています。

  ServerNet プロダクトの詳細を調べる、あるいは ServerNet のサービスおよびサポートに連絡を取るには、 http://www.servernet.com/ の URL にアクセスしてください。

- Fujitsu Synfinity Interconnect

417ページの『Synfinity Interconnect のセットアップ手順』には、このハードウェアを選択した場合に必要なハードウェア、ソフトウェア、およびプロトコルについて情報が示されています。

Synfinity プロダクトの詳細を調べる、あるいは Synfinity のサービスおよびサポートに連絡を取るには、 Fujitsu System Technologies (http://www.fujitsu.com/) にアクセスしてください。

VI を使用できるように DB2 を構成する必要があります。 419ページの『VI を使用して DB2 を実行できるようにする』には、VI を使用するのに必要な情報が収められています。

GigaNet Interconnect のセットアップ手順

この環境を設定するときに必要なハードウェアとソフトウェアのリストには、以下のプロダクトが含まれています。

- GigaNet GNN1000 ネットワーク・インターフェース・カード
- GigaNet GNX5000 スイッチ
- GigaNet GNCxx11 Copper Interconnect ケーブル
- GigaNet cLAN Software バージョン 2.0

DB2 ユニバーサル・データベースで GigaNet Interconnect を動作させるときに必要なステップを、これから示します。それぞれのステップは、各ステップで必要とされることの要旨です。ここでは、各ステップに関係するすべてが詳細に述べられていません。それぞれのステップで参照されている資料もご使用ください。そこには、詳細な手順と必要な指示が載せられています。

それぞれの GigaNet GNN1000 は、GigaNet cLAN Software の CD-ROM にパッケージされています。この CD-ROM には、GigaNet Interconnect の設定に必要なすべてのソフトウェアが含まれています。さらに、VI Architecture software developer's kit (SDK) と Adobe Acrobat Reader が含まれています。 VI Architecture SDK は VI 対応アプリケーションを開発およびテストするために使用されます。 Adobe Acrobat Reader は、CD-ROM に掲載されている、VI 対応アプリケーションを開発する方法を説明した文書を表示するために使用されます。

ステップの要旨は以下のとおりです。

1. アダプター・カードをインストールします。
2. スイッチとケーブルをインストールします。
3. アダプター・ドライバーをインストールします。
4. cLAN Management Console をインストールします。
5. Interconnect をテストします。

ステップは以下のとおりです。

1. GigaNet GNN1000 ネットワーク・インターフェース・カードをインストールします。インストールの手順については、*GigaNet GNN1000 User Guide* を参照してください。
2. GigaNet GNX5000 スイッチおよびケーブルをインストールします。インストールの手順については、 *GigaNet GNX5000 User Guide* を参照してください。
3. GNX5000 Switch に接続されているノードごとに、 GigaNet GNN1000 アダプター・ドライバー・ソフトウェアをインストールします。インストールの手順については、*GigaNet GNN1000 User Guide* を参照してください。 GigaNet によって提供されているドライバーをインストールしているのであれば、ここでさらに詳しい情報をお知らせします。
   a. 前のバージョンの GNN1000 ドライバーがインストールされていれば、それを削除します。削除するときには、ノードをリブートする必要があります。
   b. 「スタート」–>「設定」–>「コントロール パネル」–> 「ネットワーク」–>「アダプター」–>「追加」を使用して、ドライバーをインストールします。
   c. **「ディスク使用 (Have Disk...)」**をクリックし、 CD-ROM のドライバー・ディレクトリーを指定します。たとえば、F: が CD-ROM ドライブであれば、`F:¥Driver` を使います。
   d. 『GNN1000 NDIS Adapter』を選んで、**「OK」**をクリックします。
   e. ネットワーク・プロトコルを構成して、インストールを完了します。

   GigaNet アダプター・ドライバー・ソフトウェアは、 http://www.giganet.com の GigaNet Web サイトでも入手できます。 GigaNet Web サイトのサポート・ページに載せられている、ダウンロードおよびインストールの指示を参照してください。

   GNN1000 アダプター・ドライバーをインストールすると、ノードがリブートします。
4. GigaNet Interconnect の保全性をテストするときには、 GigaNet cLAN Management Console (GMC) を使用できます。 GigaNet cLAN Management Console は、Console と Agent の 2 つで構成されています。 Agent については、クラスター内のすべてのノードにインストールする必要があります。 Console については、クラスター内のノードへのアクセス権を持つ、任意のネットワーク・ノードにインストールすることができます。最も便利でお勧めのインストール方法は、クラスター内の各ノードに Console と Agent の両方をインストールすることです。

   GigaNet cLAN Management Console をインストールします。 cLAN Management Console のインストール手順とその他の情報については、 *GigaNet GNN1000 User Guide* を参照してください。ここで、インストール手順をさらに詳しく説明します。
   a. cLAN ソフトウェア CD を CD-ROM ドライブに挿入します。
   b. CD 自動インストール・メニューが表示されるのを待ちます。
   c. 「cLAN Management Console のインストール (Install cLAN Management Console)」をクリックします。

d. クラスター内の残りのノードそれぞれで、このインストール手順を繰り返します。

GigaNet cLAN Management Console ソフトウェアは、 http://www.giganet.com の GigaNet Web サイトでも入手できます。 GigaNet Web サイトのサポート・ページに載せられている、ダウンロードおよびインストールの指示を参照してください。

cLAN Management Console をインストールすると、ノードはリブートします。

5. GigaNet ハードウェアが作動していることをテストします。このテストは、以下のようにして実行します。
   a. GMC をオープンします。 (「プログラム」–>「GigaNet」–>「cLAN Management Console」)
   b. ダイアログ・ボックスが表示され、LAN の中のアクセス可能なマシンがすべて示されます。**「ESC」**を押します。
   c. メニュー・バーから**「コンソール (Console)」—>「ローカル (Local)」**を選択します。
   d. クラスター内の全メンバーが表示されており、すべて 『アクティブ』 であることを確認します。
   e. メニュー・バーから**「ユーティリティー (Utilities)」—>「VI スループット (VI Throughput)」**を選択します。これによりスループット・テストが実行され、データが実際にハードウェアを通っているかどうか確認できます。
   f. テストで使用する 2 つのコンピューターの名前を、大文字で入力します。ローカル・ノードをソース・ノードとします。
   g. **「測定開始 (Start Measuring)」**をクリックします。データは、少なくとも毎秒 65 MB の速度で転送されるはずです。
   h. 接続のテストを停止するには、**「測定停止 (Stop Measuring)」**をクリックします。
   i. クラスター内の他のノードについても、ローカル・ノード (ソース) と他方のノード (受信) との間のスループットを計測し、テストを繰り返してください。

   接続テストが行われないようであれば、 *GigaNet GNN1000 User Guide* および *GigaNet GNX5000 User Guide* のトラブルシューティングの節を参照してください。

DB2 ユニバーサル・データベース (Windows NT 版) のインストール方法とインプリメント方法については、 *DB2 エンタープライズ拡張エディション (Windows 版) 概説およびインストール* を参照してください。

ServerNet Interconnect のセットアップ手順

この環境を設定するときに必要なハードウェアとソフトウェアのリストには、以下のプロダクトが含まれています。

- ServerNet PCI Adapter Driver (SPAD)、 (プロダクト ID T0089) のバージョン 1.3.5 以上

- ServerNet Switch 1
- ServerNet Area Network Manager (SANMan)、 (プロダクト ID T0087) のバージョン 1.1.3 以上

DB2 ユニバーサル・データベースで ServerNet Interconnect を動作させるときに必要なステップを、以下に示します。それぞれのステップは、各ステップで必要とされることの要旨です。ここでは、各ステップに関係するすべてが詳細に述べられていません。それぞれのステップで参照されている資料もご使用ください。そこには、詳細な手順と必要な指示が載せられています。

ここに示されているステップでは、クラスター内で使っているノードは、最大でも 6 つであると想定しています。 7 つ以上のノードを使用する要件があるのであれば、ServerNet に連絡してお聞きください。

ステップは以下のとおりです。

1. ServerNet ネットワーク・インターフェース・カードをインストールします。インストールの手順については、 *ServerNet-I Virtual Interface Software Release Document (プロダクト ID N0031)* を参照してください。
2. ServerNet Switch 1 をインストールします。インストールの手順については、 *ServerNet-I Virtual Interface Software Release Document (プロダクト ID N0031)* を参照してください。
3. 前の ServerNet ドライバーをアンインストールします。 (初めて ServerNet をインストールするのであれば、このステップはスキップしてください。)
   a. 「ネットワーク」コントロール パネルをオープンします。 (**「スタート」**–>**「設定」**–>**「コントロール パネル」**–> **「ネットワーク」**)
   b. **「アダプター」**タブをクリックします。
   c. Tandem ServerNet PCI Adapter Driver を削除します。
   d. **「サービス」**タブをクリックします。
   e. SANMan を削除します。
   f. **「プロトコル」**タブをクリックします。
   g. Tandem ServerNet-I VI Protocol を削除します。
4. Tandem ServerNet PCI アダプター・ドライバーをインストールします。 ServerNet によって提供されているソフトウェア CD を使ってインストールしているのであれば、ここでさらに詳しい情報をお知らせします。
   a. 「ネットワーク」コントロール パネルをオープンします。 (**「スタート」**–>**「設定」**–>**「コントロール パネル」**–> **「ネットワーク」**)
   b. **「アダプター」**タブをクリックします。 (「アダプター」画面が表示されます。)
   c. それぞれのドライブまたはディレクトリー (あるいはその両方) に、新しい ServerNet ドライバーが示されていることを確認します。その後、正しいドライブまたはディレクトリー (あるいはその両方) を示しているコマンド・プロンプト

で、『ernnn.exe -d』と入力して、自己抽出プログラムを開始します。
(『ernnn.exe』は Engineering Release － ERnnn.EXE － の名前で、後の部分には番号が入ります。これにより、インストールされる ServerNet ドライバーの固有のバージョンが分かります。)

d. 抽出したファイルを入れるドライブまたはディレクトリー (あるいはその両方) に変更します。 『Spad n.n.n ¥ Free』サブディレクトリーに変更します (ここで、『n.n.n』はプロダクトの固有のバージョンを示します)。 (トラブルシューティング環境または開発環境で作業している場合、『Spad n.n.n ¥ Free』サブディレクトリーではなく、『Spad n.n.n ¥ Checked』サブディレクトリーに変更します。)

e. 『oemsetup.multi_node』ファイルの名前を、『oemsetup.inf』に変更します。

f. 「アダプター」タブで、**「追加 (Add)」**を選択します。 (「アダプターの選択」画面が表示されます。)

g. **「ディスク使用 (Have Disk...)」**をクリックします。 (「ディスクの挿入 (Insert Disk)」画面が表示されます。)

h. oemsetup.inf があるドライブまたはディレクトリー (あるいはその両方) を入力します。

i. ダイアログ・ボックスに『Tandem ServerNet PCI Adapter Driver』が示されているのを確認してから、**「OK」**をクリックします。アダプターのリストに、ServerNet アダプターが表示されていることを確認します。**「閉じる (Close)」**をクリックします。

j. コンピューターを再始動するのであれば**「はい (Yes)」**を選択します。あるいは、SANMan および VI のソフトウェア開発者キット (SDK) のインストールを続けるのであれば、**「いいえ (No)」**を選択します。

5. SANMan をインストールします。 ServerNet によって提供されているソフトウェア CD を使ってインストールしているのであれば、ここでさらに詳しい情報をお知らせします。

a. 「ネットワーク」コントロール パネルをオープンします。 (「スタート」–>「設定」–>「コントロール パネル」–> 「ネットワーク」)

b. **「サービス」**タブをクリックします。 (「サービス」画面が表示されます。)

c. それぞれのドライブまたはディレクトリー (あるいはその両方) に、新しい ServerNet ドライバーが示されていることを確認します。その後、正しいドライブまたはディレクトリー (あるいはその両方) を示しているコマンド・プロンプトで、『ernnn.exe -d』と入力して、自己抽出プログラムを開始します。
(『ernnn.exe』は Engineering Release － ERnnn.EXE － の名前で、後の部分には番号が入ります。これにより、インストールされる ServerNet ドライバーの固有のバージョンが分かります。)

d. 「サービス」タブで、「**追加 (Add)**」を選択します。 (「サービスの選択 (Select Services)」画面が表示されます。)

e. 抽出したファイルを入れるドライブまたはディレクトリー (あるいはその両方) に変更します。 『SANMan n.n.n ¥ Free』サブディレクトリーに変更します (ここで、『n.n.n』はプロダクトの固有のバージョンを示します)。 (トラブルシューティング環境または開発環境で作業している場合、『SANMan n.n.n ¥ Free』サブディレクトリーではなく、『SANMan n.n.n ¥ Checked』サブディレクトリーに変更します。)

f. スイッチのライトを見て、スイッチが X と Y のどちらなのかを判別します。一方のライトは『X』を示し、もう一方は『Y』を示します。

g. X スイッチであれば、X=1 および Y=0 を選択します。すべてのケーブルが、ネットワーク・カードの X ポートに接続されていることを確認してください。

h. Y スイッチであれば、X=0 および Y=1 を選びます。すべてのケーブルが、ネットワーク・カードの Y ポートに接続されていることを確認してください。

i. 現在のマシンのネットワーク・カードが接続されている、スイッチ (交換機) のポート番号を指定します。

j. 6 つのポートすべてについて、『PC』を選択します。

6. 仮想インターフェース・プロトコルをインストールします。 ServerNet によって提供されているソフトウェア CD を使ってインストールしているのであれば、ここでさらに詳しい情報をお知らせします。

a. 「ネットワーク」コントロール パネルをオープンします。 (**「スタート」–>「設定」–>「コントロール パネル」–> 「ネットワーク」**)

b. **「プロトコル」**タブをクリックします。 (「ネットワーク・プロトコル (Network Protocols)」画面が表示されます。)

c. それぞれのドライブまたはディレクトリー (あるいはその両方) に、新しい ServerNet ドライバーが示されていることを確認します。その後、正しいドライブまたはディレクトリー (あるいはその両方) を示しているコマンド・プロンプトで、『ernnn.exe -d』と入力して、自己抽出プログラムを開始します。
(『ernnn.exe』は Engineering Release − ERnnn.EXE − の名前で、後の部分には番号が入ります。これにより、インストールされる ServerNet ドライバーの固有のバージョンが分かります。)

d. 「プロトコル」タブで、「**追加 (Add)**」を選択します。 (「ネットワーク・プロトコルの選択 (Select Network Protocols)」画面が表示されます。)

e. **「ディスク使用 (Have Disk...)」**をクリックします。 (「ディスクの挿入 (Insert Disk)」画面が表示されます。)

f. 抽出したファイルを入れるドライブまたはディレクトリー (あるいはその両方) を入力します。

7. ServerNet ハードウェアが作動していることをテストします。使用できるテスト・プログラムはありません。ですから、DB2 を使って ServerNet ハードウェアをテストします。

ハードウェアが動作していないようであれば、 *ServerNet-I Virtual Interface Software Release Document (プロダクト ID N0031)* を参照してください。

DB2 ユニバーサル・データベース (Windows NT 版) のインストール方法と実装方法については、 *DB2 エンタープライズ拡張エディション (Windows 版) 概説およびインストール* を参照してください。

Synfinity Interconnect のセットアップ手順

この環境を設定するときに必要なハードウェアとソフトウェアのリストには、以下のプロダクトが含まれています。

- Synfinity PCI ネットワーク・インターフェース・カード
- Synfinity Six Port スイッチ
- Synfinity Interconnect ケーブル
- Synfinity Cluster Manager Software バージョン 1.10

DB2 ユニバーサル・データベースで Synfinity Interconnect を動作させるときに必要なステップを、これから示します。それぞれのステップは、各ステップで必要とされることの要旨です。ここでは、各ステップに関係するすべてが詳細に述べられていません。それぞれのステップで参照されている資料もご使用ください。そこには、詳細な手順と必要な指示が載せられています。

各 Synfinity システムは、 Synfinity Cluster Manager Software バージョン 1.10 CD-ROM と一緒に梱包されています。この CD-ROM には、 Synfinity Interconnect のセットアップに必要なすべての資料とソフトウェアが含まれています。さらに、 *Synfinity Cluster Manager Software User Guide* が含まれています。

他の VI ハードウェア、ソフトウェア、およびプロトコルがインストールされている場合は、これらをすべて削除してからでないと、 Synfinity Interconnect をインストールできないことがあります。

インストールが完了すると、 Synfinity Interconnect は外部ハードウェアとして認識されるようになり、 Windows NT コントロール パネルからは表示できなくなります。

ステップの要旨は以下のとおりです。

1. アダプター・カードをインストールします。
2. Synfinity Cluster Manager Software をインストールします。
3. スイッチとケーブルをインストールします。
4. Interconnect をテストします。

ステップは以下のとおりです。

1. Synfinity PCI Network Interface Card をインストールします。インストールの手順については、 *Synfinity Cluster Manager Software User Guide* を参照してください。

2. Switch に接続されているノード上に Synfinity Cluster Manager Software をインストールします。インストールの手順については、 *Synfinity Cluster User Guide* を参照してください。

   選択したノードが Cluster Manager になります。 CD からソフトウェアをインストールする必要があるのは、このノードだけです。

   インストールが完了したら、 Synfinity Cluster Manager ソフトウェアを実行する必要があります。 Cluster Manager からは、クラスター計画やネットワークを構成するのに役立つステップバイステップ・ガイドが示されます。ルーティングおよび配線に関する最善のオプションについてのアドバイスもあります。このステップを完了してからでないと、ケーブルを Synfinity スイッチやネットワーク・カードに接続することはできません。計画プロセスの一環として、 Cluster Manager はクラスター計画を参考にして、他のノードで使用するためのインストール可能ディスケットを作成します。このディスケットの中には、他のノード上のカードに使用できるドライバー・ソフトウェアが含まれます。詳細すべてについては、*Synfinity Cluster Manager Software User Guide* を参照してください。

3. Synfinity Switch および Cable をインストールします。インストールの手順については、 *Synfinity Cluster User Guide* を参照してください。
4. Synfinity ハードウェアが作動していることをテストします。このテストは、以下のようにして実行します。
   a. クラスター内の任意のシステムで、 Windows NT の「コマンド プロンプト (Command Prompt)」ウィンドウをオープンします。
   b. Synfinity Cluster Manager ソフトウェアをロードした ″utils″ サブディレクトリーへディレクトリーを変更します。
   c. ″vitest″ と入力し、表示されるノード番号をメモします。
   d. クラスター内の他のシステムで、「コマンド プロンプト (Command Prompt)」ウィンドウをオープンします。
   e. もう一方のシステムで Synfinity Cluster Manager ソフトウェアがロードされている ″utils″ サブディレクトリーへディレクトリーを変更します。
   f. ″vitest x″ と入力します。x は上記のステップ c でメモしたノード番号です。
   g. ″CONNECTION GOOD″ メッセージが表示されるはずです。
   h. ″NO CONNECTION″ メッセージが表示された場合は、ケーブルとハードウェアの設定を調べてください。また、問題のトラブルシューティングに関する詳細については、*Synfinity Cluster Manager Software User Guide* を参照してください。サポート Web ページ (http://www.fujitsu.com/) の ″Tech-tips″ もご覧ください。

