IBM DB2 10.1
for Linux, UNIX, and Windows

データベース・セキュリティー・ガイド

2013 年 1 月更新版

SA88-4695-01
(英文原典：SC27-3872-01)

IBM DB2 10.1
for Linux, UNIX, and Windows

データベース・セキュリティー・ガイド

2013 年 1 月更新版

SA88-4695-01
(英文原典：SC27-3872-01)

ご注意

本書および本書で紹介する製品をご使用になる前に、 419 ページの『付録 B. 特記事項』に記載されている情報をお読みください。

原典： SC27-3872-01
IBM DB2 10.1
for Linux, UNIX, and Windows
Database Security Guide
Updated January, 2013

発行： 日本アイ・ビー・エム株式会社

担当： トランスレーション・サービス・センター

第1刷 2012.12

目次

本書について

「データベース・セキュリティー・ガイド」では、DB2® セキュリティー・フィーチャーを使用して、データベースのインストールで必要なレベルのセキュリティーを実装および管理する方法について説明します。

「データベース・セキュリティー・ガイド」では、以下について詳しく説明しています。

- DB2 データベースにアクセスできるユーザーの認証の管理
- ユーザーによるデータベース・オブジェクトおよびデータへのアクセスを制御するための許可の設定

第 1 章 DB2 のセキュリティー・モデル

DB2 データベース・システムのデータと関数に対するアクセスを制御するセキュリティー・モードが 2 つあります。 DB2 データベース・システムへのアクセスは、DB2 データベース・システムの外部にある機能 (認証) によって管理するのに対し、DB2 データベース・システムの内部のアクセスは、データベース・マネージャー (許可) によって管理します。

認証

認証とは、システムがユーザーの身元を検証するプロセスのことです。ユーザー認証は、DB2 データベース・システムの外部にあるセキュリティー機能によって、認証セキュリティー・プラグイン・モジュールを経由して実行されます。オペレーティング・システム・ベースの認証に依存するデフォルトの認証セキュリティー・プラグイン・モジュールは、DB2 データベース・システムのインストール時に組み込まれます。便宜を図って、DB2 データベース・マネージャーには、Kerberos と Lightweight Directory Access Protocol (LDAP) 用の認証プラグイン・モジュールが備えられています。より一層柔軟に具体的な認証要件に対応するには、独自の認証セキュリティー・プラグイン・モジュールを作成します。

認証プロセスでは、DB2 許可 ID が生成されます。認証時には、ユーザーのグループ・メンバーシップ情報も取得されます。デフォルトのグループ情報取得機能は、オペレーティング・システム・ベースのグループ・メンバーシップ・プラグイン・モジュールに依存しています。そのモジュールは、DB2 データベース・システムのインストール時に組み込まれます。必要に応じて、LDAP などのグループ・メンバーシップ・プラグイン・モジュールを使用して、グループ・メンバーシップ情報を取得することも可能です。

許可

ユーザーが認証されると、データベース・マネージャーは、そのユーザーが DB2 のデータやリソースにアクセスする許可を持っているかどうかを確認します。 DB2 データベース・マネージャーは、許可プロセスの中で、認証済みのユーザーがどのデータベース操作を実行でき、どのデータ・オブジェクトにアクセスできるのかを示す情報を取得します。

許可 ID に与えられる許可のさまざまなソースを以下にまとめます。

1. 1 次許可: 許可 ID に直接付与されている許可。
2. 2 次許可: 許可 ID がメンバーとして属しているグループやロールに付与されている許可。
3. パブリック許可: PUBLIC に付与されている許可。
4. コンテキスト依存許可: トラステッド・コンテキスト・ロールに付与されている許可。

ユーザーに付与できる許可のカテゴリーを以下にまとめます。

- システム・レベルの許可

システム管理者 (SYSADM)、システム制御 (SYSCTRL)、システム保守 (SYSMAINT)、システム・モニター (SYSMON) の各権限によって、インスタンス・レベルの機能をさまざまな程度で制御します。権限は、特権をグループ化するため、およびインスタンス、データベース、データベース・オブジェクトに対する保守およびユーティリティー操作を制御するための手段として使用します。

- データベース・レベルの許可

  セキュリティー管理者 (SECADM)、データベース管理者 (DBADM)、アクセス制御 (ACCESSCTRL)、データ・アクセス (DATAACCESS)、SQL 管理者 (SQLADM)、ワークロード管理の管理者 (WLMADM)、および Explain (EXPLAIN) の各権限によって、データベース内のアクセスを制御します。その他のデータベース権限としては、LOAD (表にデータをロードする権限)、CONNECT (データベースに接続する権限) があります。

- オブジェクト・レベルの許可

  オブジェクト・レベルの許可では、オブジェクトに対する操作の実行時に特権をチェックします。例えば、表のデータを選択 (抽出) するユーザーには、最低でも表に対する SELECT 特権が必要です。

- 内容に基づく許可

  特定のユーザーが読み取れる表の列や行を制御するためにビューを使用します。個々の行や個々の列に対する読み取りアクセスと書き込みアクセスを持っているユーザーを判別するために、ラベル・ベースのアクセス制御 (LBAC) を使用します。

これらの機能と、アクセスをモニターするための DB2 監査機能を併用して、それぞれのデータベース・インストール・システムで必要なレベルのセキュリティーを定義し、管理します。

認証

ユーザーの認証は、DB2 データベース・システムの外部のセキュリティー機能を使用して完了します。セキュリティー機能は、オペレーティング・システムの一部であるか、別個の製品にすることができます。

セキュリティーは、ユーザーを認証するのに 2 つの項目を必要とします。それは、ユーザー ID とパスワードです。ユーザー ID は、セキュリティー機能にユーザーを知らせます。正しいパスワード、ユーザーおよびセキュリティー機能にのみ認識されている情報を提供すれば、ユーザーの身元 (ユーザー ID に対応) が検証されます。

注: 非ルート・インストールでは、オペレーティング・システム・ベースの認証は、**db2rfe** コマンドを実行して有効にしなければなりません。

認証された後、次のようにします。

- ユーザーは SQL 許可名または *authid* を使用して、DB2 に識別されなければなりません。この名前は、ユーザー ID と同じものか、またはマップ値にすることができます。例えば、UNIX オペレーティング・システムでは、デフォルトのセ

キュリティー・プラグイン・モジュールを使用している場合、DB2 *authid* は、DB2 命名規則に従ったUNIX ユーザー ID を大文字に変換することによって得られます。

- そのユーザーが属しているグループのリストが取得されます。グループ・メンバーシップは、ユーザーを許可するときに使用されます。グループは、DB2 許可名にもマップする必要がある、セキュリティー機能のエンティティーです。このマッピングは、ユーザー ID のために使用される方法と類似した方法で行われます。

DB2 データベース・マネージャーは、セキュリティー機能を使用して、以下の 2 つの方法のうちの 1 つでユーザーを認証します。

- 成功したセキュリティー・システム・ログインを識別の証拠として使用し、次のことを可能にします。
  - ローカル・データをアクセスするためのローカル・コマンドの使用
  - サーバーがクライアント認証を委託している場合のリモート接続の使用
- ユーザー ID とパスワードの妥当性検査がセキュリティー機能によって成功すると、それをユーザーのアイデンティティーの証拠として使用して、以下のことを許可します。
  - サーバーが認証の検査を必要とするリモート接続の使用
  - ログインに使用されたアイデンティティー以外のアイデンティティーの下でユーザーがコマンドを実行する場合の操作の使用

注: 一部の UNIX システムでは、DB2 データベース・マネージャーはオペレーティング・システムで失敗したパスワード入力をロギングし、LOGINRETRIES パラメーターで指定されたログイン試行の許可回数をクライアントが超過したときを検出します。

許可

許可は、DB2 の機能を使用して実行されます。それぞれの許可名に関連する許可事項を記録するために、 DB2 表と構成ファイルが使用されます。

認証済みユーザーがデータへのアクセスを試行すると、このように記録された許可は、以下の許可と比較されます。

- ユーザーの許可名
- ユーザーが所属するグループ
- ユーザーに対して直接的に、あるいはグループまたは役割を通して間接的に付与されている役割
- トラステッド・コンテキストを介して獲得された許可

この比較に基づいて、DB2 サーバーは、要求されたアクセスを許すかどうかを決定します。

記録される許可事項のタイプは、「特権」、「権限レベル」、および「LBAC 信用証明情報」の 3 つです。

特権 は、 1 ユーザーがデータベース・リソースを作成またはアクセスできるようにするために、 1 つの許可名に対して単一の許可事項を定義します。特権は、データベース・カタログに保管されます。

権限レベル は、特権をグループ化する方法を提供し、データベース・マネージャーの操作を制御します。データベース固有の権限は、データベース・カタログに保管されます。また、システム権限は、グループ・メンバーシップと関連付けられ、権限レベルに関連するグループ名は、特定のインスタンスについて、データベース・マネージャー構成ファイルの中に保管されます。

*LBAC 信用証明情報*は、ラベル・ベースのアクセス制御 (LBAC) によって保護されているデータへのアクセスを許可する LBAC セキュリティー・ラベルおよび LBAC 規則の免除です。 LBAC 信用証明情報は、データベース・カタログに保管されます。

グループは、それぞれのユーザーに個別に特権の付与または取り消しを行うことを必要とせずに、ユーザーの集合に対して許可を実行するための便利な手段を提供します。特に異なる指定がなければ、グループ許可名は、許可名が許可の目的で使用されるところであれば、どこでも使用することができます。一般に、グループ・メンバーシップは、動的 SQL およびデータベース以外のオブジェクト (インスタンス・レベルのコマンドおよびユーティリティーなど) の許可のためのものと考えられ、静的 SQL のためのものとは考えられません。この一般的なケースの例外は、特権が PUBLIC に与えられるときに生じ、この場合は静的 SQL が処理されるときに考慮されます。グループ・メンバーシップが適用されない特定のケースについては、DB2 の資料全体を通して、該当する場合にその旨の注が付いています。

ロールは、1 つ以上の特権をまとめたデータベース・オブジェクトですが、これを、GRANT ステートメントを使用してユーザー、グループ、PUBLIC、またはその他のロールに割り当てるか、あるいは、CREATE TRUSTED CONTEXT または ALTER TRUSTED CONTEXT ステートメントを使用して、トラステッド・コンテキストに割り当てることができます。ワークロード定義内で、SESSION_USER ROLE 接続属性用のロールを指定することができます。ロールの使用時には、データベース・オブジェクトに対するアクセス許可をそのロールに関連付けます。すると、そのロールのメンバーであるユーザーは、データベース・オブジェクトへのアクセス時に使用するロールに合わせて定義された特権を持つことになります。

ロールは、グループに似た機能を備えています。つまりロールは、各ユーザーごとに個別の特権の付与または取り消しを行うという手間をかけずに、ユーザーの集合に対して許可を実行します。ロールの利点の 1 つとして、これは、DB2 データベース・システムによって管理されます。ビュー、トリガー、マテリアライズ照会表 (MQT)、パッケージ、および SQL ルーチンの許可プロセスでは、グループに付与された許可とは違って、ロールに認可された許可は、検討の対象になります。ビュー、トリガー、MQT、パッケージ、および SQL ルーチンの許可プロセスでは、グループに付与された許可が検討の対象にならない理由は、DB2 データベース・システムは、グループ内でメンバーシップがいつ変更になったかを検出できないため、前述のオブジェクトを必要に応じて無効化できないからです。

注: ビュー、トリガー、MQT、パッケージ、および SQL ルーチンの許可プロセスでは、グループに付与されたロールに付与された許可は、検討の対象になりません。

SQL ステートメントの処理中に DB2 許可モデルが検討の対象とするのは、以下の一連の許可です。

1. その SQL ステートメントに関連付けられている 1 次許可 ID に付与された許可。
2. その SQL ステートメントに関連付けられている 2 次許可 ID (グループまたはロール) に付与された許可。
3. PUBLIC に直接付与されたロールや、または他のロールを介して間接的に付与されたロールも含め、PUBLIC に付与された許可。
4. トラステッド・コンテキスト・ロールに付与された許可 (該当する場合)。

DB2 データベース・マネージャーのインストールおよび使用時のセキュリティー問題

セキュリティーに関する考慮事項は、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。

DB2 データベース・マネージャーのインストールを完了するためには、ユーザー ID、グループ名、およびパスワードが必要です。 GUI ベースの DB2 データベース・マネージャーのインストール・プログラムは、さまざまなユーザー ID とグループのデフォルト値を作成します。 Linux および UNIX オペレーティング・システムにインストールする場合と Windows オペレーティング・システムにインストールする場合とでは、作成されるデフォルト値が異なります。

- UNIX および Linux オペレーティング・システムの場合、インスタンスのセットアップ・ウィンドウで DB2 インスタンスを作成することを選択すると、 DB2 データベース・インストール・プログラムはデフォルトで、それぞれ異なるユーザーを DAS (dasusr)、インスタンス所有者 (db2inst)、および fenced ユーザー (db2fenc) のために作成します。オプションとして、異なるユーザー名を指定することもできます。

  DB2 データベース・インストール・プログラムは、まだ存在していないユーザー ID を作成するため、デフォルト・ユーザー名に 1 から 99 までの数値を順番に付加します。例えば、ユーザー db2inst1 および db2inst2 がすでに存在する場合、 DB2 データベース・インストール・プログラムはユーザー db2inst3 を作成します。 10 よりも大きな数値が使用される場合、デフォルト・ユーザー ID の名前の文字部分が切り捨てられます。例えば、ユーザー ID db2fenc9 がすでに存在する場合、 DB2 データベース・インストール・プログラムはユーザー ID の c を切り捨てて、10 を付加します (db2fen10)。デフォルトの DAS ユーザーに数値が付加されるときに切り捨ては生じません (dasusr24 など)。
- Windows オペレーティング・システムの場合、DB2 データベース・インストール・プログラムはデフォルトで、 DAS ユーザー、インスタンス所有者、および fenced ユーザーのためにユーザー db2admin を作成します (必要に応じて、これとは別のユーザー名をセットアップ時に指定することができます)。 Linux および UNIX オペレーティング・システムの場合とは異なり、ユーザー ID に数値は付加されません。

管理者以外のユーザーがデフォルトを知って、データベースおよびインスタンス内で不適切な方式で使用するリスクを最小限にするには、インストールの際にデフォルトを新規または既存の任意のユーザー ID に変更します。

注: 応答ファイルのインストールでは、ユーザー ID またはグループ名にデフォルト値を使用しません。これらの値を、応答ファイルに指定しておく必要があります。

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。 Linux および 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。この場合、定義されたパスワードを持っていないユーザーは、NULL パスワードを持っているものと見なされます。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を使用します。

Linux および UNIX オペレーティング・システムで、パーティション・データベース環境 を操作する場合、リモート・メンバーでコマンドを実行するために、DB2 データベース・マネージャーはデフォルトで rsh ユーティリティー (HP-UX では remsh) を使用します。rsh ユーティリティーはネットワーク上で平文のパスワードを伝送するため、DB2 サーバーがセキュア・ネットワーク上にない場合には機密漏れが生じる可能性があります。**DB2RSHCMD** レジストリー変数を使用することにより、リモート・シェル・プログラムを、この機密漏れを防ぐ、より安全な別のものに設定できます。より安全な別のプログラムの一例として、ssh があります。リモート・シェル構成の制限については、**DB2RSHCMD** レジストリー変数の資料を参照してください。

DB2 データベース・マネージャーをインストールした後に、ユーザーに付与されたデフォルト特権の検討および (必要であれば) 変更も行ってください。デフォルトでは、インストール処理によってシステム管理 (SYSADM) の特権が、各オペレーティング・システム上で下記のユーザーに与えられます。

Linux および UNIX オペレーティング・システム
インスタンス所有者の 1 次グループに属する有効な DB2 データベースのユーザー名に対して。

Windows 環境

- ローカル Administrators グループのメンバーに対して。
- DB2 データベース・マネージャーが、ユーザーが定義されたロケーション上でそれらのユーザーのグループを列挙するように構成されている場合、ドメイン・コントローラーの Administrators グループのメンバーに対して。**DB2_GRP_LOOKUP** 環境変数を使用して、Windows オペレーティング・システムでグループの列挙を構成します。
- Windows の拡張セキュリティーが使用可能になっている場合、DB2ADMNS グループのメンバーに対して。DB2ADMNS グループのロケーションは、インストール時に決定されます。
- LocalSystem アカウントに対して

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、管理者はどのユーザーのグループが SYSADM 特権を持つようにするかを制御できます。 DB2 データベース・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セキュリティー要件を完成するために、次のガイドラインを使用し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、対応する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。

管理者は、特定インスタンスに関連付けられていると認識しやすい、インスタンス所有者ユーザー ID を作成することを考慮してください。このユーザー ID は、そのグループの名前の 1 つとして以前に作成された SYSADM グループの名前を持つ必要があります。もう一つの方法としては、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、インスタンスを変更できるユーザー ID およびグループの急増を制御できます。

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を許可される前に認証を行うため、 1 つのパスワードと関連付ける必要があります。パスワード作成時には、組織のパスワード命名ガイドラインに従うことをお勧めします。

注: インスタンス構成ファイルまたはその他のファイルを誤って削除したり上書きしたりしてしまうことを防ぐために、管理者は、サーバー上で直接実行される日常の管理タスク用に、インスタンス所有者と同じ 1 次グループに属さない別のユーザー・アカウントを使用することを検討する必要があります。

インスタンス・ディレクトリーとデータベース・ディレクトリーのファイル許可要件

DB2 データベース・システムでは、インスタンス・ディレクトリーとデータベース・ディレクトリーに最小レベルの許可を設定する必要があります。

注: インスタンス・ディレクトリーとデータベース・ディレクトリーが DB2 データベース・マネージャーによって作成された場合は、許可が正確に設定されているので、変更するべきではありません。

UNIX マシンと Linux マシンのインスタンス・ディレクトリーと `NODE000x/sqldbdir` ディレクトリーの最小の許可は、`u=rwx` と `go=rx` でなければなりません。これらの文字の意味を以下の表で説明します。

文字	意味
u	ユーザー (所有者)
g	グループ
o	他のユーザー
r	読み取り

文字	意味
w	書き込み
x	実行

例えば、/home にあるインスタンス db2inst1 の許可は、以下のとおりです。

```
drwxr-xr-x  36 db2inst1  db2grp1         4096 Jun 15 11:13 db2inst1
```

データベースが組み込まれているディレクトリーについては、NODE000*x* までのあらゆるディレクトリー・レベル (このレベルも含む) で以下の許可が必要です。

```
drwxrwxr-x  11 db2inst1  db2grp1         4096 Jun 14 15:53 NODE0000/
```

例えば、データベースが /db2/data/db2inst1/db2inst1/NODE0000 にあれば、ディレクトリー /db2、/db2/data、/db2/data/db2inst1、/db2/data/db2inst1/db2inst1、/db2/data/db2inst1/db2inst1/NODE0000 で drwxrwxr-x が必要になります。

NODE000x ディレクトリーの中では、sqldbdir ディレクトリーで許可 drwxrwxr-x が必要です。例えば、次のようになります。

```
drwx------   5 db2inst1  db2grp1          256 Jun 14 14:17 SAMPLE/
drwxr-x---   7 db2inst1  db2grp1         4096 Jun 14 13:26 SQL00001/
drwxrwxr-x   2 db2inst1  db2grp1          256 Jun 14 13:02 sqldbdir/
```

注意:
ファイルのセキュリティーを維持するために、*DBNAME* ディレクトリー (SAMPLE など) と SQL*xxxx* ディレクトリーの許可は、DB2 データベース・マネージャーによってそれらのディレクトリーが作成されたときに割り当てられた許可から変更しないでください。

認証の詳細

サーバーでの認証方式

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。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。

フェデレーテッド・データベースからデータ・ソースにアクセスする場合、データ・ソース認証処理およびフェデレーテッド認証タイプの定義を考慮する必要があります。

注: SERVER_ENCRYPT 認証の使用時にユーザー ID とパスワードを暗号化するため、および DATA_ENCRYPT 認証の使用時にユーザー ID、パスワード、およびユーザー・データを暗号化するために、DB2 データベース管理システムによって使用

される暗号ルーチンに関する証明情報について、http://www.ibm.com/security/standards/st_evaluations.shtml の Web サイトを確認することができます。

明示的トラステッド接続でのユーザーの切り替え

CLI/ODBC および XA CLI/ODBC アプリケーションでは、認証が必要となるユーザー切り替え要求を処理する時に使用される認証メカニズムは、最初にトラステッド接続自体を確立するために使用されるメカニズムと同じです。そのため、明示的トラステッド接続の確立中に使用された他のすべての折衝されたセキュリティー属性 (暗号化アルゴリズム、暗号鍵、プラグイン名など) は、そのトラステッド接続でのユーザー切り替え要求に必要なすべての認証の場合と同じと見なされます。 Java™ アプリケーションでは、ユーザー切り替え要求における認証方式を (データ・ソース・プロパティーを使用することにより) 変更することができます。

トラステッド・コンテキスト・オブジェクトは、トラステッド接続でのユーザーの切り替えに認証を必要としない ように定義できるため、明示的トラステッド接続でのユーザー切り替えフィーチャーを十分に活用するためには、ユーザー作成のセキュリティー・プラグインで以下の操作ができなければなりません。

- ユーザー ID のみのトークンを受け入れる
- そのユーザー ID の有効な DB2 許可 ID を戻す

注: CLIENT タイプの認証が有効である場合には、明示的トラステッド接続は確立できません。

指定された認証タイプ

以下の認証タイプがあります。

SERVER

その構成に有効なセキュリティー・メカニズム (例えば、セキュリティー・プラグイン・モジュール) 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。デフォルトのセキュリティー・メカニズムは、接続の試行中にユーザー 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送信され、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るというものです。

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。

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。ユーザー ID とパスワードは、クライアントからサーバーにネットワークを介して送信される際に、暗号化されます。

クライアントとサーバー間の認証方式が SERVER_ENCRYPT の場合、AES (Advanced Encryption Standard) 256 ビット・アルゴリズムを使用してユーザー ID とパスワードを暗号化することができます。このためには、

alternate_auth_enc データベース・マネージャー構成パラメーターを設定します。この構成パラメーターには、以下の 3 つの設定があります。

- NOT_SPECIFIED (デフォルト) は、クライアントが提案した暗号化アルゴリズム (AES 256 ビット・アルゴリズムを含む) をサーバーが受け入れることを意味します。
- AES_CMP は、接続クライアントが DES を提案しても、AES 暗号化をサポートしているのであれば、サーバーは AES 暗号化を求めて再び折衝するということを示します。
- AES_ONLY は、サーバーが AES 暗号化のみを受け入れることを意味します。クライアントが AES 暗号化をサポートしていない場合、接続は拒否されます。

AES 暗号化を使用できるのは、クライアントとサーバー間で折衝された認証方式が SERVER_ENCRYPT の場合のみです。

CLIENT

オペレーティング・システムのセキュリティーを使用して、アプリケーションが呼び出されたデータベース・パーティション上で認証が行われることを指定します。接続の試行中にユーザー ID およびパスワードが指定されると、それらがクライアント・ノード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。これはしばしば、シングル・サインオンと呼ばれます。

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 レベルのセキュリティー

トラステッド・クライアントとは、信頼できるローカル・セキュリティー・システムをもつクライアントのことです。

CLIENT の認証タイプが選択されている場合、固有のセキュリティーをオペレーティング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。

非セキュアの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安全な固有のセキュリティー・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。

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的に提供されないと、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 または USING 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。

すべてのクライアント (z/OS® および System i® 上の JCC タイプ 4 クライアントは含むが、z/OS、OS/390®、VM、VSE、および System i 上のネイティブ DB2 クライアントは含まない) から保護するためには、**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、クライアント側の認証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、前述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 (ユーザー ID およびパスワードが指定されなかった場合) およびサーバー (ユーザー ID およびパスワードが指定された場合) で行われます。

表 1. 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非 DRDA® クライアント認証、ユーザー ID およびパスワードなし	非トラステッドである非 DRDA クライアント認証、ユーザー ID およびパスワードあり	トラステッドである非 DRDA クライアント認証、ユーザー ID およびパスワードなし	トラステッドである非 DRDA クライアント認証、ユーザー ID およびパスワードあり	DRDA クライアント認証、ユーザー ID およびパスワードなし	DRDA クライアント認証、ユーザー ID およびパスワードあり
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

DATA_ENCRYPT

サーバーは、暗号化された SERVER 認証スキーマおよびユーザー・データの暗号化を受け入れます。認証が機能する方法は、SERVER_ENCRYPT に関して示した方法と同じです。ユーザー ID とパスワードは、クライアントからサーバーにネットワークを介して送信される際に、暗号化されます。

この認証タイプを使用するとき、以下のユーザー・データが暗号化されます。

- SQL および XQuery ステートメント
- SQL プログラム変数データ
- SQL または XQuery ステートメントのサーバー処理の出力データで、データについての説明を含むもの
- 照会から生じる応答セット・データの一部またはすべて
- ラージ・オブジェクト (LOB) データ・ストリーム
- SQLDA 記述子

DATA_ENCRYPT_CMP

サーバーは、暗号化された SERVER 認証スキーマおよびユーザー・データの暗号化を受け入れます。さらに、この認証タイプは DATA_ENCRYPT 認証タイプをサポートしていない下位レベルの製品との互換性を可能にします。これらの製品は、SERVER_ENCRYPT 認証タイプを使って、暗号化ユーザー・データがない状態での接続を許可されます。新しい認証タイプをサポートしている製品は、これを使用する必要があります。この認証タイプは、サーバーのデータベース・マネージャー構成ファイル内のみで有効であり、**CATALOG DATABASE** コマンドで使用するときには無効です。

KERBEROS

DB2 クライアントとサーバーが両方とも、 Kerberos セキュリティー・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。 Kerberos セキュリティー・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、ユーザーの身元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平文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。 Kerberos セキュリティー・プロトコルにより、リモート DB2 データベース・サーバーへのシングル・サインオンを行えるようになります。 KERBEROS 認証タイプはさまざまなオペレーティング・システムでサポートされています。詳しい情報については、関連情報セクションを参照してください。

Kerberos 認証は、以下のように行われます。

1. ドメイン・アカウントを使用してクライアント・マシンにログオンしているユーザーは、ドメイン・コントローラーの Kerberos 鍵配布センター (KDC) へ認証を行います。鍵配布センターはチケット許可チケット (TGT) をクライアントに発行します。
2. 接続の最初の段階で、サーバーはクライアントにターゲット・プリンシパル名を送信します。これは、DB2 データベース・サーバー・サービスのサービス・アカウント名です。サーバーのターゲット・プリンシパル名、およびチケット許可チケットを使用して、クライアントはチケット

許可サービス (TGS) (これもドメイン・コントローラーにあります) へサービス・チケットを要求します。クライアントのチケット許可チケット、およびサーバーのターゲット・プリンシパル名の両方が有効であれば、 TGS はクライアントへサービス・チケットを発行します。データベース・ディレクトリーに記録されるプリンシパル名は、name/instance@REALM の形式で指定できます。 (これは、Windows で指定可能な DOMAIN¥userID および userID@xxx.xxx.xxx.com フォーマットに追加されるものです。)

3. クライアントはこのサービス・チケットを通信チャネル (例えば、TCP/IP) を使用してサーバーへ送信します。
4. サーバーはクライアントのサーバー・チケットの妥当性検査を行います。クライアントのサービス・チケットが有効であれば、これで認証は完了します。

クライアント・マシンでデータベースをカタログし、サーバーのターゲット・プリンシパル名と共に Kerberos 認証タイプを明示的に指定することも可能です。そうすれば、接続の最初の段階はバイパスすることができます。

ユーザー ID およびパスワードが指定されている場合、クライアントはそのユーザー・アカウントに対するチケット許可チケットを要求し、それを認証に使用します。

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 Kerberos セキュリティー・システムを使用して認証されます。クライアント認証が SERVER_ENCRYPT である場合、クライアントはユーザー ID および暗号化パスワードを使用して認証されます。クライアント認証が指定されない場合、クライアントは Kerberos が使用可能であればそれを使用し、それが使用可能でなければパスワード暗号化を使用して認証されます。その他のクライアント認証タイプでは、認証エラーが戻されます。クライアントの認証タイプを KRB_SERVER_ENCRYPT として指定することはできません。

注: Kerberos 認証タイプは特定のオペレーティング・システムで実行されているクライアントおよびサーバーでサポートされています。詳しい情報については、関連情報セクションを参照してください。Windows オペレーティング・システムでは、クライアントおよびサーバー・マシンは両方とも同じ Windows ドメインに属しているか、またはトラステッド・ドメインに属していなければなりません。この認証タイプは、サーバーが Kerberos をサポートしており、クライアント・マシンのいくつか (すべてである必要はありません) が Kerberos 認証をサポートしている場合に使用してください。

GSSPLUGIN

サーバーが認証を行うために GSS-API プラグインを使用するように指定します。クライアント認証が指定されていない場合、サーバーは **srvcon_gssplugin_list** データベース・マネージャー構成パラメーターにリストされている Kerberos プラグインを含む、サーバーによってサポートされているプラグインのリストをクライアントに戻します。クライアントは、クライアント・プラグイン・ディレクトリーにある最初のプラグインをリス

トから選択します。クライアントがリスト内のどのプラグインもサポートしない場合、そのクライアントは Kerberos 認証方式 (それが戻される場合) を使用して認証されます。クライアント認証が GSSPLUGIN 認証方式の場合、クライアントはリスト内にあるサポートされる最初のプラグインを使用して認証されます。

GSS_SERVER_ENCRYPT
サーバーが、プラグイン認証または暗号化されたサーバー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プラグインを介して行われる場合、クライアントはサーバーがサポートするプラグインのリストにある、クライアントがサポートする最初のプラグインを使用して認証されます。

クライアント認証が指定されないで暗黙的接続が行われる場合 (つまり、接続が確立されるときにクライアントがユーザー ID とパスワードを供給しない場合)、サーバーはサーバーがサポートするプラグインのリスト、Kerberos 認証方式 (リスト内のプラグインの 1 つが Kerberos に基づくものである場合)、および暗号化サーバー認証方式を戻します。クライアントは、クライアント・プラグイン・ディレクトリーにある、最初にサポートされているプラグインを使用して認証されます。クライアントがリスト内のどのプラグインもサポートしない場合、そのクライアントは Kerberos 認証方式を使用して認証されます。クライアントが Kerberos 認証をサポートしない場合、そのクライアントは暗号化サーバー認証方式を使用して認証され、パスワードがないために接続が失敗します。クライアントは、 DB2 が提供する Kerberos プラグインがオペレーティング・システムに対して存在するか、または Kerberos ベースのプラグインが **srvcon_gssplugin_list** データベース・マネージャー構成パラメーターに指定されている場合に、Kerberos 認証方式をサポートします。

クライアント認証を指定しないで明示接続が実行されている場合 (つまり、ユーザー ID とパスワードの両方が供給されている場合)、認証タイプは SERVER_ENCRYPT と等しくなります。この場合、ユーザー ID とパスワードの暗号化に使用する暗号化アルゴリズムとして何を選択するかは、**alternate_auth_enc** データベース・マネージャー構成パラメーターの設定に依存します。

注:

1. 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 データベース・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イ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・システムのセキュリティー・ユーザーを使用して、通常の DB2 データベース・セキュリティー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、このセキュリティー上の迂回は、データベース・マネージャー構成ファイルのローカル更新にのみ制限されています。フェイルセーフのためのユーザーは、リモートで、または他の DB2 データベース・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
- Windows プラットフォームの場合: ローカル「管理者」グループに属する人
- 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・セキュリティーがないため、すべてのユーザーがローカル・セキュリティー検査に合格します。

リモート・クライアントの認証に関する考慮事項

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。

認証タイプは必須ではありません。認証タイプが指定されていない場合、クライアントはまず SERVER_ENCRYPT 認証タイプを使用して接続を試行します。サーバーが SERVER_ENCRYPT をサポートしていない場合、サーバーは、サポートしている認証タイプのリストを返します。クライアントは、リストされている最初の認証タイプを使用してサーバーに接続します。指定されていない場合、LIST DATABASE DIRECTORY コマンドを使用してリストされたデータベース・カタログには認証タイプが表示されません。認証タイプがデータベース・ディレクトリー項目で指定されていない場合、クライアントが接続するまでに時間がかかることがあります。認証タイプが指定されると、指定された値がサーバー側の値と一致した場合に、認証は即時に開始できます。不一致が検出された場合、DB2 データベースはリカバリーを試行します。リカバリーにより、相違を調整するためにさらに多くのフローが実行されるか、または DB2 データベースがリカバリー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見なされます。

認証タイプ DATA_ENCRYPT_CMP は、データ暗号化をサポートしない、前のリリースのクライアントが、DATA_ENCRYPT ではなく SERVER_ENCRYPT 認証を使用してサーバーに接続できるようにするために設計されています。この認証は、以下の条件が当てはまる場合は機能しません。

- クライアント・レベルがバージョン 7.2。
- ゲートウェイ・レベルがバージョン 8 フィックスパック 7 以降。
- サーバーがバージョン 8 フィックスパック 7 以降。

上記がすべて当てはまる場合、クライアントはサーバーに接続できません。接続できるようにするには、クライアントをバージョン 8 以降にアップグレードするか、またはバージョン 8 フィックスパック 6 以前のゲートウェイ・レベルを使用する必要があります。

接続時に使用される認証タイプは、ゲートウェイのデータベース・カタログ項目として適切な認証タイプを指定することによって決定されます。これは、DB2 Connect™ のシナリオと、パーティション・データベース環境 (クライアントの **DB2NODE** レジストリー変数が設定されている) のクライアント/サーバーのどちらにも当てはまります。適切なパーティションに「ホップ」する目的で、認証タイプをカタログ・パーティションでカタログします。このシナリオの場合、ネゴシエーションはもっぱらクライアント/サーバー間で行われるので、ゲートウェイでカタログされる認証タイプは使用されません。

認証タイプの異なるクライアントを必要とする場合には、異なる認証タイプを使って複数のデータベース別名をゲートウェイでカタログする必要が生じるかもしれません。ゲートウェイでカタログする認証タイプを決める際には、その認証タイプを、クライアントおよびサーバーで使用されるものと同じにすることができます。あるいは、NOTSPEC がデフォルトの SERVER になることを理解した上で、NOTSPEC 認証タイプを使用することもできます。

パーティション・データベースの認証に関する考慮事項

パーティション・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。

Kerberos 認証

Kerberos は、共有秘密鍵システムを使用して非セキュア・ネットワーク環境でユーザーを認証する、サード・パーティーのネットワーク認証プロトコルです。DB2 データベース・システムは、AIX®、HP-UX、Solaris、Linux IA32 および AMD64、および Windows オペレーティング・システムで Kerberos 認証プロトコルをサポートします。

概要

Kerberos 認証は 3 層のシステムで管理され、プレーン・テキストのユーザー ID とパスワードのペアではなく、暗号化されたサービス・チケットのやり取りがアプリケーション・サーバーとクライアントの間で行われます。*資格情報* と呼ばれるこれらの暗号化されたサービス・チケットは、Kerberos 鍵配布センター (KDC) という別個のサーバーによって提供されます。資格情報は存続期間が有限で、クライアントとサーバーのみがこれを認識できます。これらの機能により、たとえチケットがネットワークから傍受された場合でも、機密漏れのリスクが軽減されます。各ユーザー (Kerberos ではプリンシパル という) は、 KDC によって共有される暗号化された秘密鍵を所有します。 1 つの KDC に登録されたプリンシパルとコンピューターは、*レルム* と呼ばれます。

Kerberos の主なフィーチャーの 1 つは、シングル・サインオン環境を実現できることです。ユーザーは Kerberos レルム内のさまざまなリソースにアクセスするために、ID の照合を 1 回行うだけで済みます。このシングル・サインオン環境とは、ユーザー ID およびパスワードを提供しなくてもユーザーが DB2データベース・サーバーに接続またはアタッチできる環境であるということです。もう 1 つの利点は、ユーザー ID の管理が簡略化されることです。これは、Kerberos がプリンシパル用の中央リポジトリーを使用するためです。さらに、Kerberos は相互認証をサポートするため、クライアントはサーバーの身元を確認することができます。

セットアップ

DB2 データベース・システムで Kerberos を使用できるようにするには、事前にすべてのコンピューターに Kerberos レイヤーをインストールして構成する必要があります。標準的な構成の場合は、以下の要件を満たす必要があります。

- 適切なプリンシパルを作成する。
- クライアント・コンピューター、サーバー・コンピューター、およびプリンシパルが同じレルムに属しているか、トラステッド・レルムに属している。トラステッド・レルムは、Windows の用語では、トラステッド・ドメインと呼ばれます。
- 適切な場合には、サーバーのキータブ・ファイルを作成する。
- すべてのコンピューターの時間クロックを同期する。通常、Kerberos では 5 分の時間スキューが許容されます。時間スキューが 5 分を超える場合、資格情報の取得を試みる際に事前認証エラーが発生します。

DB2 サーバーでの Kerberos のセットアップ

DB2 データベース・システムで Kerberos 認証を使用できるようにするには、前もってすべてのコンピューターに Kerberos レイヤーをインストールして構成する必要があります。標準的な構成の場合は、このページの指示に従う必要があります。

始める前に

Linux、Sun Solaris、または HP-UX オペレーティング・システムを使用している場合は、`krb5` ライブラリー以外の Kerberos ライブラリーがシステムにインストールされていないことを確認してください。そうでない場合、Kerberos 認証が失敗し、`db2diag` ログ・ファイルにメッセージが記録されます。

Linux または Sun Solaris オペレーティング・システムを使用している場合は、IBM® Network Authentication Service (NAS) Toolkit のインスタンスをすべてアンインストールし、NAS インストール・パス・ロケーションへの参照をシステム **PATH** 変数からすべて削除してください。

このタスクについて

DB2 データベースで Kerberos 認証を使用できるかどうかは、接続するアプリケーションから提供される資格情報を使用して、セキュリティー認証が正常に作成されたかどうかによって決まります。さらに、使用可能な場合は、Kerberos 相互認証がサポートされます。この方式では、クライアントとサーバーの両方が Kerberos を使用するためにそれぞれの ID を証明する必要があります。しかし、その他の Kerberos 機能 (署名、メッセージ暗号化など) は、使用できません。

ご使用のシステムに Kerberos 製品をインストールおよび構成する方法についてさらに詳しくは、http://www.ibm.com/developerworks/data/library/techarticle/dm-0603see/index.html を参照するか、または Kerberos 製品に付属する資料を参照してください。

DB2 データベース・システム用の Kerberos サポートは、IBMkrb5 GSS-API セキュリティー・プラグインにより実現しています。このプラグインは、サーバー認証とクライアント認証の両方に使用されます。プラグイン・ライブラリーは、DB2 のインストール時に以下の場所にインストールされます。

- UNIX および Linux の 32 ビット・オペレーティング・システムの場合:
  `sqllib/security32/plugin/IBM/client` ディレクトリーおよび
  `sqllib/security32/plugin/IBM/server` ディレクトリー
- UNIX および Linux の 64 ビット・オペレーティング・システムの場合:
  `sqllib/security64/plugin/IBM/client` ディレクトリーおよび
  `sqllib/security64/plugin/IBM/server` ディレクトリー
- Windows オペレーティング・システムの場合:
  `sqllib¥security¥plugin¥IBM¥client` ディレクトリーおよび
  `sqllib¥security¥plugin¥IBM¥server` ディレクトリー

UNIX および Linux プラグインのソース・コードである `IBMkrb5.C` は、`sqllib/samples/security/plugins` ディレクトリーにあります。64 ビット Windows オペレーティング・システムの場合、プラグイン・ライブラリーの名前は `IBMkrb564.dll` です。

Kerberos とグループ

Kerberos には、グループ化の概念がありません。このため、DB2 データベース・インスタンスは Kerberos プリンシパルのグループ・リストを取得するためにローカル・オペレーティング・システムに依存します。UNIX および Linux オペレーティング・システムに依存する場合は、プリンシパルごとに同等のシステム・アカウントが必要になります。例えば、プリンシパルが *name@REALM* である場合、DB2 データベース製品は、オペレーティング・システム・ユーザー *name* が属するすべてのグループ名についてローカル・オペレーティング・システムに対して照会することにより、グループ情報を収集します。オペレーティング・システム・ユーザー *name* が存在しない場合、AUTHID は PUBLIC グループにのみ所属します。

Windows オペレーティング・システムでは、ドメイン・アカウントが Kerberos プリンシパルに自動的に関連付けられます。別個にオペレーティング・システム・アカウントを作成する追加のステップは不要です。

Kerberos keytab ファイル

セキュリティー・コンテキスト要求を受け入れるために、UNIX または Linux オペレーティング・システム上のすべての Kerberos サービスは、資格情報を *keytab* ファイル内に格納する必要があります。この要件は、DB2 データベース・インスタンスがサーバー・プリンシパルとして使用するプリンシパルに適用されます。デフォルト keytab ファイルでのみ、サーバーの鍵が検索されます。キーを keytab ファイルに追加する方法の説明については、Kerberos 製品に付属の資料を参照してください。

Windows オペレーティング・システムには keytab ファイルの概念がなく、システムがプリンシパルの資格情報の保管および獲得を自動的に処理します。

KRB5_KTNAME 環境変数を使用して、デフォルトのキータブ・ファイル名を指定できます。ただし、サーバー・プラグインは DB2 データベース・エンジン・プロセス内で実行されるため、この環境変数にアクセスできない可能性があります。この状態を回避するには、**db2set** コマンドを使用して **KRB5_KTNAME** 環境変数を **DB2ENVLIST** レジストリー変数に追加します。

```
db2set DB2ENVLIST=KRB5_KTNAME
```

Windows では Kerberos によってキータブ・ファイルが使用されないため、このオプションは Linux または UNIX サーバーのみで使用できます。

手順

DB2 サーバー用に Kerberos をセットアップするには、以下を行います。

1. 以下のいずれかのステップを実行して、Kerberos をインストールします。
   - AIX オペレーティング・システムでは、NAS (Network Authentication Services) Toolkit for DB2 を AIX バージョン 1.4 以降にインストールします。NAS パッケージは https://www.ibm.com/services/forms/preLogin.do?source=dm-nas からダウンロードできます。
   - Linux および HP-UX (64 ビットのみ) オペレーティング・システムでは、オペレーティング・システムのインストール・メディアに含まれている Kerberos パッケージ `krb5` をインストールします。
   - Sun Solaris オペレーティング・システムの場合、Kerberos サービスが Solaris 10 リリースに含まれています。追加のインストールは不要です。
   - Windows オペレーティング・システムの場合、ドメイン・コントローラーで Active Directory を使用可能にします。
2. Kerberos プラグインを使用するように DB2 製品を構成します。 248 ページの『Kerberos プラグインのデプロイ』を参照してください。
3. DB2 サーバーを再始動します。

Kerberos の命名およびマッピング

DB2 データベース・システムで Kerberos を使用する前に、クライアント・コンピューター、サーバー・コンピューター、およびプリンシパルが同じレルムに属しているか、トラステッド・レルムに属していることを確認する必要があります。

クライアント・プリンシパル

認証のために Kerberos チケットを受け取ることのできる固有の ID を、*プリンシパル* と呼びます。Kerberos のプリンシパル ID は、2 部形式 (*name@REALM*)、または複数部分からなる形式 (*name/instance@REALM*) のどちらかで定義されます。*name* コンポーネントは許可 ID (AUTHID) マッピングで使用されるため、*name* は DB2 データベースの命名規則に準拠する必要があります。これらの規則では、名前が 128 文字に制限され、文字の選択も制限されます。

注: Windows オペレーティング・システムでは、Kerberos プリンシパル ID がドメイン・ユーザーに直接関連付けられます。つまり、ドメインまたはレルムに関連付

けられていない Windows オペレーティング・システムでは、 Kerberos 認証を使用できないということです。さらに、Windows オペレーティング・システムでは、プリンシパル ID を定義する際に、*name@domain* という 2 部構成の形式のみがサポートされます。

許可 ID マッピング

オペレーティング・システムのユーザー ID の存在範囲が通常 1 つのコンピューターに限定されるのとは異なり、Kerberos プリンシパルは自身のレルム以外のレルムでも認証可能です。プリンシパル名にレルム名を付けて完全修飾することにより、プリンシパル名が重複する問題を回避できます。 Kerberos では、完全修飾プリンシパル名は以下の形式を取ります。

name/instance@REALM

ここで、*instance* には、スラッシュ (/) で区切ることで複数のインスタンス名を含めることができます。例えば、*name/instance1/instance2@REALM* などとなります。あるいは、*instance* フィールドを省略できます。

レルム名は、同じネットワーク内に定義されたすべてのレルムの中で固有でなければなりません。許可 ID とプリンシパル名 (完全修飾されたプリンシパルの中の *name* フィールド) との間で、1 対 1 のマッピングが必要です。DB2 データベース・マネージャーでは許可 ID がデフォルト・スキーマとして使用され、簡単かつ論理的に派生する必要があるので、この単純なマッピングが必要とされます。以下のようなマッピングによって引き起こされる潜在的な問題に注意してください。

- 異なるレルムに属する同じ名前の複数のプリンシパルは、同じ許可 ID にマップされます。例えば、以下の 2 つのプリンシパル名は両方とも、gregor1x という許可 ID にマップされています。
  - gregor1x@EXAMPLE.COM
  - gregor1x@WWW.COM
- 同じ名前でインスタンスが異なる複数のプリンシパルは、同じ許可 ID にマップされます。例えば、以下の 2 つのプリンシパル名は両方とも、gregor1x という許可 ID にマップされています。
  - gregor1x/bigmachine@EXAMPLE.COM
  - gregor1x/littlemachine@EXAMPLE.COM

したがって、以下のガイドラインに従ってください。

- DB2 データベース・サーバーにアクセスするすべてのトラステッド・レルムにおける名前 1 つにつき、1 つの固有の名前空間を維持します。
- 同じ *name* フィールドを伴うプリンシパルはすべて、インスタンスにかかわらず、同じユーザーに所属させます。

サーバー・プリンシパル

UNIX および Linux オペレーティング・システムでは、DB2 データベース・インスタンスのサーバー・プリンシパル名は *インスタンス名/完全修飾ホスト名@REALM* と想定されます。このプリンシパルは、Kerberos セキュリティー・コンテキストを受け入れることができなければならず、DB2 データベース・インスタンスの始動前

に既に存在している必要があります。これは、初期化の際にサーバー名がプラグインによって DB2 データベース・インスタンスに報告されるためです。

Windows オペレーティング・システムでは、サーバー・プリンシパルは通常、 DB2 データベース・サービスの開始に使用されるドメイン・アカウントによって識別されます。この状況の例外は、LocalSystem アカウントによってインスタンスが開始された場合です。この場合、サーバー・プリンシパルの名前は、*hostname* として報告されます。この ID は、クライアントとサーバーの両方が Windows ドメインに属する場合のみ有効です。

Windows オペレーティング・システムは、2 つより多くの部分から成る名前をサポートしていません。例えば、*component/component@REALM* などです。これは、Windows クライアントが UNIX サーバーに接続しようとするときに問題を起こします。そのため、UNIX Kerberos との相互運用が必要な場合は、Kerberos プリンシパルと Windows アカウントとのマッピングを Windows ドメイン内に作成する必要があります。手順については、該当する Windows の資料を参照してください。

UNIX および Linux オペレーティング・システム上の DB2 サーバーで使用される Kerberos サーバーのプリンシパル名をオーバーライドするには、**DB2_KRB5_PRINCIPAL** 環境変数を完全修飾サーバー・プリンシパル名に設定します。サーバー・プリンシパルの置換名が DB2 データベース・システムで認識されるのは、**db2start** コマンドを発行することでインスタンスを再始動してからとなります。

Kerberos 認証の使用可能化

DB2 データベース・システムで Kerberos を使用できるようにするには、事前に Kerberos 認証を使用可能に設定しておく必要があります。

クライアントでの Kerberos 認証の使用可能化

クライアントで Kerberos 認証を使用可能にするには、**clnt_krb_plugin** データベース・マネージャー構成パラメーターを、使用する Kerberos プラグインの名前に設定します。

ローカル許可では、Kerberos がクライアントで使用されるのは、**authentication** 構成パラメーターが KERBEROS または KRB_SERVER_ENCRYPT に設定されている場合です。そうでない場合、クライアント・サイドの Kerberos サポートは想定されません。

重要: Kerberos サポートが利用可能かどうかの確認は実行されません。

DB2 サーバーへのアウトバウンド接続で Kerberos 認証を使用可能にするには、データベースをカタログする際に、認証タイプとして Kerberos を代わりに指定します。以下にその例を示します。

```
CATALOG DATABASE testdb AT NODE testnode
   AUTHENTICATION KERBEROS TARGET PRINCIPAL
   service/host@REALM
```

ただし、認証情報を設定しない場合、サーバーはサーバー・プリンシパルの名前をクライアントに送ります。

サーバーでの Kerberos 認証の使用可能化

サーバーで Kerberos 認証を使用可能にするには、サーバーの **srvcon_gssplugin_list** データベース・マネージャー構成パラメーターに指定するプラグインのリストに、特定の Kerberos プラグイン名を含めます。このリストに Kerberos プラグイン名を含めることで、クライアントが接続時にサーバーをスキャンして、Kerberos 認証方式を選択することが可能になります。

この構成パラメーターが空のままで、**authentication** 構成パラメーターを KERBEROS または KRB_SERVER_ENCRYPT に設定した場合は、デフォルトの Kerberos プラグインである IBMkrb5 が代わりに使用されます。 Kerberos プラグインは 1 つだけ指定できます。

最後に、着信接続の許可のみに Kerberos を使用するには、以下に示す 2 つのオプションのいずれかに **svrcon_auth** パラメーターを設定します。

- KERBEROS (Kerberos 認証のみを使用する場合)。または、
- KRB_SERVER_ENCRYPT (Kerberos および SERVER_ENCRYPT 許可を使用する場合)。

着信接続およびローカル許可で Kerberos を使用する場合は、**svrcon_auth** 構成パラメーターを空のままにし、**authentication** 構成パラメーターの値を Kerberos オプションの 1 つに設定します。

Kerberos プラグインの作成

DB2 データベース・システムにおける Kerberos 認証の動作をカスタマイズするため、独自の Kerberos 認証プラグインを作成することができます。

Kerberos プラグインを作成するときには、以下の点を考慮してください。

- Kerberos プラグインを GSS-API プラグインとして作成します。ただし、初期化関数内で、DB2 データベース・インスタンスに戻される関数ポインター配列の *plugintype* 変数を、DB2SEC_PLUGIN_TYPE_KERBEROS に設定します。
- 特定の条件下では、サーバーがサーバー・プリンシパル名をクライアントに報告します。 Kerberos プラグインは、プリンシパルを GSS_C_NT_USER_NAME 形式 (つまり、*server/host@REALM*) で指定する必要があります。 GSS_C_NT_HOSTBASED_SERVICE 形式 (つまり、*service@host*) はサポートされていません。

Kerberos との互換性

DB2 Kerberos 認証は、IBM System z®、IBM i、および Windows システムと互換性があります。

IBM System z および IBM i との互換性

IBM System z または IBM i システムにデータベースを接続するには、**CATALOG DATABASE** コマンドの **AUTHENTICATION** パラメーターおよび **KERBEROS TARGET PRINCIPAL** パラメーターを使用して、データベースをカタログする必要があります。

IBM System z オペレーティング・システムも IBM i オペレーティング・システムも、Kerberos の相互認証セキュリティー機能をサポートしていません。

Windows での問題

Windows オペレーティング・システム上で Kerberos を使用しているときは、次の問題点に注意する必要があります。

- Windows オペレーティング・システムによる一部のエラーの検出方式および報告方式が原因で、次の条件が存在するとクライアント・セキュリティー・プラグイン・エラーが発生します。
  - アカウントの有効期限が切れている
  - パスワードが無効である
  - パスワードの有効期限が切れている
  - パスワードが管理者によって強制的に変更された
  - アカウントが使用不可である

  さらに、すべての場合に、DB2 管理ログまたは **db2diag** ログ・ファイルに「ログオンに失敗 (Logon failed)」または「ログオン拒否 (Logon denied)」というメッセージが収められます。
- ドメイン・アカウント名がローカルで定義される場合、接続においてドメイン名とパスワードが明示的に指定されていると、次のエラーにより失敗します。`The Local Security Authority cannot be contacted`。このエラーは、Windows オペレーティング・システムがローカル・ユーザーを最初に位置決めするために生じるものです。これは、接続ストリングで完全修飾ユーザー名 (例: `name@DOMAIN.IBM.COM`) を指定することにより解決されます。
- Windows アカウントの名前にはアットマーク (@) 文字を含めることができません。この文字は DB2 Kerberos プラグインによってドメイン名区切り文字と見なされるためです。
- クライアントとサーバーの両方が Windows オペレーティング・システム上にある場合は、`LocalSystem` アカウントを使用して DB2 サービスを開始できます。ただし、クライアントとサーバーが異なるドメインにある場合には、無効なターゲット・プリンシパル名のエラーによって接続が失敗する可能性があります。このエラーを回避するには、**CATALOG DATABASE** コマンドを使用して、クライアントでターゲット・プリンシパルを明示的にカタログします。その際に、完全修飾サーバー・ホスト名と完全修飾ドメイン名を使用します。次の形式を使用します。*host/server hostname@server domain name*。例えば、host248/server34.toronto.ibm.com@TORONTO.IBM.COM などとします。`LocalSystem` アカウントを使用する代わりに、有効なドメイン・アカウントを使用する方法があります。

サーバー上でのパスワードの保守

パスワードのメンテナンス作業を実行する必要があるかもしれません。 そのような作業は通常、サーバーにおいて必要ですが、その際、サーバー環境で作業できないユーザーや快適に作業できないユーザーが多数生じるため、これらの作業の実行は簡単ではありません。 DB2 データベース・システム提供する機能を使用すれば、サーバーにいなくても、パスワードを更新および確認することができます。

次に示すリリース以降のサーバー上のデータベースに接続する場合、新規パスワードを割り当てることができます。AIX および Windows オペレーティング・システ

ム上の DB2 Universal Database™ バージョン 8、Linux オペレーティング・システム上の DB2 バージョン 9.1 フィックスパック 3 以降、DB2 for z/OS バージョン 7、DB2 for i V6R1。

例えば、エラー・メッセージ SQL1404N 「パスワードの有効期限が切れています」または SQL30082N 「セキュリティー処理は、理由 1 により失敗しました (PASSWORD EXPIRED)」を受け取った場合は、次のように 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
```

許可、特権、およびオブジェクト所有権

ユーザー (許可 ID で識別される) は、指定された関数を実行する権限を持っている場合にのみ、操作を正常に実行することができます。表を作成するには、ユーザーに表作成の許可が必要であり、表を変更するには、表変更の許可が必要となります。その他も同様です。

データベース・マネージャーでは、特定のタスクを実行するのに必要なデータベース機能を使用するために、各ユーザーが特定の許可を与えられていなければなりません。ユーザーは、自分のユーザー ID にその許可を付与してもらうか、あるいはその許可を付与されたロールまたはグループのメンバーになることにより、必要な許可を取得できます。

許可には、*管理権限*、*特権* および *LBAC 信用証明情報* の 3 つの形式があります。また、オブジェクトの所有権を持つ場合は、作成されるオブジェクトに対して一定の許可が提供されます。そうした形式の許可について、次のセクションで取り上げます。

管理権限

管理権限のある担当者はいずれも、データベース・マネージャーを制御するタスクに携わり、データの安全と整合性に対する責任を持ちます。

システム・レベルの許可

システム・レベルの権限によって、インスタンス・レベルの機能を様々な度合いで制御できます。

- SYSADM (システム管理者) 権限

  SYSADM (システム管理者) 権限は、データベース・マネージャーによって作成および保守されるすべてのリソースに対する制御を可能にします。システム管理者は、SYSCTRL、SYSMAINT、および SYSMON 権限をすべて所有します。SYSADM 権限を持つユーザーは、データベース・マネージャーの制御、およびデータの保護と整合性を担当します。

- SYSCTRL 権限

  SYSCTRL 権限は、システム・リソースに影響を与える操作に対する制御を可能にします。例えば、SYSCTRL 権限を持つユーザーは、データベースの作成、更新、開始、停止、またはドロップを行うことができます。さらに、このユーザーはインスタンスの開始または停止を行うことができま

すが、表データへのアクセスはできません。 SYSCTRL 権限を持つユーザーには、SYSMON もまた与えられます。

- SYSMAINT 権限

  SYSMAINT 権限は、インスタンスに関連したすべてのデータベースに対する保守操作を実行するのに必要な権限を与えます。SYSMAINT 権限を持つユーザーは、データベースの更新と構成、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のリストア、およびデータベースのモニターを行うことができます。 SYSCTRL と同様に、SYSMAINT は表データへのアクセス権限を与えません。 SYSMAINT 権限を持つユーザーには、SYSMON 権限もまた与えられます。

- SYSMON (システム・モニター) 権限

  SYSMON (システム・モニター) 権限は、データベース・システム・モニターの使用に必要な権限を与えます。

データベース・レベルの許可

データベース・レベルの権限により、データベース内における制御が行えます。

- DBADM (データベース管理者)

  DBADM 権限レベルは、1 つのデータベースに対する管理権限を与えます。このデータベース管理者は、オブジェクトの作成およびデータベース・コマンドの発行に必要な権限を所有します。

  DBADM 権限の付与は、SECADM 権限を持つユーザーのみが行えます。DBADM 権限は、PUBLIC には付与できません。

- SECADM (セキュリティー管理者)

  SECADM 権限レベルは、1 つのデータベースに対するセキュリティーの管理権限を与えます。セキュリティー管理者権限は、データベース・セキュリティー・オブジェクト (データベースの役割、監査ポリシー、トラステッド・コンテキスト、セキュリティー・ラベル・コンポーネント、およびセキュリティー・ラベル) を管理したり、すべてのデータベース特権と権限の付与および取り消しを行ったりすることができます。SECADM 権限を持つユーザーは、所有していないオブジェクトの所有権を移行することができます。また、このようなユーザーは、AUDIT ステートメントを使用して、サーバー側の特定のデータベースまたはデータベース・オブジェクトに監査ポリシーを関連付けることもできます。

  SECADM 権限には表に格納されたデータにアクセスする固有の特権はありません。この権限の付与が行えるのは、SECADM 権限をもつユーザーだけとなります。SECADM 権限は、PUBLIC には付与できません。

- SQLADM (SQL 管理者)

  SQLADM 権限レベルには、単一のデータベース内の SQL ステートメントをモニターおよびチューニングする管理権限があります。この権限の付与が行えるのは、ACCESSCTRL または SECADM 権限をもつユーザーです。

- WLMADM (ワークロード管理の管理者)

  WLMADM 権限には、サービス・クラス、作業アクション・セット、作業クラス・セット、およびワークロードなどのワークロード管理オブジェクトを管理する管理権限があります。この権限の付与が行えるのは、ACCESSCTRL または SECADM 権限をもつユーザーです。
- EXPLAIN (Explain 権限)

  EXPLAIN 権限レベルには、データへのアクセス権を取得することなく、照会プランを Explain する管理権限があります。この権限の付与が行えるのは、ACCESSCTRL または SECADM 権限をもつユーザーだけです。
- ACCESSCTRL (アクセス制御権限)

  ACCESSCTRL 権限レベルには、以下の GRANT (および REVOKE) ステートメントを発行する管理権限があります。
  - GRANT (データベース権限)

    ACCESSCTRL 権限によって、その保有者が、ACCESSCTRL、DATAACCESS、DBADM、または SECADM 権限を付与できるようになるわけではありません。SECADM 権限を持つユーザーだけが、これらの権限を付与できます。
  - GRANT (グローバル変数特権)
  - GRANT (索引特権)
  - GRANT (モジュール特権)
  - GRANT (パッケージ特権)
  - GRANT (ルーチン特権)
  - GRANT (スキーマ特権)
  - GRANT (シーケンス特権)
  - GRANT (サーバー特権)
  - GRANT (表、ビュー、またはニックネーム特権)
  - GRANT (表スペース特権)
  - GRANT (ワークロード特権)
  - GRANT (XSR オブジェクト特権)

  ACCESSCTRL 権限の付与は、SECADM 権限を持つユーザーのみが行えます。ACCESSCTRL 権限は、PUBLIC には付与できません。
- DATAACCESS (データ・アクセス特権)

  DATAACCESS 権限レベルには、以下の特権および権限があります。
  - LOAD 権限
  - 表、ビュー、ニックネーム、およびマテリアライズ照会表での SELECT、INSERT、UPDATE、DELETE 特権
  - パッケージに関する EXECUTE 特権
  - モジュールに関する EXECUTE 特権
  - ルーチンに関する EXECUTE 特権

監査ルーチンの例外:
AUDIT_ARCHIVE、AUDIT_LIST_LOGS、AUDIT_DELIM_EXTRACT。

- すべてのグローバル変数に対する READ 特権、およびすべてのグローバル変数に対する WRITE 特権 (読み取り専用の変数を除く)
- すべての XSR オブジェクトに対する USAGE 特権
- すべてのシーケンスに対する USAGE 特権

SECADM 権限を持つユーザーだけがこれを付与できます。
DATAACCESS 権限は、PUBLIC には付与できません。

- データベース権限 (非管理用)

表やルーチンの作成、表へのデータのロードなどのアクティビティーを実行するには、特定のデータベース権限が必要です。例えば、**ロード**・ユーティリティーを使ってデータを表にロードするには、LOAD データベース権限が必要です (その表に対する INSERT 特権も必要です)。

特権

特権とは、アクションまたはタスクを実行する許可です。許可ユーザーは、オブジェクトを作成することができ、所有しているオブジェクトにアクセス権を持ち、GRANT ステートメントを使用することによって、所有オブジェクトに対する特権を他のユーザーに渡すことができます。

特権は、個々のユーザー、グループ、または PUBLIC に付与できます。 PUBLIC は、将来のユーザーを含むすべてのユーザーで構成される特殊グループです。グループのメンバーであるユーザーは、グループがサポートされている場合は、グループに付与された特権を間接的に利用できます。

CONTROL 特権: オブジェクトに対する CONTROL 特権を持っているユーザーは、そのデータベース・オブジェクトにアクセスでき、そのオブジェクトに対する他のユーザーの特権を付与または取り消すことができます。

注: CONTROL 特権は、表、ビュー、ニックネーム、索引、およびパッケージにのみ適用されます。

他のユーザーがそのオブジェクトに対する CONTROL 特権を要求した場合、SECADM または ACCESSCTRL 権限を持つユーザーが、そのオブジェクトに対する CONTROL 特権を付与することができます。CONTROL 特権は、オブジェクト所有者から取り消されることがありませんが、オブジェクト所有者は、TRANSFER OWNERSHIP ステートメントを使用して変更される場合があります。

個別特権: ユーザーが特定オブジェクトに対して特定のタスクを実行できるようにするために、個別特権を与えることができます。管理権限 ACCESSCTRL または SECADM、あるいは CONTROL 特権を持つユーザーは、ユーザーに特権を付与したり、ユーザーから特権を取り消したりできます。

個別特権およびデータベース権限は特定の機能の実行を許可しますが、同じ特権または権限を他のユーザーに与えることはできません。 GRANT ステートメントで WITH GRANT OPTION を使用すれば、表、ビュー、スキーマ、パッケージ、ルーチン、シーケンスに関する特権を他のユーザーに対して GRANT できる権利を、他

のユーザーに拡張して与えることができます。ただし、WITH GRANT OPTION を使用する場合、特権を GRANT する人が、いったん GRANT された特権を取り消すことはできません。特権を取り消すためには、SECADM 権限、ACCESSCTRL 権限、または CONTROL 特権を持っていなければなりません。

パッケージまたはルーチン内のオブジェクトに対する特権: ユーザーにパッケージまたはルーチンを実行する特権があると、パッケージまたはルーチン内で使用されるオブジェクトに対する特定の特権が必ずしも必要とされません。パッケージまたはルーチンに静的 SQL または XQuery ステートメントが含まれる場合、パッケージの所有者の特権がそれらのステートメントに使用されます。パッケージまたはルーチンに動的 SQL または XQuery ステートメントが含まれる場合、特権の検査に使用される許可 ID は、動的照会ステートメントを発行するパッケージの **DYNAMICRULES BIND** オプションの設定と、パッケージがルーチンのコンテキストで使用される際にそれらのステートメントが発行されるかどうかによって異なります (監査ルーチンでの例外:
AUDIT_ARCHIVE、AUDIT_LIST_LOGS、AUDIT_DELIM_EXTRACT)。

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。特権をオブジェクトに関連付ける場合、そのオブジェクトはすでに存在していなければなりません。例えば、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: ユーザーまたはグループを表す許可名が権限と特権を付与され、しかもその許可名で作成されたユーザーまたはグループがいない場合には、注意が必要です。後で、その許可名を使用してユーザーまたはグループが作成され、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。

すでに付与された特権を取り消すには、REVOKE ステートメントを使用します。 1 つの許可名から特権を取り消すと、すべての許可名によって付与された特権が取り消されます。

ある許可名から特権を取り消しても、その許可名によって特権を付与された他の許可名からその同じ特権が取り消されることはありません。例えば、ユーザー CLAIRE が SELECT WITH GRANT OPTION をユーザー RICK に与えた後、RICK が SELECT を BOBBY および CHRIS に与えたとします。もし CLAIRE が SELECT 特権を RICK から取り消しても、 BOBBY と CHRIS は引き続き SELECT 特権を保持します。

LBAC 信用証明情報

セキュリティー管理者は、ラベル・ベースのアクセス制御 (LBAC) を使用して、個々の行および個々の列ごとに、どのユーザーに書き込みアクセスがあり、どのユーザーに読み取りアクセスがあるのかを厳密に決定することができます。セキュリティー管理者は、セキュリティー・ポリシーを作成して LBAC システムを構成します。セキュリティー・ポリシー では、どのデータに誰がアクセスできるかの決定で使用される基準が記述されます。任意の 1 つの表を保護するために 1 つのセキュリティー・ポリシーしか使用できませんが、複数のセキュリティー・ポリシーを使用して複数の表を保護することができます。

セキュリティー・ポリシーを作成した後、セキュリティー管理者は、そのポリシーの一部分となる、セキュリティー・ラベルおよび免除と呼ばれるデータベース・オブジェクトを作成します。セキュリティー・ラベルは一連のセキュリティー基準を表現したものとなります。免除は、作成したセキュリティー・ポリシーで保護されたデータにアクセスする場合に、これを保有するユーザーがセキュリティー・ラベルの比較について、定められた規則を免れることができるものとなります。

作成が完了すると、セキュリティー・ラベルを表の個々の列と行に関連付けてそこに保持されているデータを保護することができます。セキュリティー・ラベルにより保護されるデータは、保護データと呼ばれます。セキュリティー管理者は、ユーザーにセキュリティー・ラベルを付与することにより、保護データへのアクセスを許可します。ユーザーが保護データへのアクセスを試行すると、そのユーザーのセキュリティー・ラベルが、データを保護しているセキュリティー・ラベルと比較されます。セキュリティー・ラベルには、保護ラベルによってブロックされるものと、されないものがあります。

オブジェクトの所有権

オブジェクトが作成されるとき、1 つの許可 ID に対して、そのオブジェクトの*所有権* が割り当てられます。所有権を与えられているユーザーは、任意の適用できる SQL または XQuery ステートメントを使ってそのオブジェクトを参照することを許可されます。

スキーマ内でオブジェクトを作成するとき、ステートメントの許可 ID は、暗黙的または明示的に指定されるスキーマ内でオブジェクトを作成するのに必要な特権を持っていなければなりません。つまり、許可名がスキーマの所有者であるか、スキーマに対する CREATEIN 特権を持っている必要があります。

注: 表スペース、バッファー・プール、またはデータベース・パーティション・グループを作成するときには、この要件は適用されません。これらのオブジェクトはスキーマ内には作成されません。

オブジェクトが作成されるとき、ステートメントの許可 ID がそのオブジェクトの定義者になり、オブジェクトの作成後にデフォルトでオブジェクトの所有者になります。

注: ただし、1 つの例外があります。 CREATE SCHEMA ステートメントで AUTHORIZATION オプションを指定した場合、 CREATE SCHEMA 操作の一部として作成されるすべてのオブジェクトは、 AUTHORIZATION オプションが指定する許可 ID によって所有されます。ただし、最初の CREATE SCHEMA 操作の後でスキーマ内で作成されるすべてのオブジェクトは、特定の CREATE ステートメントに関連した許可 ID によって所有されます。

例えば、ステートメント `CREATE SCHEMA SCOTTSTUFF AUTHORIZATION SCOTT CREATE TABLE T1 (C1 INT)` によって、スキーマ `SCOTTSTUFF` および表 `SCOTTSTUFF.T1` が作成され、このどちらもユーザー SCOTT によって所有されます。ここで、ユーザー BOBBY に対して `SCOTTSTUFF` スキーマに対する CREATEIN 特権が与えられ、BOBBY が表 `SCOTTSTUFF.T1` への索引を作成するとします。索引はスキーマの後で作成されるため、 `SCOTTSTUFF.T1` への索引を所有するのは BOBBY です。

特権は、作成されるオブジェクトのタイプに応じて、以下のようにオブジェクト所有者に割り当てられます。

- CONTROL 特権は、新しく作成される表、索引、およびパッケージに対して暗黙的に付与されます。この特権を持つオブジェクト作成者は、そのデータベース・オブジェクトにアクセスでき、そのオブジェクトに対する他のユーザーの特権を付与または取り消すことができます。他のユーザーがそのオブジェクトに対する CONTROL 特権を要求した場合、 ACCESSCTRL または SECADM 権限を持つユーザーが、そのオブジェクトに対する CONTROL 特権を付与する必要があります。オブジェクト所有者は、CONTROL 特権を取り消すことができません。
- ビュー定義によって参照されるすべての表、ビュー、およびニックネームに対する CONTROL 特権をオブジェクト所有者が持っている場合、新しく作成されるビューに対して CONTROL 特権が暗黙的に付与されます。
- 他のオブジェクト (トリガー、ルーチン、シーケンス、表スペース、バッファー・プールなど) には、 CONTROL 特権が関連付けられません。ただし、オブジェクト所有者は、オブジェクトに関連付けられるそれぞれの特権を自動的に受け取ります。サポートされている場合、それらの特権には WITH GRANT OPTION が付いています。そのため、オブジェクト所有者は GRANT ステートメントを使用して他のユーザーにこれらの特権を提供できます。例えば、USER1 が表スペースを作成する場合、USER1 はこの表スペースに関する WITH GRANT OPTION 付きの USEAUTH 特権を自動的に持ち、他のユーザーに対して USEAUTH 特権を付与できます。また、オブジェクト所有者は、オブジェクトの変更、コメントの追加、およびオブジェクトのドロップを行うことができます。これらの許可はオブジェクト所有者に暗黙的に与えられ、取り消すことはできません。

表の変更など、オブジェクトに対する特定の特権は、所有者によって付与できます。また ACCESSCTRL または SECADM 権限を持つユーザーによって所有者から取り消せます。 表にコメントするなど、オブジェクトに対する特定の特権は、所有者によって付与できません。また所有者から取り消せません。TRANSFER OWNERSHIP ステートメントを使用してこれらの特権を別のユーザーに移動します。オブジェクトが作成されるとき、ステートメントの許可 ID がそのオブジェクトの定義者になり、オブジェクトの作成後にデフォルトでオブジェクトの所有者になります。 ただし、**BIND** コマンドを使用してパッケージを作成し、**OWNER** *authorization id* オプションを指定する場合、パッケージ内の静的 SQL ステートメントによって作成されるオブジェクトの所有者は、*authorization id* の値です。さらに、AUTHORIZATION 節が CREATE SCHEMA ステートメントに指定される場合、AUTHORIZATION キーワードの後に指定される許可名はスキーマの所有者です。

セキュリティー管理者またはオブジェクト所有者は、TRANSFER OWNERSHIP ステートメントを使用してデータベース・オブジェクトの所有権を変更することができます。そこで、許可 ID を修飾子として使用してオブジェクトを作成してから TRANSFER OWNERSHIP ステートメントを使用して管理者オブジェクトに持つ所有権を許可 ID に移動することで、管理者は許可 ID のためにオブジェクトを作成できます。

権限の概要

インスタンス・レベルとデータベース・レベルで多様な管理権限が存在します。これらの管理権限は、特定の特権と権限を一緒にグループ化し、データベース・インストール済み環境におけるこうしたタスクに責任を持っているユーザーに付与できるようにします。

インスタンス・レベルの権限

インスタンス・レベル権限を使用すると、データベースの作成とアップグレード、表スペースの管理、インスタンス上のアクティビティーとパフォーマンスのモニターなど、インスタンス全体に関わる機能を実行できます。インスタンス・レベルの権限は、データベース表内のデータに対するアクセスは提供しません。以下の図は、インスタンス・レベルのそれぞれの管理権限によって付与される機能について要約しています。

- SYSADM - インスタンス全体を管理するユーザー用
- SYSCTRL - データベース・マネージャー・インスタンスを管理するためのユーザー用
- SYSMAINT - インスタンス内のデータベースを保守するためのユーザー用
- SYSMON - インスタンスとそのデータベースをモニターするユーザー用

高位レベルの権限を持つユーザーは、低位レベルの権限によって付与される機能も持ちます。例えば、SYSCTRL 権限を持つユーザーは SYSMAINT 権限と SYSMON 権限を持つユーザーの機能も実行できます。

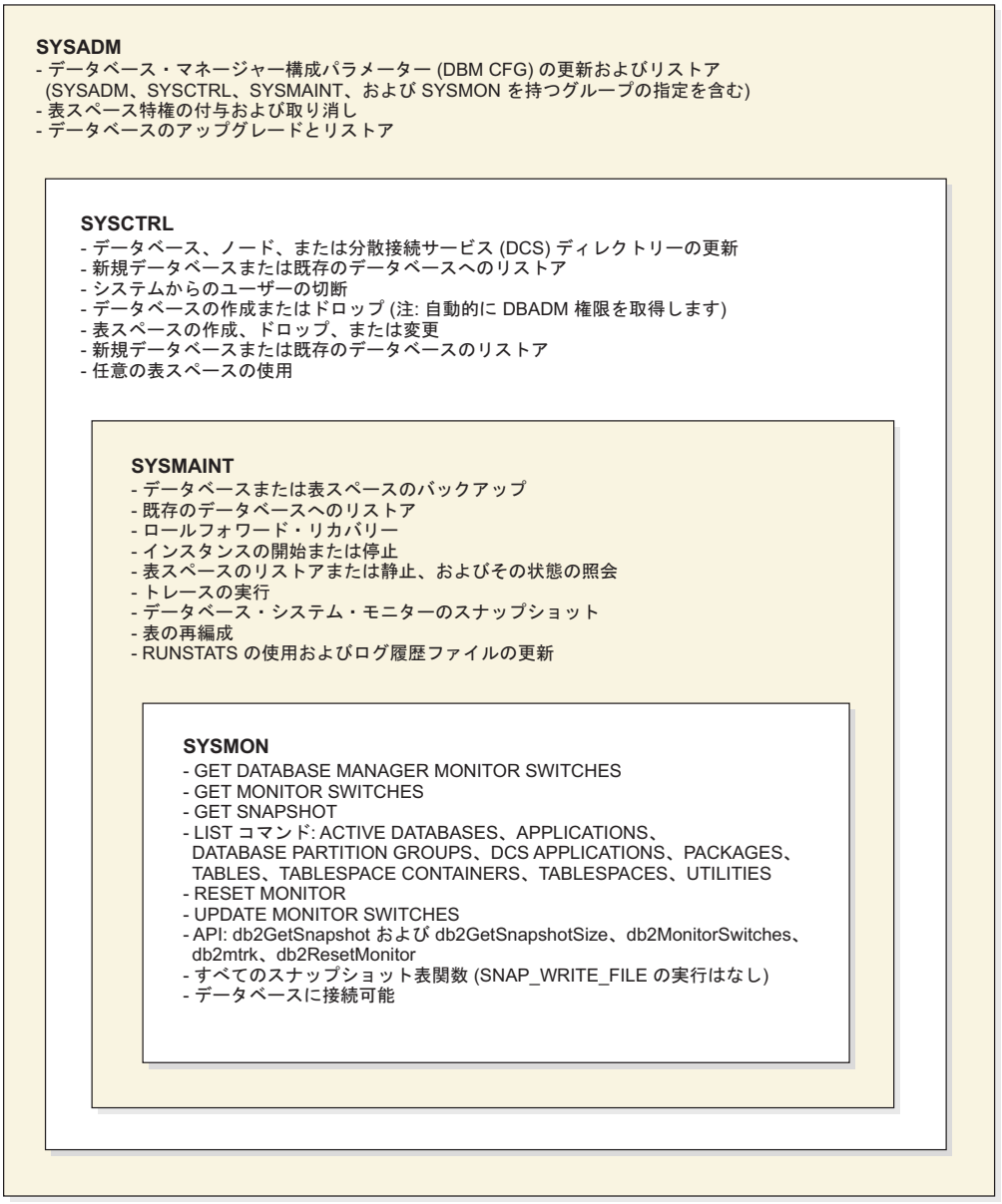


図 1. インスタンス・レベルの権限

データベース・レベルの権限

データベース・レベル権限を使用すると、特定のデータベース内の機能を実行できます。そうした機能には、特権の付与および取り消し、データの挿入、選択、削除、および更新、およびワークロードの管理などがあります。以下の図は、データベース・レベルのそれぞれの権限によって付与される機能について要約しています。管理データベース権限は、以下のとおりです。

- SECADM - データベース内のセキュリティーを管理するユーザー用
- DBADM - データベースを管理するユーザー用
- ACCESSCTRL - 権限と特権 (SECADM、DBADM、ACCESSCTRL、および DATAACCESS 権限を除く。これらの権限を付与および取り消すためには SECADM 権限が必要です) を付与および取り消すことが必要なユーザー用

- DATAACCESS - データにアクセスすることが必要なユーザー用
- SQLADM - SQL 照会をモニターおよびチューニングするユーザー用
- WLMADM - ワークロードを管理するユーザー用
- EXPLAIN - 照会プランを Explain する必要のあるユーザー用 (EXPLAIN 権限はデータ自体に対するアクセス権は付与しません)

以下の図では、低位レベルの権限で付与される機能が、高位レベルのどの権限に包含されているのかを示します (該当する場合)。例えば、DBADM 権限を持つユーザーは SQLADM 権限と EXPLAIN 権限を持つユーザーの機能、および WLMADM 権限を持つユーザーのすべての機能 (ワークロードに関する USAGE 特権の付与を除く) を実行できます。

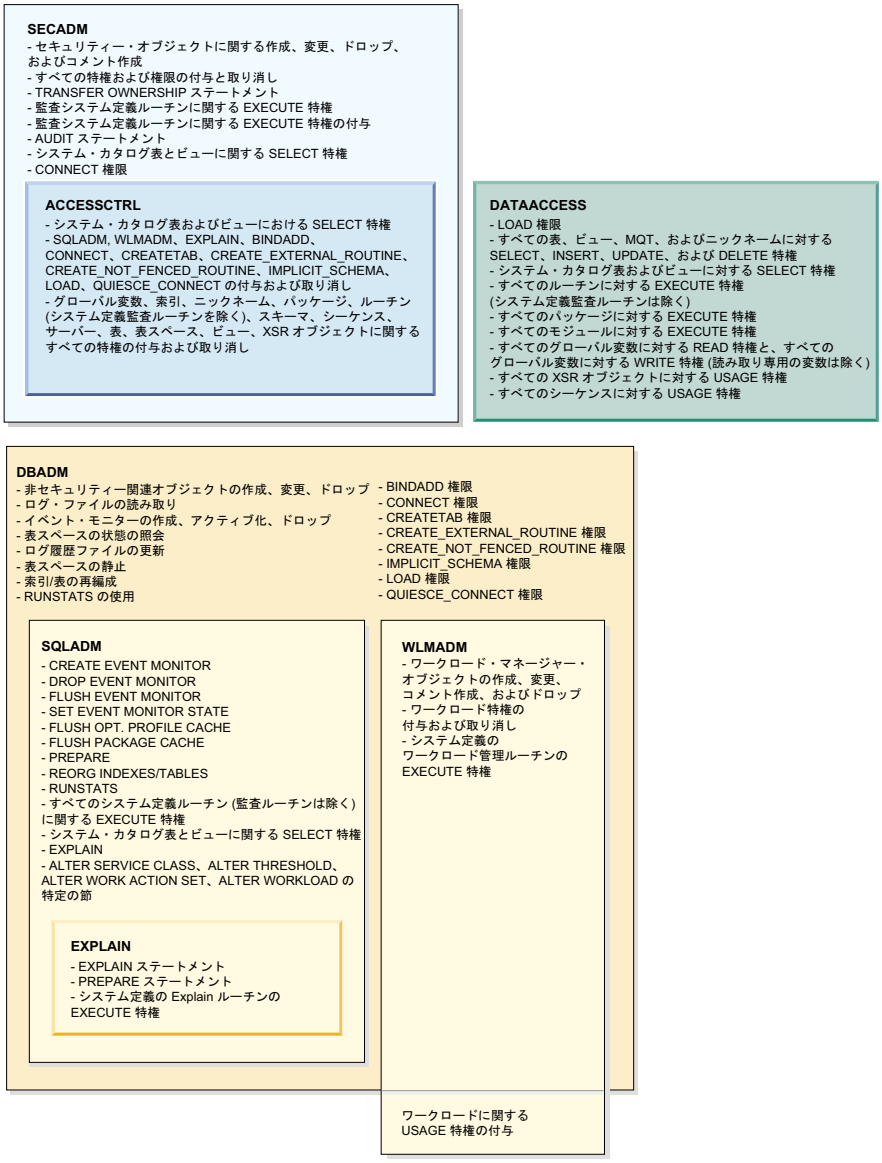


図 2. データベース・レベルの権限

インスタンス・レベルの権限

システム管理権限 (SYSADM)

SYSADM 権限レベルは、インスタンス・レベルにおける最も高いレベルの管理権限です。 SYSADM 権限を持つユーザーは、インスタンス内で一部のユーティリティーを実行し、データベース・コマンドとデータベース・マネージャー・コマンドの一部を発行できます。

SYSADM 権限は、 **sysadm_group** 構成パラメーター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セキュリティー機能によって制御されます。

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。

- データベースのアップグレード
- データベースのリストア
- データベース・マネージャー構成ファイルの変更
  (SYSADM、SYSCTRL、SYSMAINT、または SYSMON 権限のあるグループを指定することを含む)

SYSADM 権限を持つユーザーは表スペース特権を付与および取り消すことができますし、任意の表スペースを使用することもできます。

注: SYSADM 権限を持つユーザーがデータベースを作成した場合、そのユーザーには、データベースに対する ACCESSCTRL、DATAACCESS、DBADM、および SECADM 権限が自動的に付与されます。そのユーザーがそのデータベースにデータベース管理者またはセキュリティー管理者としてアクセスできないようにする場合は、そのユーザーからこれらのデータベース権限を明示的に取り消す必要があります。

バージョン 9.7 より前のリリースでは、SYSADM 権限には暗黙的な DBADM 権限が含まれ、すべての権限と特権を付与および取り消すことができました。バージョン 9.7 では、DB2 許可モデルが更新され、システム管理者、データベース管理者、およびセキュリティー管理者の権限が明確に分離されました。この機能拡張の一部として、SYSADM 権限によって付与される機能が減りました。

バージョン 9.7 では、SECADM 権限だけがすべての権限と特権を付与および取り消すことができます。

SYSADM 権限を保持するユーザーがバージョン 9.5 と同じ機能 (SECADM 権限を付与する機能以外) を得るには、セキュリティー管理者はユーザーに DBADM 権限を明示的に付与し、さらにユーザーに新しい DATAACCESS 権限と ACCESSCTRL 権限を付与する必要があります。これらの新しい権限は GRANT DBADM ON DATABASE ステートメントに WITH DATAACCESS オプションと WITH ACCESSCTRL オプション (どちらもデフォルト・オプション) を指定して使用すると付与できます。 DATAACCESS 権限は特定のデータベース内のデータに対するアクセスを許可する権限で、ACCESSCTRL 権限は特定のデータベース内の特権と非管理権限をユーザーが付与および取り消すことを許可する権限です。

Windows ローカル・システム・アカウントに関する考慮事項

Windows システムでは、**sysadm_group** データベース・マネージャー構成パラメーターが指定されないと、LocalSystem アカウントがシステム管理者と見なされます (SYSADM 権限を保持します)。 LocalSystem によって実行される DB2 アプリケーションは、バージョン 9.7 の SYSADM 権限の有効範囲の変更によって影響を受けます。通常こうしたアプリケーションは Windows サービスの形で作成され、サービス・ログオン・アカウントとしての LocalSystem アカウントで実行されます。こうしたアプリケーションで SYSADM の有効範囲内ではなくなったデータベース・アクションを実行する必要がある場合、LocalSystem アカウントに必要なデータベース特権と権限を付与する必要があります。例えば、アプリケーションでデータベース管理者機能が必要な場合、LocalSystem アカウントに、GRANT (データベース権限) ステートメントを使用して DBADM 権限を付与します。LocalSystem アカウントの許可 ID は SYSTEM であることに注意してください。

システム制御権限 (SYSCTRL)

SYSCTRL 権限は、最も高い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・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。グループが指定されると、そのグループのメンバーシップは、プラットフォーム上で使用されるセキュリティー機能によって、データベース・マネージャーの外で制御されます。

以下の操作を行えるのは、SYSCTRL 以上の権限を持つユーザーのみです。

- 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ドロップ
- 表スペースのドロップ、作成、または変更
- 任意の表スペースの使用
- 新しいデータベースまたは既存のデータベースへのリストア

さらに、SYSCTRL 権限を持つユーザーは、システム保守権限 (SYSMAINT) およびシステム・モニター権限 (SYSMON) 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
ACCESSCTRL、DATAACCESS、DBADM、および SECADM 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管理者としてアクセスすることも防止する場合は、前述の 4 つの管理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・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・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。グループが指定されると、そのグループのメンバーシップは、プラットフォーム上で使用されるセキュリティー機能によって、データベース・マネージャーの外で制御されます。

以下の操作を行えるのは、SYSMAINT 以上のシステム権限を持つユーザーのみです。

- 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リストア
- ロールフォワード・リカバリーの実行
- インスタンスの開始または停止
- 表スペースのリストア
- **db2trc** コマンドによるトレースの実行
- 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 権限を持つユーザーは、以下の操作を行うことができます。

- 表スペースの状態の照会
- ログ履歴ファイルの更新
- 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

さらに、SYSMAINT 権限を持つユーザーは、データベースに接続する特権を暗黙的に与えられ、システム・モニター権限 (SYSMON) を持つユーザーに許可された機能を実行することもできます。

システム・モニター権限 (SYSMON)

SYSMON 権限は、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を対象とするデータベース・システム・モニター・スナップショットの取得を許可します。

SYSMON 権限は、構成パラメーター **sysmon_group** によって指定されたグループに割り当てられます。グループが指定されると、そのグループのメンバーシップは、プラットフォーム上で使用されるセキュリティー機能によって、データベース・マネージャーの外で制御されます。

SYSMON 権限を持つユーザーは、以下のコマンドを実行できます。

- **GET DATABASE MANAGER MONITOR SWITCHES**
- **GET MONITOR SWITCHES**
- **GET SNAPSHOT**
- LIST (一部のコマンド):
  - **LIST ACTIVE DATABASES**
  - **LIST APPLICATIONS**
  - **LIST DATABASE PARTITION GROUPS**
  - **LIST DCS APPLICATIONS**
  - **LIST PACKAGES**
  - **LIST TABLES**
  - **LIST TABLESPACE CONTAINERS**
  - **LIST TABLESPACES**
  - **LIST UTILITIES**
- **RESET MONITOR**
- **UPDATE MONITOR SWITCHES**

SYSMON 権限を持つユーザーは、以下の API を使用できます。

- db2GetSnapshot - スナップショットの取得
- db2GetSnapshotSize - db2GetSnapshot() 出力バッファーに必要なサイズの見積もり
- db2MonitorSwitches - モニター・スイッチの入手/更新
- db2mtrk - メモリー・トラッカー
- db2ResetMonitor - モニターのリセット

SYSMON 権限を持つユーザーは、以下の SQL 表関数を使用できます。

- すべての表スナップショット関数 (あらかじめ SYSPROC.SNAP_WRITE_FILE を実行する必要はありません)

  SYSPROC.SNAP_WRITE_FILE はスナップショットを取得して、その内容をファイルに保管します。ヌルの入力パラメーターを使って表スナップショット関数を呼び出した場合、リアルタイムのシステム・スナップショットの代わりに、ファイルの内容が戻されます。

データベース権限

個々のデータベース権限により、その権限を保持する許可 ID が、特定のタイプの処置をデータベース全体に対して実行できるようになります。データベース権限は特権とは違います。特権の場合は、表や索引などの特定のデータベース・オブジェクトに対して特定の処置を取ることができます。

以下にデータベース権限を示します。

ACCESSCTRL
保有者は、監査ルーチンにおける特権、および

ACCESSCTRL、DATAACCESS、DBADM、および SECADM 権限を除き、すべてのオブジェクト特権とデータベース権限を付与および取り消すことができます。

BINDADD
保有者は、データベース内に新しいパッケージを作成できます。

CONNECT
保有者は、データベースに接続できます。

CREATETAB
保有者は、データベース内に新しい表を作成できます。

CREATE_EXTERNAL_ROUTINE
保有者は、アプリケーションによって、またデータベースの他のユーザーによって使用されるプロシージャーを作成できます。

CREATE_NOT_FENCED_ROUTINE
保有者は、*not fenced* 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。CREATE_EXTERNAL_ROUTINE は、CREATE_NOT_FENCED_ROUTINE を持つすべてのユーザーに対して自動的に付与されます。

重要: データベース・マネージャーはそのストレージや制御ブロックを、not fenced の UDF またはプロシージャーから保護しません。したがって、この権限を持つユーザーは、UDF を not fenced として登録する前に、十分にテストするよう特に注意しなければなりません。

DATAACCESS
保有者は、データベース表に保管されているデータにアクセスできます。

DBADM
保有者は、データベース管理者として振る舞えます。特に、ACCESSCTRL、DATAACCESS、および SECADM 以外の他のデータベース権限をすべて保有者に付与します。

EXPLAIN
保有者は照会プランを Explain することができます。その際、それらの照会プランが参照する表内のデータにアクセスする特権を保持している必要はありません。

IMPLICIT_SCHEMA
どのユーザーも、まだ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 保有者は表にデータをロードできます。

QUIESCE_CONNECT
保有者は静止中のデータベースにアクセスできます。

SECADM
保有者は、データベースのセキュリティー管理者として振る舞えます。

SQLADM
保有者は、SQL ステートメントをモニターおよびチューニングできます。

WLMADM
保有者は、ワークロード管理者として振る舞えます。特に、WLMADM 権限の保有者はワークロード・マネージャー・オブジェクトの作成およびドロップ、ワークロード・マネージャー特権の付与および取り消し、さらにワークロード・マネージャー・ルーチンの実行が可能です。

SECADM 権限を持つ許可 ID のみ、ACCESSCTRL、DATAACCESS、DBADM および SECADM 権限を付与できます。他のすべての権限は、ACCESSCTRL または SECADM 権限を保持する許可 ID によって付与できます。

PUBLIC からデータベース権限を除去するには、ACCESSCTRL または SECADM 権限を持つ許可 ID によって明示的に取り消さなければなりません。

セキュリティー管理者権限 (SECADM)

SECADM 権限は、特定のデータベースに対するセキュリティー管理権限です。この権限を使用すると、セキュリティー関連のデータベース・オブジェクトを作成および管理できますし、すべてのデータベース権限と特権を付与および取り消すことができます。さらに、セキュリティー管理者は監査システム・ルーチンを実行できますし、これを実行可能な他のユーザーを管理することもできます。

SECADM 権限は、カタログ表とカタログ・ビューからの SELECT はできますが、ユーザー表に格納されているデータにアクセスすることはできません。

SECADM 権限を付与できるのは、セキュリティー管理者 (SECADM 権限を保持する) だけで、この権限はユーザー、グループ、または役割に対して付与できます。PUBLIC は直接的にも間接的にも SECADM 権限を取得することはできません。

データベースには、SECADM 権限を持つタイプ USER の許可 ID が少なくとも 1 つなければなりません。SECADM 権限は、すべてのタイプ USER の許可 ID から取り消すことはできません。

SECADM 権限は、ユーザーに以下の操作を実行する能力を付与します。

- 以下のものの作成、変更、コメント作成、およびドロップ
  - 監査ポリシー
  - セキュリティー・ラベル・コンポーネント
  - セキュリティー・ポリシー
  - トラステッド・コンテキスト
- 以下のものの作成、コメント作成、およびドロップ
  - ロール
  - セキュリティー・ラベル
- データベース特権および権限の付与および取り消し
- 指定のタスクを実行するための以下の監査ルーチンの実行
  - SYSPROC.AUDIT_ARCHIVE ストアード・プロシージャーと表関数は、監査ログをアーカイブします。

– SYSPROC.AUDIT_LIST_LOGS 表関数を使用すると、関心のあるログの場所を特定できます。
– SYSPROC.AUDIT_DELIM_EXTRACT ストアード・プロシージャーは、分析を行うためにデータを区切り文字で区切られているファイルに取り出します。

またセキュリティー管理者はこうしたルーチンの EXECUTE 特権を付与および取り消すことができるので、必要に応じてこれらのタスクを委任できます。セキュリティー管理者だけが、これらのルーチンの EXECUTE 特権を付与できます。WITH GRANT OPTION 付きの EXECUTE 特権をこれらのルーチンに関して付与することはできません (SQLSTATE 42501)。

- サーバー側の特定のデータベースまたはデータベース・オブジェクトに監査ポリシーを関連付けるための AUDIT ステートメントの使用
- TRANSFER OWNERSHIP ステートメントを使用して、ステートメントの許可 ID の所有ではないオブジェクトを転送すること

他の権限は、これらの能力を付与できません。

セキュリティー管理者だけが、ACCESSCTRL、DATAACCESS、DBADM、および SECADM 権限を他のユーザー、グループ、または役割に付与できます。

バージョン 9.7 では、DB2 許可モデルが更新され、システム管理者、データベース管理者、およびセキュリティー管理者の権限が明確に分離されました。この機能拡張の一部として、SECADM 権限によって付与される機能が増えました。バージョン 9.7 より前のリリースでは、SECADM 権限によりすべての特権と権限を付与および取り消すことはできませんでした。また、SECADM 権限を付与できたのはユーザーに対してのみで、役割やグループに対してはできませんでした。さらに、SECADM 権限によって、監査組み込みプロシージャーと表関数の EXECUTE 特権を他のユーザーに付与することもできませんでした。

データベース管理権限 (DBADM)

DBADM 権限は、特定のデータベースに対する管理権限です。データベース管理者は、オブジェクトの作成およびデータベース・コマンドの発行に必要な権限を所有します。また、DBADM 権限を持つユーザーはシステム・カタログ表とビューにおける SELECT 特権を持ち、監査ルーチンを除くすべての組み込み DB2 ルーチンを実行できます。

DBADM 権限を付与または取り消すことができるのは、セキュリティー管理者 (SECADM 権限を保持する) だけで、この権限はユーザー、グループ、または役割に対して付与できます。PUBLIC は直接にも間接的にも DBADM 権限を取得することはできません。

データベースに対して DBADM 権限を保持すると、ユーザーはそのデータベースで以下のアクションを実行することができます。

- 非セキュリティー関連のデータベース・オブジェクトの作成、変更、およびドロップ
- 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、およびドロップ
- 表スペースの状態の照会

- ログ履歴ファイルの更新
- 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

SQLADM 権限と WLMADM 権限は、DBADM 権限のサブセットです。WLMADM 権限には、ワークロードにおける USAGE 特権を付与する付加的な能力もあります。

DBADM 権限とともに DATAACCESS 権限を付与する

セキュリティー管理者は、データベース管理者がデータベース内のデータにアクセスできるかどうかを指定できます。DATAACCESS 権限は、特定のデータベース内のデータに対するアクセスを許可する権限です。セキュリティー管理者は GRANT DBADM ON DATABASE ステートメントの WITH DATAACCESS オプションを使用すると、この権限をデータベース管理者に付与できます。WITH DATAACCCESS または WITHOUT DATAACCCESS のどちらのオプションも指定しないと、デフォルトで DATAACCESS 権限が付与されます。

DATAACCESS 権限なしでデータベース管理者権限を付与するには、SQL ステートメントで GRANT DBADM WITHOUT DATAACCESS を使用します。

DBADM 権限とともに ACCESSCTRL 権限を付与する

セキュリティー管理者は、データベース管理者がデータベース内の特権の付与および取り消しを行えるかどうかを指定できます。ACCESSCTRL 権限は、ユーザーが特定のデータベース内の特権および非管理権限を付与および取り消すことができるようにする権限です。セキュリティー管理者は GRANT DBADM ON DATABASE ステートメントの WITH ACCESSCTRL オプションを使用すると、この権限をデータベース管理者に付与できます。WITH ACCCESSCTRL または WITHOUT ACCCESSCTRL のどちらのオプションも指定しないと、デフォルトで ACCESSCTRL 権限が付与されます。

ACCESSCTRL 権限なしでデータベース管理者権限を付与するには、SQL ステートメントで GRANT DBADM WITHOUT ACCESSCTRL を使用します。

DBADM 権限の取り消し

セキュリティー管理者が DATAACCESS 権限または ACCESSCTRL 権限が含まれる DBADM 権限を付与した場合、こうした権限を取り消すには、DATAACCESS 権限または ACCESSCTRL 権限をセキュリティー管理者は明示的に取り消す必要があります。例えば、以下のようにセキュリティー管理者がユーザーに DBADM 権限を付与するとします。