DB2 ユニバーサル・データベース (Windows NT 版) のインストール方法と実装方法については、 *DB2 エンタープライズ拡張エディション (Windows 版) 概説およびインストール* を参照してください。

VI を使用して DB2 を実行できるようにする

詳細なインストール情報は、*DB2 エンタープライズ拡張エディション (Windows 版) 概説およびインストール* に載せられています。

DB2 エンタープライズ拡張エディション (Windows 版) 概説およびインストール に示されている手順に従って DB2 のインストールが完了したら、以下の DB2 レジストリー変数を設定して、インスタンス内のデータベース区画サーバーごとに以下のタスクを実行する必要があります。

1. DB2_VI_ENABLE=ON を設定する。

   レジストリー変数を変更するには **db2set** コマンドを使用します。インスタンス内のすべてのデータベース区画サーバーで **db2set** コマンドを実行するには、**db2_all** コマンドを使用します。 **db2_all** コマンドを実行するには、アドミニストレーターのメンバーであるユーザー・アカウントでログオンする必要があります。

   以下の例で、インスタンス内のすべてのデータベース区画サーバーで要求の並列稼働を許可するには、 ; 文字を二重引用符で囲みます。

   ```
   db2_all ";db2set DB2_VI_ENABLE=ON"
   ```

   db2_all コマンドに関する詳細は、 *管理の手引き: インプリメンテーション* の『複数データベース区画サーバーのコマンド実行』をご覧ください。

2. DB2_VI_DEVICE=nic0 を設定する。

   たとえば、次のようにします。

   ```
   db2_all ";db2set DB2_VI_DEVICE=nic0"
   ```

   注: Synfinity Interconnect の場合は、この変数を DB2_VI_DEVICE=VINIC に設定する必要があります。装置名は (VINIC) は、大文字でなければなりません。

3. DB2_VI_VIPL=vipl.dll を設定する。

   たとえば、次のようにします。

   ```
   db2_all ";db2set DB2_VI_VIPL=vipl.dll"
   ```

   注: 例で使用されている値は、レジストリー変数のデフォルトです。レジストリー変数について、詳しくは *管理の手引き: パフォーマンス* をご覧ください。

4. 区分インスタンスに対して db2start を入力する。
5. db2diag.log ファイルを検討する。区画ごとに、『VI is enabled』という 1 つのメッセージがあるはずです。
6. 高速コミュニケーション・マネージャー (FCM) 構成パラメーターを更新する必要が生じる場合もあります。 FCM に関連したリソース制約の結果として問題が発生した場合には、 FCM 構成パラメーターの値を大きくしなければなりません。他の高速相互接続環境から移行する場合、旧環境ですでに FCM 構成パラメーターの値を大きくしていたなら、逆にこれらの値を小さくしなければならないときもあります。また、Windows NT では、 DB2 デフォルトがオーバーライドされるように

DB2NTMEMSIZE レジストリー変数を設定しなければならない場合もあります。レジストリー変数について、詳しくは *管理の手引き: パフォーマンス* を参照してください。

付録J. Lightweight Directory Access Protocol (LDAP) ディレクトリー・サービス

Lightweight Directory Access Protocol (LDAP) は、ディレクトリー・サービスに対する業界標準のアクセス方式です。ディレクトリー・サービスとは、分散環境内にある複数のシステムおよびサービスについてのリソース情報を集めたリポジトリーです。クライアントとサーバーはディレクトリー・サービスを使用して、それらのリソースにアクセスします。各データベース・サーバーのインスタンスは自らの存在を LDAP サーバーに知らせるとともに、データベースの作成時にはデータベース情報を LDAP ディレクトリーへ送信します。クライアントがデータベースに接続すると、 LDAP ディレクトリーからそのサーバーのカタログ情報を取り出せます。各クライアントは、それぞれのマシンでローカルにカタログ情報を保管する必要はなくなります。クライアント・アプリケーションは、LDAP ディレクトリーの中で、データベースへ接続するのに必要な情報を探します。

キャッシュ・メカニズムが備えられているので、クライアントはローカルのディレクトリー・カタログの中で LDAP ディレクトリーを一度探すだけで済みます。情報を検索したら、その情報はローカル・マシンで格納されるか、キャッシュに入れられます。同じ情報へのその後のアクセスは、 *dir_cache* データベース管理プログラム構成パラメーターと、 DB2LDAPCACHE レジストリー変数に基づいて行われます。

- DB2LDAPCACHE=NO かつ *dir_cache*=NO の場合、情報は必ず LDAP から読み取られます。
- DB2LDAPCACHE=NO かつ *dir_cache*=YES の場合、 LDAP から一度情報が読み取られ、その情報は DB2 キャッシュに挿入されます。
- DB2LDAPCACHE=YES またはこの変数が設定されておらず、必要な情報がローカル・キャッシュの中になかった場合、情報は LDAP ディレクトリーから読み取られ、ローカル・キャッシュが更新されます。

注: DB2LDAPCACHE レジストリー変数が適用されるのは、データベースとノード・ディレクトリーだけです。

LDAP クライアント構成と LDAP サーバー構成のサポート

次の表では、サポートされる LDAP クライアント構成と LDAP サーバー構成を要約しています。

表 26. サポートされる LDAP クライアントおよびサーバー構成

	IBM SecureWay Directory V3.1 および V3.1.1	Microsoft Active Directory
IBM LDAP クライアント	サポートされる	サポートされない
Microsoft LDAP/ADSI クライアント	サポートされる	サポートされる

IBM SecureWay Directory バージョン 3.1 は、 Windows NT、AIX、および Solaris で使用可能な LDAP バージョン 3 サーバーです。 SecureWay ディレクトリーは、 AIX および AS/400 での基本オペレーティング・システムの一部として出荷され、 OS/390 Security Server との組み合わせで使用されます。

DB2 は、IBM LDAP クライアントを AIX、 Solaris、Windows NT、および Windows 98 でサポートしています。

Microsoft Active Directory は LDAP バージョン 3 サーバーです。 Windows 2000 Server オペレーティング・システムの一部として使用できます。

Microsoft LDAP クライアント・サポートは、以下の Microsoft 製品に組み込まれています。

- Outlook 98、Outlook 2000、または Outlook Express

  注: Outlook Express は、Microsoft Internet Explorer の一部としてインストールされます。
- Exchange Server
- Windows NT Server サービス・パック 4
- Windows 98 Second Edition
- Windows 2000

Microsoft LDAP クライアント・サポートは、 Active Directory Service Interface (ADSI) コンポーネントにも組み込まれています。最新バージョンの ADSI は、

http://www.microsoft.com/windows2000/techinfo/howitworks/activedirectory/adsilinks.asp

からダウンロードできます。

Windows 98、Windows NT、または Windows 2000 オペレーティング・システムで実行されている場合、 DB2 は IBM SecureWay Directory Server へのアクセス手段として、IBM LDAP クライアントまたは Microsoft LDAP クライアントの使用をサポートします。 Microsoft LDAP クライアントが使用できない場合、 DB2 は IBM LDAP クライ

アントの使用を試みます。 IBM LDAP クライアントを明示的に選択するには、**db2set** コマンドを使用して DB2LDAP_CLIENT_PROVIDER レジストリー変数を『IBM』に設定します。

Windows 2000 Active Directory のサポート

DB2 は Active Directory を次のようにして利用します。

1. DB2 データベース・サーバーは、 ibm_db2Node オブジェクトとして Active Directory に公開されます。 ibm_db2Node オブジェクト・クラスは、ServiceConnectionPoint (SCP) オブジェクト・クラスのサブクラスです。どの ibm_db2Node オブジェクトにもプロトコル構成情報が含まれていて、クライアント・アプリケーションが DB2 データベース・サーバーへ接続できるようになっています。新しいデータベースが作成されると、そのデータベースは ibm_db2Node の下にある ibm_db2Database オブジェクトとして、 Active Directory に公開されます。
2. リモート・データベースに接続するときは、 DB2 クライアントは LDAP インターフェースを使用して、 Active Directory に対して ibm_db2Database オブジェクトについての照会を行います。データベース・サーバーへ接続するためのプロトコル通信 (バインド情報) は、 ibm_db2Database オブジェクトが作成された ibm_db2Node オブジェクトから取得されます。

Active Directory を使用するよう DB2 を構成する

Microsoft Active Directory へアクセスするためには、以下の条件を満たしていなければなりません。

1. DB2 を実行するマシンが Windows 2000 ドメインに属していなければならない。
2. Microsoft LDAP クライアントがインストールされている。 Microsoft LDAP クライアントは Windows 2000 オペレーティング・システムの一部です。 Windows 98、または Windows NT の場合、 `wldap32.dll` がシステム・ディレクトリーの下にあるか確認する必要があります。
3. LDAP サポートが有効になっている。 Windows 2000 の場合、 LDAP サポートはインストール・プログラムによって有効になります。 Windows 98/NT の場合は、**db2set** コマンドを使用して DB2_ENABLE_LDAP レジストリー変数を『YES』に設定することにより、 LDAP を明示的に有効にしなければなりません。
4. DB2 を実行して Active Directory から情報を読み取るときは、ドメイン・ユーザー・アカウントにログオンする。

IBM LDAP 環境での DB2 の構成

DB2 を IBM LDAP 環境で使用できるようにするには、各マシンで次のような構成を行わなければなりません。

- LDAP サポートが有効になっている。 Windows 2000 の場合、 LDAP サポートはインストール・プログラムによって有効になります。 Windows 98/NT の場合は、

db2set コマンドを使用して DB2_ENABLE_LDAP レジストリー変数を『YES』に設定することにより、 LDAP を明示的に有効にしなければなりません。

- LDAP サーバーの TCP/IP ホスト名とポート番号を構成する。これらの値は DB2LDAPHOST 応答キーワードを使用して不在インストール時に入力することもできますし、 DB2SET コマンドを使用して後から手動設定することもできます。

  ```
  db2set DB2LDAPHOST=<hostname[:port]>
  ```

  ここで、hostname は LDAP サーバーの TCP/IP ホスト名、 [:port] はポート番号です。ポート番号が指定されなかった場合、 DB2 はデフォルトの LDAP ポート (389) を使用します。

  DB2 オブジェクトの検索は、LDAP 基本識別名 (baseDN) によって行われます。IBM SecureWay LDAP ディレクトリー・サーバー・バージョン 3.1 を使用している場合は、基本識別名を構成する必要はありません。 DB2 がこの情報をサーバーから動的に取得するからです。ただし、IBM eNetwork Directory Server バージョン 2.1 を使用している場合は、 DB2SET コマンドを使用することにより、各マシンで LDAP 基本識別名を構成しなければなりません。

  ```
  db2set DB2LDAP_BASEDN=<baseDN>
  ```

  ここで、baseDN は LDAP サーバーに定義されている LDAP 接尾部の名前です。この LDAP 接尾部は DB2 オブジェクトを収容するのに使用されます。

- LDAP ユーザーの識別名 (DN) とパスワードを構成する必要があります。ただし、これらが必要になるのは、 DB2 ユーザー固有情報を保管するのに LDAP を使用する計画がある場合だけです。

LDAP ユーザーの作成

DB2 では、 DB2 レジストリー変数と CLI 構成をユーザー・レベルで設定することがサポートされています。 (このことは AIX および Solaris プラットフォームにはあてはまりません。) ユーザー・レベルのサポートがあれば、マルチユーザー環境でユーザー固有の設定を行えます。 Windows NT Terminal Server はその一例です。ここでは、各ログオン・ユーザーが、システム環境や他のユーザーの環境を干渉することなく自分の環境をカスタマイズできます。

IBM LDAP ディレクトリーを使用するときは、 LDAP ユーザーを定義してからでないと、ユーザー・レベルの情報を LDAP に保管できません。 LDAP ユーザーを作成するには、以下のどちらかを実行します。

- ユーザー・オブジェクトのすべての属性を収容する LDIF ファイルを作成してから、LDIF インポート・ユーティリティーを実行し、そのオブジェクトを LDAP ディレクトリーにインポートします。 IBM LDAP サーバー用の LDIF ユーティリティーは『LDIF2DB』です。

- ディレクトリー管理ツール (DMT) を使用して、ユーザー・オブジェクトを作成します。このツールは、IBM SecureWay LDAP Directory Server バージョン 3.1 でしか使用できません。

ある人物のオブジェクトの属性を収容する LDIF ファイルが、次のように表示されます。

```
File name: newuser.ldif

dn: cn=Mary Burnnet, ou=DB2 UDB Development, ou=Toronto, o=ibm, c=ca
objectclass: ePerson
cn: Mary Burnnet
sn: Burnnet
uid: mburnnet
userPassword: password
telephonenumber: 1-416-123-4567
facsimiletelephonenumber: 1-416-123-4568
title: Software Developer
```

IBM LDIF インポート・ユーティリティーを使用して LDIF ファイルをインポートする LDIF コマンドの例を、以下に示します。

```
LDIF2DB -i newuser.ldif
```

注:

1. LDIF2DB コマンドは LDAP サーバー・マシンから実行しなければなりません。
2. 必要なアクセス制御リスト (ACL) を LDAP ユーザー・オブジェクトに授与することにより、 LDAP ユーザーが自分のオブジェクトの追加、削除、読み取り、書き込みを行えるようにしなければなりません。ユーザー・オブジェクトについての ACL を授与するには、 LDAP Directory Server Web Administration ツールを使用します。

DB2 アプリケーション用 LDAP ユーザーの構成

IBM LDAP クライアントを処理するときや DB2 を実行する前には、現行ログオン・ユーザーの LDAP ユーザー識別名 (DN) とパスワードを構成しなければなりません。これを行うには、db2ldcfg ユーティリティーを次のように使用します。

```
db2ldcfg -u <userDN> -w <password> -> set the user's DN and password
         -r                        -> clear the user's DN and password
```

たとえば、次のようになります。

```
db2ldcfg -u "cn=Mary Burnnet,ou=DB2 UDB Development,ou=Toronto,o=ibm,c=ca"
         -w password
```

インストール後の DB2 サーバーの登録

DB2 サーバー・インスタンスに接続する際に、クライアント・アプリケーションで使うプロトコル構成情報を公表するためには、各 DB2 サーバー・インスタンスを LDAP に登録しておかなければなりません。データベース・サーバーのインスタンスを登録するときには、*ノード名* を指定する必要があります。このノード名は、クライアント・アプリケーションがサーバーに接続するときに、そのクライアント・アプリケーションによって使われます。 CATALOG LDAP NODE コマンドを使用して、もう一つ、 LDAP ノードの別名をカタログすることができます。

注: Windows 2000 ドメイン環境で作業している場合は、インストール時に DB2 サーバー・インスタンスが、次の情報とともに、自動的に Active Directory へ登録されます。

```
nodename: TCP/IP hostname
protocol type: TCP/IP
```

TCP/IP ホスト名が 9 文字以上の場合は、8 文字に切り捨てられます。

REGISTER コマンドは次のようになります。

```
db2 register db2 server in ldap
   as <ldap_node_name>
   protocol tcpip
```

`protocol` 文節には、このデータベース・サーバーに接続するときに使用する通信プロトコルを指定します。

複数の物理マシンを含む DB2 ユニバーサル・データベース EEE のインスタンスを作成する場合、それぞれのマシンで一度 REGISTER コマンドを呼び出す必要があります。 *rah* コマンドは、すべてのマシンで REGISTER コマンドを発行する際に使用します。

注: LDAP では名前は固有でなければならないので、各マシンに同じ `ldap_node_name` を使うことはできません。 REGISTER コマンドでは、各マシンのホスト名を、 `ldap_node_name` に置き換えることができます。たとえば、次のようにします。

```
rah ">DB2 REGISTER DB2 SERVER IN LDAP AS <> PROTOCOL TCPIP"
```

『<>』には、`rah` コマンドを実行している各マシンのホスト名が入ります。めったにありませんが、複数の DB2 ユニバーサル・データベース EEE インスタンスが存在する場合、 *rah* コマンドでは、インスタンスとホスト索引の組み合わせをノード名として使用することができます。

REGISTER コマンドは、リモート DB2 サーバーに対して発行できます。そのためには、リモート・サーバーの登録時に、リモート・コンピューター名、インスタンス名、およびプロトコル構成パラメーターを指定しなければなりません。このコマンドは、次のように使います。

```
db2 register db2 server in ldap
   as <ldap_node_name>
   protocol tcpip
   hostname <host_name>
   svcename <tcpip_service_name>
   remote <remote_computer_name>
   instance <instance_name>
```

コンピューター名には、次の規則が使われます。

- TCP/IP を構成する場合、コンピューター名は TCP/IP ホスト名と同じでなければならない。
- APPN を構成する場合、パートナー LU 名をコンピューター名として使用する。

高可用性の環境あるいはフェールオーバーの環境で実行し、通信プロトコルとして TCP/IP を使用する場合、クラスター の IP アドレスを使わなければなりません。クラスターの IP アドレスを使用するなら、クライアントは、マシンごとに個別の TCP/IP ノードをカタログすることなく、いずれかのマシンでサーバーに接続することができます。 このクラスターの IP アドレスは、次に示すように、 `hostname` 文節を使って指定します。

```
db2 register db2 server in ldap
   as <ldap_node_name>
   protocol tcpip
   hostname n.nn.nn.nn
```

`n.nn.nn.nn` はクラスターの IP アドレスです。

REGISTER コマンドの詳細は、*コマンド解説書* を参照してください。

DB2 サーバー用のプロトコル情報を更新する

LDAP 内の DB2 サーバー情報は、最新のものにしておく必要があります。たとえば、プロトコル構成パラメーターまたはサーバーのネットワーク・アドレスを変更するときには、 LDAP を更新しなければなりません。

ローカル・マシンの LDAP にある DB2 サーバーを更新するには、次のコマンドを使用します。

```
db2 update ldap ...
```

更新可能なプロトコル構成パラメーターの例としては、以下のものがあります。

- TCP/IP ホスト名およびサービス名パラメーターまたはポート番号パラメーター

- IPX アドレス
- TP 名、パートナー LU、あるいはモードのような、APPC プロトコル情報
- NetBIOS ワークステーション名

リモート DB2 サーバーのプロトコル構成パラメーターを更新するには、 node 文節を指定した UPDATE LDAP コマンドを使います。

```
db2 update ldap
   node <node_name>
   hostname <host_name>
   svcename <tcpip_service_name>
```

UPDATE LDAP について詳しくは、*コマンド解説書* を参照してください。

接続のためにノード別名をカタログする

DB2 サーバーのノード名については、サーバーを LDAP に登録するときに、指定しなければなりません。アプリケーションはこのノード名を使用して、データベース・サーバーに接続します。ノード名がアプリケーション内でハード・コーディングされる場合のように、別のノード名が必要であれば、CATALOG LDAP NODE コマンドを使用して変更してください。 このコマンドは、次のようになります。

```
db2 catalog ldap node <ldap_node_name>
   as <new_alias_name>
```