```
GRANT DBADM ON DATABASE TO user1
```

デフォルトで、DATAACCESS 権限と ACCESSCTRL 権限も user1 に付与されます。

後ほど、以下のようにしてセキュリティー管理者は user1 から DBADM 権限を取り消します。

```
REVOKE DBADM ON DATABASE FROM user1
```

これで、user1 は DBADM 権限を保持しなくなりましたが、DATAACCESS 権限と ACCESSCTRL 権限は依然として保持しています。

これらの残留している権限を取り消すには、セキュリティー管理者は以下のようにして明示的に取り消す必要があります。

```
REVOKE ACCESSCTRL, DATAACCESS ON DATABASE FROM user1
```

従来のリリースの DBADM 権限との違い

バージョン 9.7 では、DB2 許可モデルが更新され、システム管理者、データベース管理者、およびセキュリティー管理者の権限が明確に分離されました。この機能拡張の一部として、DBADM 権限によって付与される機能が変更されました。バージョン 9.7 より前のリリースでは、DBADM 権限にはデータにアクセスする機能と、データベースに特権を付与および取り消す機能が自動的に含まれていました。バージョン 9.7 では、既に説明したように、これらの機能は新しい権限 DATAACCESS および ACCESSCTRL によって付与されます。

またバージョン 9.7 より前のリリースでは、DBADM 権限を付与すると以下の権限も自動的に付与されていました。

- BINDADD
- CONNECT
- CREATETAB
- CREATE_EXTERNAL_ROUTINE
- CREATE_NOT_FENCED_ROUTINE
- IMPLICIT_SCHEMA
- QUIESCE_CONNECT
- LOAD

バージョン 9.7 より前では、DBADM 権限を取り消しても、これらの権限は取り消されませんでした。

バージョン 9.7 では、これらの権限は DBADM 権限の一部になりました。バージョン 9.7 で DBADM 権限を取り消すと、これらの権限も失われます。

ただし、バージョン 9.7 にアップグレードした際にユーザーが DBADM 権限を保持していた場合、DBADM 権限を取り消してもこれらの権限は失われません。バージョン 9.7 で DBADM 権限を取り消すとユーザーがこれらの権限を失うことになるのは、バージョン 9.7 で付与された DBADM 権限を保持してこうした権限を獲得した場合のみです。

アクセス制御管理者権限 (ACCESSCTRL)

ACCESSCTRL 権限は、特定のデータベース内のオブジェクトに関する特権を付与および取り消すために必要な権限です。ACCESSCTRL 権限には、表 (カタログ表とビューを除く) に格納されているデータにアクセスするための固有の特権はありません。

ACCESSCTRL 権限を付与できるのは (SECADM 権限を持つ) セキュリティー管理者だけです。ユーザー、グループ、または役割に対して付与できます。PUBLIC は直接にも間接的にも ACCESSCTRL 権限を取得することはできません。
ACCESSCTRL 権限は、ユーザーに以下の操作を実行する能力を付与します。

- 次の管理権限の付与および取り消し
  - EXPLAIN
  - SQLADM
  - WLMADM
- 次のデータベース権限の付与および取り消し
  - BINDADD
  - CONNECT
  - CREATETAB
  - CREATE_EXTERNAL_ROUTINE
  - CREATE_NOT_FENCED_ROUTINE
  - IMPLICIT_SCHEMA
  - LOAD
  - QUIESCE_CONNECT
- 次のオブジェクトに関するすべての特権の付与と、その取り消し (誰が特権を付与したかは無関係)
  - グローバル変数
  - 索引
  - ニックネーム
  - パッケージ
  - ルーチン (監査ルーチンを除く)
  - スキーマ
  - シーケンス
  - サーバー
  - 表
  - 表スペース
  - ビュー
  - XSR オブジェクト
- システム・カタログ表およびビューに対する SELECT 特権。

この権限は、セキュリティー管理者 (SECADM) 権限のサブセットです。

データ・アクセス管理者権限 (DATAACCESS)

DATAACCESS は、特定のデータベース内のデータに対するアクセスを許可する権限です。

DATAACCESS 権限を付与できるのは (SECADM 権限を持つ) セキュリティー管理者だけです。ユーザー、グループ、または役割に対して付与できます。PUBLIC は直接にも間接的にも DATAACCESS 権限を取得することはできません。

すべての表、ビュー、マテリアライズ照会表、およびニックネームに関して、以下の権限および特権を付与します。

- データベースにおける LOAD 権限
- SELECT 特権 (システム・カタログ表およびビューを含む)
- INSERT 特権
- UPDATE 特権
- DELETE 特権

さらに、DATAACCESS 権限は以下の特権を付与します。

- すべてのパッケージに関する EXECUTE
- すべてのルーチンに関する EXECUTE (監査ルーチンを除く)
- すべてのモジュールに対する EXECUTE
- すべてのグローバル変数に対する READ、およびすべてのグローバル変数に対する WRITE (読み取り専用の変数を除く)
- すべての XSR オブジェクトに対する USAGE
- すべてのシーケンスに対する USAGE

SQL 管理者権限 (SQLADM)

SQLADM 権限は、SQL ステートメントをモニターおよびチューニングするために必要な権限です。

SQLADM 権限を付与できるのはセキュリティー管理者 (SECADM 権限を保持) か、または ACCESSCTRL 権限を持つユーザーです。SQLADM 権限は、ユーザー、グループ、ロール、または PUBLIC に対して付与できます。SQLADM 権限により、以下の機能の実行権限がユーザーに付与されます。

- 以下の SQL ステートメントの実行。
  - CREATE EVENT MONITOR
  - DROP EVENT MONITOR
  - EXPLAIN
  - FLUSH EVENT MONITOR
  - FLUSH OPTIMIZATION PROFILE CACHE
  - FLUSH PACKAGE CACHE
  - PREPARE
  - REORG INDEXES/TABLE
  - RUNSTATS
  - SET EVENT MONITOR STATE

  注: DB2AUTH レジストリー変数を SQLADM_NO_RUNSTATS_REORG に設定すると、SQLADM 権限のあるユーザーは REORG 操作も RUNSTATS 操作も実行できなくなります。
- 以下のワークロード・マネージャー SQL ステートメントの特定の節の実行。
  - ALTER SERVICE CLASS ステートメントの以下の節。
    - COLLECT AGGREGATE ACTIVITY DATA

- COLLECT AGGREGATE REQUEST DATA
- COLLECT REQUEST METRICS

– ALTER THRESHOLD ステートメントの以下の節。
- WHEN EXCEEDED COLLECT ACTIVITY DATA

– 作業アクションの変更を許可する ALTER WORK ACTION SET ステートメントの以下の節。
- ALTER WORK ACTION ... COLLECT ACTIVITY DATA
- ALTER WORK ACTION ... COLLECT AGGREGATE ACTIVITY DATA
- ALTER WORK ACTION ... WHEN EXCEEDED COLLECT ACTIVITY DATA

– ALTER WORKLOAD ステートメントの以下の節。
- COLLECT ACTIVITY METRICS
- COLLECT AGGREGATE ACTIVITY DATA
- COLLECT LOCK TIMEOUT DATA
- COLLECT LOCK WAIT DATA
- COLLECT UNIT OF WORK DATA

- システム・カタログ表およびビューに対する SELECT 特権。
- すべての組み込み DB2 ルーチン (監査ルーチンを除く) の EXECUTE 特権

SQLADM 権限は、データベース管理者 (DBADM) 権限のサブセットです。

EXPLAIN 権限は、SQLADM 権限のサブセットです。

ワークロード管理者権限 (WLMADM)

WLMADM 権限は、特定のデータベースのワークロード・オブジェクトを管理するために必要な権限です。この権限により、ワークロード・マネージャー・オブジェクトを作成、変更、ドロップ、コメント作成したり、このオブジェクトへのアクセスを付与および取り消したりすることができます。

WLMADM 権限を付与できるのはセキュリティー管理者 (SECADM 権限を保持) か、または ACCESSCTRL 権限を持つユーザーです。WLMADM 権限は、ユーザー、グループ、ロール、または PUBLIC に対して付与できます。WLMADM 権限により、以下の操作の実行権限がユーザーに付与されます。

- 以下のワークロード・マネージャー・オブジェクトの作成、変更、コメント作成、およびドロップ。
  – ヒストグラム・テンプレート
  – サービス・クラス
  – しきい値
  – 作業アクション・セット
  – 作業クラス・セット
  – ワークロード
- ワークロード特権の付与および取り消し
- 組み込みのワークロード管理ルーチンの実行。

WLMADM 権限は、データベース管理者権限 DBADM のサブセットです。

Explain 管理者権限 (EXPLAIN)

EXPLAIN 権限は、特定のデータベースのデータにアクセスしないで、照会プランを Explain するために必要な権限です。この権限はデータベース管理者権限のサブセットで、表に格納されたデータにアクセスする固有の特権はありません。

EXPLAIN 権限を付与できるのはセキュリティー管理者 (SECADM 権限を保持) か、または ACCESSCTRL 権限を持つユーザーです。Explain 権限は、ユーザー、グループ、ロール、または PUBLIC に対して付与できます。この権限により、以下の SQL ステートメントの実行権限が付与されます。

- EXPLAIN
- PREPARE
- SELECT ステートメントの出力における、または XQuery ステートメントの出力における DESCRIBE

また EXPLAIN 権限は、組み込みの Explain ルーチンの EXECUTE 特権も付与します。

EXPLAIN 権限は、SQLADM 権限のサブセットです。

LOAD 権限

データベース・レベルの LOAD 権限、および表に対する INSERT 特権を持っているユーザーは、 **LOAD** コマンドを使用してデータを表にロードすることができます。

注: DATAACCESS 権限を保有していると、LOAD コマンドへの全アクセス権限がユーザーに付与されます。

データベース・レベルの LOAD 権限、および表に対する INSERT 特権を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。

データベース・レベルの LOAD 権限、および表に対する INSERT 特権と DELETE 特権を持っているユーザーは、 **LOAD REPLACE** コマンドを使用できます。

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 特権が付与されていないと、 **LOAD RESTART** または **LOAD TERMINATE** を行うことができません。

ロード操作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。

暗黙スキーマ権限 (IMPLICIT_SCHEMA) に関する考慮事項

CREATE DATABASE コマンドに **RESTRICTIVE** オプションが指定されている場合を除き、新しいデータベースが作成されるときには PUBLIC に IMPLICIT_SCHEMA データベース権限が与えられます。

IMPLICIT_SCHEMA 権限を使用して、ユーザーはオブジェクトを作成し、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。制限のあるデータベースである場合、このスキーマに対する CREATEIN 特権は PUBLIC にはありません。暗黙的にスキーマを作成したユーザーには、そのスキーマに対する CREATEIN 特権があります。

スキーマ・オブジェクトを暗黙的に作成できる人を制御することがデータベースで必要な場合は、RESTRICTIVE オプションを指定してデータベースを作成する必要があります。制限のないデータベースである場合、PUBLIC から IMPLICIT_SCHEMA データベース権限を取り消す必要があります。このシナリオでは、スキーマ・オブジェクトを作成する方法は以下の 3 とおりのみになります。

- 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、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、オプションで、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 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。

特権

許可 ID 特権: SETSESSIONUSER

許可 ID 特権は、許可 ID に対するアクションに関する特権です。現在、このような特権として唯一あるのが、SETSESSIONUSER 特権です。

SETSESSIONUSER 特権はユーザーまたはグループに付与でき、この特権の所有者は、特権が付与されているどの許可 ID にでも、ID を切り替えることができます。ID の切り替えは、SQL ステートメント SET SESSION AUTHORIZATION を使用して行われます。SETSESSIONUSER 特権の付与は、SECADM 権限を持つユーザーのみが行えます。

注: バージョン 8 のデータベースをバージョン 9.1 以降にアップグレードすると、そのデータベースに対して明示的 DBADM 権限を持つ許可 ID には、PUBLIC に対する SETSESSIONUSER 特権が自動的に付与されます。こうすることで、セッション許可 ID を任意の許可 ID に設定できる DBADM 権限を持つ許可 ID に依存するアプリケーションが中断することがないようにしています。これは、許可 ID が SYSADM 権限を持っていても DBADM が明示的に付与されていないときは、行われません。

スキーマ特権

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、48 ページの図 3 に示されています。

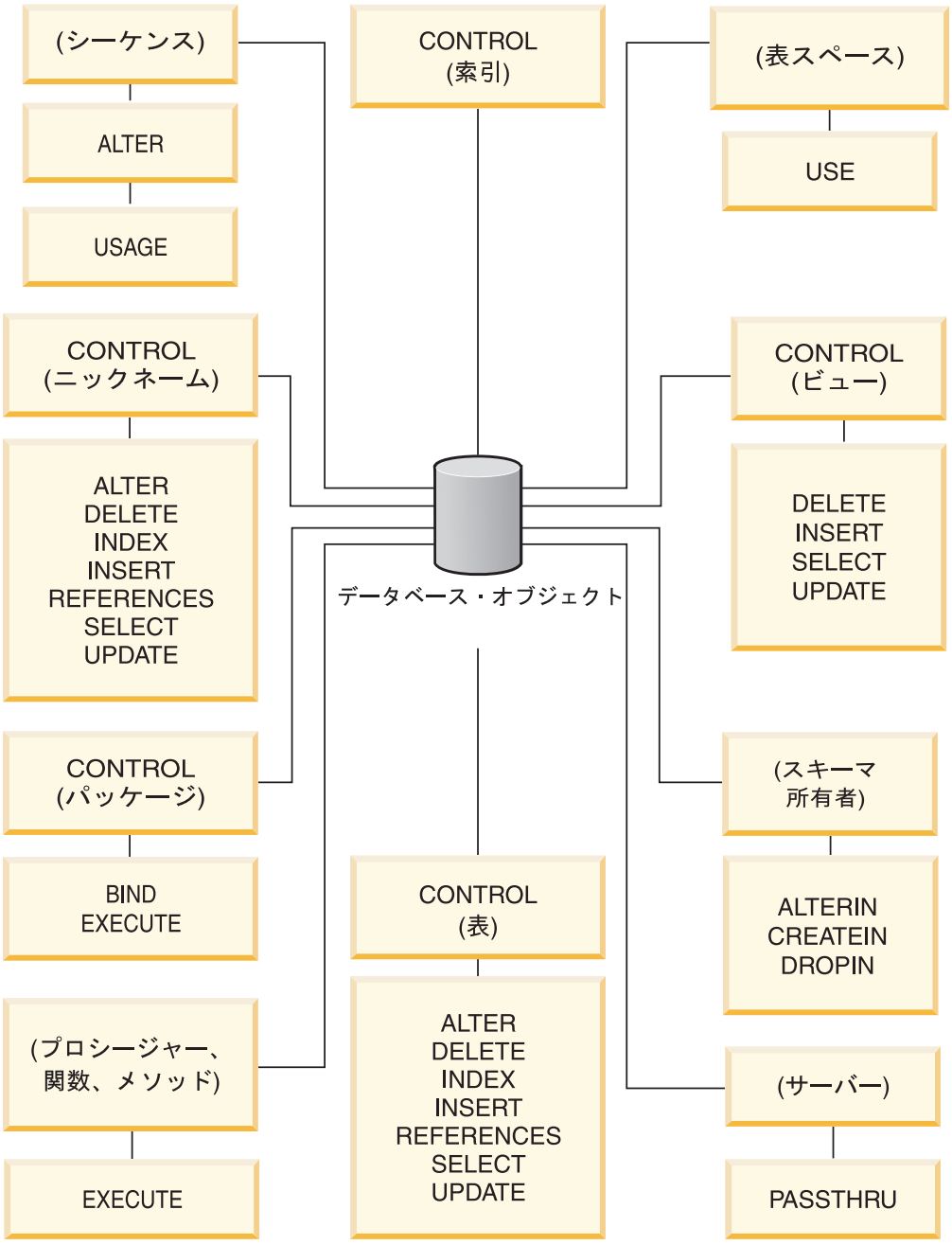


図 3. オブジェクト特権

スキーマ特権には、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、グループ、ロール、または PUBLIC 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ドロップできます。

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、ビュー、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。

表スペースの作成時に、その所有者は、その表スペースに対する WITH GRANT OPTION 付きの USE 特権を付与されます。また、SECADM 権限か ACCESSCTRL 権限を持つユーザーは表スペースの USE 特権を付与することもできます。

SYSADM 権限か SYSCTRL 権限を持つユーザーは、任意の表スペースを使用できます。

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 SYSTEM TEMPORARY 表スペースでは使用できません。

表およびビューの特権

表およびビューの特権には、データベース内の表やビューに対するアクションが含まれます。

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 権限が必要です。

- CONTROL 特権は、表やビューのドロップ、表についての個別の特権の GRANT や REVOKE を含む、表やビューに対する全ての特権を許可します。 CONTROL 特権を付与するには、ACCESSCTRL または SECADM 権限が必要です。表の作成者は、自動的にその表に対する CONTROL 特権を受け取ります。ビューの作成者は、ビュー定義の中で参照されているすべての表、ビュー、およびニックネームに対する CONTROL 特権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 は、ユーザーが表を変更することを許可します (例えば、表への列またはユニーク制約の追加)。 ALTER 特権を持つユーザーは、表または表の列に対する COMMENT ON も可能です。表に対する実行可能な変更操作については、ALTER TABLE および COMMENT ステートメントの説明を参照してください。
- DELETE 特権は、表やビューからの行の削除をユーザーに許可します。
- INDEX 特権は、表についての索引の作成をユーザーに許可します。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。
- INSERT 特権は、表やビューに対する行の挿入や、 **IMPORT** ユーティリティーの実行をユーザーに許可します。
- REFERENCES 特権は、表に対するリレーションシップの親としての指定や、外部キーの作成および削除をユーザーに許可します。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 特権は、表やビューからの行の取り出しや、表に基づくビューの作成、**EXPORT** ユーティリティーの実行をユーザーに許可します。
- UPDATE 特権は、表またはビューの項目の変更、あるいは表またはビューの 1 つ以上の特定の列の中の項目の変更をユーザーに許可します。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 GRANT する特権を GRANT することもできます。

注: ユーザーまたはグループが、ある表に対する CONTROL 特権を GRANT 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 GRANT 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
```

型付き表に関しては、表およびビューの特権に関連したインプリメンテーションがあります。

注: 特権は、表階層の各レベルで別々に付与されます。その結果、型付き表の階層内のスーパー表で特権を付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/副表のリレーションシップは、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 (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。例えば、タイプ Manager_t の副表 Manager を持つ、タイプ Employee_t の Employee 表を作成したとします。構造化タイプ Employee_t と Manager_t 間のタイプ/サブタイプのリレーションシップ、また表 Employee と Manager 間の対応する表/副表のリレーションシップ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。このリレーションシップの結果、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 ID 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 表で SELECT 特権を持つユーザーは、 Manager 表で明示的な 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、Employee 表で 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、通常の従業員とマネージャー両方に影響を与えるような、Manager に対する UPDATE 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 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。

パッケージ特権

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っています。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うことができます。

以下のいずれかの特権を使用するユーザーには、データベースに対する CONNECT 権限が必要です。

- CONTROL 特権は、パッケージの再バインド、ドロップ、または実行、およびこれらの特権を他のユーザーに与えることをユーザーに許可します。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 特権が付与されます。さらに、そのユーザーは GRANT ステートメントを使って他のユーザーにこれらの特権を付与することもできます。 (WITH GRANT OPTION を使用して特権を GRANT すれば、 BIND または EXECUTE 特権を受け取るユーザーは、その特権をさらに他のユーザーに与えることができます。) CONTROL 特権を付与するためには、ACCESSCTRL または SECADM 権限が必要です。
- パッケージの BIND 特権は、そのパッケージの再バインドやバインド、また、同じパッケージ名と作成者の新規のバージョンの追加をユーザーに許可します。
- EXECUTE 特権は、パッケージの実行をユーザーに許可します。

注: すべてのパッケージ特権は、パッケージ名と作成者が同じすべてのバージョンに適用されます。

これらのパッケージ特権に加えて、BINDADD データベース権限によって、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。

ニックネームによって参照されるオブジェクトは、オブジェクトを格納するデータ・ソースでの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・オブジェクトに対する適切な特権、または権限レベルが必要です。

DB2 データベースは DB2 ファミリーのデータ・ソースと通信するときに動的照会を使用するため、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。

索引特権

索引または索引の仕様の作成者には、索引に対する 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、索引を削除することを可能にします。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 ACCESSCTRL または SECADM 権限が必要です。

表レベルの INDEX 特権は、表に関する索引の作成をユーザーに許可します。

ニックネーム・レベルの INDEX 特権は、そのニックネームに対する索引の仕様の作成をユーザーに許可します。

シーケンス特権

シーケンスの作成者には、そのシーケンスに対する USAGE および ALTER 特権が自動的に与えられます。 USAGE 特権は、シーケンスに対して NEXT VALUE および PREVIOUS VALUE 式を使用するために必要とされます。

他のユーザーに NEXT VALUE および PREVIOUS VALUE 式の使用を許可するには、シーケンスの USAGE 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。

シーケンスに対する ALTER 特権を持つユーザーは、シーケンスの再始動、今後のシーケンス値の増分の変更といったタスクを実行できます。シーケンスの作成者は ALTER 特権を他のユーザーに GRANT でき、 WITH GRANT OPTION を使用すれば、それらのユーザーもまた、これらの特権をさらに他のユーザーに GRANT できるようになります。

ルーチン特権

EXECUTE 特権には、データベース内の関数、プロシージャー、およびメソッドといったすべてのタイプのルーチンのアクションが含まれます。 EXECUTE 特権を与えられたユーザーはルーチンを呼び出すことができ、そのルーチンからのソース関数を作成できます (関数にのみ適用されます)。また、CREATE VIEW、CREATE TRIGGER といった任意の DDL ステートメント内でルーチンを参照できます。

外部のストアード・プロシージャー、関数、またはメソッドを定義するユーザーは、 EXECUTE WITH GRANT 特権を付与されます。 WITH GRANT OPTION を使用して EXECUTE 特権を他のユーザーに GRANT した場合、そのユーザーは、さらに他のユーザーに EXECUTE 特権を GRANT できます。

ワークロードの使用特権

ワークロードの使用を有効にするには、ACCESSCTRL、SECADM、または WLMADM 権限を保持するユーザーが GRANT USAGE ON WORKLOAD ステートメントを使用して、そのワークロードの USAGE 特権をユーザー、グループ、ロールのいずれかに付与します。

DB2 データベース・システムは、一致するワークロードを検出すると、セッション・ユーザーがそのワークロードの USAGE 特権を持っているかどうかを確認します。セッション・ユーザーがそのワークロードの USAGE 特権を持っていなければ、DB2 データベース・システムは、番号付きリストの中で一致する次のワークロードを検索します。したがって、セッション・ユーザーがワークロードの USAGE 特権を持っていなければ、そのワークロードは存在しないかのように扱われる、ということです。

USAGE 特権の情報はカタログに格納されており、SYSCAT.WORKLOADAUTH ビューで表示できます。

USAGE 特権を取り消すには、REVOKE USAGE ON WORKLOAD ステートメントを使用します。

ACCESSCTRL、DATAACCESS、DBADM、SECADM、または WLMADM 権限を持つユーザーは、すべてのワークロードの USAGE 特権を暗黙的に持っています。

SYSDEFAULTUSERWORKLOAD ワークロードと USAGE 特権

RESTRICT オプションを指定しないでデータベースを作成する場合は、SYSDEFAULTUSERWORKLOAD の USAGE 特権がデータベース作成時に PUBLIC に付与されます。そうでない場合は、ACCESSCTRL、WLMADM、または SECADM 権限を持っているユーザーが USAGE 特権を明示的に付与する必要があります。

セッション・ユーザーが、SYSDEFAULTUSERWORKLOAD をはじめ、どのワークロードについても USAGE 特権を持っていない場合は、SQL エラーが戻されます。

SYSDEFAULTADMWORKLOAD ワークロードと USAGE 特権

SYSDEFAULTADMWORKLOAD の USAGE 特権をいずれかのユーザーに明示的に付与することはできません。このワークロードを使用できるのは、ACCESSCTRL、DATAACCESS、DBADM、WLMADM 権限または SECADM 権限のある SESSION 許可 ID を持っているユーザーが **SET WORKLOAD TO SYSDEFAULTADMWORKLOAD** コマンドを実行した場合に限られます。

GRANT USAGE ON WORKLOAD ステートメントも REVOKE USAGE ON WORKLOAD ステートメントも、SYSDEFAULTADMWORKLOAD には無効です。

さまざまなコンテキスト内の許可 ID

許可 ID は、識別および許可検査の 2 つの目的で使用されます。例えば、セッション許可 ID は、初期許可検査で使用されます。

特定のコンテキストで許可 ID の使用を参照する場合、次のセクションに示されているとおり、そのコンテキストを識別するように許可の参照が修飾されます。

許可 ID のコンテキスト上の参照
定義

システム許可 ID
CONNECT 処理時の CONNECT 特権の検査のように、どの初期許可検査の実行でも使用される許可 ID。 CONNECT の処理時の認証プロセスの一環として、DB2 の命名要件との互換性のある許可 ID が作成されます。これは、DB2 データベース・システム内の外部ユーザー ID を表します。システム許可 ID は、接続を作成したユーザーを表します。システム許可 ID の現行値を確認するには、SYSTEM_USER 特殊レジスターを使用します。接続でのシステム許可 ID は変更できません。

セッション許可 ID
CONNECT の処理時に実行される初期検査の後に続くどのセッション許可検査でも使用される許可 ID 。セッション許可 ID のデフォルト値は、システム許可 ID の値になります。セッション許可 ID の現行値を確認するには、SESSION_USER 特殊レジスターを使用します。 USER 特殊レジスター

は、SESSION_USER 特殊レジスターと同義です。 SET SESSION AUTHORIZATION ステートメントを使用すれば、セッション許可 ID を変更することができます。

パッケージ許可 ID
パッケージをデータベースにバインドするのに使用される許可 ID。この許可 ID は、**BIND** コマンドの **OWNER** *authorization id* オプションの値からとられます。パッケージ許可 ID は、場合によってはパッケージ・バインダーまたはパッケージ所有者と呼ばれます。

ルーチン所有者許可 ID
起動された SQL ルーチンの所有者としてシステム・カタログにリストされている許可 ID。

ルーチン呼び出し側許可 ID
SQL ルーチンを呼び出したステートメントのステートメント許可 ID である許可 ID。

ステートメント許可 ID
任意の許可要件で使用される以外に、オブジェクトの所有権の判別 (該当する場合) でも使用される特定の SQL ステートメントに関連付けられた許可 ID。これは、以下の SQL ステートメント・タイプに該当するソース許可 ID からその値をとります。

- 静的 SQL

  パッケージ許可 ID が使用されます。
- 動的 SQL (非ルーチン・コンテキストから)

  ケースごとにどの許可 ID が使用されるかを以下の表に示してあります。

パッケージの発行用の DYNAMICRULES オプションの値	**使用される許可 ID**
RUN	セッション許可 ID
BIND	パッケージ許可 ID
DEFINERUN、INVOKERUN	セッション許可 ID
DEFINEBIND、INVOKEBIND	パッケージ許可 ID

- 動的 SQL (ルーチン・コンテキストから)

  ケースごとにどの許可 ID が使用されるかを以下の表に示してあります。

パッケージの発行用の DYNAMICRULES オプションの値	**使用される許可 ID**
DEFINERUN、DEFINEBIND	ルーチン所有者許可 ID
INVOKERUN、INVOKEBIND	ルーチン呼び出し側許可 ID

ステートメント許可 ID の現行値を確認するには、CURRENT_USER 特殊レジスターを使用します。ステートメント許可 ID を直接変更することはできません。この ID は、各 SQL ステートメントの特性を反映するように、DB2 データベース・システムで自動的に変更されます。

データベースの作成時に付与されるデフォルト特権

データベースを作成すると、そのデータベース内におけるデフォルトのデータベース・レベル権限とデフォルトのオブジェクト・レベル特権が付与されます。

付与される権限および特権を、記録されるシステム・カタログ・ビューに応じて以下にリストします。

1. SYSCAT.DBAUTH
   - データベースの作成者には以下の権限が付与されます。
     - ACCESSCTRL
     - DATAACCESS
     - DBADM
     - SECADM
   - 制限のないデータベースでは、特殊なグループである PUBLIC に以下の権限が付与されます。
     - CREATETAB
     - BINDADD
     - CONNECT
     - IMPLICIT_SCHEMA
2. SYSCAT.TABAUTH

   制限のないデータベースでは、特殊なグループである PUBLIC に以下の特権が付与されます。
   - すべての SYSCAT および SYSIBM 表に関する SELECT
   - すべての SYSSTAT 表に関する SELECT および UPDATE
   - スキーマ SYSIBMADM における以下のビューでの SELECT
     - ALL_*
     - USER_*
     - ROLE_*
     - SESSION_*
     - DICTIONARY
     - TAB
3. SYSCAT.ROUTINEAUTH

   制限のないデータベースでは、特殊なグループである PUBLIC に以下の特権が付与されます。
   - スキーマ SQLJ 中のすべてのプロシージャーに関する GRANT 付きの EXECUTE
   - スキーマ SYSFUN 中のすべての関数とプロシージャーに関する GRANT 付きの EXECUTE
   - スキーマ SYSPROC 中のすべての関数とプロシージャーに関する GRANT 付きの EXECUTE (監査ルーチンを除く)
   - スキーマ SYSIBM 中のすべての表関数に関する EXECUTE
   - スキーマ SYSIBM 中の他のすべてのプロシージャーに関する EXECUTE

4. SYSCAT.MODULEAUTH

   制限のないデータベースでは、特殊なグループである PUBLIC に以下の特権が付与されます。

   - スキーマ SYSIBMADM 中の以下のモジュールに関する EXECUTE
     - DBMS_DDL
     - DBMS_JOB
     - DBMS_LOB
     - DBMS_OUTPUT
     - DBMS_SQL
     - DBMS_STANDARD
     - DBMS_UTILITY
5. SYSCAT.PACKAGEAUTH
   - データベースの作成者には以下の特権が付与されます。
     - NULLID スキーマ中に作成されたすべてのパッケージに関する CONTROL
     - NULLID スキーマ中に作成されたすべてのパッケージに関する GRANT 付きの BIND
     - NULLID スキーマ中に作成されたすべてのパッケージに関する GRANT 付きの EXECUTE
   - 制限のないデータベースでは、特殊なグループである PUBLIC に以下の特権が付与されます。
     - NULLID スキーマ内で作成されたすべてのパッケージに対する BIND
     - NULLID スキーマ中に作成されたすべてのパッケージに関する EXECUTE
6. SYSCAT.SCHEMAAUTH

   制限のないデータベースでは、特殊なグループである PUBLIC に以下の特権が付与されます。

   - スキーマ SQLJ に関する CREATEIN
   - スキーマ NULLID に関する CREATEIN
7. SYSCAT.TBSPACEAUTH

   制限のないデータベースでは、特殊なグループである PUBLIC に表スペース USERSPACE1 に関する USE 特権が付与されます。
8. SYSCAT.WORKLOADAUTH

   制限のないデータベースでは、特殊なグループである PUBLIC に SYSDEFAULTUSERWORKLOAD に関する USAGE 特権が付与されます。
9. SYSCAT.VARIABLEAUTH

   制限のないデータベースでは、特殊なグループである PUBLIC に、SYSIBM スキーマ内のスキーマ・グローバル変数に対する READ 特権が付与されます。ただし、以下の変数は除きます。

   - SYSIBM.CLIENT_ORIGUSERID
   - SYSIBM.CLIENT_USRSECTOKEN

制限のないデータベースとは、CREATE DATABASE コマンドで RESTRICTIVE オプションを使用しないで作成されたデータベースです。

アクセスの付与および取り消し

特権の付与

ほとんどのデータベース・オブジェクトに対する特権を GRANT するには、そのオブジェクトに対する ACCESSCTRL 権限、SECADM 権限、または CONTROL 特権を持つか、または WITH GRANT OPTION 特権を保持していなければなりません。また、SYSADM 権限か SYSCTRL 権限を持つユーザーは表スペース特権を付与できます。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。

このタスクについて

他ユーザーに CONTROL 特権を付与するためには、ACCESSCTRL または SECADM 権限が必要です。ACCESSCTRL、DATAACCESS、DBADM、または SECADM の各権限を付与するには、SECADM 権限がなければなりません。

GRANT ステートメントは、許可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、GROUP、および ROLE というキーワードがサポートされています。これらのオプションのキーワードが使用されない場合、データベース・マネージャーはオペレーティング・システムのセキュリティー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。また、同じ名前を持つ役割タイプの許可 ID が存在するかどうかもチェックします。許可名がユーザー、グループ、または役割を参照しているかどうかをデータベース・マネージャーが判別できないと、エラーが戻ります。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
```

特権の取り消し

REVOKE ステートメントを使用すれば、許可ユーザーは、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ます。

このタスクについて

データベース・オブジェクトに対する特権を取り消すには、ACCESSCTRL 権限、SECADM 権限、またはそのオブジェクトに対する CONTROL 特権が必要です。また表スペース特権は、SYSADM 権限と SYSCTRL 権限を持つユーザーも取り消せます。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、ACCESSCTRL または SECADM 権限が必要です。
ACCESSCTRL、DATAACCESS、DBADM、または SECADM の各権限を取り消すには、SECADM 権限がなければなりません。表スペース特権を取り消すことができるのは、SYSADM 権限か SYSCTRL 権限を保持するユーザーだけです。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: ACCESSCTRL 権限、SECADM 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 GRANT し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。
明示的に付与された表 (またはビュー) に対する特権が 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ビューに対する特権は取り消されることはありません。ビューに対する特権は DBADM 権限によって使用可能になるものであり、基本表の明示特権とは関係ないからです。

同じ名前を持つユーザー、グループ、または役割に特権が付与されている場合、特権を取り消す際には GROUP、USER、または ROLE キーワードを指定しなければなりません。次の例は、HERON というユーザーの EMPLOYEE 表に対する 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 表に対する 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ビュー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。ビューに対する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。

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ビュー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。ビューに対する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。

注: 表またはビューに対する 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見なされますが、十分な権限を持つユーザーによって再バインドされると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対する該当する権限

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。

1 つまたは複数の特権に基づいてトリガーまたは SQL 関数を定義した場合、これらの特権のいくつかを失うと、そのトリガーまたは SQL 関数を使用できなくなります。

オブジェクトの作成とドロップによる暗黙許可の管理

データベース・マネージャーは、表、パッケージなどのデータベース・オブジェクトを作成するユーザーに対して、いくつかの特権を暗黙的に付与します。特権は、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。同様に、オブジェクトをドロップすると特権はドロップされます。

このタスクについて

作成されるオブジェクトが、表、ニックネーム、索引、またはパッケージであれば、ユーザーにはそのオブジェクトに対する CONTROL 特権が与えられます。オブジェクトがビューの場合、そのビューに対する CONTROL 特権が暗黙のうちに付与されるのは、ユーザーがそのビュー定義の中で参照されるすべての表、ビュー、およびニックネームに対する CONTROL 特権を持っている場合に限られます。

明示的に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。

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。

このタスクについて

パッケージの所有者の命名には、単純なルールがあります。

- どのユーザーも、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。
- DBADM 権限を持つユーザー ID は、**OWNER** オプションを使用して、任意の許可 ID を所有者として命名できます。

DB2 データベース製品を使用して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。

パッケージを通した暗黙特権

アプリケーション・プログラムによって、および対話式ワークステーション・セッションを操作しているユーザーによって、データベース内のデータに対するアクセスが要求されることがあります。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さまざまなアクションを実行できます。そのような各アクションには、1 つまたは複数の特権が必要です。

パッケージをバインドしている個別ユーザーおよび PUBLIC に付与される特権、および個人および PUBLIC に付与されたロールに付与される特権は、静的 SQL および XQuery ステートメントのバインド時の許可検査で使用されます。グループを通して付与された特権と、グループに対して付与されたロールは、静的 SQL および XQuery ステートメントがバインドされるときの許可検査には使用されません。

パッケージのバインド時に VALIDATE RUN が指定されていない場合、パッケージをバインドする有効な許可 ID を持つユーザーは、以下のいずれかの条件を満たしていることが必要です。

- パッケージ内の静的 SQL または XQuery ステートメントを実行するのに必要なすべての特権を付与されている。
- 以下の 1 つ以上のメンバーシップにより、必要な特権を獲得済みである。
  - PUBLIC
  - PUBLIC に付与されている役割
  - ユーザーに付与されている役割

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 または XQuery 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 または XQuery 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のに適した許可 (BIND または BINDADD 特権) を持っているかどうか検査を行う場合には、PUBLIC、グループ、ロール、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、静的と動的の両方の SQL および XQuery ステートメントが入っていることがあります。静的照会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照会では、パッケージ・バインド・プログラムの特権を暗黙で取得することができます。ただし、これはパッケージに対して定められた制限の範囲内に限られます。

動的 SQL または XQuery ステートメントがパッケージに含まれる場合、必要な特権は、パッケージのプリコンパイル時またはバインド時に **DYNAMICRULES** に指定された値に応じて異なります。詳しくは、動的照会に対する **DYNAMICRULES** の影響について説明したトピックを参照してください。

ニックネームが定義されているパッケージ経由の間接特権

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、ニックネームがデータ・ソースで参照する表およびビューに関して、認証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。

例えば、パッケージ作成者の .SQC ファイルに、複数の SQL または XQuery 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、ローカル表およびニックネーム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 および XQuery ステートメントと、表とニックネームの参照の混合のみが含まれる場合、ローカル・オブジェクトとニックネームへの DB2 データベース許可検査は類似しています。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、ビュー) 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。

パッケージの実行者の許可 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: 静的 SQL および XQuery ステートメントにニックネームは指定できません。**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 データベースは DB2 ファミリーのデータ・ソースと通信するときに動的 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。

ビューを使用したデータ・アクセスの制御

ビューを使用すれば、表に対するアクセス制御や特権付与を行うことができます。

ビューを使用すると、表に対する以下のようなアクセス制御が可能になります。

- 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、許可されたユーザーは、必要な列だけを指定したビューを作成できます。
- 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、ビュー定義の副照会の中に WHERE 節を指定することにより、ビューによってアクセスする行を限定できます。
- データ・ソース表またはビュー内の行または列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 データベース・ビューを作成できます。これらのビューは、1 つまたは複数のデータ・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ビューを作成できるので、ユーザーは 1 つのビューから複数のデータ・ソースのデータにアクセスできます。これらのビューは、*マルチ・ロケーション・ビュー* と呼ばれます。このようなビューは、分散環境全体で重要な表の列の情報を結合する場合や、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。

ビューを作成するには、DATAACCESS 権限が必要です。または、そのビュー定義の中で参照されるそれぞれの表、ビュー、またはニックネームに対する CONTROL 特権あるいは SELECT 特権が必要です。さらに、ユーザーは、そのビュー用に指定されたスキーマ内にオブジェクトを作成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、DBADM 権限、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。

ニックネームを参照するビューを作成するとき、ビューでニックネームが参照するデータ・ソース・オブジェクト (表とビュー) に対する付加的な権限は必要ありません。ただし、ビューのユーザーがビューにアクセスするとき、基礎となるデータ・ソース・オブジェクトに対する 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 (表およびビュー) 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. データ・ソース表の中のユーザーのアクセスを許可する列に対してデータ・ソース・ビューを作成する
2. このビューの SELECT 特権をユーザーに付与する
3. ビュー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、ビュー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数いるとします。例えば、以下のようにします。

- 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、専用のビューを作成することによって解決できます。例えば、51 番の部署のマネージャーに対しては、次のようにビューを作成できます。

  ```
  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 ビューを照会します。このマネージャーが STAFF 表の EMP051 というビュー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 列に関するビューを作成し、対応する 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ビュー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
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

データベース管理者 (DBA) のアクセスの制御

データベース管理者 (DBADM 権限を保持するユーザー) によるデータへのアクセスをモニターしたり、制御したり、またはデータにアクセスさせないようにしたい場合があります。

データに対するアクセスのモニター

DB2 監査機能を使用すると、データベース管理者によるアクセスをモニターできます。そのためには、以下のステップに従ってください。

1. DBADM 権限を保持するユーザーに関して、どのイベントをキャプチャーするのかを決めて、そのイベントをモニターするための監査ポリシーを作成します。
2. その監査ポリシーを DBADM 権限に関連付けます。

データに対するアクセスの制御

データベース管理者によるアクセスを制御するには、役割とともにトラステッド・コンテキストを使用できます。そのためには、以下のステップに従ってください。

1. ロールを作成し、そのロールに DBADM 権限を付与します。
2. トラステッド・コンテキストを定義し、そのロールをこのトラステッド・コンテキストのデフォルトのロールとして設定します。

   そのロールのメンバーシップをいずれかの許可 ID に明示的に付与する設定は、しないでください。これを行うと、そのロールはこのトラステッド・コンテキストでのみ使用できるようになり、ユーザーが DBADM 権限を取得できるのも、そのトラステッド・コンテキストの中にいる場合に限られることになります。
3. そのトラステッド・コンテキストに対するユーザー・アクセスを制御するには、2 つの方法があります。
   - 暗黙アクセス: 各ユーザーごとに固有のトラステッド・コンテキストを作成します。そのトラステッド・コンテキストの属性に合致した通常の接続をユーザーが確立すると、そのユーザーは暗黙的に信頼され、そのロールに対するアクセスを取得します。
   - 明示アクセス: WITH USE FOR 節を使用してトラステッド・コンテキストを作成し、そのトラステッド・コンテキストにアクセスできるすべてのユーザーを定義します。それらのユーザーがデータベース要求を実行するためのアプリケーションを作成します。そのアプリケーションが明示的なトラステッド接続を確立して、ユーザーが要求を送信すると、そのアプリケーションはそのユーザー ID に切り替えて、そのユーザーとしてデータベース要求を実行します。

このトラステッド・コンテキストの使用をモニターする場合、このトラステッド・コンテキストのユーザーに関して、関心のあるイベントをキャプチャーするための監査ポリシーを作成できます。この監査ポリシーをトラステッド・コンテキストに関連付けます。

データに対するアクセスの防止

表内のデータに対するアクセスを防止するには、以下のいずれかのオプションを選択します。

- すべての表内のデータに対するアクセスを防止するには、DBADM ユーザー、役割、またはグループから DATAACCESS を取り消します。または、対象となるユーザー、役割、またはグループに DATAACCESS オプションを指定しないで DBADM を付与できます。
- 特定の 1 つの表内のデータに対するアクセスを防止するには、以下のステップに従います。

– 表内のすべての列にセキュリティー・ラベルを割り当てます。
– 役割にそのセキュリティー・ラベルを付与します。
– その役割を、表にアクセスする適正な必要性があるすべてのユーザー (または役割) に付与します。

持っている権限に関わらず、この役割のメンバーでない限り、どのユーザーもこの表内のデータにアクセスすることはできません。

間接的な方法によるデータへのアクセス権の取得

セキュリティーを正常に管理するには、ユーザーがデータへのアクセスを取得できる間接的な方法を認識しておく必要があります。

以下のリストは、ユーザーがアクセスを許可されていないデータへのアクセス権を取得するために使用できる間接的な方法を示しています。

- **カタログ・ビュー:** DB2 データベース・システムのカタログ・ビューは、データベース・オブジェクトに関するメタデータと統計を保管します。カタログ・ビューに対する SELECT アクセス権を持つユーザーは、自分には資格のないデータに関する一部の情報を取得できます。セキュリティーを向上させるには、資格あるユーザーのみがカタログ・ビューに対するアクセス権を持っていることを確認してください。

  注: DB2 Universal Database バージョン 8 以前では、カタログ・ビューに関する SELECT アクセス権はデフォルトで PUBLIC に付与されました。 DB2 バージョン 9.1 以降のデータベース・システムでは、**CREATE DATABASE** コマンドの新規の **RESTRICTIVE** オプションによって、ユーザーは、カタログ・ビューに対する SELECT アクセス権を PUBLIC に付与するか、それとも付与しないかを選択できます。
- **Explain スナップショット:** Explain スナップショットとは、SQL または XQuery ステートメントの EXPLAIN 時に収集される、圧縮された情報のことです。Explain スナップショットは、EXPLAIN_STATEMENT 表にバイナリー・ラージ・オブジェクト (BLOB) として保管され、表データに関する情報を開示する可能性のある列統計が含まれます。セキュリティーを向上させるには、Explain 表に対するアクセス権を資格あるユーザーのみに付与する必要があります。
- **セクションの Explain:** セクション Explain プロシージャー (EXPLAIN_FROM_SECTION、EXPLAIN_FROM_CATALOG、EXPLAIN_FROM_ACTIVITY、および EXPLAIN_FROM_DATA) は、パッケージ・キャッシュにあるセクションの情報を Explain 表に設定します。この情報に含まれるステートメント本文には、入力データの値が入っている可能性があります。セキュリティーを向上させるには、セクションの Explain プロシージャーおよび Explain 表に対するアクセス権を資格あるユーザーのみに付与する必要があります。
- **ログ・リーダー関数:** ログを読み取る関数の実行を許可されているユーザーは、ログ・レコードの形式を理解できれば、許可されていないデータに対するアクセス権を取得できます。以下の関数がログを読み取ります。

関数	関数の実行に必要な権限
db2ReadLog	SYSADM または DBADM

関数	関数の実行に必要な権限
db2ReadLogNoConn	なし。

- **複製:** データを複製する際には、保護データであっても宛先に再作成されます。セキュリティーを向上させるには、宛先がソースの場所と少なくとも同じほど安全であることを確認してください。
- **例外表:** データを表にロードしている際に例外表を指定すると、例外表に対するアクセス権を持つユーザーは許可されていない情報を取得できます。セキュリティーを向上させるには、例外表に対するアクセス権を許可ユーザーのみに付与し、例外表を処理し終えたらできるだけ早くドロップしてください。
- **表スペースまたはデータベースのバックアップ:** **BACKUP DATABASE** コマンドを実行する権限を持つユーザーは、保護データを含むデータベースまたは表スペースのバックアップを取り、そのデータを他の場所にリストアできます。ユーザーが本来アクセス権を持てないデータをバックアップに組み込むことができます。

  SYSADM、SYSCTRL、または SYSMAINT 権限を持つユーザーが **BACKUP DATABASE** コマンドを実行できます。
- **セッション許可の設定:** DB2 Universal Database バージョン 8 以前では、DBADM 権限を持つユーザーは SET SESSION AUTHORIZATION SQL ステートメントを使用してセッション許可 ID をデータベース・ユーザーに設定できました。DB2 バージョン 9.1 以降のデータベース・システムでは、ユーザーがセッション許可 ID を設定できるようにするには、その前に GRANT SETSESSIONUSER ステートメントを使用して明示的に許可されていなければなりません。

  しかし、既存のバージョン 8 データベースを DB2 バージョン 9.1 以降のデータベース・システムにアップグレードする際には、(例えば、SYSCAT.DBAUTH で付与されて) 既存の明示 DBADM 権限を持つユーザーは、引き続きセッション許可をデータベース・ユーザーに設定できます。それが許可されているのは、既存のアプリケーションが引き続き作動するようにするためです。セッション許可を設定できるということは、すべての保護データにアクセスできる可能性があるということになります。セキュリティーの制限を増すには、REVOKE SETSESSIONUSER SQL ステートメントを実行して、この設定をオーバーライドできます。
- **ロックのモニター:** HIST_AND_VALUES コレクション・レベルが指定されていると、DB2 データベース管理システムのロック・モニター・アクティビティーの一環として、パラメーター・マーカーに関連付けられた値がモニター出力に書き込まれます。ロック・イベント・モニターによってキャプチャーされるステートメント本文にも、値が埋め込まれている可能性があります。モニター出力に対するアクセス権を持つユーザーは、許可されていない情報に対するアクセス権を取得できます。
- **アクティビティー・モニター:** VALUES 節が指定されていると、アクティビティー・イベント・モニターを使用する DB2 データベース管理システムのモニター・アクティビティーの一環として、パラメーター・マーカーに関連付けられた値がモニター出力に書き込まれます。また、WITH DETAILS 節が指定されていると、ステートメント本文 (入力データの値を含む可能性がある) がモニター出力に書き込まれます。モニター出力に対するアクセス権を持つユーザーは、許可さ

れていない情報に対するアクセス権を取得できます。セキュリティーを向上させるには、CREATE EVENT MONITOR ステートメントおよび全イベント・モニター表に対するアクセス権を、資格あるユーザーのみに付与する必要があります。

- **パッケージ・キャッシュ・モニター:** パッケージ・キャッシュ・イベント・モニターを使用すると、DB2 データベース管理システムのパッケージ・キャッシュ・モニターの一環として、パッケージ・キャッシュからセクションが排出される際に、ステートメント本文 (入力データの値を含む可能性がある) が、必ずモニター出力に書き込まれます。セキュリティーを向上させるには、CREATE EVENT MONITOR ステートメントおよび全イベント・モニター表に対するアクセス権を、資格あるユーザーのみに付与する必要があります。
- **モニターに関する表関数、ビュー、およびレポート:** モニターに関する以下の表関数、ビュー、およびレポートは、現在実行中のステートメント、またはパッケージ・キャッシュ内のステートメントについて、そのステートメント本文を表示します。
  - SYSPROC.MON_GET_ACTIVITY_DETALS
  - SYSPROC.MON_GET_PKG_CACHE_STMT
  - SYSPROC.MON_GET_PKG_CACHE_STMT_DETALS
  - SYSIBMADM.MON_PKG_CACHE_SUMMARY
  - SYSIBMADM.MON_CURRENT_SQL
  - SYSIBMADM.MON_LOCKWAITS
  - SYSIBMADM.MONREPORT.LOCKWAIT
  - SYSIBMADM.MONREPORT.CURRENTSQL
  - SYSIBMADM.MONREPORT.PKGCACHE

  ステートメント本文には、入力データの値が入っている可能性があります。セキュリティーを向上させるには、これらの表関数およびレポートに対する EXECUTE 特権と、これらのビューに対する SELECT 特権を、資格あるユーザーのみに付与する必要があります。
- **トレース:** トレースに表データを含めることができます。この種のトレースに対するアクセス権を持つユーザーは、許可されていない情報に対するアクセス権を取得できます。
- **ダンプ・ファイル:** DB2 データベース製品は、特定の問題のデバッグに役立つように、`sqllib¥db2dump` ディレクトリーにメモリー・ダンプ・ファイルを生成することがあります。これらのメモリー・ダンプ・ファイルに表データが含まれることがあります。その場合、このファイルに対するアクセス権を持つユーザーは、許可されていない情報に対するアクセス権を取得できます。セキュリティーを向上させるには、`sqllib¥db2dump` ディレクトリーに対するアクセスを制限する必要があります。
- **db2dart: `db2dart`** ツールは、データベースを調べ、検出した設計上のエラーを報告します。このツールは表データにアクセスでき、DB2 はこのアクセスに関するアクセス制御を施行しません。**`db2dart`** ツールを実行する権限を持つユーザーや、**`db2dart`** 出力に対するアクセス権を持つユーザーは、許可されていない情報に対するアクセス権を取得できます。

- **REOPT バインド・オプション: REOPT** バインド・オプションを指定すると、再最適化可能な追加バインド SQL ステートメントごとの Explain スナップショット情報が実行時に Explain 表に入れられます。EXPLAIN は入力データ値も表示します。
- **db2cat: db2cat** ツールは、表のパックされた記述子のダンプに使用します。表のパックされた記述子には、表の内容に関する情報を開示する可能性のある統計が含まれます。**db2cat** ツールを実行するユーザーや、この出力に対するアクセス権を持つユーザーは、許可されていない情報に対するアクセス権を取得できます。

データ暗号化

DB2 データベース・システムには、ストレージ内にあるデータであれ、ネットワークを介して転送中のデータであれ、データを暗号化するための幾つかの方法があります。

ストレージ内のデータの暗号化

ストレージ内のデータを暗号化するには、以下のオプションがあります。

- データベース表内のデータを暗号化するには、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 を使用できます。
- 基礎となるオペレーティング・システム・データおよびバックアップ・ファイルを暗号化するには、IBM Database Encryption Expert を使用できます。
- AIX オペレーティング・システムで DB2 Enterprise Server Edition システムを実行している場合に、ファイル・レベルの暗号化のみを対象とするのであれば、暗号化ファイル・システム (EFS) を使用して、オペレーティング・システム・データとバックアップ・ファイルを暗号化できます。

転送中のデータの暗号化

クライアントと DB2 データベースの間で転送中のデータを暗号化するには、DATA_ENCRYPT 認証タイプを使用するか、または Secure Sockets Layer (SSL) 用の DB2 データベース・システム・サポートを使用できます。

ENCRYPT、DECRYPT_BIN、DECRYPT_CHAR、および GETHINT 関数の使用

ENCRYPT 組み込み関数は、パスワード・ベースの暗号化方式によって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・ヒントが、暗号化データに組み込まれます。暗号化したデータを復号するには、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、VARCHAR FOR BIT DATA です (32631 という制限があります)。

暗号化できるデータは、CHAR、VARCHAR、および FOR BIT DATA だけです。

DECRYPT_BIN および DECRYPT_CHAR 関数は、パスワード・ベースの復号を使用して、データを復号します。

DECRYPT_BIN は常に VARCHAR FOR BIT DATA を戻し、 DECRYPT_CHAR は常に VARCHAR を戻します。最初の引数が CHAR FOR BIT DATA または VARCHAR FOR BIT DATA の可能性があるため、結果が最初の引数と異なる場合もあります。

結果の長さは、次の 8 バイト境界までのバイト数に依存します。オプションのヒント・パラメーターが指定されている場合、結果の長さは、データ引数の長さプラス 40 に、次の 8 バイト境界までのバイト数を加えた長さになる可能性があります。オプションのヒント・パラメーターが指定されない場合、結果の長さは、データ引数の長さプラス 8 に、次の 8 バイト境界までのバイト数を加えた長さになる可能性があります。

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。例えば、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復号されます。
- 暗号化パスワード特殊レジスター。 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。 ENCRYPTION PASSWORD 特殊レジスターは、暗号化形式でのみ保管されます。

  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。

DB2 インスタンスの Secure Sockets Layer (SSL) サポートの構成

DB2 データベース・システムは、SSL をサポートしています。したがって、SSL をサポートする DB2 クライアント・アプリケーションは、SSL ソケットを使用して DB2 データベースに接続できます。CLI、CLP、および .Net Data Provider クライアント・アプリケーション、および IBM Data Server Driver for JDBC and SQLJ (タイプ 4 の接続) を使用するアプリケーションは、SSL をサポートします。

始める前に

SSL サポートを構成する前に、以下のステップを実行します。

- Windows プラットフォームでは、**PATH** 環境変数に、Linux と UNIX プラットフォームでは、**LIBPATH**、**SHLIB_PATH**、**LD_LIBRARY_PATH** のいずれかの環境変数に、IBM Global Security Kit (GSKit) ライブラリーのパスが設定されていることを確認します。DB2 データベース・システムをインストールするときに、GSKit は自動的に組み込まれます。

  Windows 32 ビット・プラットフォームでは、GSKit ライブラリーは `C:¥Program Files¥IBM¥GSK8¥lib` にあります。この場合、システムの **PATH** には `C:¥Program Files¥IBM¥GSK8¥lib` が含まれている必要があります。Windows 64 ビット・プラットフォームでは 64 ビット GSKit ライブラリーは `C:¥Program Files¥IBM¥GSK8¥lib64` にあり、32 ビット GSKit ライブラリーは `C:¥Program Files (x86)¥IBM¥GSK8¥lib` にあります。

  UNIX と Linux プラットフォームでは、GSKit ライブラリーは `sqllib/lib/gskit` にあります。

  Windows 以外のプラットフォームでは、DB2 データベース・マネージャーは GSKit をローカルにインストールし、GSKit ライブラリーは特定のインスタンスの `sqllib/lib/gskit` または `sqllib/lib64/gskit` にあります。インスタンスを始動するために、GSKit の別のコピーをグローバルな場所にインストールする必要はありません。GSKit のグローバル・コピーが存在する場合は、グローバルな GSKit のバージョンがローカルの GSKit のバージョンと同じになるようにしてください。
- 接続コンセントレーターがアクティブになっていないことを確認します。接続コンセントレーターの実行中は、DB2 インスタンスで SSL サポートが有効になりません。

  接続コンセントレーターがアクティブになっているかどうかを確認するには、**GET DATABASE MANAGER CONFIGURATION** コマンドを実行します。構成パラメーター **max_connections** が **max_coordagents** の値より大きい値に設定されている場合は、接続コンセントレーターがアクティブになっています。

このタスクについて

SSL 通信は、常に FIPS モードになります。

DB2 Connect の SSL サポート

中間サーバー・コンピューターで DB2 Connect for System i、DB2 Connect for System z、または DB2 Enterprise Server Edition を使用して、DB2 クライアントをホストまたは System i のデータベースに接続する場合、以下のいずれかの構成で SSL 接続がサポートされます。

- クライアントと DB2 Connect サーバー間
- DB2 Connect サーバーとサーバー間
- クライアントと DB2 Connect サーバーおよび DB2 Connect サーバーとサーバーの両方の間

注: SSL サポートが構成内のすべてのパスで使用可能になるためには、各クライアントまたはサーバーが SSL サポートの要件をすべて満たしている必要があります。例えば、DB2 Connect 接続コンセントレーターがオンにな

っている場合、DB2 Connect サーバーへのインバウンド要求は SSL を使用できません。ただし、ターゲット・サーバーへのアウトバウンド要求は、SSL を使用できます。

高可用性災害時リカバリー (HADR) システムの SSL サポート

SSL はクライアントと HADR 1 次サーバー間でサポートされています。SSL を使用して HADR 1 次サーバーに接続しているクライアントは、SSL を使用して HADR スタンバイ・データベースにリルートできます。ただし、SSL は HADR 1 次サーバーとスタンバイ・サーバー間ではサポートされていません。

GSKit ツールの資料: GSKCapiCmd

GSKit ツール GSKCapiCmd についての詳細は、「*GSKCapiCmd User's Guide*」を参照してください。これは、ftp://ftp.software.ibm.com/software/webserver/appserv/library/v80/GSK_CapiCmd_UserGuide.pdfから入手できます。

SSL サポートの構成

SSL サポートを構成するには、まず、鍵データベースを作成してデジタル証明書を管理します。これらの証明書および暗号鍵は、SSL 接続を確立するのに使用します。次に、DB2 インスタンスの所有者は、SSL サポートの DB2 インスタンスを構成する必要があります。

手順

1. 鍵データベースを作成し、デジタル証明書をセットアップします。
   a. GSKCapiCmd ツールを使用して、鍵データベースを作成します。これは、Certificate Management System (CMS) タイプの鍵データベースでなければなりません。 GSKCapiCmd は非 Java ベースのコマンド行ツールで、このツールを使用するためにシステムに Java がインストールされている必要はありません。

      「*GSKCapiCmd User's Guide*」に説明されているとおり、**gskcapicmd** コマンドを使用して、GSKCapiCmd を呼び出します。このコマンドのパスは Linux および UNIX プラットフォームでは `sqllib/gskit/bin` で、32 ビットと 64 ビットの Windows プラットフォームではどちらも `C:¥Program Files¥IBM¥GSK8¥bin` です。 (64 ビット・プラットフォームでは、32 ビット GSKit 実行可能ファイルおよびライブラリーもあります。この場合、このコマンドのパスは `C:¥Program Files (x86)¥IBM¥GSK8¥bin` です。)Windows プラットフォームの場合は、PATH に適切な GSKit ライブラリー・パスが必ず含まれるようにしてください。UNIX プラットフォームまたは Linux プラットフォームの場合は、LIBPATH、SHLIB_PATH、または LD_LIBRARY_PATH に適切なライブラリー・パス (sqllib/lib64/gskit など) が必ず含まれるようにしてください。

      例えば、以下のコマンドを使用すると、`mydbserver.kdb` という鍵データベースと、`mydbserver.sth` という stash ファイルが作成されます。

      ```
      gsk8capicmd_64 -keydb -create -db "mydbserver.kdb" -pw "myServerPassw0rdpw0"
            -stash
      ```

-stash オプションは、鍵データベースと同じパスに、ファイル拡張子 **.sth** で、stash ファイルを作成します。インスタンスの始動時に、GSKit は stash ファイルを使用して、鍵データベースへのパスワードを取得します。

注: この stash ファイルには強力なファイル・システム保護を使用する必要があります。デフォルトでは、インスタンス所有者のみがこのファイルへのアクセス権限 (読み取りアクセス権限と書き込みアクセス権限の両方) を持っています。

鍵データベースを作成するとき、Verisign などのいくつかの認証局 (CA) から自動的に署名者証明書が取り込まれます。

b. サーバーの証明書を鍵データベースに追加します。サーバーは、SSL ハンドシェーク中にこの証明書をクライアントに送信して、サーバーのための認証を提供します。 証明書を取得するために、GSKCapiCmd を使用して新規認証要求を作成して、署名されるようにそれを CA にサブミットするか、テスト目的で自己署名証明書を作成できます。

例えば、myselfsigned というラベルを持つ自己署名証明書を作成するには、以下の例に示されているように **GSKCapiCmd** コマンドを使用します。

```
gsk8capicmd_64 -cert -create -db "mydbserver.kdb" -pw "myServerPassw0rdpw0"
        -label "myselfsigned" -dn "CN=myhost.mycompany.com,O=myOrganization,
        OU=myOrganizationUnit,L=myLocation,ST=ON,C=CA"
```

c. DB2 サーバーとの SSL 接続を確立するクライアントを実行しているコンピューターに証明書を配布できるように、作成したばかりの証明書をファイルに抽出します。

例えば、以下の GSKCapiCmd コマンドは、mydbserver.arm というファイルに証明書を抽出します。

```
gsk8capicmd_64 -cert -extract -db "mydbserver.kdb" -pw "myServerPassw0rdpw0"
       -label "myselfsigned" -target "mydbserver.arm" -format ascii -fips
```

2. SSL をサポートするように DB2 サーバーをセットアップするには、DB2 インスタンス所有者としてログインし、以下の構成パラメーターおよび **DB2COMM** レジストリー変数を設定します。

a. **ssl_svr_keydb** 構成パラメーターを鍵データベース・ファイルの絶対パスに設定します。 例えば、以下のようにします。

```
db2 update dbm cfg using SSL_SVR_KEYDB
        /home/test/sqllib/security/keystore/key.kdb
```

ssl_svr_keydb が NULL (未設定) の場合、SSL サポートは使用できません。

b. **ssl_svr_stash** 構成パラメーターを stash ファイルの絶対パスに設定します。 例えば、以下のようにします。

```
db2 update dbm cfg using SSL_SVR_STASH
       /home/test/sqllib/security/keystore/mydbserver.sth
```

ssl_svr_stash が NULL (未設定) の場合、SSL サポートは使用できません。

c. **ssl_svr_label** 構成パラメーターをサーバーのデジタル証明書のラベル (ステップ 1 で追加した) に設定します。 **ssl_svr_label** が設定されていない場

合、鍵データベースのデフォルトの証明書が使用されます。鍵データベースにデフォルトの証明書がない場合、SSL は使用できません。 例えば、以下のようにします。db2 update dbm cfg using SSL_SVR_LABEL myselfsigned ここで、*myselfsigned* はサンプル・ラベルです。

d. **ssl_svcename** 構成パラメーターを、DB2 データベース・システムが SSL 接続を listen するポートに設定します。 TCP/IP と SSL の両方が使用可能である (**DB2COMM** レジストリー変数が 'TCPIP, SSL' に設定されている) 場合、**ssl_svcename** を **svcename** が設定されているポートとは異なるポートに設定する必要があります。**svcename** 構成パラメーターは、DB2 データベース・システムが TCP/IP 接続を listen するポートを設定します。**ssl_svcename** を **svcename** と同じポートに設定する場合、TCP/IP も SSL も使用できません。**ssl_svcename** が NULL (未設定) の場合、SSL サポートは使用できません。

   注: HADR 環境では、1 次データベース・システムまたはスタンバイ・データベース・システムの **hadr_local_svc** を、**ssl_svcename** に設定したのと同じ値に設定しないでください。また、**hadr_local_svc** を **svcename**、または **svcename** プラス 1 と同じ値に設定しないでください。

   注: DB2COMM レジストリー変数が 'TCPIP,SSL' に設定されているとき、例えば、**svcename** 構成パラメーターが NULL に設定されているために、TCPIP サポートが適切に使用できない場合、エラー SQL5043N が戻され、SSL サポートは使用できません。

e. (オプション) サーバーがどの暗号スイートを使用できるかを指定するには、**ssl_cipherspecs** 構成パラメーターを設定します。 **ssl_cipherspecs** を NULL (未設定) のままにしておくと、GSKit はクライアントとサーバーの両方によってサポートされる、使用可能な最も強力な暗号スイートを選出できます。 どの暗号スイートが使用可能であるかについての詳細は、86 ページの『サポートされる暗号スイート』を参照してください。

f. **DB2COMM** レジストリー変数に値 SSL を追加します。 例えば、以下のようにします。

   ```
   db2set -i db2inst1 DB2COMM=SSL
   ```

   db2inst1 は、DB2 インスタンス名です。 データベース・マネージャーは、同時に複数のプロトコルをサポートできます。例えば、通信プロトコルとして TCP/IP と SSL の両方を有効にするには、以下のようにします。

   ```
   db2set -i db2inst1 DB2COMM=SSL,TCPIP
   ```

g. DB2 インスタンスを再始動します。 例えば、以下のようにします。

   ```
   db2stop
   db2start
   ```

例

以下の例は、証明書を表示する方法を示しています。この例では、以下のコマンドによって作成される自己署名証明書を使用します。

```
gsk8capicmd_64 -cert -create -db "mydbserver.kdb" -pw "mydbserverpw0"
  -label "myselfsigned" -dn "CN=myhost.mycompany.com,O=myOrganization,
        OU=myOrganizationUnit,L=myLocation,ST=ON,C=CA"
```

証明書を表示するために、以下のコマンドを発行します。

```
gsk8capicmd_64 -cert -details -db "mydbserver.kdb" -pw "mydbserverpw0"
  -label "myselfsigned"
```

以下のように出力が表示されます。

```
label : myselfsigned
key size : 1024
version : X509 V3
serial : 96c2db8fa769a09d
issue:CN=myhost.mycompany.com,O=myOrganization,OU=myOrganizationUnit,
   L=myLocation,ST=ON,C=CA
subject:CN=myhost.mycompany.com,O=myOrganization,OU=myOrganizationUnit,
   L=myLocation,ST=ON,C=CA
not before : Tuesday, 24 February 2009 17:11:50 PM
not after : Thursday, 25 February 2010 17:11:50 PM
public Key
    30 81 9F 30 0D 06 09 2A 86 48 86 F7 0D 01 01 01
    05 00 03 81 8D 00 30 81 89 02 81 81 00 B6 B8 DC
    79 69 62 C9 A5 C1 5C 38 31 53 AB 27 BE 63 C0 DB
    DE C6 BC 2E A4 0D 37 45 95 22 0E 83 32 FE 67 A9
    2F D7 51 FF 40 A3 76 68 B9 E3 34 CB 7D 4A D8 38
    CA B1 6B 32 66 74 8F E2 B8 DA 8F D0 F3 62 04 BE
    C4 FE 80 2A D0 FF 27 72 37 9A 36 1D DB D3 A1 33
    A1 A6 48 33 E9 64 B9 9B 6B DB 08 60 7D 5E 0E 20
    0A 26 AA 62 3A DF D3 78 56 DC 15 DE 9F 0B 91 DD
    3B 1B 2B E2 82 FA 24 FF 81 A3 F7 3F C1 02 03 01
    00 01
public key type : RSA : 1.2.840.113549.1.1.1
finger print : SHA1 :
    2D C1 93 F8 AC A0 8F E2 C2 05 D8 23 D7 5D 87 E6
    82 3C 47 EC
signature algorithm : SHA1WithRSASignature : 1.2.840.113549.1.1.5
value
    0E 80 24 98 F6 6E 89 43 76 57 76 7F 82 95 18 6A
    43 A5 81 EC F4 82 1F 1F F2 3F E5 61 67 48 C0 59
    94 17 8E 8F DE 4F 7C 35 0C 5D A7 98 73 2A 34 7D
    1E BA 53 78 A5 E4 31 45 D1 08 86 BE 5E 57 C6 9D
    B5 E7 A7 01 3F 54 01 5E 8F 8B 2F 66 19 24 1E A4
    94 58 B0 D4 40 95 AB 98 C2 EF 1C 5C 4A 29 48 EC
    8C C0 A2 B1 AC 2A E9 3C 14 E5 77 B2 A6 55 A8 21
    CB 59 81 86 79 F0 46 35 F8 FC 99 2D EC D4 B9 EB
Trusted : enabled
```

サーバー用に (自己署名証明書ではなく) CA 署名付き証明書を取得するには、証明書署名要求を生成し、 VeriSign のような既知の CA に費用を支払い、証明書に署名してもらいます。署名付き証明書が送り返されたら、サーバーの鍵データベース内に受け取ります。以下の例は、証明書を要求して受け取る方法を示しています。ここでは、試用版の証明書を使用します。

1. まず、mydbserver.kdb 用の証明書署名要求 (Certificate Signing Request (CSR)) を作成します。 以下のコマンドによって、指定された鍵データベース内に、新規の RSA 公開鍵/秘密鍵ペアおよび PKCS10 認証要求が作成されます。CMS 鍵データベースの場合、認証要求情報は「.rdb」拡張子のファイルに保管されます。 **-file** オプションで指定するファイルは、 CA に送信する必要のあるファイルです。

   ```
   gsk8capicmd_64 -certreq -create -db "mydbserver.kdb" -pw "mydbserverpw0"
      -label "mycert" -dn "CN=myhost.mycompany.com,
       O=myOrganization,OU=myOrganizationUnit,L=myLocation,ST=ON,C=CA",
      -file "mycertRequestNew"
   ```

   以下のコマンドは、データベース・サーバー用の認証要求の詳細情報をリストします。

```
gsk8capicmd_64 -certreq -details -showOID -db "mydbserver.kdb"
   -pw "mydbserverpw0" -label "mycert"
```

以下のように出力が表示されます。

```
label : mycert
key size : 1024
subject :    Common Name (CN):
    Type : 2.5.4.3
    Value: myhost.mycompany.com
   Organization (O):
    Type : 2.5.4.10
    Value: myOrganization
   Organizational Unit (OU):
    Type : 2.5.4.11
    Value: myOrganizationUnit
   Locality (L):
    Type : 2.5.6.3
    Value: myLocation
   State (ST):
    Type : ?
    Value: Ontario
   Country or region (C):
    Type : 2.5.4.6
    Value: CA
public Key
    30 81 9F 30 0D 06 09 2A 86 48 86 F7 0D 01 01 01
    05 00 03 81 8D 00 30 81 89 02 81 81 00 9C B4 62
    3C 89 02 4E B0 D8 EA 0B B8 CC 70 63 4A 59 1F 0F
    FD 98 9A 1A 39 94 E3 43 C1 63 7A CD 21 47 57 D9
    86 6F 11 B8 91 08 AC E3 E2 21 32 FE 43 1F 07 C9
    F5 40 6B 3E 4D 56 35 05 62 D6 78 0B E3 97 28 F7
    27 31 A4 05 BE F2 3A 44 6B D8 D1 FF 1E DA 59 63
    E6 49 52 39 45 9C 1E 8E CC DA A1 D9 0F 3A 96 09
    66 5C 89 23 2E EE 31 65 8D 87 8E B9 61 C6 69 BC
    A5 DB EB 03 16 E6 33 85 14 68 BC DD F1 02 03 01
    00 01
finger print :
e0dcde10ded3a46a53c0190e84cc994e
5d7e4bad
attributes
signature algorithm1.2.840.113549.1.1.5
value
    4F 06 B4 E3 1F 00 B4 81 90 CC A2 99 4A 02 68 D0
    84 B5 7F 33 0B F0 04 D5 7D 4C 5C CB 5C D3 37 77
    E2 6D 10 17 50 19 D0 7F 61 C7 C8 54 7B DB CD 6F
    47 9F 7E 7E 5A CC 64 20 85 95 A8 5E C7 7D FB F4
    8A 7F 4B 74 6F 0A C6 EF 09 E7 0A 15 17 CC 1D D2
    5D ED 02 A1 BE 1D FC F2 65 EB 0D E2 93 BC 88 4C
    4C 73 76 16 9F 1B 12 3B 7A 01 CF E0 63 97 E8 38
    02 FB 47 EE F2 17 54 66 4D F7 7F 9E 13 DA 76 A2
```

認証要求ファイルを表示するには、次のようにします。

```
$ cat mycertRequestNew

-----BEGIN NEW CERTIFICATE REQUEST-----
MIIBrjCCARcCAQAwbjELMAkGA1UEBhMCQ0ExEDAOBgNVBAgTB09udGFyaW8xEDAO
BgNVBAcTB01hcmtoYW0xDDAKBgNVBAoTA0lCTTEMMAoGA1UECxMDREIyMR8wHQYD
VQQDExZnaWxlcmEudG9yb2xhYi5pYm0uY29tMIGfMA0GCSqGSIb3DQEBAQUAA4GN
ADCBiQKBgQCctGI8iQJOsNjqC7jMcGNKWR8P/ZiaGjmU40PBY3rNIUdX2YZvEbiR
CKzj4iEy/kMfB8n1QGs+TVY1BWLWeAvjlyj3JzGkBb7yOkRr2NH/HtpZY+ZJUjlF
nB6OzNqh2Q86lglmXIkjLu4xZY2Hjrlhxmm8pdvrAxbmM4UUaLzd8QIDAQABoAAw
DQYJKoZIhvcNAQEFBQADgYEATwa04x8AtIGQzKKZSgJo0IS1fzML8ATVfUxcy1zT
```

```
N3fibRAXUBnQf2HHyFR7281vR59+flrMZCCFlahex3379Ip/S3RvCsbvCecKFRfM
HdJd7QKhvh388mXrDeKTvIhMTHN2Fp8bEjt6Ac/gY5foOAL7R+7yF1RmTfd/nhPa
dqI=
-----END NEW CERTIFICATE REQUEST-----
```

認証要求を削除する必要がある場合は、以下のようなコマンドを使用します。

```
gsk8capicmd_64 -certreq -delete -db "mydbserver.kdb" -pw "mydbserverpw0"
   -label "mycert"
```

2. 次に、VeriSign の Web サイトを開き登録を行うと、要求ファイルをカット・アンド・ペーストして要求を送信するように求められます。試用版の場合、署名付き証明書を含む E メールが受信されます。 E メールには、試用版ルート CA 証明書および試用版中間 CA 証明書をダウンロードするためのリンクも含まれています。ノートパッドまたは vi を使用して 3 つの証明書をすべてファイルに保存します。
   - RootCert.arm
   - IntermediateCert.arm
   - MyCertificate.arm

   これら 3 つは信頼のチェーンでつながっています。

   以下のコマンドで、試用版のルート CA 証明書を mydbserver.kdb に追加します。

```
gsk8capicmd_64 -cert -add -db "mydbserver.kdb" -pw "mydbserverpw0"
   -label "trialRootCACert"  -file RootCert.arm -format ascii
```

   以下のコマンドで、試用版の中間 CA 証明書を mydbserver.kdb に追加します。

```
gsk8capicmd_64 -cert -add -db "mydbserver.kdb" -pw "mydbserverpw0"
   -label "trialIntermediateCACert" -file IntermediateCert.arm -format ascii
```

   以下のコマンドで、試用版の証明書を mydbserver.kdb 内に受け取ります。

```
$ cat SSLCertificate.cer2

-----BEGIN CERTIFICATE-----
MIIFVjCCBD6gAwIBAgIQdOydrySM+J4uUPNzbPHhVjANBgkqhkiG9w0BAQUFADCB
yzELMAkGA1UEBhMCVVMxFzAVBgNVBAoTDlZlcmlTaWduLCBJbmMuMTAwLgYDVQQL
EydGb3IgVGVzdCBQdXJwb3NlcyBPbmx5LiAgTm8gYXNzdXJhbmNlcy4xQjBABgNV
BAsTOVRlcm1zIG9mIHVzZSBhdCBodHRwczovL3d3dy52ZXJpc2lnbi5jb20vY3Bz
L3Rlc3RjYSAoYykwNTEtMCsGA1UEAxMkVmVyaVNpZ24gVHJpYWwgU2VjdXJlIFNl
cnZlciBUZXN0IENBMB4XDTA5MDIyMzAwMDAwMFoXDTA5MDMwOTIzNTk1OVowgaox
CzAJBgNVBAYTAkNBMRAwDgYDVQQIEwdPbnRhcmlvMRAwDgYDVQQHFAdNYXJraGFt
MQwwCgYDVQQKFANJQk0xDDAKBgNVBAsUA0RCMjE6MDgGA1UECxQxVGVybXMgb2Yg
dXNlIGF0IHd3dy52ZXJpc2lnbi5jb20vY3BzL3Rlc3RjYSAoYykwNTEfMB0GA1UE
AxQWZ2lsZXJhLnRvcm9sYWIuaWJtLmNvbTCBnzANBgkqhkiG9w0BAQEFAAOBjQAw
gYkCgYEAnLRiPIkCTrDY6gu4zHBjSlkfD/2Ymho5lONDwWN6zSFHV9mGbxG4kQis
4+IhMv5DHwfJ9UBrPk1WNQVi1ngL45co9ycxpAW+8jpEa9jR/x7aWWPmSVI5RZwe
jszaodkPOpYJZlyJIy7uMWWNh465YcZpvKXb6wMW5jOFFGi83fECAwEAAaOCAdcw
ggHTMAkGA1UdEwQCMAAwCwYDVR0PBAQDAgWgMEMGA1UdHwQ8MDowOKA2oDSGMmh0
dHA6Ly9TVlJTZWN1cmUtY3JsLnZlcmlzaWduLmNvbS9TVlJUcmlhbDIwMDUuY3Js
MEoGA1UdIARDMEEwPwYKYIZIAYb4RQEHFTAxMC8GCCsGAQUFBwIBFiNodHRwczov
L3d3dy52ZXJpc2lnbi5jb20vY3BzL3Rlc3RjYTAdBgNVHSUEFjAUBggrBgEFBQcD
AQYIKwYBBQUHAwIwHwYDVR0jBBgwFoAUZiKOgeAxWd0qf6tGxTYCBnAnh1oweAYI
KwYBBQUHAQEEbDBqMCQGCCsGAQUFBzABhhhodHRwOi8vb2NzcC52ZXJpc2lnbi5j
b20wQgYIKwYBBQUHMAKGNmh0dHA6Ly9TVlJTZWN1cmUtYWlhLnZlcmlzaWduLmNv
bS9TVlJUcmlhbDIwMDUtYWlhLmNlcjBuBggrBgEFBQcBDARiMGChXqBcMFowWDBW
FglpbWFnZS9naWYwITAfMAcGBSsOAwIaBBRLa7kolgYMu9BSOJsprEsHiyEFGDAm
FiRodHRwOi8vbG9nby52ZXJpc2lnbi5jb20vdnNsb2dvMS5naWYwDQYJKoZIhvcN
AQEFBQADggEBAKs1YpIeOAL6mTryIXpYfokkzRdwP5ooDutHhVbRYcPwq9ynOrHM
3gZolv8th5PpSkZAGTPr3HJZG6HnxRiQjPT88PAADR3SEzVMzQEESHfYToF1qBPZ
```

```
svigphI9eIHcg5IWwv7dyuXtkFGbTCqcvEqJiT3UHhubgMfoTuTGayhNoGt75FGU
h4kSJz3af6MNuGmQLs4wzJTepU7srlhGV1C1ujTCydax2BiWfWwO4YaFcckvHxbR
6I7vVj1PTC2RO8n5qcWJYmGU0PG3d58hJETD4E8tAReh21ShBWDgn4+e0k1XtQ8K
lB66QpsFYGTLtGyd/4w4BAgq/QLmcs+mpjc=
-----END CERTIFICATE-----

gsk8capicmd_64 -cert -receive -file MyCertificate.arm -db "mydbserver.kdb"
   -pw "mydbserverp -format ascii
```

以下のコマンドで、mydbserver.kdb 内のすべての証明書をリストします。

```
gsk8capicmd_64 -cert -list all -db "mydbserver.kdb" -pw "mydbserverpw0"

certificates found
* default, - personal, ! trusted
-!      mycert
!       trialIntermediateCACert
!       trialRootCACert
-!      myselfsigned
db2 update dbm cfg using SSL_SVR_LABEL mycert
```

非 JAVA DB2 クライアントの Secure Sockets Layer (SSL) サポートの構成

CLI、CLP、および .Net Data Provider クライアントなどの DB2 データベース・クライアントを構成して、DB2 サーバーとの通信のための Secure Sockets Layer (SSL) をサポートできます。

始める前に

注: バージョン 9.7 DB2 クライアントまたは DB2 Connect サーバーが、DB2 for z/OS V1.8、V1.9、または V1.10 システム上の z/OS サーバーとの SSL 接続を確立する場合、Communication Server for z/OS IP Service に対して APAR PK72201 用の適切な PTF を適用する必要があります。

注: GSKit バージョン 8 と、7.0.4.20 より前の GSKit 7d バージョンの間に互換性がないために、7.0.4.20 より前の GSKit 7d バージョンを使用して IDS データ・サーバーに接続する CLI アプリケーションは失敗します。問題を修正するには、IDS データ・サーバー上の GSKit ライブラリーを GSKit 7.0.4.20 以降にアップグレードします。

クライアントの SSL サポートを構成する前に、以下のステップを実行します。

- クライアントとサーバーの両方が同一の物理的なコンピューターにある場合、GSKit は自動的に DB2 サーバーとともにインストールされるので、GSKit をインストールする必要はありません。

  バージョン 9.7 フィックスパック 1 から、 64 ビット版 DB2 サーバー、32 ビット GSKit ライブラリーはインストール時に自動で組み込まれます。これらのライブラリーを Linux および UNIX オペレーティング・システムで使用するには、 **LD_LIBRARY_PATH**、**LIBPATH**、または **SHLIB_PATH** 環境変数を正しく設定する必要があります。 Windows オペレーティング・システムの場合は、 **PATH** 環境変数を、以下の表に示すように正しく設定してください。

アプリケーション	オペレーティング・システム	GSKit ライブラリーの場所	環境変数の設定
32 ビット	Linux および UNIX 64 ビット	*$INSTHOME*/sqllib/lib32/gskit	*$INSTHOME*/sqllib/lib32/gskit を *LD_LIBRARY_PATH*、*LIBPATH*、または *SHLIB_PATH* 環境変数に含めます。
64 ビット	Linux および UNIX 64 ビット	*$INSTHOME*/sqllib/lib64/gskit	*$INSTHOME*/sqllib/lib64/gskit を *LD_LIBRARY_PATH*、*LIBPATH*、または *SHLIB_PATH* 環境変数に含めます。
32 ビット	Windows 64 ビット	C:¥Program Files (x86)¥IBM¥GSK8¥lib	C:¥Program Files (x86)¥IBM¥GSK8¥lib を *PATH* 環境変数に含めます。
64 ビット	Windows 64 ビット	C:¥Program Files¥IBM¥GSK8¥lib64	C:¥Program Files¥IBM¥GSK8¥lib64 を *PATH* 環境変数に含めます。

SSL 通信は、常に FIPS モードになります。

Windows 以外のプラットフォームでは、DB2 データベース・マネージャーは GSKit をローカルにインストールし、GSKit ライブラリーは特定のインスタンスの sqllib/lib/gskit または sqllib/lib64/gskit にあります。GSKit の別のコピーをグローバルな場所にインストールする必要はありません。GSKit のグローバル・コピーが存在する場合は、グローバルな GSKit のバージョンがローカルの GSKit のバージョンと同じになるようにしてください。

- クライアントを別個のコンピューターにインストールする場合、"C" ベースのクライアントであるなら、クライアントがサーバーとの通信に SSL を使用する場合に GSKit をインストールする必要があります。GSKit ライブラリーは IBM DB2 Support Files for SSL Functionality DVD からインストールできます。あるいは、パスポート・アドバンテージでダウンロードしたイメージからインストールすることもできます。
  - Windows では **PATH** 環境変数に、Linux と UNIX では **LIBPATH**、**SHLIB_PATH**、**LD_LIBRARY_PATH** のいずれかの環境変数に、IBM Global Security Kit (GSKit) ライブラリーのパスが設定されていることを確認します。例えば、Windows では、以下のように GSKit の bin および lib ディレクトリーを **PATH** 環境変数に追加します。

    ```
    set PATH="C:¥Program Files¥ibm¥gsk8¥bin";%PATH%
    set PATH="C:¥Program Files¥ibm¥gsk8¥lib";%PATH%
    ```

GSKit ツールの資料: GSKCapiCmd

GSKit ツール GSKCapiCmd についての詳細は、「*GSKCapiCmd User's Guide*」を参照してください。これは、ftp://ftp.software.ibm.com/software/webserver/appserv/library/v80/GSK_CapiCmd_UserGuide.pdf から入手できます。

このタスクについて

SSL 通信は、常に FIPS モードになります。

手順

DB2 クライアントで SSL サポートを構成するには、以下のようにします。

1. クライアント上のサーバーのデジタル証明書の署名者証明書を取得します。サーバー証明書としては、自己署名証明書、または認証局 (CA) によって署名された証明書を使用することができます。
   - サーバー証明書として自己署名証明書を使用する場合、この署名者証明書をサーバー・コンピューター上でファイルに抽出してから、そのサーバーに SSL 接続を確立することになるクライアントを実行するコンピューターに配布する必要があります。証明書をファイルに抽出する方法については、70 ページの『DB2 インスタンスの Secure Sockets Layer (SSL) サポートの構成』を参照してください。
   - サーバー証明書が既知の CA によって署名されている場合、クライアントの鍵データベースには、ご使用のサーバー証明書に署名した CA 証明書が既に含まれている場合があります。含まれていない場合には、CA 証明書を取得する必要があります。通常、CA の Web サイトにアクセスして取得します。
2. DB2 クライアント・コンピューターでは、GSKCapiCmd ツールを使用して、CMS タイプの鍵データベースを作成します。 GSKCapiCmd ツールは非 Java ベースのコマンド行ツールです (このツールを使用するために、システムに Java をインストールする必要はありません)。

   「*GSKCapiCmd User's Guide*」に説明されているとおり、**gskcapicmd** コマンドを使用して、GSKCapiCmd を呼び出します。このコマンドのパスは Linux および UNIX オペレーティング・システムでは `sqllib/gskit/bin` で、32 ビットと 64 ビットの Windows オペレーティング・システムではどちらも `C:¥Program Files¥IBM¥GSK8¥bin` です。 (64 ビット・オペレーティング・システムでは、32 ビット GSKit 実行可能ファイルおよびライブラリーもあります。この場合、このコマンドのパスは `C:¥Program Files (x86)¥IBM¥GSK8¥bin` です。)

   例えば、以下のコマンドを使用すると、`mydbclient.kdb` という鍵データベースと、`mydbclient.sth` という stash ファイルが作成されます。

   ```
   gsk8capicmd_64 -keydb -create -db "mydbclient.kdb" -pw "myClientPassw0rdpw0"
         -stash
   ```

   -stash オプションは、鍵データベースと同じパスに、ファイル拡張子 **.sth** で、stash ファイルを作成します。接続時に、GSKit は stash ファイルを使用して、鍵データベースへのパスワードを取得します。
3. 署名者証明書をクライアントの鍵データベースに追加します。

   例えば、以下の **gsk8capicmd** コマンドを使用すると、証明書がファイル `mydbserver.arm` から `mydbclient.kdb` という鍵データベースにインポートされます。

   ```
   gsk8capicmd_64 -cert -add -db "mydbclient.kdb" -pw "myClientPassw0rdpw0"
         -label "dbselfsigned" -file "mydbserver.arm" -format ascii -fips
   ```

4. お使いのクライアントに適用できる例に示されているように、クライアント・アプリケーションに適切な接続ストリングまたは構成パラメーターを設定します。

例

CLP および組み込み SQL クライアント

CLP クライアントおよび組み込み SQL クライアントは、**CATALOG TCPIP NODE** コマンドを使用して、ノード・カタログに追加されたリモート・ホスト上のデータベースに接続できます。**SECURITY** キーワードを "SSL" に設定して **CATALOG TCPIP NODE** コマンドを発行し、その接続のために SSL を指定します。

以下の例では、CLP クライアントが SSL 接続を使用して接続できるように、ノードおよびデータベースをカタログする方法を示します。

まず、以下のようにノードおよびデータベースをカタログして、クライアント・アプリケーションがそれらへの SSL 接続を確立できるようにします。

```
catalog TCPIP NODE mynode  REMOTE  127.0.0.1  SERVER 50001  SECURITY SSL

catalog DATABASE sample  AS myssldb  AT NODE mynode  AUTHENTICATION SERVER
```

次に、**ssl_clnt_keydb** および **ssl_clnt_stash** 構成パラメーターを使用して、クライアントの鍵データベースおよび stash ファイルを指定します。以下のように、**ssl_clnt_keydb** 構成パラメーターを鍵データベース・ファイル (.kdb) の絶対パスに設定し、**ssl_clnt_stash** 構成パラメーターを stash ファイルの絶対パスに設定します。

```
db2 update dbm cfg using
      SSL_CLNT_KEYDB /home/test1/sqllib/security/keystore/clientkey.kdb
      SSL_CLNT_STASH /home/test1/sqllib/security/keystore/clientstore.sth
```

ssl_clnt_keydb または **ssl_clnt_stash** のいずれかの構成パラメーターが NULL (未設定) の場合、接続は失敗して、エラー SQL10013N がトークン GSKit Error: *GSKit_return_code* とともに返されます。

次に、以下のように CLP クライアントからサーバーに接続します。

```
db2 connect to myssldb user user1 using password
```

別の方法として、組み込み SQL アプリケーションは次のステートメントを使用して接続することができます。

```
Strcpy(dbAlias,"myssldb");
 EXEC SQL CONNECT TO :dbAlias USER :user USING :pswd;
```

CLI/ODBC クライアント・アプリケーション

CLI アプリケーションを実行している環境に応じて、接続ストリング・パラメーター (ssl_client_keystoredb および ssl_client_keystash) または DB2 構成パラメーター (**ssl_clnt_keydb** および **ssl_clnt_stash**) を使用して、クライアントの鍵データベースのパスおよび stash ファイルのパスを指定します。

- IBM Data Server Driver for ODBC and CLI を使用する場合、この例に示されているように、接続ストリング・パラメーターを使用します。

  SECURITY=SSL キーワードが含まれている接続ストリングを使用して、SQLDriverConnect を呼び出します。以下に例を示します。

```
"Database=sampledb; Protocol=tcpip; Hostname= myhost; Servicename=50001;
Security=ssl; Ssl_client_keystoredb=/home/test1/keystore/clientstore.kdb;
Ssl_client_keystash=/home/test1/keystore/clientstore.sth;"
```

この場合、Security=ssl が指定されているので、ssl_client_keystoredb および ssl_client_keystash 接続ストリング・パラメーターが設定されなければなりません。そうでない場合には、接続が失敗します。

- IBM Data Server Client または IBM Data Server Runtime Client を使用する場合、接続ストリング・パラメーターまたは DB2 構成パラメーターを使用して、クライアントの鍵データベースのパスおよび stash ファイルのパスを設定できます。**ssl_client_keystoredb** および **ssl_client_keystash** 接続ストリング・パラメーターが設定されると、それらは **ssl_clnt_keydb** または **ssl_clnt_stash** 構成パラメーターによって設定された値をオーバーライドします。

  この例では、以下のように db2cli.ini ファイルを使用して、接続ストリング・パラメーターを設定します。

  ```
  [sampledb]
  Database=sampledb
  Protocol=tcpip
  Hostname=myhost
  Servicename=50001
  Security=ssl
  SSL_client_keystoredb=/home/test1/keystore/clientstore.kdb
  SSL_client_keystash=/home/test1/keystore/clientstore.sth
  ```

  この例では、**FileDSN** CLI/ODBC キーワードを使用して、データベース接続情報が含まれている DSN ファイルを識別します。このファイルは接続ストリング・パラメーターを設定します。例えば、DSN ファイルは次のようになる場合があります。

  ```
  [ODBC]
  DRIVER=IBM DB2 ODBC DRIVER - DB2COPY1
  UID=user1
  AUTHENTICATION=SERVER
  PORT=50001
  HOSTNAME=myhost
  PROTOCOL=TCPIP
  DATABASE=SAMPLEDB
  SECURITY=SSL
  SSL_client_keystoredb=/home/test1/keystore/clientstore.kdb
  SSL_client_keystash=/home/test1/keystore/clientstore.sth
  ```

  これらの場合、Security=ssl が指定されているので、**ssl_client_keystoredb** および **ssl_client_keystash** 接続ストリング・パラメーターが設定されておらず、**ssl_clnt_keydb** および **ssl_clnt_stash** 構成パラメーターも設定されていない場合、接続は失敗します。

証明書ベースの認証

DB2 バージョン 9.7 フィックスパック 6 以降、以下の構文に示すとおり、新しい認証タイプ「CERTIFICATE」が db2dsdriver.cfg 認証パラメーターに導入されています。

```
<parameter name="Authentication"
value="CERTIFICATE | SERVER | SERVER_ENCRYPT | SERVER_ENCRYPT_AES | DATA_ENCRYPT |  KERBEROS | GSSPLUGIN"/>
```

証明書ベースの認証では、データベース・クライアント上でデータベース・パスワードを提供せずに SSL クライアント認証を使用することができます。認証情報を提供するように証明書ベースの認証を構成した場合は、他の方法 (db2dsdriver.cfg 構成ファイル、db2cli.ini 構成ファイル、接続ストリングなど) でパスワードを指定することはできません。認証パラメーターでラベルを指定する必要があるので、新しいデータ・サーバー・ドライバー構成パラメーター **SSLClientLabel** も導入されています。 CERTIFICATE を指定する場合には、CLI 構成ファイル db2cli.ini またはデータ・サーバー・ドライバー構成ファイル db2dsdriver.cfg 内で新しいラベル・パラメーター **SSLCLientLabel** も指定しなければなりません。

DB2 バージョン 9.7 フィックスパック 6 以降、**SSLClientKeystoredb** キーワードで指定された鍵ストア・データベースのパスワードを設定する新しいキーワード **SSLClientKeyStoreDBPassword** が導入されました。構成パラメーター **SSLClientKeystash** と **SSLClientKeyStoreDBPassword** は相互に排他的です。 CLI 構成ファイルかデータ・サーバー・ドライバー構成ファイルに **SSLClientKeystash** 構成パラメーターと **SSLClientKeyStoreDBPassword** 構成パラメーターを両方とも指定すると、エラー CLI0220E が返されます。そのため、証明書ベースの認証が正常に実行されるように、これらのキーワードの両方ではなく一方のみを指定することをお勧めします。

DB2 .Net Data Provider アプリケーション

DB2 .Net Data Provider アプリケーションは、接続ストリング・パラメーター SSLClientKeystoredb および SSLClientKeystash を定義することにより、クライアントの鍵データベースのパスおよび stash ファイルのパスを指定して、SSL 接続をデータベースに確立することができます。接続ストリングには、Security=SSL も含まれている必要があります。以下に例を示します。

```
String connectString = "Server=myhost:50001;Database=sampledb;Security=ssl;
SSLClientKeystoredb=/home/test1/keystore/clientstore.kdb;
SSLClientKeystash=/home/test1/keystore/clientstore.sth";
```

次に、以下の C# コード・フラグメントに示されているように、データベースに接続するために、この **connectString** を **DB2Connection** コンストラクターに渡し、**DB2Connection** オブジェクトの **Open** メソッドを使用して **connectString** で特定されるデータベースに接続します。

```
DB2Connection conn = new DB2Connection(connectString);
Conn.Open();
Return conn;
```

SSLClientKeystoredb または **SSLClientKeystash** いずれかの接続ストリング・パラメーターが NULL (未設定) の場合、接続は失敗して、エラー SQL10013N がトークン GSKit Error: *GSKit_return_code* とともに返されます。

Secure Sockets Layer (SSL)

DB2 データベース・システムは、Secure Sockets Layer (SSL) およびその後継の Transport Layer Security (TLS) をサポートして、クライアントがサーバーを認証できるようにし、暗号化を使用してクライアントとサーバー間での専用通信を提供します。認証は、デジタル証明書の交換によって実行されます。

注: このトピックで SSL について言及しているときには、特に注記がない限り、同じ情報が TLS に適用されます。

暗号化しない場合、情報のパケットは、アクセス可能なすべてのユーザーに完全に見える状態でネットワークを通過することになります。SSL を使用して、TCP/IP を使用するすべてのネットワーク上で転送中のデータを保護できます (SSL 接続は保護された TCP/IP 接続と考えることができます)。

クライアントおよびサーバーは、「SSL ハンドシェーク」を実行することにより、セキュア SSL 接続を確立できます。

SSL ハンドシェークの概要

SSL ハンドシェーク中は、公開鍵アルゴリズム、通常は RSA が使用され、クライアントとサーバー間でデジタル署名および暗号鍵が安全に交換されます。この ID および鍵情報は、クライアントとサーバー間のセッションのためのセキュア接続を確立するのに使用されます。セキュア・セッションが確立された後、クライアントとサーバー間のデータ伝送は、AES などの対称アルゴリズムを使用して暗号化されます。

クライアントとサーバーは、SSL ハンドシェーク中に以下のステップを実行します。

1. クライアントは、SSL 接続を要求し、自分がサポートする暗号スイートをリストします。
2. サーバーは、選択した暗号スイートを使用して、応答します。
3. サーバーは、デジタル証明書をクライアントに送信します。
4. クライアントは、認証目的でサーバーの証明書の妥当性を検証します。サーバーの証明書を発行した信頼できる認証局に確認するか、それ自身の鍵データベースを確認することにより、これを行うことができます。
5. クライアントとサーバーは、セッション鍵およびメッセージ認証コード (MAC) を安全にネゴシエーションします。
6. クライアントとサーバーは、選択された鍵および MAC を使用して安全に情報を交換します。

注: DB2 データベース・システムは SSL ハンドシェーク中の (オプションの) クライアントの認証をサポートしていません。

DB2 認証を使った SSL 暗号化の使用

SSL 暗号化は、KERBEROS または SERVER などの既存の DB2 認証方式すべてと組み合わせて使用することができます。これは、普通と同じように、DBM 構成パラメーターのインスタンスの認証タイプを選択した認証方式に設定することにより行います。

デジタル証明書と認証局

デジタル証明書は、認証局と呼ばれる信頼できる機関によって発行され、クライアントまたはサーバーなどのエンティティーの ID を検証します。

デジタル証明書は、次の 2 つの目的を果たします。所有者の ID を検証し、所有者の公開鍵を有効にします。それは、有効期限 (その後は認証局 (CA) によって保証されなくなります) を指定して発行されます。

デジタル証明書を取得するには、ユーザー選択の CA (Verisign や RSA など) に要求を送信します。要求には、識別名、公開鍵、および署名が含まれています。識別名 (DN) とは、証明書を申し込む各ユーザーまたはホストの固有の ID のことです。CA は、公開鍵を使用して署名を確認し、ご使用の ID の検証を何らかのレベル (これは CA によって異なる) で実行します。検証の後、CA は識別名、公開鍵、CA の識別名、および認証局の署名が含まれる署名付きデジタル証明書を送信します。この署名付き証明書を鍵データベースに保管します。

この証明書を受信側に送信するとき、受信側は ID を検証するために、以下の 2 つのステップを実行します。

1. 証明書に付属している公開鍵を使用して、デジタル署名を確認します。
2. 証明書を発行した CA が正当で信頼できるかどうかを検証します。これを行うには、受信側には CA の公開鍵が必要です。受信側は鍵データベースに CA の公開鍵の確実なコピーを既に保持している可能性がありますが、保持していない場合、受信側は追加のデジタル証明書を獲得して、CA の公開鍵を取得する必要があります。この証明書も他の CA のデジタル証明書に依存する場合があります。各々が次の CA の妥当性に依存する、複数の CA によって発行された証明書の階層がある場合もあります。ただし、受信側は最終的には、*ルート CA* の公開鍵が必要です。ルート CA とは、階層の先頭にある CA のことです。ルート CA のデジタル証明書の妥当性を信頼するには、公開鍵ユーザーはセキュアな方法 (認証済みサーバーからのダウンロード、信頼できるソースから受け取るプリロード済みソフトウェアの使用、安全に提供されたディスケット上など) で、デジタル証明書を受け取ることが必要です。

デジタル証明書を受信側に送信するアプリケーションの多くは、自分の証明書だけでなく、ルート CA 証明書までの証明書の階層を検証するのに必要な CA のデジタル証明書すべてを送信します。

デジタル証明書が完全に信頼できるように、デジタル証明書の所有者は秘密鍵を注意深く保護する (例えば、コンピューターのハード・ディスク上でそれを暗号化することにより) 必要があります。秘密鍵の暗号漏えいが発生した場合、デジタル証明書は偽の所有者によって誤用される可能性があります。

テスト目的で、自己署名デジタル証明書を使用することができます。自己署名デジタル証明書には、識別名、公開鍵、および署名が含まれています。

公開鍵の暗号方式

SSL は公開鍵アルゴリズムを使用して、認証のための暗号鍵情報とデジタル証明書情報を交換します。公開鍵暗号方式 (非対称暗号方式とも言います) は 2 つの異なる暗号鍵を使用します。公開鍵 (データ暗号化用) とそれと関連付けられた秘密鍵 (復号用) です。

これに対し、対称鍵暗号方式はただ 1 つの鍵を使用し、これが機密保護機能のある通信に関係するすべての参加者によって共有されます。この秘密鍵は、情報の暗号化と復号の両方に使用されます。この鍵は全関係者に安全に配布して保管しなけれ

ばなりませんが、これを保証するのは簡単ではありません。公開鍵暗号方式の場合、公開鍵は秘密ではありませんが、それによって暗号化されたメッセージを復号するにはそれと関連付けられた秘密鍵が必要です。この秘密鍵は、例えば鍵データベースに安全に保管するか、コンピューターのハード・ディスクに暗号化されていなければなりません。

公開鍵アルゴリズムだけでは機密保護機能のある通信を保証するものとならず、通信している相手がだれであるかが検証される必要もあります。この認証を行うため、SSL はデジタル証明書を使用します。だれかにデジタル証明書を送る時に、その証明書は公開鍵も一緒に提供します。証明書にデジタル署名するのに秘密鍵を使用しているので、通信の受信側は公開鍵を使って書名を検証することができます。デジタル証明書そのものの妥当性は、それを発行した認証局 (CA) によって保証されます。

サポートされる暗号スイート

SSL ハンドシェーク中に、クライアントとサーバーはデータの交換にどの暗号スイートを使用するかをネゴシエーションします。暗号スイートというのは、認証、暗号化、およびデータ保全性を提供するのに使用されるアルゴリズムのセットです。

DB2 データベース・システムは、FIPS モードで実行される GSKit を使用して SSL サポートを提供します。 GSKit は以下の暗号スイートをサポートします。

- TLS_RSA_WITH_AES_256_CBC_SHA
- TLS_RSA_WITH_AES_128_CBC_SHA
- TLS_RSA_WITH_3DES_EDE_CBC_SHA

それぞれの暗号スイートの名前は、認証、暗号化、およびデータ保全性検査に使用するアルゴリズムを指定します。例えば、暗号スイート TLS_RSA_WITH_AES_256_CBC_SHA は、認証のために RSA、暗号化アルゴリズムに AES 256 ビットと CBC、データ保全性のためのハッシュ関数に SHA-1 を使用します。

SSL ハンドシェーク中に、DB2 データベース・システムは、クライアントとサーバーの両方がサポートする最強の暗号スイートを自動的に選択します。サーバーが 1 つ以上の特定の暗号スイートだけを受け入れるようにする場合は、**ssl_cipherspecs** 構成パラメーターを以下の値のいずれかに設定することができます。

- TLS_RSA_WITH_AES_256_CBC_SHA
- TLS_RSA_WITH_AES_128_CBC_SHA
- TLS_RSA_WITH_3DES_EDE_CBC_SHA
- 3 つの値の任意の組み合わせ。複数の値を設定するには、値をそれぞれコンマで区切りますが、間にスペースは置かないでください。
- Null。この場合、使用可能な最強のアルゴリズムが自動的に選択されます。

選択される暗号スイートに優先順位付けをすることはできません。**ssl_cipherspecs** 構成パラメーターを設定する場合、DB2 データベース・システムは使用可能な最強の暗号スイートを選択します。この選択は、**ssl_cipherspecs** の設定時に暗号スイートを指定した順番と関係ありません。

バンドル・ライブラリーとプロセス規則

DB2 for Linux, UNIX, and Windows に GSKit を必要とするベンダー・ソフトウェアがバンドルされている場合、または GSKit を必要とするベンダー・ソフトウェアに DB2 for Linux, UNIX, and Windows がバンドルされている場合には、特定の規則に従う必要があります。

ライブラリー規則

DB2 for Linux, UNIX, and Windows に GSKit を必要とするベンダー・ソフトウェアがバンドルされている場合、そのベンダー・ソフトウェアは、DB2 for Linux, UNIX, and Windows がリンクするライブラリーを提供します。それらのライブラリーは、特定の規則に従う必要があります。この規則は、ライブラリー規則と呼ばれます。

ライブラリー規則: 短縮名を使用

GSKit ライブラリーを動的にロードする場合、呼び出し元は、ローダー関数に、パスではなく GSKit ライブラリーの基本ファイル名のみを渡す必要があります。

例えば、`dlopen("libgsk8ssl_64.so", RTLD_NOW | RTLD_GLOBAL)` が正しく、`dlopen("/usr/opt/ibm/gsk8_64/lib/libgsk8ssl_64.so", RTLD_NOW | RTLD_GLOBAL)` は誤りです。

プロセス規則

GSKit を必要とするベンダー・ソフトウェアに DB2 for Linux, UNIX, and Windows がバンドルされている場合、ベンダー・ソフトウェアは IBM Data Server Clientとリンクします。ベンダー・ソフトウェアは、特定の規則に従う必要があります。この規則は、プロセス規則と呼ばれます。

プロセス規則: 環境検索パスのセットアップ

プロセスは、GSKit ライブラリーを検索するための環境検索パスをセットアップする必要があります。プロセスは、このセットアップを行うことで、組み込まれているライブラリーが同じ場所から GSKit ライブラリーをロードできるようにする必要があります。

AIX では、プロセスは、実行可能ファイルの LIBPATH または RPATH を GSKit ライブラリーのパスに設定することができます。 **setuid** および **setgid** のケースでは、プロセスは、db2chglibpath を使用して、GSKit の検索パスを実行可能ファイルの RPATH に組み込むことができます。そうすることによってのみ、その場所にある GSKit ライブラリーを使用することができます。 Linux、Sun、および HP-UX では、プロセスは、LD_LIBRARY_PATH を GSKit ライブラリーのパスに設定することができます。 **setuid** および **setgid** のケースでは、プロセスは、db2chglibpath を使用して、GSKit の検索パスを IBM Data Server Client・ライブラリーの RPATH に組み込むことができます。そうすることによってのみ、その場所にある GSKit ライブラリーを使用することができます。例えば、プロセスがサーバー・インスタンスでグローバル GSKit を使用する必要がある場合、あるいはクライアント・インスタンスまたはサーバー・インスタンスで独自のローカル GSKit を使用する必要がある場合には、db2chglibpath を使用して RPATH を変更することができます。

シンボリック・リンクを使用する方法と制約事項

DB2 for Linux, UNIX, and Windows を UNIX プラットフォームおよび Linux プラットフォームにインストールすると、ローカル GSKit ライブラリーもインストールされます。それらのライブラリーは、<db2_install_path>/lib64/gskit_db2 または <db2_install_path>/lib32/gskit_db2 にあります。

他の IBM 製品のインストール中に GSKit ライブラリーの別のコピーがインストールされる場合もあります。これらのライブラリーは、製品によって、ローカル GSKit ライブラリーの場合とグローバル GSKit ライブラリーの場合があります。DB2 for Linux, UNIX, and Windows と、GSKit ライブラリーが含まれる別の IBM 製品の両方が同じマシンにインストールされた場合、いくつかの相互運用性の問題が発生する可能性があります。これらの相互運用性の問題は、GSKit が 1 つのプロセスで 1 つの GSKit ソースのライブラリーしか許可しないことが原因で発生します。相互運用性の問題は、予測不能の動作やランタイム・エラーを引き起こす可能性があります。

同じマシン上に複数の GSKit がインストールされている場合に、確実に単一の GSKit ライブラリー・ソースが使用されるようにするには、シンボリック・リンクを使用する方法があります。 DB2 for Linux, UNIX, and Windows の初期インストール中に、インストーラーは、<db2_install_path>/lib64/gskit または <db2_install_path>/lib32/gskit から <db2_install_path>/lib64/gskit_db2 または <db2_install_path>/lib32/gskit_db2 へのシンボリック・リンクを作成します。これがデフォルトのロケーションで、このロケーションから GSKit ライブラリーがロードされます。 DB2 for Linux, UNIX, and Windows がバンドルされている製品が、シンボリック・リンクを上述したデフォルトのディレクトリーから別の GSKit コピーのライブラリー・ディレクトリーに変更する場合は、新しくインストールされた GSKit が、同じレベルかより新しいレベルのものでなければなりません。この制約事項は、そのライブラリーがグローバルであってもローカルであっても適用されます。 DB2 for Linux, UNIX, and Windows のアップグレードまたは更新では、シンボリック・リンクは保持されます。新しくインストールされたコピーにデフォルトのロケーションを指すシンボリック・リンクがある場合は、古いインストール・コピーに関連付けられているシンボリック・リンクが保持されます。新しくインストールされたコピーにデフォルトのロケーションを指していないシンボリック・リンクがある場合、新しくインストールされたコピーでは新しいインストール・コピーに関連付けられているシンボリック・リンクが使用されます。シンボリック・リンク <db2_install_path>/lib64/gskit または <db2_install_path>/lib32/gskit は DB2 for Linux, UNIX, and Windows インストール・コピーのパスにあることから、いくつかの制約があります。例えば、いずれかの DB2 コピー用に作成されたインスタンスが複数存在する場合、シンボリック・リンクを変更するとすべてのインスタンスに影響します。

DB2 for Linux, UNIX, and Windows に組み込まれている GSKit のバージョンは、8.0.14.14 (Solaris x64 では 8.0.15.1) です。

例

DB2for Linux, UNIX, and Windows には、LDAP クライアントがバンドルされています。 DB2 for Linux, UNIX, and Windows のプロセスは、プロセス規則に従います。プロセス規則に従うために、RPATH で指定される環境検索パスは、その

GSKit のローカル・コピーに設定されます。 LDAP クライアント・ライブラリーは、その同じ場所から GSKit ライブラリーをロードします。ライブラリー規則に従う LDAP クライアント・ライブラリーは、GSKIT_LOCAL_INSTALL_MODE が設定されていれば、GSKit ライブラリーをその基本ファイル名でロードします。

LDAP サーバーには DB2 for Linux, UNIX, and Windows がバンドルされています。 LDAP プロセスはプロセス規則に従います。環境検索パスは、GSKit のグローバル・コピーに設定され、IBM Data Server Client・ライブラリーはその同じ場所から GSKit ライブラリーをロードします。ライブラリー規則に従う IBM Data Server Client・ライブラリーは、GSKit ライブラリーをその基本ファイル名でロードします。

GSKit 戻りコード

一部の DB2 データベース・マネージャー・メッセージは、IBM Global Security Kit (GSKit) からの戻りコードを表示する場合があります。

一般 GSKit 戻りコード

表 2. GSKit の一般戻りコード

戻りコード (16 進)	戻りコード (10 進)	定数	説明
0x00000000	0	GSK_OK	タスクが正常に完了しました。正常に完了すると、すべての関数呼び出しはこのメッセージ発行します。
0x00000001	1	GSK_INVALID_HANDLE	環境つまり SSL ハンドルが無効です。指定されたハンドルは、成功した open 関数呼び出しの結果ではありませんでした。
0x00000002	2	GSK_API_NOT_AVAILABLE	ダイナミック・リンク・ライブラリー (DLL) はアンロードされてしまい、使用できません。 (Windows のみ。)
0x00000003	3	GSK_INTERNAL_ERROR	内部エラー。このエラーはサービスに報告してください。
0x00000004	4	GSK_INSUFFICIENT_STORAGE	操作を実行するのに十分なメモリーが使用できません。
0x00000005	5	GSK_INVALID_STATE	ハンドルで初期化操作を 2 度行ったなど、ハンドルが無効な操作状態にあります。
0x00000006	6	GSK_KEY_LABEL_NOT_FOUND	指定された鍵ラベルが鍵ファイルにありません。
0x00000007	7	GSK_CERTIFICATE_NOT_AVAILABLE	パートナーから証明書を受け取っていません。
0x00000008	8	GSK_ERROR_CERT_VALIDATION	証明書妥当性検査エラー。
0x00000009	9	GSK_ERROR_CRYPTO	暗号化処理でエラー。
0x0000000a	10	GSK_ERROR_ASN	証明書の ASN フィールドの妥当性検査でエラー。

表 2. GSKit の一般戻りコード (続き)

戻りコード (16 進)	戻りコード (10 進)	定数	説明
0x0000000b	11	GSK_ERROR_LDAP	LDAP サーバーへの接続でエラー。
0x0000000c	12	GSK_ERROR_UNKNOWN_ERROR	内部エラー。このエラーはサービスに報告してください。
0x00000065	101	GSK_OPEN_CIPHER_ERROR	内部エラー。このエラーはサービスに報告してください。
0x00000066	102	GSK_KEYFILE_IO_ERROR	鍵ファイルの読み取りで入出力エラー。
0x00000067	103	GSK_KEYFILE_INVALID_FORMAT	鍵ファイルの内部形式が無効です。鍵ファイルを作成し直してください。
0x00000068	104	GSK_KEYFILE_DUPLICATE_KEY	鍵ファイルに同じキーの項目が 2 つあります。 iKeyman ユーティリティーを使用して重複キーを除去してください。
0x00000069	105	GSK_KEYFILE_DUPLICATE_LABEL	鍵ファイルに同じラベルを持つ項目が 2 つあります。 iKeyman ユーティリティーを使用して重複ラベルを除去してください。
0x0000006a	106	GSK_BAD_FORMAT_OR_ INVALID_PASSWORD	保全性検査に対して鍵ファイル・パスワードが使用されます。鍵ファイルが破損しているか、パスワード ID が間違っているかのいずれかです。
0x0000006b	107	GSK_KEYFILE_CERT_EXPIRED	鍵ファイルのデフォルト鍵の証明書の有効期限が切れています。 iKeyman ユーティリティーを使用して有効期限の切れた証明書を除去してください。
0x0000006c	108	GSK_ERROR_LOAD_GSKLIB	GSKit ダイナミック・リンク・ライブラリーの 1 つのロードでエラーが発生しました。 GSKit が正しくインストールされていることを確認してください。
0x0000006d	109	GSK_PENDING_CLOSE_ERROR	GSK_ENVIRONMENT_ CLOSE_OPTIONS が GSK_DELAYED_ ENVIRONMENT_CLOSE に設定されて gsk_environment_close() が呼び出された後で、 GSKit 環境で接続が試行されたことを示します。
0x000000c9	201	GSK_NO_KEYFILE_PASSWORD	パスワードも stash ファイル名も指定されていないため、鍵ファイルが初期化できませんでした。

表 2. *GSKit* の一般戻りコード (続き)

戻りコード (16 進)	戻りコード (10 進)	定数	説明
0x000000ca	202	GSK_KEYRING_OPEN_ERROR	鍵ファイルか Microsoft 証明書ストアが開けません。パスが間違って指定されたか、ファイル許可のためにファイルが開けないか、ファイル形式が間違っています。
0x000000cb	203	GSK_RSA_TEMP_KEY_PAIR	一時鍵ペアを生成できません。このエラーはサービスに報告してください。
0x000000cc	204	GSK_ERROR_LDAP_NO_SUCH_OBJECT	ユーザー名オブジェクトが指定されましたが、見つかりません。
0x000000cd	205	GSK_ERROR_LDAP_INVALID_ CREDENTIALS	LDAP 照会に使用されたパスワードが正しくありません。
0x000000ce	206	GSK_ERROR_BAD_INDEX	LDAP サーバーのフェイルオーバー・リストに対する索引が正しくありません。
0x000000cd	207	GSK_ERROR_FIPS_NOT_SUPPORTED	GSKit を FIPS モードにしようとして失敗しました。
0x0000012d	301	GSK_CLOSE_FAILED	GSKit 環境のクローズ要求が正しく処理されなかったことを示します。これはおそらく gsk_close_environment() 呼び出しの後で gsk_secure_socket*() コマンドが試行されたためです。
0x00000191	401	GSK_ERROR_BAD_DATE	システム日付が無効な値に設定されました。
0x00000192	402	GSK_ERROR_NO_CIPHERS	SSLV2 も SSLV3 も有効にされていません。
0x00000193	403	GSK_ERROR_NO_CERTIFICATE	パートナーから必須の証明書を受け取っていません。
0x00000194	404	GSK_ERROR_BAD_CERTIFICATE	受け取った証明書の形式が間違っています。
0x00000195	405	GSK_ERROR_UNSUPPORTED_ CERTIFICATE_TYPE	受け取った証明書のタイプはサポートされませんでした。
0x00000196	406	GSK_ERROR_IO	データ読み取りまたは書き込み操作で入出力エラーが発生しました。
0x00000197	407	GSK_ERROR_BAD_KEYFILE_LABEL	鍵ファイル内の指定されたラベルは見つかりませんでした。
0x00000198	408	GSK_ERROR_BAD_KEYFILE_ PASSWORD	指定された鍵ファイル・パスワードは正しくありません。この鍵ファイルは使用できませんでした。鍵ファイルも破損している可能性があります。

表 2. GSKit の一般戻りコード (続き)

戻りコード (16 進)	戻りコード (10 進)	定数	説明
0x00000199	409	GSK_ERROR_BAD_KEY_LEN_ FOR_EXPORT	制限された暗号化環境で、鍵サイズが長すぎてサポートされません。
0x0000019a	410	GSK_ERROR_BAD_MESSAGE	パートナーから誤った形式の SSL メッセージを受け取りました。
0x0000019b	411	GSK_ERROR_BAD_MAC	メッセージ認証コード (MAC) が正常に検査されませんでした。
0x0000019c	412	GSK_ERROR_UNSUPPORTED	サポートされない SSL プロトコルか、サポートされない証明書タイプです。
0x0000019d	413	GSK_ERROR_BAD_CERT_SIG	受け取った証明書に正しくない署名が入っていました。
0x0000019e	414	GSK_ERROR_BAD_CERT	パートナーから間違った形式の証明書を受け取りました。
0x0000019f	415	GSK_ERROR_BAD_PEER	パートナーから無効な SSL プロトコルを受け取りました。
0x000001a0	416	GSK_ERROR_PERMISSION_DENIED	この内部エラーはサービスに報告してください。
0x000001a1	417	GSK_ERROR_SELF_SIGNED	自己署名証明書が無効です。
0x000001a2	418	GSK_ERROR_NO_READ_FUNCTION	読み取り操作が失敗しました。この内部エラーはサービスに報告してください。
0x000001a3	419	GSK_ERROR_NO_WRITE_FUNCTION	書き込み操作が失敗しました。この内部エラーはサービスに報告してください。
0x000001a4	420	GSK_ERROR_SOCKET_CLOSED	プロトコルが完了する前にパートナーがソケットをクローズしました。
0x000001a5	421	GSK_ERROR_BAD_V2_CIPHER	指定された V2 暗号が無効です。
0x000001a6	422	GSK_ERROR_BAD_V3_CIPHER	指定された V3 暗号が無効です。
0x000001a7	423	GSK_ERROR_BAD_SEC_TYPE	この内部エラーはサービスに報告してください。
0x000001a8	424	GSK_ERROR_BAD_SEC_ TYPE_COMBINATION	この内部エラーはサービスに報告してください。
0x000001a9	425	GSK_ERROR_HANDLE_ CREATION_FAILED	ハンドルを作成できませんでした。この内部エラーはサービスに報告してください。
0x000001aa	426	GSK_ERROR_INITIALIZATION_ FAILED	初期化に失敗しました。この内部エラーはサービスに報告してください。
0x000001ab	427	GSK_ERROR_LDAP_NOT_AVAILABLE	証明書を妥当性検査している時に、指定された LDAP ディレクトリーにアクセスできません。

表 2. *GSKit* の一般戻りコード (続き)

戻りコード (16 進)	戻りコード (10 進)	定数	説明
0x000001ac	428	GSK_ERROR_NO_PRIVATE_KEY	指定された鍵に秘密鍵が入っていませんでした。
0x000001ad	429	GSK_ERROR_PKCS11_ LIBRARY_NOTLOADED	指定された PKCS11 共有ライブラリーをロードする試みが失敗しました。
0x000001ae	430	GSK_ERROR_PKCS11_TOKEN_ LABELMISMATCH	PKCS #11 ドライバーが、呼び出し元が指定したトークンを見つけることに失敗しました。
0x000001af	431	GSK_ERROR_PKCS11_TOKEN_ NOTPRESENT	PKCS #11 トークンがスロットにありません。
0x000001b0	432	GSK_ERROR_PKCS11_TOKEN_ BADPASSWORD	PKCS #11 トークンにアクセスするパスワード/ピンが無効です。
0x000001b1	433	GSK_ERROR_INVALID_V2_HEADER	受け取った SSL ヘッダーは正しい SSLV2 形式のヘッダーではありませんでした。
0x000001b2	434	GSK_CSP_OPEN_ERROR	ハードウェア・ベースの暗号サービス・プロバイダー (CSP) にアクセスできません。指定された CSP 名がシステムに登録されていないか、指定された CSP 名は登録されているが証明書ストアのオープンに失敗したかのいずれかです。
0x000001b3	435	GSK_CONFLICTING_ATTRIBUTE_ SETTING	PKCS11、CMS 鍵データベース、および Microsoft Crypto API の間で属性の設定が矛盾します。
0x000001b4	436	GSK_UNSUPPORTED_PLATFORM	要求された機能は、アプリケーションが実行されているプラットフォームでサポートされていません。例えば、Microsoft Crypto API は Windows 2000 以外のプラットフォームではサポートされません。
0x000001b5	437	GSK_ERROR_INCORRECT_ SESSION_TYPE	リセット・セッション・タイプのコールバック関数から誤った値が戻されました。 許されるのは、GSKit の GSK_SERVER_SESSION か GSK_SERVER_SESSION _WITH_CL_AUTH だけです。
0x000001f5	501	GSK_INVALID_BUFFER_SIZE	バッファー・サイズが負または 0 です。
0x000001f6	502	GSK_WOULD_BLOCK	非ブロッキング入出力で使用されました。

表 2. GSKit の一般戻りコード (続き)

戻りコード (16 進)	戻りコード (10 進)	定数	説明
0x00000259	601	GSK_ERROR_NOT_SSLV3	reset_cipher には SSLV3 が必要ですが、接続は SSLV2 を使用します。
0x0000025a	602	GSK_MISC_INVALID_ID	gsk_secure_soc_misc 関数呼び出しに無効な ID が指定されました。
0x000002bd	701	GSK_ATTRIBUTE_INVALID_ID	関数呼び出しの ID が無効です。これは、SSL 接続のためのハンドルを使用すべきときに、環境ハンドルを指定することによって引き起こされる場合もあります。
0x000002be	702	GSK_ATTRIBUTE_INVALID_LENGTH	属性の長さが負で、無効です。
0x000002bf	703	GSK_ATTRIBUTE_INVALID_ ENUMERATION	指定された列挙タイプに対して列挙値が無効です。
0x000002c0	704	GSK_ATTRIBUTE_INVALID_ SID_CACHE	セッション ID (SID) キャッシュ・ルーチンを置き換えるためのパラメーター・リストが無効。
0x000002c1	705	GSK_ATTRIBUTE_INVALID_ NUMERIC_VALUE	数値属性を設定する時に、指定された値は設定されている特定の属性に対して無効です。
0x000002c2	706	GSK_CONFLICTING_VALIDATION_ SETTING	追加の証明書妥当性検査で矛盾するパラメーターが設定されました。
0x000002c3	707	GSK_AES_UNSUPPORTED	指定した暗号に、実行システムでサポートされていない AES 暗号が含まれています。
0x000002c4	708	GSK_PEERID_LENGTH_ERROR	ピア ID の長さが間違っています。 16 バイト以下でなければなりません。
0x000002c5	709	GSK_CIPHER_INVALID_WHEN_ FIPS_MODE_OFF	指定された暗号は、FIPS モードがオフの時は指定できません。
0x000002c6	710	GSK_CIPHER_INVALID_WHEN_ FIPS_MODE_ON	FIPS モードで、FIPS で承認されない暗号が選択されました。
0x00000641	1601	GSK_TRACE_STARTED	トレースが正常に開始されました。
0x00000642	1602	GSK_TRACE_STOPPED	トレースが正常に停止されました。
0x00000643	1603	GSK_TRACE_NOT_STARTED	以前に開始されたトレース・ファイルはないので、停止できません。
0x00000644	1604	GSK_TRACE_ALREADY_STARTED	トレース・ファイルは既に開始されているので、もう一度開始することはできません。

表 2. GSKit の一般戻りコード (続き)

戻りコード (16 進)	戻りコード (10 進)	定数	説明
0x00000645	1605	GSK_TRACE_OPEN_FAILED	トレース・ファイルがオープンできません。 gsk_start_trace() の第 1 パラメーターは有効な絶対パス・ファイル名でなければなりません。

鍵管理戻りコード

表 3. 鍵管理戻りコード

戻りコード (16 進)	戻りコード (10 進)	定数
0x00000000	0	GSKKM_ERR_OK
0x00000000	0	GSKKM_ERR_SUCCESS
0x00000001	1	GSKKM_ERR_UNKNOWN
0x00000002	2	GSKKM_ERR_ASN
0x00000003	3	GSKKM_ERR_ASN_INITIALIZATION
0x00000004	4	GSKKM_ERR_ASN_PARAMETER
0x00000005	5	GSKKM_ERR_DATABASE
0x00000006	6	GSKKM_ERR_DATABASE_OPEN
0x00000007	7	GSKKM_ERR_DATABASE_RE_OPEN
0x00000008	8	GSKKM_ERR_DATABASE_CREATE
0x00000009	9	GSKKM_ERR_DATABASE_ALREADY_EXISTS
0x0000000a	10	GSKKM_ERR_DATABASE_DELETE
0x0000000b	11	GSKKM_ERR_DATABASE_NOT_OPENED
0x0000000c	12	GSKKM_ERR_DATABASE_READ
0x0000000d	13	GSKKM_ERR_DATABASE_WRITE
0x0000000e	14	GSKKM_ERR_DATABASE_VALIDATION
0x0000000f	15	GSKKM_ERR_DATABASE_INVALID_VERSION
0x00000010	16	GSKKM_ERR_DATABASE_INVALID_PASSWORD
0x00000011	17	GSKKM_ERR_DATABASE_INVALID_FILE_TYPE
0x00000012	18	GSKKM_ERR_DATABASE_CORRUPTION
0x00000013	19	GSKKM_ERR_DATABASE_PASSWORD_ CORRUPTION
0x00000014	20	GSKKM_ERR_DATABASE_KEY_INTEGRITY
0x00000015	21	GSKKM_ERR_DATABASE_DUPLICATE_KEY
0x00000016	22	GSKKM_ERR_DATABASE_DUPLICATE_ KEY_RECORD_ID
0x00000017	23	GSKKM_ERR_DATABASE_DUPLICATE_ KEY_LABEL
0x00000018	24	GSKKM_ERR_DATABASE_DUPLICATE_ KEY_SIGNATURE

表 3. 鍵管理戻りコード (続き)

戻りコード (16 進)	戻りコード (10 進)	定数
0x00000019	25	GSKKM_ERR_DATABASE_DUPLICATE_KEY_UNSIGNED_CERTIFICATE
0x0000001a	26	GSKKM_ERR_DATABASE_DUPLICATE_KEY_ISSUER_AND_SERIAL_NUMBER
0x0000001b	27	GSKKM_ERR_DATABASE_DUPLICATE_KEY_SUBJECT_PUBLIC_KEY_INFO
0x0000001c	28	GSKKM_ERR_DATABASE_DUPLICATE_KEY_UNSIGNED_CRL
0x0000001d	29	GSKKM_ERR_DATABASE_DUPLICATE_LABEL
0x0000001e	30	GSKKM_ERR_DATABASE_PASSWORD_ENCRYPTION
0x0000001f	31	GSKKM_ERR_DATABASE_LDAP
0x00000020	32	GSKKM_ERR_CRYPTO
0x00000021	33	GSKKM_ERR_CRYPTO_ENGINE
0x00000022	34	GSKKM_ERR_CRYPTO_ALGORITHM
0x00000023	35	GSKKM_ERR_CRYPTO_SIGN
0x00000024	36	GSKKM_ERR_CRYPTO_VERIFY
0x00000025	37	GSKKM_ERR_CRYPTO_DIGEST
0x00000026	38	GSKKM_ERR_CRYPTO_PARAMETER
0x00000027	39	GSKKM_ERR_CRYPTO_UNSUPPORTED_ALGORITHM
0x00000028	40	GSKKM_ERR_CRYPTO_INPUT_GREATER_THAN_MODULUS
0x00000029	41	GSKKM_ERR_CRYPTO_UNSUPPORTED_MODULUS_SIZE
0x0000002a	42	GSKKM_ERR_VALIDATION
0x0000002b	43	GSKKM_ERR_VALIDATION_KEY
0x0000002c	44	GSKKM_ERR_VALIDATION_DUPLICATE_EXTENSIONS
0x0000002d	45	GSKKM_ERR_VALIDATION_KEY_WRONG_VERSION
0x0000002e	46	GSKKM_ERR_VALIDATION_KEY_EXTENSIONS_REQUIRED
0x0000002f	47	GSKKM_ERR_VALIDATION_KEY_VALIDITY
0x00000030	48	GSKKM_ERR_VALIDATION_KEY_VALIDITY_PERIOD
0x00000031	49	GSKKM_ERR_VALIDATION_KEY_VALIDITY_PRIVATE_KEY_USAGE
0x00000032	50	GSKKM_ERR_VALIDATION_KEY_ISSUER_NOT_FOUND
0x00000033	51	GSKKM_ERR_VALIDATION_KEY_MISSING_REQUIRED_EXTENSIONS

表 3. 鍵管理戻りコード (続き)

戻りコード (16 進)	戻りコード (10 進)	定数
0x00000034	52	GSKKM_ERR_VALIDATION_KEY_BASIC_ CONSTRAINTS
0x00000035	53	GSKKM_ERR_VALIDATION_KEY_SIGNATURE
0x00000036	54	GSKKM_ERR_VALIDATION_KEY_ROOT_KEY_ NOT_TRUSTED
0x00000037	55	GSKKM_ERR_VALIDATION_KEY_IS_REVOKED
0x00000038	56	GSKKM_ERR_VALIDATION_KEY_AUTHORITY_ KEY_IDENTIFIER
0x00000039	57	GSKKM_ERR_VALIDATION_KEY_PRIVATE_KEY_ USAGE_PERIOD
0x0000003a	58	GSKKM_ERR_VALIDATION_SUBJECT_ ALTERNATIVE_NAME
0x0000003b	59	GSKKM_ERR_VALIDATION_ISSUER_ ALTERNATIVE_NAME
0x0000003c	60	GSKKM_ERR_VALIDATION_KEY_USAGE
0x0000003d	61	GSKKM_ERR_VALIDATION_KEY_ UNKNOWN_CRITICAL_EXTENSION
0x0000003e	62	GSKKM_ERR_VALIDATION_KEY_PAIR
0x0000003f	63	GSKKM_ERR_VALIDATION_CRL
0x00000040	64	GSKKM_ERR_MUTEX
0x00000041	65	GSKKM_ERR_PARAMETER
0x00000042	66	GSKKM_ERR_NULL_PARAMETER
0x00000043	67	GSKKM_ERR_NUMBER_SIZE
0x00000044	68	GSKKM_ERR_OLD_PASSWORD
0x00000045	69	GSKKM_ERR_NEW_PASSWORD
0x00000046	70	GSKKM_ERR_PASSWORD_EXPIRATION_TIME
0x00000047	71	GSKKM_ERR_THREAD
0x00000048	72	GSKKM_ERR_THREAD_CREATE
0x00000049	73	GSKKM_ERR_THREAD_WAIT_FOR_EXIT
0x0000004a	74	GSKKM_ERR_IO
0x0000004b	75	GSKKM_ERR_LOAD
0x0000004c	76	GSKKM_ERR_PKCS11
0x0000004d	77	GSKKM_ERR_NOT_INITIALIZED
0x0000004e	78	GSKKM_ERR_DB_TABLE_CORRUPTED
0x0000004f	79	GSKKM_ERR_MEMORY_ALLOCATE
0x00000050	80	GSKKM_ERR_UNSUPPORTED_OPTION
0x00000051	81	GSKKM_ERR_GET_TIME
0x00000052	82	GSKKM_ERR_CREATE_MUTEX
0x00000053	83	GSKKM_ERR_CMDCAT_OPEN
0x00000054	84	GSKKM_ERR_ERRCAT_OPEN

表 3. 鍵管理戻りコード (続き)

戻りコード (16 進)	戻りコード (10 進)	定数
0x00000055	85	GSKKM_ERR_FILENAME_NULL
0x00000056	86	GSKKM_ERR_FILE_OPEN
0x00000057	87	GSKKM_ERR_FILE_OPEN_TO_READ
0x00000058	88	GSKKM_ERR_FILE_OPEN_TO_WRITE
0x00000059	89	GSKKM_ERR_FILE_OPEN_NOT_EXIST
0x0000005a	90	GSKKM_ERR_FILE_OPEN_NOT_ALLOWED
0x0000005b	91	GSKKM_ERR_FILE_WRITE
0x0000005c	92	GSKKM_ERR_FILE_REMOVE
0x0000005d	93	GSKKM_ERR_BASE64_INVALID_DATA
0x0000005e	94	GSKKM_ERR_BASE64_INVALID_MSGTYPE
0x0000005f	95	GSKKM_ERR_BASE64_ENCODING
0x00000060	96	GSKKM_ERR_BASE64_DECODING
0x00000061	97	GSKKM_ERR_DN_TAG_NULL
0x00000062	98	GSKKM_ERR_DN_CN_NULL
0x00000063	99	GSKKM_ERR_DN_C_NULL
0x00000064	100	GSKKM_ERR_INVALID_DB_HANDLE
0x00000065	101	GSKKM_ERR_KEYDB_NOT_EXIST
0x00000066	102	GSKKM_ERR_KEYPAIRDB_NOT_EXIST
0x00000067	103	GSKKM_ERR_PWDFILE_NOT_EXIST
0x00000068	104	GSKKM_ERR_PASSWORD_CHANGE_MATCH
0x00000069	105	GSKKM_ERR_KEYDB_NULL
0x0000006a	106	GSKKM_ERR_REQKEYDB_NULL
0x0000006b	107	GSKKM_ERR_KEYDB_TRUSTCA_NULL
0x0000006c	108	GSKKM_ERR_REQKEY_FOR_CERT_NULL
0x0000006d	109	GSKKM_ERR_KEYDB_PRIVATE_KEY_NULL
0x0000006e	110	GSKKM_ERR_KEYDB_DEFAULT_KEY_NULL
0x0000006f	111	GSKKM_ERR_KEYREC_PRIVATE_KEY_NULL
0x00000070	112	GSKKM_ERR_KEYREC_CERTIFICATE_NULL
0x00000071	113	GSKKM_ERR_CRLS_NULL
0x00000072	114	GSKKM_ERR_INVALID_KEYDB_NAME
0x00000073	115	GSKKM_ERR_UNDEFINED_KEY_TYPE
0x00000074	116	GSKKM_ERR_INVALID_DN_INPUT
0x00000075	117	GSKKM_ERR_KEY_GET_BY_LABEL
0x00000076	118	GSKKM_ERR_LABEL_LIST_CORRUPT
0x00000077	119	GSKKM_ERR_INVALID_PKCS12_DATA
0x00000078	120	GSKKM_ERR_PKCS12_PWD_CORRUPTION
0x00000079	121	GSKKM_ERR_EXPORT_TYPE
0x0000007a	122	GSKKM_ERR_PBE_ALG_UNSUPPORT
0x0000007b	123	GSKKM_ERR_KYR2KDB

表 3. 鍵管理戻りコード (続き)

戻りコード (16 進)	戻りコード (10 進)	定数
0x0000007c	124	GSKKM_ERR_KDB2KYR
0x0000007d	125	GSKKM_ERR_ISSUING_CERTIFICATE
0x0000007e	126	GSKKM_ERR_FIND_ISSUER_CHAIN
0x0000007f	127	GSKKM_ERR_WEBDB_DATA_BAD_FORMAT
0x00000080	128	GSKKM_ERR_WEBDB_NOTHING_TO_WRITE
0x00000081	129	GSKKM_ERR_EXPIRE_DAYS_TOO_LARGE
0x00000082	130	GSKKM_ERR_PWD_TOO_SHORT
0x00000083	131	GSKKM_ERR_PWD_NO_NUMBER
0x00000084	132	GSKKM_ERR_PWD_NO_CONTROL_KEY
0x00000085	133	GSKKM_ERR_SIGNATURE_ALGORITHM
0x00000086	134	GSKKM_ERR_INVALID_DATABASE_TYPE
0x00000087	135	GSKKM_ERR_SECONDARY_KEYDB_TO_OTHER
0x00000088	136	GSKKM_ERR_NO_SECONDARY_KEYDB
0x00000089	137	GSKKM_ERR_CRYPTOGRAPHIC_TOKEN_ LABEL_NOT_EXIST
0x0000008a	138	GSKKM_ERR_CRYPTOGRAPHIC_TOKEN_ PASSWORD_REQUIRED
0x0000008b	139	GSKKM_ERR_CRYPTOGRAPHIC_TOKEN_ PASSWORD_NOT_REQUIRED
0x0000008c	140	GSKKM_ERR_CRYPTOGRAPHIC_TOKEN_ LIBRARY_NOT_LOADED
0x0000008d	141	GSKKM_ERR_CRYPTOGRAPHIC_TOKEN_ NOT_SUPPORT
0x0000008e	142	GSKKM_ERR_CRYPTOGRAPHIC_TOKEN_ FUNCTION_FAILED
0x0000008f	143	GSKKM_ERR_LDAP_USER_NOT_FOUND
0x00000090	144	GSKKM_ERR_LDAP_INVALID_PASSWORD
0x00000091	145	GSKKM_ERR_LDAP_QUERY_ENTRY_FAILED
0x00000092	146	GSKKM_ERR_INVALID_CERT_CHAIN
0x00000093	147	GSKKM_ERR_CERT_ROOT_NOT_TRUSTED
0x00000094	148	GSKKM_ERR_CERT_REVOKED
0x00000095	149	GSKKM_ERR_CRYPTOGRAPHIC_OBJECT_ FUNCTION_FAILED
0x00000096	150	GSKKM_ERR_NO_AVAILABLE_CRL_ DATASOURCE
0x00000097	151	GSKKM_ERR_NO_TOKEN_PRESENT
0x00000098	152	GSKKM_ERR_FIPS_NOT_SUPPORTED
0x00000099	153	GSKKM_ERR_FIPS_CONFLICT_SETTING
0x0000009a	154	GSKKM_ERR_PASSWORD_STRENGTH_FAILED

保存済みデータの暗号化のための IBM Database Encryption Expert

IBM Database Encryption Expert は、ネイティブ DB2 セキュリティーとともに使用するとき、広範囲にわたる脅威に対するデータおよびデータベース・アプリケーションの有効な保護を提供する、包括的なソフトウェア・データ・セキュリティー・ソリューションです。

Database Encryption Expert は、組織が非公開の機密データを強力に保護し、規則および法律に準拠することを確実にするのに役立ちます。Database Encryption Expert の主な利点は以下のとおりです。

- DB2 データベース・システム用の、実績のある強力なデータ・セキュリティー
- ライブ・ファイル、構成ファイル、ログ・ファイル、およびバックアップ・データの保護
- アプリケーション、データベース、およびストレージ環境に対して透過的
- オンライン環境とオフライン環境の両方でのデータ保護のための統一されたポリシーおよびキー管理
- パフォーマンス要件を満たしている

Database Encryption Expert により、オフラインのデータベース・バックアップの暗号化とオンライン (「ライブ」) のデータベース・ファイルの暗号化ができます。これは、ディスク上のデータ (ネットワークを介して移動している「移動中データ」に対し「保管済みデータ」とも言います) の暗号化です。

- バックアップの場合、データはバックアップされているときに暗号化されるので、バックアップ・デバイス上のデータは暗号化されています。データをリカバリーする必要がある場合、リカバリー・サーバーはデータが暗号化されていることを認識し、データの暗号化解除を行います。
- データベース・ファイルの場合、DB2 データベースからのデータを含むオペレーティング・システムのデータ・ファイルが暗号化されます。これにより、「生の」データベース・ファイルを読み取ろうとする無許可ユーザーからデータ・ファイルを保護します。

Database Encryption Expert は、ユーザー、データベース、アプリケーション、およびストレージに対して透過的です。コードの変更または既存のインフラストラクチャーへの変更は必要ありません。Database Encryption Expert は、ユーザーがこれまでと同じ方法でデータへのアクセスを継続している間に、どんなストレージ環境でもデータを保護することができます。

Database Encryption Expert は、実行可能ファイル、構成ファイル、ライブラリーなどへの変更を防ぐことができ、それによりアプリケーションへの攻撃を防ぐので、データベース・アプリケーションを保護することができます。

注: DB2 pureScale®環境では Database Encryption Expert は使用できません。

Database Encryption Expert のアーキテクチャー

Database Encryption Expert は、Web ベースのユーザー・インターフェースおよびコマンド行ユーティリティーを使用して管理する、エージェントとサーバー・ソフト

ウェア・パッケージのセットです。Database Encryption Expert の管理者は、セキュリティーおよび暗号化がインプリメントされる方法を規定するセキュリティー・ポリシーを構成します。

これらのセキュリティー・ポリシーが定義される方法に従って、Database Encryption Expert バックアップ・エージェントが DB2 のバックアップを暗号化し、Database Encryption Expert ファイル・システム・エージェントが DB2 データ・ファイルを暗号化します。

Encryption Expert Security Server は、セキュリティー・ポリシー、暗号鍵、およびイベント・ログ・ファイルを保管します。セキュリティー・ポリシーには、アクセスを許可または拒否するために満たさなければならないセキュリティー規則のセットが含まれています。各セキュリティー規則は、誰が、いつ、どの保護データにどのようにアクセスするかを評価し、これらの基準が一致する場合、Security Server はアクセスを許可または拒否します。

図 4 は、Database Encryption Expert のアーキテクチャーを図示しています。

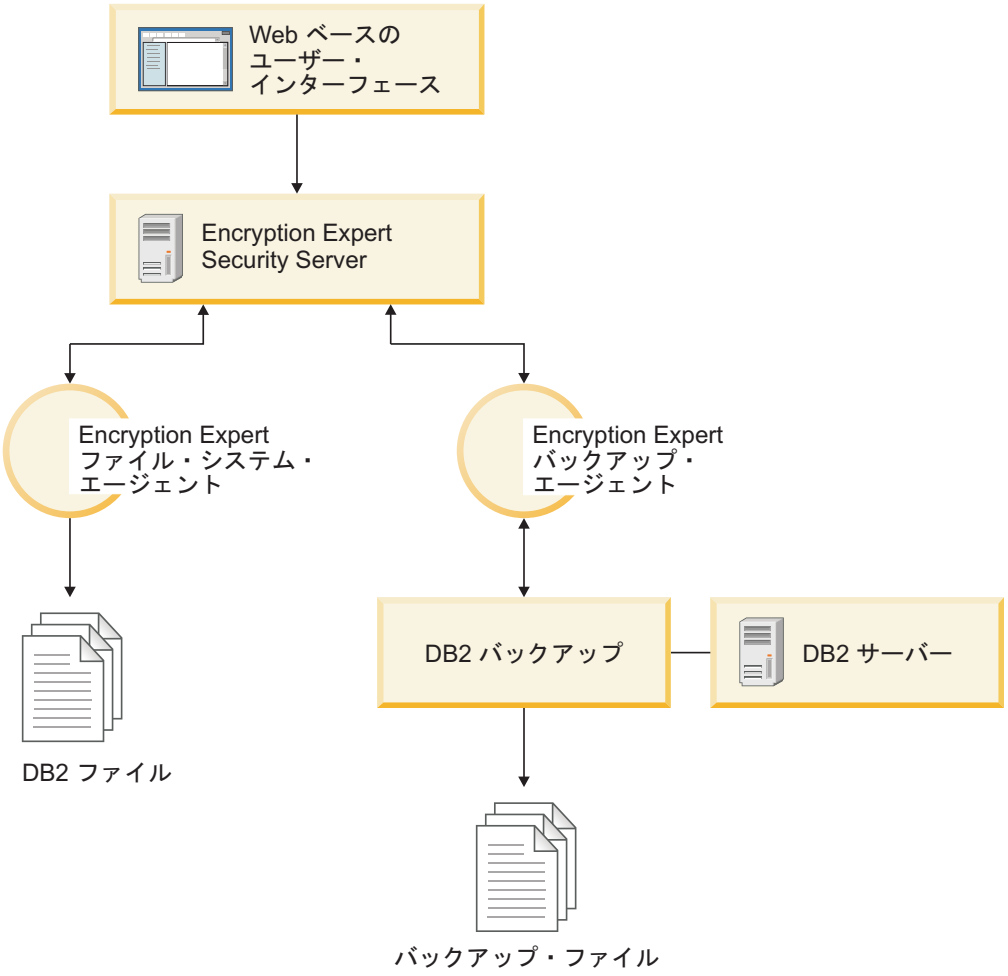


図 4. Database Encryption Expert のアーキテクチャー

ファイル・システム・エージェント

Database Encryption Expert ファイル・システム・エージェント・プロセスは、常にバックグラウンドで実行中です。このエージェントは、保護しているデータ・ファイル、ディレクトリー、または実行可能ファイルへのアクセスの試行を代行受信します。Database Encryption Expert ファイル・システム・エージェントは、アクセスの試行を Security Server に転送し、Security Server は、適用されるポリシーに基づいて、試行されたアクセスを認可または拒否します。

Database Encryption Expert の保護は、単なるファイルへのアクセスの許可または拒否を超えて拡張され、ファイルを暗号化することもできます。ファイル内容だけが暗号化されますが、ファイルのメタデータはそのままにされます。したがって、名前、タイム・スタンプ、ファイル・タイプなどをただ表示するためだけに、暗号化されたファイルを復号する必要はありません。これにより、データ管理アプリケーションはファイル内容を公開せずに、機能を実行することができます。例えば、バックアップ・マネージャーは内容を表示できませんが、特定のデータをバックアップできます。

暗号化されたファイルが無許可ユーザーによってアクセスされても、その内容は適切な Security Server の承認と暗号鍵なしでは、価値がありません。逆に、正しいポリシーおよび許可を持つユーザーは、暗号化および復号が行われていることに気付きません。

バックアップ・エージェント

通常 DB2 バックアップ API システムによって実行されるすべてのデータベースのバックアップ機能は、ネイティブ・データベースの圧縮を含めて、Database Encryption Expert のサーバーによってサポートされます。追加のコマンド行引数を除いて、DB2 のバックアップ・オペレーターは Database Encryption Expert の介入に気付きません。Database Encryption Expert は静的保存済みデータとアクティブ・オンライン・データをバックアップおよびリストアします。

基本的なバックアップおよびリストアの構成がサポートされています。基本構成では、データは 1 つのサーバーと複数のエージェントを使用して暗号化され、バックアップされます。また、データは元々バックアップの作成に使用されたのと同じサーバーを使って構成されたエージェントで復号およびリストアされます。

単一サイトおよびマルチサイト構成も、バックアップおよびリストアでサポートされています。単一サイトのシナリオでは、構成データは単一のデータ・センターで複数の Security Server にミラーリングされます。マルチサイトのシナリオでは、バックアップは異なるデータ・センターの異なる Encryption Expert サーバー上にリストアされます。

監査ロギング

Database Encryption Expert のエージェント・アクティビティーは、集中監査ロギング機能によって詳しくモニターされ、ログに記録されています。バックアップ、リストア、およびセキュリティー管理操作を含むすべての監査可能イベントがログに記録できます。これには、初期化、シャットダウンおよび再始動などの Database Encryption Expert のシステム・イベント、および異なる Database Encryption Expert コンポーネント間のネットワーク接続と切断が含まれています。

Database Encryption Expert の資料

Database Encryption Expert についての詳細は、以下の Web ページに進んでください。http://publib.boulder.ibm.com/infocenter/mptoolic/v1r0/topic/com.ibm.db2tools.eet.doc.ug/eetwelcome.htm

AIX 暗号化ファイル・システム (EFS) によるデータベース暗号化

AIX オペレーティング・システムで実行されている DB2 Enterprise Server Edition の場合、AIX 暗号化ファイル・システム (EFS) を使用して暗号化データベースをセットアップするオプションがあります。 EFS の詳細については、AIX 資料を参照してください。