LDAP ノードをアンカタログするには、UNCATALOG LDAP NODE コマンドを使用します。このコマンドは、次のようになります。

```
db2 uncatalog ldap node <ldap_node_name>
```

DB2 サーバーの登録を解除する

LDAP から特定のインスタンスの登録を解除すると、そのインスタンスを参照するすべてのノード、あるいは、そのインスタンスを参照する別名、オブジェクト、およびデータベース・オブジェクトも除去されます。

ローカルまたはリモート・マシンのいずれでも、DB2 サーバーの登録を解除する場合には、そのサーバーの LDAP ノード名を指定する必要があります。

```
db2 deregister db2 server in ldap
   node <node_name>
```

DB2 サーバーの登録を解除すると、 DB2 サーバーのその同じインスタンスを参照する LDAP ノード項目および LDAP データベース項目も、すべてアンカタログされます。

データベースの登録

インスタンス内でデータベースを作成するときに、そのデータベースは LDAP へ自動的に登録されます。登録することにより、クライアント・マシンでデータベースおよびノードをカタログせずに、データベースへリモート・クライアント接続できるようになります。 クライアントがデータベースへ接続しようとする場合に、ローカル・マシンのデータベース・ディレクトリーにそのデータベースが存在していなければ、 LDAP ディレクトリーが検索されます。

LDAP ディレクトリーにその名前が存在する場合、データベースはローカル・マシンで作成されますが、 LDAP ディレクトリーにおける名前の競合を警告する警告メッセージが戻されます。このため、LDAP ディレクトリーでは、手動でデータベースをカタログすることができます。ユーザーは、CATALOG LDAP DATABASE コマンドを使用して、リモート・サーバーのデータベースを LDAP に登録できます。リモート・データベースを登録するときには、リモート・データベース・サーバーを表す LDAP ノードの名前を指定します。リモート・データベース・サーバーは、データベースを登録する**前に**、 REGISTER DB2 SERVER IN LDAP コマンドを使用して、 LDAP に登録**しなければなりません**。

データベースを手動で LDAP に登録するには、CATALOG LDAP DATABASE コマンドを使います。

```
db2 catalog ldap database <dbname>
   at node <node_name>
   with "My LDAP database"
```

リモート・サーバーへ接続する

LDAP 環境では、ATTACH コマンドで LDAP ノード名を使用することにより、リモート・データベース・サーバーへ接続することができます。

```
db2 attach to <ldap_node_name>
```

クライアント・アプリケーションが特定のノードまたはデータベースに初めて接続する場合、そのノードはローカル・ノード・ディレクトリーに含まれていないので、 DB2 は LDAP ディレクトリーの中でその宛先ノード項目を探します。 LDAP ディレクトリーでその項目が見つかったら、リモート・サーバーのプロトコル情報が取り出されます。データベースに接続し、LDAP ディレクトリーでその項目が見つかったら、データベース情報が取り出されます。この情報を使用して、 DB2 はローカル・マシンのデータベース項目およびノード項目を自動的にカタログします。次回にクライアント・アプリケーションが同じノードまたはデータベースに接続するときには、ローカル・データベース・ディレクトリーの情報が使われるので、 LDAP ディレクトリーを検索する必要はありません。

詳細情報: キャッシュ・メカニズムが備えられているので、クライアントはローカルのディレクトリー・カタログの中で LDAP ディレクトリーを一度探すだけで済みます。

情報を検索したら、その情報はローカル・マシンで格納されるか、キャッシュに入れられます。同じ情報へのその後のアクセスは、 *dir_cache* データベース管理プログラム構成パラメーターと、 DB2LDAPCACHE レジストリー変数に基づいて行われます。

- DB2LDAPCACHE=NO かつ *dir_cache*=NO の場合、情報は必ず LDAP から読み取られます。
- DB2LDAPCACHE=NO かつ *dir_cache*=YES の場合、 LDAP から一度情報が読み取られ、その情報は DB2 キャッシュに挿入されます。
- DB2LDAPCACHE=YES またはこの変数が設定されておらず、必要な情報がローカル・キャッシュの中になかった場合、情報は LDAP ディレクトリーから読み取られ、ローカル・キャッシュが更新されます。

注: LDAP 情報をキャッシュに入れることは、ユーザー・レベルの CLI や DB2 プロファイル・レジストリー変数にはあてはまりません。また、データベース、ノード、および DCS ディレクトリー用の『メモリー内』キャッシュも存在しますが、ノード・ディレクトリーだけのためのキャッシュは存在しません。

データベースの登録を解除する

以下の場合に、データベースは LDAP から自動的に登録解除されます。

- データベースを除去した場合。
- 所有インスタンスが LDAP から登録解除された場合。

データベースは、次のようにして、LDAP から手動で登録解除することができます。

```
db2 uncatalog ldap database <dbname>
```

ローカル・データベースおよびノード・ディレクトリーの LDAP 項目を最新表示する

LDAP 情報は変更されることがあるため、ローカルおよびノード・ディレクトリーの LDAP 項目を最新表示する必要があります。 LDAP 内の項目をキャッシュするには、ローカル・データベースおよびノード・ディレクトリーが使われます。

詳細情報: キャッシュ・メカニズムが備えられているので、クライアントはローカルのディレクトリー・カタログの中で LDAP ディレクトリーを一度探すだけで済みます。情報を検索したら、その情報はローカル・マシンで格納されるか、キャッシュに入れられます。同じ情報へのその後のアクセスは、 *dir_cache* データベース管理プログラム構成パラメーターと、 DB2LDAPCACHE レジストリー変数に基づいて行われます。

- DB2LDAPCACHE=NO かつ *dir_cache*=NO の場合、情報は必ず LDAP から読み取られます。
- DB2LDAPCACHE=NO かつ *dir_cache*=YES の場合、 LDAP から一度情報が読み取られ、その情報は DB2 キャッシュに挿入されます。

- DB2LDAPCACHE=YES またはこの変数が設定されておらず、必要な情報がローカル・キャッシュの中になかった場合、情報は LDAP ディレクトリーから読み取られ、ローカル・キャッシュが更新されます。

注: LDAP 情報をキャッシュに入れることは、ユーザー・レベルの CLI や DB2 プロファイル・レジストリー変数にはあてはまりません。また、データベース、ノード、および DCS ディレクトリー用の『メモリー内』キャッシュも存在しますが、ノード・ディレクトリーだけのためのキャッシュは存在しません。

LDAP リソースを参照するデータベース項目を最新表示するには、次のコマンドを使用します。

```
db2 refresh ldap database directory
```

LDAP リソースを参照するローカル・マシンのノード項目を最新表示するには、次のコマンドを使います。

```
db2 refresh ldap node directory
```

最新表示の一環として、ローカル・データベースおよびノード・ディレクトリーに保管されているすべての LDAP 項目が除去されます。次回にアプリケーションがデータベースまたはノードにアクセスすると、情報を LDAP から直接に読み取り、ローカル・データベースまたはノード・ディレクトリーに新しい項目を生成します。

最新表示を周期的に実行するには、以下のことを行えます。

- 周期的に実行する最新表示をスケジュールする。
- システムのブート時に REFRESH コマンドを実行する。
- 使用可能な管理パッケージを使い、すべてのクライアント・マシンで REFRESH コマンドを呼び出す。
- DB2LDAPCACHE=『NO』を設定して、 LDAP 情報がデータベース、ノード、および DCS ディレクトリーのキャッシュに入れられないようにする。

検索

DB2 は、現行の LDAP ディレクトリー区画を検索します (Windows 2000 環境の場合は、現行の Active Directory ドメインを検索します)。複数の LDAP ディレクトリー区画またはドメインがある環境では、検索の範囲を設定することができます。たとえば、現在の区画またはドメインに情報がなければ、他の区画またはドメインすべての自動検索を要求できます。一方、検索の範囲については、ローカル・マシンだけの検索に制限することも可能です。

この検索範囲は、 DB2 プロファイルのレジストリー変数 DB2LDAP_SEARCH_SCOPE によって制御されます。 LDAP において、検索範囲の値をグローバル・レベルで設定するには、 *db2set* コマンドで『-gl』オプションを使用します。これは『LDAP 内でグローバル』ということです。

```
db2set -gl db2ldap_search_scope=<value>
```

可能な値には、『local』、『domain』、あるいは『global』があります。デフォルト値は『domain』です。この値は、検索範囲を現行のディレクトリー区画に限定します。
LDAP での検索範囲を設定すると、組織全体のために、デフォルトの検索範囲を設定できるようになります。たとえば、新しいデータベースを作成したら、検索範囲を『global』に初期設定することができます。このように設定すれば、どのクライアント・マシンであっても、他のすべての区画またはドメインを検索して、その中から特定の区画またはドメインで定義されているデータベースを見つけられます。最初にそれぞれのクライアントに接続してから、各マシンに項目を記録したら、検索範囲を『local』に変更することができます。『local』に変更すると、各クライアントはどの区画またはドメインも走査しません。

注: DB2 プロファイルのレジストリー変数 DB2LDAP_SEARCH_SCOPE は、 LDAP 内におけるグローバル・レベルでの変数の設定をサポートする唯一のレジストリー変数です。

ホスト・データベースの登録

ホスト・データベースを LDAP で登録するときは、 2 通りの構成が可能です。

- ホスト・データベースへの直接接続、または
- ホスト・データベースへのゲートウェイ経由の接続

前者の場合、ユーザーはホスト・サーバーを LDAP に登録してから、そのホスト・サーバーのノード名を指定してホスト・データベースを LDAP でカタログします。後者の場合、ユーザーはゲートウェイ・サーバーを LDAP に登録してから、そのゲートウェイ・サーバーのホスト名を指定して LDAP でホスト・データベースをカタログします。

上記の 2 つの場合の例として、次のような状況を考慮してみましょう。
NIAGARA_FALLS という名前のホスト・データベースがあるとします。このホスト・データベースは、 APPN と TCP/IP を使用して着信接続を受け入れることができます。あるクライアントが DB2 コネクトをインストールしていないためにそのホストへ直接接続できない場合は、『goto@niagara』という名前のゲートウェイを使用してホストに接続します。

以下のステップを実行する必要があります。

1. APPN 接続を可能にするため、ホスト・データベース・サーバーを LDAP に登録します。 REMOTE 文節と INSTANCE 文節は任意です。 NODETYPE 文節を『DCS』に設定して、これがホスト・データベース・サーバーであることを示します。

   ```
   db2 register ldap as nfappn appn network CAIBMOML partnerlu NFLU
      mode IBMRDB remote mvssys instance msvinst nodetype dcs
   ```

2. TCP/IP 接続を可能にするため、ホスト・データベース・サーバーを LDAP に登録します。このサーバーの TCP/IP ホスト名は『myhost』、ポート番号は『446』です。ステップ 1 と同じように、NODETYPE 文節を『DCS』に設定して、これがホスト・データベース・サーバーであることを示します。

```
db2 register ldap as nftcpip tcpip hostname myhost svcename 446
   remote mvssys instance mvsinst nodetype dcs
```

3. TCP/IP 接続を可能にするため、DB2 コネクト・ゲートウェイ・サーバーを LDAP に登録します。このゲートウェイ・サーバーの TCP/IP ホスト名は『niagara』、ポート番号は『50000』です。

```
db2 register ldap as whasf tcpip hostname niagara svcename 50000
   remote niagara instance goto nodetype server
```

4. TCP/IP 接続を使用してホスト・データベースを LDAP でカタログします。このホストのデータベース名は『NIAGARA_FALLS』、データベース別名は『nftcpip』です。 GWNODE 文節は、DB2 コネクト・ゲートウェイ・サーバーのノード名を指定するために使用されています。

```
db2 catalog ldap database NIAGARA_FALLS as nftcpip at node nftcpip
   gwnode whasf authentication dcs
```

5. APPN 接続を使用してホスト・データベースを LDAP でカタログします。

```
db2 catalog ldap database NIAGARA_FALLS as nfappn at node nfappn
   gwnode whasf authentication dcs
```

上記のように登録とカタログが完了した後、 TCP/IP を使用してホストに接続したいのであれば、『nftcpip』に接続します。 APPN を使用してホストに接続したいのであれば、『nfappn』に接続します。 DB2 コネクトがクライアント・ワークステーションにインストールされていない場合、接続は TCP/IP によってゲートウェイ経由で確立されます。そこからホストまでの部分については、『nftcpip』を使用する場合は TCP/IP によって、『nfappn』を使用する場合は APPN によって接続されます。

通常は、LDAP においてホスト・データベース情報を手動で構成できるので、それぞれのクライアントでは、各マシンのデータベースおよびノードをローカルに手動でカタログする必要はありません。次のように処理します。

1. ホスト・データベース・サーバーを LDAP に登録します。そのためには、ホスト・データベース・サーバーのリモート・コンピューター名、インスタンス名、およびノード・タイプを、 REGISTER コマンドの REMOTE 文節、INSTANCE 文節、および NODETYPE 文節にそれぞれ指定しなければなりません。 REMOTE 文節は、ホスト・サーバー・マシンのホスト名と LU 名のどちらに設定しても構いません。 INSTANCE 文節は、8 文字以下の文字ストリングで設定します。 (たとえば、インスタンス名を『DB2』のように設定します。) NODE TYPE 文節は必ず『DCS』に設定して、これがホスト・データベース・サーバーであることを示さなければなりません。

2. CATALOG LDAP DATABASE コマンドを使ってホスト・データベースを LDAP に登録します。さらに、PARMS 文節を使用して、任意の DRDA パラメーターを指定することができます。データベース認証タイプは『DCS』に設定するようにします。

ユーザー・レベルで DB2 レジストリー変数を設定する

LDAP 環境では、DB2 プロファイルのレジストリー変数をユーザー・レベルで設定できます。これにより、それぞれの DB2 環境をカスタマイズすることができます。 DB2 プロファイルのレジストリー変数をユーザー・レベルで設定するには、 -ul オプションを使います。

```
db2set -ul <variable>=<value>
```

注: これは AIX や Solaris ではサポートされません。

DB2 はキャッシュ・メカニズムを備えています。 DB2 プロファイルのユーザー・レベルのレジストリー変数は、ローカル・マシンにキャッシュされます。 -ul パラメーターを指定すると、DB2 は、いつもキャッシュから DB2 レジストリー変数を読み取ります。キャッシュは、以下の場合に更新されます。

- DB2 レジストリー変数をユーザー・レベルで更新またはリセットした場合。
- LDAP プロファイル変数をユーザー・レベルで更新するためのコマンドは、次のとおりです。

  ```
  db2set -ur
  ```

インストールの完了後に LDAP サポートを使用可能にする

インストールの作業が完了した後のある時点で LDAP サポートを使用可能にするには、各マシンで以下のような手順を使用します。

- LDAP サポートのバイナリー・ファイルをインストールします。セットアップ・プログラムを実行し、**カスタム・インストール**から、 LDAP Directory Exploitation サポートを選択します。セットアップ・プログラムによりバイナリー・ファイルがインストールされ、 DB2 プロファイルのレジストリー変数 DB2_ENABLE_LDAP が『YES』にセットされます。

  注: Windows 98/NT および UNIX プラットフォームの場合は、 **db2set** コマンドを使用して DB2_ENABLE_LDAP レジストリー変数を『YES』に設定することにより、 LDAP を明示的に有効にしなければなりません。

- (UNIX プラットフォームの場合のみ) 以下のコマンドを使用して、 LDAP サーバーの TCP/IP ホスト名と (オプショナル) ポート番号を宣言します。

  ```
  db2set DB2LDAPHOST=<base_domain_name>[:port_number]
  ```

ここで、base_domain_name は LDAP サーバーの TCP/IP ホスト名、 [:port] はポート番号です。ポート番号が指定されなかった場合、 DB2 はデフォルトの LDAP ポート (389) を使用します。

DB2 オブジェクトの検索は、LDAP 基本識別名 (baseDN) によって行われます。IBM SecureWay LDAP ディレクトリー・サーバー・バージョン 3.1 を使用している場合は、基本識別名を構成する必要はありません。 DB2 がこの情報をサーバーから動的に取得するからです。ただし、IBM eNetwork Directory Server バージョン 2.1 を使用している場合は、 DB2SET コマンドを使用することにより、各マシンで LDAP 基本識別名を構成しなければなりません。

```
db2set DB2LDAP_BASEDN=<baseDN>
```

ここで、baseDN は LDAP サーバーに定義されている LDAP 接尾部の名前です。この LDAP 接尾部は DB2 オブジェクトを収容するのに使用されます。

- REGISTER LDAP AS コマンドを使用し、 DB2 サーバーの現在のインスタンスを LDAP に登録します。たとえば、次のようにします。

  ```
  db2 register ldap as <node-name> protocol tcpip
  ```

- LDAP に登録したいデータベースがあれば、CATALOG LDAP DATABASE コマンドを実行します。たとえば、次のようにします。

  ```
  db2 catalog ldap database <dbname> as <alias_dbname>
  ```

- LDAP ユーザーの識別名 (DN) とパスワードを入力します。ただし、これらが必要になるのは、 DB2 ユーザー固有情報を保管するのに LDAP を使用する計画がある場合だけです。

LDAP サポートを使用不可にする

LDAP サポートを使用不可にするには、以下の手順を使います。

- DB2 サーバーのインスタンスごとに、DB2 サーバーを LDAP から登録解除します。

  ```
  db2 deregister db2 server in ldap node <nodename>
  ```

- DB2 プロファイルのレジストリー変数 DB2_ENABLE_LDAP を『NO』にセットします。

LDAP サポートと DB2 コネクト

LDAP サポートが DB2 コネクト・ゲートウェイにおいて有効で、データベースがゲートウェイ・データベース・ディレクトリーでは見つからなかった場合、 DB2 は LDAP を検索し、見つかった情報を保持しておこうとします。

機密保護についての考慮事項

LDAP ディレクトリーの情報にアクセスする前に、アプリケーションまたはユーザーは、LDAP サーバーによって認証されます。この認証作業のことを、LDAP サーバーに対するバインド といいます。

匿名のユーザーが LDAP ディレクトリーに格納されている情報を追加、削除、あるいは変更してしまわないように、その情報にアクセス制御を設定することは重要です。

アクセス制御は、デフォルトでコンテナー・レベルで継承され、適用できるものです。新しいオブジェクトが作成される場合、その新しいオブジェクトは、親オブジェクトと同じ機密保護属性を継承します。コンテナー・オブジェクトにアクセス制御を定義するときには、 LDAP サーバーで使用できる管理ツールを使えます。

デフォルトでは、アクセス制御は以下のように定義されています。

- LDAP 内のデータベースおよびノード項目については、すべて (つまり任意の匿名ユーザー) が読み取りアクセス権限を持っています。読み取り / 書き込みアクセス権限を持っているのは、ディレクトリー管理者と、そのオブジェクトの所有者または作成者だけです。
- ユーザー・プロファイルについては、プロファイル所有者と、ディレクトリー管理者が読み取り / 書き込みアクセス権限を持っています。他のユーザーのプロファイルにアクセスしようとしているユーザーは、ディレクトリー管理者権限を持っていなければ、そこにアクセスできません。