注: パーティション・データベース環境 で作業する場合、EFS を使用するにはデータベースは単一のデータベース・パーティションになければなりません。

JFS2 ファイル・システムで基礎となる EFS を使用すると、データベース表内のデータが含まれるオペレーティング・システム・ファイルを暗号化できます。

暗号化をセットアップするステップは、以下のとおりです。

1. システムで EFS を使用可能にします。
2. DB2 データベース・デーモンが実行されているユーザー・アカウントの鍵ストアをロードします。
3. データベース・ファイル・システムで EFS を使用可能にします。
4. 暗号化するオペレーティング・システム・ファイルを判別します。
5. EFS 保護が必要なデータベース表が含まれるファイルを暗号化します。

システムにおける EFS の使用可能化

EFS を使用可能にする前に、`clic.rte` ファイル・セットをインストールする必要があります。 `clic.rte` インストール・イメージは、Expansion Pack CD にあります。

以下のコマンドを root として実行して、システムで EFS を使用可能にします。

```
% efsenable -a
```

efsenable コマンドを実行する必要があるのは、一度だけです。

鍵ストアのロード

以下の構成例では、データベース・デーモンが実行される DB2 ユーザー・アカウントは abst という名前です。ユーザー abst には鍵ストアがなければならず、abst がメンバーとなっているグループにも鍵ストアがなければなりません。

1. DB2 デーモンを開始する前に、すべての鍵ストアを abst プロセスに関連付ける必要があります。

   以下の例に示されているように、**efskeymgr -V** コマンドを使用して関連付けを検査できます。

   ```
   # lsuser abst
   abst id=203 pgrp=abstgp groups=abstgp,staff ...
   ```

```
# efskeymgr -V
List of keys loaded in the current process:
  Key #0:
                    Kind ..................... User key
                    Id (uid / gid) ......... 203
                    Type ..................... Private
key
                    Algorithm ................ RSA_1024
                    Validity ................. Key is
valid
                    Fingerprint ..............
24c88df2:d91cb6a2:c3e11b6a:4c13f8b4:666fabd8

  Key #1:
                    Kind ..................... Group
key
                    Id (uid / gid) ......... 1
                    Type ..................... Private
key
                    Algorithm ................ RSA_1024
                    Validity ................. Key is
valid
                    Fingerprint ..............
03fead42:57e7646e:a1715626:cfa56c8e:8abed1c1

  Key #2:
                    Kind ..................... Group
key
                    Id (uid / gid) ......... 212
                    Type ..................... Private
key
                    Algorithm ................ RSA_1024
                    Validity ................. Key is
valid
                    Fingerprint ..............
339dfb19:bc850f4c:5551c975:7fe4961b:2dddf3bc
```

2. abst プロセスに関連付けられているものとして表示される鍵ストアがない場合、次のコマンドを使用して鍵ストアのロードを試行してください。`% efskeymgr -o ksh`

   このコマンドにより、鍵ストア・パスワードを求めるプロンプトが出ます。これは、最初、ログイン・パスワードに設定されています。

3. `% efskeymgr -V` コマンドを再実行して、ユーザーとグループの鍵がロードされていることを確認してください。

   ユーザーとグループの鍵の両方がリストされている必要があります。依然としてグループ鍵ストアがリストされていない場合には、ステップ 4 に進みます。

4. グループが作成された方法によっては、グループ鍵ストアが存在しない場合もあります。**efskeymgr -V** コマンドによってユーザーのグループ鍵ストアがリストされない場合、グループ鍵ストアを作成する必要があります。

   root または RBAC 役割 aix.efs_admin として、グループ鍵ストアを作成します。

   `% efskeymgr -C` *group_name*

5. 各アプリケーション・ユーザーに、以下のようにしてグループ鍵ストア・アクセスを割り当てます。

   `% efskeymgr -k group /group_name -s user/`*user_name*

ユーザーが既にログインしている場合、グループ鍵ストアに対するアクセス権をすぐに持つことはなく、**efskeymgr -o ksh** コマンドを使用して鍵ストアを再ロードするか、再ログインする必要があります。

データベース・ファイル・システムにおける EFS の使用可能化

EFS は JFS2 ファイル・システムでのみ実行し、明示的に使用可能にする必要があります。

データベースが既存のファイル・システムにある場合は、`% chfs -a efs=yes` *`filesystem`* コマンドを実行して EFS を使用可能にします。例えば、以下のようにします。

```
% chfs -a efs=yes /test01
```

新しいファイル・システムを作成する場合は、**smit** コマンドまたは **crfs** コマンドに `-a efs=yes` オプションを指定して EFS を使用可能にできます。例えば、以下のようにします。

```
% crfs -v jfs2 -a efs=yes -m mount_point -d devide -A yes
```

これでファイル・システムで EFS が使用可能になりましたが、オンにはなっていません。暗号化データが必要な特定のデータベース表に対してのみ EFS をオンにしてください (詳しくは、AIX EFS 資料で **efsmgr** コマンドおよび継承について参照してください)。

暗号化するファイルの判別

EFS 暗号化で保護する特定のデータベース表が含まれるファイルを判別するには、以下のステップに従ってください。ここでは例として EMPLOYEE 表を使用します。

1. 以下の例のような照会を使用して、表の TBSPACEID を検出します。

   ```
   SELECT TABNAME, TBSPACEID FROM syscat.tables WHERE tabname='EMPLOYEE'
   ```

   この照会の結果が以下のようになるとします。

TABNAME	TBSPACEID
EMPLOYEE	2

2. 以下の例と同様の照会を使用して、この TBSPACEID の表スペースをルックアップします。

   ```
   LIST TABLESPACE CONTAINERS FOR 2
   ```

   この照会の結果が以下のようになるとします。

コンテナー ID	名前	タイプ
0	/test01/abst/NODE0000/BAR/T0000002/C0000000.LRG	ファイル

   これで、この表スペースが `/test01/abst/NODE0000/BAR/T0000002/C0000000.LRG` というオペレーティング・システム・ファイルに含まれていることが分かります。これが、暗号化の必要なファイルです。

ファイルの暗号化

データまたはデータベースに大きな変更を加える場合にするのと同様、まずデータベースをバックアップします。

ファイルを暗号化するには、以下のステップに従ってください。

1. ファイルをリストします。例えば、以下のようにします。

   ```
   # ls -U /test01/abst/NODE0000/BAR/T0000002/C0000000.LRG

   -rw-------- 1 abst abstgp 33554432 Jul 30 18:01
   /test01/abst/NODE0000/BAR/T0000002/C0000000.LRG
   ```

2. **efsmgr** コマンドを使用してファイルを暗号化します。例えば、以下のようにします。

   ```
   # efsmgr -e /test01/abst/NODE0000/BAR/T0000002/C0000000.LRG
   ```

   このファイルを再びリストすると、許可ストリングの末尾に「e」が表示されます。これは、ファイルが暗号化されていることを示します。例えば、以下のようにします。

   ```
   # ls -U /test01/abst/NODE0000/BAR/T0000002/C0000000.LRG

   -rw-------e 1 abst abstgp 33554432 Jul 30 18:03
   /test01/abst/NODE0000/BAR/T0000002/C0000000.LRG
   ```

3. DB2 データベース・マネージャーを開始し、通常の方法で使用します。EMPLOYEE 表とこの暗号化表スペースに追加するすべてのデータは、基礎となるファイル・システム内で EFS によって暗号化されます。データを取り出すときには、DB2 データベース・マネージャーによって暗号化解除されて、通常どおりに提示されます。

DB2 アクティビティーの監査

DB2 監査機能の紹介

機密データへのアクセスを管理するためには、さまざまな認証およびアクセス制御メカニズムを使用して、規則を確立し、受け入れ可能なデータ・アクセスを制御することができます。しかし、認識されていない動作や受け入れることのできない動作からデータを保護したり、そのような動作を発見したりするためには、DB2 監査機能を使用してデータ・アクセスをモニターできます。

不適切なデータ・アクセスを正常にモニターして分析することにより、データ・アクセスの制御を改善し、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することができます。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提供します。

DB2 監査機能は、事前に定義した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保存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。これらのレコードを分析することで、使用パターンが明らかになり、システム誤用を識別することができます。そのようなシステム誤用を識別した場合、それを制限または除去できます。

監査機能は、インスタンス・レベルと個々のデータベース・レベルの両方で監査を行い、すべてのインスタンス・レベルおよびデータベース・レベルのアクティビティーを、それぞれ別々のログに分けて記録する能力を備えています。システム管理者 (SYSADM 権限を保持するユーザー) は、**db2audit** ツールを使用して、インスタンス・レベルで監査を構成したり、いつそのような監査情報を収集するかを制御したりすることができます。**db2audit** ツールでは、システム管理者がインスタンス監査ログとデータベース監査ログの両方をアーカイブできるほか、アーカイブされているいずれかのタイプのログから監査データを抽出することもできます。

セキュリティー管理者 (データベース内で SECADM 権限を持つユーザー) は、監査ポリシーと SQL ステートメント AUDIT を組み合わせて使用することによって、個々のデータベースごとに監査要件を構成および制御できます。セキュリティー管理者は、以下の監査ルーチンを使用して、指定のタスクを実行できます。

- SYSPROC.AUDIT_ARCHIVE ストアード・プロシージャーは、監査ログをアーカイブします。
- SYSPROC.AUDIT_LIST_LOGS 表関数を使用すると、関心のあるログの場所を特定できます。
- SYSPROC.AUDIT_DELIM_EXTRACT ストアード・プロシージャーは、分析を行うためにデータを区切り文字で区切られているファイルに取り出します。

セキュリティー管理者は別のユーザーにこうしたルーチンの EXECUTE 特権を付与できるので、必要に応じてこれらのタスクを委任できます。

パーティション・データベース環境において作動しているとき、ユーザーが接続しているデータベース・パーティション (コーディネーター・パーティション)、またはカタログ・パーティション (同じデータベース・パーティションではない場合) で、多数の監査可能なイベントが発生します。つまり、監査レコードが複数のデータベース・パーティションによって生成される可能性があります。それぞれの監査レコードの中には、コーディネーター・パーティションと起点パーティション (監査レコードの起点となっているパーティション) を識別する情報が含まれています。

インスタンス・レベルでは、監査機能は、**db2audit start** および **db2audit stop** コマンドを使用することによって明示的に停止および開始する必要があります。インスタンス・レベルの監査を開始する際、監査機能は既存の監査構成情報を使用します。監査機能は DB2 データベース・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てもアクティブのままです。実際、インスタンスが停止されるとき、監査レコードが監査ログに生成される可能性があります。データベース・レベルで監査を開始するには、最初に監査ポリシーを作成し、その監査ポリシーをモニター対象のオブジェクト (許可 ID、データベース権限、トラステッド・コンテキスト、または特定の表など) と関連付ける必要があります。

監査レコードの区分

生成される監査レコードにはさまざまな区分があります。監査するために使用可能なイベントの区分に関する以下の記述では、各区分の名前に続いて、区分タイプの識別に使用される 1 つの単語のキーワードがあることに注意してください。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にレコードを生成する。
- 許可検査 (CHECKING)。 DB2 データベース・オブジェクトや関数に対するアクセス試行または操作試行の許可検査のときに、レコードを生成する。
- オブジェクト保守 (OBJMAINT)。データ・オブジェクトを作成またはドロップするとき、および特定のオブジェクトに変更を加えるときにレコードを生成する。
- セキュリティー保守 (SECMAINT)。以下の場合にレコードを生成する。
  - オブジェクト特権やデータベース権限を付与または取り消すとき
  - セキュリティー・ラベルや免除を付与または取り消すとき
  - LBAC セキュリティー・ポリシーのグループ許可、ロール許可、あるいはオーバーライドまたは制限属性を変更するとき
  - SETSESSIONUSER 特権を付与または取り消すとき
  - SYSADM_GROUP、SYSCTRL_GROUP、SYSMAINT_GROUP、または SYSMON_GROUP 構成パラメーターのいずれかに変更を加えるとき
- 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。
- ユーザー妥当性検査 (VALIDATE)。ユーザーを認証しているとき、またはシステムのセキュリティー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を同時に使用することで、イベントのグループを 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。例えば、動的照会の照会ステートメント、静的照会のパッケージ ID、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提供する SQL または XQuery ステートメントは、かなり長くなる可能性があり、CONTEXT レコード内にすべてが示されます。このため、CONTEXT レコードが非常に大きくなる可能性があります。
- 実行 (EXECUTE)。SQL ステートメントの実行中にレコードを生成する。

上記でリストされた区分のいずれにおいても、失敗、成功、またはその両方を監査できます。

データベース・サーバーに対して何らかの操作が行われる場合、いくつかのレコードが生成されることがあります。監査ログに生成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、記録されるイベントの区分の数によって決定されます。さらに、成功、失敗、またはその両方を監査するかどうかによっても異なります。このため、監査対象のイベントの選択は重要です。

監査ポリシー

セキュリティー管理者は、監査ポリシーを使用することによって、必要なデータやオブジェクトに関する情報だけを収集するように監査機能を構成できます。

セキュリティー管理者は、監査ポリシーを作成して、個々のデータベース内で何を監査の対象にするかを制御することができます。監査ポリシーを関連付けることができるオブジェクトには、以下のものがあります。

- データベース全体

  データベース内で発生する監査可能イベントすべてが、監査ポリシーに従って監査の対象となります。

- 表

  表 (非型付き)、MQT (マテリアライズ照会表)、またはニックネームへのデータ操作言語 (DML) アクセスおよび XQUERY アクセスは、すべて監査の対象となります。表がアクセス中になっているときは、ポリシーが他にも監査の対象となる区分を指示している場合でも、EXECUTE 区分の監査イベント (データを伴う場合と伴わない場合があります) のみが生成されます。

- トラステッド・コンテキスト

  特定のトラステッド・コンテキストで定義されたトラステッド接続内で発生する監査可能イベントすべてが、監査ポリシーに従って監査の対象となります。

- ユーザー、グループ、またはロールを示す許可 ID

  指定されたユーザーによって開始される監査可能イベントすべてが、監査ポリシーに従って監査の対象となります。

  指定されたグループやロールに属しているユーザーが開始する監査可能イベントは、すべて監査ポリシーに従って監査の対象となります。他のロールやグループを介する場合など、間接的にロールに属している場合も、監査の対象になります。

  同様なデータのキャプチャーは、グループのワークロードを定義してアクティビティーの詳細をキャプチャーすることにより、Work Load Management イベント・モニターでも行うことができます。なお、ワークロードのマッピングには、許可 ID だけではなく、属性も関係する場合があるので注意が必要です。このことが原因となって、監査の際に期待した細分度が得られなかったり、それらの他の属性が変更された場合には接続が別の (おそらく、モニターされていない) ワークロードにマップされてしまったりすることがあります。監査用のソリューションを使用するなら、ユーザー、グループ、またはロールの監査を確実に行うことができます。

- 権限 (SYSADM、SECADM、DBADM、SQLADM、WLMADM、ACCESSCTRL、DATAACCESS、SYSCTRL、SYSMAINT、SYSMON)

  指定された権限を持つユーザーによって開始される監査可能イベントすべてが、そのイベントにその権限が必要であるかどうかにかかわりなく、監査ポリシーに従って監査の対象となります。

セキュリティー管理者は、複数の監査ポリシーを作成できます。例えば、会社は、機密データを監査するためのポリシーと、DBADM 権限を持つユーザーのアクティビティーを監査するためのポリシーを必要とするかもしれません。1 つのステートメントで複数の監査ポリシーが有効になれば、それぞれの監査ポリシーで監査が必

要とされるイベントすべてを (1 回の監査だけで) 監査することができます。例えば、データベースの監査ポリシーで特定の表における成功した EXECUTE イベントの監査が必要とされ、ユーザーの監査ポリシーで同じ表における EXECUTE イベントの失敗の監査が必要とされる場合は、その表へのアクセスの試行が成功した場合と失敗した場合の両方の監査が行われます。

特定のオブジェクトごとに有効にできる監査ポリシーは、1 つだけです。例えば、同じ表に複数の監査ポリシーを同時に関連付けることはできません。

監査ポリシーは、ビューや型付き表には関連付けることができません。監査ポリシーが関連付けられている表にアクセスするビューは、基礎表のポリシーに従って、監査の対象となります。

表に適用される監査ポリシーが、その表に基づいている MQT に自動的に適用されることはありません。表に監査ポリシーを関連付けている場合は、その表に基づいているすべての MQT に同じポリシーを関連付けてください。

トランザクション中に実行される監査は、監査ポリシーと、トランザクション開始時の関連付けに基づいて実行されます。例えば、セキュリティー管理者が、あるユーザーに監査ポリシーを関連付けていて、監査の時点でそのユーザーがトランザクションに入っている場合、監査ポリシーは、そのトランザクション内で実行される残りのステートメントには一切影響を与えません。また、監査ポリシーに対する変更は、コミットされるまで有効になりません。セキュリティー管理者が ALTER AUDIT POLICY ステートメントを発行する際は、そのステートメントがコミットされるまで、変更は有効になりません。

セキュリティー管理者は、監査ポリシーの作成には CREATE AUDIT POLICY ステートメントを、監査ポリシーの変更には ALTER AUDIT POLICY ステートメントを使用します。これらのステートメントでは、以下を指定できます。

- 監査の対象とするイベントの状況値: `None`、`Success`、`Failure`、または `Both`。

  指定された状況値と一致する監査可能イベントだけが監査の対象になります。
- 監査中にエラーが発生した場合のサーバーの振る舞い。

セキュリティー管理者は、AUDIT ステートメントを使用することにより、現行サーバーにおいて、現行のデータベースと監査ポリシーまたはデータベース・オブジェクトと監査ポリシーを関連付けることができます。オブジェクトが使用中になっているときは常に、この監査ポリシーに従って監査が行われます。

監査ポリシーを削除する場合、セキュリティー管理者は DROP ステートメントを使用します。監査ポリシーは、何らかのオブジェクトに関連付けられているとドロップできません。AUDIT REMOVE ステートメントを使用して、残っているオブジェクトとの関連付けをすべて除去してください。監査ポリシーにメタデータを追加する場合、セキュリティー管理者は COMMENT ステートメントを使用します。

完全な接続が確立される前に生成されるイベント

接続とユーザー切り替え操作の過程で生成されるいくつかのイベントの場合、利用できる監査ポリシー情報は、データベースに関連付けられているポリシーだけになります。このようなイベントを、次の表に示します。

表 4. 接続イベント

イベント	監査区分	コメント
CONNECT	CONTEXT	
CONNECT_RESET	CONTEXT	
AUTHENTICATION	VALIDATE	これには、トラステッド接続内でのユーザーの接続および切り替えの両方における認証が含まれます。
CHECKING_FUNC	CHECKING	試行されるアクセスは SWITCH_USER です。

これらのイベントは、データベースに関連付けられている監査ポリシーに基づいてのみ監査され、ユーザー、ユーザーのグループ、または権限などのそれ以外のオブジェクトに関連付けられている監査ポリシーでは監査されません。接続の過程で発生する CONNECT および AUTHENTICATION イベントについては、データベースがアクティブになるまで、インスタンス・レベルの監査設定が使用されます。データベースは、最初の接続の過程で、または ACTIVATE DATABASE コマンドが発行されたときにアクティブになります。

ユーザー切り替えの影響

トラステッド接続内でユーザーの切り替えが行われる場合、元のユーザーの名残が後に残ることはありません。このような場合、元のユーザーに関連付けられていた監査ポリシーは考慮されなくなり、新しいユーザーに合わせて適合する監査ポリシーが再評価されます。なお、トラステッド接続に関連付けられている監査ポリシーは引き続き有効です。

SET SESSION USER ステートメントが使用される場合は、セッション許可 ID だけが切り替わります。元のユーザーの許可 ID (システム許可 ID) の監査ポリシーは有効なまま残り、それに加えて新しいユーザーの監査ポリシーも使用されます。セッション内で複数の SET SESSION USER ステートメントが発行される場合は、元のユーザー (システム許可 ID) と現行ユーザー (セッション許可 ID) に関連付けられている監査ポリシーだけが考慮されます。

データ定義言語に関する制約事項

以下のデータ定義言語 (DDL) ステートメントは、AUDIT 排他 SQL ステートメントと呼ばれます。

- AUDIT
- CREATE AUDIT POLICY、ALTER AUDIT POLICY、および DROP AUDIT POLICY
- DROP ROLE および DROP TRUSTED CONTEXT (ドロップされるロールまたはトラステッド・コンテキストが監査ポリシーに関連付けられている場合)

AUDIT 排他 SQL ステートメントには、使用に際していくつかの制約事項があります。

- 各ステートメントの後には COMMIT または ROLLBACK を発行する必要があります。

- これらのステートメントを XA トランザクションなどのグローバル・トランザクション内で発行することはできません。

コミットされていない AUDIT 排他 DDL ステートメントは、全パーティションを通じて同時に 2 つ以上存在してはなりません。コミットされていない 1 つの AUDIT 排他 DDL ステートメントが実行されている間は、後続の AUDIT 排他 DDL ステートメントは現行の AUDIT 排他 DDL ステートメントがコミットまたはロールバックされるまで待機します。

注: 変更はカタログに記録されますが、COMMIT が発行されるまでは、ステートメントを発行した接続においても、変更は有効になりません。

特定の表に対するアクセスすべてを監査する例

EMPLOYEE 表に極めて機密性の高い情報が含まれていて、その表のデータに対するありとあらゆる SQL アクセスを監査しようとしている企業について考えます。表に対する全アクセスのトラッキングには、EXECUTE 区分を使用できます。この区分では、SQL ステートメント、およびオプションとしてそのステートメントの実行時に提供される入力データの値を監査できます。

表でのアクティビティーのトラッキングには、2 つのステップがあります。まず最初に、セキュリティー管理者は EXECUTE 区分を指定する監査ポリシーを作成します。そして次に、そのポリシーを表に関連付けます。

```
CREATE AUDIT POLICY SENSITIVEDATAPOLICY
    CATEGORIES EXECUTE STATUS BOTH ERROR TYPE AUDIT
COMMIT

AUDIT TABLE EMPLOYEE USING POLICY SENSITIVEDATAPOLICY
COMMIT
```

SYSADM または DBADM によるアクションすべてを監査する例

セキュリティー準拠の証明を完成させるには、企業は、システム管理 (SYSADM) またはデータベース管理 (DBADM) 権限を持つユーザーによるデータベース内でのありとあらゆるアクティビティーがモニター可能であることを示す必要があります。

データベース内でのすべてのアクションをキャプチャーするためには、EXECUTE 区分と SYSADMIN 区分の両方を監査する必要があります。セキュリティー管理者は、これらの 2 つの区分を監査する監査ポリシーを作成します。セキュリティー管理者は、AUDIT ステートメントを使用して、この監査ポリシーを SYSADM 権限や DBADM 権限に関連付けることができます。そして、SYSADM または DBADM 権限を保持しているすべてのユーザーについては、すべての監査可能イベントのログが記録されるようになります。次の例は、このような監査ポリシーを作成して、それを SYSADM および DBADM 権限に関連付ける方法を示しています。

```
CREATE AUDIT POLICY ADMINSPOLICY CATEGORIES EXECUTE STATUS BOTH,
   SYSADMIN STATUS BOTH ERROR TYPE AUDIT
COMMIT
AUDIT SYSADM, DBADM USING POLICY ADMINSPOLICY
COMMIT
```

特定のロールによるアクセスすべてを監査する例

自社の Web アプリケーションが自社の企業データベースにアクセスできるようにしている企業があります。その Web アプリケーションを使用している個人については、厳密なことがわかりません。使用されるロールだけがわかっており、そのロールを使用してデータベース許可を管理しています。企業は、そのロールに属している個人がデータベースにサブミットしている要求を調べて、それらのユーザーが Web アプリケーション以外からデータベースにアクセスしていないかどうかを確認するために、そのロールに属しているすべてのユーザーのアクションをモニターすることを望んでいます。

EXECUTE 区分には、この状況でユーザーのアクティビティーをトラッキングするために必要なレベルの監査が含まれています。最初のステップは、次のように、適切な監査ポリシーを作成して、Web アプリケーションで使用されるロール (この例では、TELLER と CLERK のロール) にその監査ポリシーを関連付けます。

```
CREATE AUDIT POLICY WEBAPPPOLICY CATEGORIES EXECUTE WITH DATA
   STATUS BOTH ERROR TYPE AUDIT
COMMIT
AUDIT ROLE TELLER, ROLE CLERK USING POLICY WEBAPPPOLICY
COMMIT
```

データベースの監査を使用可能にする例

ある企業は、SAMPLE という名前のデータベースで DDL の変更 (例: ALTER TABLE) を行っているユーザーを特定しようとしています。

```
CONNECT TO SAMPLE

CREATE AUDIT POLICY ALTPOLICY CATEGORIES AUDIT STATUS BOTH,
    OBJMAINT STATUS BOTH,  CHECKING STATUS BOTH,
    EXECUTE STATUS BOTH, ERROR TYPE NORMAL

AUDIT DATABASE USING POLICY ALTPOLICY
```

監査ログの保管と分析

監査ログをアーカイブすると、アクティブ監査ログはアーカイブ・ディレクトリーに移され、サーバーは新しいアクティブ監査ログの書き込みを開始します。後ほど、アーカイブ・ログのデータを区切り文字で区切られているファイルに抽出し、そのデータをそうしたファイルから DB2 データベース表にロードして分析できます。

監査ログのロケーションを構成すると、監査ログを大容量の高速なディスクに置くことができ、パーティション・データベース環境 では、オプションでメンバーごとにディスクを分けることも可能です。パーティション・データベース環境 では、アクティブ監査ログのパスを、メンバーごとに固有のディレクトリーにすることができます。メンバーごとに固有のディレクトリーを使用すると、各メンバーが別々のディスクに書き込みを行うため、ファイルの競合を防ぐのに役立ちます。

Windows オペレーティング・システムでの監査ログのデフォルト・パスは *instance*¥security¥auditdata で、Linux および UNIX オペレーティング・システムでのデフォルト・パスは *instance*/security/auditdata です。デフォルトのロケーションを使用することを望まない場合は、別のディレクトリーを選択できます (代替のロケーションとして使用するディレクトリーがまだない場合は、ご使用のシ

ステムに新規ディレクトリーを作成できます)。アクティブ監査ログのロケーションとアーカイブされた監査ログのロケーションのパスを設定するには、次の例のように、**datapath** および **archivepath** パラメーターを指定した **db2audit configure** コマンドを使用します。

```
db2audit configure datapath /auditlog archivepath /auditarchive
```

db2audit を使用して設定された監査ログの保管場所は、インスタンス内のすべてのデータベースに適用されます。

注: サーバー上に複数のインスタンスが存在する場合、各インスタンスはそれぞれ別個のデータおよびアーカイブ・パスを持つべきです。

パーティション・データベース環境 におけるアクティブ監査ログのパス (datapath)

パーティション・データベース環境 では、各パーティションで同じアクティブ監査ログのロケーション (**datapath** パラメーターで設定される) を使用する必要があります。これを実現する方法は 2 つあります。

1. **datapath** パラメーターを指定する際に、データベース・パーティション式を使用します。データベース・パーティション式を使用すると、監査ログ・ファイルのパスにパーティション番号を含めることができるため、データベース・パーティションごとに異なるパスになります。
2. すべてのメンバーで同一の共有ドライブを使用します。

データベース・パーティション式は、**datapath** パラメーターに指定する値の中のどの位置でも使用できます。例えば、3 つのメンバーを持つ、データベース・パーティション番号が 10 のシステムで、次のコマンドを実行します。

```
db2audit configure datapath '/pathForNode $N'
```

以下のパスを使用することになります。

- /pathForNode10
- /pathForNode20
- /pathForNode30

注: アーカイブ・ログ・ファイルのパス (**archivepath** パラメーター) の指定にデータベース・パーティション式を使用することはできません。

アクティブ監査ログのアーカイブ

db2audit ツールでは、システム管理者がインスタンス監査ログとデータベース監査ログの両方をアーカイブできるほか、アーカイブされているいずれかのタイプのログから監査データを抽出することもできます。

セキュリティー管理者、または監査ルーチンの EXECUTE 特権をセキュリティー管理者から付与されたユーザーは、SYSPROC.AUDIT_ARCHIVE ストアード・プロシージャーを実行することにより、アクティブ監査ログをアーカイブできます。ログからデータを抽出して区切りファイルにロードするには、SYSPROC.AUDIT_DELIM_EXTRACT ストアード・プロシージャーを使用できます。

監査ルーチンを使用して監査ログをアーカイブおよび抽出するためには、以下のステップを実行します。

1. アプリケーションがストアード・プロシージャー SYSPROC.AUDIT_ARCHIVE を使用してアクティブ監査ログのアーカイブを定期的に実行するように、スケジュールを作成します。
2. どのアーカイブされたログ・ファイルについて調べるかを決めます。SYSPROC.AUDIT_LIST_LOGS 表関数を使用して、アーカイブされた監査ログをすべてリストします。
3. ログからデータを抽出して区切りファイルにロードするため、ファイル名を SYSPROC.AUDIT_DELIM_EXTRACT ストアード・プロシージャーにパラメーターとして渡します。
4. 監査データを分析用の DB2 データベース表にロードします。

アーカイブされたログ・ファイルは、すぐに分析用の表にロードする必要はなく、将来の分析のために保管しておくことができます。例えば、企業の監査が行われるときにしか閲覧する必要がないということもあるかもしれません。

アーカイブの途中で、アーカイブ・パスのディスク・スペースがいっぱいになってしまった、あるいはアーカイブ・パスが存在しない、などの問題が発生した場合は、アーカイブ・プロセスは失敗し、監査ログ・データのパスに .bk という拡張子を持つ中間ログ・ファイルが生成されます (例えば db2audit.instance.log.0.20070508172043640941.bk)。 (アーカイブ・パスに十分なディスク・スペースを割り振ることによって、またはアーカイブ・パスを作成することによって) 問題が解決されたなら、この中間ログをアーカイブ・パスに移動させてください。アーカイブ・パスに移動させた後は、この中間ログは正常にアーカイブされたログと同じ方法で扱うことができます。

パーティション・データベース環境 におけるアクティブ監査ログのアーカイブ

パーティション・データベース環境 では、インスタンスの実行中にアーカイブ・コマンドが発行されると、自動的にすべてのメンバーでアーカイブ・プロセスが実行されます。アーカイブされたログ・ファイルの名前には、すべてのメンバーで同じタイム・スタンプが使用されます。例えば、3 つのメンバーを持つ、データベース・パーティション番号が 10 のシステムで、次のコマンドを実行します。

```
db2audit archive to /auditarchive
```

すると、以下のファイルが作成されます。

- /auditarchive/db2audit.log.10.*timestamp*
- /auditarchive/db2audit.log.20.*timestamp*
- /auditarchive/db2audit.log.30.*timestamp*

アーカイブ・コマンドが発行されたときにインスタンスが実行中でない場合は、以下のいずれかの方法によって、どのメンバーでアーカイブを実行するかを制御できます。

- 現行メンバーのみのアーカイブを実行する場合は、**db2audit** コマンドに **node** オプションを使用します。

- すべてのメンバーでアーカイブを実行する場合は、**db2_all** コマンドを使用します。

  例えば、以下のようにします。

  ```
  db2_all db2audit archive node to /auditarchive
  ```

  これにより、コマンドが呼び出されたメンバーを示すように **DB2NODE** 環境変数が設定されます。

あるいは別の方法として、各メンバー別に個々のアーカイブ・コマンドを発行することもできます。例えば、以下のようにします。

- メンバー 10 で:

  ```
  db2audit archive node 10 to /auditarchive
  ```

- メンバー 20 で:

  ```
  db2audit archive node 20 to /auditarchive
  ```

- メンバー 30 で:

  ```
  db2audit archive node 30 to /auditarchive
  ```

注: インスタンスが実行中でない場合、アーカイブされた監査ログ・ファイル名のタイム・スタンプは、メンバーごとに異なります。

注: すべてのメンバーでアーカイブ・パスを共有することが勧められていますが、必須ではありません。

注: AUDIT_DELIM_EXTRACT ストアード・プロシージャーと AUDIT_LIST_LOGS 表関数でアクセスできるのは、現行の (コーディネーター) メンバーから見えるアーカイブされたログ・ファイルだけです。

ログのアーカイブと表へのデータ抽出の例

監査データを確実にキャプチャーし、将来の使用に備えて保管しておくために、ある企業では、6 時間ごとに新しい監査ログを作成し、現行の監査ログを WORM ドライブにアーカイブする必要があります。この企業では、セキュリティー管理者、または AUDIT_ARCHIVE ストアード・プロシージャーの EXECUTE 特権をセキュリティー管理者により付与されたユーザーによって 6 時間ごとに以下の SYSPROC.AUDIT_ARCHIVE ストアード・プロシージャーへの呼び出しが発行されるよう、スケジュールを立てています。アーカイブされたログのパスはデフォルトのアーカイブ・パス /auditarchive で、アーカイブはすべてのメンバーで実行されます。

```
CALL SYSPROC.AUDIT_ARCHIVE( '/auditarchive', -2 )
```

セキュリティー手順の一環として、この企業は、監査データ内で注意を必要とする多くの疑わしい動作や許可されないアクティビティーを識別し、定義しています。この企業は、1 つ以上の監査ログからすべてのデータを抽出し、それをリレーショナル表に置き、SQL 照会を使用してこれらのアクティビティーを探すことを希望しています。監査する適切な区分を判別してあり、データベースや他のデータベース・オブジェクトには必要な監査ポリシーが関連付けられています。

例えば、SYSPROC.AUDIT_DELIM_EXTRACT ストアード・プロシージャーを呼び出し、デフォルトの区切り文字を使用して、2006 年 4 月のタイム・スタンプで作成されたすべてのメンバーのすべての区分について、アーカイブされた監査ログを抽出できます。

```
CALL SYSPROC.AUDIT_DELIM_EXTRACT(
     '', '', '/auditarchive',  'db2audit.%.200604%', '' )
```

別の例として、SYSPROC.AUDIT_DELIM_EXTRACT ストアード・プロシージャーを呼び出し、調べているタイム・スタンプが付いているファイルから、EXECUTE 区分の成功イベントについてのアーカイブされた監査レコードと、CHECKING 区分の失敗イベントについてのアーカイブされた監査レコードを抽出できます。

```
CALL SYSPROC.AUDIT_DELIM_EXTRACT( '', '', '/auditarchive',
    'db2audit.%.20060419034937', 'category
     execute status success, checking status failure );
```

監査ログ・ファイル名:

監査ログ・ファイルの名前は、そのログがインスタンス・レベルのログかデータベース・レベルのログかを区別し、パーティション・データベース環境では、そのログがどのパーティションで作成されたかを示します。アーカイブされた監査ログのファイル名には、いつアーカイブ・コマンドが実行されたのかを示すタイム・スタンプが付加されます。

アクティブ監査ログ・ファイル名

パーティション・データベース環境では、アクティブ監査ログのパスを各パーティションに固有のディレクトリーにして、各パーティションが個々のファイルに書き込みを行うようにすることができます。監査レコードの起源を正確に追跡するためには、監査ログ・ファイルの名前の一部にパーティション番号を含めます。例えば、パーティション 20 の場合、インスタンス・レベルの監査ログ・ファイル名は db2audit.instance.log.20 となります。このインスタンスの testdb というデータベースの監査ログ・ファイルは、db2audit.db.testdb.log.20 となります。

非パーティション・データベース環境では、パーティション番号は 0 (ゼロ) と見なされます。この場合、インスタンス・レベルの監査ログ・ファイル名は db2audit.instance.log.0 になります。このインスタンスの testdb というデータベースの監査ログ・ファイルは、db2audit.db.testdb.log.0 になります。

アーカイブされた監査ログ・ファイルの名前

アクティブ監査ログのアーカイブ時には、ファイル名に *YYYYMMDDHHMMSS* (*YYYY* は年、*MM* は月、*DD* は日、*HH* は時間、*MM* は分、*SS* は秒) という形式で現行タイム・スタンプが付加されます。

アーカイブ監査ログのファイル名の形式は、監査ログのレベルによって異なります。

インスタンス・レベルのアーカイブされた監査ログ
: インスタンス・レベルのアーカイブされた監査ログのファイル名は、db2audit.instance.log.*partition*.*YYYYMMDDHHMMSS* となります。

データベース・レベルのアーカイブされた監査ログ

データベース・レベルのアーカイブされた監査ログのファイル名は、db2audit.db*database*.log.*partition.YYYYMMDDHHMMSS* となります。

非パーティション・データベース環境の場合は、*partition* の値は 0 (ゼロ) になります。

タイム・スタンプは、アーカイブ・コマンドが実行された時刻を示します。したがって、この時刻は、必ずしもログの最後のレコードの時刻を正確に反映しているわけではありません。アーカイブされた監査ログ・ファイルには、ログ・ファイル名のタイム・スタンプよりも数秒後のタイム・スタンプを持つレコードが含まれていることがあります。これは次のような理由によります。

- アーカイブ・コマンドが発行される際、監査機能は、すべての処理中のレコードの書き込みが完了するまで、アーカイブされたログ・ファイルの作成を待機します。
- 複数のマシンがある環境では、リモート・マシンのシステム時刻とアーカイブ・コマンドが発行されたマシンのシステム時刻が同期化されていないことがあります。

パーティション・データベース環境の場合は、アーカイブの実行時にサーバーが稼働していれば、パーティション間でタイム・スタンプが整合され、アーカイブが実行されたパーティションで生成されたタイム・スタンプがその他のパーティションにも反映されます。

DB2 監査データを保持する表の作成:

監査データをデータベース表で操作する前に、データを保持するための表を作成する必要があります。表のデータを無許可ユーザーから保護するために、これらの表を 1 つの別個のスキーマで作成することを考慮してください。

始める前に

- スキーマの作成に必要な権限および特権については、 CREATE SCHEMA ステートメントの説明を参照してください。
- 表の作成に必要な権限および特権については、 CREATE TABLE ステートメントの説明を参照してください。
- 表を保持するために、どの表スペースを使用するかを決定します。 (このトピックでは、表スペースの作成方法については説明しません。)

注: 監査データを保持するために作成する必要がある表の形式は、リリースによって変わることがあります。新しい列が追加されたり、既存の列のサイズが変更されたりする場合もあります。スクリプト db2audit.ddl は、監査レコードを格納するための正しい形式の表を作成します。

このタスクについて

以下のいくつかの例は、区切りファイルに含まれるレコードを保持するための表を作成する方法を示しています。必要に応じて、これらの表を格納するための 1 つの別個のスキーマを作成することもできます。

ファイルに含まれるすべてのデータを必ずしも使用する必要がない場合には、表の定義から列を省略するか、必要に応じて、特定の表の作成を回避することができま

す。表の定義から列を省略した場合、これらの表にデータをロードするためのコマンドを変更する必要があります。

手順

1. **db2** コマンドを発行して、 DB2 コマンド・ウィンドウをオープンします。
2. オプション: 表を保持するためのスキーマを作成します。 この例では、スキーマの名前は AUDIT です。

   ```
   CREATE SCHEMA AUDIT
   ```

3. オプション: AUDIT スキーマを作成した場合には、表を作成する前に、そのスキーマに切り替えます。

   ```
   SET CURRENT SCHEMA = 'AUDIT'
   ```

4. スクリプト db2audit.ddl を実行して、監査レコードを格納する表を作成します。

   スクリプト db2audit.ddl は、sqllib/misc ディレクトリー (Windows の場合は sqllib¥misc ディレクトリー) にあります。このスクリプトは、データベースへの接続が存在しており、8K の表スペースが使用可能であることを前提としています。このスクリプトを実行するためのコマンドは **db2 +o -tf sqllib/misc/db2audit.ddl** です。スクリプトによって作成される表は AUDIT、CHECKING、OBJMAINT、SECMAINT、SYSADMIN、VALIDATE、CONTEXT、および EXECUTE です。

5. 表を作成した後、セキュリティー管理者なら SYSPROC.AUDIT_DELIM_EXTRACT ストアード・プロシージャーを使用して、あるいはシステム管理者なら **db2audit extract** コマンドを使用して、アーカイブされた監査ログ・ファイルから区切りファイルに監査レコードを抽出することができます。 区切りファイルの監査データは、ここで作成したデータベース表にロードすることができます。

DB2 監査データの表へのロード:

監査ログ・ファイルを区切り文字付きファイルにアーカイブして抽出し、監査データを保持するためのデータベース表を作成した後に、監査データを分析のために区切り文字付きファイルからデータベース表にロードすることができます。

このタスクについて

ロード・ユーティリティーを使用して、監査データを表にロードします。それぞれの表ごとに、別個のロード・コマンドを発行してください。表定義から 1 つまたは複数の列を省略した場合には、データを正常にロードするために、**LOAD** コマンドの内容を変更する必要があります。さらに、監査データの抽出時にデフォルト以外の区切り文字を指定した場合にもまた、使用する **LOAD** コマンドのバージョンを変更する必要があります。

手順

1. **db2** コマンドを発行して、 DB2 コマンド・ウィンドウをオープンします。
2. AUDIT 表をロードするために、以下のコマンドを発行します。

   ```
   LOAD FROM audit.del OF DEL MODIFIED BY DELPRIORITYCHAR LOBSINFILE
            INSERT INTO schema.AUDIT
   ```

注: DELPRIORITYCHAR 修飾子を指定して、バイナリー・データが正しく構文解析されるようにします。

注: LOAD コマンドの **LOBSINFILE** オプションを指定します (ラージ・オブジェクトのインライン・データは 32 KB に制限されるという制約があるため)。状況によっては、**LOBS FROM** オプションも使用する必要がある場合があります。

注: ファイル名を指定するときには、絶対パスを使用してください。例えば、Windows オペレーティング・システムの C: ドライブに DB2 データベース・システムがインストールされている場合、audit.del ファイルの完全修飾ファイル名として、C:¥Program Files¥IBM¥SQLLIB¥*instance*¥security¥audit.del と指定します。

3. CHECKING 表をロードするために、以下のコマンドを発行します。

```
LOAD FROM checking.del OF DEL MODIFIED BY DELPRIORITYCHAR LOBSINFILE
          INSERT INTO schema.CHECKING
```

4. OBJMAINT 表をロードするために、以下のコマンドを発行します。

```
LOAD FROM objmaint.del OF DEL MODIFIED BY DELPRIORITYCHAR LOBSINFILE
          INSERT INTO schema.OBJMAINT
```

5. SECMAINT 表をロードするために、以下のコマンドを発行します。

```
LOAD FROM secmaint.del OF DEL MODIFIED BY DELPRIORITYCHAR LOBSINFILE
          INSERT INTO schema.SECMAINT
```

6. SYSADMIN 表をロードするために、以下のコマンドを発行します。

```
LOAD FROM sysadmin.del OF DEL MODIFIED BY DELPRIORITYCHAR LOBSINFILE
          INSERT INTO schema.SYSADMIN
```

7. VALIDATE 表をロードするために、以下のコマンドを発行します。

```
LOAD FROM validate.del OF DEL MODIFIED BY DELPRIORITYCHAR LOBSINFILE
          INSERT INTO schema.VALIDATE
```

8. CONTEXT 表をロードするために、以下のコマンドを発行します。

```
LOAD FROM context.del OF DEL MODIFIED BY DELPRIORITYCHAR LOBSINFILE
          INSERT INTO schema.CONTEXT
```

9. EXECUTE 表をロードするために、以下のコマンドを発行します。

```
LOAD FROM execute.del OF DEL MODIFIED BY DELPRIORITYCHAR LOBSINFILE
          INSERT INTO schema.EXECUTE
```

10. これらの表へのデータのロードを完了したら、sqllib ディレクトリーの security/auditdata サブディレクトリーから .del ファイルを削除します。
11. 監査データを表にロードしたら、これらの表から分析のためにデータを選択する準備ができました。

次のタスク

表にあらかじめデータが入っていて、再びデータを入れたい場合には、既存の表データに新しい表データを追加するために、 **INSERT** オプションを使用してください。以前の **db2audit extract** 操作によるレコードを表から除去する場合には、**REPLACE** オプションを使って表を再びロードします。

監査のアーカイブおよび抽出のストアード・プロシージャー:

セキュリティー管理者は、 SYSPROC.AUDIT_ARCHIVE ストアード・プロシージャーと表関数、SYSPROC.AUDIT_DELIM_EXTRACT ストアード・プロシージャー、

および SYSPROC.AUDIT_LIST_LOGS 表関数を使用して、監査ログをアーカイブしたり、区切り文字で区切られたファイルにデータを抽出したりできます。

セキュリティー管理者は、これらのルーチンの EXECUTE 特権を別のユーザーに付与して、こうしたルーチンの使用をそのユーザーに委任できます。セキュリティー管理者だけが、これらのルーチンの EXECUTE 特権を付与できます。WITH GRANT OPTION 付きの EXECUTE 特権をこれらのルーチンに関して付与することはできません (SQLSTATE 42501)。

これらのストアード・プロシージャーや表関数を使用してデータベースの監査ログをアーカイブまたはリストするためには、データベースに接続している必要があります。

アーカイブしたファイルを別のデータベース・システムにコピーする場合で、そのアクセスにストアード・プロシージャーおよび表関数を使用することを望む場合は、データベース名が同じになるようにするか、同じデータベース名を含むようにファイルの名前を変更してください。

これらのストアード・プロシージャーや表関数では、インスタンス・レベルの監査ログはアーカイブまたはリストされません。インスタンス・レベルの監査ログをアーカイブおよび抽出するためには、システム管理者は **db2audit** コマンドを使用する必要があります。

これらのストアード・プロシージャーや表関数を使用して、以下の操作を実行できます。

表 5. 監査システムのストアード・プロシージャーおよび表関数

ストアード・プロシージャーおよび表関数	操作	コメント
AUDIT_ARCHIVE	現行監査ログをアーカイブします。	アーカイブのパスを入力として取ります。アーカイブ・パスが指定されない場合、このストアード・プロシージャーは監査構成ファイルからアーカイブ・パスを取得します。 アーカイブは各メンバーで実行され、同期化されたタイム・スタンプが監査ログ・ファイルのファイル名に付加されます。
AUDIT_LIST_LOGS	現行データベースの指定されたパスにある、アーカイブされた監査ログのリストを戻します。	

表 5. 監査システムのストアード・プロシージャーおよび表関数 (続き)

ストアード・プロシージャーおよび表関数	操作	コメント
AUDIT_ DELIM_EXTRACT	バイナリー・アーカイブ・ログからデータを抽出し、区切りファイルにロードします。	抽出された監査レコードは、DB2 データベース表にロードするのに適した区切り形式のファイルに置かれます。出力は区分ごとに 1 つずつ独立したファイルに置かれます。加えて、監査データに含まれているラージ・オブジェクトを保持するためのファイル **auditlobs** が作成されます。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** • **execute.del** • **auditlobs** ファイルが既に存在する場合は、そこに出力が追加されます。 **auditlobs** ファイルは、CONTEXT または EXECUTE 区分が抽出される場合に作成されます。抽出できるのは、現行データベースのアーカイブされた監査ログだけです。また、コーディネーター・メンバーから見えるファイルだけが抽出されます。 アーカイブされた監査ログは、インスタンスの所有者にしか削除できません。

SQL ステートメントを監査するための EXECUTE 区分

ユーザーによって発行される SQL ステートメントを正確に追跡するには、EXECUTE 区分を使用します。バージョン 9.5 以前のリリースでは、CONTEXT 区分を使用してこの情報を探さなければなりませんでした。

包括的なセキュリティー・ポリシーの一環として、ある企業が、一定の年数までさかのぼってデータベース内の特定の表に対する特定の要求の影響を分析できるような機能を必要とすることがあります。これを行うために、企業は、選択した任意の時点でのデータベースを再構成できるように、週単位のバックアップとそれに関連

するログ・ファイルをアーカイブするポリシーを確立する必要があります。さらに、データベースに対するすべての要求についての十分なデータベース監査情報を収集する必要があります。これにより、将来の任意の時点で、該当するリストアされたデータベースに対する任意の要求を再生して分析できるようになります。この要件は、静的 SQL ステートメントと動的 SQL ステートメントの両方を扱うことができるものです。

この EXECUTE 区分は、SQL ステートメントのテキストに加えて、コンパイル環境や、後でステートメントを再現するのに必要なその他の値もキャプチャーします。例えば、ステートメントを再現すると、SELECT ステートメントがどの行を戻したのかを正確に知ることができます。ステートメントを再実行するためには、まず、データベース表をステートメント発行時の状態にリストアする必要があります。

EXECUTE 区分を使用して監査を行う場合は、静的 SQL と動的 SQL の両方のステートメント・テキストが記録され、入力パラメーター・マーカーやホスト変数も記録されます。入力値を使用するかどうかにかかわらず、EXECUTE 区分を構成して監査できます。

注: グローバル変数は監査されません。

EXECUTE イベントの監査は、イベントの完了時に行われます (SELECT ステートメントの場合は、カーソルのクローズ時に行われます)。イベント完了時の状況も保管されます。EXECUTE イベントは完了時に監査されるため、長時間にわたって実行される照会は、すぐには監査ログに現れません。

注: ステートメントの準備は、実行の一部とは見なされません。ほとんどの許可検査は準備の際に実行されます (例えば SELECT 特権)。つまり、許可エラーが原因となって準備の途中で失敗するステートメントの場合は、EXECUTE イベントは生成されません。

特定の EXECUTE レコードごとに、「Statement Value Index」、「Statement Value Type」、および「Statement Value Data」フィールドが繰り返される場合があります。抽出によって生成されるレポート形式の場合は、各レコードに複数の値がリストされます。区切りファイル形式の場合は、複数の行が使用されます。最初の行には、STATEMENT のイベント・タイプが示され、値はありません。続く行には、SQL ステートメントに関連付けられているデータ値ごとに 1 行ずつ、DATA のイベント・タイプが示されます。STATEMENT 行と DATA 行は、イベント相関関係子とアプリケーション ID のフィールドを使用してリンクさせることができます。「Statement Text」、「Statement Isolation Level」、および「Compilation Environment Description」の列は、DATA イベントにはありません。

監査されるステートメント・テキストと入力データ値は、ディスクに保管される際、データベース・コード・ページに変換されます (監査されたフィールドはすべてデータベース・コード・ページで保管されます)。入力データのコード・ページとデータベース・コード・ページに互換性がなかった場合は、エラーは戻されず、変換されないままのデータが代わりにログとして記録されます。各データベースは独自の監査ログを持っているため、異なるコード・ページを持つデータベースがあっても、問題は起こりません。

ROLLBACK および COMMIT は、アプリケーションによって実行されたときに監査されます。また、BIND などの別のコマンドの一部として暗黙的に発行されたときも監査されます。

監査されている表へのアクセスがあったために EXECUTE イベントが監査された後、作業単位内で他にどのステートメントが実行されるかに影響を与えるステートメントがすべて監査されます。 COMMIT、ROLLBACK、ROLLBACK TO SAVEPOINT、および SAVEPOINT がこのようなステートメントに該当します。

「Savepoint ID」フィールド

どのステートメントが ROLLBACK TO SAVEPOINT ステートメントの影響を受けたかは、「Savepoint ID」フィールドを使用してトラッキングできます。通常の DML ステートメント (SELECT、INSERT など) では、現行のセーブポイント ID が監査されています。しかし、ROLLBACK TO SAVEPOINT ステートメントの場合は、代わりにロールバック先のセーブポイント ID が監査されます。したがって、次の例が示すように、ロールバック先の ID と同じかそれよりも大きなセーブポイント ID を持っているステートメントは、すべてロールバックされます。表は、ステートメントの実行の順序を示しています。2 以上のセーブポイント ID を持つイベントはすべてロールバックされます。値 3 (最初の INSERT ステートメントより) だけが、表 T1 に挿入されます。

表 6. ROLLBACK TO SAVEPOINT ステートメントの影響を示すためのステートメントのシーケンス

ステートメント	Savepoint ID
INSERT INTO T1 VALUES (3)	1
SAVEPOINT A	2
INSERT INTO T1 VALUES (5)	2
SAVEPOINT B	3
INSERT INTO T1 VALUES (6)	3
ROLLBACK TO SAVEPOINT A	2
COMMIT	

WITH DATA オプション

WITH DATA オプションが指定されている場合は、すべての入力値が監査されるわけではありません。LOB、LONG、XML、および構造化タイプのパラメーターは、NULL として示されます。

日付、時刻、およびタイム・スタンプのフィールドは、ISO 形式で記録されます。

あるポリシーで WITH DATA が指定されていて、SQL ステートメントの実行に関係するオブジェクトに関連付けられている別のポリシーで WITHOUT DATA が指定されている場合は、WITH DATA が優先され、その特定のステートメントに関してデータが監査されます。例えば、ユーザーに関連付けられている監査ポリシーでは WITHOUT DATA が指定されていて、表に関連付けられているポリシーでは WITH DATA が指定されている場合、そのユーザーがその表にアクセスするときは、ステートメントに使用される入力データが監査されます。

位置指定更新ステートメントや位置指定削除ステートメントでどの行が変更されたかは、判別できません。ログに記録されるのは基礎となる SELECT ステートメントの実行だけで、個々の FETCH は記録されません。ステートメントが発行されたときにカーソルがどの行にあったかを EXECUTE レコードから判別することは不可能です。後でステートメントを再現する際に唯一可能なのは、SELECT ステートメントを発行して、どの範囲の行が影響を受けている可能性があるかを確認することだけです。

過去のアクティビティーの再現の例

この例では、ある企業が、自社の包括的なセキュリティー・ポリシーの一環として、7 年前までさかのぼってデータベース内の特定の表に対する特定の要求の影響を分析できるようにしておく必要があるとします。これを行うために、この企業は、選択した任意の時点のデータベースを再構成できるような、週単位のバックアップをアーカイブするポリシーとそれに関連付けるログ・ファイルを設けます。関連する、リストアされたデータベースに対するあらゆる要求を再現し、分析できるようにしておくために、データベース監査には、データベースに対して行われたすべての要求に関する十分な情報をキャプチャーさせる必要があります。この要件は、静的 SQL ステートメントと動的 SQL ステートメントの両方を含みます。

この例は、SQL ステートメントの発行時に実施される必要のある監査ポリシーと、監査ログをアーカイブし、後でそれを抽出し、分析するためのステップを示しています。

1. EXECUTE 区分を監査する監査ポリシーを作成し、このポリシーをデータベースに適用します。

   ```
   CREATE AUDIT POLICY STATEMENTS CATEGORIES EXECUTE WITH DATA
      STATUS BOTH ERROR TYPE AUDIT
   COMMIT

   AUDIT DATABASE USING POLICY STATEMENTS
   COMMIT
   ```

2. 定期的に監査ログをアーカイブし、アーカイブ・コピーを作成します。

   一定の間隔 (ログに記録されるデータの量に応じて、週に一度、一日に一度など) で、セキュリティー管理者、または SYSPROC.AUDIT_ARCHIVE ストアード・プロシージャーの EXECUTE 特権をセキュリティー管理者によって付与されたユーザーによって、以下のステートメントが実行される必要があります。これらのアーカイブされたファイルは、どれほどの期間でも必要なだけ保持しておくことができます。2 つの入力パラメーターを使用して、AUDIT_ARCHIVE プロシージャーが呼び出されます。1 つはアーカイブ・ディレクトリーのパス、もう 1 つはアーカイブをすべてのメンバーで実行することを示す -2 です。

   ```
   CALL SYSPROC.AUDIT_ARCHIVE( '/auditarchive', -2 )
   ```

3. セキュリティー管理者、または SYSPROC.AUDIT_LIST_LOGS 表関数の EXECUTE 特権をセキュリティー管理者から付与されたユーザーは、AUDIT_LIST_LOGS を使用し、2006 年 4 月以降の入手可能な監査ログをすべて調べて、どのログに必要なデータが含まれている可能性があるかを判別します。

   ```
   SELECT FILE FROM TABLE(SYSPROC.AUDIT_LIST_LOGS('/auditarchive'))
      AS T WHERE FILE LIKE 'db2audit.dbname.log.0.200604%'
   FILE
   ```

```
------------------------------------
...
db2audit.dbname.log.0.20060418235612
db2audit.dbname.log.0.20060419234937
db2audit.dbname.log.0.20060420235128
```

4. この出力から、セキュリティー管理者は、必要なログが 1 つのファイル db2audit.dbname.log.20060419234937 に含まれていることを観察します。タイム・スタンプは、このファイルが、監査員が確認したいと思っている日の終わりにアーカイブされたファイルであることを示しています。

   セキュリティー管理者、または SYSPROC.AUDIT_DELIM_EXTRACT ストアード・プロシージャーの EXECUTE 特権をセキュリティー管理者によって付与されたユーザーは、このファイル名を AUDIT_DELIM_EXTRACT への入力として使用して、監査データを CSV ファイルに抽出します。これらのファイルの監査データは、DB2 データベース表にロードできます。監査員は、そのデータベース表でデータを分析し、調べている特定のステートメントを見つけることができます。監査員が調べているのは 1 つの SQL ステートメントだけであったとしても、そのステートメントに何らかの影響を与えるステートメントがある場合は、その作業単位内の複数のステートメントを調べることが必要になることもあります。

5. ステートメントを再現するためには、セキュリティー管理者は以下のアクションを行う必要があります。
   - 発行する抽出ステートメントを監査レコードから判別する。
   - ステートメントを発行したユーザーを監査レコードから判別する。
   - ユーザーがステートメントを発行したときのユーザーの許可 (すべての LBAC 保護を含む) を正確に再作成する。
   - 監査レコードのコンパイル環境に関する列を SET COMPILATION ENVIRONMENT ステートメントに組み合わせて使用し、コンパイル環境を再現する。
   - ステートメントが発行されたときの正確な状態に、データベースをリストアする。

   実動システムに支障が出ないようにするため、データベースのリストアやステートメントの再現は、2 次データベース・システム上で行います。ステートメントを発行したユーザーとして実行中のセキュリティー管理者は、「Statement Value Data」エレメントに指定された任意の入力変数をもとにステートメント・テキストから検出されたステートメントを再発行できます。

過去のアクティビティーの再生を有効にする:

包括的なセキュリティー・ポリシーの一環として、ある企業が、一定の年数までさかのぼってデータベース内の特定の表に対する特定の要求の影響を分析できるような機能を必要とすることがあります。

始める前に

企業は、選択した任意の時点でのデータベースを再構成できるように、週単位のバックアップとそれに関連するログ・ファイルをアーカイブするポリシーを確立する必要があります。

このタスクについて

将来の任意の時点で、該当するリストアされたデータベースに対する任意の要求を再生して分析できるようにするには、データベースに対するすべての要求についての十分なデータベース監査情報を収集する必要があります。この要件は、静的 SQL ステートメントと動的 SQL ステートメントの両方を扱うことができるものです。WITH DATA を使ってログに記録された EXECUTE 区分には、過去の SQL ステートメントの再生に必要な情報が含まれます (ただしデータベース内のデータがステートメント発行時の状態にリストアされることを想定しています)。

制約事項

以下のような権限と特権が必要です。

- SECADM 権限 (監査ポリシーを作成するために必要)
- EXECUTE 特権 (監査ルーチンおよびプロシージャーのために必要)

手順

過去のアクティビティーの再生を有効にするには、SECADM として、

1. EXECUTE 区分を監査する監査ポリシーを作成して、このポリシーをデータベースに適用します。

   ```
   CREATE AUDIT POLICY STATEMENTS CATEGORIES EXECUTE WITH DATA
      STATUS BOTH ERROR TYPE AUDIT
   COMMIT
   AUDIT DATABASE USING POLICY STATEMENTS
   COMMIT
   ```

2. 定期的に監査ログをアーカイブし、アーカイブ・コピーを作成します。 監査ログをアーカイブするには、以下のコマンドを定期的に実行してください。その際、アーカイブ・ディレクトリーへのパスを指定し、すべてのメンバーでアーカイブを実行することを指示する -2 を指定します。

   ```
   CALL SYSPROC.AUDIT_ARCHIVE( '/auditarchive', -2 )
   ```

3. 監査ログ・ファイルが作成されたことを確認します。 このようなアーカイブ・ファイルは、その後、企業のビジネス・ポリシーで定められた年数にわたって保持されます。 監査ログ・ファイルを確認するには、以下を実行します。

   ```
   SELECT FILE FROM SESSION.AUDIT_ARCHIVE_RESULTS
   ```

タスクの結果

これでデータと情報をアーカイブするように環境がセットアップされ、ログに記録されたデータベース・アクティビティーを将来の時点で再生できます。

過去のデータベース・アクティビティーの再生:

必要なすべてのデータ、ログ、および情報が使用可能であれば、過去のデータベース・アクティビティーを再生することができます。この参照トピックでは、SECADM が過去のデータベース・アクティビティーを再生する方法を例示します。

説明

ある時点で、企業の監査員が、過去に発生した特定ユーザーのアクティビティーを分析する必要があると診断したとします。 SECADM はバックアップ・データベー

ス・イメージ、バックアップ・ログ、および監査ログを組み合わせて使用することで、該当するデータベースを再構成し、監査員による分析の対象となるアクティビティーを再生することができます。例えば 2006 年 4 月 19 日に発生した特定ユーザーのアクティビティーが対象となる場合、SECADM は、以下に例示するような流れで監査員の分析を支援できます。

例

1. SECADM は AUDIT_LIST_LOGS を発行して、2006 年 4 月以降の使用可能なすべての監査ログを検出します。

```
SELECT FILE FROM TABLE(SYSPROC.AUDIT_LIST_LOGS('/auditarchive'))
   AS T WHERE FILE LIKE 'db2audit.db.sample.log.0.200604%'
FILENAME
---------------------------------------
...
db2audit.db.sample.log.0.20060418235612
db2audit.db.sample.log.0.20060419234937
db2audit.db.sample.log.0.20060420235128
```

2. この出力から、SECADM は、必要なログが db2audit.db.sample.log.20060419234937 ファイルに入っていると推測します。ログは 2006 年 4 月 19 日の営業日の終わりに取られたものです。
3. これが SYSPROC.AUDIT_DELIM_EXTRACT ストアード・プロシージャーの入力として使われます。プロシージャーに渡される引数は次のとおりです。
   - 区切り文字 (デフォルト)
   - 出力パス
   - アーカイブされた監査ログへのパス
   - 抽出されるファイルを判別するためのファイル名フィルター
   - 抽出されるそれぞれ区分の状況 (この場合、唯一の区分は EXECUTE です)

```
CALL SYSPROC.AUDIT_DELIM_EXTRACT( '', '', '/auditarchive',
   'db2audit.db.sample.log.0.20060419234937',
   'category execute' )
```

4. この時点で、監査データは区切り文字で区切られたファイルに入っています。SECADM は EXECUTE 区分から AUDITDATA.EXECUTE 表に監査データをロードします。以下を実行することにより、表を作成できます。

```
db2 CONNECT TO sample
db2 SET CURRENT SCHEMA AUDITDATA
db2 -tvf sqllib/misc/db2audit.ddl
```

5. 次に、execute.del から AUDITDATA.EXECUTE 表にデータをロードします。これを行うには、以下のコマンドを実行します。

```
db2 LOAD FROM FILE execute.del OF DEL MODIFIED BY LOBSINFILE
   INSERT INTO AUDITDATA.EXECUTE
```

6. これで、SECADM は AUDITDATA スキーマ内の監査表の中にすべての監査データを用意しました。このデータを分析して、監査員による調査の対象となっている特定のステートメントを見つけることができます。

   注: 監査員が調べているのは 1 つの SQL ステートメントだけであったとしても、そのステートメントに何らかの影響を与えるステートメントがある場合は、その作業単位内の複数のステートメントを調べることが必要になることもあります。

7. ステートメントを再生するには、以下のアクションを行う必要があります。

- 発行されたステートメントを監査レコードから正確に判別する必要があります。
- ステートメントを発行したユーザーを監査レコードから正確に判別する必要があります。
- ステートメント発行時のユーザーの許可を正確に再作成する必要があります (すべての LBAC 保護を含む)。
- コンパイル環境を再現する必要があります。そうするには、監査レコードの中のコンパイル環境列を SET COMPILATION ENVIRONMENT ステートメントと組み合わせて使用します。
- ステートメント発行時のデータベースの状態を正確に再作成する必要があります。

注: 実動システムに支障が出ないようにするため、データベースのリストアやステートメントの再生は、すべて 2 次データベース・システム上で行ってください。

8. SECADM は、ステートメントの実行が始まる時点までロールフォワードする必要があります。ステートメントのローカル開始時刻 (local_start_time) は EXECUTE 監査レコードに含まれています。例えば、以下のような EXECUTE 監査レコードを使用する場合、

```
timestamp=2006-04-10-13.20.51.029203;
  category=EXECUTE;
  audit event=STATEMENT;
  event correlator=1;
  event status=0;
  database=SAMPLE;
  userid=smith;
  authid=SMITH;
  session authid=SMITH;
  application id=*LOCAL.prodrig.060410172044;
  application name=myapp;
  package schema=NULLID;
  package name=SQLC2F0A;
  package section=201;
  uow id=2;
  activity id=3;
  statement invocation id=0;
  statement nesting level=0;
  statement text=SELECT * FROM DEPARTMENT WHERE DEPTNO = ? AND DEPTNAME = ?;
  statement isolation level=CS;
  compilation environment=
    isolation level=CS
    query optimization=5
    min_dec_div_3=NO
    degree=1
    sqlrules=DB2
    refresh age=+00000000000000.000000
    schema=SMITH
    maintained table type=SYSTEM
    resolution timestamp=2006-04-10-13.20.51.000000
    federated asynchrony=0;
  value index=0;
  value type=CHAR;
  value data=C01;
  value index=1;
  value type=VARCHAR;
  value index=INFORMATION CENTER;
  local_start_time=2006-04-10-13.20.51.021507;
```

ロールフォワード・ステートメントは次のようになります。

```
ROLLFORWARD DATABASE sample
TO 2006-04-10-13.20.51.021507
USING LOCAL TIME AND COMPLETE
```

9. さらに、コンパイル環境を設定する必要があります。コンパイル環境変数は SET COMPILATION ENVIRONMENT ステートメントを使って設定できます。ステートメントを発行したユーザーとして実行している SECADM は、「Statement Value Data」エレメントにある任意の入力変数を使って、ステートメント・テキストで検出されたステートメントを再生することができます。以下の C 組み込み SQL のサンプル・プログラムは COMPILATION ENVIRONMENT を設定して、監査員による分析対象の SELECT ステートメントを再生します。

```
EXEC SQL INCLUDE SQLCA;

EXEC SQL  BEGIN DECLARE SECTION;
     SQL TYPE IS BLOB(1M) hv_blob;
EXEC SQL  END DECLARE SECTION;

EXEC SQL DECLARE c1 CURSOR FOR SELECT COMPENVDESC
   FROM AUDITDATA.EXECUTE TIMESAMP= '2006-04-10-13.20.51.029203';
EXEC SQL DECLARE c2 CURSOR FOR SELECT *
   FROM DEPARTMENT
   WHERE DEPTNO = 'C01'
   AND DEPTNAME = 'INFORMATION CENTER';
EXEC SQL OPEN c1;

EXEC SQL FETCH c1 INTO :hv_blob;

EXEC SQL SET COMPILATION ENVIRONMENT :hv_blob;

EXEC SQL OPEN c2;

....

EXEC SQL CLOSE c1;
EXEC SQL CLOSE c2;
```

監査機能の管理

監査機能の動作

このトピックでは、監査レコードをログに書き込むタイミングがデータベースのパフォーマンスにどのように影響を与えることがあるか、監査機能内で起こるエラーをどのように管理するか、およびさまざまな状況においてどのように監査レコードが生成されるかを理解するのに役立つ、背景情報を提供します。

アクティブ・ログに監査レコードを書き込むタイミングの制御

アクティブ・ログへの監査レコードの書き込みは、そのレコードの生成の原因となるイベントの発生と同期的に、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、いつ監査レコードが書き込まれるかを決定します。

audit_buf_sz の値がゼロ (0) の場合、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待機のために、DB2 データベースのパフォーマンスが低下します。

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 ページの数となります。ディスクに書き込む前の複数の監査レコードを維持するために、内部バッファーが使用されます。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、動作を継続できます。

非同期の場合、空きがあるバッファーに監査レコードがしばらく保持される可能性があります。長時間にわたってこれが発生しないようにするために、データベース・マネージャーは定期的に監査レコードの書き込みを強制します。さらに、監査機能の許可ユーザーもまた、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。また、アーカイブ操作中は、バッファーが自動的にフラッシュされます。

エラーが発生したときには、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、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、エラーのために書き込みできない監査レコードは多くて 1 つなので、失われる可能性があるレコードは 1 つだけです。

監査機能のエラーの管理

ERRORTYPE 監査機能パラメーターの設定により、どのように DB2 データベース・システムと監査機能の間でエラーを管理するかを制御します。監査機能がアクティブであり、監査機能パラメーター ERRORTYPE の設定が AUDIT であるとき、監査機能は DB2 データベースの他の部分と同じように扱われます。監査レコードを (同期モードの場合はディスクに、非同期モードの場合は監査バッファーに) 書き込まないと、ステートメントに関連した監査イベントは正常終了したものと見なされません。このモードの実行時にエラーが検出されると、監査レコードを生成するステートメントに関する負の SQLCODE がアプリケーションに戻されます。

エラー・タイプが NORMAL に設定されると、 **db2audit** からのエラーはすべて無視され、操作の SQLCODE が戻されます。

さまざまな状況で生成される監査レコード

API または照会ステートメントや監査設定に応じて、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 つまたは複数の監査レコードが生成されたりします。例えば、SELECT 副照会による SQL UPDATE ステートメントの結果として、表の UPDATE 特権に対する許可検査の結果を含む 1 つの監査レコードと、表の SELECT 特権に対する許可検査の結果を含む別の監査レコードが生成される可能性があります。

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る時点ですべての許可検査の監査レコードが生成されます。同一ユーザーによるステートメントの再使用では許可検査されないため、再度監査されることはありません。ただし、特権情報を含んでいるカタログ表のいずれかが変更された場合には、次の作業単位において、キャッシュされた動的 SQL または XQuery ステートメントのステートメント特権が再び検査され、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唯一の監査可能イベントは、ユーザーがそのパッケージを実行する特権を持っているかどうかの許可検査です。パッケージ内の静的 SQL または XQuery ステートメント用に必要な許可検査および監査レコードの作成 (作成される場合) は、パッケージがプリコンパイルまたはバインドされるときに実行されます。パッケージ内部の静的 SQL または XQuery ステートメントの実行は、EXECUTE 区分を使用して監査可能です。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 つのパッケージが再びバインドされるとき、静的 SQL または XQuery ステートメントにより要求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。

許可検査が実行時に行われるステートメント (例えば、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、監査レコード内の (コンテキスト・イベントを除く) すべてのイベント用に記録されるセクション番号は、ステートメントの実際のセクション番号にかかわらずゼロ (0) にリセットされます。

監査機能のヒントと技法

監査を管理するベスト・プラクティスには、監査ログを定期的にアーカイブすること、監査ポリシーの作成時にエラー・タイプ AUDIT を使用すること、および以下に説明されている他のヒントを使用することが含まれます。

監査ログのアーカイブ

監査ログは定期的にアーカイブする必要があります。監査ログをアーカイブすると、現行監査ログはアーカイブ・ディレクトリーに移され、サーバーは新しいアクティブ監査ログの書き込みを開始します。各アーカイブ・ログの名前には、後で分析するために情報を必要とするログ・ファイルを識別するのに役立つタイム・スタンプが含まれています。

長期保管の場合、アーカイブ・ファイルのグループを zip 圧縮できます。

必要でなくなったアーカイブ監査ログについては、インスタンス所有者はファイルをオペレーティング・システムから削除するだけで済みます。

エラー処理

監査ポリシーを作成するときに、単にテスト監査ポリシーを作成しているのでなければ、エラー・タイプ AUDIT を使用する必要があります。例えば、エラー・タイプが AUDIT に設定されているときにディスク・スペースの不足などのエラーが発生すると、エラーが戻されます。エラー条件を修正してからでなければ、監査可能アクションをそれ以上続行できません。一方、エラー・タイプが NORMAL に設定されている場合、ロギングが失敗するだけで、エラーはユーザーに戻されません。エラーが発生しなかったかのように操作は続行します。

アーカイブの途中で、アーカイブ・パスのディスク・スペースがいっぱいになってしまった、あるいはアーカイブ・パスが存在しない、などの問題が発生した場合は、アーカイブ・プロセスは失敗し、監査ログ・データのパスに `.bk` という拡張子を持つ中間ログ・ファイルが生成されます (例えば

db2audit.instance.log.0.20070508172043640941.bk)。 (アーカイブ・パスに十分なディスク・スペースを割り振ることによって、またはアーカイブ・パスを作成することによって) 問題が解決されたなら、この中間ログをアーカイブ・パスに移動させてください。アーカイブ・パスに移動させた後は、この中間ログは正常にアーカイブされたログと同じ方法で扱うことができます。

DDL ステートメントの制限

AUDIT 排他 SQL ステートメントと呼ばれるいくつかのデータ定義言語 (DDL) ステートメントは、次の作業単位まで有効ではありません。そのため、以下の各ステートメントの直後に COMMIT ステートメントを使用することが推奨されています。

AUDIT 排他 SQL ステートメントは以下のとおりです。

- AUDIT
- CREATE AUDIT POLICY、ALTER AUDIT POLICY、および DROP AUDIT POLICY
- DROP ROLE および DROP TRUSTED CONTEXT (ドロップされるロールまたはトラステッド・コンテキストが監査ポリシーに関連付けられている場合)

アーカイブ・データを保持するための表形式は変わることがある

セキュリティー管理者なら SYSPROC.AUDIT_DEL_EXTRACT ストアード・プロシージャーを使用して、あるいはシステム管理者なら **db2audit extract** コマンドを使用して、アーカイブされた監査ログ・ファイルから区切りファイルに監査レコードを抽出することができます。区切りファイルのデータは、分析用の DB2 データベース表にロードできます。監査データを保持するために作成する必要がある表の形式は、リリースによって変わることがあります。

重要: スクリプト db2audit.ddl は、監査レコードを格納するための正しい形式の表を作成します。列が追加されたり、既存の列のサイズが変更されたりする場合もあるので、リリースごとに db2audit.ddl を実行するようにしなければなりません。

CHECKING イベントの使用

CHECKING イベントを操作しているとき、ほとんどの場合、監査レコードのオブジェクト・タイプのフィールドは、オブジェクトにアクセスしようとするユーザー ID が必要な特権または権限を持っているか検査する対象のオブジェクトとなります。例えば、ユーザーが 1 列を追加することによって表を ALTER しようとする場合、CHECKING イベントの監査レコードは、試みたアクセスが 『ALTER』 であり、検査されているオブジェクト・タイプが 『TABLE』 (検査されているのは表特権なので、列ではない) であったことを示します。

ただし、オブジェクトの CREATE、BIND、または DROP をユーザー ID に許可するデータベース権限があるかどうか検査する場合には、データベースに対して検査が行われますが、オブジェクト・タイプ・フィールドは (データベース自体ではなく) 作成、バインド、またはドロップする対象のオブジェクトを指定します。

表に索引を作成するとき、索引の作成特権が必要です。このため、 CHECKING イベントの監査レコードのアクセス試行タイプは 『CREATE』 ではなく 『INDEX』 になります。

パッケージのバインドについて作成される監査レコード

既存のパッケージをバインドしているとき、パッケージの DROP に対して OBJMAINT イベントの監査レコードが作成されます。さらに、パッケージの新しいコピーの CREATE に対して別の OBJMAINT イベントの監査レコードが作成されます。

ROLLBACK 後の CONTEXT イベント情報の使用

データ定義言語 (DDL) は、正常であるとしてログに記録される 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、イベントのロギングの後、エラーが原因で ROLLBACK が発生するかもしれません。その場合、オブジェクトが作成されないか、 GRANT または REVOKE 操作が不完全になります。このような場合に、CONTEXT イベントの使用が重要となります。このような CONTEXT イベントの監査レコード (特にイベントを終了するステートメント) は、操作試行の完了状態を示します。

ロード区切り文字

監査レコードを DB2 データベース表にロードするためにふさわしい区切り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を明確に指定しなければなりません。区切りファイルを抽出する際、これは、次のコマンドを使用して行うことができます。

```
db2audit extract delasc delimiter load_delimiter
```