注: 認証検査は必ず LDAP サーバーによって実行されます。 DB2 によって実行されることはありません。 LDAP 認証検査は DB2 認証とは関係ありません。 SYSADM 権限を持つアカウントまたは許可 ID であっても、 LDAP ディレクトリーに対するアクセス権は持っていない可能性があります。

LDAP コマンドまたは API を実行するときに、バインド識別名 (bindDN) とパスワードを指定しなかった場合、 DB2 はデフォルトの証明書を使用して LDAP サーバーにバインドします。この証明書の権限では要求したコマンドを実行できない場合もあります。その場合、エラーが戻されます。

DB2 コマンドまたは API では、 USER 文節と PASSWORD 文節を使用してユーザーの bindDN とパスワードを明示的に指定できます。 DB2 コマンドについて詳しくは *コマンド解説書* を、 DB2 API について詳しくは *管理 API 解説書* を参照してください。

Windows 2000 Active Directory の機密保護についての考慮事項

Active Directory において、DB2 データベースとノードの各オブジェクトは、 DB2 サーバーがインストールされているマシンのコンピューター・オブジェクトの下に作成されます。データベース・サーバーを Active Directory に登録したり、データベースのカ

タログを Active Directory に作成するためには、コンピューター・オブジェクトの下にあるオブジェクトを作成したり更新したりできるだけのアクセス権限が必要です。

デフォルトでは、コンピューター・オブジェクトの下にあるオブジェクトは、認証を受けたすべてのユーザーが読み取ることができます。管理者 (Administrators、Domain Administrators、および Enterprise Administrators グループに属するユーザー) であれば更新することもできます。特定のユーザーまたはグループにアクセス権限を授与するには、 *Active Directory Users and Computer* 管理コンソール (MMC) を次のようにして使います。

1. *Active Directory Users and Computer* 管理ツールを開始します。
   (「スタート」—>「プログラム」—>「管理ツール (Administration Tools)」—>「Active Directory Users and Computer」)
2. 「*表示 (View)*」から「*拡張機能 (Advanced Features)*」を選択します。
3. 「*コンピューター (Computers)*」コンテナーを選択します。
4. DB2 がインストールされているサーバー・マシンを表すコンピューター・オブジェクトの上を右クリックし、「*プロパティー (Properties)*」を選択します。
5. 「*機密保護 (Security)*」タブを選択し、指定したユーザーまたはグループに必要なアクセス権限を追加します。

ユーザー・レベルの DB2 レジストリー変数および CLI 設定は、ユーザー・オブジェクトの下にある DB2 プロパティー・オブジェクトで保守されます。 DB2 レジストリー変数または CLI 設定をユーザー・レベルで設定するためには、そのユーザーに「ユーザー (User)」オブジェクトの下にオブジェクトを作成できるだけのアクセス権限が必要です。

デフォルトでは、「ユーザー (User)」オブジェクトの下にオブジェクトを作成できるのは管理者だけです。 DB2 レジストリー変数または CLI 設定をユーザー・レベルで設定できるアクセス権限をユーザーに授与するには、 *Active Directory Users and Computer* 管理コンソール (MMC) を次のようにして使います。

1. *Active Directory Users and Computer* 管理ツールを開始します。
   (「スタート」—>「プログラム」—>「管理ツール (Administration Tools)」—>「Active Directory Users and Computer」)
2. 「ユーザー (Users)」コンテナーの下にあるユーザー・オブジェクトを選択します。
3. ユーザー・オブジェクトの上を右クリックし、「*プロパティー (Properties)*」を選択します。
4. 「*機密保護 (Security)*」タブを選択します。
5. 「追加 (Add)」ボタンを使用して、ユーザー名をリストに追加します。
6. 「書き込み (Write)」および「すべての子オブジェクトの作成 (Create All Child Objects)」アクセスを付与します。

7. 「拡張 (Advanced)」設定を使用して、「このオブジェクトとすべての子オブジェクト (This object and all child objects)」に適用される許可を設定します。
8. 「継承可能な許可を親からこのオブジェクトへ継承するのを許可する (Allow inheritable permissions from parent to propagate to this object)」チェック・ボックスを選択します。

DB2 オブジェクト・クラスおよび属性によってディレクトリー・スキーマを拡張する

LDAP ディレクトリー・スキーマは、オブジェクト・クラスおよび属性を定義し、情報を LDAP ディレクトリー項目に格納します。オブジェクト・クラスは、一連の必須および任意指定属性で構成されています。 LDAP ディレクトリーの各項目には、それぞれに関連付けられたオブジェクト・クラスがあります。

DB2 が情報を LDAP へ格納する前に、LDAP サーバーのディレクトリー・スキーマには、 DB2 で使用するオブジェクト・クラスと属性が含まれていなければなりません。新しいオブジェクト・クラスおよび属性を基本スキーマに追加する作業のことを、ディレクトリー・スキーマの拡張といいます。

注: IBM SecureWay LDAP Directory v3.1 を使っている場合、 DB2 によって必要とされるオブジェクト・クラスおよび属性はすべて、基本スキーマに含まれています。基本スキーマを、DB2 オブジェクト・クラスと属性で拡張する必要はありません。

IBM eNetwork Directory バージョン 2.1 でディレクトリー・スキーマを拡張する

IBM eNetwork Directory バージョン 2.1 を使用している場合、 DB2 で使われるオブジェクト・クラスおよび属性で基本スキーマを拡張しなければなりません。

IBM eNetwork Directory バージョン 2.1 の基本スキーマを拡張するときは、以下のステップを使います。

1. DB2 属性定義ファイル `db2.at` とオブジェクト・クラス定義ファイル `db2.oc` を、システム属性定義ファイル `slapd.at.conf` とオブジェクト・クラス定義ファイル `slapd.oc.conf` があるディレクトリーにコピーします。 DB2 属性およびオブジェクト・クラス定義ファイルは、 `sqllib` サブディレクトリーの `cfg` サブディレクトリーにあります。システム属性およびオブジェクト・クラス定義ファイルは、 `%LDAPHome%` サブディレクトリーの `etc` サブディレクトリーにあります。
2. DB2 属性およびオブジェクト・クラス定義ファイルを検討します。現在の LDAP ディレクトリー・スキーマで定義されているオブジェクト・クラスおよび属性があれば、すべてコメント化します。
3. 次のようにして、`slapd.oc.conf` ファイルの最後に 1 行追加します。

   ```
   include db2.oc
   ```

4. 次のようにして、`slapd.at.conf` ファイルの最後に 1 行追加します。

```
include db2.at
```

5. LDAP サーバーを再始動します。

Windows 2000 Active Directory でディレクトリー・スキーマを拡張する

DB2 が情報を Windows 2000 Active Directory に保管できるようにするためには、ディレクトリー・スキーマを拡張して、新しい DB2 オブジェクト・クラスおよび属性を組み込めるようにする必要があります。新しいオブジェクト・クラスおよび属性をディレクトリー・スキーマに追加する作業のことを、スキーマの*拡張* といいます。

Windows 2000 ドメインの一部となっているマシンに DB2 を最初にインストールする前に、 DB2 スキーマ・インストール・プログラム **db2schex** を実行して、 Active Directory のスキーマを拡張しておかなければなりません。

db2schex プログラムは、当製品の CD-ROM 中にあります。 `db2` ディレクトリーの下の `common` サブディレクトリーにあります。たとえば、次のようになります。

```
x:¥db2¥common
```

ここで、`x:` は CD-ROM ドライブです。

コマンドは、次のように使用します。

```
db2schex
```

このコマンドに関連して、他にも次の任意指定文節があります。

- -b UserDN

  ユーザーの識別名を指定します。

- -w Password

  バインド・パスワードを指定します。

- -u

  スキーマをアンインストールします。

- -k

  エラーを無視して、アンインストールを強制的に続行します。

注:

1. UserDN とパスワードを指定しなかった場合、 **db2schex** は現在ログオンしているユーザーが指定されたものとしてバインドします。
2. userDN 文節には Windows NT ユーザー名を指定できます。
3. スキーマを更新するためには、 Schema Administrators グループのメンバーであるか、スキーマを更新するための権限を委任されている必要があります。

次に例を示します。

- DB2 スキーマをインストールする方法:

```
db2schex
```

- DB2 スキーマをインストールし、バインド DN とパスワードを指定する方法:

```
db2schex -b "cn=A Name,dc=toronto1,dc=ibm,dc=com"
         -w password
```

または

```
db2schex -b Administrator -w password
```

- DB2 スキーマをアンインストールする方法:

```
db2schex -u
```

- DB2 スキーマをアンインストールし、エラーは無視する方法:

```
db2schex -u -k
```

Active Directory 用の DB2 スキーマ・インストール・プログラムは、以下のタスクを実行します。

注:

1. どのサーバーが Schema Master であるかを検出します。
2. Schema Master となっているドメイン・コントローラーにバインドします。
3. スキーマにクラスや属性を追加する権限をユーザーに授与します。
4. Schema Master を書き込み可能にします (つまり、レジストリー内の安全インターロックを取り除きます)。
5. すべての新しい属性を作成します。
6. すべての新しいオブジェクト・クラスを作成します。
7. エラーがあるかどうかを検出し、もしある場合にはスキーマに加えられた変更をロールバックするようプログラムに指示します。

Windows 2000 Active Directory 内の DB2 オブジェクト

DB2 は、Active Directory 内のオブジェクトを以下の 2 箇所に作成します。

1. DB2 データベースおよびノード・オブジェクトは、 DB2 サーバーがインストールされているマシンのコンピューター・オブジェクトの下に作成されます。 Windows NT ドメインに属していない DB2 サーバー・マシンの場合は、 DB2 データベースおよびノード・オブジェクトは「システム (System)」コンテナーの下に作成されます。
2. ユーザー・レベルの DB2 レジストリー変数および CLI 設定は、「ユーザー (User)」オブジェクトの下にある DB2 プロパティー・オブジェクトに保管されます。これらのオブジェクトには、そのユーザーに固有の情報が入ります。

DB2 で使用するオブジェクト・クラスおよび属性

以下の表では、DB2 で使用するオブジェクト・クラスについて説明しています。

表 27. *cimManagedElement*

クラス	**cimManagedElement**
Active Directory LDAP 表示名	適用されない
Active Directory 共通名 (cn)	適用されない
説明	IBM Schema で使用される多くのシステム管理オブジェクト・クラスの基本クラスを提供します。
SubClassOf	最上位
必須属性	
任意選択属性	説明
タイプ	抽象型
OID (オブジェクト ID)	1.3.18.0.2.6.132
GUID (グローバル固有 ID)	b3afd63f-5c5b-11d3-b818-002035559151

表 28. *cimSetting*

クラス	**cimSetting**
Active Directory LDAP 表示名	適用されない
Active Directory 共通名 (cn)	適用されない
説明	IBM Schema での構成および設定に使用する基本クラスを提供します。
SubClassOf	cimManagedElement
必須属性	
任意選択属性	settingID
タイプ	抽象型
OID (オブジェクト ID)	1.3.18.0.2.6.131
GUID (グローバル固有 ID)	b3afd64d-5c5b-11d3-b818-002035559151

表 29. *eProperty*

クラス	**eProperty**
Active Directory LDAP 表示名	ibm-eProperty
Active Directory 共通名 (cn)	ibm-eProperty
説明	ユーザーが選択できるプロパティーに、アプリケーション特有の設定を指定するときに使います。
SubClassOf	cimSetting
必須属性	

表 29. eProperty (続き)

クラス	eProperty
任意選択属性	propertyType cisPropertyType cisProperty cesPropertyType cesProperty binPropertyType binProperty
タイプ	構造型
OID (オブジェクト ID)	1.3.18.0.2.6.90
GUID (グローバル固有 ID)	b3afd69c-5c5b-11d3-b818-002035559151

表 30. DB2Node

クラス	DB2Node
Active Directory LDAP 表示名	ibm-db2Node
Active Directory 共通名 (cn)	ibm-db2Node
説明	DB2 サーバーを表します。
SubClassOf	eSap / ServiceConnectionPoint
必須属性	db2nodeName
任意選択属性	db2nodeAlias db2instanceName db2Type host / dNSHostName (注 2 参照) protocolInformation/ServiceBindingInformation
タイプ	構造型
OID (オブジェクト ID)	1.3.18.0.2.6.116
GUID (グローバル固有 ID)	b3afd65a-5c5b-11d3-b818-002035559151

表 30. DB2Node (続き)

クラス	DB2Node
特別な注意事項	1. *DB2Node* クラスは、 IBM SecureWay ディレクトリーの下にある *eSap* オブジェクト・クラスと、 Microsoft Active Directory の下にある *ServiceConnectionPoint* オブジェクト・クラスから派生します。 2. *host* は IBM SecureWay 環境で使用されます。 *dNSHostName* 属性は Microsoft Active Directory で使用されます。 3. *protocolInformation* は、 IBM SecureWay 環境でのみ使用されます。 Microsoft Active Directory の場合は、 ServiceConnectionPoint クラスを継承する *ServiceBindingInformation* がプロトコル情報を収容するために使用されます。

DB2Node オブジェクトの *protocolInformation* 属性 (IBM SecureWay Directory で使用)、または *ServiceBindingInformation* 属性 (Microsoft Active Directory で使用) には、DB2 データベース・サーバーにバインドするための通信プロトコル情報が収容されます。これは、サポートされているネットワーク・プロトコルを記述するトークンから成ります。各トークンはセミコロンで区切られます。トークンとトークンの間にスペースはありません。アスタリスク (*) は任意指定パラメーターを指定するために使用できます。

TCP/IP のトークンは以下のとおりです。

- 『TCPIP』
- サーバーのホスト名または IP アドレス
- サービス名 (svcename) またはポート番号 (例: 50000)
- (任意指定) 機密保護 (『NONE』または『SOCKS』)

APPN のトークンは以下のとおりです。

- 『APPN』
- ネットワーク ID
- パートナー LU
- トランザクション・プログラム (TP) 名 (Support Application TP のみ、 Service TP - TP in HEX はサポートしない)
- モード
- 機密保護 (『NONE』、『SAME』、または『PROGRAM』のいずれか)
- (任意指定) LAN アダプターのアドレス

- (任意指定) パスワード変更 LU

注: DB2 (Windows NT、または Windows 98 版) クライアントでは、 APPN 情報がローカルの SNA スタックで構成されておらず、かつ LAN アダプター・アドレスと任意指定の変更パスワード LU が LDAP にある場合、その DB2 クライアントはこの情報を使用して SNA スタックを構成しようとします (スタックの構成方法が分かっている場合)。このサポートは、DB2 (AIX 版 または Solaris 版) クライアントには適用されません。

IPX/SPX のトークンは以下のとおりです。

- 『IPXSPX』
- IPX アドレス

IPX/SPX listener は、 AIX および Solaris 環境の DB2 サーバーで有効です (クライアントでは無効)。 NetBIOS と NPIPE は AIX および Solaris ではサポートされていません。

NetBIOS のトークンは以下のとおりです。

- 『NETBIOS』
- サーバー NetBIOS のワークステーション名

名前付きパイプのトークンは以下のとおりです。

- 『NPIPE』
- サーバーのコンピューター名
- サーバーのインスタンス名

表 31. DB2Database

クラス	DB2Database
Active Directory LDAP 表示名	ibm-db2Database
Active Directory 共通名 (cn)	ibm-db2Database
説明	DB2 データベースを表します。
SubClassOf	最上位
必須属性	db2databaseName db2nodePtr

表 31. DB2Database (続き)

クラス	DB2Database
任意選択属性	db2databaseAlias db2additionalParameter db2ARLibrary db2authenticationLocation db2gwPtr db2databaseRelease DCEPrincipalName
タイプ	構造型
OID (オブジェクト ID)	1.3.18.0.2.6.117
GUID (グローバル固有 ID)	b3afd659-5c5b-11d3-b818-002035559151

表 32. db2additionalParameters

属性	db2additionalParameters
Active Directory LDAP 表示名	ibm-db2AdditionalParameters
Active Directory 共通名 (cn)	ibm-db2AdditionalParameters
説明	ホスト・データベース・サーバーへの接続時に使用する、任意の追加パラメーターを含んでいます。
構文	大文字小文字を無視したストリング
最大の長さ	1024
複合値	単一値
OID (オブジェクト ID)	1.3.18.0.2.4.426
GUID (グローバル固有 ID)	b3afd315-5c5b-11d3-b818-002035559151

表 33. db2authenticationLocation

属性	db2authenticationLocation
Active Directory LDAP 表示名	ibm-db2AuthenticationLocation
Active Directory 共通名 (cn)	ibm-db2AuthenticationLocation
説明	認証が行われる場所を指定します。
構文	大文字小文字を無視したストリング
最大の長さ	64

表 33. db2authenticationLocation (続き)

属性	db2authenticationLocation
複合値	単一値
OID (オブジェクト ID)	1.3.18.0.2.4.425
GUID (グローバル固有 ID)	b3afd317-5c5b-11d3-b818-002035559151
注	有効な値は、CLIENT、SERVER、DCS、DCE、KERBEROS、SVRENCRYPT、または DCSENCRYPT です。

表 34. db2ARLibrary

属性	db2ARLibrary
Active Directory LDAP 表示名	ibm-db2ARLibrary
Active Directory 共通名 (cn)	ibm-db2ARLibrary
説明	アプリケーション・リクエスター・ライブラリーの名前
構文	大文字小文字を無視したストリング
最大の長さ	256
複合値	単一値
OID (オブジェクト ID)	1.3.18.0.2.4.427
GUID (グローバル固有 ID)	b3afd316-5c5b-11d3-b818-002035559151

表 35. db2databaseAlias

属性	db2databaseAlias
Active Directory LDAP 表示名	ibm-db2DatabaseAlias
Active Directory 共通名 (cn)	ibm-db2DatabaseAlias
説明	データベース別名
構文	大文字小文字を無視したストリング
最大の長さ	1024
複合値	複合値
OID (オブジェクト ID)	1.3.18.0.2.4.422
GUID (グローバル固有 ID)	b3afd318-5c5b-11d3-b818-002035559151

表 36. db2databaseName

属性	db2databaseName
Active Directory LDAP 表示名	ibm-db2DatabaseName

表 36. db2databaseName (続き)

属性	db2databaseName
Active Directory 共通名 (cn)	ibm-db2DatabaseName
説明	データベース名
構文	大文字小文字を無視したストリング
最大の長さ	1024
複合値	単一値
OID (オブジェクト ID)	1.3.18.0.2.4.421
GUID (グローバル固有 ID)	b3afd319-5c5b-11d3-b818-002035559151

表 37. db2databaseRelease

属性	db2databaseRelease
Active Directory LDAP 表示名	ibm-db2DatabaseRelease
Active Directory 共通名 (cn)	ibm-db2DatabaseRelease
説明	データベース・リリース番号
構文	大文字小文字を無視したストリング
最大の長さ	64
複合値	単一値
OID (オブジェクト ID)	1.3.18.0.2.4.429
GUID (グローバル固有 ID)	b3afd31a-5c5b-11d3-b818-002035559151