この *load _delimiter* は、単一文字 (例えば ") または 16 進数の値で示される 4 バイト・ストリング (例えば 『0x3b』) です。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 0x3b
```

抽出時に区切り文字としてデフォルトのロード区切り文字以外を使用した場合、**LOAD** コマンドでは **MODIFIED BY** オプションを使用する必要があります。区切り文字として使用される 『0x3b』 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 chardel0x3b 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、" (二重引用符) がオーバーライドされます。

db2cluster コマンドのセキュリティー・モデル

db2cluster コマンドは DB2 クラスター・サービスとのメイン・インターフェースであり、IBM DB2 pureScale Feature に対して提供されているクラスター・マネージャーおよび共有ファイル・システム・クラスターの両方でそのように機能します。ユーザーが使用できる **db2cluster** コマンド・オプションは、ユーザーの権限によって異なります。

db2cluster コマンドのセキュリティー・モデルの観点からすると、ユーザー・グループは、各ユーザー・グループによって実行される可能性のあるタスクのタイプによって、以下の 3 つに分けられます。

- システムでユーザー ID を持つすべてのユーザー

  このグループのユーザーは、**db2cluster** コマンドを使用して DB2 pureScale インスタンスに関する情報を報告できます。ただし、変更は行えません。

- SYSADM、SYSCTL、または SYSMAINT グループ

  このグループのユーザーは、**db2cluster** コマンドを使用して、インスタンスの稼働状態を維持したり、クラスター・マネージャー上でいくつかの管理タスクを実行したりできます。定義上は、このグループのユーザーは、インスタンスのユーザー ID、インスタンス所有者の 1 次グループのメンバー、またはインスタンス所有者の 1 次グループ以外のメンバーのいずれかになります。 DB2 では、通常の日常的なアクティビティーは、インスタンス所有者の 1 次グループ以外のメンバーシップを持つユーザー ID を使用して実行することが勧められています。

- DB2 クラスター・サービス管理者

  このグループのユーザーには、データベース内のデータにアクセスするための要求事項はありません。これは管理役割であり、以下を行うために使用されます。

  – DB2 の DB2 クラスター・サービスの一部のインストールと構成
  – クラスター・ドメイン内のクラスター・インスタンスの保守および共有ファイル・システム・クラスターの保守

  DB2 クラスター・サービス管理者の役割は、オペレーティング・システムに対してルートが所有しているユーザー ID へのアクセス権限を持つエンド・ユーザー (例えばオペレーティング・システムの管理者) です。 DB2 クラスター・サービスは、DB2 pureScale Featureを使用しているか、統合化された HA を持つパーティション・データベース環境を使用しているかに関係なく、すべてのクラスター化環境に影響を与える可能性があります。そうなると、
  DBADM、SECADM、SQLADM、WLMADM、EXPLAIN、ACCESSCTRL、および DATAACCESS などの役割 (データベース上で機能する) によって、クラスター管理のための適切なレベルの権限が提供されなくなります。 DB2 クラスター・サービス管理者は、SYSADM、SYSCTL、または SYSMAINT グループのユーザー ID を持つユーザーと同じ人である場合があります。

  注: ただユーザーが SYSADM 特権を持っているというだけで、そのユーザーがオペレーティング・システムの管理特権を持つということにはなりません。

db2cluster のクラスター・マネージャー・タスク

- システムでユーザー ID を持つすべてのユーザーは、**-list** および **-verify** オプションを使用して、クラスター・ドメインの現行の状態に関する情報を取得できます。
- SYSADM、SYSMAINT、または SYSCTL グループのユーザーは、**-list** および **-set** オプションを使用して、優先 1 次クラスター・キャッシング・ファシリティーを照会および変更できます。また、これらのユーザーは、**-clear -alert** オプションを使用して、現行インスタンス (DB2INSTANCE レジストリー変数によって定義されている) 内のいずれかのホスト、メンバー、およびクラスター・キャッシング・ファシリティーのアラートをクリアできます。さらに、このグループのユーザーは、クラスター・リソースを作成および削除したり、クラスター・マネージャーのリソース・モデルを修復したりできます。ただし、それらの作業は、DB2 サービス担当員の指示の下でのみ行うよう強く勧められています。
- DB2 クラスター・サービス管理者は、クラスター・ドメイン内のすべてのホストのすべてのクラスター・インスタンスのDB2 クラスター・サービス全体に影響を与える管理タスクを実行できます。このユーザーは、タイブレーカー・デバイスやホスト障害検出時間の設定などの構成タスクを **-set** オプションを使用して実行できます。同様に、DB2 クラスター・サービス管理者は、**-enter** オプションを使用してホストを保守モードに移行させる、あるいは **-commit** オプションを使用して、クラスター・マネージャーに対する変更または更新をコミットするなどの保守関連のタスクを実行できます。さらに、このユーザーは、クラスター・マネージャーのピア・ドメインでの拡張保守操作も行います。例えば、ドメインの作成、削除、開始、停止や、ホストの追加または除去などです。ただし、それらの作業は、DB2 サービス担当員の指示の下でのみ行うよう強く勧められています。

db2cluster の共有ファイル・システム・タスク

- システムでユーザー ID を持つすべてのユーザーは、**-list** および **-verify** オプションを使用して、クラスター・ドメインの現行の状態に関する情報を取得できます。それらのユーザーは、**db2cluster** コマンド・オプションを使用して、さまざまなファイル・システム操作を実行することもできます。ただし、実行可能な操作は、通常のファイル・システム権限によって制限されています。コマンドを実行するユーザー ID に、使用されているデバイスの読み取りおよび書き込み権限がある場合、そのユーザーはファイル・システムを作成し、ディスクを追加できます。ファイル・システムが作成され、マウントされた後、そのファイル・システムへのアクセスは、それを作成したユーザー ID と DB2 クラスター・サービス管理者だけに限定され、それらのユーザーだけがファイル・システムの除去、削除、またはリバランスを行えるようになります。そのファイル・システムを作成したユーザー ID または DB2 クラスター・サービス管理者のいずれかは、通常のファイル・システムとほとんど同じように、他のユーザーがアクセスできるディレクトリーを作成できます。
- DB2 クラスター・サービス管理者は、クラスター・ドメイン内のすべてのホストのすべてのクラスター・インスタンスのDB2 クラスター・サービス全体に影響を与える管理タスクを実行できます。このユーザーは、**-set** オプションを使用して、タイブレーカー・デバイスの変更オプションを実行できます。同様に、DB2 クラスター・サービス管理者は、**-enter** オプションを使用してホストを保守モードに移行させる、あるいは **-commit** オプションを使用して共有ファイル・システ

ムに対する変更または更新をコミットするなどの保守関連のタスクを実行できます。さらに、このユーザーは、共有ファイル・システムのクラスターでの拡張保守操作も行います。例えば、ドメインの作成、削除、開始、停止や、ホストの追加または除去などです。ただし、それらの作業は、DB2 サービス担当員の指示の下でのみ行うよう強く勧められています。

第 2 章 ロール

ロールは、特権の管理を簡素化します。つまり、グループと同等の機能は提供されますが、同じ制約事項は設けられません。

ロールは、1 つ以上の特権をまとめたデータベース・オブジェクトですが、これを、GRANT ステートメントを使用してユーザー、グループ、PUBLIC、またはその他のロールに割り当てるか、あるいは、CREATE TRUSTED CONTEXT または ALTER TRUSTED CONTEXT ステートメントを使用して、トラステッド・コンテキストに割り当てることができます。ワークロード定義内で、SESSION_USER ROLE 接続属性用のロールを指定することができます。

ロールは、次のように、データベース・システム内での特権の管理がより簡単になるという利点を備えています。

- セキュリティー管理者は、組織の構造を映し出すような方法で、そのデータベースへのアクセスを制御することができます (組織における役職や担当業務に対応したロールをデータベース内に作成できます)。
- ユーザーには、その役職や担当業務に応じたロールに対するメンバーシップが付与されます。ユーザーの役職や担当業務の変更に応じて、ロールに対するメンバーシップを簡単に付与または取り消すことができます。
- 特権の割り当てが簡素化されます。管理者は、特定の役職や担当業務に該当する個々のユーザーに一連の同じ特権を付与するのではなく、その役職や担当業務に応じたロールに対してこの一連の特権を付与してから、その役職や担当業務に該当する各ユーザーにそのロールを付与することができます。
- そのロールを付与されたすべてのユーザーに対し、更新が適用されます。つまり管理者は、個人ごとに各ユーザーの特権を更新する必要はありません。
- ビュー、トリガー、マテリアライズ照会表 (MQT)、静的 SQL、および SQL ルーチンの作成時には、ロールに対して付与された特権および権限が常に使用されます。この場合、グループに付与された特権および権限は (直接でも間接にでも) 使用されません。

  その理由は、グループはサード・パーティー・ソフトウェア (例えば、オペレーティング・システムまたは LDAP ディレクトリー) によって管理されるので、DB2 データベース・システムは、グループ内のメンバーシップがいつ変更になったかを判別できないからです。ロールはデータベース内部で管理されるので、DB2 データベース・システムは、許可がいつ変更されたかを判別して、それに応じたアクションをとることができます。グループが検討の対象にならないのと同じ理由で、グループに付与されたロールも検討の対象にはなりません。
- ユーザーに割り当てられたすべてのロールは、ユーザーが接続を確立したときに有効になるので、ロールに付与されたすべての特権と許可も、ユーザーが接続するときに有効となります。ロールを明示的に有効または無効にすることはできません。
- セキュリティー管理者は、ロールの管理を他人に委任することができます。

データベース内で付与できるどの DB2 特権および権限でも、ロールに付与することができます。例えば、以下のどの権限および特権でも、ロールに付与することができます。

- DBADM、SECADM、DATAACCESS、ACCESSCTRL、SQLADM、WLMADM、LOAD、および IMPLICIT_SCHEMA データベース権限
- CONNECT、CREATETAB、CREATE_NOT_FENCED、BINDADD、CREATE_EXTERNAL_ROUTINE、または QUIESCE_CONNECT データベース権限
- 任意のデータベース・オブジェクト特権 (CONTROL を含む)

ユーザーがデータベースに接続したとき、そのユーザーのロールは自動的に有効になり、許可の検討の対象になります。つまり、SET ROLE ステートメントを使用してロールを活動化する必要はありません。例えば、ビュー、マテリアライズ照会表 (MQT)、トリガー、パッケージ、または SQL ルーチンを作成すると、ロールを通して取得した特権が適用されます。ただし、自分がメンバーとして所属するグループに付与されたロールを通して取得した特権は適用されません。

ロールには所有者はいません。セキュリティー管理者は、GRANT ステートメントの WITH ADMIN OPTION 節を使用して、ロールの管理を別のユーザーに委任することができます。それによって、他のユーザーがロールのメンバーシップを制御できるようになります。

制約事項

ロールの使用に関しては、次のようないくつかの制約事項があります。

- ロールはデータベース・オブジェクトを所有できません。
- 以下のデータベース・オブジェクトの作成時には、グループに付与された許可およびロールは検討の対象にはなりません。
  - 静的 SQL を格納するパッケージ。
  - ビュー
  - マテリアライズ照会表 (MQT)
  - トリガー
  - SQL ルーチン

  オブジェクトを作成するユーザーに対してか、または PUBLIC に対して直接または間接的に (例えばロール階層を介して) 付与されたロールだけが、上記のオブジェクトの作成時に検討の対象になります。

ロールのメンバーシップの作成と付与

セキュリティー管理者は、ロールの作成、ドロップ、付与、取り消し、およびコメント作成の権限を保有します。セキュリティー管理者は GRANT (ロール) ステートメントを使用して、ロール内のメンバーシップを許可 ID に付与し、REVOKE (ロール) ステートメントを使用して、ロール内のメンバーシップを許可 ID から取り消します。

セキュリティー管理者は、WITH ADMIN OPTION を持ったロール内のメンバーシップを許可 ID に付与することで、ロール内のメンバーシップの管理をその許可 ID

に委任することができます。 GRANT (ロール) ステートメントで WITH ADMIN OPTION 節を使用すれば、別のユーザーが以下を行えるようになります。

- ロールを他人に付与する。
- 他人のロールを取り消す。
- ロールに関するコメントを作成する。

WITH ADMIN OPTION 節を使って、以下を行うことはできません。

- ロールをドロップする。
- ロールの WITH ADMIN OPTION を許可 ID から取り消す。
- 他の誰かに WITH ADMIN OPTION を付与する (SECADM 権限の保有者でない場合)。

セキュリティー管理者がロールを作成した後、データベース管理者は GRANT ステートメントを使用して、権限および特権をそのロールに割り当てることができます。データベース内で付与できるどの DB2 特権および権限でも、ロールに付与することができます。SYSADM 権限などのインスタンスのレベルの権限をロールに割り当てることはできません。

セキュリティー管理者、またはあるロールに対しセキュリティー管理者から WITH ADMIN OPTION 付きでメンバーシップを付与されているすべてのユーザーは、GRANT (ロール) ステートメントを使用して、そのロールに対するメンバーシップを他のユーザー、グループ、PUBLIC、またはロールに付与することができます。ユーザーに対して直接、WITH ADMIN OPTION 付きであるロールのメンバーシップを付与することも、または PUBLIC、グループ、またはロールを通して間接的に付与することもできます。

ユーザーに割り当てられたすべてのロールは、そのユーザーがセッションを確立したときに有効になります。 DB2 データベース・システムでの許可の検査の際、ユーザーのロールに関連付けられたすべての特権と権限が考慮の対象になります。一部のデータベース・システムは、SET ROLE ステートメントを使用して特定のロールを活動化します。 DB2 データベース・システムは、SET ROLE ステートメントを使用する製品との互換性への配慮から、SET ROLE をサポートしています。 DB2 データベース・システムでは、セッション・ユーザーがロールのメンバーかどうかが SET ROLE ステートメントによって検査されて、メンバーでない場合は、エラーが戻されます。

あるロールでユーザーのメンバーシップを取り消す場合は、セキュリティー管理者、またはそのロールに対し WITH ADMIN OPTION 特権を保有するユーザーが、REVOKE (ロール) ステートメントを使用してこれを行います。

例

ロールは一連の特権を持っており、そのロールに対するメンバーシップを付与されたユーザーは、そのような特権を継承します。特権をこのように継承することによって、あるユーザーの特権を別のユーザーに再度割り当てるときに、特権を個別に管理する手間を省くことができます。ロールの使用により、必要な操作は、あるユーザーのロールに対するメンバーシップを取り消して、そのロールに対するメンバーシップを他のユーザーに付与するということだけで済みます。

例えば、DEV 部署に配属されている社員 BOB および ALICE は、表 SERVER、CLIENT、および TOOLS に対する SELECT 特権を持っているとします。ある日、上司が彼らを QA という別の部署に異動することを決定したので、データベース管理者は、表 SERVER、CLIENT、および TOOLS での選択を行う彼らの特権を取り消す必要が生じました。その後、部署 DEV には TOM という新入社員が配属されたので、データベース管理者は、表 SERVER 、CLIENT、および TOOLS に対する SELECT 特権を TOM に付与しなければなりません。

ロールを使用する場合、次のようなステップを行います。

1. セキュリティー管理者は、ロール DEVELOPER を次のようにして作成します。

   ```
   CREATE ROLE DEVELOPER
   ```

2. データベース管理者 (DBADM 権限の保有者) は、次のようにして、表 SERVER、CLIENT、および TOOLS に対する SELECT をロール DEVELOPER に付与します。

   ```
   GRANT  SELECT ON TABLE SERVER TO ROLE DEVELOPER
   GRANT  SELECT ON TABLE CLIENT TO ROLE DEVELOPER
   GRANT  SELECT ON TABLE TOOLS TO ROLE DEVELOPER
   ```

3. セキュリティー管理者は、次のようにして、部署 DEV 内のユーザー BOB と ALICE にロール DEVELOPER を付与します。

   ```
   GRANT ROLE DEVELOPER TO USER BOB, USER ALICE
   ```

4. BOB および ALICE が部署 DEV 外に配置換えになった場合、セキュリティー管理者は次のようにして、ロール DEVELOPER をユーザー BOB と ALICE から取り消します。

   ```
   REVOKE ROLE DEVELOPER FROM USER BOB, USER ALICE
   ```

5. TOM が部署 DEV に配属された場合、セキュリティー管理者は次のようにして、ロール DEVELOPER をユーザー TOM に付与します。

   ```
   GRANT ROLE DEVELOPER TO USER TOM
   ```

ロールの階層

ある 1 つのロールが別のロール内のメンバーシップを付与されると、ロール階層が形成されます。

ある 1 つのロールに対して、他のロールが付与されると、前者のロールの中に後者のロールが格納されます。後者のロールは、前者のロールのすべての特権を継承します。例えば、ロール DOCTOR がロール SURGEON に対して付与された場合、SURGEON には DOCTOR が格納されていることになります。ロール SURGEON は、ロール DOCTOR のすべての特権を継承します。

ロール階層内で循環を行うことはできません。*循環* が起きるのは、ある 1 つのロールが別のロールに付与されてから、後者のロールが前者のロールに付与されるという、循環方式でロールが付与される場合です。例えば、ロール DOCTOR がロール SURGEON に対して付与された後、ロール SURGEON が元のロール DOCTOR に戻って付与されることになります。ロール階層内で循環を確立した場合、エラーが戻されます (SQLSTATE 428GF)。

ロール階層の作成例

以下の例は、病院内の医療レベルを表すロール階層を作成する方法を示しています。

DOCTOR、SPECIALIST、および SURGEON という各ロールについて考察します。ロール階層を作成するには、循環を作成しないで、ロールを別のロールに付与します。ロール DOCTOR がロール SPECIALIST に付与され、ロール SPECIALIST がロール SURGEON に付与されます。

ロール SURGEON をロール DOCTOR に付与すると、循環が作成されるので、許可されません。

セキュリティー管理者は、次のような SQL ステートメントを実行して、ロール階層を作成します。

```
CREATE ROLE DOCTOR
CREATE ROLE SPECIALIST
CREATE ROLE SURGEON

GRANT ROLE DOCTOR TO ROLE SPECIALIST

GRANT ROLE SPECIALIST TO ROLE SURGEON
```

ロールからの特権の取り消しの効果

特権を取り消した場合、場合によっては、ビュー、パッケージ、またはトリガーなどの従属データベース・オブジェクトが無効になるかまたは作動不能になることがあります。

許可 ID から特定の特権を取り消す一方で、ロールを介するかまたは別の手段を講じて特権を維持した場合に、データベース・オブジェクトに何が起きるかを以下の例で示します。

ロールからの特権の取り消しの例

1. セキュリティー管理者は、次のように、ロール DEVELOPER を作成し、ユーザー BOB にこのロール内のメンバーシップを付与します。

   ```
   CREATE ROLE DEVELOPER
   GRANT ROLE DEVELOPER TO USER BOB
   ```

2. ユーザー ALICE が、次のように表 WORKITEM を作成します。

   ```
   CREATE TABLE WORKITEM (x int)
   ```

3. データベース管理者は、次のようにして、表 WORKITEM に対する SELECT および INSERT 特権を PUBLIC とロール DEVELOPER に付与します。

   ```
   GRANT SELECT ON TABLE ALICE.WORKITEM TO PUBLIC
   GRANT INSERT ON TABLE ALICE.WORKITEM TO PUBLIC
   GRANT SELECT ON TABLE ALICE.WORKITEM TO ROLE DEVELOPER
   GRANT INSERT ON TABLE ALICE.WORKITEM TO ROLE DEVELOPER
   ```

4. ユーザー BOB は、次のようにして、ビュー PROJECT を作成します。このビューは、表 WORKITEM と、表 WORKITEM に従属するパッケージ PKG1 を使用します。

   ```
   CREATE VIEW PROJECT AS SELECT * FROM ALICE.WORKITEM
   PREP emb001.sqc BINDFILE PACKAGE USING PKG1 VERSION 1
   ```

5. データベース管理者が、次のように、表 ALICE.WORKITEM に対する PUBLIC の SELECT 特権を取り消しても、ビュー BOB.PROJECT は操作可能のままで、パッケージ PKG1 も有効のままになります。なぜなら、ビューを定義した BOB はこれまでどおり、ロール DEVELOPER 内の自分のメンバーシップを通して必要な特権を保持しているからです。

   ```
   REVOKE SELECT ON TABLE ALICE.WORKITEM FROM PUBLIC
   ```

6. データベース管理者が、次のように、表 ALICE.WORKITEM に対するロール DEVELOPER の SELECT 特権を取り消した場合、ビュー BOB.PROJECT は作動不能になり、パッケージ PKG1 は無効になります。なぜなら、ビューとパッケージを定義した BOB は、他の手段を通して必要な特権を保有していないからです。

   ```
   REVOKE SELECT ON TABLE ALICE.WORKITEM FROM ROLE DEVELOPER
   ```

DBADM 権限の取り消しの例

以下の例では、ロール DEVELOPER は、DBADM 権限を保有していて、ユーザー BOB に付与されます。

1. セキュリティー管理者は、ロール DEVELOPER を次のようにして作成します。

   ```
   CREATE ROLE DEVELOPER
   ```

2. システム管理者は、次のように、DBADM 権限をロール DEVELOPER に付与します。

   ```
   GRANT DBADM ON DATABASE TO ROLE DEVELOPER
   ```

3. セキュリティー管理者は、次のようにして、ユーザー BOB にこのロール内のメンバーシップを付与します。

   ```
   GRANT ROLE DEVELOPER TO USER BOB
   ```

4. ユーザー ALICE が、次のように表 WORKITEM を作成します。

   ```
   CREATE TABLE WORKITEM (x int)
   ```

5. ユーザー BOB は、次のようにして、ビュー PROJECT を作成します。このビューは、表 WORKITEM、表 WORKITEM に従属するパッケージ PKG1、およびやはり表 WORKITEM に従属するトリガー TRG1 を使用します。

   ```
   CREATE VIEW PROJECT AS SELECT * FROM ALICE.WORKITEM
   PREP emb001.sqc BINDFILE PACKAGE USING PKG1 VERSION 1
   CREATE TRIGGER TRG1 AFTER DELETE ON ALICE.WORKITEM
                  FOR EACH STATEMENT MODE DB2SQL
          INSERT INTO ALICE.WORKITEM VALUES (1)
   ```

6. セキュリティー管理者は、次のように、ユーザー BOB のロール DEVELOPER を取り消します。

   ```
   REVOKE ROLE DEVELOPER FROM USER BOB
   ```

   ロール DEVELOPER を取り消すと、ユーザー BOB は DBADM 権限を喪失します。なぜなら、この権限の根拠であったロールが取り消されるからです。ビュー、パッケージ、およびトリガーは、次のような影響を受けます。

   - ビュー BOB.PROJECT はこれまでどおり有効です。
   - パッケージ PKG1 は無効になります。
   - トリガー BOB.TRG1 はこれまでどおり有効です。

ビュー BOB.PROJECT およびトリガー BOB.TRG1 は使用可能であるのに対して、パッケージ PKG1 は使用不可になります。 DBADM 権限が喪失しても、その DBADM 権限を保有していた許可 ID で作成されたビューおよびトリガー・オブジェクトは影響を受けません。

WITH ADMIN OPTION 節を使用したロール保守のデリゲート

セキュリティー管理者は、GRANT (ロール) SQL ステートメントで WITH ADMIN OPTION 節を使用すれば、ロール内のメンバーシップの管理を他の誰かに委任 (デリゲート) することができます。

WITH ADMIN OPTION 節を使えば、ロール内のメンバーシップを他のユーザーに付与する権限、ロールのメンバーのそのロール内のメンバーシップを取り消す権限、およびロールをドロップしないでそのロールに関するコメントを作成する権限を、別のユーザーに与えることができます。

WITH ADMIN OPTION 節は、ロールに対する WITH ADMIN OPTION を別のユーザーに付与する権限をどのユーザーにも与えません。また、ロールに対して別の許可 ID が持つ WITH ADMIN OPTION を取り消す権限を与えることもありません。

WITH ADMIN OPTION 節の使用を例示する例

1. セキュリティー管理者はロール DEVELOPER を作成し、次のように WITH ADMIN OPTION 節を使用して、この新しいロールをユーザー BOB に付与します。

   ```
   CREATE ROLE DEVELOPER
   GRANT ROLE DEVELOPER TO USER BOB WITH ADMIN OPTION
   ```

2. ユーザー BOB は、例えば他のユーザー ALICE を対象として、次のようにしてこのロール内のメンバーシップの付与および取り消しを行うことができます。

   ```
   GRANT ROLE DEVELOPER TO USER ALICE
   REVOKE ROLE DEVELOPER FROM USER ALICE
   ```

3. ユーザー BOB は、別のユーザーを対象としてロールのドロップや、WITH ADMIN OPTION の付与を行うことはできません (この 2 つの操作を実行できるのはセキュリティー管理者のみです)。 BOB が以下のコマンドを発行すると、失敗します。

   ```
   DROP ROLE DEVELOPER  - FAILURE!
                        - セキュリティー管理者のみが、
   ロールをドロップすることができます。
   GRANT ROLE DEVELOPER TO USER ALICE WITH ADMIN OPTION - FAILURE!
                        - セキュリティー管理者のみが、
   WITH ADMIN OPTION を付与することができます。
   ```

4. ユーザー BOB は、ロール DEVELOPER のユーザーのロール管理特権 (WITH ADMIN OPTION で与えられる) を取り消すことはできません。なぜなら、当人はセキュリティー管理者 (SECADM) 権限を持っていないからです。 BOB が以下のコマンドを発行すると、次のように失敗します。

   ```
   REVOKE ADMIN OPTION FOR ROLE DEVELOPER FROM USER SANJAY - FAILURE!
   ```

5. セキュリティー管理者は、ユーザー BOB のロール DEVELOPER (WITH ADMIN OPTION で与えられる) のロール管理特権を取り消すことはできますが、ユーザー BOB はこれまでどおりロール DEVELOPER を付与されたままになります。

   ```
   REVOKE ADMIN OPTION FOR ROLE DEVELOPER FROM USER BOB
   ```

   ただし、セキュリティー管理者が、次のようにユーザー BOB のロール DEVELOPER を単純に取り消した場合、BOB は、ロール DEVELOPER のメンバーであることによって与えられたすべての特権と、WITH ADMIN OPTION 節を通して与えられたロールに対する権限を喪失します。

   ```
   REVOKE ROLE DEVELOPER FROM USER BOB
   ```

グループに対するロールの比較

グループに付与された特権および権限は、ビュー、マテリアライズ照会表 (MQT)、SQL ルーチン、トリガー、および静的 SQL を格納したパッケージの作成時には検討の対象にはされません。この制約事項が適用されないようにするには、グループではなくロールを使用します。

ユーザーはロールを介して、DB2 データベース・システムで制御されるロールを通して取得した特権を使ってデータベース・オブジェクトを作成することができます。グループとユーザーは、例えば、オペレーティング・システムや LDAP サーバーなどによって、DB2 データベース・システムから外部的に制御されます。

グループの使用をロールに置き換える例

以下の例は、ロールを使用してグループを置き換える方法を示しています。

DEVELOPER_G、TESTER_G、および SALES_G という 3 つのグループがあると仮定します。以下の表に示されているとおり、ユーザー BOB、ALICE、および TOM は、これらのグループのメンバーであるとします。

表 7. グループおよびユーザーの例

グループ	このグループに所属するユーザー
DEVELOPER_G	BOB
TESTER_G	ALICE、TOM
SALES_G	ALICE、BOB

1. セキュリティー管理者は、グループの代わりに使用するロール DEVELOPER、TESTER、および SALES を作成します。

   ```
   CREATE ROLE DEVELOPER
   CREATE ROLE TESTER
   CREATE ROLE SALES
   ```

2. セキュリティー管理者は、次のように、これらのロール内のメンバーシップをユーザーに付与します (グループ内でユーザーのメンバーシップを設定するのは、システム管理者の責務です)。

   ```
   GRANT  ROLE DEVELOPER TO USER BOB
   GRANT  ROLE TESTER TO USER ALICE, USER TOM
   GRANT  ROLE SALES TO USER BOB, USER ALICE
   ```

3. データベース管理者は、次のように、グループが保有していたものに似た特権または権限をこのロールに付与することができます。

   ```
   GRANT privilege ON object TO ROLE DEVELOPER
   ```

   データベース管理者は次に、グループのこれらの特権を取り消すと同時に、グループをシステムからも除去するようシステム管理者に依頼することができます。

ロールを通して取得した特権を使用したトリガーの作成例

以下の例は、ユーザー BOB が、ロール DEVELOPER を通して必要な特権を保有しているときに、トリガー TRG1 を正常に作成できるという例を示しています。

1. まず、ユーザー ALICE が次のように表 WORKITEM を作成します。

   ```
   CREATE TABLE WORKITEM (x int)
   ```

2. 次に、ALICE の表を変更する特権が、データベース管理者によってロール DEVELOPER に付与されます。

   ```
   GRANT ALTER ON ALICE.WORKITEM TO ROLE DEVELOPER
   ```

3. ユーザー BOB は、ロール DEVELOPER のメンバーであるので、トリガー TRG1 を正常に作成できます。

   ```
   CREATE TRIGGER TRG1 AFTER DELETE ON ALICE.WORKITEM
             FOR EACH STATEMENT MODE DB2SQL INSERT INTO ALICE.WORKITEM VALUES (1)
   ```

IBM Informix Dynamic Server からのマイグレーション後のロールの使用

IBM Informix® Dynamic Server から DB2 データベース・システムにマイグレーションしてロールを使用する場合、気を付ける必要のある点がいくつかあります。

Informix Dynamic Server (IDS) の SQL ステートメント GRANT ROLE は、節 WITH GRANT OPTION を提供します。 DB2 データベース・システムの GRANT ROLE ステートメントは、同じ機能を備えた節 WITH ADMIN OPTION (これは SQL 標準に準拠します) を提供します。 IDS から DB2 データベース・システムへのマイグレーションでは、**dbschema** ツールが CREATE ROLE および GRANT ROLE ステートメントを生成した後、**dbschema** ツールが WITH GRANT OPTION のすべてのオカレンスを WITH ADMIN OPTION に置き換えます。

IDS データベース・システムでは、SET ROLE ステートメントは特定のロールを活動化します。 DB2 データベース・システムは、SET ROLE ステートメントをサポートしますが、その目的は、この SQL ステートメントを使用する他の製品との互換性を保つことに限られます。 SET ROLE ステートメントは、セッション・ユーザーがロールのメンバーかどうかを検査し、メンバーでなければ、エラーを戻します。

dbschema の出力例

ロール DEVELOPER、TESTER、および SALES が IDS データベースに格納されていると仮定します。ユーザー BOB、ALICE、および TOM は、それぞれ別々のロールを付与されています。つまり、ロール DEVELOPER は BOB に付与され、ロール TESTER は ALICE に付与され、ロール TESTER および SALES が TOM に付与されています。 DB2 データベース・システムにマイグレーションするには、

dbschema ツールを使用して、次のようにこのデータベース用の CREATE ROLE および GRANT ROLE ステートメントを作成します。

```
CREATE ROLE DEVELOPER
CREATE ROLE TESTER
CREATE ROLE SALES

GRANT  DEVELOPER TO BOB
GRANT  TESTER TO ALICE, TOM
GRANT  SALES TO TOM
```

DB2 データベース・システム内にデータベースを作成する必要があります。その後、そのデータベース内で上記のステートメントを実行し、ロールとそのロールの割り当てを再作成することができます。

第 3 章 トラステッド・コンテキストおよびトラステッド接続の使用

DB2 への接続が確立されている場合、アプリケーション内で要求を行うことにより、明示的トラステッド接続を確立できます。確立する接続の属性と一致する属性を指定した CREATE TRUSTED CONTEXT ステートメントを使用して、セキュリティー管理者がトラステッド・コンテキストを事前に定義しておくことが必要です (後述のステップ 1 を参照)。

始める前に

接続を確立する時の明示的トラステッド接続を要求するために使用する API は、使用するアプリケーションのタイプによって異なります (ステップ 2 の表を参照)。

明示的トラステッド接続を確立した後、アプリケーションで接続のユーザー ID を別のユーザー ID に切り替えることができます。ただしこれはアプリケーションのタイプに合った適切な API を使用して行われます (ステップ 3 の表を参照)。

手順

1. セキュリティー管理者は、CREATE TRUSTED CONTEXT ステートメントを使用してサーバーにトラステッド・コンテキストを定義します。 例:

```
CREATE TRUSTED CONTEXT MYTCX
       BASED UPON CONNECTION USING SYSTEM AUTHID NEWTON
       ATTRIBUTES (ADDRESS   '192.0.2.1')
       WITH USE FOR PUBLIC WITHOUT AUTHENTICATION
       ENABLE
```

2. トラステッド接続を確立するには、アプリケーションで以下のいずれかの API を使用します。

オプション	説明
アプリケーション	**API**
CLI/ODBC	SQLConnect、SQLSetConnectAttr
XA CLI/ODBC	Xa_open
JAVA	getDB2TrustedPooledConnection、 getDB2TrustedXAConnection

3. 別のユーザーに切り替えるには、認証のあるなしに関係なく、アプリケーションで以下のいずれかの API を使用します。

オプション	説明
アプリケーション	**API**
CLI/ODBC	SQLSetConnectAttr
XA CLI/ODBC	SQLSetConnectAttr
JAVA	getDB2Connection、reuseDB2Connection

オプション	説明
.NET	DB2Connection.ConnectionString キーワード: TrustedContextSystemUserID および TrustedContextSystemPassword

切り替えは新しいユーザー ID を認証して行う場合も認証せずに行う場合もありますが、それはこの明示的トラステッド接続に関連付けられたトラステッド・コンテキスト・オブジェクトの定義によって異なります。例えば、セキュリティー管理者が以下のトラステッド・コンテキスト・オブジェクトを作成すると想定します。

```
CREATE TRUSTED CONTEXT CTX1
 BASED UPON CONNECTION USING SYSTEM AUTHID USER1
 ATTRIBUTES (ADDRESS   '192.0.2.1')
  WITH USE FOR USER2 WITH AUTHENTICATION,
               USER3 WITHOUT AUTHENTICATION
  ENABLE
```

さらに、明示的トラステッド接続が確立されたと仮定します。 USER3 は、認証を必要としないトラステッド・コンテキスト CTX1 のユーザーとして定義されているので、トラステッド接続のユーザー ID を認証情報を提供せずに USER3 に切り替える要求は許可されます。ただし、USER2 は、認証情報を提供する必要があるトラステッド・コンテキスト CTX1 のユーザーとして定義されているので、トラステッド接続のユーザー ID を認証情報を提供せずに USER2 に切り替える要求は失敗します。

明示的トラステッド接続の確立とユーザーの切り替えの例

以下の例では、中間層サーバーは、エンド・ユーザーの代わりにいくつかのデータベース要求を発行する必要がありますが、そのエンド・ユーザーに代わってデータベース接続を確立するためのエンド・ユーザーの資格情報へのアクセス権限がありません。

中間層サーバーがデータベースへの明示的トラステッド接続を確立することを許可するトラステッド・コンテキスト・オブジェクトをデータベース・サーバー上に作成できます。明示的トラステッド接続の確立後、中間層サーバーは、その接続の現行ユーザー ID を、データベース・サーバーで新規ユーザー ID を認証する必要なく新規ユーザー ID に切り替えることができます。以下の CLI コード・スニペットは、前述のステップ 1 で定義されたトラステッド・コンテキスト MYTCX を使用してトラステッド接続を確立する方法と、認証なしでトラステッド接続のユーザーを切り替える方法を示しています。

```
int main(int argc, char *argv[])
{
 SQLHANDLE henv;         /* environment handle */
 SQLHANDLE hdbc1;       /* connection handle */
 char origUserid[10] = "newton";
  char password[10] = "test";
   char switchUserid[10] = "zurbie";
 char dbName[10] = "testdb";

// Allocate the handles
SQLAllocHandle( SQL_HANDLE_ENV, &henv );
SQLAllocHandle( SQL_HANDLE_DBC, &hdbc1 );
```

```
// Set the trusted connection attribute
SQLSetConnectAttr( hdbc1, SQL_ATTR_USE_TRUSTED_CONTEXT,
SQL_TRUE, SQL_IS_INTEGER );

// Establish a trusted connection
SQLConnect( hdbc1, dbName, SQL_NTS, origUserid, SQL_NTS,
password, SQL_NTS );

//Perform some work under user ID "newton"
. . . . . .

 // Commit the work
SQLEndTran(SQL_HANDLE_DBC, hdbc1, SQL_COMMIT);


// Switch the user ID on the trusted connection
SQLSetConnectAttr( hdbc1,
SQL_ATTR_TRUSTED_CONTEXT_USERID, switchUserid,
SQL_IS_POINTER
);

//Perform new work using user ID "zurbie"
. . . . . .

//Commit the work
SQLEndTranSQL_HANDLE_DBC, hdbc1, SQL_COMMIT);

// Disconnect from database
SQLDisconnect( hdbc1 );

 return 0;

}  /* end of main */
```

次のタスク

実際にユーザー ID が切り替わるのはどのタイミングか?

トラステッド接続上でユーザーを切り替えるコマンドが発行されても、次のステートメントがサーバーに送信されるまで、ユーザー切り替え要求は実行されません。次の例でこのことを示します。この例では、次のステートメントが発行されるまで、**list applications** コマンドで元のユーザー ID が表示されます。

1. USERID1 を使って明示的なトラステッド接続を確立します。
2. USERID2 に切り替えるように、**getDB2Connection** などのユーザー切り替えコマンドを発行します。
3. db2 list applications を実行します。まだ USERID1 が接続されていると表示されます。
4. このトラステッド接続で executeQuery("values current sqlid") のようなステートメントを発行し、サーバーでユーザー切り替え要求が実行されるようにします。
5. db2 list applications を再度実行します。今度は USERID2 が接続されていると表示されます。

トラステッド・コンテキストおよびトラステッド接続

トラステッド・コンテキストとは、データベースと外部エンティティー (アプリケーション・サーバーなど) の間の接続における信頼関係を定義するデータベース・オブジェクトのことをいいます。

信頼関係は、以下の属性のセットに基づいています。

- システム許可 ID: データベース接続を確立するユーザーを表します
- IP アドレス (またはドメイン・ネーム): データベース接続を確立するホストを表します
- データ・ストリーム暗号化: データベース・サーバーとデータベース・クライアントの間のデータ通信のための暗号化設定がある場合にはそれを表します

ユーザーがデータベース接続を確立するときに、DB2 データベース・システムは、接続がデータベース内のトラステッド・コンテキスト・オブジェクトの定義と一致するかどうかを検査します。一致していた場合、データベース接続は信頼できると見なされます。

トラステッド接続を使用すると、このトラステッド接続の起動側では、トラステッド接続の有効範囲外では使用できない追加機能を取得することができます。追加機能は、トラステッド接続が明示的であるか暗黙的であるかによって異なります。

明示的トラステッド接続の起動側には以下の機能があります。

- その接続の現行ユーザー ID を、認証のあるなしに関係なく別のユーザー ID に切り替える
- トラステッド・コンテキストのロール継承フィーチャーにより追加の特権を取得する

暗黙的トラステッド接続は、明示的に要求されていないトラステッド接続で、明示的トラステッド接続要求ではなく通常の接続要求により確立されます。暗黙接続を取得するためにアプリケーション・コードを変更する必要はありません。また、暗黙的トラステッド接続を取得するかどうかは、接続戻りコードには影響はありません (明示的トラステッド接続を要求する場合は、接続戻りコードは要求が成功したかどうかを示します)。暗黙的トラステッド接続の起動側は、トラステッド・コンテキストのロール継承フィーチャーにより追加の特権を取得できるだけで、ユーザー ID を切り替えることはできません。

トラステッド・コンテキストを使用するとどのようにセキュリティーが向上するか

3 層アプリケーション・モデルは、クライアント・アプリケーションとデータベース・サーバーの間に中間層を置くことにより、標準的な 2 層クライアントおよびサーバー・モデルを拡張します。このモデルは、Web ベースのテクノロジーや Java 2 Enterprise Edition (J2EE) プラットフォームの登場により、近年特に大きな人気を得ています。3 層アプリケーション・モデルをサポートするソフトウェア・プロダクトの一例として、IBM WebSphere® Application Server (WAS) があります。

3 層アプリケーション・モデルでは、クライアント・アプリケーションを実行するユーザーの認証、およびデータベース・サーバーとの相互作用の管理は、中間層が処理します。従来の方法では、データベース・サーバーとのすべての相互作用は、

データベース・サーバーに対して中間層を識別するユーザー ID と資格情報の組み合わせを使用して、その中間層により確立されたデータベース接続を介して行われます。つまり、データベース・サーバーは、中間層のユーザー ID に関連付けられたデータベース特権を使用して、すべてのデータベース・アクセスで行う必要がある許可検査および監査を行います。これには、ユーザーの代わりに中間層により実行されるアクセスも含まれます。

3 層アプリケーション・モデルには多くの利点がありますが、データベース・サーバーとのすべての相互作用 (例えば、ユーザー要求) を中間層の許可 ID で行うようにすると、いくつかのセキュリティー上の問題が生じます。これらを要約すると以下のようになります。

- ユーザー ID の消失

  企業によっては、アクセス制御の目的で、データベースにアクセスしている実際のユーザーの ID を知りたい場合があります。
- ユーザーの説明責任の減少

  監査による説明責任は、データベース・セキュリティーにおける基本原則です。ユーザーの ID が不明であると、中間層の固有の目的のために中間層により実行されるトランザクションと、ユーザーのために中間層により実行されるトランザクションを区別することが難しくなります。
- 中間層の許可 ID に特権を付与しすぎる

  中間層の許可 ID には、すべてのユーザーからのすべての要求を実行するために必要なすべての特権が含まれていなければなりません。これには、特定の情報にアクセスする必要がないユーザーがアクセス権限を取得できてしまうというセキュリティー問題があります。
- 弱いセキュリティー

  前述の特権の問題に加えて、現行方式では、接続するために中間層により使用される許可 ID には、ユーザー要求によりアクセスされる可能性があるすべてのリソースへの特権を付与する必要があります。中間層の許可 ID の暗号漏えいが発生すると、それらすべてのリソースは公開されてしまいます。
- 同じ接続を使用するユーザー間で相互に影響を及ぼす

  直前のユーザーによる変更が現行ユーザーに影響を与える場合があります。

明らかに、実際のユーザーの ID およびデータベース特権が、そのユーザーに代わって中間層により実行されるデータベース要求で使用されるようなメカニズムが必要です。この目標を達成するための最も簡単な方法は、中間層がユーザーの ID とパスワードを使用して新規接続を確立した後、ユーザーの要求をその接続を介して送信するというものです。この方法は単純ですが、以下に挙げるようないくつかの欠点があります。

- 特定の中間層では不適当。多くの中間層サーバーには、接続を確立するために必要なユーザー認証資格情報がありません。
- パフォーマンス上のオーバーヘッド。新しい物理接続を作成し、データベース・サーバーでユーザーを再認証することに関連した、パフォーマンス上の明らかなオーバーヘッドがあります。

- 保守上のオーバーヘッド。一元的なセキュリティー・セットアップ、またはシングル・サインオンを使用していない状況では、2 つのユーザー定義 (1 つは中間層上、もう 1 つはサーバー上) を持つことによる保守上のオーバーヘッドがあります。この状況では、異なる場所にあるパスワードを変更することが必要です。

トラステッド・コンテキスト機能は、この問題を解決します。セキュリティー管理者は、データベースと中間層の間の信頼関係を定義するトラステッド・コンテキスト・オブジェクトをデータベースに作成できます。その後、中間層ではデータベースへの明示的トラステッド接続を確立できますが、この接続では、接続の現行ユーザー ID を、認証のあるなしに関係なく別のユーザー ID に切り替える機能が中間層に付与されます。トラステッド・コンテキストは、エンド・ユーザーの ID アサーション問題を解決するだけでなく、別の利点もあります。それは、データベース・ユーザーが特権を使用できるようになる時期を制御する機能です。ユーザーが特権を使用できる時期を制御できないと、全体的なセキュリティーの低下につながります。例えば、特権が、最初に意図した目的以外で使用される場合があります。セキュリティー管理者は、1 つ以上の特権を 1 つのロールに割り当て、そのロールをトラステッド・コンテキスト・オブジェクトに割り当てることができます。そのトラステッド・コンテキストの定義と一致するトラステッド・データベース接続 (明示的または暗黙的) のみがそのロールに関連付けられた特権を利用できます。

パフォーマンスの向上

トラステッド接続を使用すると以下の利点があるため、パフォーマンスを最大限に発揮します。

- 接続の現行ユーザー ID が切り替わる時に新規接続は確立されません。
- トラステッド・コンテキスト定義が切り替え先のユーザー ID の認証を必要としない場合には、データベース・サーバーで新規ユーザーを認証することに関連したオーバーヘッドは発生しません。

トラステッド・コンテキストの作成例

セキュリティー管理者が以下のトラステッド・コンテキスト・オブジェクトを作成すると想定します。

```
CREATE TRUSTED CONTEXT CTX1
  BASED UPON CONNECTION USING SYSTEM AUTHID USER2
  ATTRIBUTES (ADDRESS   '192.0.2.1')
  DEFAULT ROLE managerRole
   ENABLE
```

ユーザー *user1* が IP アドレス 192.0.2.1 からトラステッド接続を要求した場合、DB2 データベース・システムは、トラステッド接続を確立できなかったためユーザー *user1* が非トラステッド接続を取得したことを示す警告 (SQLSTATE 01679、SQLCODE +20360) を戻します。しかし、ユーザー *user2* が IP アドレス 192.0.2.1 からトラステッド接続を要求した場合には、接続属性はトラステッド・コンテキスト CTX1 により条件が満たされるため、要求は受け入れられます。ユーザー *user2* はトラステッド接続を確立したため、そのユーザーはトラステッド・コンテキストのロール managerRole に関連付けられたすべての特権および権限を取得できます。このトラステッド接続の有効範囲外では、ユーザー *user2* はこれらの特権および権限を使用できません。

トラステッド・コンテキストを使用したロール・メンバーシップの継承

トラステッド接続の現行ユーザーは、トラステッド・コンテキストを使用したロールの自動継承により追加の特権を取得できます。ただしこれは、関連したトラステッド・コンテキスト定義の一部としてセキュリティー管理者により指定されている場合のみ該当します。

デフォルトでは、ロールはトラステッド接続のすべてのユーザーが継承できます。セキュリティー管理者は、トラステッド・コンテキスト定義を使用して特定のユーザーが継承するロールを指定することもできます。

トラステッド接続を使用している時にセッション許可 ID が保持できるアクティブなロールには以下のものがあります。

- セッション許可 ID が通常はメンバーであると見なされるロール。加えて
- トラステッド・コンテキストのデフォルトのロールまたはトラステッド・コンテキストのユーザー固有のロール (定義されている場合)

注:

- 接続に成功した時にセキュリティー・プラグインにより作成されるシステム許可 ID とセッション許可 ID が互いに異なるように作成されたカスタム・セキュリティー・プラグインを使用して、ユーザー認証を構成する場合には、トラステッド・コンテキストのロールは、その接続がトラステッド接続である場合であってもその接続を介して継承することはできません。
- ロールを使用して取得したトラステッド・コンテキスト特権は、動的 DML 操作においてのみ有効です。これらは、以下においては無効です。
  - DDL 操作
  - 非動的 SQL (BIND、REBIND、暗黙的な再バインド、追加バインドなどの静的 SQL ステートメントに関係した操作)

トラステッド・コンテキストのユーザー固有の特権の取得

セキュリティー管理者は、以下のようにするため、トラステッド・コンテキスト定義を使用してロールをトラステッド・コンテキストに関連付けることができます。

- デフォルトで、トラステッド接続のすべてのユーザーが、指定されたロールを継承できる
- トラステッド接続の特定のユーザーが、指定されたロールを継承できる

トラステッド接続のユーザーが新しい許可 ID に切り替わり、この新しい許可 ID にトラステッド・コンテキストのユーザー固有のロールが存在する場合、例で示されているように、ユーザー固有のロールがトラステッド・コンテキストのデフォルトのロールをオーバーライドします (トラステッド・コンテキストが存在する場合)。

デフォルトのロールおよびユーザー固有のロールを割り当てるトラステッド・コンテキストの作成例

セキュリティー管理者が以下のトラステッド・コンテキスト・オブジェクトを作成すると想定します。

```
CREATE TRUSTED CONTEXT CTX1
  BASED UPON CONNECTION USING SYSTEM AUTHID USER1
 ATTRIBUTES (ADDRESS   '192.0.2.1')
  WITH USE FOR USER2 WITH AUTHENTICATION,
               USER3 WITHOUT AUTHENTICATION
  DEFAULT ROLE AUDITOR
  ENABLE
```

USER1 がトラステッド接続を確立すると、ロール AUDITOR に付与された特権はこの許可 ID に継承されます。同様に、これらの同じ特権は、トラステッド接続の現行許可 ID が USER3 のユーザー ID に切り替わった時にこのユーザーにも継承されます。 (接続のユーザー ID がある時点で USER2 に切り替わった場合には、USER2 もトラステッド・コンテキストのデフォルトのロールである AUDITOR を継承します。) セキュリティー管理者は、トラステッド・コンテキストのデフォルトのロールとは異なるロールを USER3 に継承させるように選択することができます。これは、以下のように、このユーザーに固有のロールを割り当てることにより行えます。

```
CREATE TRUSTED CONTEXT CTX1
 BASED UPON CONNECTION USING SYSTEM AUTHID USER1
 ATTRIBUTES (ADDRESS   '192.0.2.1')
  WITH USE FOR USER2 WITH AUTHENTICATION,
               USER3 WITHOUT AUTHENTICATION ROLE OTHER_ROLE
  DEFAULT ROLE AUDITOR
  ENABLE
```

トラステッド接続の現行ユーザー ID が USER3 に切り替わっても、このユーザーはトラステッド・コンテキストのデフォルトのロールを継承しなくなります。代わりに、セキュリティー管理者によってそのユーザーに割り当てられた固有のロールである OTHER_ROLE を継承します。

明示的なトラステッド接続でユーザー ID を切り替えるための規則

明示的トラステッド接続では、接続のユーザー ID を別のユーザー ID に切り替えることができます。特定の規則が適用されます。

1. 切り替え要求が明示的トラステッド接続から出されず、切り替え要求が処理のためにサーバーに送信された場合、その接続はシャットダウンされ、エラー・メッセージが戻されます (SQLSTATE 08001、SQLCODE-30082、理由コード 41)。
2. 切り替え要求がトランザクション境界で出されず、トランザクションがロールバックされ、さらに切り替え要求が処理のためにサーバーに送信された場合、その接続は非接続状態になり、エラー・メッセージが戻されます (SQLSTATE 58009、SQLCODE -30020)。
3. 切り替え要求がストアード・プロシージャー内から出された場合、エラー・メッセージが戻され (SQLCODE -30090、理由コード 29)、これはこの環境では正しくない操作であることを示します。接続状態は維持され、接続は非接続状態にはなりません。以降の要求は処理できます。
4. 切り替え要求が (データベース接続ではなく) インスタンス接続のサーバーに送信された場合、その接続はシャットダウンされ、エラー・メッセージが戻されます (SQLCODE -30005)。

5. 切り替え要求がトラステッド接続で許可されない許可 ID で出された場合、エラー (SQLSTATE 42517、SQLCODE -20361) が戻され、接続は非接続状態になります。
6. トラステッド接続での切り替え要求が認証付きで許可される (WITH AUTHENTICATION が指定されている) 許可 ID で出されているのに適切な認証トークンが提供されていない場合、エラー (SQLSTATE 42517、SQLCODE -20361) が戻され、接続は非接続状態になります。
7. トラステッド接続に関連付けられたトラステッド・コンテキスト・オブジェクトが使用不可になり、そのトラステッド接続への切り替え要求が出された場合、エラー (SQLSTATE 42517、SQLCODE -20361) が戻され、接続は非接続状態になります。

   この場合、受け入れられる唯一のユーザー切り替え要求は、トラステッド接続を確立するユーザー ID または NULL ユーザー ID を指定する要求です。トラステッド接続を確立したユーザー ID への切り替えが行われた場合、このユーザー ID は、トラステッド・コンテキストのロール (トラステッド・コンテキストのデフォルトのロールもトラステッド・コンテキストのユーザー固有のロールも) を継承しません。
8. トラステッド接続に関連付けられたトラステッド・コンテキスト・オブジェクトのシステム許可 ID 属性が変更され、そのトラステッド接続への切り替え要求が出された場合、エラー (SQLSTATE 42517、SQLCODE -20361) が戻され、接続は非接続状態になります。

   この場合、受け入れられる唯一のユーザー切り替え要求は、トラステッド接続を確立するユーザー ID または NULL ユーザー ID を指定する要求です。トラステッド接続を確立したユーザー ID への切り替えが行われた場合、このユーザー ID は、トラステッド・コンテキストのロール (トラステッド・コンテキストのデフォルトのロールもトラステッド・コンテキストのユーザー固有のロールも) を継承しません。
9. トラステッド接続に関連付けられたトラステッド・コンテキスト・オブジェクトがドロップされ、そのトラステッド接続への切り替え要求が出された場合、エラー (SQLSTATE 42517、SQLCODE -20361) が戻され、接続は非接続状態になります。

   この場合、受け入れられる唯一のユーザー切り替え要求は、トラステッド接続を確立するユーザー ID または NULL ユーザー ID を指定する要求です。トラステッド接続を確立したユーザー ID への切り替えが行われた場合、このユーザー ID は、トラステッド・コンテキストのロール (トラステッド・コンテキストのデフォルトのロールもトラステッド・コンテキストのユーザー固有のロールも) を継承しません。
10. トラステッド接続での切り替え要求が許可されたユーザー ID で出されたにもかかわらず、そのユーザー ID がデータベースに対する CONNECT 特権を保持していない場合、その接続は非接続状態になり、エラー・メッセージが戻されます (SQLSTATE 08004、SQLCODE -1060)。
11. トラステッド・コンテキストのシステム許可 ID が WITH USE FOR 節に出現する場合、DB2 データベース・システムは、そのシステム許可 ID に戻すためのユーザー切り替え要求のシステム許可 ID の認証設定を受け入れます。トラ

ステッド・コンテキストのシステム許可 ID が WITH USE FOR 節に出現しない場合には、そのシステム許可 ID に戻すためのユーザー切り替え要求は認証なしの場合でも常に許可されます。

注: 接続が非接続状態になったときに、以下の要求だけは受け入れられ、「アプリケーションの状態にエラーがあります。データベース接続が存在しません。」(SQLCODE -900) エラーが戻されません。

- ユーザー切り替え要求
- COMMIT または ROLLBACK ステートメント
- DISCONNECT、CONNECT RESET、または CONNECT 要求

注: トラステッド接続のユーザー ID が新しいユーザー ID に切り替わると、古いユーザーの接続環境の痕跡はすべてなくなります。言い換えると、ユーザー ID の切り替えにより、環境は新しい接続環境と全く同一になります。例えば、接続における古いユーザー ID で TEMPORARY 表または WITH HOLD カーソルをオープンしている場合、その接続のユーザー ID が新しいユーザー ID に切り替わると、これらのオブジェクトは完全に失われます。

注: Java トラステッド接続には、非接続状態がありません。ユーザー切り替え操作が失敗すると、Java は例外をスローし、接続が切断されます。

トラステッド・コンテキストの問題判別

明示的トラステッド接続は、トラステッド接続の明示的で固有の要求により正常に確立された接続です。明示的トラステッド接続を要求した時にトラステッド接続を使用する資格がない場合には、通常の接続が確立され、警告が出されます (+20360)。

ユーザーがトラステッド接続を確立できなかった理由を判別するため、セキュリティー管理者は、システム・カタログにあるトラステッド・コンテキスト定義と、接続属性を調べる必要があります。特に、接続を確立した IP アドレス、データ・ストリームまたはネットワークの暗号化レベル、および接続を行うシステム許可 ID を調べます。 **db2pd** ユーティリティーの **-application** オプションはこの情報に加えて、以下の追加情報も戻します。

- 接続信頼タイプ: 接続がトラステッドであるかそうでないかを示します。接続がトラステッドである場合、これが明示的トラステッド接続であるか暗黙的トラステッド接続であるかも示します。
- トラステッド・コンテキスト名: トラステッド接続に関連付けられたトラステッド・コンテキストの名前。
- 継承したロール: トラステッド接続を介して継承したロール。

明示的なトラステッド接続を取得するのに失敗する最も一般的な原因は、以下のとおりです。

- DB2 サーバーとの通信にクライアント・アプリケーションが TCP/IP を使用していません。 TCP/IP は、トラステッド接続 (明示的または暗黙的) を確立するために使用できる DB2 サーバーとクライアント・アプリケーションが通信するためにサポートされる唯一のプロトコルです。
- データベース・サーバー認証タイプは、CLIENT に設定されます。

- データベース・サーバーに、使用可能なトラステッド・コンテキスト・オブジェクトがありません。着信接続の属性の突き合わせでトラステッド・コンテキストを対象とするには、そのトラステッド・コンテキスト・オブジェクトの定義で明示的に ENABLE を記述しなければなりません。
- データベース・サーバー上のトラステッド・コンテキスト・オブジェクトが、提示されるトラステッド属性と一致しません。例えば、以下のいずれかの状況が当てはまる場合があります。
  - 接続のシステム許可 ID が、どのトラステッド・コンテキスト・オブジェクトのシステム許可 ID とも一致しません。
  - 接続元の IP アドレスが、接続の対象となるトラステッド・コンテキスト・オブジェクトのどの IP アドレスとも一致しません。
  - 接続に使用されるデータ・ストリーム暗号化が、接続の対象となるトラステッド・コンテキスト・オブジェクトの ENCRYPTION 属性の値と一致しません。

  db2pd ツールを使用して、接続の確立元の IP アドレス、接続が使用するデータ・ストリームまたはネットワークの暗号化レベル、および接続を行うシステム許可 ID を調べることができます。 SYSCAT.CONTEXTS および SYSCAT.CONTEXTATTRIBUTES カタログ・ビューを調べると、特定のトラステッド・コンテキスト・オブジェクトの定義 (システム許可 ID、許可される IP アドレスのセット、および ENCRYPTION 属性の値など) を知ることができます。

ユーザー切り替えの失敗の最も一般的な原因は、以下のとおりです。

- 切り替え先のユーザー ID に、データベース上での CONNECT 特権がありません。この場合、SQL1060N が返されます。
- 明示的トラステッド接続と関連付けられたトラステッド・コンテキスト・オブジェクトの WITH USE FOR 節で、切り替え先のユーザー ID または PUBLIC が定義されていません。
- ユーザーの切り替えには認証が必要ですが、ユーザーが証明書を提示しなかったか、提示した証明書が正しくありません。
- ユーザー切り替え要求が行われたのは、トランザクション境界ではありません。
- トラステッド接続と関連付けられたトラステッド・コンテキストが、使用不可にされたか、ドロップされたか、変更されています。この場合、トラステッド接続を確立したユーザー ID への切り替えだけが許可されます。

第 4 章 行および列のアクセス制御 (RCAC) の概要

DB2 バージョン 10.1 では、データ・セキュリティーの追加のレイヤーとして、行および列のアクセス制御 (RCAC) が導入されました。行および列のアクセス制御は、粒度の細かいアクセス制御 (FGAC) とも呼ばれます。RCAC 制御は、行レベル、列レベル、またはその両方で、表にアクセスします。RCAC は、表特権モデルを補完するために使用できます。

さまざまな政府の規制を遵守するために、情報を適切に保護するための手続きと方法を導入する必要があります。組織内の個人は、各自のジョブ・タスクを実行するために必要なデータのサブセットに対するアクセスのみを許可されます。例えば、その地域の政府の規制により、医師は自分の患者のカルテを見ることはできても、それ以外の患者のカルテを見ることはできない場合があります。同じ規制により、患者が同意しない限り、医療サービスの提供者は、患者の個人情報 (例えば、自宅の電話番号) にアクセスできない場合もあります。

行および列のアクセス制御を使用すると、ユーザーが各自の仕事に必要なデータのみにアクセスできるようにできます。例えば、DB2 for Linux, UNIX, and Windows と RCAC を実行する病院のシステムでは、患者の情報とデータをフィルター操作して、特定の医師が必要とするデータのみが含まれるようにすることができます。その医師にとっては、他の患者は存在しないのと同じです。同様に、この病院の患者サービス担当員が患者の表を照会すると、患者の名前や電話番号の列は表示できますが、病歴の列はマスクされます。データがマスクされると、実際の病歴の代わりに、NULL または代替値が表示されます。

行および列のアクセス制御 (RCAC) には、以下の利点があります。

- 行および列のアクセス制御の規則を根本的に免除されているデータベース・ユーザーはいない。

  DATAACCESS 権限のあるユーザーのように、高いレベルの権限を持つユーザーも、これらの規則から免除されません。セキュリティー管理者 (SECADM) の権限を持つユーザーのみが、データベース内で行および列のアクセス制御を管理できます。したがって、RCAC を使用すると、DATAACCESS 権限を持つユーザーが、データベース内のすべてのデータに自由にアクセスできないようにすることができます。
- 表が SQL を介してどのようにアクセスされるかに関わらず、表データを保護できる。

  アプリケーション、応急の照会ツール、レポート生成ツールなどすべてに RCAC 規則が適用されます。規則の適用はデータ中心の方法で行われます。
- この追加のデータ・セキュリティー・レイヤーを活用するために、アプリケーションの変更は必要ない。

  すなわち、行および列レベルのアクセス制御は、既存のアプリケーションには明らかでない方法で確立および定義されます。しかし、RCAC は、これまでのような「何を要求するか」から「誰が何を要求するか」へ移行するという、重要なパラダイムのシフトを意味します。同一の照会に対する結果セットは、照会が行わ

れたコンテキストに応じて変化しますが、警告やエラーは戻されません。このような振る舞いは、まさにこのソリューションの意図を示しています。その意図するところは、照会に具体的な許可が与えられていない限り、その照会からは、表内のデータの全体像が見えないということを、アプリケーション設計者および DBA は意識すべきだということです。

行および列のアクセス制御 (RCAC) の規則

行および列のアクセス制御 (RCAC) は、データそれ自身に対するアクセス制御を、表レベルで設定します。この機能の実装の基礎になるものは、行および列に対して作成される SQL 規則です。

行および列のアクセス制御は、セキュリティー管理者がプライバシーおよびセキュリティー・ポリシーを管理するアクセス制御モデルです。RCAC は、すべてのユーザーに、代替ビューではなく、同じ表へのアクセスを許可します。ただし、RCAC では、表に関連付けられているポリシーで指定される、個々のユーザー権限または規則に基づいて表へのアクセスが制限されます。規則には、行に対する機能のセットと、列に対する機能のセットの 2 種類があります。

- 行の許可
  - 行の許可は、特定の表の行アクセス制御規則を記述するデータベース・オブジェクトです。
  - 行アクセス制御規則は、あるユーザーがアクセス権を持っている行のセットを記述する、SQL 検索条件です。
- 列マスク
  - 列マスクは、表内の特定の列に対する列アクセス制御規則を表現するデータベース・オブジェクトです。
  - 列アクセス制御規則は、あるユーザーに表示が許可されている列値、およびその許可の条件を記述する SQL CASE 式です。

RCAC 規則の管理のための SQL ステートメント

以下の SQL ステートメントを使用して、RCAC 規則を作成、変更、およびドロップできます。

RCAC 許可およびマスクを管理するための組み込み関数

以下の組み込みスカラー関数を使用して、許可およびマスクの条件を表現します。例えば、特定の行にアクセスするためには、ユーザーが 1 つ以上のロールに属していなければならない、あるいは 1 つ以上のグループに属していなければならないなどです。

シナリオ: ExampleHMO における行および列アクセス制御の使用

このシナリオでは、行および列のアクセス制御のユーザーとして、大規模で変動の大きい患者リストを管理する全国組織、ExampleHMO が登場します。ExampleHMO は、プライバシーとセキュリティーに関する政府の規制要件、およびマネジメント部門のビジネス目標をデータベース・ポリシーに反映させるために、行および列のアクセス制御を使用します。

ExampleHMO のように、患者の健康情報および個人情報を取り扱う組織は、政府のプライバシーおよびデータ保護の規制 (例えば、医療保険の相互運用性と説明責任に関する法律 (HIPAA)) を遵守する必要があります。それらのプライバシーおよびデータ保護の規制では、患者の重要な医療情報や個人情報が、適切な特権の所有者によってのみ共有、表示、および変更されることを保証しています。法令に違反した場合は、民事および刑事訴訟を含む、重い罰則が適用されます。

ExampleHMO は、組織のデータベース・システムに保管されているデータが保護されており、特権ユーザーのみがそのデータにアクセスできることを保証しなければなりません。標準的プライバシー規則に従って、特定の患者情報に対するアクセスおよび変更は、特権ユーザーのみに許可される必要があります。

シナリオ: ExampleHMO における行および列アクセス制御の使用 - セキュリティー・ポリシー

ExampleHMO は、DB2 データベースへのデータ・アクセスが、特定のセキュリティー・ポリシーに従って可能になるセキュリティー戦略を実装します。

このセキュリティー・ポリシーは、政府のプライバシーおよびデータ保護の規制に準拠します。最初の列はポリシー、および組織が直面する課題の概略です。2 番目の列は、その課題に対処するための、DB2 の行および列アクセス制御フィーチャーの概略です。

セキュリティーの課題	**セキュリティーの課題に対処するための行および列アクセス制御フィーチャー**
列へのアクセスを特権ユーザーのみに制限する。 例えば、パートナー企業の薬品研究者である Jane は、患者の重要な医療情報や、保険番号などの個人データを表示することはできません。	重要なデータを Jane から見えないようにフィルタリング、または隠すために、列マスクを使用できます。
行へのアクセスを特権ユーザーのみに制限する。Lee 医師は、自分の患者の患者情報を表示することのみ許可されています。ExampleHMO システム内のすべての患者の情報は表示できません。	特定の行を表示できるユーザーを制御するために、行の許可を実装できます。
必要性に基づいてデータを制限する。	行の許可を使用して、この課題に対処できます。また、ユーザー・レベルで、表レベルのデータを制限できます。
RCAC で保護されたデータ上のその他のデータベースオブジェクト (UDF、トリガー、ビューなど) を制限する。	行および列のアクセス制御は、データ・レベルでデータを保護します。行および列のアクセス制御ソリューションのこのデータ中心の性質により、UDF、トリガー、ビューなどのデータベース・オブジェクトにもセキュリティー・ポリシーが適用されます。

シナリオ: ExampleHMO における行および列アクセス制御の使用 - データベース・ユーザーおよびロール

このシナリオでは、何人かのユーザーが ExampleHMO のデータを作成、保護、および使用します。それらのユーザーは、それぞれ異なるユーザー権限およびデータベース権限を持っています。

ExampleHMO はセキュリティー戦略を実装して、データベースのデータにアクセスする方法をクラス分けしました。内部または外部からのデータへのアクセス権限は、データにアクセスするユーザーの職務およびアクセス権の分類に基づきます。ExampleHMO では、これらの職務を分類するために、以下のデータベース・ロールを作成しました。

PCP
初期治療医。

DRUG_RESEARCH
薬品研究者。

ACCOUNTING
経理担当者。

MEMBERSHIP
オプトインまたはオプトアウトで患者が加入するメンバー用。

PATIENT
患者。

以下のユーザーが、ExampleHMO のデータを作成、保護、および使用します。

Alex
ExampleHMO の主任セキュリティー管理者。SECADM 権限があります。

Peter
ExampleHMO のデータベース管理者。DBADM 権限があります。

Paul
ExampleHMO のデータベース開発者。トリガーおよびユーザー定義関数を作成する特権があります。

Lee 医師
ExampleHMO の医師。PCP ロールに属しています。

Jane
ExampleHMO のパートナーである Innovative Pharmaceutical Company の薬品研究者。DRUG_RESEARCH ロールに属しています。

John
ExampleHMO の会計部門。ACCOUNTING ロールに属しています。

Tom
ExampleHMO メンバーシップ役員。MEMBERSHIP ロールに属しています。

Bob
ExampleHMO の患者。PATIENT ロールに属しています。

このシナリオに示されている SQL ステートメントおよびコマンドの例を試す場合は、これらのユーザー ID と、リストされている権限を作成してください。

以下のサンプル SQL ステートメントは、システムでユーザーが作成済みであることを想定しています。これらの SQL ステートメントは、各ロールを作成して、ExampleHMO データベース内のさまざまな表に対する **SELECT** および **INSERT** 許可をユーザーに付与します。

```
--Creating roles and granting authority

CREATE ROLE PCP;

CREATE ROLE DRUG_RESEARCH;

CREATE ROLE ACCOUNTING;

CREATE ROLE MEMBERSHIP;

CREATE ROLE PATIENT;

GRANT ROLE PCP TO USER LEE;
GRANT ROLE DRUG_RESEARCH TO USER JANE;
GRANT ROLE ACCOUNTING  TO USER JOHN;
GRANT ROLE MEMBERSHIP TO USER TOM;
GRANT ROLE PATIENT TO USER BOB;
```

シナリオ: ExampleHMO における行および列アクセス制御の使用 - データベース表

このシナリオは、ExampleHMO データベースの 2 つの表 (PATIENT 表および PATIENTCHOICE 表) に注目します。

PATIENT 表には、患者の基本情報および健康情報が保管されます。このシナリオでは、PATIENT 表内の次の列について検討します。

SSN
患者の保険番号。患者の保険番号は、個人情報とみなされます。

NAME
患者の名前。患者の名前は、個人情報とみなされます。

ADDRESS
患者の住所。患者の住所は、個人情報とみなされます。

USERID
患者のデータベース ID。

PHARMACY
患者の医療情報。

ACCT_BALANCE
患者の請求書情報。

PCP_ID
患者の初期治療医のデータベース ID。

PATIENTCHOICE 表には、個々の患者のオプトインおよびオプトアウト情報が保管されています。この情報により、その患者が自分の健康情報を、この表に含まれている部外者に対して研究目的で開示するかどうかが決定されます。このシナリオでは、PATIENTCHOICE 表内の次の列について検討します。

SSN
患者の保険番号を使用して、患者とその選択項目を突き合わせます。

CHOICE
患者が選択できる選択項目の名前。

VALUE
患者がその選択項目に対して行った決定。

例えば、123-45-6789, drug_research, opt-in の行は、SSN が 123-45-6789 である患者が、自分の情報を、医療研究の目的で開示することに同意していることを示します。

以下の SQL ステートメントの例は、PATIENT、PATIENTCHOICE、および ACCT_HISTORY の各表を作成します。表に権限が付与され、データが挿入されます。

```
--Patient table storing information regarding patient
CREATE TABLE PATIENT (
 SSN CHAR(11),
 USERID VARCHAR(18),
 NAME VARCHAR(128),
 ADDRESS VARCHAR(128),
 PHARMACY VARCHAR(250),
 ACCT_BALANCE DECIMAL(12,2) WITH DEFAULT,
 PCP_ID VARCHAR(18)
 );


--Patientchoice table which stores what patient opts
--to expose regarding his health information

CREATE TABLE PATIENTCHOICE (
 SSN CHAR(11),
 CHOICE VARCHAR(128),
 VALUE VARCHAR(128)
);

--Log table to track account balance
CREATE TABLE ACCT_HISTORY(
 SSN   CHAR(11),
 BEFORE_BALANCE DECIMAL(12,2),
 AFTER_BALANCE DECIMAL(12,2),
 WHEN  DATE,
 BY_WHO VARCHAR(20)
);


--Grant authority

GRANT SELECT, UPDATE ON TABLE PATIENT TO ROLE PCP;

GRANT SELECT ON TABLE PATIENT TO ROLE DRUG_RESEARCH;

GRANT SELECT, UPDATE ON TABLE PATIENT TO ROLE ACCOUNTING;
GRANT SELECT ON TABLE ACCT_HISTORY TO ROLE ACCOUNTING;

GRANT SELECT, UPDATE, INSERT ON TABLE PATIENT TO ROLE MEMBERSHIP;
GRANT INSERT ON TABLE PATIENTCHOICE TO ROLE MEMBERSHIP;

GRANT SELECT ON TABLE PATIENT TO ROLE PATIENT;

GRANT SELECT, ALTER ON TABLE PATIENT TO USER ALEX;

GRANT ALTER, SELECT ON TABLE PATIENT TO USER PAUL;
GRANT INSERT ON TABLE ACCT_HISTORY TO USER PAUL;
```

```
--Insert patient data

INSERT INTO PATIENT
  VALUES('123-55-1234', 'MAX', 'Max', 'First Strt', 'hypertension', 89.70,'LEE');
INSERT INTO PATIENTCHOICE
  VALUES('123-55-1234', 'drug-research', 'opt-out');


INSERT INTO PATIENT
  VALUES('123-58-9812', 'MIKE', 'Mike', 'Long Strt', null, 8.30,'JAMES');
INSERT INTO PATIENTCHOICE
  VALUES('123-58-9812', 'drug-research', 'opt-out');


INSERT INTO PATIENT
  VALUES('123-11-9856', 'SAM', 'Sam', 'Big Strt', null, 0.00,'LEE');
INSERT INTO PATIENTCHOICE
  VALUES('123-11-9856', 'drug-research', 'opt-in');


INSERT INTO PATIENT
  VALUES('123-19-1454', 'DUG', 'Dug', 'Good Strt', null, 0.00,'JAMES');
INSERT INTO PATIENTCHOICE
  VALUES('123-19-1454', 'drug-research', 'opt-in');
```

シナリオ: ExampleHMO における行および列アクセス制御の使用 - セキュリティー管理

セキュリティー管理、およびセキュリティー管理者 (SECADM) ロールは、ExampleHMO の患者データと社内データを保護する上で重要な役割を果たします。ExampleHMO のマネジメントは、複数の人々にデータベース管理権限とセキュリティー管理権限を持たせることを決定しました。

ExampleHMO のマネジメント・チームは、データへの管理アクセス用のロールを作成することを決定しました。チームはまた、DATAACCESS 権限を持つユーザーであっても、保護されている健康データおよび個人データをデフォルトで表示できないようにすることを決定しました。

マネジメント・チームは、ExampleHMO の唯一のセキュリティー管理者として、Alex を選びます。今後は、Alex がすべてのデータ・アクセス権限を制御します。Alex はこの権限を使用して、行の許可、列マスク、関数やトリガーを保護するかどうか、などのセキュリティー規則を定義します。これらの規則により、特定のデータに対してどのユーザーが、自分の制御下でアクセスできるかが制御されます。

データベース管理者の Peter が必要な表を作成し、必要なロールをセットアップした後で、職務が分類されます。データベース管理とセキュリティー管理の職務は分けられ、Alex がセキュリティー管理者になります。

Peter はデータベースに接続し、Alex に SECADM 権限を付与します。Peter は現在、DBADM、DATAACCESS、および SECADM の各権限を保持しているので、SECADM 権限を付与することができます。

```
-- To seperate duties of security administrator from system administrator,
-- the SECADMN Peter grants SECADM authority to user Alex.

GRANT SECADM ON DATABASE TO USER ALEX;
```

Alex は、SECADM 権限を受け取った後、データベースに接続し、Peter のセキュリティー管理者の特権を取り消します。これで職務が分けられ、ExampleHMO の内外の他のユーザーに対してデータ・アクセス権を付与できる唯一の権限を、Alex が保持することになります。次の SQL ステートメントは、Alex が Peter の SECADM 権限を取り消した方法を示します。

```
--revokes the SECADMIN authority for Peter

REVOKE SECADM ON DATABASE FROM USER PETER;
```

シナリオ: ExampleHMO における行および列アクセス制御の使用 - 行の許可

セキュリティー管理者の Alex は、行および列アクセス制御の一部である、行の許可を使用して、ExampleHMO データベースへのデータ・アクセスの制限を開始します。行の許可は、ユーザーに返されるデータを行ごとにフィルタリングします。

患者は、自分のデータの表示を許可されています。医師は、自分の患者すべてのデータを表示できますが、他の医師にかかっている患者のデータは表示できません。MEMBERSHIP、ACCOUNTING、または DRUG_RESEARCH ロールに属するユーザーは、すべての患者の情報にアクセスできます。セキュリティー管理者の Alex は、これらの許可を実装して、特定の行にアクセスできるユーザーを、必要性に基づいて制限できるようにするよう依頼されます。

行の許可は、データベースにログオンしているユーザーに基づいて、行を制限またはフィルタリングします。ExampleHMO では、行の許可により、PATIENT という表に水平方向のデータ制約が作成されます。

Alex は、以下の行の許可を実装して、各ロールに属するユーザーが、表示の特権を持つ結果セットのみを表示できるように制限します。

```
CREATE PERMISSION ROW_ACCESS ON PATIENT
------------------------------------------------------
-- Accounting information:
-- ROLE PATIENT is allowed to access his or her own row
-- ROLE PCP is allowed to access his or her patients' rows
-- ROLE MEMBERSHIP, ACCOUNTING, and DRUG_RESEARCH are
-- allowed to access all rows
------------------------------------------------------------
FOR ROWS WHERE(VERIFY_ROLE_FOR_USER(SESSION_USER,'PATIENT') = 1
AND
PATIENT.USERID = SESSION_USER) OR
(VERIFY_ROLE_FOR_USER(SESSION_USER,'PCP') = 1
AND
PATIENT.PCP_ID = SESSION_USER) OR
 (VERIFY_ROLE_FOR_USER(SESSION_USER,'MEMBERSHIP') = 1 OR
 VERIFY_ROLE_FOR_USER(SESSION_USER,'ACCOUNTING') = 1 OR
 VERIFY_ROLE_FOR_USER(SESSION_USER, 'DRUG_RESEARCH') = 1)
ENFORCED FOR ALL ACCESS
ENABLE;
```

Alex は、行の許可を作成した後も、他の社員からすべてのデータが引き続き表示可能であることを確認します。行の許可は、それが定義された対象となる表に対してアクティブ化されるまでは、適用されません。 Alex は、次のようにして、許可をアクティブ化する必要があります。

```
--Activate row access control to implement row permissions

ALTER TABLE PATIENT ACTIVATE ROW ACCESS CONTROL;
```

シナリオ: ExampleHMO における行および列アクセス制御の使用 - 列マスク

セキュリティー管理者の Alex は、行および列アクセス制御の一部である、列マスクを使用して、ExampleHMO データベースへのデータ・アクセスの制限を強化します。列マスクは、データの表示を許可されていないユーザーに対して、戻されるデータを列ごとに隠します。

患者の支払いの詳細は、会計部門のユーザーからのみアクセス可能にしなければなりません。勘定残高は、その他のデータベース・ユーザーから見られてはなりません。Alex は、ACCOUNTING ロールに属さないユーザーからのアクセスを防止するように依頼されます。

Alex は、以下の列マスクを実装して、各ロールに属するユーザーが、表示の特権を持つ結果セットのみを表示できるように制限します。

```
--Create a Column MASK ON ACCT_BALANCE column on the PATIENT table

CREATE MASK ACCT_BALANCE_MASK ON PATIENT FOR
-------------------------------------------------------------
-- Accounting information:
-- Role ACCOUNTING is allowed to access the full information
-- on column ACCT_BALANCE.
-- Other roles accessing this column will strictly view a
-- zero value.
-------------------------------------------------------------
COLUMN ACCT_BALANCE RETURN
 CASE WHEN VERIFY_ROLE_FOR_USER(SESSION_USER,'ACCOUNTING') = 1
   THEN ACCT_BALANCE
  ELSE 0.00
 END
ENABLE;
```

Alex は、列マスクを作成した後も、他の社員からデータが引き続き表示可能であることを確認します。列マスクは、それが定義された対象となる表に対してアクティブ化されるまでは、適用されません。 Alex は、次のようにして、マスクをアクティブ化する必要があります。

```
--Activate column access control to implement column masks

ALTER TABLE PATIENT ACTIVATE COLUMN ACCESS CONTROL;
```

Alex は、マネジメントから、患者の保険番号を表示しないよう依頼されます。患者、医師、会計担当者、または MEMBERSHIP ロールに属するユーザーのみが SSN 列を表示できるようにします。

また、患者の PHARMACY の詳細を保護するため、PHARMACY 列の情報が、薬品研究者または医師によってのみ表示できるようにする必要があります。薬品研究者は、患者が情報の開示に同意した場合のみ、データを見ることができます。

Alex は、以下の列マスクを実装して、各ロールに属するユーザーが、表示の特権を持つ結果セットのみを表示できるように制限します。

```
CREATE MASK SSN_MASK ON PATIENT FOR
-----------------------------------------------------
-- Personal contact information:
-- Roles PATIENT, PCP, MEMBERSHIP, and ACCOUNTING are allowed
-- to access the full information on columns SSN, USERID, NAME,
-- and ADDRESS. Other roles accessing these columns will
-- strictly view a masked value.
-----------------------------------------------------
COLUMN SSN RETURN
 CASE WHEN
  VERIFY_ROLE_FOR_USER(SESSION_USER,'PATIENT') = 1 OR
  VERIFY_ROLE_FOR_USER(SESSION_USER,'PCP') = 1 OR
  VERIFY_ROLE_FOR_USER(SESSION_USER,'MEMBERSHIP') = 1 OR
  VERIFY_ROLE_FOR_USER(SESSION_USER,'ACCOUNTING') = 1
 THEN SSN
  ELSE CHAR('XXX-XX-' || SUBSTR(SSN,8,4)) END
ENABLE;

CREATE MASK PHARMACY_MASK ON PATIENT FOR
-----------------------------------------------------
-- Medical information:
-- Role PCP is allowed to access the full information on
-- column PHARMACY.
-- For the purposes of drug research, Role DRUG_RESEARCH can
-- conditionally see a patient's medical information
-- provided that the patient has opted-in.
-- In all other cases, null values are rendered as column
-- values.
-----------------------------------------------------
COLUMN PHARMACY RETURN
  CASE WHEN
  VERIFY_ROLE_FOR_USER(SESSION_USER,'PCP') = 1 OR
  (VERIFY_ROLE_FOR_USER(SESSION_USER,'DRUG_RESEARCH')=1
  AND
   EXISTS (SELECT 1 FROM PATIENTCHOICE C
 WHERE PATIENT.SSN = C.SSN AND C.CHOICE = 'drug-research' AND C.VALUE = 'opt-in'))
  THEN PHARMACY
  ELSE NULL
END
ENABLE;
```

Alex は、これら 2 つの列マスクを作成した後で、意図したユーザーによってのみデータが表示可能であることを確認します。PATIENT 表では、すでに列アクセス制御がアクティブ化されています。

シナリオ: ExampleHMO における行および列アクセス制御の使用 - データの挿入

病院が新規の患者を治療のために受け入れると、ExampleHMO データベースに、新規の患者レコードを追加する必要があります。

Bob は新しい患者で、彼の記録が ExampleHMO データベースに追加されます。必要なセキュリティー権限を持つユーザーが Bob の新規レコードを作成する必要があります。ExampleHMO メンバーシップ部門に属し、MEMBERSHIP ロールを持つ Tom が、Bob を新規メンバーとして登録します。Tom は、ExampleHMO データベースに接続すると、以下の SQL ステートメントを実行して、Bob を ExampleHMO データベースに追加します。

```
INSERT INTO PATIENT
  VALUES('123-45-6789', 'BOB', 'Bob', '123 Some St.', 'hypertension', 9.00,'LEE');
INSERT INTO PATIENTCHOICE
  VALUES('123-45-6789', 'drug-research', 'opt-in');
```

Tom は、ExampleHMO データベースの PATIENT 表で Bob を照会して、彼がデータベースに追加されていることを確認します。

```
Select * FROM PATIENT WHERE NAME = 'Bob';

SSN         USERID    NAME    ADDRESS       PHARMACY     ACCT_BALANCE  PCP_ID

----------- --------- ------- ------------- ------------ ------------- ------
123-45-6789 BOB       Bob     123 Some St.  XXXXXXXXXXX  0.00          LEE
```

シナリオ: ExampleHMO における行および列アクセス制御の使用 - データ更新

入院中に、Bob の治療法が変更されます。その結果、ExampleHMO データベース内の彼のレコードを更新しなければなりません。

Bob の担当医師である Lee 医師は、治療法の変更を勧め、Bob の薬を変更します。ExampleHMO システムの Bob のレコードは更新する必要があります。ExampleHMO データベースに設定されている、行の許可の規則によれば、ある行のデータを表示できないユーザーはその行内のデータを更新できません。Bob の PCPID には Lee 医師の ID が含まれており、行の許可が設定されているため、Lee 医師は Bob のレコードに対して、以下のサンプル SQL ステートメントを使用することにより、表示と更新の両方を実行できます。

```
UPDATE PATIENT SET PHARMACY = 'codeine' WHERE NAME = 'Bob';
```

Lee 医師は、以下の方法で更新を確認します。

```
Select * FROM PATIENT WHERE NAME = 'Bob';

SSN         USERID    NAME    ADDRESS      PHARMACY    ACCT_BALANCE   PCP_ID

----------- --------- ------- ------------ ----------- -------------- ------
123-45-6789 BOB       Bob     123 Some St. codeine     0.00           LEE
```

患者の Dug は、Lee 医師の同僚の James 医師による治療を受けています。Lee 医師は、Dug のレコードにも同様の更新を試みます。

```
UPDATE PATIENT SET PHARMACY = 'codeine' WHERE NAME = 'Dug';
SQL0100W  No row was found for FETCH, UPDATE or DELETE; or the result of a query
 is an empty table. SQLSTATE=02000
```

Dug の PCPID には Lee 医師の ID が含まれておらず、行の許可が設定されているため、Lee 医師は Dug のレコードを表示することも、更新することもできません。

シナリオ: ExampleHMO における行および列アクセス制御の使用 - データ照会

行および列のアクセス制御を使用すると、同一のデータベース照会で、異なるロールに属するユーザーが、異なる結果セットを受け取ることができます。例えば、DATAACCESS 権限を持つデータベース管理者の Peter は、PATIENT 表のデータをまったく見ることができません。

Peter、Bob、Lee 医師、Tom、Jane、および John がそれぞれデータベースに接続して、次の SQL 照会を試みます。

```
SELECT SSN, USERID, NAME, ADDRESS, PHARMACY, ACCT_BALANCE, PCP_ID FROM PATIENT;
```

照会の結果は、だれが照会を実行するかによって異なります。これらの照会には、Alex が作成した、行および列のアクセス制御の規則が適用されます。

これは Peter に表示される結果セットです。

```
SSN         USERID    NAME      ADDRESS     PHARMACY    ACC_BALANCE PCP_ID

----------- --------- --------- ----------- ----------- ----------- -----------

  0 record(s) selected.
```

表にはデータが含まれており、Peter はデータベース管理者ですが、彼にはデータを表示する権限がありません。

これは Bob に表示される結果セットです。

```
SSN         USERID    NAME     ADDRESS     PHARMACY    ACC_BALANCE PCP_ID

----------- --------- --------- ----------- ----------- ----------- ------
123-45-6789 BOB       Bob      123 Some St.XXXXXXXXXX 0.00        LEE

1 record(s) selected.
```

Bob は患者なので、自分のデータしか見ることができません。Bob は PATIENT ロールに属しています。PHARMACY 列と ACC_BALANCE 列のデータは、彼からは隠されています。

これは Lee 医師に表示される結果セットです。

```
SSN         USERID   NAME     ADDRESS     PHARMACY            ACC_BALANCE PCP_ID

----------- -------- -------- ----------- ------------------- ----------- ------
123-55-1234 MAX      Max      First Strt  hypertension        0.00        LEE
123-11-9856 SAM      Sam      Big Strt    High blood pressure 0.00        LEE
123-45-6789 BOB      Bob      123 Some St.codeine             0.00        LEE

3 record(s) selected.
```

Lee 医師は、自分が担当する患者のデータだけを見ることができます。 Lee 医師は PCP ロールに属しています。ACC_BALANCE 列のデータは、彼からは隠されています。

これは Tom に表示される結果セットです。

```
SSN         USERID   NAME     ADDRESS     PHARMACY    ACC_BALANCE PCP_ID

----------- -------- -------- ----------- ----------- ----------- -----------
123-55-1234 MAX      Max      First Strt  XXXXXXXXXX 0.00        LEE
123-58-9812 MIKE     Mike     Long Strt   XXXXXXXXXX 0.00        JAMES
123-11-9856 SAM      Sam      Big Strt    XXXXXXXXXX 0.00        LEE
123-19-1454 DUG      Dug      Good Strt   XXXXXXXXXX 0.00        JAMES
123-45-6789 BOB      Bob      123 Some St.XXXXXXXXXX 0.00        LEE

5 record(s) selected.
```

Tom はすべてのメンバーを見ることができます。Tom は MEMBERSHIP ロールに属しています。彼には PHARMACY 列および ACC_BALANCE 列のデータを見る特権がありません。

これは Jane に表示される結果セットです。

```
SSN         USERID   NAME     ADDRESS     PHARMACY            ACC_BALANCE PCP_ID

----------- -------- -------- ----------- ------------------- ----------- -------
XXX-XX-1234 MAX      Max      First Strt  XXXXXXXXXX          0.00        LEE
XXX-XX-9812 MIKE     Mike     Long Strt   XXXXXXXXXX          0.00        JAMES
XXX-XX-9856 SAM      Sam      Big Strt    High blood pressure 0.00        LEE
XXX-XX-1454 DUG      Dug      Good Strt   Influenza           0.00        JAMES
XXX-XX-6789 BOB      Bob      123 Some St.codeine             0.00        LEE

5 record(s) selected.
```

Jane はすべてのメンバーを見ることができます。DRUG_RESEARCH ロールに属しています。SSN 列および ACC_BALANCE 列のデータは、彼女からは隠されています。PHARMACY データは、患者が薬品研究会社とデータを共有することを選択した場合のみ、表示可能になります。

これは John に表示される結果セットです。

```
SSN         USERID   NAME     ADDRESS     PHARMACY    ACC_BALANCE PCP_ID

----------- -------- -------- ----------- ----------- ----------- --------
123-55-1234 MAX      Max      First Strt  XXXXXXXXXX  89.70       LEE
123-58-9812 MIKE     Mike     Long Strt   XXXXXXXXXX  8.30        JAMES
123-11-9856 SAM      Sam      Big Strt    XXXXXXXXXX  0.00        LEE
123-19-1454 DUG      Dug      Good Strt   XXXXXXXXXX  0.00        JAMES
123-45-6789 BOB      Bob      123 Some St.XXXXXXXXXX  9.00        LEE

5 record(s) selected.
```

John はすべてのメンバーを見ることができます。ACCOUNTING ロールに属しています。PHARMACY 列のデータは、彼からは隠されています。

シナリオ: ExampleHMO における行および列アクセス制御の使用 - ビューの作成

行および列のアクセス制御が定義されている表に対して、ビューを作成することができます。セキュリティー管理者の Alex は、医療研究者が使用できるビューを PATIENT 表に対して作成するよう依頼されます。

ExampleHMO と協力関係にある研究者は、患者がそれを認めている場合、患者の限定されたデータにアクセスできます。Alex と IT チームは、研究に関連する特定の患者情報をリストするビューを作成するよう依頼されます。レポートには、患者の保険番号、患者名、および患者が選択した開示オプションを含める必要があります。

作成されたビューは、患者の基本情報と、健康状態開示オプションを取り出します。このビューにより患者情報が保護され、患者の許可がある場合のみ、その他の目的で取り出すことができます。

Alex と IT チームは、以下のビューを実装します。

```
CREATE VIEW PATIENT_INFO_VIEW AS
SELECT P.SSN, P.NAME FROM PATIENT P, PATIENTCHOICE C
WHERE P.SSN = C.SSN AND
   C.CHOICE = 'drug-research' AND
   C.VALUE = 'opt-in';
```

Alex と彼のチームがビューを作成すると、ユーザーはこのビューを照会できます。ユーザーは、ビューの作成対象である基本表に定義されている、行および列のアクセス制御規則に従って、データを見ることができます。

Alex は、このビューに対する以下の照会により、以下の結果セットを受け取ります。

```
SELECT SSN, NAME FROM PATIENT_INFO_VIEW;

SSN         NAME
----------- ----------

  0 record(s) selected.
```

Lee 医師は、このビューに対する以下の照会により、以下の結果セットを受け取ります。

```
SELECT SSN, NAME FROM PATIENT_INFO_VIEW;

SSN         NAME
----------- ----------
123-11-9856 Sam
123-45-6789 Bob

  2 record(s) selected.
```

Bob は、このビューに対する以下の照会により、以下の結果セットを受け取ります。

```
SELECT SSN, NAME FROM PATIENT_INFO_VIEW;

SSN         NAME
----------- ----------
123-45-6789 Bob

  1 record(s) selected.
```

シナリオ: ExampleHMO における行および列アクセス制御の使用 - セキュア関数

行および列のアクセス制御定義内で呼び出すことができる関数は、あらかじめセキュアであると判断されている必要があります。セキュリティー管理者の Alex は、ExampleHMO のデータベース開発者である Paul が、彼の新しい会計アプリケーション用にセキュア関数を作成する方法について検討します。

ExampleHMO で、プライバシーおよびセキュリティーのポリシーが実施された後で、Alex は、会計部門が強力な会計アプリケーションを開発したことを知らされます。ExampleHMOAccountingUDF は、PATIENT.ACCT_BALANCE 表および行に対する列マスク ACCT_BALANCE_MASK で使用される SQL スカラー・ユーザー定義関数 (UDF) です。

列マスク内で呼び出すことができるのは、セキュアな UDF のみです。 Alex は、UDF を作成した Paul と、UDF 内部の操作がセキュアであることを確認するために相談します。

Alex は、この関数がセキュアであることに納得してから、Paul にシステム特権を付与します。これにより、Paul は UDF をセキュアに変更することができます。

```
GRANT CREATE_SECURE_OBJECT ON DATABASE TO USER PAUL;
```

セキュアな UDF を作成するため、あるいは UDF をセキュアに変更するためには、開発者に CREATE_SECURE_OBJECT 権限を付与する必要があります。

Paul は、以下の関数を作成します。

```
CREATE FUNCTION EXAMPLEHMOACCOUNTINGUDF(X DECIMAL(12,2))
       RETURNS DECIMAL(12,2)
       LANGUAGE SQL
       CONTAINS SQL
       DETERMINISTIC
       NO EXTERNAL ACTION
       RETURN X*(1.0 + RAND(X));
```

Paul は、この関数を変更して、セキュアにします。

```
ALTER FUNCTION EXAMPLEHMOACCOUNTINGUDF SECURED;
```

Alex は、ここでマスク ACC_BALANCE_MASK をドロップしてから再作成し、新しい UDF を使用できるようにします。

```
--Drop the mask to recreate

DROP MASK ACCT_BALANCE_MASK;



CREATE MASK EXAMPLEHMO.ACCT_BALANCE_MASK ONPATIENT FOR
-------------------------------------------------------------
-- Accounting information:
-- Role ACCOUNTING is allowed to invoke the secured UDF
-- ExampleHMOAccountingUDFL passing column ACCT_BALANCE as
-- the input argument
-- Other ROLEs accessing this column will strictly view a
-- zero value.
-------------------------------------------------------------
COLUMN ACCT_BALANCE RETURN
CASE WHEN VERIFY_ROLE_FOR_USER(SESSION_USER,'ACCOUNTING') = 1
THEN EXAMPLEHMOACCOUNTINGUDF(ACCT_BALANCE)
ELSE 0.00
END
ENABLE;
```

PCP ロールを持つ Lee 医師は、薬品分析のユーザー定義関数を呼び出す必要があります。DrugUDF は、患者の薬品情報を戻します。従来は、DrugUDF を呼び出す SELECT ステートメントを発行すると、Lee 医師は結果セットを迅速に受け取ることができました。PATIENT 表が、行および列のアクセス制御で保護されるようになってからは、同一の照会が結果セットを戻すまでの時間が長くかかるようになります。

Lee 医師は、ExampleHMO の IT スタッフと、セキュリティー管理者の Alex に、この性能低下について相談します。Alex は Lee 医師に、UDF がセキュアでない場合は、照会を最適化することもできず、結果セットが戻されるまでの時間が長くなることを説明します。

Alex は、Lee 医師と、UDF の所有者の Paul とともに、UDF 内部の操作がセキュアであることを確認するために相談します。Paul は、Alex に付与された CREATE_SECURE_OBJECT 特権をまだ保持しているため、Alex は Paul に、UDF をセキュアに変更するよう依頼します。

```
--Function for ExampleHMO Pharmacy department

CREATE FUNCTION DRUGUDF(PHARMACY VARCHAR(5000))
      RETURNS VARCHAR(5000)
      NO EXTERNAL ACTION
      BEGIN ATOMIC
       IF PHARMACY IS NULL THEN
          RETURN NULL;
        ELSE
          RETURN 'Normal';
        END IF;
      END;

--Secure the UDF

ALTER FUNCTION DRUGUDF SECURED;

--Grant execute permissions to Dr.Lee

GRANT EXECUTE ON FUNCTION DRUGUDF TO USER LEE;
```

Lee 医師は照会を発行することができ、照会は予期したとおりに最適化されます。

```
--Querying after the function is secured

SELECT PHARMACY FROM PATIENT
      WHERE DRUGUDF(PHARMACY) = 'Normal' AND SSN = '123-45-6789';

PHARMACY
--------
codeine

  1 record(s) selected.
```

シナリオ: ExampleHMO における行および列アクセス制御の使用 - セキュア・トリガー

行または列のアクセス制御がアクティブ化されている表に定義されるトリガーは、セキュアである必要があります。セキュリティー管理者の Alex は、ExampleHMO のデータベース開発者である Paul が、彼の新しい会計アプリケーション用にセキュア・トリガーを作成する方法について検討します。

Alex 会計部門と相談し、PATIENT 表に AFTER UPDATE トリガーが必要であることを知ります。このトリガーは、ACCT_BALANCE 列の履歴をモニターします。

Alex は、このトリガーを作成するために必要な特権を持つ Paul に、行および列のアクセス保護表に定義されるすべてのトリガーは、セキュアであるとマークされる必要があることを説明します。Paul と Alex は、新しいトリガーのアクションを検討して、セキュアであると判断します。

ExampleHMO_ACCT_BALANCE_TRIGGER は、PATIENT 表の ACCT_BALANCE 列をモニターします。この列が更新されるたびに、このトリガーが起動され、ACCT_HISTORY 表に、現在の勘定残高の詳細を挿入します。

Paul はトリガーを作成します。

```
CREATE TRIGGER HOSPITAL.NETHMO_ACCT_BALANCE_TRIGGER
  AFTER UPDATE OF ACCT_BALANCE ON PATIENT
  REFERENCING OLD AS O NEW AS N
  FOR EACH ROW MODE DB2SQL SECURED
  BEGIN  ATOMIC
```

```
  INSERT INTO ACCT_HISTORY
  (SSN, BEFORE_BALANCE, AFTER_BALANCE, WHEN, BY_WHO)
  VALUES(O.SSN, O.ACCT_BALANCE, N.ACCT_BALANCE,
  CURRENT TIMESTAMP, SESSION_USER);
END;
```

会計部門の John は、SSN が「123-45-6789」である患者 Bob の勘定残高を更新する必要があります。

John は、更新を実行する前に Bob のデータを確認します。

```
SELECT ACCT_BALANCE FROM PATIENT WHERE SSN = '123-45-6789';

ACCT_BALANCE
--------------
9.00

  1 record(s) selected.