表 38. db2nodeAlias

属性	db2nodeAlias
Active Directory LDAP 表示名	ibm-db2NodeAlias
Active Directory 共通名 (cn)	ibm-db2NodeAlias
説明	ノード別名
構文	大文字小文字を無視したストリング
最大の長さ	1024
複合値	複合値
OID (オブジェクト ID)	1.3.18.0.2.4.420
GUID (グローバル固有 ID)	b3afd31d-5c5b-11d3-b818-002035559151

表 39. db2nodeName

属性	db2nodeName
Active Directory LDAP 表示名	ibm-db2NodeName
Active Directory 共通名 (cn)	ibm-db2NodeName
説明	ノード名
構文	大文字小文字を無視したストリング
最大の長さ	64
複合値	単一値
OID (オブジェクト ID)	1.3.18.0.2.4.419
GUID (グローバル固有 ID)	b3afd31e-5c5b-11d3-b818-002035559151

表 40. db2nodePtr

属性	db2nodePtr
Active Directory LDAP 表示名	ibm-db2NodePtr
Active Directory 共通名 (cn)	ibm-db2NodePtr
説明	データベースを所有しているデータベース・サーバーを表す、ノード (DB2Node) オブジェクトへのポインター。
構文	識別名
最大の長さ	1000
複合値	単一値
OID (オブジェクト ID)	1.3.18.0.2.4.423
GUID (グローバル固有 ID)	b3afd31f-5c5b-11d3-b818-002035559151
特別な注意事項	この関連により、クライアントは、データベースへ接続するためのプロトコル通信情報を取り出すことができます。

表 41. db2gwPtr

属性	db2gwPtr
Active Directory LDAP 表示名	ibm-db2GwPtr
Active Directory 共通名 (cn)	ibm-db2GwPtr
説明	データベースへのアクセス元となるゲートウェイ・サーバーを表す、ノード・オブジェクトへのポインター。
構文	識別名

表 41. db2gwPtr (続き)

属性	db2gwPtr
最大の長さ	1000
複合値	単一値
OID (オブジェクト ID)	1.3.18.0.2.4.424
GUID (グローバル固有 ID)	b3afd31b-5c5b-11d3-b818-002035559151

表 42. db2instanceName

属性	db2instanceName
Active Directory LDAP 表示名	ibm-db2InstanceName
Active Directory 共通名 (cn)	ibm-db2InstanceName
説明	データベース・サーバー・インスタンスの名前
構文	大文字小文字を無視したストリング
最大の長さ	256
複合値	単一値
OID (オブジェクト ID)	1.3.18.0.2.4.428
GUID (グローバル固有 ID)	b3afd31c-5c5b-11d3-b818-002035559151

表 43. db2Type

属性	db2Type
Active Directory LDAP 表示名	ibm-db2Type
Active Directory 共通名 (cn)	ibm-db2Type
説明	データベース・サーバーのタイプ
構文	大文字小文字を無視したストリング
最大の長さ	64
複合値	単一値
OID (オブジェクト ID)	1.3.18.0.2.4.418
GUID (グローバル固有 ID)	b3afd320-5c5b-11d3-b818-002035559151
注	有効なデータベース・サーバーのタイプは、SERVER、MPP、および DCS です。

表 44. DCEPrincipalName

属性	DCEPrincipalName
Active Directory LDAP 表示名	ibm-DCEPrincipalName
Active Directory 共通名 (cn)	ibm-DCEPrincipalName

表 44. DCEPrincipalName (続き)

属性	DCEPrincipalName
説明	DCE プリンシパル名
構文	大文字小文字を無視したストリング
最大の長さ	2048
複合値	単一値
OID (オブジェクト ID)	1.3.18.0.2.4.443
GUID (グローバル固有 ID)	b3afd32d-5c5b-11d3-b818-002035559151

表 45. cesProperty

属性	cesProperty
Active Directory LDAP 表示名	ibm-cesProperty
Active Directory 共通名 (cn)	ibm-cesProperty
説明	この属性の値を使用すれば、アプリケーション固有の設定に関する構成パラメーターを提供できます。たとえば、値に XML 形式のデータを含めることができます。この属性の値は、 cesPropertyType 属性値ではすべて同じ種類でなければなりません。
構文	大文字小文字を区別するストリング
最大の長さ	32700
複合値	複合値
OID (オブジェクト ID)	1.3.18.0.2.4.307
GUID (グローバル固有 ID)	b3afd2d5-5c5b-11d3-b818-002035559151

表 46. cesPropertyType

属性	cesPropertyType
Active Directory LDAP 表示名	ibm-cesPropertyType
Active Directory 共通名 (cn)	ibm-cesPropertyType
説明	この属性の値を使用すれば、 cesProperty 属性のすべての値の構文、意味、その他の特性を記述できます。たとえば、『XML』という値を使用すれば、cesProperty 属性のすべての値が XML 構文としてエンコードされていることを示せるかもしれません。
構文	大文字小文字を無視したストリング
最大の長さ	128

表 46. *cesPropertyType (続き)*

属性	cesPropertyType
複合値	複合値
OID (オブジェクト ID)	1.3.18.0.2.4.308
GUID (グローバル固有 ID)	b3afd2d6-5c5b-11d3-b818-002035559151

表 47. *cisProperty*

属性	cisProperty
Active Directory LDAP 表示名	ibm-cisProperty
Active Directory 共通名 (cn)	ibm-cisProperty
説明	この属性の値を使用すれば、アプリケーション固有の設定に関する構成パラメーターを提供できます。たとえば、値に INI ファイルを含めることができます。この属性の値は、 cisPropertyType 属性値ではすべて同じ種類でなければなりません。
構文	大文字小文字を無視したストリング
最大の長さ	32700
複合値	複合値
OID (オブジェクト ID)	1.3.18.0.2.4.309
GUID (グローバル固有 ID)	b3afd2e0-5c5b-11d3-b818-002035559151

表 48. *cisPropertyType*

属性	cisPropertyType
Active Directory LDAP 表示名	ibm-cisPropertyType
Active Directory 共通名 (cn)	ibm-cisPropertyType
説明	この属性の値を使用すれば、 cisProperty 属性のすべての値の構文、意味、その他の特性を記述できます。たとえば、『INI File』という値を使用すれば、 cisProperty 属性のすべての値が INI ファイルであることを示せるかもしれません。
構文	大文字小文字を無視したストリング
最大の長さ	128
複合値	複合値
OID (オブジェクト ID)	1.3.18.0.2.4.310
GUID (グローバル固有 ID)	b3afd2e1-5c5b-11d3-b818-002035559151

表 49. binProperty

属性	**binProperty**
Active Directory LDAP 表示名	ibm-binProperty
Active Directory 共通名 (cn)	ibm-binProperty
説明	この属性の値を使用すれば、アプリケーション固有の設定に関する構成パラメーターを提供できます。たとえば、値にバイナリーでエンコードされた Lotus 1-2-3 のプロパティーを含めることができます。この属性の値は、 binPropertyType 属性値ではすべて同じ種類でなければなりません。
構文	バイナリー
最大の長さ	250000
複合値	複合値
OID (オブジェクト ID)	1.3.18.0.2.4.305
GUID (グローバル固有 ID)	b3afd2ba-5c5b-11d3-b818-002035559151

表 50. binPropertyType

属性	**binPropertyType**
Active Directory LDAP 表示名	ibm-binPropertyType
Active Directory 共通名 (cn)	ibm-binPropertyType
説明	この属性の値を使用すれば、 binProperty 属性のすべての値の構文、意味、その他の特性を記述できます。たとえば、『Lotus 123』という値を使用すれば、 binProperty 属性のすべての値がバイナリーでエンコードされた Lotus 1-2-3 のプロパティーであることを示せるかもしれません。
構文	大文字小文字を無視したストリング
最大の長さ	128
複合値	複合値
OID (オブジェクト ID)	1.3.18.0.2.4.306
GUID (グローバル固有 ID)	b3afd2bb-5c5b-11d3-b818-002035559151

表 51. PropertyType

属性	**PropertyType**
Active Directory LDAP 表示名	ibm-propertyType
Active Directory 共通名 (cn)	ibm-propertyType

表 51. PropertyType (続き)

属性	PropertyType
説明	この属性の値は、 eProperty オブジェクトの意味特性を記述します。
構文	大文字小文字を無視したストリング
最大の長さ	128
複合値	複合値
OID (オブジェクト ID)	1.3.18.0.2.4.320
GUID (グローバル固有 ID)	b3afd4ed-5c5b-11d3-b818-002035559151

表 52. settingID

属性	settingID
Active Directory LDAP 表示名	適用されない
Active Directory 共通名 (cn)	適用されない
説明	命名属性。 eProperty などの cimSetting から派生したオブジェクト項目を示すために使用できます。
構文	大文字小文字を無視したストリング
最大の長さ	256
複合値	単一値
OID (オブジェクト ID)	1.3.18.0.2.4.325
GUID (グローバル固有 ID)	b3afd596-5c5b-11d3-b818-002035559151

付録K. コントロール・センターの拡張

バージョン 7 では、新しいプラグイン・アーキテクチャーを使って機能を追加することにより、 DB2 ユニバーサル・データベースのコントロール・センターを拡張することができます。

このプラグイン・アーキテクチャーの目的は、「コントロール・センター」ポップアップ・メニューにある所定のオブジェクトに項目を追加したり、ツールバーに新しいボタンを追加する機能を提供することです。実装しなければならない一連の Java インターフェースは、それらのツールに付属しています。このようなインターフェースは、追加のアクションが組み込まれるコントロール・センターとの間でやり取りするときに使います。

パフォーマンスの考慮事項

プラグイン拡張機能 (db2plug.zip) は、コントロール・センター・ツールの起動時にロードされます。そのため、ZIP ファイルのサイズによっては、ツールの起動時間が長くなる可能性があります。しかし、プラグインの ZIP ファイルはほとんどの場合に小さいもので、その影響は最小限に抑えられるはずです。

パッケージについての考慮事項

拡張クラス・ファイルには、Java アーカイブ・ファイルの規則に従って、 ZIP をかける必要があります。コントロール・センター・ツールをアプリケーションとして実行するためには、 ZIP ファイル (db2plug.zip) をクラスパス 内に置いておかなければなりません。コントロール・センター・ツールをアプレットとして実行するには、ZIP ファイルを、コントロール・センターの HTML ファイルで <codebase> タグが示す位置に置く必要があります。

ZIP ファイルは圧縮せずに作成し、全クラス・ファイルの相対パス位置を維持するようにします (`zip -r0 db2plug.zip *.class`)。

インターフェースについての説明

以下のインターフェースが付属しています。

- CCExtension
- CCObject
- CCMenuAction
- CCToolbarAction

インターフェースについては次の節で説明され、その後に例が続きます。

CCExtension

CCExtension インターフェースを使うと、新しいツールバー・ボタン、新しいメニュー項目を追加し、さらに既存のメニュー・アクションをオーバーライドすることにより、コントロール・センターのユーザー・インターフェースを拡張できます。

外部インターフェースは次のように定義されます。

```
public interface CCExtension
{
   /**
    * Get an array of CCObject subclass objects which define
    * a list of objects to be inserted or overridden in the
    * Control Center
    * @return CCObject[] CCObject subclass objects array
    */
   public CCObject[] getObjects();

   /**
    * Get an array of CCToolbarAction subclass objects which represent
    * a list of buttons to be added to the Control Center
    * main toolbar.
    * @return CCToolbarAction[] CCToolbarAction subclass objects array
    */
   public CCToolbarAction[] getToolbarActions();
}
```

CCExtension を使用するには、″com.ibm.db2.tools.cc.navigator″ パッケージをインポートし、このインターフェースを実装する Java クラスを作成します。この新しいクラスには、 getObjects() および getToolbarActions() メソッドを実装する必要があります。

getObjects() メソッドは CCObject の配列を戻します。この配列では、ユーザーが新しいメニュー・アクションを追加するとき、あるいは事前定義された一連のメニュー・アクションを削除するときの、既存のオブジェクトが定義されます。

getToolbarActions() メソッドは CCToolbarAction の配列を戻します。この配列は、コントロール・センターのメイン・ツールバーに追加されます。

コントロール・センターの拡張機能を定義するときには、 1 つの CCExtension サブクラス・ファイルを使うこともできますし、複数の CCExtension サブクラス・ファイルを使用することも可能です。これらの拡張機能を使うコントロール・センターでは、以下のセットアップ手順を使います。

1. すべての CCExtension サブクラス・ファイルを含む、 ″db2plug.zip″ ファイルを作成します。このファイルは圧縮しないようにします。たとえば、CCExtension ファイルがプラグイン・パッケージに含まれていて、そのプラグインのディレクトリーにある場合、以下のコマンドを使用します。

   ```
   zip -r0 db2plug.zip plugin¥*.class
   ```

このコマンドを発行すると、プラグイン・パッケージの全クラス・ファイルが db2plug.zip ファイルに入れられ、その相対パス情報が保存されます。

2. コントロール・センターをアプレットとして実行するには、db2plug.zip ファイルを、コントロール・センターの HTML ファイルで <codebase> タグが示す位置に置きます。コントロール・センターをアプリケーションとして実行するには、db2plug.zip を、 CLASSPATH 環境変数で示されたディレクトリーに入れておきます。

複数のアーカイブをサポートしているブラウザーでは、「コントロール・センター」の HTML ページのアーカイブ・リストに、 ″db2plug.zip″ と追加するだけで済みます。それ以外のブラウザーでは、CCExtension、CCObject、 CCToolbarAction、および CCMenuAction サブクラス・ファイルはすべて、それぞれが属するパッケージに応じてその相対ディレクトリーに入れておく必要があります。

CCObject

CCObject インターフェースを使用すると、既存オブジェクトのメニュー・アクションの動作を変更できます。

外部インターフェースは次のように定義されます。

```
public interface CCObject
{
   /**
    * The following static constants defines a list of object type
    * available to be added to the Control Center tree.
    */
   public static final int UDB_SYSTEMS_FOLDER                          = 0;
   public static final int UDB_SYSTEM                                  = 1;
   public static final int UDB_INSTANCES_FOLDER                        = 2;
   public static final int UDB_INSTANCE                                = 3;
   public static final int UDB_DATABASES_FOLDER                        = 4;
   public static final int UDB_DATABASE                                = 5;
   public static final int UDB_TABLES_FOLDER                           = 6;
   public static final int UDB_TABLE                                   = 7;
   public static final int UDB_TABLESPACES_FOLDER                      = 8;
   public static final int UDB_TABLESPACE                              = 9;
   public static final int UDB_VIEWS_FOLDER                            = 10;
   public static final int UDB_VIEW                                    = 11;
   public static final int UDB_ALIASES_FOLDER                          = 12;
   public static final int UDB_ALIAS                                   = 13;
   public static final int UDB_TRIGGERS_FOLDER                         = 14;
   public static final int UDB_TRIGGER                                 = 15;
   public static final int UDB_SCHEMAS_FOLDER                          = 16;
   public static final int UDB_SCHEMA                                  = 17;
   public static final int UDB_INDEXES_FOLDER                          = 18;
   public static final int UDB_INDEX                                   = 19;
   public static final int UDB_CONNECTIONS_FOLDER                      = 20;
   public static final int UDB_CONNECTION                              = 21;
   public static final int UDB_REPLICATION_SOURCES_FOLDER              = 22;
   public static final int UDB_REPLICATION_SOURCE                      = 23;
   public static final int UDB_REPLICATION_SUBSCRIPTIONS_FOLDER        = 24;
```

```
public static final int UDB_REPLICATION_SUBSCRIPTION                = 25;
public static final int UDB_BUFFERPOOLS_FOLDER                      = 26;
public static final int UDB_BUFFERPOOL                              = 27;
public static final int UDB_APPLICATION_OBJECTS_FOLDER              = 28;
public static final int UDB_USER_DEFINED_DISTINCT_DATATYPES_FOLDER  = 29;
public static final int UDB_USER_DEFINED_DISTINCT_DATATYPE          = 30;
public static final int UDB_USER_DEFINED_DISTINCT_FUNCTIONS_FOLDER  = 31;
public static final int UDB_USER_DEFINED_DISTINCT_FUNCTION          = 32;
public static final int UDB_PACKAGES_FOLDER                         = 33;
public static final int UDB_PACKAGE                                 = 34;
public static final int UDB_STORE_PROCEDURES_FOLDER                 = 35;
public static final int UDB_STORE_PROCEDURE                         = 36;
public static final int UDB_USER_AND_GROUP_OBJECTS_FOLDER           = 37;
public static final int UDB_DB_USERS_FOLDER                         = 38;
public static final int UDB_DB_USER                                 = 39;
public static final int UDB_DB_GROUPS_FOLDER                        = 40;
public static final int UDB_DB_GROUP                                = 41;
public static final int UDB_DRDA_TABLE                              = 42;

public static final int S390_SUBSYSTEMS_FOLDER                      = 43;
public static final int S390_SUBSYSTEM                              = 44;
public static final int S390_BUFFERPOOLS_FOLDER                     = 45;
public static final int S390_BUFFERPOOL                             = 46;
public static final int S390_VIEWS_FOLDER                           = 47;
public static final int S390_VIEW                                   = 48;
public static final int S390_DATABASES_FOLDER                       = 49;
public static final int S390_DATABASE                               = 50;
public static final int S390_TABLESPACES_FOLDER                     = 51;
public static final int S390_TABLESPACE                             = 52;
public static final int S390_TABLES_FOLDER                          = 53;
public static final int S390_TABLE                                  = 54;
public static final int S390_INDEXS_FOLDER                          = 55;
public static final int S390_INDEX                                  = 56;
public static final int S390_STORAGE_GROUPS_FOLDER                  = 57;
public static final int S390_STORAGE_GROUP                          = 58;
public static final int S390_ALIASES_FOLDER                         = 59;
public static final int S390_ALIAS                                  = 60;
public static final int S390_SYNONYMS_FOLDER                        = 61;
public static final int S390_SYNONYM                                = 62;
public static final int S390_APPLICATION_OBJECTS_FOLDER             = 63;
public static final int S390_COLLECTIONS_FOLDER                     = 64;
public static final int S390_COLLECTION                             = 65;
public static final int S390_PACKAGES_FOLDER                        = 66;
public static final int S390_PACKAGE                                = 67;
public static final int S390_PLANS_FOLDER                           = 68;
public static final int S390_PLAN                                   = 69;
public static final int S390_PROCEDURES_FOLDER                      = 70;
public static final int S390_PROCEDURE                              = 71;
public static final int S390_DB_USERS_FOLDER                        = 72;
public static final int S390_DB_USER                                = 73;
public static final int S390_LOCATIONS_FOLDER                       = 74;
public static final int S390_LOCATION                               = 75;
public static final int S390_DISTINCT_TYPES_FOLDER                  = 76;
public static final int S390_DISTINCT_TYPE                          = 77;
public static final int S390_USER_DEFINED_FUNCTIONS_FOLDER          = 78;
```

```
public static final int S390_USER_DEFINED_FUNCTION                  = 79;
public static final int S390_TRIGGERS_FOLDER                        = 80;
public static final int S390_TRIGGER                                = 81;
public static final int S390_SCHEMAS_FOLDER                         = 82;
public static final int S390_SCHEMA                                 = 83;
public static final int S390_CATALOG_TABLES_FOLDER                  = 84;
public static final int S390_CATALOG_TABLE                          = 85;

public static final int DCS_GATEWAY_CONNECTIONS_FOLDER              = 86;
public static final int DCS_GATEWAY_CONNECTION                      = 87;

/**
 * Total number of object types
 */
public static final int NUM_OBJECT_TYPES                            = 88;

/**
 * Get the name of these object
 * The function returns the name of this object. This name
 * can be of three types:
 * (1) Fully qualified name
 *     Syntax: xxxxx-yyyyy-zzzzz
 *             where xxxxx-yyyyy is the fully quality name of the
 *             parent object and zzzzz is the name of the new object.
 *     Note: Parent and child object name is separated by '-' character.
 *     If a schema name is required to identify object, the fully
 *     qualified name is represented by xxxxx-yyyyy-wwwww.zzzzz
 *     where wwwww is the schema name.
 *     Only the behavior of the object that matches this fully
 *     qualified name will be affected.
 * (2) Parent fully qualified name
 *     Syntax: xxxxx-yyyyy
 *             where xxxxx-yyyyy is the fully qualified name of the
 *             parent object.
 *     When the object type is folder (ie. DATABASES_FOLDER), the
 *     getName() should only return the fully qualified name of the
 *     folder's parent.
 *     Only the behavior of the object that match this name
 *     and the specific type return by the getType() function will be
 *     affected.
 * (3) null
 *     Syntax: null
 *     If null is return, the CCActions returns by the getActions()
 *     call will be applied to all objects of type returns by the
 *     getType() call.
 * @return String object name
 */
public String getName();

/**
 * Get the type of this object
 * @return int return one of the static type constants defined
 * in this interface
 */
public int getType();
```

```
    /**
     * Get the CCMenu Action array which defines the list of menu actions
     * to be created for the selected object
     * return CCMenuAction[] CCMenuAction array
     */
    public CCMenuAction[] getMenuActions();

    /**
     * Check if this object is editable.
     * If not, the Alter related menu items will be removed from
     * the object's popup menu return boolean If false, the Alter
     * menu item will be removed from the object's popup menu
     */
    public boolean isEditable();

    /**
     * Check if this object is configurable.
     * If not, the configuration related menu items will be
     * removed from the object's popup menu return boolean If
     * false, the Configuration related menu item will be removed
     * from the object's popup menu
     */
    public boolean isConfigurable();
}
```