SELECT * FROM ACCT_HISTORY WHERE SSN = '123-45-6789';

SSN         BEFORE_BALANCE AFTER_BALANCE  WHEN       BY_WHO
----------- -------------- -------------- ---------- --------------------

  0 record(s) selected.
```

John は、その後で更新を実行します。

```
UPDATE PATIENT SET ACCT_BALANCE = ACCT_BALANCE * 0.9 WHERE SSN = '123-45-6789';
```

PATIENT 表にはトリガーが定義されているので、この更新がトリガーを起動します。トリガーは SECURED と定義されているため、更新は正常に完了します。John は、更新を実行した後で、Bob のデータを見ます。

```
SELECT ACCT_BALANCE FROM PATIENT WHERE SSN = '123-45-6789';

ACCT_BALANCE
--------------
8.10

  1 record(s) selected.

SELECT * FROM ACCT_HISTORY WHERE SSN = '123-45-6789';

SSN         BEFORE_BALANCE AFTER_BALANCE  WHEN       BY_WHO
----------- -------------- -------------- ---------- --------------------
123-45-6789 9.00           8.10           2010-10-10 JOHN
  1 record(s) selected.
```

シナリオ: ExampleHMO における行および列アクセス制御の使用 - 権限の取り消し

Alex はセキュリティー管理者として、セキュア・オブジェクトを作成できるユーザーの制御を行います。開発者がセキュア・オブジェクトの作成を完了すると、Alex はデータベースに対する彼らの権限を取り消します。

データベース開発者の Paul は、開発作業を終了しました。 Alex はすぐ、次のようにして、Paul の作成権限を取り消します。

```
REVOKE CREATE_SECURE_OBJECT ON DATABASE FROM USER PAUL;
```

今後、Paul がセキュア・オブジェクトを作成する必要が生じた場合は、Paul は再び作成権限を得られるよう、Alex に相談しなければなりません。

シナリオ: ExampleBANK における行および列アクセス制御の使用

このシナリオでは、行および列のアクセス制御のユーザーとして、大きな顧客ベースと多数の支店を持つ銀行、ExampleBANK が登場します。ExampleBANK は、プライバシーとセキュリティーに関する企業の要件、およびマネジメント部門のビジネス目標をデータベース・ポリシーに反映させるために、行および列のアクセス制御を使用します。

顧客の投資、貯蓄、および個人情報を取り扱う、ExampleBANK のような組織では、組織内の情報の共有は、必要性に基づく共有に限定されます。このデータ保護により、顧客の重要な資産情報や個人情報が、適切な特権を有する行員によってのみ共有、表示、および変更されることが保証されます。

シナリオ: ExampleBANK における行および列アクセス制御の使用 - セキュリティー・ポリシー

ExampleBANK は、DB2 データベースへのデータ・アクセスが、特定のセキュリティー・ポリシーに従って可能になるセキュリティー戦略を実装します。

このセキュリティー・ポリシーは、ExampleBANK のプライバシーおよびデータ保護の規制に準拠します。最初の列はポリシー、および ExampleBANK が直面する課題の概略です。2 番目の列は、その課題に対処するための、DB2 の行および列アクセス制御 (RCAC) フィーチャーの概略です。

セキュリティーの課題	セキュリティーの課題に対処するための行および列アクセス制御フィーチャー
行へのアクセスを許可ユーザーのみに制限する。出納係は、自分の支店に属する顧客データのみ表示することを許可されています。ExampleBANK の銀行全体のシステムに含まれるすべての顧客データを表示することはできません。	特定の行を表示できるユーザーを制御するために、行の許可を実装できます。
口座番号は、顧客サービス担当者が口座更新アプリケーションを使用する場合のみ、アクセス可能です。このアプリケーションは、ストアード・プロシージャー ACCOUNTS.ACCTUPDATE を介して識別されます。	顧客サービス担当者が ACCOUNTS.ACCTUPDATE アプリケーションの外部でデータを照会したときに、重要なデータをフィルタリング、または見えなくするために、列マスクを使用できます。

シナリオ: ExampleBANK における行および列アクセス制御の使用 - データベース・ユーザーおよびロール

このシナリオでは、何人かのユーザーが ExampleBANK のデータを使用します。それらのユーザーは、それぞれ異なるユーザー権限を持っています。

ExampleBANK はセキュリティー戦略を実装して、データベースのデータにアクセスする方法をクラス分けしました。内部からのデータへのアクセス権限は、データ

にアクセスするユーザーの職務およびアクセス権の分類に基づきます。
ExampleBANK では、これらの職務を分類するために、以下のデータベース・ロールを作成しました。

TELLER
支店の出納係。

TELEMARKERTER
テレフォン・マーケティングおよびセールス担当者。

CSR
顧客サービス担当者。

以下のユーザーが ExampleBANK のデータを使用します。

ZURBIE
ExampleBANK の顧客サービス担当者。CSR ロールに属しています。

NEWTON
ExampleBANK の支店の出納係。TELLER ロールに属しています。

PLATO
ExampleBANK のテレフォン・マーケティングおよびセールス担当者。
TELEMARKETER ロールに属しています。

このシナリオに示されている SQL ステートメントおよびコマンドの例を試す場合は、これらのユーザー ID と、リストされている権限を作成してください。

以下のサンプル SQL ステートメントは、システムでユーザーが作成済みであることを想定しています。これらの SQL ステートメントは、各ロールを作成して、ExampleBANK データベース内のさまざまな表に対する SELECT 許可をユーザーに付与します。

```
--Creating roles and granting authority

CREATE ROLE TELLER;

CREATE ROLE CSR;

CREATE ROLE TELEMARKERTER;

GRANT ROLE TELLER TO USER NEWTON;
GRANT ROLE CSR TO USER ZURBIE;
GRANT ROLE TELEMARKERTER TO USER PLATO;
```

シナリオ: ExampleBANK における行および列アクセス制御の使用 - データベース表

このシナリオは、ExampleBANK データベースの 2 つの表 (CUSTOMER 表および INTERNAL_INFO 表) に注目します。

INTERNAL_INFO 表は、ExampleBANK で働く従業員に関する情報を保管します。このシナリオでは、INTERNAL 表内の次の列について検討します。

HOME_BRANCH
従業員が所属する支店の ID。

EMP_ID
従業員の ID。

CUSTOMER 表は、従業員の個人情報を保管します。

ACCOUNT
顧客の口座番号。

NAME
顧客名。

INCOME
顧客の収入。

BRANCH
顧客の支店 ID。

以下の SQL ステートメントの例は顧客、および INTERNAL_INFO 表を作成します。表に権限が付与され、データが挿入されます。

```
--Client table storing information regarding client information
CREATE TABLE RCACTSPM.CUSTOMER (
   ACCOUNT VARCHAR(19),
   NAME VARCHAR(20),
   INCOME INTEGER,
   BRANCH CHAR(1)
);

--Internal_info table which stores employee information

CREATE TABLE RCACTSPM.INTERNAL_INFO (
   HOME_BRANCH CHAR(1),
   EMP_ID VARCHAR(10));

--Grant authority

GRANT SELECT ON RCACTSPM.CUSTOMER TO USER NEWTON, USER ZURBIE, USER PLATO;

--Insert data

INSERT INTO RCACTSPM.CUSTOMER VALUES ('1111-2222-3333-4444', 'Alice', 22000, 'A');
INSERT INTO RCACTSPM.CUSTOMER VALUES ('2222-3333-4444-5555', 'Bob', 71000, 'A');
INSERT INTO RCACTSPM.CUSTOMER VALUES ('3333-4444-5555-6666', 'Carl', 123000, 'B');
INSERT INTO RCACTSPM.CUSTOMER VALUES ('4444-5555-6666-7777', 'David', 172000, 'C');

INSERT INTO RCACTSPM.INTERNAL_INFO VALUES ('A', 'NEWTON');
INSERT INTO RCACTSPM.INTERNAL_INFO VALUES ('B', 'ZURBIE');
INSERT INTO RCACTSPM.INTERNAL_INFO VALUES ('C', 'PLATO');
```

シナリオ: ExampleBANK における行および列アクセス制御の使用 - 行の許可

ExampleBANK のセキュリティー管理者は、行および列アクセス制御の一部である、行の許可を使用して、データ・アクセスの制限を開始します。行の許可は、ユーザーに返されるデータを行ごとにフィルタリングします。

出納係は、自分の支店の顧客データのみ、表示することが許可されています。テレマーケティング担当者および CSR は、システム内にある、ExampleBANK のすべての顧客を表示できますが、テレマーケティング担当者は、完全な口座番号を見ることはできません。

行の許可は、データベースにログオンしているユーザーに基づいて、行を制限またはフィルタリングします。ExampleBANK では、行の許可により、CUSTOMER 表に水平方向のデータ制約が作成されます。

セキュリティー管理者は、以下の行の許可を実装して、各ロールに属するユーザーが、表示の特権を持つ結果セットのみを表示できるように制限します。

```
CREATE PERMISSION TELLER_ROW_ACCESS ON RCACTSPM.CUSTOMER
-------------------------------------------------------
-- Teller information:
-- ROLE TELLER is allowed to access client data only
-- in their branch.
-------------------------------------------------------------
FOR ROWS WHERE VERIFY_ROLE_FOR_USER(USER, 'TELLER') = 1
AND
BRANCH = (SELECT HOME_BRANCH FROM RCACTSPM.INTERNAL_INFO WHERE EMP_ID = USER)
ENFORCED FOR ALL ACCESS
ENABLE;

CREATE PERMISSION CSR_ROW_ACCESS ON RCACTSPM.CUSTOMER
-----------------------------------------------------
-- CSR and telemarketer information:
-- ROLE TELEMARKETER and CSR are allowed to access all client
-- data rows in ExampleBANK.
-------------------------------------------------------------
FOR ROWS WHERE VERIFY_ROLE_FOR_USER (USER, 'CSR') = 1
OR
VERIFY_ROLE_FOR_USER (USER, 'TELEMARKETER') = 1
ENFORCED FOR ALL ACCESS
ENABLE;
```

セキュリティー管理者は、行の許可を作成した後も、従業員からすべてのデータが引き続き表示可能であることを確認します。行の許可は、それが定義された対象となる表に対してアクティブ化されるまでは、適用されません。セキュリティー管理者は、次のようにして、許可をアクティブ化する必要があります。

```
--Activate row access control to implement row permissions

ALTER TABLE RCACTSPM.CUSTOMER ACTIVATE ROW ACCESS CONTROL;
```

シナリオ: ExampleBANK における行および列アクセス制御の使用 - 列マスク

ExampleBANK のセキュリティー管理者は、行および列アクセス制御の一部である、列マスクを使用して、データ・アクセスの制限を強化します。列マスクは、データの表示を許可されていないユーザーまたはアプリケーションに対して、戻されるデータを列ごとに隠します。

顧客サービス担当者は、ExampleBANK システム内のすべての顧客を見ることができますが、特定のアプリケーションを使用しているとき以外は、完全な口座番号を表示することはできません。

セキュリティー管理者は、以下の列マスクを実装して、顧客サービス担当者が、表示の特権を持つ結果セットのみを表示できるように制限します。

```
CREATE MASK ACCOUNT_COL_MASK ON RCACTSPM.CUSTOMER FOR
-----------------------------------------------------------
-- Account number information:
-- Role customer service representative (CSR) is allowed to
-- access account number information only when they are using
```

```
-- the account update application. This application is
-- identified through stored procedure ACCOUNTS.ACCTUPDATE.
-- If a CSR queries this data outside of this application, the
-- account information is masked and the first 12 digits are
-- replaced with "x".
-------------------------------------------------------------
COLUMN ACCOUNT RETURN
  CASE WHEN (VERIFY_ROLE_FOR_USER (USER, 'CSR') = 1 AND
             ROUTINE_SPECIFIC_NAME = 'ACCTUPDATE' AND
             ROUTINE_SCHEMA = 'ACCOUNTS' AND
             ROUTINE_TYPE = 'P')
       THEN ACCOUNT
       ELSE 'xxxx-xxxx-xxxx-' || SUBSTR(ACCOUNT,16,4)
  END
ENABLE;
```

セキュリティー管理者は、列マスクを作成した後も、すべての従業員からデータが引き続き表示可能であることを確認します。列マスクは、それが定義された対象となる表に対してアクティブ化されるまでは、適用されません。セキュリティー管理者は、次のようにして、マスクをアクティブ化する必要があります。

```
--Activate column access control to implement column masks

ALTER TABLE RCACTSPM.CUSTOMER ACTIVATE COLUMN ACCESS CONTROL;
```

シナリオ: ExampleBANK における行および列アクセス制御の使用 - データ照会

行および列のアクセス制御を使用すると、同一のデータベース照会で、異なるロールに属するユーザーが、異なる結果セットを受け取ることができます。例えば、出納係の Newton は、自分の支店以外の顧客のデータを一切見ることができません。

Newton、Zurbie、および Plato がそれぞれデータベースに接続して、次の SQL 照会を試みます。

```
SELECT * FROM RCACTSPM.CUSTOMER;
```

照会の結果は、だれが照会を実行するかによって異なります。これらの照会には、セキュリティー管理者が作成した、行および列のアクセス制御の規則が適用されます。

これは Newton に表示される結果セットです。

```
ACCOUNT             NAME                 INCOME      BRANCH
------------------- -------------------- ----------- ------
xxxx-xxxx-xxxx-4444 Alice                      22000 A
xxxx-xxxx-xxxx-5555 Bob                        71000 A

  2 record(s) selected.
```

Newton は A 支店の出納係なので、その支店に属する ExampleBANK の顧客のみを見ることができます。

これは Zurbie に表示される結果セットです。

```
ACCOUNT             NAME                 INCOME      BRANCH
------------------- -------------------- ----------- ------
xxxx-xxxx-xxxx-4444 Alice                      22000 A
xxxx-xxxx-xxxx-5555 Bob                        71000 A
```

```
xxxx-xxxx-xxxx-6666 Carl                     123000 B
xxxx-xxxx-xxxx-7777 David                    172000 C

  4 record(s) selected.
```

Zurbie は顧客サービス担当者なので、システム内の ExampleBANK の顧客をすべて見ることができます。しかし、ACCOUNTS.ACCTUPDATE アプリケーションを使用しているとき以外は、顧客の完全な口座番号を見ることはできません。この照会は ACCOUNTS.ACCTUPDATE の外部から発行されたので、番号の一部はマスクされています。

これは Plato に表示される結果セットです。

```
ACCOUNT             NAME                 INCOME      BRANCH
------------------- -------------------- ----------- ------
xxxx-xxxx-xxxx-4444 Alice                      22000 A
xxxx-xxxx-xxxx-5555 Bob                        71000 A
xxxx-xxxx-xxxx-6666 Carl                      123000 B
xxxx-xxxx-xxxx-7777 David                     172000 C

  4 record(s) selected.
```

Plato はテレマーケティング担当者なので、システム内の ExampleBANK のすべての顧客を見ることができます。

第 5 章 ラベル・ベースのアクセス制御 (LBAC)

ラベル・ベースのアクセス制御 (LBAC) は、データにどのユーザーがアクセスできるかに対する制御を大きく向上させます。 LBAC を使用すると、個々の行および個々の列に対して、どのユーザーに書き込みアクセスがあり、どのユーザーに読み取りアクセスがあるのかを厳密に決定することができます。

LBAC の動作

LBAC 機能は非常に構成しやすく、特定の安全保護環境と一致するように調整することができます。すべての LBAC 構成はセキュリティー管理者 により実行されます。セキュリティー管理者は、SECADM 権限が付与されているユーザーです。

セキュリティー管理者は、セキュリティー・ラベル・コンポーネントを作成して LBAC システムを構成します。セキュリティー・ラベル・コンポーネント は、ユーザーがデータの一部にアクセスするかどうかを判別するのに使用する基準を表すデータベース・オブジェクトです。例えば、その基準はユーザーが特定の部門に所属しているかどうか、または特定のプロジェクトで作業しているかどうかになります。セキュリティー・ポリシー では、どのデータに誰がアクセスできるかを判断するために使用される基準を記述します。セキュリティー・ポリシーには、1 つ以上のセキュリティー・ラベル・コンポーネントが含まれています。任意の 1 つの表を保護するために 1 つのセキュリティー・ポリシーしか使用できませんが、複数のセキュリティー・ポリシーを使用して複数の表を保護することができます。

セキュリティー・ポリシーを作成した後、セキュリティー管理者は、そのポリシーの一部であるセキュリティー・ラベル というオブジェクトを作成します。セキュリティー・ラベルには、セキュリティー・ラベル・コンポーネントが含まれています。セキュリティー・ラベルに厳密に何が含まれるかはセキュリティー・ポリシーにより決定され、特定のデータ項目にアクセスできるユーザーを決定するために組織が使用する基準を示すように構成することができます。例えば、ある人の会社内での立場とその人がどのプロジェクトに参加しているかを参照して、その人が表示することができるデータを判断する場合には、各ラベルにその情報が含まれるようにセキュリティー・ラベルを構成することができます。 LBAC は柔軟であるため、非常に複雑な基準だけでなく、各ラベルが "high" または "low" のいずれかの信頼レベルを示すだけであるような非常に単純なシステムに至るまで、自由にセットアップできます。

作成が完了すると、セキュリティー・ラベルを表の個々の列と行に関連付けてそこに保持されているデータを保護することができます。セキュリティー・ラベルにより保護されるデータは、保護データ と呼ばれます。セキュリティー管理者は、ユーザーにセキュリティー・ラベルを付与することにより、保護データへのアクセスを許可します。ユーザーが保護データへのアクセスを試行すると、そのユーザーのセキュリティー・ラベルが、データを保護しているセキュリティー・ラベルと比較されます。セキュリティー・ラベルには、保護ラベルによってブロックされるものと、そうでないものがあります。

ユーザー、ロール、またはグループは、複数のセキュリティー・ポリシーに対する (複数の) セキュリティー・ラベルを同時に保持することが許可されています。ただし、どのセキュリティー・ポリシーに対しても、ユーザー、ロール、またはグループは読み取りアクセス用に最大 1 つのラベル、書き込みアクセス用に最大 1 つのラベルしか保持することができません。

セキュリティー管理者はユーザーに免除を付与することもできます。*免除* があれば、本来はセキュリティー・ラベルによってアクセスできない保護データにアクセスすることができます。セキュリティー・ラベルと免除をまとめて、*LBAC 信用証明情報* といいます。

LBAC 信用証明情報がアクセスを許可しない保護列にアクセスしようとすると、アクセスは失敗し、エラー・メッセージを受け取ります。

LBAC 信用証明情報が読み取りを許可しない保護行の読み取りを試行すると、DB2 はそれらの行が存在しないかのように動作します。それらの行は、実行するすべての SQL ステートメント (SELECT、UPDATE、DELETE を含む) において、その一部として選択することはできません。集約関数であっても、LBAC 信用証明情報が読み取りを許可しない行は無視します。例えば、COUNT(*) 関数は、読み取りアクセスを持つ行のみのカウントを戻します。

ビューと LBAC

ビューを、無保護の表にビューを定義する際と同様に、保護された表に定義することができます。そのようなビューにアクセスする際には、基礎表に対する LBAC 保護が施行されます。使用される LBAC 信用証明情報は、セッション許可 ID の LBAC 信用証明情報となります。同じビューに 2 人のユーザーがアクセスすると、それぞれの LBAC 信用証明情報により異なる行が表示される可能性があります。

参照保全制約と LBAC

以下の規則は、参照保全制約がある場合に LBAC 規則が施行される方法を説明しています。

- **規則 1**: LBAC 読み取りアクセス規則は、子表の内部で生成されたスキャンには適用されません。これは、孤立した子ができないようにするためです。
- **規則 2**: LBAC 読み取りアクセス規則は、親表の内部で生成されたスキャンには適用されません。
- **規則 3**: 子表に対して CASCADE 操作が実行される際に LBAC 書き込み規則が適用されます。例えば、ユーザーが親を削除したものの、LBAC 書き込み規則違反となるためにどの子も削除できない場合には、削除をロールバックする必要があり、エラーが出されます。

LBAC を使用したストレージ・オーバーヘッド

LBAC を使用して表を行レベルで保護する場合、追加のストレージ・コストは行セキュリティー・ラベル列のコストです。このコストは、選択したセキュリティー・ラベルのタイプによって異なります。例えば、表を保護するために 2 つのコンポーネントを持つセキュリティー・ポリシーを作成する場合、そのセキュリティー・ポリシーからのセキュリティー・ラベルは 16 バイト (コンポーネントごとに 8 バイト) になります。行セキュリティー・ラベル列は NULL 不可 VARCHAR 列として

扱われるため、この場合の合計コストは行ごとに 20 バイトになります。通常、行ごとの合計コストは (*N**8 + 4) バイトです。ここで、*N* はセキュリティー・ポリシーの表を保護するコンポーネントの数です。

LBAC を使用して表を列レベルで保護する場合、列セキュリティー・ラベルはメタデータです (つまり、列のメタデータとともに SYSCOLUMNS カタログ表に格納されます)。このメタデータは、列を保護するセキュリティー・ラベルの ID に過ぎません。この場合、ユーザー表はストレージ・オーバーヘッドの影響を受けません。

LBAC が行わない動作

- LBAC は、任意アクセス制御により禁止されているデータへのアクセスは、決して許可しません。

  例: 表からの読み取りの許可がない場合には、その表からのデータの読み取りは許可されません。普通なら LBAC によってアクセスが許可されるはずの行および列に関しても同様です。
- LBAC 信用証明情報は、保護データへのアクセスのみを制限します。無保護のデータへのアクセスには影響がありません。
- 表またはデータベースをドロップする場合、その表またはデータベースに保護データが含まれている場合であっても、LBAC 信用証明情報はチェックされません。
- データをバックアップする際には LBAC 信用証明情報はチェックされません。表のバックアップを実行できる場合、どの行がバックアップされるかについて、データの LBAC 保護により制限されることはまったくありません。また、バックアップ・メディア上のデータは LBAC により保護されません。データベース上のデータのみが保護されます。
- LBAC は、次のタイプの表を保護するために使用することはできません。
  - ステージング表
  - ステージング表が依存する表
  - 型付き表
- LBAC 保護はニックネームには適用できません。

LBAC チュートリアル

LBAC の基本的な使用方法を説明するチュートリアルを、オンラインで利用することができます。『DB2 Label-Based Access Control, a practical guide』 (http://www.ibm.com/developerworks/data) をご覧ください。

LBAC セキュリティー・ポリシー

セキュリティー管理者は、セキュリティー・ポリシーを使用して、表の個々の行および個々の列ごとに、誰に書き込みアクセスがあり、誰に読み取りアクセスがあるかを規定する基準を定義します。

セキュリティー・ポリシーには、次の情報が含まれます。

- ポリシーの一部であるセキュリティー・ラベルにおいて、どのセキュリティー・ラベル・コンポーネントが使用されるか

- それらのセキュリティー・ラベル・コンポーネントを比較する際に、どの規則が使用されるか
- ポリシーにより保護されるデータにアクセスする際に、どのオプションの動作が使用されるか
- セキュリティー・ポリシーで保護されているデータへのアクセス権の行使時に、どのような追加のセキュリティー・ラベルおよび免除を考慮の対象とするか。例えば、ロールおよびグループに付与されたセキュリティー・ラベルを考慮の対象とするかどうかのオプションは、セキュリティー・ポリシーを通して制御されます。

すべての保護されている表は、関連付けられたセキュリティー・ポリシーを 1 つだけ持たなければなりません。その表の行および列は、そのセキュリティー・ポリシーの一部であるセキュリティー・ラベルでのみ保護することができ、保護データの全アクセスは、そのポリシーの規則に従います。単一のデータベースで複数のセキュリティー・ポリシーを持つことができますが、特定の表を保護するセキュリティー・ポリシーを複数持つことはできません。

セキュリティー・ポリシーの作成

セキュリティー・ポリシーを作成する人は、セキュリティー管理者でなければなりません。セキュリティー・ポリシーは、SQL ステートメントの CREATE SECURITY POLICY で作成します。セキュリティー・ポリシーでリストされるセキュリティー・ラベル・コンポーネントは、CREATE SECURITY POLICY ステートメントを実行する前に作成する必要があります。セキュリティー・ポリシーが作成される際にコンポーネントがリストされる順序は、コンポーネント間の何らかの優先順位またはその他の関係を示すものではありませんが、SECLABEL のような組み込み関数でセキュリティー・ラベルを作成する際にその順序を知っておくことは重要です。

作成したセキュリティー・ポリシーから、セキュリティー・ラベルを作成し、データを保護できます。

セキュリティー・ポリシーの変更

セキュリティー管理者は、ALTER SECURITY POLICY ステートメントを使用して、セキュリティー・ポリシーを変更することができます。

セキュリティー・ポリシーのドロップ

セキュリティー・ポリシーをドロップする人は、セキュリティー管理者でなければなりません。セキュリティー・ポリシーは、SQL ステートメントの DROP を使用してドロップします。

セキュリティー・ポリシーは、表に関連付けられている (追加されている) 場合は、ドロップできません。

LBAC セキュリティー・ラベル・コンポーネントの概要

セキュリティー・ラベル・コンポーネント は、ラベル・ベースのアクセス制御 (LBAC) の一部であるデータベース・オブジェクトです。セキュリティー・ラベル・コンポーネントは、組織のセキュリティー構造をモデル化するために使用します。

セキュリティー・ラベル・コンポーネントは、ユーザーがデータの特定の部分にアクセスする必要があるかどうかを判別するために使用できる任意の基準を表すことができます。そのような基準の代表的な例として、以下のものがあります。

- ユーザーに対する信頼の程度
- ユーザーの所属部門
- ユーザーが特定のプロジェクトに参加しているかどうか

例: あるユーザーが所属する部門によって、どのデータにアクセスできるかに影響を与えるようにする場合、dept という名前のコンポーネントを作成し、そのコンポーネントのエレメントで会社のさまざまな部門の名前を定義することができます。その後、コンポーネント dept をセキュリティー・ポリシーに組み込みます。

セキュリティー・ラベル・コンポーネントのエレメント は、そのコンポーネントに対して許可される 1 つの特定の「設定値」です。

例: 信頼のレベルを表すセキュリティー・ラベル・コンポーネントとして、Top Secret、Secret、Classified、および Unclassified の 4 つのエレメントを設けることができます。

セキュリティー・ラベル・コンポーネントの作成

セキュリティー・ラベル・コンポーネントを作成する人は、セキュリティー管理者でなければなりません。セキュリティー・ラベル・コンポーネントは、SQL ステートメントの CREATE SECURITY LABEL COMPONENT で作成します。

セキュリティー・ラベル・コンポーネントを作成する際には、以下を指定する必要があります。

- コンポーネントの名前
- そのコンポーネントのタイプ (ARRAY、TREE、または SET)
- 許可されるエレメントの完全なリスト
- タイプ ARRAY および TREE では、各エレメントがコンポーネントの構造に収まる方法を記述する必要があります。

セキュリティー・ラベル・コンポーネントを作成した後、これらのコンポーネントに基づいてセキュリティー・ポリシーを作成することができます。このセキュリティー・ポリシーから、セキュリティー・ラベルを作成し、データを保護できます。

コンポーネントのタイプ

次の 3 つのタイプのセキュリティー・ラベル・コンポーネントがあります。

- TREE: 各エレメントはツリー構造内のノードを表します
- ARRAY: 各エレメントは線形スケール上の点を表します

- SET: 各エレメントはある集合の 1 人のメンバーを表します

エレメントが互いに関連し合うことができるさまざまな方法をモデル化するためにタイプを使用することができます。例えば、会社内の 1 つ以上の部門を記述するコンポーネントを作成する場合、おそらく TREE のコンポーネント・タイプを使用するはずです。なぜなら、ほとんどのビジネス構造はツリーの形式になっているからです。ユーザーが持つ信頼のレベルを表すコンポーネントを作成する場合、おそらくタイプ ARRAY のコンポーネントを使用するはずです。なぜなら、信頼の 2 つのレベルがある場合、一方は常に他方より高いからです。

エレメントが互いに持つことができる関係の詳細記述などの、各タイプの詳細は、タイプごとのセクションで説明されています。

セキュリティー・ラベル・コンポーネントの変更

セキュリティー管理者は、ALTER SECURITY LABEL COMPONENT ステートメントを使用して、セキュリティー・ラベル・コンポーネントを変更することができます。

セキュリティー・ラベル・コンポーネントのドロップ

セキュリティー・ラベル・コンポーネントをドロップする人は、セキュリティー管理者でなければなりません。セキュリティー・ラベル・コンポーネントは、SQL ステートメントの DROP でドロップします。

LBAC セキュリティー・ラベル・コンポーネント・タイプ: SET

SET は、ラベル・ベースのアクセス制御 (LBAC) セキュリティー・ポリシーで使用できるセキュリティー・ラベル・コンポーネントのタイプの 1 つです。

タイプ SET のコンポーネントは、エレメントの順不同のリストです。このタイプのコンポーネントのエレメントに対して行うことができる比較は、特定のエレメントがリストにあるかどうかだけです。

LBAC セキュリティー・ラベル・コンポーネント・タイプ: ARRAY

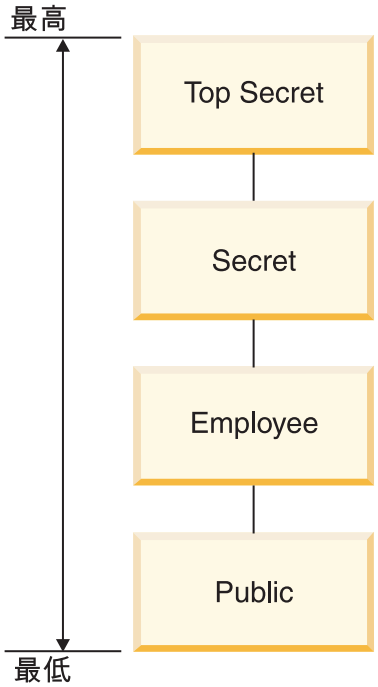
ARRAY は、セキュリティー・ラベル・コンポーネントのタイプの 1 つです。

ARRAY タイプのコンポーネントでは、コンポーネントが作成されるときにエレメントがリストされている順序によりスケールが定義されます。そこでは、リストされている最初のエレメントが最高の値となり、最後のエレメントが最低になります。

例: コンポーネント mycomp が次のように定義されているとします。

```
CREATE SECURITY LABEL COMPONENT mycomp
  ARRAY [ 'Top Secret', 'Secret', 'Employee', 'Public' ]
```

この場合、エレメントは、以下のような構造に編成されているかのようにして処理されます。



タイプ ARRAY のコンポーネントでは、エレメントは互いに以下の種類の関係を持つことができます。

より高い
ARRAY 節の中でエレメント A がエレメント B より前にリストされている場合には、エレメント A はエレメント B より高くなります。

より低い
ARRAY 節の中でエレメント A がエレメント B より後にリストされている場合には、エレメント A はエレメント B より低くなります。

LBAC セキュリティー・ラベル・コンポーネント・タイプ: TREE

TREE は、ラベル・ベースのアクセス制御 (LBAC) セキュリティー・ポリシーで使用できるセキュリティー・ラベル・コンポーネントのタイプの 1 つです。

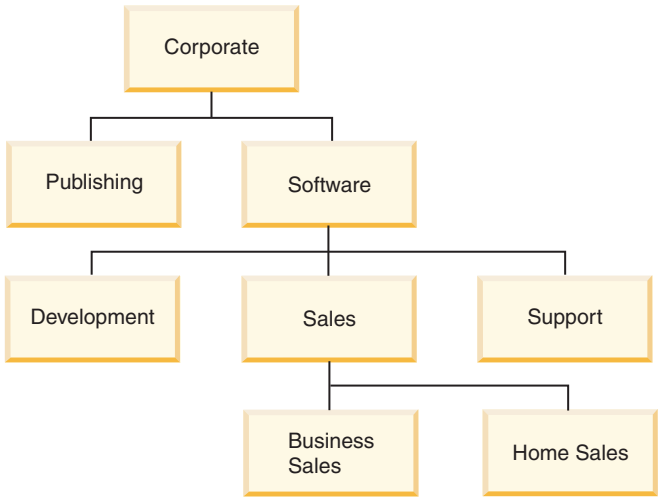
TREE タイプのコンポーネントでは、エレメントは、ツリー構造に配置されているかのようにして処理されます。タイプ TREE のコンポーネントの一部であるエレメントを指定する際、それが、他のどのエレメントの下位に置かれるかも指定する必要があります。唯一の例外は、ツリーの ROOT であると指定する必要のある最初のエレメントです。これにより、エレメントをツリー構造に編成することができます。

例: コンポーネント mycomp が次のように定義されているとします。

```
CREATE SECURITY LABEL COMPONENT mycomp
TREE (
    'Corporate'      ROOT,
    'Publishing'     UNDER 'Corporate',
    'Software'       UNDER 'Corporate',
    'Development'    UNDER 'Software',
    'Sales'          UNDER 'Software',
```

```
    'Support'        UNDER 'Software'
    'Business Sales' UNDER 'Sales'
    'Home Sales'     UNDER 'Sales'
)
```

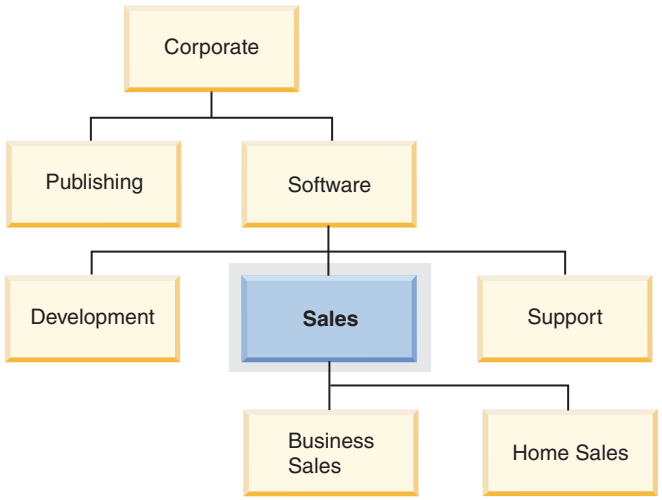
この場合、エレメントは、以下のようなツリー構造に編成されているかのようにして処理されます。



タイプ TREE のコンポーネントでは、エレメントは互いに以下のタイプの関係を持つことができます。

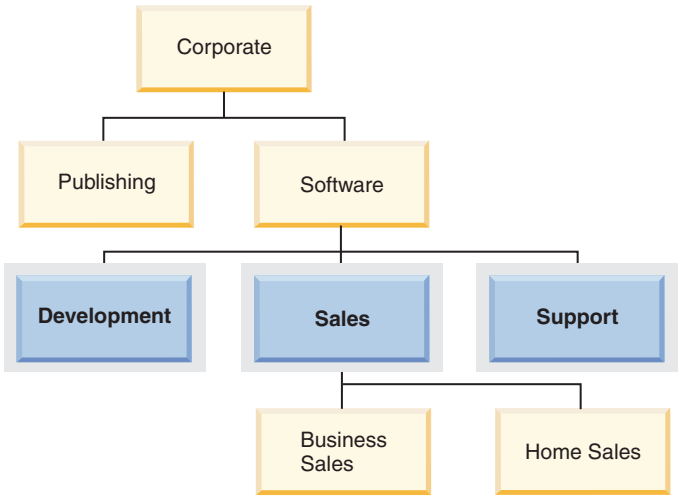
親 エレメント B がエレメント A の下にある場合には、エレメント A はエレメント B の親となります。

例: この図は、Business Sales エレメントの親を示しています。



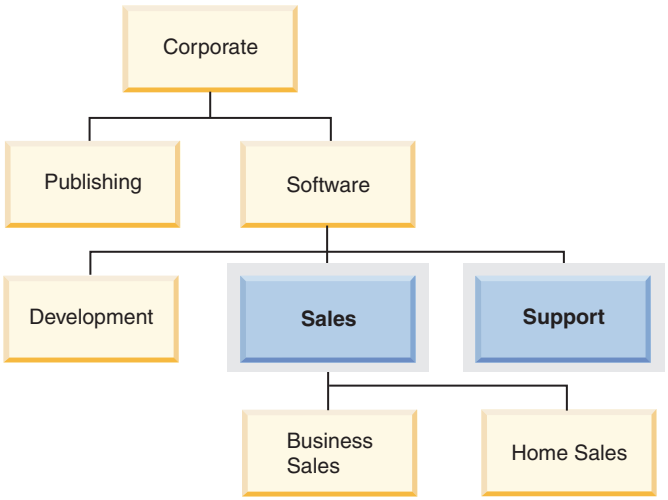
子 エレメント A がエレメント B の下にある場合には、エレメント A はエレメント B の子となります。

例: この図は、Software エレメントの子を示しています。



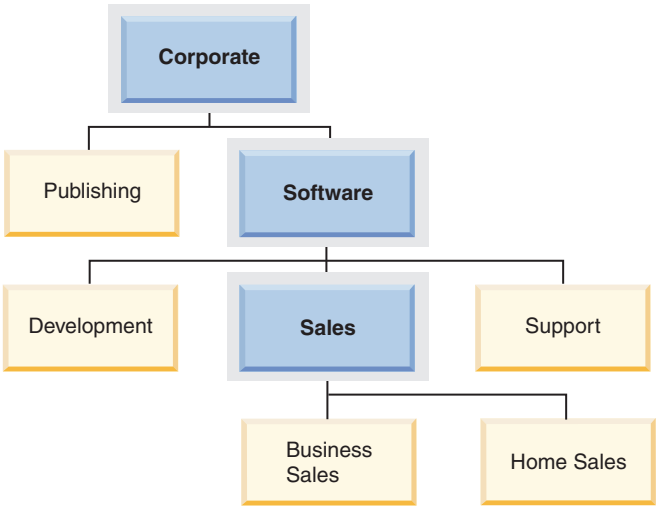
兄弟　2 つのエレメントが同じ親を持つ場合には、それらは互いに兄弟です。

例: この図は、Development エレメントの兄弟を示しています。



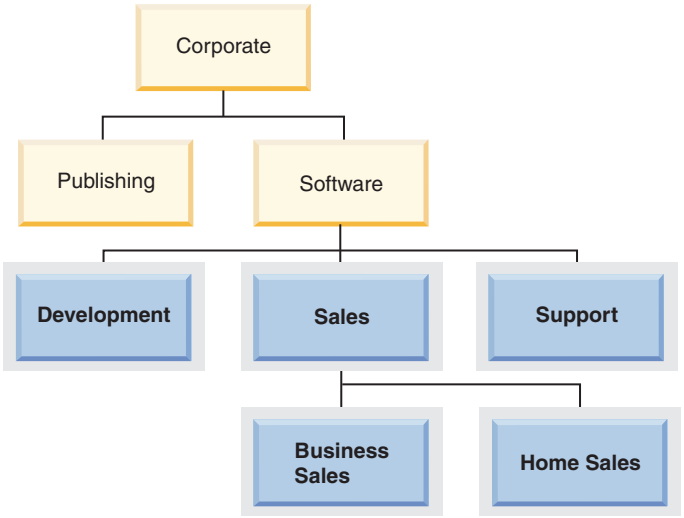
上位　エレメント A が B の親である場合、またはエレメント A が B の親の親である場合 (以下同様の条件が続く) には、エレメント A はエレメント B の上位となります。ルート・エレメントは、ツリー内の他のすべてのエレメントの上位です。

例: この図は、Home Sales エレメントの上位を示しています。



下層　エレメント A が B の子である場合、またはエレメント A が B の子の子である場合 (以下同様の条件が続く) には、エレメント A はエレメント B の下層となります。

例: この図は、Software エレメントの下層を示しています。



LBAC セキュリティー・ラベル

ラベル・ベースのアクセス制御 (LBAC) において、セキュリティー・ラベル は、セキュリティー基準の特定のセットを記述するデータベース・オブジェクトです。セキュリティー・ラベルは、データを保護するためにデータに適用されます。ユーザーが保護データにアクセスすることを許可するために、セキュリティー・ラベルは、ユーザーに付与されます。

ユーザーが保護データへのアクセスを試行すると、ユーザーのセキュリティー・ラベルが、データを保護しているセキュリティー・ラベルと比較されます。セキュリティー・ラベルには、保護しているセキュリティー・ラベルによってブロックされるものと、そうでないものがあります。あるユーザーのセキュリティー・ラベルがブロックされると、そのユーザーはデータにアクセスすることができません。

どのセキュリティー・ラベルもただ 1 つのセキュリティー・ポリシーの一部となっており、そのセキュリティー・ポリシーのコンポーネントごとに 1 つの値が含まれます。セキュリティー・ラベル・コンポーネントについて言及する場合の *値* とは、そのコンポーネントにより許可されるゼロ個以上のエレメントが含まれるリストのことを言います。 ARRAY タイプのコンポーネントの値には、ゼロ個または 1 個のエレメントを含めることができ、他のタイプの値には、ゼロ個以上のエレメントを含めることができます。エレメントが含まれない値は、*空の値* と呼ばれます。

例: TREE タイプのコンポーネントに、人的資源、販売、および配送という 3 つのエレメントが含まれる場合には、以下のものはそのコンポーネントの有効な値の一部となります。

- 人的資源 (または任意のエレメントがそれ自体で有効です)
- 人的資源、配送 (またはエレメントの他の任意の組み合わせ。ただし同じエレメントを複数回含めることはできません)
- *空の値*

特定のセキュリティー・ラベルが別のセキュリティー・ラベルをブロックするかどうかは、ラベル内の各コンポーネントの値と、表のセキュリティー・ポリシーで指定される LBAC 規則セットにより決定されます。比較をする方法の詳細は、LBAC セキュリティー・ラベルの比較方法を取り上げているトピックに説明されています。

セキュリティー・ラベルがテキスト・ストリングに変換されるときは、セキュリティー・ラベル値のフォーマットを取り上げているトピックに説明されているフォーマットを使用します。

セキュリティー・ラベルの作成

セキュリティー・ラベルを作成する人は、セキュリティー管理者でなければなりません。セキュリティー・ラベルは、SQL ステートメントの CREATE SECURITY LABEL で作成します。セキュリティー・ラベルを作成する際には、以下を指定します。

- ラベルの名前
- そのラベルが含まれるセキュリティー・ポリシー
- セキュリティー・ポリシーに含まれる 1 つ以上のコンポーネントの値

値が指定されていないコンポーネントでは、空の値を持つと想定されます。セキュリティー・ラベルには、少なくとも 1 つの空以外の値が含まれていなければなりません。

セキュリティー・ラベルの変更

セキュリティー・ラベルは、変更することができません。セキュリティー・ラベルを変更する唯一の方法は、それをドロップし、再作成することです。しかし、セキュリティー・ラベルのコンポーネント であれば、セキュリティー管理者が変更できます (ALTER SECURITY LABEL COMPONENT ステートメントを使用)。

セキュリティー・ラベルのドロップ

セキュリティー・ラベルをドロップする人は、セキュリティー管理者でなければなりません。セキュリティー・ラベルは、SQL ステートメントの DROP でドロップします。データベース内のどこかにあるデータを保護するために使用されているか、1 人以上のユーザーにより現在保持されているセキュリティー・ラベルは、ドロップできません。

セキュリティー・ラベルの付与

ユーザー、グループ、またはロールにセキュリティー・ラベルを付与する人は、セキュリティー管理者でなければなりません。セキュリティー・ラベルを付与するには、SQL ステートメントの GRANT SECURITY LABEL を使用します。セキュリティー・ラベルを付与する際、読み取りアクセス、書き込みアクセス、または読み取りと書き込み両方のアクセスに対して付与することができます。ユーザー、グループ、またはロールは、同じタイプのアクセスの場合に同じセキュリティー・ポリシーの複数のセキュリティー・ラベルを保有することはできません。

セキュリティー・ラベルの取り消し

ユーザー、グループ、またはロールのセキュリティー・ラベルを取り消す人は、セキュリティー管理者でなければなりません。セキュリティー・ラベルを取り消すには、SQL ステートメントの REVOKE SECURITY LABEL を使用します。

セキュリティー・ラベルと互換性のあるデータ・タイプ

セキュリティー・ラベルは、SYSPROC.DB2SECURITYLABEL のデータ・タイプを持っています。 SYSPROC.DB2SECURITYLABEL と VARCHAR(128) FOR BIT DATA の間で、データ変換がサポートされています。

ユーザーが保持するセキュリティー・ラベルの判別

次の照会を使用して、ユーザーが保持するセキュリティー・ラベルを判別することができます。

```
SELECT A.grantee, B.secpolicyname, c.seclabelname
FROM syscat.securitylabelaccess A, syscat.securitypolicies B, syscat.securitylabels C
WHERE A.seclabelid = C.seclabelid and B.secpolicyid = C.secpolicyid
```

セキュリティー・ラベル値の形式

セキュリティー・ラベルの値を、文字ストリングの形式で表すことがあります (例えば組み込み関数 SECLABEL を使用するとき)。

セキュリティー・ラベルの値をストリングで表すときは、以下の形式になります。

- コンポーネントの値は、CREATE SECURITY POLICY ステートメントでのセキュリティー・ポリシーのコンポーネントのリストと同じ順序で、左から右にリストする。
- エレメントは、そのエレメントの名前で表す。
- コンポーネントが異なるエレメントは、コロン (:) で分離する。
- 同一コンポーネントに複数のエレメントを指定する場合は、エレメントを括弧 (()) で囲み、コンマ (,) で分離する。

- 空の値は、一組の空括弧 (()) で表す。

例: セキュリティー・ラベルは、Level、Department、Projects という 3 つのコンポーネントをこの順序で持つセキュリティー・ポリシーの一部です。このセキュリティー・ラベルは次の値を持ちます。

表 8. セキュリティー・ラベルの値の例

コンポーネント	値
Level	Secret
Department	*空の値*
Projects	• Epsilon 37 • Megaphone • Cloverleaf

このセキュリティー・ラベル値をストリングで表すと、次のようになります。

```
'Secret:():(Epsilon 37,Megaphone,Cloverleaf)'
```

LBAC セキュリティー・ラベルが比較される方法

ラベル・ベースのアクセス制御 (LBAC) により保護されたデータへのアクセスを試行すると、そのユーザーの LBAC 信用証明情報が 1 つ以上のセキュリティー・ラベルと比較され、アクセスがブロックされるかどうかが確認されます。 LBAC 信用証明情報には、保持しているすべてのセキュリティー・ラベルに加え、保持しているすべての免除が含まれます。

行うことができる比較は、2 つのタイプしかありません。LBAC 信用証明情報と読み取りアクセス用の単一のセキュリティー・ラベルとの比較、または LBAC 信用証明情報と書き込みアクセス用の単一のセキュリティー・ラベルとの比較をすることができます。更新および削除は、読み取りの後に書き込みをするものとして処理されます。操作において、複数の比較を行う必要がある場合、それぞれの比較は別個に実行されます。

どのセキュリティー・ラベルが使用されるか

複数のセキュリティー・ラベルを保持することが可能ですが、1 つだけが保護セキュリティー・ラベルと比較されます。使用されるラベルは、以下の基準を満たすものです。

- アクセスされている表を保護しているセキュリティー・ポリシーの一部である。
- アクセスのタイプ (読み取りまたは書き込み) 用に付与されたものである。

これらの基準を満たすセキュリティー・ラベルを持っていない場合には、すべてのコンポーネントに対して空の値を持つデフォルトのセキュリティー・ラベルが想定されます。

比較が行われる方法

セキュリティー・ラベルは、コンポーネントごとに比較されます。セキュリティー・ラベルにいずれかのコンポーネントの値が含まれていない場合には、空の値が想定されます。各コンポーネントを調べる際、そのコンポーネントのユーザーの値

に含まれるエレメントが保護ラベル内の同じコンポーネントの値に含まれるエレメントによりブロックされるかどうかを決定するために LBAC 規則セットの該当する規則が使用されます。ユーザーのいずれかの値がブロックされる場合には、LBAC 信用証明情報は保護セキュリティー・ラベルによりブロックされます。

比較で使用される LBAC 規則セットは、セキュリティー・ポリシーで指定されます。どんな規則か、および各規則がいつ使用されるかについて調べるには、その規則セットの説明を参照してください。

免除が比較に与える影響

2 つの値を比較するために使用している規則に対する免除を保持している場合には、その比較は行われず、保護値はセキュリティー・ラベル内の値をブロックしないと想定されます。

例: LBAC 規則セットは DB2LBACRULES で、セキュリティー・ポリシーには 2 つのコンポーネントがあります。一方のコンポーネントはタイプ ARRAY で、他方はタイプ TREE です。ユーザーには、規則 DB2LBACREADTREE に対する免除が付与されています。この規則は、タイプ TREE のコンポーネントの値同士を比較する際に読み取りアクセス用に使用される規則です。ユーザーが保護データの読み取りを試行する場合には、その規則は使用されないため、TREE コンポーネント用にユーザーが持っている値はいずれも (それが空の値である場合であっても)、アクセスをブロックしません。ユーザーがデータを読み取ることができるかどうかは、ラベルの ARRAY コンポーネントの値に完全に依存しています。

LBAC 規則セットの概要

LBAC 規則セットは、セキュリティー・ラベルを比較する際に使用される事前定義された規則のセットです。 2 つのセキュリティー・ラベルの値が比較される際、一方の値が別の値をブロックするかどうかを判別するために、規則セット内の 1 つ以上の規則が使用されます。

それぞれの LBAC 規則セットは、固有の名前で識別されます。セキュリティー・ポリシーを作成する際、そのポリシーで使用される LBAC 規則セットを指定する必要があります。そのポリシーの一部であるセキュリティー・ラベル同士を比較する際には、その LBAC 規則セットを使用します。

規則セット内のそれぞれの規則も、固有の名前で識別されます。その規則に免除を付与する際には規則の名前を使用します。

1 つのセットに含まれる規則の数や、それぞれの規則が使用される時期は、規則セットごとに異なる場合があります。

サポートされる LBAC 規則セットは、現在 1 つだけです。その規則セットの名前は DB2LBACRULES です。

LBAC 規則セット: DB2LBACRULES

DB2LBACRULES LBAC 規則セットは、セキュリティー・ラベル・コンポーネントの値を比較するための従来型の規則のセットを提供します。これは、ライトアップおよびライトダウン両方から保護します。

ライトアップおよびライトダウンの説明

ライトアップおよびライトダウンは、タイプ ARRAY のコンポーネントの書き込みアクセスのみに適用されます。ライトアップは、書き込み対象のデータを保護する値が自分の値より高い場合に発生します。ライトダウンは、そのデータを保護する値が自分の値より低い場合に発生します。デフォルトではライトアップもライトダウンも許可されません。つまり、自分が持っている値と同じ値で保護されているデータしか書き込むことはできません。

同じコンポーネントに対する 2 つの値を比較する際、どちらの規則が使用されるかは、コンポーネントのタイプ (ARRAY、SET、または TREE) およびどのタイプのアクセス (読み取りまたは書き込み) が試行されているかにより異なります。次の表は、規則をリストし、それぞれの規則がいつ使用されるかを示し、アクセスがブロックされるかどうかを規則が判別する方法を説明しています。

表 9. DB2LBACRULES 規則のサマリー

規則名	次のタイプのコンポーネントの値を比較する場合に使用	次のタイプのアクセスに使用	次の条件が満たされた場合にアクセスをブロック
DB2LBACREADARRAY	ARRAY	読み取り	ユーザーの値が、保護値より低い。
DB2LBACREADSET	SET	読み取り	ユーザーが保持しない保護値が 1 つ以上ある。
DB2LBACREADTREE	TREE	読み取り	ユーザーの値がいずれも、保護値のいずれとも等しくないか、その上位でない。
DB2LBACWRITEARRAY	ARRAY	書き込み	ユーザーの値が保護値より高いか、保護値より低い。[1]
DB2LBACWRITESET	SET	書き込み	ユーザーが保持しない保護値が 1 つ以上ある。
DB2LBACWRITETREE	TREE	書き込み	ユーザーの値がいずれも、保護値のいずれとも等しくないか、その上位でない。

注:

1. DB2LBACWRITEARRAY 規則は、2 つの異なる規則が結合したものであると考えることができます。一方は自分のレベルより高いデータに書き込むこと (ライトアップ) を防ぎ、他方は自分のレベルより低いデータに書き込む (ライトダウン) ことを防ぎます。この規則に免除を付与すると、ユーザーをこれらの規則の一方または両方から免除することができます。

規則が空の値を処理する方法

すべての規則は空の値を同じ方法で処理します。空の値は他の値をブロックせず、空以外のすべての値によりブロックされます。

DB2LBACREADSET および DB2LBACWRITESET の例

以下の例は、保護データの読み取りまたは書き込みを試行しているユーザーに対して有効です。ここでは、値は、one two three four というエレメントを持つタイプ SET のコンポーネント用であると想定します。

表 10. DB2LBACREADSET および DB2LBACWRITESET 規則を適用する例

ユーザーの値	保護値	アクセスのブロック
'one'	'one'	ブロックされません。値は同じです。
'(one,two,three)'	'one'	ブロックされません。ユーザーの値にはエレメント 'one' が含まれています。
'(one,two)'	'(one,two,four)'	ブロックされます。エレメント 'four' は保護値には含まれていますが、ユーザーの値には含まれていません。
'()'	'one'	ブロックされます。空の値は空以外のすべての値によりブロックされます。
'one'	'()'	ブロックされません。空の値ではどの値もブロックされません。
'()'	'()'	ブロックされません。空の値ではどの値もブロックされません。

DB2LBACREADTREE および DB2LBACWRITETREE

以下の例は、読み取りアクセスと書き込みアクセスの両方に有効です。ここでは、TREE タイプのコンポーネントの値が以下の方法で定義されたと想定します。

```
CREATE SECURITY LABEL COMPONENT mycomp
TREE (
    'Corporate'      ROOT,
    'Publishing'     UNDER 'Corporate',
    'Software'       UNDER 'Corporate',
    'Development'    UNDER 'Software',
    'Sales'          UNDER 'Software',
    'Support'        UNDER 'Software'
    'Business Sales' UNDER 'Sales'
    'Home Sales'     UNDER 'Sales'
)
```

これは、エレメントが次のように配置されていることを意味します。

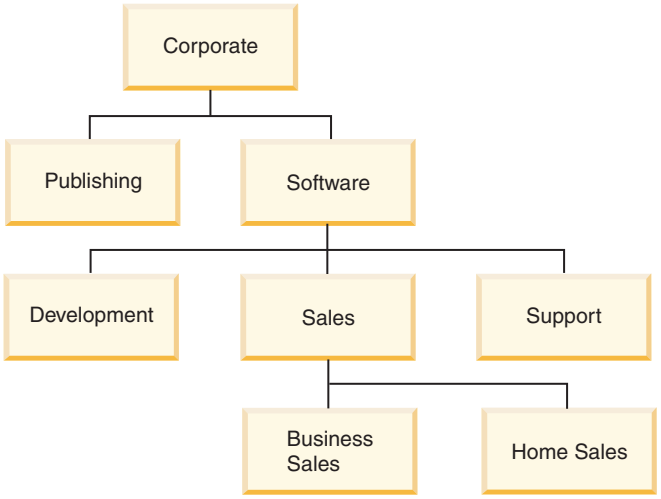


表 11. DB2LBACREADTREE および DB2LBACWRITETREE 規則を適用する例

ユーザーの値	保護値	アクセスのブロック
'(Support,Sales)'	'Development'	ブロックされます。エレメント 'Development' はユーザーの値の 1 つではなく、'Support' および 'Sales' のいずれも 'Development' の上位ではありません。
'(Development,Software)'	'(Business Sales,Publishing)'	ブロックされません。エレメント 'Software' は 'Business Sales' の上位です。
'(Publishing,Sales)'	'(Publishing,Support)'	ブロックされません。エレメント 'Publishing' は両方の値のセットに含まれています。
'Corporate'	'Development'	ブロックされません。ルート値は他のすべての値の上位です。
'()'	'Sales'	ブロックされます。空の値は空以外のすべての値によりブロックされます。
'Home Sales'	'()'	ブロックされません。空の値ではどの値もブロックされません。
'()'	'()'	ブロックされません。空の値ではどの値もブロックされません。

DB2LBACREADARRAY の例

以下の例は読み取りアクセス専用です。ここでは、値は、以下の配置における以下のエレメントが含まれるタイプ ARRAY のコンポーネント用であると想定します。

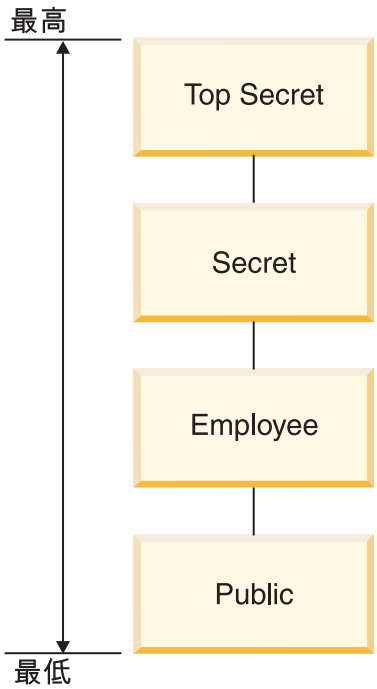


表 12. DB2LBACREADARRAY 規則を適用する例

ユーザーの値	保護値	読み取りアクセスのブロック
'Secret'	'Employee'	ブロックされません。エレメント 'Secret' はエレメント 'Employee' より高位にあります。
'Secret'	'Secret'	ブロックされません。値は同じです。
'Secret'	'Top Secret'	ブロックされます。エレメント 'Top Secret' はエレメント 'Secret' より高位にあります。
'()'	'Public'	ブロックされます。空の値は空以外のすべての値によりブロックされます。
'Public'	'()'	ブロックされません。空の値ではどの値もブロックされません。
'()'	'()'	ブロックされません。空の値ではどの値もブロックされません。

DB2LBACWRITEARRAY の例

以下の例は書き込みアクセス専用です。ここでは、値は、以下の配置における以下のエレメントが含まれるタイプ ARRAY のコンポーネント用であると想定します。

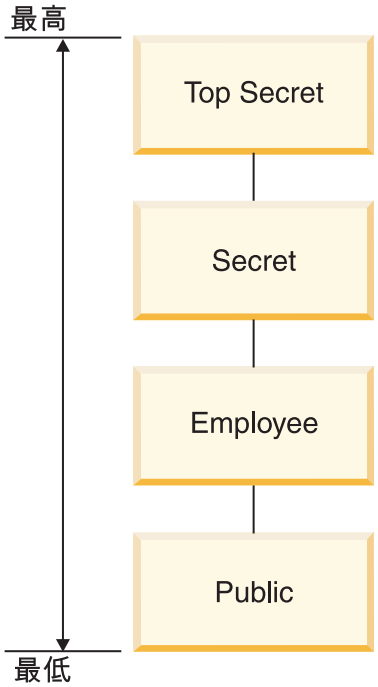


表 13. DB2LBACWRITEARRAY 規則を適用する例

ユーザーの値	保護値	書き込みアクセスのブロック
'Secret'	'Employee'	ブロックされます。エレメント 'Employee' はエレメント 'Secret' より下位にあります。
'Secret'	'Secret'	ブロックされません。値は同じです。
'Secret'	'Top Secret'	ブロックされます。エレメント 'Top Secret' はエレメント 'Secret' より高位にあります。
'()'	'Public'	ブロックされます。空の値は空以外のすべての値によりブロックされます。
'Public'	'()'	ブロックされません。空の値ではどの値もブロックされません。
'()'	'()'	ブロックされません。空の値ではどの値もブロックされません。

LBAC 規則の免除

特定のセキュリティー・ポリシーの特定の規則に関する LBAC 規則免除を保有していれば、そのセキュリティー・ポリシーによって保護されているデータにアクセスしようとした場合、その規則は施行されません。

付与されたセキュリティー・ポリシー以外のセキュリティー・ポリシーのセキュリティー・ラベルを比較する場合には、免除による影響はありません。

例:

T1 および T2 という 2 つの表があります。 T1 はセキュリティー・ポリシー P1 によって保護されており、T2 はセキュリティー・ポリシー P2 によって保護されています。両方のセキュリティー・ポリシーは 1 つのコンポーネントを持ってい ま

す。それぞれのコンポーネントのタイプは ARRAY です。T1 および T2 にはそれぞれ、1 行のデータのみが含まれます。セキュリティー・ポリシー P1 のもとで読み取りアクセスに対して保持しているセキュリティー・ラベルにより、T1 の行にアクセスすることはできません。セキュリティー・ポリシー P2 のもとで読み取りアクセスに対して保持しているセキュリティー・ラベルにより、T2 の行に読み取りアクセスすることはできません。

ここで、P1 の下の DB2LBACREADARRAY に対する免除が付与されます。T1 から行を読み取ることはできますが、T2 から読み取ることはできません。これは、T2 が別のセキュリティー・ポリシーによって保護されており、そのポリシー内の DB2LBACREADARRAY 規則に対する免除が保持されていないためです。

複数の免除を保持することができます。セキュリティー・ポリシーによって使用されるすべての規則に対する免除を保持している場合、そのセキュリティー・ポリシーによって保護されているすべてのデータに対する完全なアクセス権を持ちます。

LBAC 規則の免除の付与

LBAC 規則の免除を付与する人は、セキュリティー管理者でなければなりません。LBAC 規則の免除を付与するには、SQL ステートメント GRANT EXEMPTION ON RULE を使用します。

LBAC 規則を付与する場合、以下の情報を提供します。

- 免除の対象となる 1 つ以上の規則
- 免除の対象となるセキュリティー・ポリシー
- 免除を付与する対象のユーザー、グループ、またはロール

重要: LBAC 規則の免除により、非常に強力なアクセス権が提供されます。アクセス権を付与する場合には、注意深く考慮してください。

LBAC 規則の免除の取り消し

LBAC 規則の免除を取り消す人は、セキュリティー管理者でなければなりません。LBAC 規則の免除を取り消すには、SQL ステートメント REVOKE EXEMPTION ON RULE を使用します。

ユーザーが保持する規則の免除の判別

次の照会を使用して、ユーザーが保持する規則の免除を判別することができます。

```
SELECT A.grantee, A.accessrulename, B.secpolicyname
FROM syscat.securitypolicyexemptions A, syscat.securitypolicies B
WHERE A.secpolicyid = B.secpolicyid
```

LBAC セキュリティー・ラベルを管理するための組み込み関数

ラベル・ベースのアクセス制御 (LBAC) セキュリティー・ラベルを管理するために、組み込み関数 SECLABEL、SECLABEL_BY_NAME、および SECLABEL_TO_CHAR が提供されています。

それぞれの組み込み関数の概要はここで説明されており、詳細については「*SQL リファレンス*」で説明されています。

SECLABEL

この組み込み関数は、セキュリティー・ポリシーと、ラベル内の各コンポーネントの値を指定することによりセキュリティー・ラベルを作成するために使用されます。戻り値は DB2SECURITYLABEL のデータ・タイプを持っているセキュリティー・ラベルであり、そのセキュリティー・ラベルは指示されたセキュリティー・ポリシーの一部で、コンポーネント用の指示された値を持っています。指示された値を持つセキュリティー・ラベルがすでに存在している必要はありません。

例: 表 T1 には 2 つの列があり、最初の列のデータ・タイプは DB2SECURITYLABEL で、2 番目の列のデータ・タイプは INTEGER です。T1 はセキュリティー・ポリシー P1 により保護され、P1 には level、departments、および groups という 3 つのセキュリティー・ラベル・コンポーネントがあります。UNCLASSIFIED がコンポーネント level のエレメントであり、ALPHA および SIGMA が両方ともコンポーネント departments のエレメントであり、G2 がコンポーネント groups のエレメントである場合には、セキュリティー・ラベルは次のように挿入できます。

```
INSERT INTO T1 VALUES
   ( SECLABEL( 'P1', 'UNCLASSIFIED:(ALPHA,SIGMA):G2' ), 22 )
```

SECLABEL_BY_NAME

この組み込み関数は、セキュリティー・ポリシーの名前と、そのセキュリティー・ポリシーの一部であるセキュリティー・ラベルの名前を受け入れます。この組み込み関数はその後、指示されたセキュリティー・ラベルを DB2SECURITYLABEL として戻します。DB2SECURITYLABEL のデータ・タイプを持つ列に既存のセキュリティー・ラベルを挿入する際にこの関数を使用する必要があります。

例: 表 T1 には 2 つの列があり、最初の列のデータ・タイプは DB2SECURITYLABEL で、2 番目の列のデータ・タイプは INTEGER です。L1 という名前のセキュリティー・ラベルは、セキュリティー・ポリシー P1 の一部です。次の SQL は、セキュリティー・ラベルを挿入します。

```
INSERT INTO T1 VALUES ( SECLABEL_BY_NAME( 'P1', 'L1' ), 22 )
```

次の SQL ステートメントは作動しません。

```
INSERT INTO T1 VALUES ( P1.L1, 22 )     // Syntax Error!
```

SECLABEL_TO_CHAR

この組み込み関数は、セキュリティー・ラベルを構成する値のストリング表記を戻します。

例: 表 T1 の列 C1 のデータ・タイプは DB2SECURITYLABEL です。T1 はセキュリティー・ポリシー P1 により保護され、P1 には level、departments、および groups という 3 つのセキュリティー・ラベル・コンポーネントがあります。T1 には 1 つの行があり、列 C1 には各コンポーネントに以下のエレメントを持つ値があります。

コンポーネント	エレメント
level	SECRET

コンポーネント	エレメント
departments	DELTA および SIGMA
groups	G3

行の読み取りを許可する LBAC 信用証明情報を持つユーザーは、次の SQL ステートメントを実行します。

```
SELECT SECLABEL_TO_CHAR( 'P1', C1 ) AS C1 FROM T1
```

出力は次のようになります。

```
C1

'SECRET:(DELTA,SIGMA):G3'
```

LBAC を使用したデータの保護

ラベル・ベースのアクセス制御 (LBAC) を使用すると、データの行またはデータの列の一方または両方を保護することができます。表内のデータは、表を保護するセキュリティー・ポリシーの一部であるセキュリティー・ラベルによってのみ保護できます。データ保護 (セキュリティー・ポリシーの追加を含む) は、表の作成時に、またはそれ以降に表を変更することによって行うことができます。

セキュリティー・ポリシーを表に追加し、同じ CREATE TABLE または ALTER TABLE ステートメントの一部としてその表のデータを保護することができます。

一般規則として、現行の LBAC 信用証明情報がデータへの書き込みを許可しないようにそのデータを保護することは、許可されません。

表へのセキュリティー・ポリシーの追加

CREATE TABLE ステートメントの SECURITY POLICY 節を使用して表を作成するときに、セキュリティー・ポリシーを表に追加することができます。 ALTER TABLE ステートメントの ADD SECURITY POLICY 節を使用して、セキュリティー・ポリシーを既存の表に追加することができます。表にセキュリティー・ポリシーを追加するために SECADM 権限または LBAC 信用証明情報は必要ではありません。

セキュリティー・ポリシーは、LBAC により保護できない表のタイプには追加できません。LBAC で保護できない表タイプのリストについては、LBAC の概説を参照してください。

どの表にも複数のセキュリティー・ポリシーを追加することはできません。

行の保護

表を作成する際に DB2SECURITYLABEL のデータ・タイプの列を組み込むことにより、新しい表内の保護された行を許可することができます。さらに CREATE TABLE ステートメントでは、セキュリティー・ポリシーを表に追加する必要があります。そのような表を作成するために SECADM 権限または LBAC 信用証明情報は必要ではありません。

DB2SECURITYLABEL のデータ・タイプを持つ列を追加することにより、既存の表内の保護された行を許可することができます。そのような列を追加するためには、表がすでにセキュリティー・ポリシーで保護されているか、列を追加する ALTER TABLE ステートメントもセキュリティー・ポリシーを表に追加するかのいずれかでなければなりません。列が追加されると、既存のすべての行を保護するために、書き込みアクセス用に保持しているセキュリティー・ラベルが使用されます。表を保護するセキュリティー・ポリシーの一部である書き込みアクセス用のセキュリティー・ラベルを保持していない場合には、DB2SECURITYLABEL のデータ・タイプを持つ列を追加することはできません。

表にタイプ DB2SECURITYLABEL の列が組み込まれた後、その列にセキュリティー・ラベルを保管することにより、それぞれの新しいデータの行を保護します。これがどのようになるかの詳細は、LBAC 保護データの挿入および更新に関するトピックに説明されています。タイプ DB2SECURITYLABEL の列を持つ表に行を挿入するには、LBAC 信用証明情報が必要です。

DB2SECURITYLABEL のデータ・タイプを持つ列は、ドロップできず、他のデータ・タイプに変更できません。

列の保護

CREATE TABLE ステートメントの SECURED WITH 列オプションを使用して表を作成するときに、列を保護することができます。 ALTER TABLE ステートメントの SECURED WITH オプションを使用して、既存の列に保護を追加することができます。

特定のセキュリティー・ラベルで列を保護するには、そのセキュリティー・ラベルにより保護されるデータへの書き込みを許可する LBAC 信用証明情報がなければなりません。 SECADM 権限を持っている必要はありません。

列は、表を保護するセキュリティー・ポリシーの一部であるセキュリティー・ラベルによってのみ保護できます。セキュリティー・ポリシーを持たない表の列を保護することはできません。セキュリティー・ポリシーで表を保護して、同じステートメントで 1 つ以上の列を保護することが許可されています。

表内の任意の数の列を保護することができますが、1 つの列を複数のセキュリティー・ラベルで保護することはできません。

LBAC 保護データの読み取り

ラベル・ベースのアクセス制御 (LBAC) により保護されたデータの読み取りを試行すると、読み取り用の LBAC 信用証明情報が、データを保護しているセキュリティー・ラベルと比較されます。保護ラベルが信用証明情報をブロックしない場合、データを読み取ることが許可されます。

保護された列の場合、保護セキュリティー・ラベルは、表のスキーマで定義されます。その列の保護セキュリティー・ラベルは、表のすべての行において同じです。保護された行の場合、保護セキュリティー・ラベルは、タイプ DB2SECURITYLABEL の列の行に保管されます。それは、表内の行ごとに異なる場合があります。

LBAC 信用証明情報がセキュリティー・ラベルと比較される方法の詳細は、LBAC セキュリティー・ラベルの比較方法に関するトピックで説明されています。

保護された列の読み取り

保護された列からの読み取りを試行する際、LBAC 信用証明情報はその列を保護するセキュリティー・ラベルと比較されます。この比較を基にして、アクセスはブロックまたは許可されます。アクセスがブロックされる場合にはエラーが戻され、ステートメントは失敗します。そうでない場合は、ステートメントは通常通り進行します。

LBAC 信用証明情報が読み取りを許可しない列の読み取りを試行すると、ステートメント全体が失敗します。

例:

表 T1 には保護された列が 2 つあります。列 C1 はセキュリティー・ラベル L1 により保護されています。列 C2 はセキュリティー・ラベル L2 により保護されています。

ユーザー Jyoti は、セキュリティー・ラベル L1 へのアクセスを許可する読み取り用 LBAC 信用証明情報を持っているものの、L2 に対するものは持っていないと想定します。Jyoti が次の SQL ステートメントを発行すると、ステートメントは失敗します。

```
SELECT * FROM T1
```

SELECT 節に、ワイルドカード (*) の一部として列 C2 が含まれているために、ステートメントは失敗します。

Jyoti が次の SQL ステートメントを発行すると、それは成功します。