注: 現時点で、 CCObject における最後の 2 つのメソッド isEditable() と isConfigurable() は、常に **true** を戻すはずです。

CCMenuAction

CCMenuAction インターフェースを使うと、コントロール・センター・オブジェクトで使用する新しいアクションを定義できます。

外部インターフェースは次のように定義されます。

```
public interface CCMenuAction
{
    /**
     * Get the name of this action
     * @return String Name text on the menu item
     */
    public String getMenuText();

    /**
     * Invoked when an action occurs. Use the getActionCommand()
     * method of the ActionEvent to get the fully qualified name of
     * the invoked Control Center object.
     * @param e Action event
     */
    public void actionPerformed(ActionEvent e);
}
```

CCToolBarAction

CCToolbarAction インターフェースを使用すると、コントロール・センターのツールバーに新しいアクションを定義できます。

外部インターフェースは次のように定義されます。

```
public interface CCToolbarAction
{
   /**
    * Get the name of this action
    * @return String Name text on the menu item, or toolbar
    * button hover help
    */
   public String getHoverHelpText();

   /**
    * Get the icon for the toolbar button
    * Any toolbar CCAction should implement this function and return
    * a valid ImageIcon object. Otherwise, the button will have no icon.
    * @return ImageIcon Icon to be displayed
    */
   public ImageIcon getIcon();

   /**
    * Invoked when an action occurs.
    * @param e Action event
    */
   public void actionPerformed(ActionEvent e);
}
```

使用のシナリオ

以下の例で示されるコードには、次のような働きがあります。

1. SAMPLE データベースのアクションを更新する (462ページの『MySample.java』を参照)
2. 全データベース・オブジェクトのアクションを更新する (463ページの『MyDatabaseActions.java』を参照)
3. 新しいインスタンス・オブジェクトを追加する (464ページの『MyInstance.java』を参照)
4. DB2 インスタンスのアクションを更新する (464ページの『MyDB2.java』を参照)
5. Databases フォルダーのアクションを更新する (465ページの『MyDatabases.java』を参照)
6. SYSIBM.SYSPLAN 表のアクションを更新する (466ページの『MySYSPLAN.java』を参照)
7. 新しい表オブジェクトを追加する (466ページの『MyTable.java』を参照)
8. Application オブジェクトの下の DB_User オブジェクトのアクションを更新する (467ページの『MyDBUser.java』を参照)

9. コントロール・センターのツールバーにボタンを追加する (468ページの『MyToolbarAction.java』を参照)

主要な拡張機能ファイルは MyExtension.java です。すべてのクラス・ファイルがプラグインのディレクトリーに格納され、次のコマンドで ZIP をかけられます。

```
zip -r0 db2plug.zip plugin
```

生成された db2plug.zip は、コントロール・センターをアプリケーションとして実行しているか、あるいはアプレットとして実行しているかに応じて、 CLASSPATH 内か codebase ディレクトリー内に入れられます。

MyExtension.java

```
package plugin;
import com.ibm.db2.tools.cc.navigator.*;

public class MyExtension implements CCExtension
{
   public CCObject[] getObjects()
   {
      CCObject[] objs = new CCObject[10];
      objs[0] = new MySample();
      objs[1] = new MyDatabaseActions();
      objs[2] = new MyInstance();
      objs[3] = new MyDB2();
      objs[4] = new MyDatabases();
      objs[5] = new MySYSPLAN();
      objs[6] = new MyTable();
      objs[7] = new MyDBUser();
      return objs;
   }

   public CCAction[] getActions()
   {
      CCAction[] actions = new CCAction[1];
      actions[0] = new MyToolbarAction();
      return actions;
   }
}
```

MySample.java

```
package plugin;
import com.ibm.db2.tools.cc.navigator.*;

public class MySample implements CCObject
{
   public String getName()
   {
      return "LOCAL - DB2 - SAMPLE";
   }
```

```
   public int getType()
   {
      return DATABASE;
   }

   public javax.swing.ImageIcon getIcon()
   {
      return null;
   }

   public boolean isNew()
   {
      return false;
   }

   public CCAction[] getActions()
   {
      CCAction[] acts = new CCAction[2];
      acts[0] = new MyAlterAction();
      acts[1] = new MyAction();
      return acts;
   }

}
```

MyDatabaseActions.java

```
package plugin;
import com.ibm.db2.tools.cc.navigator.*;

public class MyDatabaseActions implements CCObject
{
   public String getName()
   {
      return null;
   }

   public int getType()
   {
      return DATABASE;
   }

   public javax.swing.ImageIcon getIcon()
   {
      return null;
   }

   public boolean isNew()
   {
      return false;
   }

   public CCAction[] getActions()
   {
      CCAction[] acts = new CCAction[2];
```

```
        acts[0] = new MyDropAction();
        acts[1] = new MyAction();
        return acts;
    }

}
```

MyInstance.java

```
package plugin;
import com.ibm.db2.tools.cc.navigator.*;

public class MyInstance implements CCObject
{
    public String getName()
    {
        return "LOCAL - MyInstance";
    }

    public int getType()
    {
        return INSTANCE;
    }

    public javax.swing.ImageIcon getIcon()
    {
        return null;
    }

    public boolean isNew()
    {
        return true;
    }

    public CCAction[] getActions()
    {
        CCAction[] acts = new CCAction[2];
        acts[0] = new MyAlterAction();
        acts[1] = new MyAction();
        return null;
    }

}
```

MyDB2.java

```
package plugin;
import com.ibm.db2.tools.cc.navigator.*;

public class MyDB2 implements CCObject
{
    public String getName()
    {
        return "LOCAL - DB2";
```

```
    }

    public int getType()
    {
       return INSTANCE;
    }

    public javax.swing.ImageIcon getIcon()
    {
       return null;
    }

    public boolean isNew()
    {
       return false;
    }

    public CCAction[] getActions()
    {
       CCAction[] acts = new CCAction[3];
       acts[0] = new MyAlterAction();
       acts[1] = new MyAction();
       acts[2] = new MyCascadeAction();
       return acts;
    }
}
```

MyDatabases.java

```
package plugin;
import com.ibm.db2.tools.cc.navigator.*;

public class MyDatabases implements CCObject
{
   public String getName()
   {
      return "LOCAL - DB2 - Databases";
   }

   public int getType()
   {
      return DATABASE;
   }

   public javax.swing.ImageIcon getIcon()
   {
      return null;
   }

   public boolean isNew()
   {
      return false;
   }

   public CCAction[] getActions()
```

```
    {
       CCAction[] acts = new CCAction[1];
       acts[0] = new MyCreateAction();
       return acts;
    }

}
```

MySYSPLAN.java

```
package plugin;
import com.ibm.db2.tools.cc.navigator.*;

public class MySYSPLAN implements CCObject
{
   public String getName()
   {
      return "LOCAL - DB2 - SAMPLE - SYSIBM - SYSPLAN";
   }

   public int getType()
   {
      return TABLE;
   }

   public javax.swing.ImageIcon getIcon()
   {
      return null;
   }

   public boolean isNew()
   {
      return false;
   }

   public CCAction[] getActions()
   {
      CCAction[] acts = new CCAction[2];
      acts[0] = new MyAlterAction();
      acts[1] = new MyAction();
      return acts;
   }

}
```

MyTable.java

```
package plugin;
import com.ibm.db2.tools.cc.navigator.*;

public class MyTable implements CCObject
{
   public String getName()
   {
```

```
        return "LOCAL - DB2 - SAMPLE - SYSIBM - MyTable";
    }

    public int getType()
    {
        return TABLE;
    }

    public javax.swing.ImageIcon getIcon()
    {
        return null;
    }

    public boolean isNew()
    {
        return true;
    }

    public CCAction[] getActions()
    {
        CCAction[] acts = new CCAction[2];
        acts[0] = new MyAlterAction();
        acts[1] = new MyAction();
        return acts;
    }

}
```

MyDBUser.java

```
package plugin;
import com.ibm.db2.tools.cc.navigator.*;

public class MyDBUser implements CCObject
{
    public String getName()
    {
        return "LOCAL - DB2 - TEST-DB Users";
    }

    public int getType()
    {
        return DB_USER;
    }

    public javax.swing.ImageIcon getIcon()
    {
        return null;
    }

    public boolean isNew()
    {
        return false;
    }
```

```
    public CCAction[] getActions()
    {
       CCAction[] acts = new CCAction[2];
       acts[0] = new MyAlterAction();
       acts[1] = new MyAction();
       return acts;
    }

}
```

MyToolbarAction.java

```
package plugin;
import com.ibm.db2.tools.cc.navigator.*;
import javax.swing.*;

public class MyToolbarAction extends CCAction
{
   public MyToolbarAction()
   {
      super("MyToolbarAction");
   }

   public ImageIcon getIcon()
   {
      return <Your icon>;
   }

   public boolean actionPerformed(String objectName)
   {
      System.out.println( "My action performed, object name = " +
                          objectName );
      return true;
   }
}
```

MyAlterAction.java

```
package plugin;
import com.ibm.db2.tools.cc.navigator.*;

public class MyAlterAction extends CCAction
{
   public MyAlterAction()
   {
      super(0);
   }

   public boolean actionPerformed(String objectName)
   {
      System.out.println( "My alter action performed, object name = " +
```

```
                              objectName );
      return true;
   }
}
```

MyAction.java

```
package plugin;
import com.ibm.db2.tools.cc.navigator.*;

public class MyAction extends CCAction
{
   public MyAction()
   {
      super("MyAction");
   }

   public boolean actionPerformed(String objectName)
   {
      System.out.println( "My action performed, object name = " +
                              objectName );
      return true;
   }
}
```

MyDropAction.java

```
package plugin;
import com.ibm.db2.tools.cc.navigator.*;

public class MyDropAction extends CCAction
{
   public MyDropAction()
   {
      super(1);
   }

   public boolean actionPerformed(String objectName)
   {
      System.out.println( "My drop action performed, object name = " +
                              objectName );
      return true;
   }
}
```

MyCascadeAction.java

```
package plugin;
import com.ibm.db2.tools.cc.navigator.*;

public class MyCascadeAction extends CCAction
{
   public MyCascadeAction()
   {
```

```
        super(11,2);
    }

    public boolean actionPerformed(String objectName)
    {
        System.out.println( "My cascade action performed, object name = " +
                            objectName );
        return true;
    }
}
```

MyCreateAction.java

```
package plugin;
import com.ibm.db2.tools.cc.navigator.*;

public class MyCreateAction extends CCAction
{
    public MyCreateAction()
    {
        super(0);
    }

    public boolean actionPerformed(String objectName)
    {
        System.out.println( "My create action performed, object name = " +
                            objectName );
        return true;
    }
}
```

付録L. DB2 ライブラリーの使用法

DB2 ユニバーサル・データベース ライブラリーは、オンライン・ヘルプ、ブック (PDF および HTML)、および HTML 形式のサンプル・プログラムから成っています。このセクションでは、ユーザーに提供される情報について紹介し、その入手方法を示します。

オンライン製品情報をご利用になるには、インフォメーション・センターを使用することができます。詳細については、487ページの『インフォメーション・センターを使用した情報へのアクセス』を参照してください。ここではタスク情報、DB2 ブック、トラブルシューティング情報、サンプル・プログラム、および Web の DB2 情報を見ることができます。

DB2 PDF ファイルおよびハードコピー版資料

DB2 情報

以下に示す表では、DB2 ブックを 4 つのカテゴリーに分類しています。

DB2 の手引きおよび解説書
これらの資料は、すべてのプラットフォームに共通の DB2 情報を含んでいます。

DB2 のインストールおよび構成の情報
これらの資料は、特定のプラットフォーム上の DB2 ごとに用意されています。たとえば、 OS/2、Windows、および UNIX ベースのプラットフォームで稼働するそれぞれの DB2 用に、別個の*概説*および*インストール* 資料が用意されています。

プラットフォーム共通のサンプル・プログラム (HTML 形式)
これらのサンプルは、アプリケーション開発クライアントとともにインストールされるサンプル・プログラムの HTML 版です。これらのサンプルは参考用であり、実際のプログラムに代わるものではありません。

リリース情報
これらのファイルには、DB2 ブックには含められなかった最新の情報が記載されています。

インストール情報、リリース情報、およびチュートリアルは、製品 CD-ROM から HTML 形式で参照することができます。ほとんどの資料は、製品 CD-ROM から HTML 形式で表示できますし、 DB2 の資料 CD-ROM から Adobe Acrobat (PDF) 形

式で表示し印刷することができます。 IBM にハードコピー版の資料を注文したい場合は、 483ページの『印刷資料の注文方法』を参照してください。注文可能な資料については、以下の表をご覧ください。

OS/2 および Windows プラットフォームの場合、 HTML ファイルは `sqllib¥doc¥html` ディレクトリーにインストールできます。 DB2 情報はいくつかの言語で提供されています。しかし、すべての言語に翻訳されているわけではありません。ある言語で情報が提供されていない場合は、英語版の情報が提供されます。

UNIX プラットフォームの場合、言語ごとに異なる複数の HTML ファイルを `doc/%L/html` ディレクトリーにインストールできます。ここで、*%L* は地域を表しています。詳細については、適切な*概説およびインストールの手引き* を参照してください。

DB2 ブックを入手して情報を利用するには、次のようなさまざまな方法があります。

- 486ページの『オンライン情報の表示』
- 490ページの『オンライン情報の検索』
- 483ページの『印刷資料の注文方法』
- 483ページの『PDF 資料の印刷』

表 53. DB2 情報

名前	記述	資料番号 PDF ファイル名	HTML ディレクトリー
	DB2 の手引きおよび解説書情報		
管理の手引き	*管理の手引き: 計画* は、データベース概念について概説し、設計 (たとえば、論理および物理データベース設計) に関する情報を提供し、高い可用性について解説しています。	SC88-8513 db2d1x70	db2d0
	管理の手引き: インプリメンテーション は、設計、データベースへのアクセス、監査、バックアップ、および回復などのインプリメンテーションについて説明しています。	SC88-8511 db2d2x70	
	管理の手引き: パフォーマンス は、データベース環境について解説し、さらにアプリケーションのパフォーマンスの評価と調整の方法について説明しています。	SC88-8512 db2d3x70	
管理 API 解説書	データベースの管理に使用できる DB2 アプリケーション・プログラミング・インターフェース (API) およびデータ構造について説明します。また、この資料は、アプリケーションから API を呼び出す方法も示します。	SC88-8514 db2b0x70	db2b0
アプリケーション構築の手引き	環境設定に関する情報を提供し、Windows、OS/2、および UNIX ベースのプラットフォームでの DB2 アプリケーションのコンパイル、リンク、実行の各ステップについて説明します。	SC88-8515 db2axx70	db2ax
APPC, CPI-C, and SNA Sense Codes	DB2 ユニバーサル・データベース製品をご使用中に発生する可能性のあるセンス・コード APPC、CPI-C、および SNA についての一般情報を提供します。 HTML 形式でのみご利用いただけます。	資料番号なし db2apx70	db2ap

表 53. DB2 情報 (続き)

名前	記述	資料番号 PDF ファイル名	HTML ディレクトリー
アプリケーション開発の手引き	DB2 データベースにアクセスするアプリケーションを、組み込み SQL または Java (JDBC および SQLJ) を使用して開発する方法について説明します。さらに、ストアード・プロシージャーの作成方法、ユーザー定義関数の作成方法、ユーザー定義タイプの作成方法、トリガーの使用方法、区画化されている環境または統合されているシステムでのアプリケーションの開発方法などについて解説されています。	SC88-8516 db2a0x70	db2a0
コール・レベル・インターフェースの手引きおよび解説書	DB2 データベースにアクセスするアプリケーションを、 DB2 コール・レベル・インターフェース (Microsoft ODBC 仕様互換の呼び出し可能 SQL) を使用して開発する方法について説明します。	SC88-8517 db2l0x70	db2l0
コマンド解説書	コマンド行プロセッサーの使用法について説明し、データベースの管理に使用できる DB2 コマンドについて解説しています。	SC88-8518 db2n0x70	db2n0
コネクティビティー 補足	DB2 (AS/400 版)、DB2 (OS/390 版)、DB2 (MVS 版)、または DB2 (VM 版) を DRDA アプリケーション・リクエスターとして DB2 ユニバーサル・データベースとともに使用するためのセットアップ情報および参照情報を提供します。また、この資料は DRDA アプリケーション・サーバーを DB2 コネクト アプリケーション・リクエスターとともに使用する方法の詳細を示します。 HTML と PDF でのみ利用可能	資料番号なし db2h1x70	db2h1
データ移動ユーティリティー手引きおよび解説書	データの移動を行う DB2 ユーティリティー (インポート、エクスポート、ロード、AutoLoader、および DPROP など) の使用法について説明しています。	SC88-8522 db2dmx70	db2dm

表 53. DB2 情報 (続き)

名前	記述	資料番号 PDF ファイル名	HTML ディレクトリー
データウェアハウスセンター 管理の手引き	データウェアハウスセンターを使用してデータウェアハウスを構築および保守する方法を説明します。	SC88-8545 db2ddx70	db2dd
データウェアハウスセンター アプリケーション統合の手引き	プログラマーがアプリケーションをデータウェアハウスセンターおよび情報カタログ・マネージャーと統合するのに役立つ情報を提供します。	SC88-8546 db2adx70	db2ad
DB2 コネクト 使用者の手引き	DB2 コネクト製品の概念、プログラミング、および一般的な使用方法に関する情報を提供します。	SC88-8521 db2c0x70	db2c0
DB2 クエリー・パトローラー 管理の手引き	DB2 クエリー・パトローラー・システムの運用の概説を行い、運用および管理に関する詳細情報、および管理用グラフィカル・ユーザー・インターフェース・ユーティリティーについてのタスク情報を提供します。	SC88-8525 db2dwx70	db2dw
DB2 クエリー・パトローラー 使用者の手引き	DB2 クエリー・パトローラーのツールや関数の使用方法を説明します。	SC88-8527 db2wwx70	db2ww
用語集	DB2 およびそのコンポーネントで使用される用語の定義を示します。 HTML 形式と SQL 解説書 で利用可能	資料番号なし db2t0x70	db2t0
イメージ、オーディオ、およびビデオ・エクステンダー 管理およびプログラミングの手引き	DB2 エクステンダーの一般情報について提供し、画像、音声、およびビデオ (IAV) エクステンダーの管理と構成について、および IAV エクステンダーを使用したプログラミングについて説明しています。さらに、参照情報、診断情報 (メッセージ解説)、およびサンプルも収録されています。	SC88-8609 dmbu7x70	dmbu7
情報カタログ・マネージャー 管理の手引き	情報カタログを管理するためのガイドです。	SC88-8547 db2dix70	db2di
情報カタログ・マネージャー プログラミングの手引きおよび解説書	情報カタログ・マネージャー用の体系化されたインターフェースの定義を示します。	SC88-8549 db2bix70	db2bi

表 53. DB2 情報 (続き)

名前	記述	資料番号 PDF ファイル名	HTML ディレクトリー
情報カタログ・マネージャー 使用者の手引き	情報カタログ・マネージャー・ユーザー・インターフェースの使用に関する情報を提供します。	SC88-8548 db2aix70	db2ai
インストールおよび構成補足	プラットフォーム固有の DB2 クライアントの計画、インストール、およびセットアップのガイドです。この補足資料には、バインド、クライアント / サーバー通信の設定、 DB2 GUI ツール、DRDA AS、分散インストール、分散要求の構成、および異種データ・ソースへのアクセスについても説明されています。	GC88-8524 db2iyx70	db2iy
メッセージ解説書	DB2、情報カタログ・マネージャー、およびデータウェアハウスセンターから出されるメッセージとコードをリストし、取るべき処置を解説しています。	第 1 巻 GC88-8543 db2m1x70 第 2 巻 GC88-8544 db2m2x70	db2m0
OLAP Integration Server Administration Guide	OLAP Integration Server の Administration Manager コンポーネントの使用方法を説明します。	SC27-0787 db2dpx70	n/a
OLAP Integration Server Metaoutline User's Guide	標準の OLAP Metaoutline インターフェースを使用して (Metaoutline Assistant を使用するのではなく) OLAP metaoutline を作成しデータを取り込む方法を説明しています。	SC27-0784 db2upx70	n/a
OLAP Integration Server Model User's Guide	(Model Assistant ではなく) 標準的な OLAP Model Interface を使用して OLAP モデルを作成する方法を説明します。	SC27-0783 db2lpx70	n/a
OLAP のセットアップおよびユーザーズ・ガイド	OLAP スターター・キットの構成およびセットアップに関する情報を提供します。	SC88-8652 db2ipx70	db2ip
Hyperion Essbase スプレッドシート アドイン ユーザーズ ガイド for Excel	Excel 作表計算プログラムを使用して OLAP データを分析する方法を説明します。	SC88-8724 db2epx70	db2ep

表 53. DB2 情報 (続き)

名前	記述	資料番号 PDF ファイル名	HTML ディレクトリー
Hyperion Essbase スプレッドシート アドイン ユーザーズ ガイド for 1-2-3	ロータス 1-2-3 作表計算プログラムを使用して OLAP データを分析する方法を説明します。	SC88-8723 db2tpx70	db2tp
レプリケーションの手引きおよび解説書	DB2 に付属の IBM レプリケーション・ツールの計画、構成、管理、および使用方法に関する情報を提供します。	SC88-8550 db2e0x70	db2e0
地理情報エクステンダー使用者の手引きおよび解説書	地理情報エクステンダーのインストール、構成、管理、プログラミング、およびトラブルシューティングに関する情報を提供します。また、地理情報データの概念についての重要事項を示し、地理情報エクステンダー固有の参照情報 (メッセージおよび SQL) を提供します。	SC88-8624 db2sbx70	db2sb
SQL 概説	SQL の概念を紹介し、構造体とタスクの例を多数提供しています。	SC88-8539 db2y0x70	db2y0
SQL 解説書	SQL の構文、セマンティクス、および言語規則について説明します。また、この資料には、各リリース間の互換性、製品の制限事項、およびカタログ・ビューも含まれます。	第 1 巻 SC88-8540 db2s1x70 第 2 巻 SC88-8657 db2s2x70	db2s0
システム・モニター 手引きおよび解説書	データベースおよびデータベース・マネージャーに関連したさまざまな情報を収集する方法を示します。この資料は、この情報を利用して、データベース活動の把握、パフォーマンス向上、および問題原因の判別を行う方法を説明しています。	SC88-8523 db2f0x70	db2f0

表 53. DB2 情報 (続き)

名前	記述	資料番号 PDF ファイル名	HTML ディレクトリー
テキスト・エクステンダー *管理およびプログラミング*	DB2 エクステンダーの一般情報、テキスト・エクステンダーの管理および構成情報、およびテキスト・エクステンダーを使用したプログラミングの方法について解説します。この資料には、参照情報、診断情報 (メッセージ解説)、およびサンプルが含まれています。	SC88-8610 desu9x70	desu9
問題判別の手引き	エラーの原因の判別、問題からの回復、および DB2 カスタマー・サービスの支援の下での診断ツールの使用法を記載しています。	GD88-7271 db2p0x70	db2p0
新機能	DB2 ユニバーサル・データベース バージョン 7 の新しい機能および拡張機能について説明します。	SC88-8541 db2q0x70	db2q0
DB2 のインストールおよび構成の情報			
DB2 コネクト エンタープライズ・エディション (OS/2 および Windows 版) 概説およびインストール	OS/2 および Windows 32 ビット・オペレーティング・システム版の DB2 コネクト エンタープライズ・エディションで、計画、移行、インストール、および構成を行う場合の情報を提供します。また、この資料はサポートされている多数のクライアントのインストールおよびセットアップについても説明します。	GC88-8520 db2c6x70	db2c6
DB2 コネクト エンタープライズ・エディション (UNIX 版) 概説およびインストール	UNIX ベースのプラットフォームでの DB2 コネクト エンタープライズ・エディションの計画、移行、インストール、構成、およびタスクに関する情報を提供します。また、この資料はサポートされている多数のクライアントのインストールおよびセットアップについても説明します。	GC88-8519 db2cyx70	db2cy

表 53. DB2 情報 (続き)

名前	記述	資料番号 PDF ファイル名	HTML ディレクトリー
DB2 コネクト パーソナル・エディション 概説およびインストール	OS/2 および Windows 32 ビット・オペレーティング・システムの DB2 コネクト パーソナル・エディションで、計画、移行、インストール、および構成を行う場合のタスク情報を提供します。また、この資料はサポートされているすべてのクライアントのインストールおよびセットアップについても説明します。	GC88-8533 db2c1x70	db2c1
DB2 コネクト パーソナル・エディション (Linux 版) 概説およびインストール	サポートされる Linux 配布プログラムの DB2 コネクト パーソナル・エディションで、計画、インストール、移行、および構成を行う場合の情報を提供します。	GC88-8528 db2c4x70	db2c4
DB2 データ・リンク・マネージャー 概説およびインストール	AIX および Windows 32 ビット・オペレーティング・システムの DB2 データ・リンク・マネージャーで、計画、インストール、構成を行う場合の情報を提供します。	GC88-8532 db2z6x70	db2z6
DB2 エンタープライズ拡張エディション (UNIX 版) 概説およびインストール	UNIX ベースのプラットフォームでの DB2 エンタープライズ拡張エディションの計画、インストール、および構成に関する情報を提供します。また、この資料はサポートされている多数のクライアントのインストールおよびセットアップについても説明します。	GC88-8530 db2v3x70	db2v3
DB2 エンタープライズ拡張エディション (Windows 版) 概説およびインストール	Windows 32 ビット・オペレーティング・システムの DB2 エンタープライズ拡張エディションで、計画、インストール、および構成を行う場合の情報を提供します。また、この資料はサポートされている多数のクライアントのインストールおよびセットアップについても説明します。	GC88-8529 db2v6x70	db2v6

表 53. DB2 情報 (続き)

名前	記述	資料番号 PDF ファイル名	HTML ディレクトリー
DB2 ユニバーサル・データベース (OS/2 版) 概説およびインストール	OS/2 オペレーティング・システムでの DB2 ユニバーサル・データベースの計画、インストール、移行、および構成に関する情報を提供します。また、この資料はサポートされている多数のクライアントのインストールおよびセットアップについても説明します。	GC88-8534 db2i2x70	db2i2
DB2 ユニバーサル・データベース (UNIX 版) 概説およびインストール	UNIX ベースのプラットフォームでの DB2 ユニバーサル・データベースの計画、インストール、移行、および構成に関する情報を提供します。 また、この資料はサポートされている多数のクライアントのインストールおよびセットアップについても説明します。	GC88-8536 db2ixx70	db2ix
DB2 ユニバーサル・データベース (Windows 版) 概説およびインストール	Windows 32 ビット・オペレーティング・システムの DB2 ユニバーサル・データベースで、計画、インストール、移行、および構成を行う場合の情報を提供します。また、この資料はサポートされている多数のクライアントのインストールおよびセットアップについても説明します。	GC88-8537 db2i6x70	db2i6
DB2 パーソナル・エディション 概説およびインストール	OS/2 および Windows 32 ビット・オペレーティング・システム版の DB2 ユニバーサル・データベース パーソナル・エディションで、計画、インストール、移行、および構成を行う場合の情報を提供します。	GC88-8535 db2i1x70	db2i1
DB2 パーソナル・エディション (Linux 版) 概説およびインストール	サポートされる Linux 配布プログラムの DB2 ユニバーサル・データベース パーソナル・エディションで、計画、インストール、移行、および構成を行う場合の情報を提供します。	GC88-8538 db2i4x70	db2i4
DB2 クエリー・パトローラー インストールの手引き	DB2 クエリー・パトローラーのインストール情報を提供します。	GC88-8526 db2iwx70	db2iw

表 53. DB2 情報 (続き)

名前	記述	資料番号 PDF ファイル名	HTML ディレクトリー
ウェアハウス・マネージャー インストールの手引き	ウェアハウス・エージェント、ウェアハウス・トランスフォーマー、および情報カタログ・マネージャーのインストール情報を提供します。	GC88-8572 db2idx70	db2id
プラットフォーム共通のサンプル・プログラム (HTML 形式)			
サンプル・プログラム (HTML)	DB2 のサポートするすべてのプラットフォームでのプログラム言語用に、サンプル・プログラム (HTML 形式) を提供します。これらのサンプル・プログラムは、参照用としてのみ提供されています。サンプルは、すべてのプログラミング言語で利用できるわけではありません。 HTML サンプルが利用できるのは、 DB2 アプリケーション開発クライアントがインストールされている場合だけです。 プログラムの詳細については、アプリケーション構築の手引き を参照してください。	資料番号なし	db2hs
リリース情報			
DB2 コネクト リリース情報	DB2 コネクトの資料には含められなかった最新の情報が収録されています。	注 #2 を参照してください。	db2cr
DB2 インストール情報	DB2 ブックには含められなかったインストールに関する最新の情報が収録されています。	製品 CD-ROM からのみ利用できます。	
DB2 リリース情報	DB2 ブックには含められなかった DB2 製品とその機能に関する最新の情報が収録されています。	注 #2 を参照してください。	db2ir

注:

1. ファイル名の 6 桁目の文字 *x* は、その資料の言語を表します。たとえば、ファイル名 `db2d0e70` は、*管理の手引き* の英語版であることを示し、ファイル名 `db2d0f70` は同じ資料のフランス語版を示します。資料の言語を表すためにファイル名の 6 桁目で使用されている文字は以下のとおりです。

言語	識別子
ブラジル・ポルトガル語	b
ブルガリア語	u
チェコ語	x
デンマーク語	d
オランダ語	q
英語	e
フィンランド語	y
フランス語	f
ドイツ語	g
ギリシャ語	a
ハンガリー語	h
イタリア語	i
日本語	j
韓国語	k
ノルウェー語	n
ポーランド語	p
ポルトガル語	v
ロシア語	r
簡体字中国語	c
スロベニア語	l
スペイン語	z
スウェーデン語	s
繁体字中国語	t
トルコ語	m

2. DB2 ブックには含められなかった最新の情報が、「リリース情報」で HTML 形式および ASCII ファイルとして利用できます。 HTML 版は、インフォメーション・センターおよび製品 CD-ROM からご利用になれます。 ASCII ファイルの参照方法:
   - UNIX ベースのプラットフォームでは、ファイル `Release.Notes` を参照してください。このファイルは `DB2DIR/Readme/%L` ディレクトリーにあります。ここで `%L` は地域名を、 `DB2DIR` は以下のものを表します。
     - `/usr/lpp/db2_07_01` (AIX の場合)
     - `/opt/IBMdb2/V7.1` (HP-UX、DYNIX/ptx、Solaris、および Silicon Graphics IRIX の場合)
     - `/usr/IBMdb2/V7.1` (Linux の場合)
   - これ以外のプラットフォームでは、ファイル `RELEASE.TXT` を参照してください。このファイルは、製品がインストールされているディレクトリーにあります。OS/2 プラットフォームでは、 **IBM DB2** フォルダーをダブルクリックし、**Release Notes** アイコンをダブルクリックすることもできます。

PDF 資料の印刷

資料のハードコピー版が必要な場合、 DB2 の資料 CD-ROM にある PDF ファイルを印刷することができます。 Adobe Acrobat Reader を使用すれば、資料全体または特定のページを印刷することができます。ライブラリー内の各資料のファイルについては、473ページの表53 を参照してください。

Adobe Acrobat Reader の最新版は、Adobe の Web サイト http://www.adobe.co.jp/ から入手できます。

PDF ファイルは、DB2 の資料 CD-ROM に収録されており、ファイル拡張子 PDF が付いています。 PDF ファイルにアクセスするには以下のようにします。

1. DB2 の資料 CD-ROM を挿入します。 UNIX ベースのプラットフォームの場合は、DB2 資料 CD-ROM をマウントします。マウントの手順については、*概説およびインストール* を参照してください。
2. Acrobat Reader を起動します。
3. 以下に示すいずれかの位置から必要な PDF ファイルを開きます。
   - OS/2 および Windows プラットフォームでは:

     x:¥doc¥*language* ディレクトリー。ここで、*x* は CD-ROM ドライブを、*language* は 2 桁の言語を表す国コード (たとえば、EN は英語) を示します。
   - UNIX ベースのプラットフォームでは、以下のようにします。

     CD-ROM の */cdrom*/doc/*%L* ディレクトリー。ここで、*/cdrom* は CD-ROM のマウント・ポイントを、 *%L* は地域名を表します。

さらに、PDF ファイルを CD-ROM からローカル・ドライブまたはネットワーク・ドライブにコピーし、そこから参照することもできます。

印刷資料の注文方法

ハードコピー版の DB2 ブックは、個別に注文することができます。資料を注文するには、IBM 承認の販売業者または営業担当員に連絡してください。

DB2 オンライン文書

オンライン・ヘルプへのアクセス

すべての DB2 コンポーネントで、オンライン・ヘルプを利用できます。以下の表に、さまざまな種類のヘルプを示します。

ヘルプの種類	内容	利用方法
コマンド・ヘルプ	コマンド行プロセッサーのコマンド構文について説明します。	コマンド行プロセッサーの対話モードから、次のように入力します。 `? command` ここで *command* はキーワードまたはコマンド全体を表します。 たとえば、`? catalog` と入力すると、すべての CATALOG コマンドに関するヘルプが表示され、`? catalog database` と入力すると、 CATALOG DATABASE コマンドのヘルプが表示されます。
クライアント構成アシスタントのヘルプ コマンド・センターのヘルプ コントロール・センターのヘルプ データウェアハウスセンターのヘルプ イベント・アナライザーのヘルプ 情報カタログ・マネージャーのヘルプ サテライト管理センターのヘルプ スクリプト・センターのヘルプ	そのウィンドウまたはノートブックで実行できるタスクについて説明します。このヘルプは、知っておく必要のある概説および前提条件に関する情報を含みます。また、ウィンドウやノートブックの制御の使用方法を示します。	ウィンドウまたはノートブックから、**「ヘルプ (Help)」**押しボタンをクリックするか、または **F1** キーを押します。

ヘルプの種類	内容	利用方法

メッセージ・ヘルプ	メッセージの原因、および取るべき処置を説明します。	コマンド行プロセッサーの対話モードから、次のように入力します。
		`? XXXnnnnn`
		ここで、*XXXnnnnn* は有効なメッセージ識別子を表します。
		たとえば、`? SQL30081` と入力すると、メッセージ SQL30081 に関するヘルプを表示します。
		一度に 1 画面分のメッセージ・ヘルプを表示させるには、次のように入力します。
		`? XXXnnnnn | more`
		メッセージ・ヘルプをファイルに保管するには、次のように入力します。
		`? XXXnnnnn > filename.ext`
		ここで、*filename.ext* はメッセージ・ヘルプを保管するファイルを表します。