```
SELECT C1 FROM T1
```

SELECT 節の中で保護されている列は C1 のみで、Jyoti の LBAC 信用証明情報は Jyoti がその列を読み取ることを許可しています。

保護された行の読み取り

ある行を読み取ることを許可する LBAC 信用証明情報をあるユーザーが持っていない場合、そのユーザーにとっては、その行は存在していないかのようになります。

保護された行を読み取る際、LBAC 信用証明情報が読み取りアクセスを許可する行のみが戻されます。タイプ DB2SECURITYLABEL の列が SELECT 節の一部でない場合でも、そのように処理されます。

ユーザーの LBAC 信用証明情報に応じて、保護された行を持つ表では、異なるユーザーには異なる行が表示される可能性があります。例えば、T1 に保護された行があり、2 人のユーザーが異なる LBAC 信用証明情報を持っている場合、ステートメント `SELECT COUNT(*) FROM T1` を実行するそれらユーザーは、異なる結果を受け取る可能性があります。

LBAC 信用証明情報は、SELECT ステートメントだけでなく、UPDATE、DELETE のような他の SQL ステートメントにも影響します。ある行を読み取ることを許可する LBAC 信用証明情報を持っていない場合、その行に影響を与えることはできません。

例:

表 T1 には以下のような行と列があります。列 ROWSECURITYLABEL のデータ・タイプは DB2SECURITYLABEL です。

表 14. 表 T1 の値の例

LASTNAME	DEPTNO	ROWSECURITYLABEL
Rjaibi	55	L2
Miller	77	L1
Fielding	11	L3
Bird	55	L2

ユーザー Dan は、セキュリティー・ラベル L1 により保護されるデータの読み取りを許可する LBAC 信用証明情報を持っているものの、L2 または L3 により保護されるデータに対するものは持っていないと想定します。

Dan は次の SQL ステートメントを発行します。

```
SELECT * FROM T1
```

SELECT ステートメントは Miller の行のみを戻します。エラー・メッセージや警告は戻されません。

表 T1 の Dan のビューは次のようになります。

表 15. 表 T1 のビューの値の例

LASTNAME	DEPTNO	ROWSECURITYLABEL
Miller	77	L1

Rjaibi、Fielding、および Bird の行は戻されません。なぜなら読み取りアクセスはそれらのセキュリティー・ラベルによりブロックされるからです。 Dan はこれらの行を削除または更新することはできません。これらの行は、集約関数に組み込むこともできません。Dan にとっては、これらの行は存在していないかのようになります。

Dan は次の SQL ステートメントを発行します。

```
SELECT COUNT(*) FROM T1
```

Miller の行しかユーザー Dan は読み取ることができないため、ステートメントは 1 の値を戻します。

保護された列を含む保護された行の読み取り

列のアクセスは、行のアクセスの前にチェックされます。選択している列のいずれかを保護しているセキュリティー・ラベルによって読み取りアクセス用の LBAC 信

用証明情報がブロックされる場合には、ステートメント全体が失敗します。ブロックされない場合、ステートメントは継続し、LBAC 信用証明情報が読み取りアクセスを許可する対象であるセキュリティー・ラベルにより保護される行のみが戻されます。

例

表 T1 の列 LASTNAME はセキュリティー・ラベル L1 で保護されています。列 DEPTNO はセキュリティー・ラベル L2 で保護されています。列 ROWSECURITYLABEL のデータ・タイプは DB2SECURITYLABEL です。T1 (データを含む) は次のようになります。

表 16. 表 T1 の値の例

LASTNAME *L1 により保護される*	**DEPTNO** *L2 により保護される*	**ROWSECURITYLABEL**
Rjaibi	55	L2
Miller	77	L1
Fielding	11	L3

ユーザー Sakari は、セキュリティー・ラベル L1 により保護されるデータの読み取りを許可する LBAC 信用証明情報を持っているものの、L2 または L3 に対するものは持っていないと想定します。

Sakari は次の SQL ステートメントを発行します。

```
SELECT * FROM T1
```

SELECT 節は列 DEPTNO が含まれるワイルドカード (*) を使用しているため、ステートメントは失敗します。列 DEPTNO は、Sakari の LBAC 信用証明情報が Sakari に読み取りを許可しないセキュリティー・ラベル L2 により保護されています。

Sakari は今度は次の SQL ステートメントを発行します。

```
SELECT LASTNAME, ROWSECURITYLABEL FROM T1
```

SELECT 節には Sakari が読み取りできない列が含まれていないため、ステートメントは継続します。ただし、他の各行はセキュリティー・ラベル L2 または L3 により保護されるため、1 つの行のみが戻されます。

表 17. 表 T1 に対する照会の出力の例

LASTNAME	**ROWSECURITYLABEL**
Miller	L1

LBAC 保護データの挿入

保護された列にデータを挿入しようとする際、または保護された行を持つ表に新しい行を挿入しようとする際、その INSERT ステートメントの処理方法は LBAC 信用証明情報によって決まります。

保護された列への挿入

保護された列にデータの挿入を試行する際、書き込み用の LBAC 信用証明情報はその列を保護するセキュリティー・ラベルと比較されます。この比較を基にして、アクセスはブロックまたは許可されます。

2 つのセキュリティー・ラベルが比較される方法の詳細は、LBAC セキュリティー・ラベルの比較方法に関するトピックで説明されています。

アクセスが許可される場合、ステートメントは通常通り進行します。アクセスがブロックされる場合には挿入は失敗し、エラーが戻されます。

行を挿入しているものの、保護される列の値を指定しない場合には、デフォルト値がある場合にはそれが挿入されます。これは、LBAC 信用証明情報がその列への書き込みアクセスを許可しない場合であっても、そのように処理されます。デフォルトは次の場合に使用可能です。

- 列が WITH DEFAULT オプションで宣言された
- 列が生成された列である
- 列には、BEFORE トリガーにより指定されるデフォルト値がある
- 列のデータ・タイプは DB2SECURITYLABEL で、その場合、書き込みアクセス用に保持するセキュリティー・ラベルはデフォルト値である

保護された行への挿入

保護された行を持つ表に新しい行を挿入する際、タイプ DB2SECURITYLABEL の列には値を指定する必要はありません。その列に値を指定しない場合、列には、書き込みアクセス用にユーザーに付与されたセキュリティー・ラベルが自動的に取り込まれます。書き込みアクセス用のセキュリティー・ラベルがユーザーに付与されていない場合、エラーが戻され、挿入は失敗します。

SECLABEL のような組み込み関数を使用することにより、タイプ DB2SECURITYLABEL の列に挿入されるセキュリティー・ラベルを明示的に指定することができます。ただし、指定したセキュリティー・ラベルは、挿入を試行しているセキュリティー・ラベルで保護されるデータへの書き込みを LBAC 信用証明情報が許可している場合にのみ、使用されます。

書き込みできないセキュリティー・ラベルを指定した場合には、行われる処理は、表を保護しているセキュリティー・ポリシーにより異なります。セキュリティー・ポリシーに RESTRICT NOT AUTHORIZED WRITE SECURITY LABEL オプションがある場合は、挿入が失敗し、エラーが戻されます。セキュリティー・ポリシーに RESTRICT NOT AUTHORIZED WRITE SECURITY LABEL オプションがない場合や、代わりに OVERRIDE NOT AUTHORIZED WRITE SECURITY LABEL オプションがある場合は、指定したセキュリティー・ラベルが無視され、書き込みアクセス用に保持しているセキュリティー・ラベルがあれば、そのセキュリティー・ラベルが代わりに使用されます。書き込みアクセス用のセキュリティー・ラベルを保持していない場合は、エラーが戻されます。

例

表 T1 は、RESTRICT NOT AUTHORIZED WRITE SECURITY LABEL オプションを付けずに作成された P1 という名前のセキュリティー・ポリシーにより保護されています。表 T1 には 2 つの列がありますが、行はありません。列は LASTNAME と LABEL です。列 LABEL のデータ・タイプは DB2SECURITYLABEL です。

ユーザー Joe は、書き込みアクセス用のセキュリティー・ラベル L2 を保持しています。セキュリティー・ラベル L2 は、セキュリティー・ラベル L2 により保護されるデータへの書き込みを Joe に許可するものの、セキュリティー・ラベル L1 または L3 により保護されるデータに対しては許可しないと想定します。

Joe は次の SQL ステートメントを発行します。

```
INSERT INTO T1 (LASTNAME, DEPTNO) VALUES ('Rjaibi', 11)
```

INSERT ステートメントにはセキュリティー・ラベルが含まれていなかったため、Joe の書き込みアクセス用セキュリティー・ラベルは LABEL 行に挿入されます。

表 T1 は次のようになります。

表 18. 最初の INSERT ステートメント後の例示表 T1 の値

LASTNAME	LABEL
Rjaibi	L2

Joe は以下の SQL ステートメントを発行します。その中で、列 LABEL に挿入するセキュリティー・ラベルを明示的に指定します。

```
INSERT INTO T1 VALUES ('Miller', SECLABEL_BY_NAME('P1', 'L1') )
```

ステートメント内の SECLABEL_BY_NAME 関数は、セキュリティー・ポリシー P1 の一部で L1 という名前のセキュリティー・ラベルを戻します。Joe は L1 で保護されるデータへの書き込みが許可されていないため、L1 を列 LABEL に挿入することは許可されません。

T1 を保護するセキュリティー・ポリシーは RESTRICT NOT AUTHORIZED WRITE SECURITY LABEL オプションを付けずに作成されたため、書き込み用に Joe が保持するセキュリティー・ラベルが代わりに挿入されます。エラーまたはメッセージは戻されません。

表は次のようになります。

表 19. 2 回目の INSERT ステートメント後の例示表 T1 の値

LASTNAME	LABEL
Rjaibi	L2
Miller	L2

もし表を保護しているセキュリティー・ポリシーが RESTRICT NOT AUTHORIZED WRITE SECURITY LABEL オプションを付けて作成されていたとすると、挿入は失敗し、エラーが戻されたはずです。

次に Joe は LBAC 規則の 1 つに対する免除が付与されます。Joe の新しい LBAC 信用証明情報は、セキュリティー・ラベル L1 および L2 で保護されるデータへの書き込みを許可すると想定します。書き込みアクセス用に Joe に付与されたセキュリティー・ラベルは変更されず、L2 のままです。

Joe は次の SQL ステートメントを発行します。

```
INSERT INTO T1 VALUES ('Bird', SECLABEL_BY_NAME('P1', 'L1') )
```

Joe の新しい LBAC 信用証明情報のため、Joe はセキュリティー・ラベル L1 により保護されるデータに書き込むことができます。そのため、L1 の追加は許可されます。表は次のようになります。

表 20. 3 回目の INSERT ステートメント後の例示表 T1 の値

LASTNAME	**LABEL**
Rjaibi	L2
Miller	L2
Bird	L1

LBAC 保護データの更新

LBAC 信用証明情報が、データへの書き込みアクセスを許可していなければ、データを更新することはできません。保護された行を更新する場合、LBAC 信用証明情報が行への読み取りアクセスも許可していなければなりません。

保護された列の更新

保護された列にあるデータの更新を試行する際、LBAC 信用証明情報はその列を保護するセキュリティー・ラベルと比較されます。行われる比較は書き込みアクセスに対するものです。書き込みアクセスがブロックされる場合にはエラーが戻され、ステートメントは失敗します。ブロックされない場合、更新は継続します。

LBAC 信用証明情報がセキュリティー・ラベルと比較される方法の詳細は、LBAC セキュリティー・ラベルの比較方法に関するトピックで説明されています。

例:

列 DEPTNO がセキュリティー・ラベル L2 により保護され、列 PAYSCALE がセキュリティー・ラベル L3 により保護される表 T1 があると想定します。T1 (そのデータを含む) は次のようになります。

表 21. 表 T1

EMPNO	**LASTNAME**	**DEPTNO** *保護ラベルは* **L2**	**PAYSCALE** *保護ラベルは* **L3**
1	Rjaibi	11	4
2	Miller	11	7
3	Bird	11	9

ユーザー Lhakpa には LBAC 信用証明情報がありません。Lhakpa は次の SQL ステートメントを発行します。

```
UPDATE T1 SET EMPNO = 4
   WHERE LASTNAME = "Bird"
```

このステートメントは、保護された列を更新しないため、エラーなく実行されます。 T1 は次のようになります。

表 22. 更新後の表 T1

EMPNO	**LASTNAME**	**DEPTNO** *保護ラベルは* **L2**	**PAYSCALE** *保護ラベルは* **L3**
1	Rjaibi	11	4
2	Miller	11	7
4	Bird	11	9

Lhakpa は今度は次の SQL ステートメントを発行します。

```
UPDATE T1 SET DEPTNO = 55
   WHERE LASTNAME = "Miller"
```

DEPTNO は保護されていて Lhakpa には LBAC 信用証明情報がないため、このステートメントは失敗し、エラーが戻されます。

Lhakpa に LBAC 信用証明情報が付与されていて、それが以下の表で要約されているアクセスを許可すると想定します。それらの信用証明情報がどんなもので、セキュリティー・ラベルにどんなエレメントが入っているかは、この例では重要ではありません。

データを保護しているセキュリティー・ラベル	**読み取りの可否**	**書き込みの可否**
L2	いいえ	はい
L3	いいえ	いいえ

Lhakpa は次の SQL ステートメントを再度発行します。

```
UPDATE T1 SET DEPTNO = 55
   WHERE LASTNAME = "Miller"
```

今度は、Lhakpa の LBAC 信用証明情報が、列 DEPTNO を保護しているセキュリティー・ラベルにより保護されるデータへの書き込みを許可しているため、ステートメントはエラーなく実行されます。その同じ列から読み取りができないことは関係ありません。 T1 のデータは次のようになります。

表 23. 2 回目の更新後の表 T1

EMPNO	**LASTNAME**	**DEPTNO** *保護ラベルは* **L2**	**PAYSCALE** *保護ラベルは* **L3**
1	Rjaibi	11	4

表 23. 2 回目の更新後の表 T1 (続き)

EMPNO	LASTNAME	DEPTNO *保護ラベルは* L2	PAYSCALE *保護ラベルは* L3
2	Miller	55	7
4	Bird	11	9

今度は Lhakpa は次の SQL ステートメントを発行します。

```
UPDATE T1 SET DEPTNO = 55, PAYSCALE = 4
   WHERE LASTNAME = "Bird"
```

列 PAYSCALE はセキュリティー・ラベル L3 により保護され、Lhakpa の LBAC 信用証明情報は Lhakpa がその列に書き込むことを許可しません。 Lhakpa はその列に書き込むことができないため、更新は失敗し、データは変更されません。

保護された行の更新

ユーザーの LBAC 信用証明情報が、ある行の読み取りを許可していない場合には、そのユーザーにとってはその行は存在していないかのようになるため、そのユーザーがその行を更新する方法はありません。読み取ることができる行においては、その更新を行うためには、行への書き込みもできなければなりません。

行の更新を試行する際、書き込み用の LBAC 信用証明情報はその行を保護するセキュリティー・ラベルと比較されます。書き込みアクセスがブロックされる場合、更新は失敗し、エラーが戻されます。書き込みアクセスがブロックされない場合には、更新は継続します。

実行される更新は、DB2SECURITYLABEL のデータ・タイプを持つ列の処理を除き、無保護の行への更新と同様に行われます。その列の値を明示的に設定しない場合、そこには書き込みアクセス用に保持しているセキュリティー・ラベルが自動的に設定されます。書き込みアクセス用のセキュリティー・ラベルを持っていない場合、エラーが戻され、ステートメントは失敗します。

更新で DB2SECURITYLABEL のデータ・タイプを持つ列を明示的に設定した場合には、LBAC 信用証明情報は再度チェックされます。実行しようとしている更新で、現行の LBAC 信用証明情報では書き込みが許可されない行が作成されることになる場合の処理は、表を保護しているセキュリティー・ポリシーによって異なります。セキュリティー・ポリシーに RESTRICT NOT AUTHORIZED WRITE SECURITY LABEL オプションがある場合は、更新が失敗し、エラーが戻されます。セキュリティー・ポリシーに RESTRICT NOT AUTHORIZED WRITE SECURITY LABEL オプションがない場合や、代わりに OVERRIDE NOT AUTHORIZED WRITE SECURITY LABEL オプションがある場合は、指定したセキュリティー・ラベルが無視され、書き込みアクセス用に保持しているセキュリティー・ラベルがあれば、そのセキュリティー・ラベルが代わりに使用されます。書き込みアクセス用のセキュリティー・ラベルを保持していない場合は、エラーが戻されます。

例:

表 T1 が、P1 という名前のセキュリティー・ポリシーにより保護され、データ・タイプが DB2SECURITYLABEL である LABEL という名前の列を持つと想定します。

T1 (そのデータを含む) は次のようになります。

表 24. 表 T1

EMPNO	LASTNAME	DEPTNO	LABEL
1	Rjaibi	11	L1
2	Miller	11	L2
3	Bird	11	L3

ユーザー Jenni は、セキュリティー・ラベル L0 および L1 により保護されるデータへの読み取りおよび書き込みを許可するものの、他のセキュリティー・ラベルにより保護されるデータに対する読み取りおよび書き込みは許可しない LBAC 信用証明情報を持っていると想定します。 Jenni が読み取りおよび書き込み両方のために保持しているセキュリティー・ラベルは L0 です。Jenni の信用証明情報全体の詳細、およびラベルにどんなエレメントが入っているかは、この例では重要ではありません。

Jenni は次の SQL ステートメントを発行します。

```
SELECT * FROM T1
```

Jenni の表には 1 行だけ表示されます。

表 25. Jenni の SELECT 照会の結果

EMPNO	LASTNAME	DEPTNO	LABEL
1	Rjaibi	11	L1

ラベル L2 および L3 により保護される行は結果セットには組み込まれません。なぜなら Jenni の LBAC 信用証明情報は、これらの行の読み取りを Jenni に許可しないからです。 Jenni にとっては、これらの行は存在していないかのようになります。

Jenni は以下の SQL ステートメントを発行します。

```
UPDATE T1 SET DEPTNO = 44 WHERE DEPTNO = 11;
SELECT * FROM T1;
```

照会により戻される結果セットは、以下のようになります。

表 26. Jenni の UPDATE & SELECT 照会の結果

EMPNO	LASTNAME	DEPTNO	LABEL
1	Rjaibi	44	L0

表の実際のデータは次のようになります。

表 27. 表 T1

EMPNO	LASTNAME	DEPTNO	LABEL
1	Rjaibi	44	L0

表 27. 表 T1 (続き)

EMPNO	LASTNAME	DEPTNO	LABEL
2	Miller	11	L2
3	Bird	11	L3

ステートメントはエラーなく実行されましたが、最初の行のみが影響を受けました。 2 番目および 3 番目の行を Jenni は読み取ることができないため、それらの行が WHERE 節の条件を満たす場合であっても、それらはステートメントによる更新で選択されません。

LABEL 列が UPDATE ステートメントで明示的に設定されていなかったにもかかわらず、更新された行のその列の値が変更されたことに注目してください。その列には、Jenni が書き込み用に保持しているセキュリティー・ラベルが設定されます。

今度は Jenni は、どのセキュリティー・ラベルによって保護されるデータへの読み取りをも許可する LBAC 信用証明情報が付与されます。 Jenni の書き込み用 LBAC 信用証明情報は変更されません。依然として、Jenni は L0 および L1 により保護されるデータのみに書き込むことができます。

Jenni は再度次の SQL ステートメントを発行します。

```
UPDATE T1 SET DEPTNO = 44 WHERE DEPTNO = 11
```

今度は、2 番目と 3 番目の行のために更新は失敗します。 Jenni はそれらの行の読み取りができるため、それらはステートメントによる更新で選択されます。しかし、それらはセキュリティー・ラベル L2 および L3 により保護されているため、Jenni はそれらに書き込むことはできません。更新は行われず、エラーが戻されます。

Jenni はここで次の SQL ステートメントを発行します。

```
UPDATE T1
 SET DEPTNO = 55, LABEL = SECLABEL_BY_NAME( 'P1', 'L2' )
 WHERE LASTNAME = "Rjaibi"
```

ステートメント内の SECLABEL_BY_NAME 関数は、L2 という名前のセキュリティー・ラベルを戻します。Jenni は、最初の行を保護するセキュリティー・ラベルの明示的設定を試行します。 Jenni の LBAC 信用証明情報は、最初の行の読み取りを許可するため、その行は更新のために選択されます。 Jenni の LBAC 信用証明情報は、セキュリティー・ラベル L0 により保護されている行への書き込みを許可するため、Jenni はその行を更新することが許可されます。しかし、Jenni の LBAC 信用証明情報は、セキュリティー・ラベル L2 により保護されている行への書き込みを許可しないため、Jenni は列 LABEL にその値を設定することは許可されません。ステートメントは失敗し、エラーが戻されます。その行内の列は更新されません。

Jenni はここで次の SQL ステートメントを発行します。

```
UPDATE T1 SET LABEL = SECLABEL_BY_NAME( 'P1', 'L1' ) WHERE LASTNAME = "Rjaibi"
```

Jenni はセキュリティー・ラベル L1 により保護されている行への書き込みができるため、ステートメントは成功します。

T1 は次のようになります。

表 28. 表 T1

EMPNO	LASTNAME	DEPTNO	LABEL
1	Rjaibi	44	L1
2	Miller	11	L2
3	Bird	11	L3

保護された列を含む保護された行の更新

保護された行を持つ表内の保護された列の更新を試行する場合には、LBAC 信用証明情報は、更新による影響を受けるすべての保護された列の書き込みを許可していなければなりません。それ以外の場合、更新は失敗し、エラーが戻されます。これは、保護された列の更新に関する前のセクションで説明されているとおりです。更新による影響を受けるすべての保護された列の更新が許可されている場合でも、LBAC 信用証明情報が読み取りおよび書き込みの両方を許可する行しか更新できません。これは、保護された行の更新に関する前のセクションで説明されているとおりです。DB2SECURITYLABEL のデータ・タイプを持つ列の処理は、保護された列が更新による影響を受けるかどうかに関わりなく同じです。

DB2SECURITYLABEL のデータ・タイプを持つ列がそれ自体保護された列である場合には、LBAC 信用証明情報は、その列への書き込みを許可しなければなりません。そうでない場合、その表のどの行も更新できません。

LBAC 保護データの削除またはドロップ

LBAC によって保護されている表内のデータを削除できるかどうかは、ご使用の LBAC 信用証明情報に依存します。

保護された行の削除

ユーザーの LBAC 信用証明情報が、ある行の読み取りを許可しない場合には、そのユーザーにとってはその行は存在していないかのようになるため、そのユーザーがそれを削除する方法はありません。読み取ることができる行を削除するには、LBAC 信用証明情報がその行への書き込みも許可していなければなりません。保護された列を持つ表の任意の行を削除するには、表内の保護されたすべての列への書き込みを許可する LBAC 信用証明情報を持っていなければなりません。

行の削除を試行する際、書き込み用の LBAC 信用証明情報はその行を保護するセキュリティー・ラベルと比較されます。保護セキュリティー・ラベルが LBAC 信用証明情報による書き込みアクセスをブロックする場合、DELETE ステートメントは失敗し、エラーが戻され、行は削除されません。

例

保護された表 T1 には以下の行があります。

LASTNAME	DEPTNO	LABEL
Rjaibi	55	L2
Miller	77	L1

LASTNAME	DEPTNO	LABEL
Bird	55	L2
Fielding	77	L3

Pat は、アクセスが次の表に要約されるような LBAC 信用証明情報を持っていると想定します。

セキュリティー・ラベル	読み取りアクセス	書き込みアクセス
L1	はい	はい
L2	はい	いいえ
L3	いいえ	いいえ

Pat の LBAC 信用証明情報とセキュリティー・ラベルの厳密な詳細は、この例では重要ではありません。

Pat は次の SQL ステートメントを発行します。

```
SELECT * FROM T1 WHERE DEPTNO != 999
```

ステートメントが実行され、以下の結果セットを戻します。

LASTNAME	DEPTNO	LABEL
Rjaibi	55	L2
Miller	77	L1
Bird	55	L2

T1 の最後の行は結果に含まれません。なぜなら Pat はその行への読み取りアクセスがないからです。 Pat にとっては、その行は存在していないかのようになります。

Pat は次の SQL ステートメントを発行します。

```
DELETE FROM T1 WHERE DEPTNO != 999
```

Pat は、最初と 3 番目の行への書き込みアクセスがありません。その 2 つの行は L2 により保護されています。そのため、Pat は行の読み取りはできたとしても、それらの削除はできません。 DELETE ステートメントは失敗し、行は削除されません。

Pat は次の SQL ステートメントを発行します。

```
DELETE FROM T1 WHERE DEPTNO = 77;
```

Pat は、LASTNAME 列が Miller である行に書き込むことができるため、このステートメントは成功します。それは、このステートメントにより選択される唯一の行です。LASTNAME 列が Fielding である行は選択されません。なぜなら Pat の LBAC 信用証明情報はその行への読み取りを許可しないからです。その行は削除の対象としては決して考慮されないため、エラーは発生しません。

表の実際の行は、次のようになります。

LASTNAME	DEPTNO	LABEL
Rjaibi	55	L2
Bird	55	L2
Fielding	77	L3

保護された列を持つ行の削除

保護された列を持つ表の任意の行を削除するには、表内の保護されたすべての列への書き込みを許可する LBAC 信用証明情報を持っていなければなりません。LBAC 信用証明情報が書き込みを許可しない行が表にある場合には、削除は失敗し、エラーが戻されます。

表に保護された列および保護された行がある場合、特定の行を削除するには、表の保護されたすべての列への書き込みと、さらに削除したい行に対する読み取りおよび書き込みを許可する LBAC 信用証明情報を持っていなければなりません。

例

保護された表 T1 では、列 DEPTNO はセキュリティー・ラベル L2 により保護されています。T1 には以下の行が含まれます。

LASTNAME	DEPTNO *L2 により保護される*	LABEL
Rjaibi	55	L2
Miller	77	L1
Bird	55	L2
Fielding	77	L3

ユーザー Benny は次の表で要約されるアクセスを許可する LBAC 信用証明情報を持っていると想定します。

セキュリティー・ラベル	読み取りアクセス	書き込みアクセス
L1	はい	はい
L2	はい	いいえ
L3	いいえ	いいえ

Benny の LBAC 信用証明情報とセキュリティー・ラベルの厳密な詳細は、この例では重要ではありません。

Benny は次の SQL ステートメントを発行します。

```
DELETE FROM T1 WHERE DEPTNO = 77
```

Benny は列 DEPTNO への書き込みアクセスがないため、このステートメントは失敗します。

ここで、Benny が次の表で要約されているアクセスを持つように、Benny の LBAC 信用証明情報は変更されます。

セキュリティー・ラベル	読み取りアクセス	書き込みアクセス
L1	はい	はい
L2	はい	はい
L3	はい	いいえ

Benny は次の SQL ステートメントを再度発行します。

```
DELETE FROM T1 WHERE DEPTNO = 77
```

今回、Benny には、列 DEPTNO への書き込みアクセスがあるため、削除は継続します。 DELETE ステートメントは、LASTNAME 列に Miller の値がある行のみを選択します。 LASTNAME 列に Fielding の値がある行は選択されません。なぜなら Benny の LBAC 信用証明情報はその行の読み取りを許可しないからです。ステートメントによる削除でその行が選択されていないため、Benny がその行に書き込むことができなくても関係ありません。

選択された 1 行はセキュリティー・ラベル L1 により保護されます。Benny の LBAC 信用証明情報は、L1 により保護されるデータへの書き込みを許可するため、削除は成功します。

表 T1 の実際の行は、次のようになります。

LASTNAME	**DEPTNO** **L2** *により保護される*	**LABEL**
Rjaibi	55	L2
Bird	55	L2
Fielding	77	L3

保護データのドロップ

LBAC 信用証明情報が、セキュリティー・ラベルにより保護される列への書き込みを許可していなければ、その列をドロップできません。

DB2SECURITYLABEL のデータ・タイプを持つ列は、表からドロップできません。これを除去するには、最初に表からセキュリティー・ポリシーをドロップする必要があります。セキュリティー・ポリシーをドロップするとき、表は LBAC で保護されなくなり、列のデータ・タイプは自動的に DB2SECURITYLABEL から VARCHAR(128) FOR BIT DATA に変更されます。その後、列をドロップできます。

LBAC 信用証明情報は、保護データが含まれる表全体またはデータベース全体のドロップを妨げることはありません。ある表またはデータベースをドロップするための正常な権限を持っている場合には、データベースに保護データが含まれている場合であっても、ドロップするために LBAC 信用証明情報は必要ありません。

データからの LBAC 保護の除去

表からセキュリティー・ポリシーを除去するには、SECADM 権限がなければなりません。表からセキュリティー・ポリシーを除去するには、ALTER TABLE ステートメントの DROP SECURITY POLICY 節を使用します。これにより、表のすべての行と列の保護も自動的に除去されます。

行から保護を除去する

保護される行を持つ表では、すべての行がセキュリティー・ラベルによって保護されている必要があります。個別の行から LBAC 保護を除去することはできません。

表からセキュリティー・ポリシーを除去するという例外を除いては、タイプ DB2SECURITYLABEL の列を変更したり除去することはできません。

列から保護を除去する

列の保護は、SQL ステートメント ALTER TABLE の DROP COLUMN SECURITY 節を使用して除去することができます。列から保護を除去するには、表を変更するのに必要な通常の特権および権限に加えて、その列に対する読み取り/書き込みを行うための LBAC 信用証明情報を持っている必要があります。

第 6 章 セキュリティー情報のためのシステム・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、 (データベース作成時に作成される) システム・カタログという 1 組のビュー内に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。

次のビューおよび表関数は、ユーザーが保持する特権、特権を認可するユーザーの ID、およびオブジェクト所有者に関する情報をリストします。

SYSCAT.COLAUTH
列の特権のリスト

SYSCAT.DBAUTH
データベースの特権のリスト

SYSCAT.INDEXAUTH
索引の特権のリスト

SYSCAT.MODULEAUTH
モジュールの特権のリスト

SYSCAT.PACKAGEAUTH
パッケージの特権のリスト

SYSCAT.PASSTHRUAUTH
サーバーの特権のリスト

SYSCAT.ROLEAUTH
ロールの特権のリスト

SYSCAT.ROUTINEAUTH
ルーチン (関数、メソッド、およびストアード・プロシージャー) の特権のリスト

SYSCAT.SCHEMAAUTH
スキーマの特権のリスト

SYSCAT.SEQUENCEAUTH
シーケンスの特権のリスト

SYSCAT.SURROGATEAUTHIDS
別の許可 ID が代理を務めることのできる許可 ID をリストします。

SYSCAT.TABAUTH
表とビューの特権のリスト

SYSCAT.TBSPACEAUTH
表スペースの特権のリスト

SYSCAT.VARIABLEAUTH
変数の特権のリスト

SYSCAT.WORKLOADAUTH
ワークロードの特権のリスト

SYSCAT.XSROBJECTAUTH
XSR オブジェクトの特権のリスト

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 SYSCTRL、および SYSMON はシステム・カタログにリストされません。

CREATE ステートメントと GRANT ステートメントを使うと、特権に関する情報はシステム・カタログ表に格納されます。 ACCESSCTRL および SECADM 権限を持つユーザーは、システム・カタログ・ビューに対する SELECT 特権の GRANT と取り消しを行うことができます。

付与された特権を持つ許可名の検索

PRIVILEGES および他の管理ビューを使用して、データベースに特権を付与された許可名に関する情報を検索することができます。

このタスクについて

例えば次の照会では、付与された明示特権および許可 ID に加えて、 PRIVILEGES 管理ビューから他の情報を検索します。

```
SELECT AUTHID, PRIVILEGE, OBJECTNAME, OBJECTSCHEMA, OBJECTTYPE
       FROM SYSIBMADM.PRIVILEGES
```

次の照会は、AUTHORIZATIONIDS 管理ビューを使用して、特権または権限を付与されたすべての許可 ID を検索し、それらのタイプを示します。

```
SELECT AUTHID, AUTHIDTYPE FROM SYSIBMADM.AUTHORIZATIONIDS
```

SYSIBMADM.OBJECTOWNERS 管理ビューおよび SYSPROC.AUTH_LIST_GROUPS_FOR_AUTHID 表関数を使用してセキュリティーに関する情報を検索することもできます。

バージョン 9.1 より前には、すべての特権に関する情報が、いずれか 1 つのシステム・カタログ・ビューに含まれることはありませんでした。バージョン 9.1 より前のリリースでは、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、システムのセキュリティー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必要があります。これによって、有効でなくなった許可名を識別できます。

注: リモート・データベース・クライアントをサポートする場合、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。

このタスクについて

```
SELECT DISTINCT GRANTEE, GRANTEETYPE 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

PRIVILEGES および他の管理ビューを使用して、データベースに特権を付与された許可名に関する情報を検索することができます。

このタスクについて

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 (およびそのタイプ) を検索します。

```
SELECT DISTINCT AUTHID, AUTHIDTYPE FROM SYSIBMADM.PRIVILEGES
    WHERE OBJECTNAME = 'EMPLOYEE' AND OBJECTSCHEMA = 'JAMES'
```

バージョン 9.1 より前のリリースでは、次の照会は同じ情報を検索します。

```
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 IN ('G','Y'))
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。

一部の許可名は、個別のユーザーだけでなく、グループ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・ビュー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。

このタスクについて

PRIVILEGES および他の管理ビューを使用して、データベースに特権を付与された許可名に関する情報を検索することができます。例えば、次の照会は現行セッションの許可 ID に付与されたすべての特権を検索します。

```
SELECT * FROM SYSIBMADM.PRIVILEGES
 WHERE AUTHID = SESSION_USER AND AUTHIDTYPE = 'U'
```

このステートメント内のキーワード SESSION_USER は、現行ユーザーの許可名の値と等しい特殊レジスターです。

バージョン 9.1 より前のリリースでは、次の例は類似の情報を検索します。例えば、以下のステートメントは、個々の許可名 JAMES 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 = 'JAMES' AND GRANTEETYPE = 'U'
```

表の特権のうちユーザー JAMES 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  = 'JAMES'
```

以下のステートメントは、ユーザー JAMES 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  = 'JAMES'
```

システム・カタログ・ビューのセキュリティー

システム・カタログ・ビューはデータベース内のすべてのオブジェクトを記述するので、ユーザーが機密データを持つとき、それらのアクセスを制限する場合があります。

このタスクについて

以下の権限には、すべてのカタログ表に対する SELECT 特権があります。

- ACCESSCTRL
- DATAACCESS
- DBADM
- SECADM
- SQLADM

また、以下のインスタンス・レベル権限では、
SYSCAT.BUFFERPOOLS、SYSCAT.DBPARTITIONGROUPS

、SYSCAT.DBPARTITIONGROUPDEF、SYSCAT.PACKAGES、
および SYSCAT.TABLES を SELECT
することができます。

- SYSADM
- SYSCTRL
- SYSMAINT
- SYSMON

CREATE DATABASE ... RESTRICTIVE コマンドを使用して、特権が PUBLIC に自動的に付与されないデータベースを作成することができます。この場合、次の通常のデフォルト認可アクションはいずれも発生しません。

- CREATETAB
- BINDADD
- CONNECT
- IMPLICIT_SCHEMA
- スキーマ SQLJ 中のすべてのプロシージャーに関する GRANT 付きの EXECUTE
- スキーマ SYSPROC 中のすべての関数とプロシージャーに関する GRANT 付きの EXECUTE
- NULLID スキーマ内で作成されたすべてのパッケージに対する BIND
- NULLID スキーマ中に作成されたすべてのパッケージに関する EXECUTE
- スキーマ SQLJ に関する CREATEIN
- スキーマ NULLID に関する CREATEIN
- 表スペース USERSPACE1 に関する USE
- SYSIBM カタログ表への SELECT アクセス
- SYSCAT カタログ・ビューへの SELECT アクセス
- SYSIBMADM 管理ビューへの SELECT アクセス
- SYSSTAT カタログ・ビューへの SELECT アクセス
- SYSSTAT カタログ・ビューへの UPDATE アクセス

RESTRICTIVE オプションを使用してデータベースを作成した場合、PUBLIC には権限が付与されません。以下の照会を実行して、PUBLIC でアクセス可能なスキーマがないことを確認できます。

```
SELECT DISTINCT OBJECTSCHEMA FROM SYSIBMADM.PRIVILEGES WHERE AUTHID='PUBLIC'

OBJECTSCHEMA
------------
```

DB2 データベース・マネージャーのバージョン 9.1 より前のリリースでは、データベースの生成時に、システム・カタログ・ビューに対する SELECT 特権が PUBLIC に付与されます。多くの場合、これによってセキュリティー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・ビューについての SELECT 特権の付与と取り消しは、他のビューの場合と同じ方法で行いますが、そのためには ACCESSCTRL または SECADM 権限が必要です。

いずれのユーザーも他のユーザーがアクセスするオブジェクトを知ることができないようにする場合には、最低でも、次のカタログおよび管理ビューへのアクセスを制限することを検討すべきです。

- SYSCAT.COLAUTH
- SYSCAT.DBAUTH
- SYSCAT.INDEXAUTH
- SYSCAT.PACKAGEAUTH
- SYSCAT.PASSTHRUAUTH
- SYSCAT.ROUTINEAUTH
- SYSCAT.SCHEMAAUTH
- SYSCAT.SECURITYLABELACCESS
- SYSCAT.SECURITYPOLICYEXEMPTIONS
- SYSCAT.SEQUENCEAUTH
- SYSCAT.SURROGATEAUTHIDS
- SYSCAT.TABAUTH
- SYSCAT.TBSPACEAUTH
- SYSCAT.XSROBJECTAUTH
- SYSIBMADM.AUTHORIZATIONIDS
- SYSIBMADM.OBJECTOWNERS
- SYSIBMADM.PRIVILEGES

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。

各列にどの統計が集められているかも調べてください。システム・カタログに記録される統計には、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ることがあります。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・ビューについての 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・ビューに対するアクセスを限定する場合は、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ビューを定義することができます。

例えば、次のビュー 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、現行セッションの許可名の値と等しくなります。

以下のステートメントは、このビュー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、次の 2 つのステートメントの発行によってビューおよび基本表についての 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
REVOKE SELECT ON TABLE SYSIBM.SYSTABAUTH FROM PUBLIC
```

第 7 章 ファイアウォール・サポート

ファイアウォール は、ネットワーク・ゲートウェイ・サーバーに位置し、システムまたはネットワークへの無許可アクセスを防ぐために使用される、関連したプログラムのセットです。

ファイアウォールには、以下の 4 つのタイプがあります。

1. ネットワーク・レベル、パケット・フィルター、またはスクリーニング・ルーター・ファイアウォール
2. 従来のアプリケーション・レベルのプロキシー・ファイアウォール
3. 回線レベル、または透過性のプロキシー・ファイアウォール
4. Stateful Multi-Layer Inspection (SMLI) ファイアウォール

上記でリストされたファイアウォールのタイプのいずれかに入る既存のファイアウォール製品もあります。他にも、上記でリストされたタイプの組み合わせとなる、多くのファイアウォール製品があります。

スクリーニング・ルーター・ファイアウォール

スクリーニング・ルーター・ファイアウォールは、ネットワーク・レベルまたはパケット・フィルター・ファイアウォールとも呼ばれます。このようなファイアウォールは、着信パケットをプロトコル属性によってスクリーニングすることにより動作します。スクリーニングされるプロトコル属性には、送信元または宛先アドレス、プロトコルのタイプ、送信元または宛先のポート、または他のプロトコルに特有の属性が含まれます。

すべてのファイアウォール・ソリューション (SOCKS を除く) では、 DB2 データベースによって使用されるすべてのポートを着信および発信パケットのために開けておく必要があります。 DB2 データベースは、DB2 データベース・ツールによって使用される DB2 Administration Server (DAS) 用として、ポート 523 を使用します。サービス・ファイルを使用して、サーバー・データベース・マネージャー構成ファイル内のサービス名をそのポート番号にマップし、すべてのサーバー・インスタンスが使用するポートを判別してください。

アプリケーション・プロキシー・ファイアウォール

プロキシーまたはプロキシー・サーバーは、 Web クライアントと Web サーバーの間の中継地点として動作する技術です。プロキシー型ファイアウォールは、クライアントからの要求に対するゲートウェイの役割を果たします。

クライアントの要求がファイアウォールで受信されると、最終のサーバー宛先アドレスがプロキシー・ソフトウェアによって判別されます。アプリケーション・プロキシーがアドレスを変換し、必要に応じてさらにアクセス・コントロール検査およびロギングを行い、クライアントに代わってサーバーに接続します。

ファイアウォール・マシン上の DB2 Connect 製品は、宛先サーバーに対するプロキシーの役割を果たすことができます。また、最終的な宛先サーバーに対するホップ・サーバーとして動作する、ファイアウォール上の DB2 データベース・サーバーは、アプリケーション・プロキシーに似た役割を果たします。

回線レベルのファイアウォール

回線レベルのファイアウォールは、透過性プロキシー・ファイアウォールとも呼ばれます。

透過性プロキシー・ファイアウォールは、プロキシー認証および識別に必要とされる以上には、要求または応答を変更しません。透過性プロキシー・ファイアウォールとしては、 SOCKS があります。

DB2 データベース・システムは SOCKS バージョン 4 をサポートします。

Stateful Multi-Layer Inspection (SMLI) ファイアウォール

Stateful Multi-Layer Inspection (SMLI) ファイアウォールは、オープン・システム間相互接続 (OSI) モデルの 7 層すべてを調べる、パケット・フィルター操作の高性能な形式を使用します。

各パケットが調べられ、類似したパケットの既知の状態と比較されます。スクリーニング・ルーター・ファイアウォールがパケット・ヘッダーのみを調べるのに対し、 SMLI ファイアウォールはデータも含むパケット全体を調べます。

第 8 章 セキュリティー・プラグイン

DB2 データベース・システムでの認証は、セキュリティー・プラグイン を使用して行われます。セキュリティー・プラグインは、動的にロード可能なライブラリーであり、認証セキュリティー・サービスを提供します。

グループ検索プラグイン
特定のユーザーのグループ・メンバーシップ情報を検索します。

ユーザー ID/パスワード認証プラグイン
ユーザー ID/パスワード認証プラグインを使用して、以下の認証タイプが実装されます。

- CLIENT
- SERVER
- SERVER_ENCRYPT
- DATA_ENCRYPT
- DATA_ENCRYPT_CMP

上記のような認証タイプによって、ユーザー認証がどこでどのように行われるかが決まります。使用される認証タイプは、次の方法によって決まります。

- 接続操作またはアタッチ操作の場合、**srvcon_auth** 構成パラメーターの値を指定すると、その値は **authentication** 構成パラメーターの値よりも優先されます。
- それ以外の場合は、**authentication** 構成パラメーターの値が使用されます。

GSS-API 認証プラグイン
GSS-API の正式名称は、Generic Security Service Application Program Interface Version 2 (IETF RFC2743) および Generic Security Service API Version 2: C-Bindings (IETF RFC2744) です。Kerberos プロトコルは、GSS-API 認証メカニズムを実装する主要な手段です。以下の認証タイプは、GSS-API 認証プラグインを使用してインプリメントされます。

- KERBEROS
- GSSPLUGIN
- KRB_SERVER_ENCRYPT
- GSS_SERVER_ENCRYPT

KRB_SERVER_ENCRYPT および GSS_SERVER_ENCRYPT は、GSS-API 認証およびユーザー ID/パスワード認証の両方をサポートします。ただし、GSS-API 認証の方が望ましい認証タイプです。クライアント・サイドの Kerberos サポートは、Solaris、AIX、HP-UX (64 ビットのみ)、Windows、および Linux オペレーティング・システムで使用できます。Windows オペレーティング・システムでは、デフォルトで Kerberos サポートが使用可能になっています。

DB2 データベース・マネージャーは、クライアントとサーバーの両方でこれらのプラグインをサポートします。

注: 認証タイプによって、ユーザーがどこでどのように認証されるかが決まります。特定の認証タイプを使用するには、**authentication** データベース・マネージャー構成パラメーターの値を設定します。

各々のプラグインを単独で使用するか、または他のプラグインと一緒に使用することができます。例えば、特定のサーバー・サイドの認証プラグインを使用するものの、クライアントおよびグループの認証では DB2 のデフォルト値を受け入れるようにすることができます。一方、グループ検索またはクライアントの認証プラグインのみを使用し、サーバー・サイドのプラグインを使用しないようにすることもできます。

GSS-API 認証を使用する場合は、プラグインは、クライアントとサーバーの両方に必要です。

認証のデフォルトの動作では、オペレーティング・システム・レベルの認証メカニズムを実装するユーザー ID/パスワード・プラグインが使用されます。

DB2 データベース製品には、グループ検索用、ユーザー ID/パスワード認証用、および GSS-API 認証用のプラグインが組み込まれています。独自のプラグインを作成するか、またはサード・パーティーからプラグインを購入することによって、DB2 のクライアントおよびサーバーの認証動作をさらにカスタマイズすることができます。

DB2 クライアント上のセキュリティー・プラグインのデプロイメント

DB2 クライアントは、1 つのグループ検索プラグインおよび 1 つのユーザー ID/パスワード認証プラグインをサポートできます。

一方、GSS-API 認証プラグインを使用するクライアントは、DB2 サーバーに実装された GSS-API プラグインのリストをスキャンすることで、使用するプラグインを特定します。クライアントに実装された GSS-API 認証プラグインと一致する、最初の認証プラグイン名が選択されます。実装されたサーバー側 GSS-API プラグインのリストは、**srvcon_gssplugin_list** データベース・マネージャー構成パラメーターを使用して指定します。以下の図は、DB2 クライアント上のセキュリティー・プラグイン・インフラストラクチャーを描写しています。

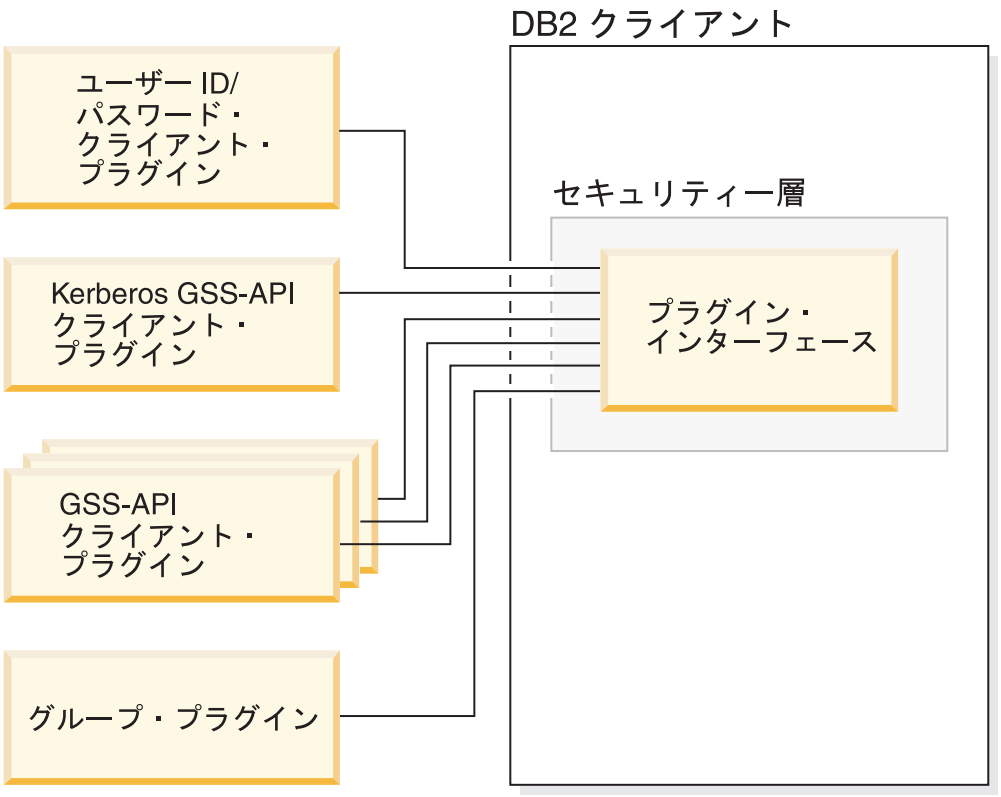


図 5. DB2 クライアント上のセキュリティー・プラグインのデプロイメント

DB2 サーバー上のセキュリティー・プラグインのデプロイメント

DB2 サーバーは 1 つのグループ検索プラグイン、1 つのユーザー ID/パスワード認証プラグイン、および複数の GSS-API プラグインをサポートできます。使用可能な GSS-API プラグインは、**srvcon_gssplugin_list** データベース・マネージャー構成パラメーターの値のリストとして指定できます。ただし、このリストの中の 1 つの GSS-API プラグインのみ、Kerberos プラグインにすることができます。

データベース・サーバーでサーバー・サイドのセキュリティー・プラグインをデプロイすることに加えて、データベース・サーバーでクライアント認証プラグインを実装しなければならない場合があります。 **db2start** および **db2trc** コマンドなどのインスタンス・レベルの操作の実行時には、DB2 データベース・マネージャーはクライアント認証プラグインを使用して、その操作に対する許可検査を実行します。したがって、サーバーの認証プラグインに対応するクライアント認証プラグインをインストールすべき場合があります。このプラグイン名は、サーバーの **authentication** データベース・マネージャー構成パラメーターによって指定します。

authentication と **srvcon_auth** 構成パラメーターは、異なる値に設定できます。このシナリオでは、一方のメカニズムがデータベース接続の認証用に使用され、他方のメカニズムがローカル許可用に使用されます。

このアプローチでは、以下の方法が最も一般的です。

- **srvcon_auth** 構成パラメーターを GSSPLUGIN に設定します。さらに、
- **authentication** 構成パラメーターを SERVER に設定します。

srvcon_auth 構成パラメーターは、着信接続で使用される認証タイプをオーバーライドするための手段となります。これらの接続では、**srvcon_auth** 構成パラメーターで指定した認証メソッドを使用しますが、この値が空のままである場合は、**authentication** パラメーターの値が代わりに使用されます。

データベース・サーバー上でクライアント認証プラグインを使用しない場合、**db2start** コマンドなどのインスタンス・レベルの操作は失敗します。

以下の図は、DB2 サーバー上のセキュリティー認証プラグイン・インフラストラクチャーの概要です。

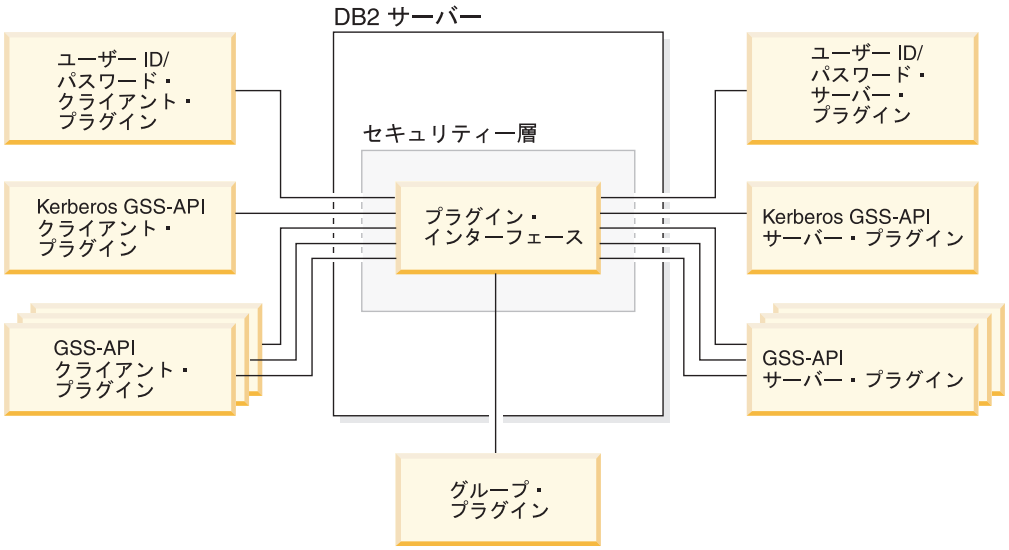


図 6. DB2 サーバー上のセキュリティー・プラグインのデプロイメント

セキュリティー・プラグインの有効化

データベース・マネージャー構成パラメーターを設定することにより、各認証メカニズムに使用するプラグインを指定できます。それらのパラメーターの概要を以下の表に示します。

表 29. セキュリティー認証プラグインのデータベース・マネージャー構成パラメーター

説明	パラメーター名
クライアント・ユーザー ID/パスワード・プラグイン	CLNT_PW_PLUGIN
クライアント Kerberos プラグイン	CLNT_KRB_PLUGIN
グループ・プラグイン	GROUP_PLUGIN
ローカル許可用の GSS プラグイン	LOCAL_GSSPLUGIN
サーバー・プラグイン・モード	SRV_PLUGIN_MODE
GSS プラグインのサーバー・リスト	SRVCON_GSSPLUGIN_LIST
サーバー・ユーザー ID/パスワード・プラグイン	SRVCON_PW_PLUGIN
サーバー接続認証	SRVCON_AUTH
データベース・マネージャー認証	AUTHENTICATION

これらのパラメーターに値を設定しない場合は、グループ検索、ID/パスワード管理、および Kerberos 認証 (サーバーで **authentication** パラメーターが KERBEROS に設定されている場合) には、DB2 製品が提供するデフォルトのプラグインが使用されます。ただし、デフォルトの GSS-API プラグインは提供されません。したがって、**authentication** パラメーターに GSSPLUGIN の認証タイプを指定した場合は、**srvcon_gssplugin_list** 構成パラメーターに GSS-API 認証プラグインも指定しなければなりません。

セキュリティー・プラグインのロード

データベース・マネージャーの始動時に、データベース・マネージャー構成パラメーターで指定された、サポート対象のプラグインがすべてロードされます。

接続またはアタッチの操作中に、DB2 クライアントは、サーバーとクライアントとのネゴシエーション済みのセキュリティー・メカニズムに適合したプラグインをロードします。クライアント・アプリケーションによって、複数のセキュリティー・プラグインが並行してロードされて使用されることがあります。このような事態が発生するのは、例えば、さまざまなインスタンスからのそれぞれ異なるデータベースへの同時接続をもったスレッド化されたプログラムにおいてです。このシナリオでは、クライアント・プログラムは、GSS-API プラグイン (G1) を使用するサーバー A に初期接続を行います。サーバー A は、サポートされるプラグインのリストをクライアントに送信し、合致する G1 プラグインがクライアントでロードされます。続いてクライアント・プログラムは、別のスレッドで、GSS-API プラグイン (G2) を使用するサーバー B に接続します。クライアントは G2 について通知され、これがロードされます。こうして、G1 と G2 の両方のプラグインがクライアントで同時に使用されるようになります。

接続またはアタッチの操作以外のアクション (データベース・マネージャー構成の更新、データベース・マネージャーの開始と停止、または DB2 トレースのオン/オフなど) でも、許可のメカニズムが必要です。そのようなアクションの場合、DB2 クライアント・プログラムは、以下に示すように別のデータベース・マネージャー構成パラメーターで指定されるプラグインをロードします。

- **authentication** 構成パラメーターを GSSPLUGIN に設定した場合、DB2 データベース・マネージャーは **local_gssplugin** 構成パラメーターで指定されたプラグインを使用します。
- **authentication** 構成パラメーターを KERBEROS に設定した場合、DB2 データベース・マネージャーは **clnt_krb_plugin** 構成パラメーターで指定されたプラグインを使用します。
- それ以外の場合、DB2 データベース・マネージャーは **clnt_pw_plugin** 構成パラメーターで指定されたプラグインを使用します。

セキュリティー・プラグインは、データベース・サーバーに対する IPv4 アドレス・プロトコルによる接続と IPv6 アドレス・プロトコルによる接続の両方でサポートされます。

セキュリティー・プラグインの作成

セキュリティー認証プラグインを作成する場合は、DB2 データベース・マネージャーで使用される標準認証機能を実装する必要があります。 3 種類のプラグインについて、実装する必要がある機能を以下に示します。

グループ検索プラグイン

- ユーザーが所属するグループのリストを検出して戻します

ユーザー ID/パスワード認証プラグイン

- デフォルトのセキュリティー・コンテキストを特定します (クライアント・プラグインの場合のみ)
- パスワードの妥当性検査と、オプションで変更を行います
- 特定のストリングが正当なユーザーを表すものかどうか判別します (サーバー・プラグインの場合のみ)
- クライアントで指定されたユーザー ID またはパスワードを、サーバーへの送信前に変更します (クライアント・プラグインの場合のみ)
- 特定のユーザーに関連付けられた DB2 許可 ID を戻します

GSS-API 認証プラグイン

- デフォルトのセキュリティー・コンテキストを特定します (クライアント・プラグインの場合のみ)
- 必要な GSS-API 関数を実装します
- ユーザー ID とパスワードに基づいて初期資格情報を生成し、オプションでパスワードを変更します (クライアント・プラグインの場合のみ)
- セキュリティー・チケットを作成して受け入れます
- 特定の GSS-API セキュリティー・コンテキストに関連付けられた DB2 許可 ID を戻します

CLP または動的 SQL ステートメントを介して発行する接続ステートメントに、最大 255 文字までのユーザー ID を渡すことができます。

重要: セキュリティー・プラグインのコーディング、確認、およびテストが十分に行われないと、インストールされている DB2 データベース・システムの保全性が損なわれることがあります。DB2 データベース製品では、多数の一般的なタイプの障害に対する予防措置がとられていますが、ユーザー作成のセキュリティー・プラグインをデプロイする場合には、完全な保全性が確保されるとは限りません。

セキュリティー・プラグイン・ライブラリーの位置

セキュリティー・プラグインを (自分で作成するか、またはサード・パーティーから購入して) 取得したら、データベース・サーバー上の特定の場所にコピーします。

DB2 クライアントは、クライアント・サイドのユーザー認証プラグインを次のディレクトリーで探します。

- UNIX 32 ビット: `$DB2PATH/security32/plugin/client`
- UNIX 64 ビット: `$DB2PATH/security64/plugin/client`

- WINDOWS 32 ビットおよび 64 ビット: $DB2PATH¥security¥plugin¥*instance name*¥client

注: Windows ベースのプラットフォームの場合、サブディレクトリーの *instance name* および *client* は自動的には作成されません。これらは、インスタンスの所有者が手動で作成しなければなりません。

DB2 データベース・マネージャーは、サーバー・サイドのユーザー認証プラグインを次のディレクトリーで探します。

- UNIX 32 ビット: $DB2PATH/security32/plugin/server
- UNIX 64 ビット: $DB2PATH/security64/plugin/server
- WINDOWS 32 ビットおよび 64 ビット: $DB2PATH¥security¥plugin¥*instance name*¥server

注: Windows ベースのプラットフォームの場合、サブディレクトリーの *instance name* および *server* は自動的には作成されません。これらは、インスタンスの所有者が手動で作成しなければなりません。

DB2 データベース・マネージャーは、グループ・プラグインを次のディレクトリーで探します。

- UNIX 32 ビット: $DB2PATH/security32/plugin/group
- UNIX 64 ビット: $DB2PATH/security64/plugin/group
- WINDOWS 32 ビットおよび 64 ビット: $DB2PATH¥security¥plugin¥*instance name*¥group

注: Windows ベースのプラットフォームの場合、サブディレクトリーの *instance name* および *group* は自動的には作成されません。これらは、インスタンスの所有者が手動で作成しなければなりません。

セキュリティー・プラグインの命名規則

セキュリティー・プラグインのライブラリーには、プラットフォーム固有のファイル名拡張子が付いていなければなりません。 C または C++ で書かれたセキュリティー・プラグイン・ライブラリーには、プラットフォーム固有のファイル名拡張子が付いていなければなりません。

- Windows: .dll
- AIX: .a または .so。両方の拡張子が存在する場合、.a 拡張子が使用されます。
- Linux、 HP IPF、および Solaris: .so

注: また、ユーザーは、DB2 Universal JDBC ドライバーをもったセキュリティー・プラグインを開発することもできます。
例えば、MyPlugin というセキュリティー・プラグイン・ライブラリーがあるとします。サポートされている各オペレーティング・システムに対する適切なライブラリー・ファイル名は、次のとおりです。

- Windows 32 ビット: MyPlugin.dll
- Windows 64 ビット: MyPlugin64.dll
- AIX 32 または 64 ビット: MyPlugin.a または MyPlugin.so

- SUN 32 または 64-bit、Linux 32 または 64 ビット、IPF 上の HP 32 または 64 ビット: MyPlugin.so

注: 接尾部 "64" は、64 ビットWindows のセキュリティー・プラグインのライブラリー名にのみ必要です。

セキュリティー・プラグインの名前を使ってデータベース・マネージャー構成を更新する際は、接尾部 "64" のないライブラリーの絶対パス名を使用し、ファイル拡張子と名前の修飾パス部分を省略してください。オペレーティング・システムが何であっても、MyPlugin というセキュリティー・プラグイン・ライブラリーは、次のように登録されます。

```
UPDATE DBM CFG USING CLNT_PW_PLUGIN MyPlugin
```

セキュリティー・プラグイン名は、大文字小文字の区別があり、ライブラリー名と完全に一致している必要があります。 DB2 データベース・システムは、関連するデータベース・マネージャー構成パラメーターの値を使用してライブラリー・パスを組み立て、そのライブラリー・パスを使用してセキュリティー・プラグイン・ライブラリーをロードします。

セキュリティー・プラグイン名の競合を防ぐために、使用する認証方式、およびそのプラグインを作成した会社の識別シンボルを使って、プラグインに名前を付けてください。例えば、Foo, Inc. という会社が FOOsomemethod という認証方式をインプリメントするプラグインを作成した場合、そのプラグインには FOOsomemethod.dll といった名前を付けることができます。

プラグイン名の最大長 (ファイル拡張子および接尾部の 64 を含まない) は、 32 バイトまでに制限されています。データベース・サーバーによってサポートされるプラグインの最大数はありませんが、データベース・マネージャー構成内のコンマで区切られたプラグインのリストの最大長は 255 バイトです。次のように、この 2 つの制限を識別する 2 つの定義が、インクルード・ファイル sqlenv.h にあります。

```
#define SQL_PLUGIN_NAME_SZ     32     /* plug-in name */
#define SQL_SRVCON_GSSPLUGIN_LIST_SZ 255 /* GSS API plug-in list */
```

セキュリティー・プラグイン・ライブラリー・ファイルには、以下のファイル許可がなければなりません。

- インスタンス所有者によって所有される。
- システム上のすべてのユーザーが読み取れる。
- システム上のすべてのユーザーが実行できる。

セキュリティー・プラグインの 2 部構成ユーザー ID のサポート

Windows 上の DB2 データベース・マネージャーは、2 部構成ユーザー ID の使用と、2 部構成ユーザー ID から 2 部構成許可 ID へのマッピングをサポートしています。

例えば、ドメインとユーザー ID からなる Windows オペレーティング・システムの 2 部構成ユーザー ID (例えば MEDWAY¥pieter) について考えます。この例では、MEDWAY はドメインであり、pieter はユーザー名です。 DB2 データベース・シス

テムでは、この 2 部構成ユーザー ID を、1 部構成許可 ID と 2 部構成許可 ID のどちらにマップするかを指定することができます。

2 部構成ユーザー ID から 2 部構成許可 ID へのマッピングもサポートされていますが、デフォルト動作ではありません。デフォルトでは、 1 部構成ユーザー ID と 2 部構成ユーザー ID はどちらも 1 部構成許可 ID にマップされます。 2 部構成ユーザー ID から 2 部構成許可 ID へのマッピングもサポートされていますが、デフォルト動作ではありません。

2 部構成ユーザー ID から 1 部構成ユーザー ID へのデフォルト・マッピングを使用すれば、ユーザーは次のようにしてデータベースに接続することができます。

```
db2 connect to db user MEDWAY¥pieter using pw
```

この状況下でデフォルト動作を使用すると、ユーザー ID `MEDWAY¥pieter` は許可 ID `PIETER` に解決されます。 2 部構成ユーザー ID から 2 部構成許可 ID へのマッピングのサポートが有効になっている場合、許可 ID は `MEDWAY¥PIETER` となります。

2 部構成ユーザー ID から 2 部構成許可 ID へのマップをDB2 で有効にするために、以下の 2 セットの認証プラグインが DB2 に用意されています。

- 一方のセットは、1 部構成ユーザー ID を 1 部構成許可 ID にマップし、2 部構成ユーザー ID を 1 部構成許可 ID にマップするだけです。
- もう一方のセットは、1 部構成ユーザー ID または 2 部構成ユーザー ID の両方を、2 部構成許可 ID にマップします。

作業環境におけるユーザー名を、さまざまな場所で定義された複数のアカウント (ローカル・アカウント、ドメイン・アカウント、およびトラステッド・ドメイン・アカウントなど) にマップできる場合は、2 部構成許可 ID のマッピングを有効にするプラグインを指定することができます。

`PIETER` などの 1 部構成許可 ID と、ドメインとユーザー ID を結合した `MEDWAY¥PIETER`のような 2 部構成許可 ID は、機能的に個別の許可 ID であることに注意してください。このような許可 ID の一方に関連付けられている特権セットは、他方の許可 ID に関連付けられている特権セットとは完全に異なります。 1 部構成許可 ID と 2 部構成許可 ID を扱う際には注意が必要です。

次の表は、DB2 データベース・システムに用意されているプラグインの種類と、特定の認証インプリメンテーション用のプラグイン名を示しています。

表 30. DB2 セキュリティー・プラグイン

認証タイプ	1 部構成のユーザー ID のプラグインの名前	2 部構成のユーザー ID のプラグインの名前
ユーザー ID/パスワード (クライアント)	IBMOSauthclient	IBMOSauthclientTwoPart
ユーザー ID/パスワード (サーバー)	IBMOSauthserver	IBMOSauthserverTwoPart
Kerberos	IBMkrb5	IBMkrb5TwoPart

注: Windows 64 ビット・プラットフォームでは、ここにリストされているプラグイン名に "64" という文字が付加されます。

ユーザー ID/パスワード・プラグインまたは Kerberos プラグインを必要とする認証タイプを指定している場合は、上記の表の「1 部構成ユーザー ID のプラグインの名前」列にリストされているプラグインがデフォルトで使用されます。

2 部構成ユーザー ID を 2 部構成許可 ID にマップするには、2 部構成プラグイン (これはデフォルトのプラグインではありません) の使用を指定する必要があります。セキュリティー・プラグインは、セキュリティー関連のデータベース・マネージャー構成パラメーターを設定することによって、インスタンス・レベルで指定します。それは次のようにします。

2 部構成ユーザー ID を 2 部構成許可 ID にマップするサーバー認証の場合は、以下のように設定する必要があります。

- `srvcon_pw_plugin` を `IBMOSauthserverTwoPart` に設定
- `clnt_pw_plugin` を `IBMOSauthclientTwoPart` に設定

2 部構成ユーザー ID を 2 部構成許可 ID にマップするクライアント認証の場合は、以下のように設定する必要があります。

- `srvcon_pw_plugin` を `IBMOSauthserverTwoPart` に設定
- `clnt_pw_plugin` を `IBMOSauthclientTwoPart` に設定

2 部構成ユーザー ID を 2 部構成許可 ID にマップする Kerberos 認証の場合は、以下のように設定する必要があります。

- `srvcon_gssplugin_list` を `IBMOSkrb5TwoPart` に設定
- `clnt_krb_plugin` を `IBMkrb5TwoPart` に設定

セキュリティー・プラグイン・ライブラリーは、Microsoft Windows Security Account Manager 互換形式で指定された 2 パーツのユーザー ID を受け入れます。これは、例えば *domain¥user ID* の形式です。接続時の DB2 の認証プロセスおよび許可プロセスでは、ドメインとユーザー ID の両方の情報が使用されます。

新規データベースを作成する際は、既存データベース内の 1 部構成許可 ID と競合しないよう、2 部構成プラグインをインプリメントすることを検討するようお勧めします。 2 部構成許可 ID を使用する新規のデータベースは、1 部構成許可 ID を使用するデータベースとは別のインスタンス内で作成する必要があります。

セキュリティー・プラグイン API のバージョン管理

DB2 データベース・システムは、セキュリティー・プラグイン API のバージョン番号をサポートします。そのようなバージョン番号は、DB2 UDB、バージョン 8.2 では、1 で始まる整数になります。

DB2 がセキュリティー・プラグイン API に渡すバージョン番号は、DB2 がサポートできる最高のバージョン番号であり、構造のバージョン番号に対応します。プラグインは、それより高い API バージョンをサポートできる場合、DB2 が要求したバージョン用の関数ポインターを戻さなければなりません。プラグインがそれより低いバージョンの API しかサポートしない場合、そのプラグインは低いバージョン用の関数ポインターを指定する必要があります。いずれのケースでも、セキュリティー・プラグイン API は、サポートしている API のバージョン番号を、関数構造のバージョン・フィールドに入れて戻す必要があります。

DB2 の場合、セキュリティー・プラグインのバージョン番号は、必要な場合にのみ変化します (例えば、API のパラメーターが変更された場合など)。バージョン番号が DB2 のリリース番号とともに自動的に変わるわけではありません。

セキュリティー・プラグインの 32 ビットと 64 ビットに関する考慮事項

通常、32 ビット DB2 インスタンスは、32 ビット・セキュリティー・プラグインを使用し、64 ビット DB2 インスタンスは 64 ビット・セキュリティー・プラグインを使用します。しかし、64 ビット・インスタンス上では、DB2 は、32 ビット・プラグイン・ライブラリーを必要とする 32 ビット・アプリケーションをサポートします。

32 ビットと 64 ビットの両方のアプリケーションが実行できるデータベース・インスタンスは、ハイブリッド・インスタンスといいます。ハイブリッド・インスタンスがあり、32 ビット・アプリケーションを実行しようとしている場合は、必要な 32 ビット・セキュリティー・プラグインが、 32 ビット・プラグイン・ディレクトリー内に用意されていることを確認してください。 Linux および UNIX オペレーティング・システム (Linux on IPF を除く) 上の 64 ビット DB2 インスタンスでは、`security32`と `security64` のディレクトリーが現れます。 x64 または IPF 上での Windows における 64 ビット DB2 インスタンスの場合、32 ビットと 64 ビットの両方のセキュリティー・プラグインが同一のディレクトリー内にありますが、64 ビット・プラグイン名には、接尾部の 64 が付いています。

32 ビット・インスタンスを 64 ビット・インスタンスにアップグレードする予定の場合は、64 ビット用に再コンパイルされたセキュリティー・プラグインを取得する必要があります。

64 ビット・プラグイン・ライブラリーを提供しないベンダーからセキュリティー・プラグインを購入した場合、32 ビット・アプリケーションを実行する 64 ビット・スタブをインプリメントできます。この場合、セキュリティー・プラグインは、ライブラリーではなく外部プログラムになります。

セキュリティー・プラグインの問題判別

セキュリティー・プラグインの問題は、SQL エラー経由と管理通知ログ経由の 2 とおりの方法で報告されます。

以下は、セキュリティー・プラグインに関係した SQLCODE 値です。

- SQLCODE -1365 は、**db2start** または **db2stop** の間にプラグイン・エラーが発生すると戻されます。
- SQLCODE -1366 は、ローカル許可の問題がある場合に戻されます。
- SQLCODE -30082 は、すべての接続に関係したプラグイン・エラーで戻されます。

管理通知ログは、セキュリティー・プラグインのデバッグおよび管理のための良い情報源です。UNIX 上で管理通知ログを参照するには、`sqllib/db2dump/`*`instance name`*`.N.nfy` を調べます。Windows オペレーティング・システム上で管理通知ログを参照するには、「イベント ビューア」ツールを使用します。「イベント ビューア」ツールは、Windows オペレーティング・システムの「スタート」ボタンから

「設定」->「コントロール パネル」->「管理ツール」->「イベント ビューア」の順にナビゲートすると見つかります。以下は、セキュリティー・プラグインに関係した管理通知ログ値です。

- 13000 は、GSS-API セキュリティー・プラグイン API の呼び出しがエラーによって失敗し、オプションのエラー・メッセージが戻されたことを示します。

  ```
  SQLT_ADMIN_GSS_API_ERROR (13000)
  Plug-in "plug-in name" received error code "error code" from
  GSS API  "gss api name" with the error message "error message"
  ```

- 13001 は、DB2 セキュリティー・プラグイン API の呼び出しがエラーで失敗し、オプションのエラー・メッセージが戻されたことを示します。

  ```
  SQLT_ADMIN_PLUGIN_API_ERROR(13001)
  Plug-in "plug-in name" received error code "error code" from DB2
  security plug-in API "gss api name" with the error message
  "error message"
  ```

- 13002 は、DB2 がプラグインをアンロードできなかったことを示します。

  ```
  SQLT_ADMIN_PLUGIN_UNLOAD_ERROR (13002)
  Unable to unload plug-in "plug-in name". No further action required.
  ```

- 13003 は、無効なプリンシパル名を示します。

  ```
  SQLT_ADMIN_INVALID_PRIN_NAME (13003)
  The principal name "principal name" used for "plug-in name"
  is invalid. Fix the principal name.
  ```

- 13004 は、プラグイン名が無効なことを示します。パス区切り記号 (UNIX の場合は "/"、Windows の場合は "¥") をプラグイン名の一部として使用することはできません。

  ```
  SQLT_ADMIN_INVALID_PLGN_NAME (13004)
  The plug-in name "plug-in name" is invalid. Fix the plug-in name.
  ```

- 13005 は、セキュリティー・プラグインがロードできなかったことを示します。プラグインを正しいディレクトリーに入れ、該当するデータベース・マネージャー構成パラメーターを更新してください。

  ```
  SQLT_ADMIN_PLUGIN_LOAD_ERROR (13005)
  Unable to load plug-in "plug-in name". Verify the plug-in existence and
  directory where it is located is correct.
  ```

- 13006 は、セキュリティー・プラグインによって予期しないエラーが検出されたことを示します。すべての **db2support** 情報を収集し、可能であれば **db2trc** を取り込んでから、IBM サポートに問い合わせてください。

  ```
  SQLT_ADMIN_PLUGIN_UNEXP_ERROR (13006)
  Plug-in encountered unexpected error. Contact IBM Support for further assistance.
  ```

注: Windows 64 ビットのデータベース・サーバー上でセキュリティー・プラグインを使用しているときに、セキュリティー・プラグインのロード・エラーが表示された場合は、32 ビットおよび 64 ビットの場合の考慮事項とセキュリティー・プラグインの命名規則に関するトピックを参照してください。 64 ビット・プラグイン・ライブラリーには、ライブラリー名に 64 という接尾部が付いていなければなりませんが、セキュリティー・プラグインのデータベース・マネージャー構成パラメーターへの入力にはこの接尾部は含めません。

プラグインの使用可能化

グループ検索プラグインの展開

DB2 セキュリティー・システムのグループ検索動作をカスタマイズするには、独自のグループ検索プラグインを開発するか、または第三者からこれを購入できます。

始める前に

データベース管理システムに適したグループ検索プラグインを入手した後、それを展開することができます。

手順

- グループ検索プラグインをデータベース・サーバー上に展開するには、以下のステップを実行します。
  1. グループ検索プラグイン・ライブラリーをサーバーのグループ・プラグイン・ディレクトリーにコピーします。
  2. データベース・マネージャー構成パラメーター **group_plugin** をプラグインの名前で更新します。
- グループ検索プラグインをデータベース・クライアント上に展開するには、以下のステップを実行します。
  1. グループ検索プラグイン・ライブラリーをクライアントのグループ・プラグイン・ディレクトリーにコピーします。
  2. データベース・クライアント上で、データベース・マネージャー構成パラメーター **group_plugin** をプラグインの名前で更新します。

ユーザー ID/パスワード・プラグインのデプロイ

DB2 セキュリティー・システムのユーザー ID/パスワード認証動作をカスタマイズするには、独自のユーザー ID/パスワード認証プラグインを開発するか、または第三者からこれを購入できます。

始める前に

すべてのユーザー ID/パスワード・ベース認証プラグインは、意図しているそれらプラグインの使用法に応じて、クライアントのプラグイン・ディレクトリーか、サーバーのプラグイン・ディレクトリーのいずれかに置く必要があります。プラグインがクライアントのプラグイン・ディレクトリーに置かれる場合、それはローカル許可検査のためと、クライアントがサーバーと接続しようとするときのクライアントの妥当性検査のために使用されます。プラグインがサーバーのプラグイン・ディレクトリーに置かれる場合、それはサーバーへの着信接続の処理のためと、 USER または GROUP キーワードが指定されずに GRANT ステートメントが発行された場合の許可 ID の存在とその有効性の検査のために使用されます。ほとんどの場合、ユーザー ID/パスワード認証で必要となるのは、サーバー・サイドのプラグインのみです。一般にそれほど役に立ちませんが、クライアントのユーザー ID/パスワード・プラグインのみを使用することも可能です。非常に稀なことですが、クライアントとサーバーの両方に、一致するユーザー ID/パスワード・プラグインが必要になることもあります。

注: *既存の プラグインの新しい* バージョンをデプロイする場合は、DB2 サーバー、またはプラグインを使用するすべてのアプリケーションをまず停止する必要があります。新しいバージョンを (同じ名前で) 上書きコピーするときに、プラグインを使用するプロセスがまだ実行中になっていると、未定義の動作 (トラップなど) が発生します。この制約事項は、プラグインを初めてデプロイする場合や、プラグインが使用中になっていない場合には適用されません。
データベース管理システムに適したユーザー ID/パスワード認証プラグインを入手した後、それらをデプロイすることができます。

手順

- ユーザー ID/パスワード認証プラグインをデータベース・サーバー上にデプロイするには、データベース・サーバー上で以下のステップを実行します。
  1. ユーザー ID/パスワード認証プラグイン・ライブラリーをサーバーのプラグイン・ディレクトリーにコピーします。
  2. データベース・マネージャー構成パラメーター **srvcon_pw_plugin** をサーバー・プラグインの名前で更新します。 このプラグインは、サーバーが接続 (CONNECT) 要求およびアタッチ (**ATTACH**) 要求を処理する際に使用します。
  3. 次のいずれかを行ってください。
     - データベース・マネージャー構成パラメーター **srvcon_auth** に、CLIENT、SERVER、SERVER_ENCRYPT、DATA_ENCRYPT、または DATA_ENCRYPT_CMP の認証タイプを設定します。あるいは、
     - データベース・マネージャー構成パラメーター **srvcon_auth** を NOT_SPECIFIED に設定し、**authentication** を CLIENT、SERVER、SERVER_ENCRYPT、DATA_ENCRYPT、または DATA_ENCRYPT_CMP のいずれかの認証タイプに設定します。
- ユーザー ID/パスワード認証プラグインをデータベース・クライアント上にデプロイするには、各クライアント上で以下のステップを実行します。
  1. ユーザー ID/パスワード認証プラグイン・ライブラリーをクライアントのプラグイン・ディレクトリーにコピーします。
  2. データベース・マネージャー構成パラメーター **clnt_pw_plugin** をクライアント・プラグインの名前で更新します。このプラグインは、認証がどこで行われているかに関係なく、つまり、データベース構成パラメーター **authentication** が CLIENT に設定されているとき以外にもロードされ、呼び出されます。
- ユーザー ID/パスワード認証プラグインを使用したクライアント、サーバー、またはゲートウェイでのローカル許可に関しては、各クライアント、サーバー、またはゲートウェイ上で以下のステップを実行します。
  1. ユーザー ID/パスワード認証プラグイン・ライブラリーをクライアント、サーバー、またはゲートウェイ上のクライアント・プラグイン・ディレクトリーにコピーします。
  2. データベース・マネージャー構成パラメーター **clnt_pw_plugin** をプラグインの名前で更新します。
  3. **authentication** データベース・マネージャー構成パラメーターに、 CLIENT、SERVER、SERVER_ENCRYPT、DATA_ENCRYPT、または DATA_ENCRYPT_CMP を設定します。

GSS-API プラグインのデプロイ

DB2 セキュリティー・システムの認証動作をカスタマイズするには、 GSS-API を使用して独自の認証プラグインを開発するか、または第三者からこれを購入できます。

始める前に

Kerberos 以外のプラグイン・タイプの場合は、クライアントとサーバー上に、同じプラグイン・タイプだけでなく、一致するプラグイン名がなければなりません。クライアントとサーバー上のプラグインは、同一のベンダーのものである必要はありませんが、これらは互換性のある GSS-API トークンを生成して使用する必要があります。 Kerberos プラグインは標準化されているので、クライアントとサーバー上にはどのような組み合わせの Kerberos プラグインをデプロイしても構いません。しかし、それほど標準化されていない *x.509* 証明書などの種々の GSS-API メカニズムのインプリメンテーション間には、DB2 データベース・システムとの部分的な互換性しかない可能性があります。すべての GSS-API 認証プラグインは、意図しているそれらプラグインの使用法に応じて、クライアントのプラグイン・ディレクトリーか、サーバーのプラグイン・ディレクトリーのいずれかに置く必要があります。プラグインがクライアントのプラグイン・ディレクトリーに置かれる場合、それはローカル許可検査のためと、クライアントがサーバーと接続しようとするときに使用されます。プラグインがサーバーのプラグイン・ディレクトリーに置かれる場合、それはサーバーへの着信接続の処理のためと、 USER または GROUP キーワードが指定されずに GRANT ステートメントが発行された場合の許可 ID の存在とその有効性の検査のために使用されます。

注: *既存の プラグインの新しい* バージョンをデプロイする場合は、DB2 サーバー、またはプラグインを使用するすべてのアプリケーションをまず停止する必要があります。新しいバージョンを (同じ名前で) 上書きコピーするときに、プラグインを使用するプロセスがまだ実行中になっていると、未定義の動作 (トラップなど) が発生します。この制約事項は、プラグインを初めてデプロイする場合や、プラグインが使用中になっていない場合には適用されません。

データベース管理システムに適した GSS-API 認証プラグインを入手した後、それらをデプロイすることができます。

手順

- GSS-API 認証プラグインをデータベース・サーバー上にデプロイするには、サーバー上で以下のステップを実行します。
  1. GSS-API 認証プラグイン・ライブラリーをサーバー・プラグイン・ディレクトリーにコピーします。 このディレクトリーには、多数の GSS-API プラグインをコピーすることができます。
  2. データベース・マネージャー構成パラメーター **srvcon_gssplugin_list** を、GSS-API プラグイン・ディレクトリーにインストールされたプラグインの名前の順番のコンマ区切りのリストで更新します。
  3. 次のいずれかを行ってください。
     – データベース・マネージャー構成パラメーター **srvcon_auth** を GSSPLUGIN または GSS_SERVER_ENCRYPT に設定することは、サーバーが GSSAPI PLUGIN 認証方式を使用できるようにする一つの方法です。あるいは、

– データベース・マネージャー構成パラメーター **srvcon_auth** を NOT_SPECIFIED に設定し、**authentication** を GSSPLUGIN または GSS_SERVER_ENCRYPT に設定することは、サーバーが GSSAPI PLUGIN 認証方式を使用できるようにする一つの方法です。

- GSS-API 認証プラグインをデータベース・クライアント上にデプロイするには、各クライアント上で以下のステップを実行します。
  1. GSS-API 認証プラグイン・ライブラリーをクライアントのプラグイン・ディレクトリーにコピーします。 このディレクトリーには、多数の GSS-API プラグインをコピーすることができます。クライアントは、クライアント上で使用できる、サーバーのプラグインのリストに含まれている最初の GSS-API を選出することによって、接続 (CONNECT) またはアタッチ (ATTACH) 操作中の認証用の GSS-API プラグインを選択します。
  2. オプション: クライアントがアクセスするデータベースをカタログし、クライアントが GSS-API 認証プラグインのみを認証メカニズムとして受け入れることを示します。 例えば、以下のようにします。

     ```
     CATALOG DB testdb AT NODE testnode AUTHENTICATION GSSPLUGIN
     ```

- GSS-API 認証プラグインを使用したクライアント、サーバー、またはゲートウェイでのローカル許可に関しては、以下のステップを実行します。
  1. GSS-API 認証プラグイン・ライブラリーをクライアント、サーバー、またはゲートウェイ上のクライアント・プラグイン・ディレクトリーにコピーします。
  2. データベース・マネージャー構成パラメーター **local_gssplugin** をプラグインの名前で更新します。
  3. **authentication** データベース・マネージャー構成パラメーターに、GSSPLUGIN、または GSS_SERVER_ENCRYPT を設定します。

Kerberos プラグインのデプロイ

DB2 セキュリティー・システムの Kerberos 認証の動作をカスタマイズするには、独自の Kerberos 認証プラグインを作成するか、またはサード・パーティーから購入することができます。

始める前に

既存のプラグインの新しいバージョンをデプロイする場合は、そのプラグインを使用している DB2 サーバーとすべてのアプリケーションを停止する必要があります。プラグインの新しいバージョンを (同じ名前で) デプロイする際に、そのプラグインがプロセスで使用中の場合は、未定義の動作 (トラップなど) が発生します。

このタスクについて

Kerberos 認証プラグインは、データベース・サーバーまたはデータベース・クライアントにデプロイできます。

手順

- Kerberos 認証プラグインをデータベース・サーバー上にデプロイするには、サーバー上で以下のステップを実行します。

1. Kerberos 認証プラグイン・ライブラリーをサーバー・プラグイン・ディレクトリーにコピーします。
2. **srvcon_gssplugin_list** データベース・マネージャー構成パラメーター (順序付けされたコンマ区切りのリスト) の設定を更新して、Kerberos サーバー・プラグイン名を含めます。このリストの中の 1 つのプラグインのみを Kerberos プラグインにすることができます。 Kerberos プラグインがリストに含まれていない場合、エラーが戻されます。複数の Kerberos プラグインがリストに含まれている場合、エラーが戻されます。この構成パラメーターの値がブランクで、**authentication** 構成パラメーターが KERBEROS または KRB_SVR_ENCRYPT に設定されている場合は、デフォルトの DB2 Kerberos プラグインである IBMkrb5 が使用されます。
3. 必要に応じ、**srvcon_auth** データベース・マネージャー構成パラメーターの値を設定します。 Kerberos プラグインをデプロイする場合、**srvcon_auth** データベース・マネージャー構成パラメーターの許容値は、以下のとおりです。
   - KERBEROS
   - KRB_SERVER_ENCRYPT
   - GSSPLUGIN
   - GSS_SERVER_ENCRYPT
   - ブランク。ただし、**authentication** 構成パラメーターが、このリスト内の上記に示す値のいずれかに設定されている場合のみ。

- Kerberos 認証プラグインをデータベース・クライアントにデプロイするには、クライアント上で以下のステップを実行します。
  1. Kerberos 認証プラグイン・ライブラリーをクライアントのプラグイン・ディレクトリーにコピーします。
  2. **clnt_krb_plugin** データベース・マネージャー構成パラメーターを、Kerberos プラグインの名前に設定します。**clnt_krb_plugin** 構成パラメーターの値がブランクの場合、クライアントは Kerberos 認証を使用できません。 Windowsでは、デフォルト値は IBMkrb5 です。これを変更する必要があるのは、カスタマイズされた Kerberos プラグインの場合のみです。UNIX では、デフォルト値がブランクであるため、この値を設定する必要があります。Kerberos 認証プラグインを使用したクライアント、サーバー、またはゲートウェイでのローカル許可に関しては、以下のステップを実行します。
     a. Kerberos 認証プラグイン・ライブラリーをクライアント、サーバー、またはゲートウェイ上のクライアント・プラグイン・ディレクトリーにコピーします。
     b. **clnt_krb_plugin** データベース・マネージャー構成パラメーターを、プラグインの名前に設定します。
     c. **authentication** データベース・マネージャー構成パラメーターに、KERBEROS または KRB_SERVER_ENCRYPT を設定します。
  3. オプション: クライアントがアクセスするデータベースをカタログし、クライアントが Kerberos 認証プラグインのみを使用することを示します。以下の例では、testdb データベースをカタログします。