SQL ヘルプ	SQL ステートメントの構文について説明します。	コマンド行プロセッサーの対話モードから、次のように入力します。 `help statement` ここで、*statement* は SQL ステートメントを表します。 たとえば、`help SELECT` と入力すると、 SELECT ステートメントのヘルプが表示されます。 **注:** UNIX ベースのプラットフォームでは、SQL ヘルプを利用できません。
SQLSTATE ヘルプ	SQL 状態およびクラス・コードについて説明します。	コマンド行プロセッサーの対話モードから、次のように入力します。 `? sqlstate or ? class code` ここで、*sqlstate* は有効な 5 桁の SQL 状態を、*class code* は SQL 状態の最初の 2 桁を表します。 たとえば、`? 08003` によって SQL 状態 08003 のヘルプが表示され、 `? 08` によってクラス・コード 08 のヘルプが表示されます。

オンライン情報の表示

この製品に付属のブックは、ハイパーテキスト・マークアップ言語 (HTML) ソフトコピー形式です。ソフトコピー形式では情報を検索または表示したり、ハイパーテキスト・リンクを利用して関連情報に移動したりすることができます。また、1 つの端末を超えてライブラリーを容易に共用することができます。

オンライン・ブックやサンプル・プログラムは、 HTML バージョン 3.2 仕様に準拠するすべてのブラウザーを使って表示できます。

オンライン・ブックまたはサンプル・プログラムは、次のようにして表示します。

- DB2 管理ツールを実行している場合、インフォメーション・センターを使用します。
- ブラウザーで、**「ファイル (File)」 —>「ページを開く (Open Page)」** をクリックします。次のようなページを開いて、DB2 情報に関する説明とリンクを表示してください。
  - UNIX ベースのプラットフォームでは、以下のページを開きます。

    INSTHOME/sqllib/doc/%*L*/html/index.htm

    ここで *%L* はロケール名です。
  - その他のプラットフォームでは、以下のページを開きます。

    sqllib\doc\html\index.htm

    パスは DB2 がインストールされているドライブです。

    インフォメーション・センターをインストールしていない場合、 **DB2 Information** アイコンをダブルクリックしてページを開くことができます。このアイコンは、ご使用のシステムに応じて、製品のメイン・フォルダー内または Windows 「スタート」メニューにあります。

Netscape ブラウザーのインストール

システムに Web ブラウザーがインストールされていない場合、製品の箱の中にある Netscape CD-ROM から Netscape をインストールすることができます。インストールに関する詳細な説明については、以下を参照してください。

1. Netscape CD-ROM を挿入します。
2. UNIX ベースのプラットフォームでは、CD-ROM をマウントします。マウントの手順については、*概説およびインストール* を参照してください。
3. インストールの手順については、 CDNAV*nn*.txt ファイルを参照します。ここで、*nn* は 2 桁の言語識別子を表します。ファイルは CD-ROM のルート・ディレクトリーにあります。

インフォメーション・センターを使用した情報へのアクセス

インフォメーション・センターを使用すると、DB2 製品情報にすばやくアクセスすることができます。インフォメーション・センターは、DB2 管理ツールを使用できるすべてのプラットフォームで利用できます。

インフォメーション・センターは「インフォメーション・センター (Information Center)」アイコンをダブルクリックすることによってオープンできます。このアイコンのある場所はシステムによって異なります。メイン・プロダクト・フォルダーか Windows の**「スタート」**メニューのどちらかです。

Windows プラットフォームの DB2 では、ツールバーおよび**ヘルプ**・メニューを使用して、インフォメーション・センターにアクセスすることもできます。

インフォメーション・センターは 6 種類の情報を提供します。適切なタブをクリックすると、種類ごとに提供されているトピックが表示されます。

タスク (Tasks) DB2 を使用して実行できる主要なタスク。

参照 (Reference)
DB2 参照情報 (キーワード、コマンド、API など)。

ブック (Books) DB2 ブック。

トラブルシューティング (Troubleshooting)
エラー・メッセージのカテゴリーと、メッセージに対する回復処置。

サンプル・プログラム (Sample Programs)
DB2 アプリケーション開発クライアントに付属のサンプル・プログラム。 DB2 アプリケーション開発クライアントをインストールしていない場合、このタブは表示されません。

Web WWW 上にある DB2 情報。この情報にアクセスするには、ご使用のシステムから Web への接続が必要です。

リストから項目を 1 つ選択すると、インフォメーション・センターはビューアーを立ち上げて情報を表示します。選択した情報の種類に応じて、ビューアーはシステム・ヘルプ・ビューアー、エディター、または Web ブラウザーです。

インフォメーション・センターには検索機能が備わっており、リストを参照せずに特定のトピックを探すことができます。

テキストの全検索を行うには、インフォメーション・センター内のハイパーテキスト・リンク「**DB2 オンライン情報の検索 (Search DB2 Online Information)**」検索フォームに従います。

通常、HTML 検索サーバーは自動的に始動します。 HTML 情報の検索がうまくいかない場合は、以下の方法の 1 つを使用して、検索サーバーを始動しなければならない場合もあります。

Windows では

「スタート」をクリックし、**「プログラム」—>「IBM DB2」—>「Information」—>「Start HTML Search Server」**を選択します。

OS/2 では

「DB2 (OS/2 版)」フォルダーをダブルクリックして、**「Start HTML Search Server」**アイコンをダブルクリックします。

HTML 情報の検索でこの他の問題が発生した場合は、リリース情報を参照してください。

注: 検索機能は、Linux、DYNIX/ptx、および Silicon Graphics IRIX 環境では利用できません。

DB2 ウィザードの使用

ウィザードを使用すると、各タスクをステップごとに進めることによって、さまざまな管理タスクを遂行することができます。ウィザードは、コントロール・センターおよびクライアント構成アシスタントを通して使用できます。以下の表では、ウィザードとその目的をリストしています。

注: データベース作成、索引作成、マルチサイト更新の構成、およびパフォーマンス構成ウィザードは、区分データベース環境で使用できます。

ウィザード	内容	利用方法
データベース追加 (Add Database)	クライアント・ワークステーション上にデータベースのカタログを作成します。	クライアント構成アシスタントから、**「追加 (Add)」**をクリックします。
データベース・バックアップ *(Back up Database)*	バックアップ計画を決定、作成、およびスケジュールします。	「コントロール・センター (Control Center)」からバックアップするデータベースを右クリックし、**「バックアップ (Backup)」—>「ウィザードを使用するデータベース (Database Using Wizard)」**を選択します。
マルチサイト更新の構成 (Configure Multisite Update)	マルチサイト更新、分散トランザクション、または 2 フェーズ・コミットを構成します。	「コントロール・センター (Control Center)」から、**「データベース (Databases)」**フォルダーを右クリックして、**「マルチサイト更新 (Multisite Update)」**を選択します。

ウィザード	内容	利用方法
データベース作成 (Create Database)	データベースを作成し、いくつかの基本的な構成タスクを実行します。	「コントロール・センター (Control Center)」から、**「データベース (Databases)」**フォルダーを右クリックして、**「作成 (Create)」—> 「ウィザードを使用するデータベース (Database Using Wizard)」**を選択します。
表作成 (Create Table)	基本的なデータ・タイプを選択して、表の基本キーを作成します。	「コントロール・センター (Control Center)」から、**「表 (Tables)」**アイコンを右クリックして、**「作成 (Create)」—>「ウィザードを使用する表 (Table Using Wizard)」**を選択します。
表スペース作成 (Create Table Space)	新しい表スペースを作成します。	「コントロール・センター (Control Center)」から、**「表スペース (Table Spaces)」**アイコンを右クリックして、**「作成 (Create)」—> 「ウィザードを使用する表スペース (Table Space Using Wizard)」**を選択します。
索引作成 (Create Index)	すべての照会について、作成すべき索引および除去すべき索引を提案します。	「コントロール・センター (Control Center)」から、**「索引 (Index)」**アイコンを右クリックして、**「作成 (Create)」—>「ウィザードを使用する索引 (Index Using Wizard)」**を選択します。
パフォーマンス構成 (Performance Configuration)	ビジネス要件に適合するように構成パラメーターを更新して、データベースのパフォーマンスを調整します。	「コントロール・センター (Control Center)」から、調整したいデータベースを右クリックして、**「ウィザードを使用するパフォーマンスの構成 (Configure Performance Using Wizard)」**を選択します。 区分データベース環境では、「Database Partitions」視点から、調整したい最初のデータベース区画を右クリックして、**「ウィザードを使用するパフォーマンスの構成 (Configure Performance Using Wizard)」**を選択します。

ウィザード	内容	利用方法
データベース復元 *(Restore Database)*	障害の後、データベースを回復します。どのバックアップを使用し、どのログを再生するかの判別を支援します。	「コントロール・センター (Control Center)」から復元するデータベースを右クリックし、**「復元 (Restore)」—>「ウィザードを使用するデータベース (Database Using Wizard)」**を選択します。

文書サーバーのセットアップ

デフォルトでは、DB2 情報はローカル・システムにインストールされます。つまり、DB2 情報にアクセスする必要のある各担当者が同じファイルをインストールする必要があります。 DB2 情報を 1 か所に格納するには、次のようにします。

1. ¥sqllib¥doc¥html のすべてのファイルとサブディレクトリーを、ローカル・システムから Web サーバーにコピーします。各ブックには独自のサブディレクトリーがあり、そのブックを構成する必要な HTML および GIF ファイルが入っています。ディレクトリー構造は常に同じ状態に保つ必要があります。
2. Web サーバーを構成して、ファイルを新しい場所で検索するようにします。さらに詳しい情報については、 *インストールおよび構成 補足* の NetQuestion 付録を参照してください。
3. インフォメーション・センターの Java バージョンをご使用の場合は、すべての HTML ファイルのベース URL を指定できます。この URL はブックのリストに使用してください。
4. 資料ファイルが表示されるようになったなら、よく使うトピックにはブックマークを付けておいてください。ブックマークを付けるページは、たとえば以下のものがあります。
   - ブックのリスト
   - 頻繁に使用されるブックの目次
   - 頻繁に参照する情報 (たとえば、ALTER TABLE トピックなど)
   - 検索フォーム

中央のマシンから DB2 ユニバーサル・データベース オンライン文書ファイルを提供する方法については、 *インストールおよび構成 補足* の NetQuestion 付録を参照してください。

オンライン情報の検索

HTML ファイルの情報を検索するには、以下の方法のどれか 1 つを使用してください。

- 最上部にある**「検索 (Search)」**をクリックします。検索フォームを使用して特定のトピックを見つけます。この機能は、 Linux、DYNIX/ptx、または Silicon Graphics IRIX 環境ではご利用になれません。
- 最上部にある**「索引 (Index)」**をクリックします。索引を使用して、ブック内の特定のトピックを見つけます。
- HTML 資料またはヘルプの目次あるいは索引を表示してから、 Web ブラウザーの検索機能を利用して資料内の特定のトピックを見つけます。
- Web ブラウザーのブックマーク機能を使用して、特定のトピックにすばやく戻ります。
- インフォメーション・センターの検索機能を使用して、特定のトピックを検索します。詳しくは、487ページの『インフォメーション・センターを使用した情報へのアクセス』を参照してください。

付録M. 特記事項

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、プログラムまたはサービスの操作性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。実施権の許諾については、下記の宛先に書面にてご照会ください。

〒106-0032 東京都港区六本木 3 丁目 2-31
AP 事業所
IBM World Trade Asia Corporation
Intellectual Property Law & Licensing

本書において、日本では発表されていない IBM 製品 (機械およびプログラム)、プログラミングまたはサービスについて言及または説明する場合があります。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。

本書には技術的な誤りまたは誤植が含まれている可能性があります。本書に対して、周期的に変更が行われ、これらの変更は、文書の新規エディションに組み込まれます。IBM は予告なしに、随時、この文書に記載されている製品またはプログラムに対して、改良または変更を行うことがあります。

本書において IBM 以外の Web サイトに言及している場合がありますが、便宜のため記載しただけであり、決してそれらの Web サイトを推奨するものではありません。それらの Web サイトにある資料は、この IBM 製品の資料の一部ではありません。それらの Web サイトは、お客様の責任でご使用ください。

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

本プログラムのライセンス保持者で、(i) 独自に作成したプログラムとその他のプログラム（本プログラムを含む）との間での情報交換、および (ii) 交換された情報の相互利用を可能にすることを目的として、本プログラムに関する情報を必要とする方は、下記に連絡してください。

IBM Canada Ltd.
Office of the Lab Director
1150 Eglinton Ave. East
Toronto, Ontario
M3C 1H7
CANADA

本プログラムに関する上記の情報は、適切な使用条件の下で使用することができますが、有償の場合もあります。

本書で説明されているライセンス・プログラムまたはその他のライセンス資料は、IBM 所定のプログラム契約の契約条項、IBM プログラムのご使用条件、またはそれと同等の条項に基づいて、 IBM より提供されます。

この文書に含まれるいかなるパフォーマンス・データも、管理環境下で決定されたものです。そのため、他の操作環境で得られた結果は、異なる可能性があります。一部の測定が、開発レベルのシステムで行われた可能性がありますが、その測定値が、一般に利用可能なシステムのものと同じである保証はありません。さらに、一部の測定値が、推定値である可能性があります。実際の結果は、異なる可能性があります。お客様は、お客様の特定の環境に適したデータを確かめる必要があります。

IBM 以外の製品に関する情報は、その製品の供給者、出版物、もしくはその他の公に利用可能なソースから入手したものです。IBM は、それらの製品のテストは行っておりません。したがって、他社製品に関する実行性、互換性、またはその他の要求については確証できません。 IBM 以外の製品の性能に関する質問は、それらの製品の供給者にお願いします。

IBM の将来の方向または意向に関する記述については、予告なしに変更または撤回される場合があり、単に目標を示しているものです。

本書には、日常の業務処理で用いられるデータや報告書の例が含まれています。より具体性を与えるために、それらの例には、個人、企業、ブランド、あるいは製品などの名前が含まれている場合があります。これらの名称はすべて架空のものであり、名称や住所が類似する企業が実在しているとしても、それは偶然にすぎません。

著作権使用許諾:

本書には、様々なオペレーティング・プラットフォームでのプログラミング手法を例示するサンプル・アプリケーション・プログラムがソース言語で掲載されています。お客様は、サンプル・プログラムが書かれているオペレーティング・プラットフォームのア

プリケーション・プログラミング・インターフェースに準拠したアプリケーション・プログラムの開発、使用、販売、配布を目的として、いかなる形式においても、IBM に対価を支払うことなくこれを複製し、改変し、配布することができます。このサンプル・プログラムは、あらゆる条件下における完全なテストを経ていません。従って IBM は、これらのサンプル・プログラムについて信頼性、利便性もしくは機能性があることをほのめかしたり、保証することはできません。

それぞれの複製物、サンプル・プログラムのすべての部分、またはすべての派生した創作物には、次のように、著作権表示を入れていただく必要があります。

© (お客様の会社名) (西暦年). このコードの一部は、IBM Corp. のサンプル・プログラムから取られています。 © Copyright IBM Corp. _年を入れる_. All Rights Reserved.

商標

以下は、 IBM Corporation の商標です。

ACF/VTAM
AISPO
AIX
AIX/6000
AIXwindows
AnyNet
APPN
AS/400
BookManager
CICS
C Set++
C/370
DATABASE 2
DataHub
DataJoiner
DataPropagator
DataRefresher
DB2
DB2 Connect
DB2 Extenders
DB2 OLAP Server
DB2 Universal Database
Distributed Relational Database Architecture
DRDA
eNetwork
Extended Services
FFST
First Failure Support Technology
IBM
IMSIMS/ESA
LAN DistanceMVS
MVS/ESA
MVS/XA
Net.Data
OS/2
OS/390
OS/400
PowerPC
QBIC
QMF
RACF RISC System/6000
RS/6000
S/370
SP
SQL/DS
SQL/400
System/370
System/390
SystemView
VisualAge
VM/ESA
VSE/ESA
VTAM
WebExplorer
WIN-OS/2

以下の用語は、他社の商標あるいは登録商標です。

Tivoli および NetView は、米国およびその他の国における Tivoli Systems Inc. の商標です。

Java およびすべての Java 関連の商標およびロゴは Sun Microsystems, Inc. の米国およびその他の国における商標または登録商標です。

UNIX は、The Open Group がライセンスしている米国およびその他の国における登録商標です。

他の会社名、製品名およびサービス名等はそれぞれ各社の商標または登録商標です。

索引

日本語, 数字, 英字, 特殊文字の順に配列されています。なお, 濁音と半濁音は清音と同等に扱われています。

[ア行]

[カ行]

[サ行]

[タ行]

[ナ行]

[ハ行]

［マ行］

[ヤ行]

[ラ行]

[数字]

A

B

C

D

E

F

G

S

T

U

V

W

[特殊文字]

IBM と連絡をとる

技術上の問題がある場合は、時間をとって「問題判別の手引き」に定義されている処置を検討し、それらの提案を実行した後で、DB2 顧客サービスに連絡をとってください。この資料には、DB2 顧客サービスがお客さまを支援するために必要とする情報が説明されています。

製品情報

以下の情報は英語で提供されます。内容は英語版製品に関する情報です。

http://www.ibm.com/software/data/

DB2 World Wide Web ページには、ニュース、製品説明、研修スケジュールなどの DB2 に関する最新情報が提供されています。ただし、提供されている情報は英語です。

http://www.ibm.com/software/data/db2/library/

「DB2 Product and Service Technical Library」では、よくされる質問 (FAQ)、修正内容、資料、および最新の DB2 技術情報などの情報へのアクセスが提供されています。

注: この情報のご提供は英語のみとなりますのでご注意ください。

http://www.elink.ibmlink.ibm.com/pbl/pbl/

「International Publications」注文用 Web サイトでは、マニュアルの注文方法についての情報を提供しています。ただし、提供されている情報は英語です。

http://www.ibm.com/education/certify/

IBM の「Professional Certification Program」Web サイトでは、 DB2 を含むさまざまな IBM 製品の認証テストの情報を提供しています。ただし、提供されている情報は英語です。

ftp.software.ibm.com

匿名でログオンしてください。ディレクトリー `/ps/products/db2` には、 DB2 および多数の他製品に関連したデモ、修正プログラム、情報、およびツールがあります。ただし、提供されている情報は英語です。

comp.databases.ibm-db2, bit.listserv.db2-l

これらのインターネット・ニュースグループは、ユーザーが DB2 製品に関する自分の経験について話し合うために利用できます。ただし、提供されている情報は英語です。

Compuserve: GO IBMDB2

このコマンドを入力すると、IBM DB2 Family forum にアクセスできます。すべての DB2 製品が、このフォーラムでサポートされています。ただし、提供されている情報は英語です。

米国以外の国で IBM に連絡する方法については、 *IBM Software Support Handbook* の Appendix A を参照してください。この資料にアクセスするには、 Web ページ: http://www.ibm.com/support/ にアクセスし、ページの最下部にある「IBM Software Support Handbook」リンク・ボタンを選択します。

注: 国によっては、 IBM が承認している販売業者が、 IBM サポート・センターの代わりにそれら販売業者のサポート・センターに連絡する場合があります。

Printed in Japan

SC88-8511-01

日本アイ・ビー・エム株式会社
〒106-8711　東京都港区六本木3-2-12