```
CATALOG DB testdb AT NODE testnode AUTHENTICATION KERBEROS
        TARGET PRINCIPAL service/host@REALM
```

LDAP ベースの認証とグループ検索サポート

DB2 データベース・マネージャーと DB2 Connect は、LDAP セキュリティー・プラグイン・モジュールを使用することによって、さらに透過的 LDAP を使用することにより、LDAP ベースの認証とグループ検索機能をサポートしています。

AIX オペレーティング・システムにおいて、LDAP ベースの認証サポートが拡張されました。DB2 V9.7 フィックスパック 1 から、透過的 LDAP のサポートが、DB2 製品のサポート対象と同じバージョン・レベルの Linux、HP-UX および Solaris オペレーティング・システムにまで拡張されました。 LDAP により、透過的 LDAP 認証を使用したユーザー認証とグループ・メンバーシップの集中管理が可能になります。 DB2 インスタンスがオペレーティング・システムを介してユーザーを認証し、ユーザーのグループを取得するように構成することができます。そのようにすると、オペレーティング・システムが LDAP サーバーを介して認証を実行します。透過的 LDAP 認証を使用可能にするには、**DB2AUTH** 各種レジストリー変数を OSAUTHDB に設定します。サポートされているオペレーティング・システムは次のとおりです。

- AIX
- HP-UX
- Linux
- Solaris

LDAP ベースの認証をインプリメントするための別のオプションは、LDAP セキュリティー・プラグインを使用するというものです。DB2 データベース・マネージャーは、LDAP セキュリティー・プラグイン・モジュールを使用して、LDAP ディレクトリーに定義されているユーザーを認証します。その場合は、ユーザーとグループを、DB2 製品のサポート対象と同じバージョン・レベルのオペレーティング・システムに対して定義する必要がありません。サポートされているオペレーティング・システムは次のとおりです。

- AIX
- Itanium ベースの HP Integrity Series システム (IA-64) 上の HP-UX
- IA32、x64、または zSeries® ハードウェア上の Linux
- Solaris
- Windows

セキュリティー・プラグイン・モジュールの使用に対応した LDAP サーバーは、以下のとおりです。

- IBM Lotus® Domino® LDAP Server バージョン 8.0 以降
- IBM Tivoli® Directory Server (ITDS) バージョン 6.2 以降 (GSKit 7.0.4.20 以降同梱)
- Microsoft Active Directory (MSAD) バージョン 2008 以降
- Novell eDirectory バージョン 8.8 以降
- OpenLDAP サーバー、バージョン 2.4 以降
- Sun Java System Directory Server Enterprise Edition バージョン 5.2 FP4 以降
- z/OS Integrated Security Services LDAP Server バージョン V1R6 以降

注: LDAP プラグイン・モジュールを使用する場合は、データベースに関連付けられているすべてのユーザーを LDAP サーバーで定義する必要があります。その中には、DB2 インスタンス所有者 ID と fenced ユーザーの両方が含まれます。(これらのユーザーは通常、オペレーティング・システムで定義されていますが、LDAP でも定義しなければなりません。)さらに、LDAP グループ・プラグイン・モジュールを使用する場合は、許可に必要なグループを LDAP サーバーで定義することも必要です。その中には、データベース・マネージャー構成で定義されている SYSADM、SYSMAINT、SYSCTRL、SYSMON の各グループが含まれます。

DB2 セキュリティー・プラグイン・モジュールは、後で取り上げるサーバー・サイドの認証、クライアント・サイドの認証、グループ検索のために使用できます。それぞれの環境に応じて、それらのタイプのうち、1 つか 2 つまたはすべてのプラグインを使用する必要があります。

DB2 セキュリティー・プラグイン・モジュールを使用するには、以下の手順を実行します。

1. 必要なのは、サーバー・プラグイン・モジュールか、クライアント・プラグイン・モジュールか、グループ・プラグイン・モジュールか、それらの組み合わせかを決定します。
2. プラグイン・モジュールを構成するために、IBM LDAP セキュリティー構成ファイル (デフォルトの名前は `IBMLDAPSecurity.ini`) の値を設定します。適切な値については、LDAP 管理者に問い合わせる必要があります。
3. プラグイン・モジュールを使用可能にします。
4. さまざまな LDAP ユーザー ID で接続をテストします。

サーバー認証プラグイン

サーバー認証プラグイン・モジュールは、CONNECT ステートメントと ATTACH ステートメントでクライアントから渡されるユーザー ID とパスワードのサーバー妥当性検査を実行します。必要に応じて、LDAP ユーザー ID を DB2 許可 ID に対応付けることも可能です。通常、サーバー・プラグイン・モジュールが必要になるのは、LDAP ユーザー ID とパスワードを使用して、DB2 データベース・マネージャーに対してユーザーを認証する場合です。

クライアント認証プラグイン

クライアント認証プラグイン・モジュールを使用するのは、クライアント・システムでユーザー ID とパスワードの確認を実行する場合、つまり、DB2 サーバーの SRVCON_AUTH 設定または AUTHENTICATION 設定が CLIENT になっている場合です。クライアントは、CONNECT ステートメントまたは ATTACH ステートメントに指定されているユーザー ID とパスワードを確認してから、そのユーザー ID を DB2 サーバーに送信します。ただし、クライアント認証は、セキュリティーの確保が難しいので、通常はお勧めできません。

クライアント認証プラグイン・モジュールは、データベース・サーバーのローカル・オペレーティング・システム・ユーザー ID と、そのユーザーに関連付けられている DB2 許可 ID が異なる場合にも必要になることがあります。クライアント・サイド・プラグインを使用すれば、データベース・サーバーで `db2start` などのローカル・コマンドの許可検査を実行する前に、ローカル・オペレーティング・

システム・ユーザー ID と DB2 許可 ID を対応付けることが可能になります。

グループ検索プラグイン

グループ検索プラグイン・モジュールを使用すれば、LDAP サーバーから特定ユーザーのグループ・メンバーシップ情報を取得できます。LDAP を使用してグループ定義を格納する場合は、このモジュールが必要です。最も一般的なのは、以下のようなシナリオです。

- すべてのユーザーとグループが LDAP サーバーで定義されている場合
- データベース・サーバーでローカルに定義されているユーザーが、LDAP サーバーでも同じユーザー ID で定義されている場合 (インスタンス所有者と fenced ユーザーを含む)
- DB2 サーバーでパスワードを確認する場合 (つまり、DBM 構成ファイルで、AUTHENTICATION または SRVCON_AUTH の値が、SERVER、SERVER_ENCRYPT、DATA_ENCRYPT のいずれかに設定されている場合)

通常は、サーバーにサーバー認証プラグイン・モジュールとグループ検索プラグイン・モジュールをインストールするだけで十分です。DB2 クライアントには基本的に、LDAP プラグイン・モジュールをインストールする必要はありません。

LDAP グループ検索プラグイン・モジュールだけをインストールして、他の形式の認証プラグイン (Kerberos など) と組み合わせて使用することも可能です。その場合は、LDAP グループ検索プラグイン・モジュールに、ユーザーに関連付けられている DB2 許可 ID が渡されます。そのプラグイン・モジュールは、LDAP ディレクトリーで、AUTHID_ATTRIBUTE が一致するユーザーを検索し、そのユーザー・オブジェクトに関連付けられているグループを取得します。

認証およびグループ検索のための透過的 LDAP の構成 (AIX)

DB2 V9.7 では、透過的 LDAP に基づく認証およびグループ検索が AIX オペレーティング・システムでサポートされます。このサポートを有効にするには、いくつかの構成手順を実行する必要があります。

始める前に

以下の手順では、LDAP サーバーが RFC 2307 に準拠すること、およびユーザーとグループの情報を保管するように構成されていることを想定します。

手順

1. LDAP 用の AIX クライアント・システムを構成するには、以下の手順を実行します。
   a. root 権限を持つユーザーとしてログインします。
   b. LDAP クライアント・ファイル・セットが AIX システムにインストール済みであることを確認します。 AIX では、ITDS V5.2 (AIX V5.3 に付属)、ITDS V6.1 (AIX V6.1 に付属)、および ITDS V6.2 (AIX 拡張パックに付属) の 3 つのバージョンの LDAP クライアントがすべて機能します。以下は、AIX V5.3 システムにインストールされた ITDS V5.2 ファイル・セットを示しています。

```
$ lslpp -l "ldap*"
  Fileset                      Level  State      Description
  ----------------------------------------------------------------------------
Path: /usr/lib/objrepos
  ldap.client.adt            5.2.0.0  COMMITTED  Directory Client SDK
  ldap.client.rte            5.2.0.0  COMMITTED  Directory Client Runtime (No
                                                 SSL)
  ldap.html.en_US.config     5.2.0.0  COMMITTED  Directory Install/Config
                                                 Gd-U.S. English
  ldap.html.en_US.man        5.2.0.0  COMMITTED  Directory Man Pages - U.S.
                                                 English
  ldap.msg.en_US             5.2.0.0  COMMITTED  Directory Messages - U.S.
                                                 English

Path: /etc/objrepos
  ldap.client.rte            5.2.0.0  COMMITTED  Directory Client Runtime (No
                                                 SSL)
```

c. **-c** オプションと共に **mksecldap** コマンドを使用して、クライアントを構成します。 **mksecldap** コマンドの詳細と、これを使用してクライアントを構成する方法については、http://publib.boulder.ibm.com/infocenter/pseries/v5r3/index.jsp?topic=/com.ibm.aix.security/doc/security/setup_ldap_sec_info_server.htmを参照してください。

d. /etc/security/user ファイルの中のデフォルト・スタンザを更新します。

   LDAP が正しく構成されて LDAP ディレクトリーにユーザーが入っていることを確認した後、LDAP を使用するデフォルト・ユーザーを設定する必要があります。こうすることで、制限のない LDAP ディレクトリー内の任意のユーザーを使って AIX クライアントにログインできるようになります。

   /etc/security/user ファイル内の **SYSTEM** および **REGISTRY** 属性は、認証方式およびユーザー管理に使われるデータベースを指定するために使用されます。 LDAP 認証とユーザー管理を有効にするには、デフォルト・スタンザにある **SYSTEM** および **REGISTRY** 属性を LDAP に設定します。以下に例を示します。

   ```
   chsec -f /etc/security/user -s default -a "SYSTEM=LDAP or files"
   chsec -f /etc/security/user -s default -a "REGISTRY=LDAP"
   ```

   DB2 は、次の **SYSTEM** 属性をサポートします。

   - LDAP
   - ファイル

   DB2 は、次の **REGISTRY** 属性をサポートします。

   - LDAP
   - KRB5LDAP
   - KRB5ALDAP
   - ファイル
   - KRB5files
   - KRB5Afiles

   他の **SYSTEM** 属性や **REGISTRY** 属性を使用する構成は作動する可能性がありますが、サポートされていません。

   スタンザの **SYSTEM** および **REGISTRY** 属性についての詳細は、http://publib.boulder.ibm.com/infocenter/aix/v6r1/index.jsp?topic=/com.ibm.aix.files/doc/aixfiles/user.htm?を参照してください。

詳しくは、「Integrating AIX into Heterogeneous LDAP Environments」というタイトルのレッドブック (http://www.redbooks.ibm.com/abstracts/sg247165.html) を参照してください。

2. DB2 インスタンス上で透過的 LDAP 認証を構成するには、次のようにします。
   a. **DB2AUTH** 各種レジストリー変数を OSAUTHDB に設定します。SYSADM 権限を持つユーザーとして **db2set DB2AUTH=OSAUTHDB** を実行します。
   b. **UPDATE DBM CFG** コマンドを使用して、データベース・サーバー・インスタンス上の認証を以下のいずれか 1 つに設定します。
      - SERVER
      - SERVER_ENCRYPT
      - DATA_ENCRYPT
   c. デフォルトのクライアント・ユーザー ID/パスワード・プラグイン (CLNT_PW_PLUGIN)、サーバー・ユーザー ID/パスワード・プラグイン (SRVCON_PW_PLUGIN)、およびグループ・プラグイン (GROUP_PLUGIN) を使用していることを確認します。
   d. DB2 インスタンスを再始動します。

さまざまな認証方式を使用する際の考慮事項

AIX での透過的 LDAP に基づく認証およびグループ・ルックアップのサポートでは、Kerberos 認証のサポートが拡張されました。

AIX で透過的 LDAP と共に Kerberos 認証を使用するために、追加の処理が行われました。さまざまな場所にあるアカウントを管理して Kerberos などのさまざまな認証方式を使用する必要がある場合には、以下のものを /usr/lib/security/methods.cfg および /etc/security/users に含める必要があります。

/usr/lib/security/methods.cfg では、ファイル、LDAP および Kerberos 認証用に以下のものが必要です。

注: KRB5A は、Microsoft Active Directory を Keberos 鍵配布センター (KDC) として使用するためのものです。

LDAP 用:

```
program = /usr/lib/security/LDAP
program_64 =/usr/lib/security/LDAP64
```

KRB5A 用:

```
program = /usr/lib/security/KRB5A
program_64 = /usr/lib/security/KRB5A_64
options = tgt_verify=no,authonly,is_kadmind_compat=no
```

KRB5 用:

```
program = /usr/lib/security/KRB5
program_64 = /usr/lib/security/KRB5_64
options = kadmind=no
```

KRB5Afiles 用:

```
options = db=BUILTIN,auth=KRB5A
```

KRB5files 用:

```
options = db=BUILTIN,auth=KRB5
```

KRB5ALDAP 用:

```
options = db=LDAP,auth=KRB5A
```

KRB5LDAP 用:

```
options = db=LDAP,auth=KRB5
```

例

以下の例は、異なる方法で管理される 4 つのアカウントを示しています。それぞれ別個の認証方式が使用されます。

frank のアカウントがファイルに保管され、ファイルを使って認証される場合には、/etc/security/users の中で frank のスタンザは次のようになります。

```
frank:
        SYSTEM = files
        registry = files
```

karen のアカウントがファイルに保管され、Kerberos を使って認証される場合には、/etc/security/users の中で karen のスタンザは次のようになります。

```
karen:
        SYSTEM = KRB5files
        registry = KRB5files
```

luke のアカウントが LDAP に保管され、Kerberos を使って認証される場合には、/etc/security/users の中で luke のスタンザは次のようになります。

```
luke:
        SYSTEM = KRB5LDAP
        registry = KRB5LDAP
```

lucy のアカウントが LDAP に保管され、LDAP を使って認証される場合には、/etc/security/users の中で lucy のスタンザは次のようになります。

```
lucy:
        SYSTEM = LDAP
        registry = LDAP
```

LDAP にユーザーが定義されているかどうか判別するには、以下のコマンドを使ってユーザーを照会できます。

```
$ lsuser -R LDAP lucy
lucy id=1234 pgrp=staff groups=staff home=/home/lucy shell=/bin/ksh registry=LDAP
```

認証およびグループ検索のための透過的 LDAP の構成 (Linux)

DB2 V9.7 フィックスパック 1 から、Linux オペレーティング・システムで、DB2 データベース・サーバーが LDAP ベースの認証を透過的に使用できるようにするには、 Pluggable Authentication Module (PAM) を使用します。ユーザーおよびグループの情報を格納するように、事前に LDAP サーバーを構成しておく必要があります。

始める前に

DB2 データベースで、透過的 LDAP サポートを使用可能にするには、以下の作業を完了させます。

1. PAM を使用してユーザーを認証するように、オペレーティング・システムを構成する
2. DB2 インスタンスを構成する

以下の手順では、LDAP サーバーが RFC 2307 に準拠すると想定しています。

手順

1. オペレーティング・システムを LDAP および PAM 用に構成するには、以下の手順を実行します。
   a. root 権限を持つユーザーとしてログインします。
   b. nss_ldap および pam_ldap パッケージがインストールされていることを確認します。これら 2 つのパッケージは、/lib(64) または /usr/lib(64) ディレクトリー内に、libnss_ldap.so および libpam_ldap.so として存在します。
   c. オペレーティング・システムが LDAP サーバーにバインドできるように /etc/ldap.conf ファイルを変更することによって、オペレーティング・システムを LDAP クライアント・マシンとしてセットアップします。 以下に、/etc/ldap.conf ファイルのサンプルを示します。

```
host <host>             # Address of ldap server
base <base>             # The DN of the search base.
rootbinddn <binddn>     # The bind DN to bind to LDAP
ldap_version 3          # LDAP version
pam_login_attribute uid  # user ID attribute for pam user lookups
nss_base_group <group>   # nsswitch configuration pertaining to group
                # search lookup
```

   d. /etc/ldap.secret ファイルにパスワードを設定します。 root ユーザーのみ、このファイルへの読み取りまたは書き込みができるようにします。
   e. /etc/pam.d/db2 にある PAM 構成ファイルを変更または作成します。このファイルへの読み取りおよび書き込みは、root ユーザーのみできるようにします。使用するオペレーティング・システムのバージョンに応じて、構成ファイルを変更しなければならない場合があります。以下に、SUSE Linux Enterprise Server 10 用のサンプル構成ファイルを示します。

```
auth     sufficient pam_unix2.so
auth     required   pam_ldap.so     use_first_pass
account  sufficient pam_unix2.so
account  required   pam_ldap.so
password required   pam_pwcheck.so
password sufficient pam_unix2.so    use_authtok use_first_pass
password required   pam_ldap.so     use_first_pass
session  required   pam_unix2.so
```

   Red Hat Enterprise Linux 5 の場合、以下のように構成ファイルを変更します。

```
#%PAM-1.0

auth    required    pam_env.so
auth    sufficient  pam_unix.so likeauth nullok
auth    sufficient  pam_ldap.so use_first_pass
auth    required    pam_deny.so

account  required   pam_unix.so
```

```
account  sufficient pam_succeed_if.so uid < 100 quiet
account  sufficient pam_ldap.so
account  required   pam_permit.so

password requisite  pam_cracklib.so retry=3 dcredit=-1 ucredit=-1
password sufficient pam_unix.so nullok use_authtok md5 shadowremember=3
password sufficient pam_ldap.so  use_first_pass
password required   pam_deny.so

session  required   pam_limits.so
session  required   pam_unix.so
```

DB2 は、pam_ldap.so、pam_unix.so、および pam_unix2.so を使用する PAM 構成をサポートします。他の PAM モジュールを使用する構成は作動する可能性がありますが、サポートされていません。

f. LDAP を介してグループ検索を実行するように Linux システムをセットアップします。 /etc/nsswitch.conf ファイル内の **group** および **passwd** の項目を見つけ、検索方法として ldap が記入されていることを確認します。 以下は、**group** および **passwd** 項目の例です。

```
group:          files ldap
passwd:         files ldap
```

2. 透過的 LDAP 認証を使用するように DB2 インスタンスを構成するには、以下の手順を実行します。

   a. **DB2AUTH** 各種レジストリー変数を OSAUTHDB に設定します。 SYSADM 権限を持つユーザーで、以下のコマンドを実行します。

   ```
   db2set DB2AUTH=OSAUTHDB
   ```

   b. サーバー認証を、以下の任意のタイプに設定します。

   - SERVER
   - SERVER_ENCRYPT
   - DATA_ENCRYPT

   c. デフォルトのクライアント・ユーザー ID/パスワード・プラグイン (CLNT_PW_PLUGIN)、サーバー・ユーザー ID/パスワード・プラグイン (SRVCON_PW_PLUGIN)、およびグループ・プラグイン (GROUP_PLUGIN) を使用していることを確認します。

   d. DB2 インスタンスを再始動します。

認証およびグループ検索のための透過的 LDAP の構成 (HP-UX)

DB2 V9.7 フィックスパック 1 から、HP-UX オペレーティング・システムで、DB2 データベース・サーバーが LDAP ベースの認証を透過的に使用できるようにするには、 Pluggable Authentication Module (PAM) を使用します。ユーザーおよびグループの情報を格納するように、事前に LDAP サーバーを構成しておく必要があります。

始める前に

以下の手順では、LDAP サーバーが RFC 2307 に準拠すると想定しています。

手順

1. IBM Tivoli Directory Server (ITDS) バージョン 6.1 を使用している場合は、HP-UX システムが LDAP サーバーに接続する前に、LDAP サーバーをセットアップします。 HP-UX オペレーティング・システムに LDAP サーバーを構成するには、以下の手順を実行します。
   a. root 権限を持つユーザーとして LDAP サーバーにログインします。
   b. **idsldapadd** コマンドを以下のように実行します。

```
idsldapadd -D <root> -w <password> -h <hostname> -p <port> -c -i duaconfigschema.ldif

説明
<root> - LDAP にバインドするバインド DN
<password> - バインド DN のパスワード
<hostname> - LDAP サーバーのホスト名
<port> - LDAP サーバーを実行しているポートデフォルトは 389 です。
<schema.ldif> - DUAConfigProfle スキーマを含む LDIF ファイル
```

   Netscape または Red Hat Directory Server のどちらかを使用している場合、duaconfigschema.ldif にリストされているオブジェクト・クラスは、LDAP-UX セットアップ・プログラムを使用して自動的に LDAP サーバーに追加されます。一方、ITDS を使用している場合、 HP-UX クライアント 上で LDAP-UX セットアップ・プログラムを実行する前に、オブジェクト・クラスを手動で追加する必要があります。
2. オペレーティング・システムを LDAP および PAM 用に構成するには、以下の手順を実行します。
   a. root 権限を持つユーザーとしてログインします。
   b. LDAP-UX Client Services をインストールし、 LDAP-UX セットアップ・プログラムを実行します。 以下の画面が表示されます。

```
[ctrl-B]=Go Back screen 2
Hewlett-Packard Company
LDAP-UX Client Services Setup Program
--------------------------------------------------------------------------
Select which Directory Server you want to connect to:
1. Netscape or Red Hat Directory
2. Windows 2000/2003/2003 R2 Active Directory
To accept the default shown in brackets, press the Return key.
Directory Server: [1]:
```

   Netscape または Red Hat Directory Server に接続する場合と同様に、オプション 1 を選択し、以下の指示に従います。

   LDAP-UX のインストールについて詳しくは、「*LDAP-UX Client Services B.04.15 Administrator's Guide*」を参照してください。
   c. /etc/pam.conf にある PAM 構成ファイルを編集します。以下のテキストをファイルに追加します。

```
db2 auth required       libpam_hpsec.so.1
db2 auth sufficient                 libpam_unix.so.1
db2 auth required       libpam_ldap.so.1 use_first_pass
```

   上記の構成では、初めにローカル・ファイル・システムで、ユーザー ID およびパスワードを検索します。ユーザーが見つからなかった場合、またはローカル・ファイル・システムでの認証に失敗した場合にのみ、 LDAP 検索が実行されます。

DB2 は、libpam_ldap.so と libpam_unix.so を使用する PAM 構成をサポートします。他の PAM モジュールを使用する構成は作動する可能性がありますが、サポートされていません。

d. LDAP を介してグループ検索を実行するように HP-UX システムをセットアップします。 /etc/nsswitch.conf ファイル内の **group** および **passwd** の項目を見つけ、検索方法として ldap が記入されていることを確認します。 以下は、**group** および **passwd** 項目の例です。

```
group:          files ldap
passwd:         files ldap
```

3. 透過的 LDAP 認証を使用するように DB2 インスタンスを構成するには、以下の手順を実行します。

a. **DB2AUTH** 各種レジストリー変数を OSAUTHDB に設定します。 SYSADM 権限を持つユーザーで、以下のコマンドを実行します。

```
db2set DB2AUTH=OSAUTHDB
```

b. UPDATE DBM CFG コマンドを使用して、データベース・サーバー・インスタンス上の認証を以下のいずれか 1 つに設定します。

- SERVER
- SERVER_ENCRYPT
- DATA_ENCRYPT
- CLIENT

c. Client Userid-Password Plugin (clnt_pw_plugin)、Server Userid-Password Plugin (srvcon_pw_plugin)、および Group Plugin (group_plugin) に対して、デフォルトの空の値を使用していることを確認します。デフォルトのプラグインは IBMOSauthclient、IBMOSauthserver、および IBMOSgroups であり、プラグイン名の値を空のままにしておいた場合、これらのプラグインが暗黙的に指定されます。

d. DB2 インスタンスを再始動します。

注: IBMLDAPSecurity.ini は透過的 LDAP では使用されません。このファイルは、LDAP プラグイン・モジュールでのみ使用されます。

認証およびグループ検索のための透過的 LDAP の構成 (Solaris)

DB2 V9.7 フィックスパック 1 から、Solaris オペレーティング・システムで、DB2 データベース・サーバーが LDAP ベースの認証を透過的に使用できるようにするには、 Pluggable Authentication Module (PAM) を使用します。ユーザーおよびグループの情報を格納するように、事前に LDAP サーバーを構成しておく必要があります。

始める前に

以下の手順では、LDAP サーバーが RFC 2307 に準拠すると想定しています。

このタスクについて

この作業は、Solaris 10 に適用可能なステップを説明したものです。Solaris オペレーティング・システムのバージョンによっては、指示が若干異なる可能性があります。

手順

1. オペレーティング・システムを LDAP および PAM 用に構成するには、以下の手順を実行します。
   a. root 権限を持つユーザーとしてログインします。
   b. nss_ldap および pam_ldap パッケージがインストールされていることを確認します。これらの 2 つのパッケージは、/usr/lib および /usr/lib/security ディレクトリー内に、nss_ldap.so および pam_ldap.so として存在します。
   c. オペレーティング・システムを LDAP クライアント・マシンとして動作するようにセットアップします。 ldapclient(1M) インターフェースを使用して、**ldapclient** コマンドを実行できます。以下は、出力例です。

```
ldapclient manual -a credentialLevel=proxy ¥
-a authenticationMethod=simple ¥
-a proxyDN=<root> ¥
-a proxyPassword=<password> ¥
-a defaultSearchBase=<base> ¥
-a serviceSearchDescriptor=group:<group> ¥
-a domainName=<domain> ¥
-a defaultServerList=<IP>
```

      説明

      <root> LDAP にバインドするバインド DN です。これは、 LDAP サーバーを使用してユーザー・アカウントおよびグループを検索することを許可された LDAP サーバーのユーザー項目の DN です。

      <password>
      バインド DN のパスワードです。

      <base> 検索基底 DN です。これは、ユーザーおよびグループ項目より 1 つ上位のレベルにします。

      <group>
      グループ情報が格納されている場所の基底 DN です。

      <domain>
      LDAP サーバーのドメイン名です。

      <IP> LDAP サーバーの IP アドレスです。

      詳しくは、ldapclient(1M) のマニュアルを参照してください。

   d. /etc/pam.conf にある PAM 構成ファイルを編集します。以下のテキストをファイルに追加します。

```
db2 auth requisite          pam_authtok_get.so.1
db2 auth required           pam_unix_cred.so.1
db2 auth sufficient         pam_unix_auth.so.1
db2 auth required           pam_ldap.so.1
```

      上記の構成では、初めにローカル・ファイル・システムで、ユーザー ID およびパスワードを検索します。ユーザーが見つからなかった場合、またはローカル・ファイル・システムでの認証に失敗した場合にのみ、 LDAP 検索が実行されます。

DB2 は、pam_ldap.so と pam_unix_auth.so を使用する PAM 構成をサポートします。他の PAM モジュールを使用する構成は作動する可能性がありますが、サポートされていません。

e. LDAP を介してグループ検索を実行するように Solaris システムをセットアップします。 /etc/nsswitch.conf ファイル内の **group** および **passwd** の項目を見つけ、検索方法として ldap が記入されていることを確認します。 以下は、**group** および **passwd** 項目の例です。

```
group:          files ldap
passwd:         files ldap
```

2. 以下の手順を実行して、透過的 LDAP 認証を使用するように DB2 インスタンスを構成します。

a. **DB2AUTH** 各種レジストリー変数を OSAUTHDB に設定します。 SYSADM 権限を持つユーザーで、以下のコマンドを実行します。

```
db2set DB2AUTH=OSAUTHDB
```

b. サーバー認証を、以下の任意のタイプに設定します。

- SERVER
- SERVER_ENCRYPT
- DATA_ENCRYPT

c. デフォルトのクライアント・ユーザー ID/パスワード・プラグイン (CLNT_PW_PLUGIN)、サーバー・ユーザー ID/パスワード・プラグイン (SRVCON_PW_PLUGIN)、およびグループ・プラグイン (GROUP_PLUGIN) を使用していることを確認します。

d. DB2 インスタンスを再始動します。

注: IBMLDAPSecurity.ini は透過的 LDAP では使用されません。このファイルは、LDAP プラグイン・モジュールでのみ使用されます。

LDAP プラグイン・モジュールの構成

LDAP プラグイン・モジュールを構成するには、それぞれの環境に合わせて、IBM LDAP セキュリティー・プラグイン構成ファイルを更新する必要があります。ほとんどの場合は、LDAP 管理者に問い合わせて、適切な構成値を確認することが必要です。

IBM LDAP セキュリティー・プラグイン構成ファイルのデフォルトの名前と場所は、以下のとおりです。

- UNIX: INSTHOME/sqllib/cfg/IBMLDAPSecurity.ini
- Windows: %DB2PATH%¥cfg¥IBMLDAPSecurity.ini

オプションとして、DB2LDAPSecurityConfig 環境変数を使用して、このファイルの場所を指定することも可能です。Windows では、グローバル・システム環境で DB2LDAPSecurityConfig を設定し、DB2 サービスからその値を参照できるようにしておく必要があります。

適切な構成値を確認するときに役立つ情報を以下の表にまとめます。

表 31. サーバー関連の値

パラメーター	説明
LDAP_HOST	LDAP サーバーの名前。 これは、LDAP サーバーのホスト名または IP アドレスのスペース区切りリストであり、オプションとしてそれぞれのポート番号を組み込むこともできます。 例えば、`host1[:port] [host2:[port2] ... ]` のようになります。 デフォルトのポート番号は 389 です。 SSL が有効になっている場合は 636 になります。
ENABLE_SSL	SSL サポートを使用可能にするには、ENABLE_SSL を TRUE を設定します (GSKit をインストールしておく必要があります)。これは、オプション・パラメーターであり、デフォルトでは FALSE (SSL サポートなし) になります。
SSL_KEYFILE	SSL 鍵リングのパス。 鍵ファイルが必要になるのは、GSKit のインストールで自動的にトラステッドに設定されない証明書を LDAP サーバーで使用している場合に限られます。 例えば、`SSL_KEYFILE = /home/db2inst1/IBMLDAPSecurity.kdb` のようになります。
SSL_PW	SSL 鍵リングのパスワード。例えば、`SSL_PW = keyfile-password` のようになります。

表 32. ユーザー関連の値

パラメーター	説明
USER_OBJECTCLASS	ユーザー用の LDAP オブジェクト・クラス。 通常は、USER_OBJECTCLASS を `inetOrgPerson` (Microsoft Active Directory の場合はそのユーザー) に設定します。 例えば、`USER_OBJECTCLASS = inetOrgPerson` のようになります。
USER_BASEDN	ユーザー検索時に使用する LDAP 基底 DN。 指定しない場合は、ユーザー検索が LDAP ディレクトリーのルートから始まります。 いくつかの LDAP サーバーでは、このパラメーターの値を指定することが必須になっています。 例えば、`USER_BASEDN = o=ibm` のようになります。

USERID_ATTRIBUTE	ユーザー ID に対応する LDAP ユーザー属性。
	ユーザーが非修飾ユーザー ID で DB2 CONNECT ステートメントを実行すると、USERID_ATTRIBUTE 属性、USER_OBJECTCLASS 属性、USER_BASEDN 属性 (指定されている場合) の組み合わせによって LDAP 検索フィルターが構成されます。
	例えば、`USERID_ATTRIBUTE = uid` の場合に、以下のステートメントを実行するとします。
	`db2 connect to MYDB user bob using bobpass`
	結果として、以下のような検索フィルターが生成されます。
	`&(objectClass=inetOrgPerson)(uid=bob)`

AUTHID_ATTRIBUTE	DB2 許可 ID に対応する LDAP ユーザー属性。 通常は、USERID_ATTRIBUTE と同じです。 例えば、`AUTHID_ATTRIBUTE = uid` のようになります。

表 33. グループ関連の値

パラメーター	説明
GROUP_ OBJECTCLASS	グループ用の LDAP オブジェクト・クラス。 通常は、groupOfNames または groupOfUniqueNames (Microsoft Active Directory の場合は group) です。 例えば、GROUP_OBJECTCLASS = groupOfNames のようになります。
GROUP_BASEDN	グループ検索時に使用する LDAP 基底 DN。 指定しない場合は、グループ検索が LDAP ディレクトリーのルートから始まります。いくつかの LDAP サーバーでは、このパラメーターの値を指定することが必須になっています。 例えば、GROUP_BASEDN = o=ibm のようになります。
GROUPNAME_ ATTRIBUTE	グループの名前に対応する LDAP グループ属性。 例えば、GROUPNAME_ATTRIBUTE = cn のようになります。
GROUP_LOOKUP_ METHOD	ユーザーのグループ・メンバーシップを確認するための方法を指定します。可能な値は以下のとおりです。 • SEARCH_BY_DN: ユーザーをメンバーとしてリストするグループを検索します。メンバーシップは、GROUP_LOOKUP_ATTRIBUTE として定義するグループ属性 (通常は member または uniqueMember) によって示します。 • USER_ATTRIBUTE: この場合は、ユーザーの所属するグループがユーザー・オブジェクト自体の属性としてリストされます。この設定では、GROUP_LOOKUP_ATTRIBUTE として定義するユーザー属性を検索して、ユーザーの所属するグループを取得します (基本的に、Microsoft Active Directory の場合は memberOf、IBM Tivoli Directory Server の場合は ibm-allGroups です)。 例: GROUP_LOOKUP_METHOD = SEARCH_BY_DN GROUP_LOOKUP_METHOD = USER_ATTRIBUTE
GROUP_LOOKUP_ ATTRIBUTE	グループ・メンバーシップを確認するために使用する属性の名前 (GROUP_LOOKUP_METHOD の説明を参照)。 例: GROUP_LOOKUP_ATTRIBUTE = member GROUP_LOOKUP_ATTRIBUTE = ibm-allGroups
NESTED_GROUPS	NESTED_GROUPS を TRUE にすると、DB2 データベース・マネージャーは、グループ・メンバーシップを再帰的に検索するために、検出されるすべてのグループのグループ・メンバーシップを検索しようとします。 A が B に所属し、B が A に所属するといった循環も、正しく処理されます。 このパラメーターはオプションであり、デフォルトでは FALSE になります。

表 34. その他の値

パラメーター	説明

SEARCH_DN, SEARCH_PW	LDAP サーバーが匿名アクセスをサポートしていない場合や、ユーザーまたはグループの検索時に匿名アクセスでは不十分な場合は、検索実行時に使用する DN とパスワードをオプションとして指定できます。
	
	例:
	`SEARCH_DN = cn=root`
	`SEARCH_PW = rootpassword`

DEBUG	DEBUG を TRUE に設定すると、LDAP 関連の問題のデバッグに役立つ追加情報が **db2diag** ログ・ファイルに書き込まれます。 ほとんどの追加情報は、DIAGLEVEL 4 (INFO) でログに記録されます。DEBUG は、デフォルトで false になります。

LDAP プラグイン・モジュールの使用可能化

コンパイル済みのバイナリー LDAP プラグイン・モジュールが DB2 インスタンス・ディレクトリーに入っています。

DB2 インスタンスの中でそれぞれの LDAP プラグイン・モジュールが格納されている場所を以下の表にまとめます。

表 35. 64 ビットの UNIX システムと Linux システム

プラグイン・モジュールのタイプ	ロケーション
サーバー	`/sqllib/security64/plugin/IBM/server`
クライアント	`/sqllib/security64/plugin/IBM/client`
group	`/sqllib/security64/plugin/IBM/group`

表 36. 32 ビットの UNIX システムと Linux システム

プラグイン・モジュールのタイプ	ロケーション
サーバー	`/sqllib/security32/plugin/IBM/server`
クライアント	`/sqllib/security32/plugin/IBM/client`
group	`/sqllib/security32/plugin/IBM/group`

表 37. Windows システム (64 ビットと 32 ビットの両方)

プラグイン・モジュールのタイプ	ロケーション
サーバー	`%DB2PATH%¥security¥plugin¥IBM¥`*instance-name*`¥server`
クライアント	`%DB2PATH%¥security¥plugin¥IBM¥`*instance-name*`¥client`
group	`%DB2PATH%¥security¥plugin¥IBM¥`*instance-name*`¥group`

注: 64 ビットの Windowsプラグイン・モジュールの場合は、ファイル名に 64 という数字が入っています。

DB2 コマンド行プロセッサーを使用して、データベース・マネージャー構成を更新し、必要なプラグイン・モジュールを使用可能にします。

- サーバー・プラグイン・モジュール:

  ```
  UPDATE DBM CFG USING SRVCON_PW_PLUGIN IBMLDAPauthserver
  ```

- クライアント・プラグイン・モジュール:

  ```
  UPDATE DBM CFG USING CLNT_PW_PLUGIN IBMLDAPauthclient
  ```

- グループ・プラグイン・モジュール:

  ```
  UPDATE DBM CFG USING GROUP_PLUGIN IBMLDAPgroups
  ```

db2 terminate コマンドを使用して、DB2 コマンド行プロセッサーの実行中のすべてのバックエンド・プロセスを終了してから、**db2stop** コマンドと **db2start** コマンドを使用してインスタンスを停止して再始動します。

LDAP ユーザー ID による接続

DB2 インスタンスで LDAP セキュリティー・プラグインを構成すると、ユーザーはさまざまな種類のユーザー・ストリングを使ってデータベースに接続できるようになります。

LDAP ディレクトリー内のオブジェクトの位置は、識別名 (DN) によって定義します。DN は基本的に、ある種の階層を反映した複数パーツの名前です。例えば、次のようになります。

```
cn=John Smith, ou=Sales, o=WidgetCorp
```

ユーザーのユーザー *ID* は、ユーザー・オブジェクトに関連した属性 (通常は **uid** 属性) によって定義します。単純なストリング (例えば jsmith) の場合もあれば、組織階層の一部を反映した E メール・アドレス (例えば jsmith@sales.widgetcorp.com) のような場合もあります。

ユーザーの DB2 *許可 ID* は、DB2 データベース内のユーザーに関連した名前です。

以前は、サーバーのホスト・オペレーティング・システムでユーザーを定義するのが普通で、ユーザー ID と許可 ID は同じでした (ただし、許可 ID は基本的に大文字で表記します)。DB2 LDAP プラグイン・モジュールを使用すれば、LDAP ユーザー・オブジェクトのさまざまな属性をユーザー ID と許可 ID に関連付けることが可能になります。ほとんどの場合は、ユーザー ID と許可 ID を同じストリングにして、USERID_ATTRIBUTE と AUTHID_ATTRIBUTE の両方に同じ属性名を使用できます。ただし、許可 ID には組み込みたくない余分の情報がユーザー ID 属性に含まれていることが多い環境では、プラグインの初期設定ファイルで別の AUTHID_ATTRIBUTE を構成することも可能です。サーバーは、その AUTHID_ATTRIBUTE 属性の値を取り出して、ユーザーの内部 DB2 表記としてその値を使用します。

例えば、LDAP ユーザー ID が E メール・アドレス (例えば jsmith@sales.widgetcorp.com) のようになっているとします。DB2 許可 ID としてユーザーの部分 (jsmith) だけを使用する場合は、以下のようにします。

1. 短縮名が含まれている新しい属性を LDAP サーバーのすべてのユーザー・オブジェクトに関連付けます。
2. AUTHID_ATTRIBUTE にその新しい属性の名前を設定します。

ユーザーは、完全な LDAP ユーザー ID とパスワードを使用して、DB2 データベースに接続できます。例えば、次のようにします。

```
db2 connect to MYDB user 'jsmith@sales.widgetcorp.com' using 'pswd'
```

ところが、内部では、DB2 データベース・マネージャーは、AUTHID_ATTRIBUTE で取得した短縮名 (この場合は jsmith) を使用してユーザーを参照します。

LDAP プラグイン・モジュールを使用可能にして構成すると、ユーザーは、さまざまなストリングを使用して DB2 データベースに接続できるようになります。

- 完全 DN。例えば、以下のようにします。

  ```
  connect to MYDB user 'cn=John Smith, ou=Sales, o=WidgetCorp'
  ```

- 部分 DN。部分 DN を使用して LDAP ディレクトリーを検索するときに、適切な検索基底 DN が定義されていれば、1 つの一致項目が検出されます。例えば、以下のようにします。

  ```
  connect to MYDB user 'cn=John Smith' connect to MYDB user uid=jsmith
  ```

- 単純なストリング (等号が含まれていないストリング)。そのストリングは、USERID_ATTRIBUTE で修飾されて、部分 DN として処理されます。例えば、以下のようにします。

  ```
  connect to MYDB user jsmith
  ```

注: CONNECT ステートメントまたは **ATTACH** コマンドに指定するストリングにスペースや特殊文字が含まれている場合は、そのストリングを単一引用符で区切っておく必要があります。

グループ検索に関する考慮事項

LDAP サーバーでは通常、ユーザー・オブジェクトの属性またはグループ・オブジェクトの属性としてグループ・メンバーシップ情報を表します。

- ユーザー・オブジェクトの属性として

  各ユーザー・オブジェクトには、GROUP_LOOKUP_ATTRIBUTE という属性があります。この属性に対して照会を実行すれば、そのユーザーのすべてのグループ・メンバーシップを取得できます。

- グループ・オブジェクトの属性として

  各グループ・オブジェクトにも、GROUP_LOOKUP_ATTRIBUTE という属性があります。その属性を使用すれば、そのグループのメンバーになっているすべてのユーザーをリストできます。特定ユーザーが属しているすべてのグループを列挙するために、ユーザー・オブジェクトがメンバーとして含まれているすべてのグループを検索することも可能です。

そのどちらかの方法で構成できる LDAP サーバーは多くありますし、両方の方法を同時にサポートしている LDAP サーバーもいくつかあります。ご使用の LDAP サーバーの構成方法については、LDAP 管理者に問い合わせてください。

LDAP プラグイン・モジュールを構成するときには、GROUP_LOOKUP_METHOD パラメーターを使用して、グループ検索の実行方法を指定できます。

- ユーザー・オブジェクトの GROUP_LOOKUP_ATTRIBUTE 属性を使用してグループ・メンバーシップを確認する必要がある場合は、GROUP_LOOKUP_METHOD = USER_ATTRIBUTE を設定します。
- グループ・オブジェクトの GROUP_LOOKUP_ATTRIBUTE 属性を使用してグループ・メンバーシップを確認する必要がある場合は、GROUP_LOOKUP_METHOD = SEARCH_BY_DN を設定します。

多くの LDAP サーバーは、グループ・オブジェクトの GROUP_LOOKUP_ATTRIBUTE 属性を使用してメンバーシップを確認します。その場合は、以下の例のように構成できます。

```
GROUP_LOOKUP_METHOD = SEARCH_BY_DN
GROUP_LOOKUP_ATTRIBUTE = groupOfNames
```

Microsoft Active Directory は通常、ユーザー属性としてグループ・メンバーシップを格納します。その場合は、以下の例のように構成できます。

```
GROUP_LOOKUP_METHOD = USER_ATTRIBUTE
GROUP_LOOKUP_ATTRIBUTE = memberOf
```

IBM Tivoli Directory Server は、両方の方法を同時にサポートしています。ユーザーのグループ・メンバーシップを照会するには、特別なユーザー属性 **ibm-allGroups** を以下の例のように使用できます。

```
GROUP_LOOKUP_METHOD = USER_ATTRIBUTE
GROUP_LOOKUP_ATTRIBUTE = ibm-allGroups
```

他の LDAP サーバーでも、グループ・メンバーシップを取得するために、それと似たような特別な属性が用意されている場合があります。一般に、ユーザー属性によってメンバーシップを取得する方が、ユーザーをメンバーとしてリストするグループを検索するよりも高速です。

LDAP ユーザーの認証とグループの検索に関するトラブルシューティング

LDAP ユーザーの認証やグループの検索で問題が発生した場合は、**db2diag** ログ・ファイルと管理ログがトラブルシューティングのための情報源になります。

LDAP プラグイン・モジュールは通常、障害発生時の LDAP 戻りコード、検索フィルター、その他の役立つデータをログに記録します。LDAP プラグイン・モジュールの構成ファイルで DEBUG オプションを有効にすると、さらに多くの情報が **db2diag** ログ・ファイルに書き込まれます。その情報は確かにトラブルシューティングに役立ちますが、すべての追加データを 1 つのファイルに書き込む処理に関連したオーバーヘッドを考えると、実動システムでそのオプションを広範囲に使用することはお勧めできません。

データベース・マネージャーで **diaglevel** 構成パラメーターを 4 に設定して、LDAP プラグイン・モジュールから送られてくるすべてのメッセージをキャプチャーできるようにしてください。

セキュリティー・プラグインの作成

DB2 によるセキュリティー・プラグインのロード方法

DB2 データベース・システムがセキュリティー・プラグイン関数を呼び出すのに必要な情報を持てるよう、セキュリティー・プラグインには正しくセットアップされた初期化関数が必要です。

各プラグイン・ライブラリーには、以下のような、プラグイン・タイプに応じた特定の名前を持つ初期化関数が含まれていなければなりません。

- サーバー・サイド認証プラグイン: db2secServerAuthPluginInit()
- クライアント・サイド認証プラグイン: db2secClientAuthPluginInit()
- グループ・プラグイン: db2secGroupPluginInit()

この関数は、プラグイン初期化関数と呼ばれます。プラグイン初期化関数は、指定されたプラグインを初期化し、プラグインの関数を呼び出すために必要な情報を DB2 に提供します。プラグイン初期化関数は、以下のパラメーターを受け入れます。

- プラグインを呼び出す DB2 インスタンスがサポートできる関数ポインター構造の最高バージョン番号
- インプリメンテーションを必要とするすべての API を指すポインターを収めた構造を指すポインター
- **db2diag** ログ・ファイルにログ・メッセージを追加する関数を指すポインター
- エラー・メッセージ・ストリングを指すポインター
- エラー・メッセージの長さ

以下は、グループ検索プラグインの初期化関数の関数シグニチャーです。

```
SQL_API_RC SQL_API_FN db2secGroupPluginInit(
  db2int32 version,
  void *group_fns,
  db2secLogMessage *logMessage_fn,
  char **errormsg,
  db2int32  *errormsglen);
```

注: プラグイン・ライブラリーを C++ でコンパイルする場合は、 extern "C" を使用してすべての関数を宣言する必要があります。 DB2 は、基礎オペレーティング・システムの動的ローダーを利用して、C++ のユーザー作成プラグイン・ライブラリーの内部で使用されている C++ コンストラクターおよびデストラクターを処理します。

初期化関数は、規定の関数名を使用しなければならない、プラグイン・ライブラリー内の唯一の関数です。その他のプラグイン関数は、初期化関数から戻された関数ポインターを通して参照されます。サーバー・プラグインは、DB2 サーバーの始動時にロードされます。クライアント・プラグインは、クライアント上で必要とされ

るときにロードされます。 DB2 は、プラグイン・ライブラリーをロードするとすぐに、この初期化関数の位置を解決して呼び出します。この関数固有のタスクは、以下のとおりです。

- 関数ポインターを、適切な関数構造を指すポインターにキャストする
- ライブラリー内の他の関数を指すポインターを指定する
- 返される関数ポインター構造のバージョン番号を指定する

DB2 はプラグイン初期化関数を複数回呼び出すことがあります。このことが起こるのは、アプリケーションが動的に DB2 クライアント・ライブラリーをロードしてからこれをアンロードして再ロードし、再ロードの前と後の両方にプラグインから認証関数を実行した場合です。このような場合は、プラグイン・ライブラリーがアンロードされず、したがって再ロードもされないことがあります。ただし、この動作はオペレーティング・システムによって異なります。

別の例として、データベース・サーバー自身がクライアントとして振る舞うことがある、ストアード・プロシージャーやフェデレーテッド・システム呼び出しの実行時にも、DB2 がプラグイン初期化関数を複数回呼び出すことがあります。データベース・サーバー上のクライアント・プラグインとサーバー・プラグインが同じファイル内にある場合、 DB2 はプラグイン初期化関数を 2 回呼び出す可能性があります。

`db2secGroupPluginInit` が複数回呼び出されたことを検出した場合、プラグインは、プラグイン・ライブラリーを終了して再初期化するよう指示されたものとして、このイベントを処理する必要があります。したがって、プラグイン初期化関数は、`db2secPluginTerm` を呼び出すと実行されるクリーンアップ・タスクをすべて実行してから、再び関数ポインターのセットを戻す必要があります。

UNIX または Linux ベースのオペレーティング・システムが稼働している DB2 サーバーでは、DB2 は異なるプロセスでプラグイン・ライブラリーを複数回ロードして再初期化することがあります。

セキュリティー・プラグイン・ライブラリーの開発に関する制約事項

プラグイン・ライブラリーの開発方法に影響を及ぼす制約事項が幾つかあります。

次のリストは、プラグイン・ライブラリーの作成に関連した制約事項の概要を示しています。

C-linkage
: プラグイン・ライブラリーは、C-linkage とリンクされていなければなりません。プロトタイプ、プラグインのインプリメントに必要なデータ構造、およびエラー・コード定義を規定するヘッダー・ファイルは、C/C++ の場合にのみ準備されます。プラグイン・ライブラリーが C++ としてコンパイルされている場合は、 DB2 がロード時に解決する関数を extern "C" を用いて宣言する必要があります。

.NET 共通言語ランタイムはサポートされていません。
: プラグイン・ライブラリーのソース・コードのコンパイルおよびリンクにおいて、 .NET 共通言語ランタイム (CLR) はサポートされません。

シグナル・ハンドラー

プラグイン・ライブラリーは、シグナル・ハンドラーをインストールしたり、シグナル・マスクを変更したりしてはなりません。なぜなら、これをすると、DB2 のシグナル・ハンドラーが妨げられるからです。 DB2 のシグナル・ハンドラーが妨げられると、プラグイン・コード自体にあるトラップを含めたエラーを報告してリカバリーする DB2 の機能が著しく妨げられます。さらに、プラグイン・ライブラリーは、決して C++ 例外をスローしないようにする必要があります。これによって、DB2 で使用されるエラー処理が妨げられるおそれがあるからです。

スレッド・セーフ

プラグイン・ライブラリーは、スレッド・セーフおよび再入可能でなければなりません。プラグイン初期化関数は、再入可能でなくてもよい唯一の API です。プラグイン初期化関数は異なるプロセスから複数回呼び出される可能性がありますが、その場合は、プラグインがすべての使用済みリソースをクリーンアップして、プラグイン自体を再初期化します。

終了ハンドラー、および標準 C ライブラリーとオペレーティング・システム呼び出しのオーバーライド

プラグイン・ライブラリーは、標準 C ライブラリーやオペレーティング・システム呼び出しをオーバーライドしてはなりません。さらに、プラグイン・ライブラリーは、終了ハンドラーや `pthread_atfork` ハンドラーをインストールしてはなりません。終了ハンドラーはプログラムが終了する前にアンロードされる可能性があるので、終了ハンドラーを使用することはお勧めしません。

ライブラリーの従属関係

Linux または UNIX では、プラグイン・ライブラリーをロードするプロセスは、`setuid` か `setgid` になります。このことは、プロセスが `$LD_LIBRARY_PATH`、 `$SHLIB_PATH`、または `$LIBPATH` 環境変数を利用して従属ライブラリーを検索できないことを意味します。そのため、プラグイン・ライブラリーは追加のライブラリーに依存しないようにする必要があります。ただし、従属ライブラリーが以下の状態のような他の方法でアクセス可能になっている場合を除きます。

- `/lib` または `/usr/lib` の中に入れる。
- それらが常駐するディレクトリーを OS ワイド (Linux 上の `ld.so.conf` ファイル内など) で指定する。
- プラグイン・ライブラリー自体の `RPATH` で指定する。

この制限は、Windows オペレーティング・システムには当てはまりません。

シンボルの重複

可能であれば、プラグイン・ライブラリーは、シンボルの重複の可能性を減らすオプションとして使用できるオプション (アンバインドされた外部シンボル参照を削減するオプションなど) を用いてコンパイルおよびリンクする必要があります。例えば、HP、Solaris、および Linux 上で "-Bsymbolic" リンカー・オプションを使用するなら、シンボルの重複に関係した問題を防ぐことができます。ただし、AIX で作成されたプラグインの場合は、`"-brtl"` リンカー・オプションは明示的にも暗黙的にも使用しないでください。

32 ビット・アプリケーションと 64 ビット・アプリケーション

32 ビット・アプリケーションは、32 ビット・プラグインを使用する必要があります。 64 ビット・アプリケーションは、64 ビット・プラグインを使用する必要があります。詳細については、32 ビットと 64 ビットの考慮事項に関するトピックを参照してください。

テキスト・ストリング

入力テキスト・ストリングがヌル終了になっているという保証はなく、出力ストリングがヌル終了である必要はありません。その代わり、すべての入力ストリングに対して整数の長さが指定され、戻される長さとして整数を指すポインターが指定されます。

許可 ID パラメーターの引き渡し

DB2 がプラグインに渡す許可 ID (authid) パラメーター (入力 authid パラメーター) には、埋め込みブランクが除かれた大文字の authid が含まれます。プラグインが DB2 に戻す authid パラメーター (出力 authid パラメーター) には特別な処理は必要ありませんが、DB2 は authid を大文字に変換し、内部 DB2 規格に準じてブランクを埋め込みます。

パラメーターのサイズ制限

プラグイン API は、パラメーターの長さ制限として以下を使用します。

```
#define DB2SEC_MAX_AUTHID_LENGTH 255
#define DB2SEC_MAX_USERID_LENGTH 255
#define DB2SEC_MAX_USERNAMESPACE_LENGTH 255
#define DB2SEC_MAX_PASSWORD_LENGTH 255
#define DB2SEC_MAX_DBNAME_LENGTH 128
```

特定のプラグイン・インプリメンテーションでは、許可 ID、ユーザー ID、およびパスワードの最大長は、小さくする必要があるか、あるいは強制的に小さくされる可能性があります。特に、DB2 データベース・システムに付属しているオペレーティング・システム認証プラグインは、オペレーティング・システムの限界が上記の限界より低い場合、オペレーティング・システムが施行する最大ユーザー長、最大グループ長、および最大ネーム・スペース長の限界の制約を受けます。

AIX でのセキュリティー・プラグイン・ライブラリーの拡張子

AIX システムでは、セキュリティー・プラグイン・ライブラリーは、*.a* または *.so* というファイル名拡張子を持つことができます。プラグイン・ライブラリーをロードするのに使用するメカニズムは、次のように、どの拡張子が使用されているかによって異なります。

- ファイル名拡張子が *.a* のプラグイン・ライブラリーは、共有オブジェクト・メンバーを含むアーカイブであると想定されます。そのようなメンバーには、*shr.o* (32 ビット) または *shr64.o* (64 ビット) という名前を付けなければなりません。 32 ビットおよび 64 ビットの両方のメンバーを 1 つのアーカイブに収容することができ、それによって、両方のタイプのプラットフォームにデプロイすることができます。

  例えば、32 ビット・アーカイブ・スタイルのプラグイン・ライブラリーを作成するには、次のようにします。

  ```
  xlc_r -qmkshrobj -o shr.o MyPlugin.c -bE:MyPlugin.exp
  ar rv MyPlugin.a shr.o
  ```

- ファイル名拡張子が *.so* のプラグイン・ライブラリーは、動的にロード可能な共有オブジェクトであると想定されます。そのようなオブジェクトは、その作成時に使用したコンパイラーおよびリンカーのオプションに応じて、32 ビットまたは 64 ビットのどちらかになります。例えば、32 ビットのプラグイン・ライブラリーを作成するには、次のようにします。

  ```
  xlc_r -qmkshrobj -o MyPlugin.so MyPlugin.c -bE:MyPlugin.exp
  ```

  AIX 以外のすべてのプラットフォームでは、セキュリティー・プラグイン・ライブラリーは、常に動的にロード可能な共有オブジェクトであるとみなされます。

fork プラグイン・ライブラリーは fork しないでください。ファイル記述子およびソケットが子プロセス内に複製されるため、ハングしたり誤った動作になったりするおそれがあるからです。ファイルにオープン・ファイル記述子がある場合は特に、子プロセスが fork されると、偽のファイル・ロック競合が発生する可能性があります。また、セマフォーなどの他のリソースが fork によっていくつも継承される可能性もあります。

セキュリティー・プラグインに関する制約事項

セキュリティー・プラグインの使用に関する幾らかの制約事項があります。

DB2 データベース・ファミリーのサポートに関する制約事項

GSS-API プラグインを使用して、Linux、UNIX、およびWindows 上の DB2 クライアントと、DB2 for z/OS などの別の DB2 ファミリー・サーバーとの間の接続を認証することはできません。また、クライアントとして機能する他の DB2 データベース・ファミリー製品から Linux、UNIX、または Windows 上の DB2 サーバーへの接続も認証できません。

ただし、Linux、UNIX、または Windows 上の DB2 クライアントを使用して他の DB2 データベース・ファミリー・サーバーに接続する場合には、クライアント・サイドのユーザー ID/パスワード・プラグイン (IBM 提供のオペレーティング・システム認証プラグインなど) を使用したり、独自のユーザー ID/パスワード・プラグインを作成したりすることができます。また、組み込みの Kerberos プラグインの使用や、自分独自のプラグインのインプリメントを行ってもかまいません。

Linux、 UNIX、または Windows 上の DB2 クライアントでは、GSSPLUGIN 認証タイプを使用してデータベースをカタログしてはなりません。

AUTHID ID に関する制限DB2 データベース・システムのバージョン 9.5 以降では、128 バイトの許可 ID を持つことができますが、その許可 ID がオペレーティング・システムのユーザー ID またはグループ名として解釈される場合、オペレーティング・システムの命名上の制約が適用されます (例えば、ユーザー ID の制限は 8 または 30 文字、グループ名の制限は 30 文字です)。このため、128 バイトの許可 ID を付与できますが、この許可 ID を持つユーザーとしては接続することができません。独自のセキュリティー・プラグインを作成した場合は、許可 ID の拡張されたサイズを最大限に活用することができます。例えば、セキュリティー・プラグインに 30 バイトのユーザー ID を与えて、接続可能な認証中に、セキュリティー・プラグインが 128 バイトの許可 ID を返すことができます。

InfoSphere® フェデレーション・サーバーのサポートに関する制約事項

DB2 II は、GSS_API プラグインからの委任証明書を使用して、データ・ソースへのアウトバウンド接続を確立することをサポートしていません。データ・ソースへの接続には、引き続き CREATE USER MAPPING コマンドを使用する必要があります。

データベース管理サーバーのサポートに関する制約事項

DB2 Administration Server (DAS) はセキュリティー・プラグインをサポートしていません。DAS はオペレーティング・システムの認証メカニズムのみをサポートします。

DB2 クライアントでのセキュリティー・プラグインに関する問題および制約事項 (Windows)

Windows オペレーティング・システム上の DB2 クライアント内でデプロイする予定のセキュリティー・プラグインの開発時には、プラグイン終了関数の中でどの補助ライブラリーもアンロードしないでください。この制約事項は、グループ、ユーザー ID とパスワード、Kerberos、および GSS-API プラグインを含む、すべてのタイプのクライアント・セキュリティー・プラグインに対して適用されます。このような、db2secPluginTerm、db2secClientAuthPluginTerm、および db2secServerAuthPluginTerm といった終了 API は、どの Windows プラットフォームでも呼び出されないので、該当するリソース・クリーンアップを行う必要があります。

この制約事項は、Windows での DLL のアンロードに関連したクリーンアップ問題に関係しています。

AIX 上での .a または .so の拡張子の付いたプラグイン・ライブラリーのロード

AIX では、セキュリティー・プラグイン・ライブラリーには、.a または .so のファイル名拡張子をつけることができます。プラグイン・ライブラリーをロードするのに使用するメカニズムは、次のように、どの拡張子が使用されているかによって異なります。

- .a のファイル名拡張子の付いたプラグイン・ライブラリー

  .a のファイル名拡張子の付いたプラグイン・ライブラリーは、共有オブジェクト・メンバーを収容するアーカイブであるとみなされます。そのようなメンバーには、shr.o (32 ビット) または shr64.o (64 ビット) という名前を付けなければなりません。 32 ビットおよび 64 ビットの両方のメンバーを 1 つのアーカイブに収容することができ、それによって、両方のタイプのプラットフォームにデプロイすることができます。

  例えば、32 ビット・アーカイブ・スタイルのプラグイン・ライブラリーを作成するには、次のようにします。

  ```
  xlc_r -qmkshrobj -o shr.o MyPlugin.c -bE:MyPlugin.exp
  ar rv MyPlugin.a shr.o
  ```

- .so のファイル名拡張子の付いたプラグイン・ライブラリー

.so のファイル名拡張子の付いたプラグイン・ライブラリーは、動的にロード可能な共有オブジェクトであるとみなされます。そのようなオブジェクトは、その作成時に使用したコンパイラーおよびリンカーのオプションに応じて、32 ビットまたは 64 ビットのどちらかになります。例えば、32 ビットのプラグイン・ライブラリーを作成するには、次のようにします。

```
xlc_r -qmkshrobj -o MyPlugin.so MyPlugin.c -bE:MyPlugin.exp
```

AIX 以外のすべてのプラットフォームでは、セキュリティー・プラグイン・ライブラリーは、常に動的にロード可能な共有オブジェクトであるとみなされます。

GSS-API セキュリティー・プラグインでは、メッセージの暗号化と署名はサポートされない

メッセージの暗号化および署名は、GSS-API セキュリティー・プラグインでは使用できません。

セキュリティー・プラグインの戻りコード

すべてのセキュリティー・プラグイン API は、API の実行の成功や失敗を示すために整数の値を戻す必要があります。戻りコード値 0 は、API が正常に実行したことを示します。-3、-4、および -5 以外のすべての負の戻りコードは、API がエラーを検出したことを示します。

-3、-4、または -5 が付く戻りコードを除き、セキュリティー・プラグイン API から戻されるすべての負の戻りコードは、SQLCODE -1365、SQLCODE -1366、または SQLCODE -30082 にマップされます。-3、-4、および -5 の値は、許可 ID が有効なユーザーまたはグループを表しているかどうかを示すために使用されます。

すべてのセキュリティー・プラグイン API の戻りコードは、DB2 の組み込みディレクトリー SQLLIB/include にある db2secPlugin.h で定義されます。

すべてのセキュリティー・プラグインの戻りコードに関する詳細については、以下の表で説明しています。

表 38. セキュリティー・プラグインの戻りコード

戻りコード	定義値	意味	関連 API
0	DB2SEC_PLUGIN_OK	プラグイン API が正常に実行されました。	すべて
-1	DB2SEC_PLUGIN_UNKNOWNERROR	プラグイン API で想定外のエラーが発生しました。	すべて
-2	DB2SEC_PLUGIN_BADUSER	入力として渡されたユーザー ID が定義されていません。	db2secGenerateInitialCred db2secValidatePassword db2secRemapUserid db2secGetGroupsForUser
-3	DB2SEC_PLUGIN _INVALIDUSERORGROUP	このユーザーまたはグループがありません。	db2secDoesAuthIDExist db2secDoesGroupExist

表 38. セキュリティー・プラグインの戻りコード (続き)

戻りコード	定義値	意味	関連 API
-4	DB2SEC_PLUGIN_USERSTATUSNOTKNOWN	ユーザー状況が不明です。これは DB2 ではエラーとして扱われません。これは、GRANT ステートメントが、 authid がユーザーまたはオペレーティング・システム・グループのどちらを表しているか判別するために使用します。	db2secDoesAuthIDExist
-5	DB2SEC_PLUGIN_GROUPSTATUSNOTKNOWN	グループ状況が不明です。これは DB2 ではエラーとして扱われません。これは、GRANT ステートメントが、 authid がユーザーまたはオペレーティング・システム・グループのどちらを表しているか判別するために使用します。	db2secDoesGroupExist
-6	DB2SEC_PLUGIN_UID_EXPIRED	ユーザー ID が期限切れです。	db2secValidatePassword db2GetGroupsForUser db2secGenerateInitialCred
-7	DB2SEC_PLUGIN_PWD_EXPIRED	パスワードが期限切れです。	db2secValidatePassword db2GetGroupsForUser db2secGenerateInitialCred
-8	DB2SEC_PLUGIN_USER_REVOKED	ユーザーが失効しています。	db2secValidatePassword db2GetGroupsForUser
-9	DB2SEC_PLUGIN_USER_SUSPENDED	ユーザーが一時失効しています。	db2secValidatePassword db2GetGroupsForUser
-10	DB2SEC_PLUGIN_BADPWD	パスワードが無効です。	db2secValidatePassword db2secRemapUserid db2secGenerateInitialCred
-11	DB2SEC_PLUGIN_BAD_NEWPASSWORD	新規パスワードが無効です。	db2secValidatePassword db2secRemapUserid
-12	DB2SEC_PLUGIN_CHANGEPASSWORD_NOTSUPPORTED	パスワード変更はサポートされていません。	db2secValidatePassword db2secRemapUserid db2secGenerateInitialCred
-13	DB2SEC_PLUGIN_NOMEM	メモリー不足のため、プラグインがメモリーを割り振れませんでした。	すべて
-14	DB2SEC_PLUGIN_DISKERROR	プラグインがディスク・エラーを検出しました。	すべて
-15	DB2SEC_PLUGIN_NOPERM	ファイルの許可が不適切なため、プラグインがファイルにアクセスできませんでした。	すべて

表 38. セキュリティー・プラグインの戻りコード (続き)

戻りコード	定義値	意味	関連 API
-16	DB2SEC_PLUGIN_NETWORKERROR	プラグインがネットワーク・エラーを検出しました。	すべて
-17	DB2SEC_PLUGIN_CANTLOADLIBRARY	プラグインが必要なライブラリーをロードできません。	db2secGroupPluginInit db2secClientAuthPluginInit db2secServerAuthPluginInit
-18	DB2SEC_PLUGIN_CANT_OPEN_FILE	欠落ファイルや不適切なファイル許可以外の理由のために、プラグインがファイルをオープンして読み取ることができません。	すべて
-19	DB2SEC_PLUGIN_FILENOTFOUND	ファイル・システムにファイルがないために、プラグインがファイルをオープンして読み取ることができません。	すべて
-20	DB2SEC_PLUGIN_CONNECTION_DISALLOWED	接続できるデータベース、または特定のデータベースに接続できない TCP/IP アドレスについての制約事項のために、プラグインが接続を拒否しています。	すべてのサーバー・サイドのプラグイン API。
-21	DB2SEC_PLUGIN_NO_CRED	GSS API プラグインのみ: 初期クライアント証明書がありません。	db2secGetDefaultLoginContext db2secServerAuthPluginInit
-22	DB2SEC_PLUGIN_CRED_EXPIRED	GSS API プラグインのみ: クライアント証明書が期限切れです。	db2secGetDefaultLoginContext db2secServerAuthPluginInit
-23	DB2SEC_PLUGIN_BAD_PRINCIPAL_NAME	GSS API プラグインのみ: プリンシパル名が無効です。	db2secProcessServer PrincipalName
-24	DB2SEC_PLUGIN_NO_CON_DETAILS	この戻りコードは、db2secGetConDetails コールバック (例えば、DB2 からプラグインへの) によって戻され、 DB2 がクライアントの TCP/IP アドレスを判別できないことを示します。	db2secGetConDetails
-25	DB2SEC_PLUGIN_BAD_INPUT_PARAMETERS	プラグイン API を呼び出すとき、いくつかのパラメーターが無効か、または欠落しています。	すべて
-26	DB2SEC_PLUGIN_INCOMPATIBLE_VER	プラグインによって報告された API のバージョンに、DB2 との互換性がありません。	db2secGroupPluginInit db2secClientAuthPluginInit db2secServerAuthPluginInit
-27	DB2SEC_PLUGIN_PROCESS_LIMIT	プラグインが新規プロセスを作成するために、十分なリソースを使用できません。	すべて

表 38. セキュリティー・プラグインの戻りコード (続き)

戻りコード	定義値	意味	関連 API
-28	DB2SEC_PLUGIN_NO_LICENSES	プラグインがユーザー・ライセンスの問題を検出しました。基礎メカニズムのライセンスが限界に達している可能性があります。	すべて
-29	DB2SEC_PLUGIN_ROOT_NEEDED	プラグインは、root 特権が必要なアプリケーションの実行を試行しています。	すべて
-30	DB2SEC_PLUGIN_UNEXPECTED_SYSTEM_ERROR	プラグインで想定外のシステム・エラーが発生しました。現行システム構成がサポートされていない可能性があります。	すべて

セキュリティー・プラグインのエラー・メッセージ処理

セキュリティー・プラグイン API でエラーが発生すると、 API は errormsg フィールドに ASCII テキスト・ストリングを戻して、戻りコードよりも具体的な問題の説明を提示することがあります。

例えば、errormsg ストリングに、 "File /home/db2inst1/mypasswd.txt does not exist." などのメッセージが含められます。 DB2 はこのストリングをまるごと DB2 管理通知ログに書き込み、さらに、短縮版をいくつかの SQL メッセージにトークンとして組み込みます。 SQL メッセージ内のトークンは限られた長さにしかできないため、これらのメッセージは短くし、これらのメッセージの重要な変数の部分がストリングの先頭に来るようにしてください。デバッグに役立てるため、エラー・メッセージにはセキュリティー・プラグインの名前を追加することを考慮してください。

パスワード期限切れエラーなどの緊急でないエラーに関しては、errormsg ストリングは、 DIAGLEVEL データベース・マネージャー構成パラメーターに 4 が設定されている場合にのみダンプされます。

これらのエラー・メッセージ用のメモリーは、セキュリティー・プラグインによって割り振られる必要があります。したがって、プラグインは、このメモリーを解放するための API である db2secFreeErrormsg を備えていなければなりません。

errormsg フィールドは、 API がゼロ以外の値を戻した場合にのみ DB2 によってチェックされます。そのため、プラグインは、エラーがない場合は、この戻りエラー・メッセージ用のメモリーを割り振るべきではありません。

初期化時には、メッセージ・ロギング関数ポインター logMessage_fn が、グループ、クライアント、およびサーバーのプラグインに渡されます。プラグインはこの関数を使用してデバッグ情報を **db2diag** ログ・ファイルに記録できます。例えば、以下のようにします。

```
// Log an message indicate init successful
(*(logMessage_fn))(DB2SEC_LOG_CRITICAL,
                   "db2secGroupPluginInit successful",
                    strlen("db2secGroupPluginInit successful"));
```

db2secLogMessage 関数の各パラメーターについて詳しくは、各プラグイン・タイプの初期化 API を参照してください。

セキュリティー・プラグイン API の呼び出し順序

DB2 データベース・マネージャーがセキュリティー・プラグイン API を呼び出す順序は、セキュリティー・プラグイン API が呼び出されるシナリオによって異なります。

DB2 データベース・マネージャーがセキュリティー・プラグイン API を呼び出す主なシナリオを以下に示します。

- クライアントでのデータベース接続 (暗黙的および明示的)
  - CLIENT
  - サーバー・ベース (SERVER、SERVER_ENCRYPT、DATA_ENCRYPT)
  - GSSAPI および Kerberos
- クライアント、サーバー、またはゲートウェイでのローカル許可
- サーバーでのデータベース接続
- サーバーでの GRANT ステートメント
- サーバーで許可 ID が所属するグループのリストを取得する

注: DB2 データベース・サーバーは、ローカル許可が必要な **db2start**、**db2stop**、および **db2trc** などのデータベース・アクションを、クライアント・アプリケーションと同様に取り扱います。

DB2 データベース・マネージャーがセキュリティー・プラグイン API を呼び出す順序は、これらの各操作ごとに異なります。これらの各シナリオにおいて、DB2 データベース・マネージャーが呼び出す API の順序を以下に示します。

CLIENT - 暗黙的

ユーザー構成認証タイプが CLIENT の場合、DB2 クライアント・アプリケーションは以下のセキュリティー・プラグイン API を呼び出します。

- db2secGetDefaultLoginContext();
- db2secValidatePassword();
- db2secFreetoken();

暗黙的な認証の場合、すなわち、特定のユーザー ID やパスワードを指定せずに接続する場合は、ユーザー ID/パスワード・プラグインを使用していると、db2secValidatePassword API が呼び出されます。必要に応じ、プラグイン作成者はこの API によって暗黙的な認証を禁止することができます。

CLIENT - 明示的

明示的な認証の場合、すなわち、ユーザー ID とパスワードの両方が指定されているデータベースに接続する場合は、**authentication** データベース・マネージャー構成パラメーターが CLIENT に設定されていると、DB2 クライアント・アプリケーションは、インプリメンテーションが必要とする場合には、以下のセキュリティー・プラグイン API を複数回呼び出します。

- db2secRemapUserid();
- db2secValidatePassword();

- db2secFreeToken();

サーバー・ベース (SERVER、SERVER_ENCRYPT、DATA_ENCRYPT) - 暗黙的
暗黙的な認証の場合、クライアントとサーバーがユーザーID/パスワードの認証を折衝している場合 (例えば、サーバー側の **srvcon_auth** パラメーターが SERVER、SERVER_ENCRYPT、DATA_ENCRYPT、または DATA_ENCRYPT_CMP に設定されている場合)、クライアント・アプリケーションは以下のセキュリティー・プラグイン API を呼び出します。

- db2secGetDefaultLoginContext();
- db2secFreeToken();

サーバー・ベース (SERVER、SERVER_ENCRYPT、DATA_ENCRYPT) - 明示的
明示的な認証の場合、クライアントとサーバーがユーザーID/パスワードの認証を折衝している場合 (例えば、サーバー側の **srvcon_auth** パラメーターが SERVER、SERVER_ENCRYPT、DATA_ENCRYPT、または DATA_ENCRYPT_CMP に設定されている場合)、クライアント・アプリケーションは以下のセキュリティー・プラグイン API を呼び出します。

- db2secRemapUserid();

GSSAPI および Kerberos - 暗黙的
暗黙的な認証の場合、クライアントとサーバーが GSS-API または Kerberos 認証を折衝している場合 (例えば、サーバー側の **srvcon_auth** パラメーターが KERBEROS、KRB_SERVER_ENCRYPT、GSSPLUGIN、または GSS_SERVER_ENCRYPT に設定されている場合)、クライアント・アプリケーションは以下のセキュリティー・プラグイン API を呼び出します。
(gss_init_sec_context() を呼び出すときは、GSS_C_NO_CREDENTIAL が入力証明書として使用されます。)

- db2secGetDefaultLoginContext();
- db2secProcessServerPrincipalName();
- gss_init_sec_context();
- gss_release_buffer();
- gss_release_name();
- gss_delete_sec_context();
- db2secFreeToken();

マルチフロー GSS-API サポートを使用すると、インプリメンテーションが必要とする場合には、 gss_init_sec_context() を複数回呼び出すことができます。

GSSAPI および Kerberos - 明示的
折衝された認証タイプが GSS-API または Kerberos の場合は、クライアント・アプリケーションが GSS-API プラグイン用に、以下のセキュリティー・プラグイン API をこの順序で呼び出します。特に記述されていない場合、これらの API は暗黙的な認証と明示的な認証の両方に使用されます。

- db2secProcessServerPrincipalName();
- db2secGenerateInitialCred(); (明示的な認証の場合のみ)
- gss_init_sec_context();
- gss_release_buffer ();

- gss_release_name();
- gss_release_cred();
- db2secFreeInitInfo();
- gss_delete_sec_context();
- db2secFreeToken();

サーバーから相互認証トークンが戻され、インプリメンテーションが必要とする場合には、API gss_init_sec_context() が複数回呼び出されることがあります。

クライアント、サーバー、またはゲートウェイでのローカル許可

ローカル許可の場合は、使用される DB2 コマンドが、以下のセキュリティー・プラグイン API を呼び出します。

- db2secGetDefaultLoginContext();
- db2secGetGroupsForUser();
- db2secFreeToken();
- db2secFreeGroupList();

これらの API が、ユーザー ID/パスワードと GSS-API の両方の認証メカニズム用に呼び出されます。

サーバーでのデータベース接続

データベース・サーバー上でのデータベース接続の場合は、DB2 エージェント・プロセスまたはスレッドが、ユーザー ID/パスワード認証メカニズム用に以下のセキュリティー・プラグイン API を呼び出します。

- db2secValidatePassword(); (**authentication** データベース構成パラメーターが CLIENT でない場合のみ)
- db2secGetAuthIDs();
- db2secGetGroupsForUser();
- db2secFreeToken();
- db2secFreeGroupList();

データベースへの CONNECT の場合は、DB2 エージェント・プロセスまたはスレッドが、GSS-API 認証メカニズム用に以下のセキュリティー・プラグイン API を呼び出します。

- gss_accept_sec_context();
- gss_release_buffer();
- db2secGetAuthIDs();
- db2secGetGroupsForUser();
- gss_delete_sec_context();
- db2secFreeGroupListMemory();

サーバーでの GRANT ステートメント

USER または GROUP キーワードを指定しない GRANT ステートメント (例えば、 "GRANT CONNECT ON DATABASE TO user1") の場合、 DB2 エージェント・プロセスは user1 がユーザー、グループ、またはその両方のいず

れであるかを判別できなければなりません。そのため、DB2 エージェント・プロセスまたはスレッドは以下のセキュリティー・プラグイン API を呼び出します。

- db2secDoesGroupExist();
- db2secDoesAuthIDExist();

サーバーで authid が所属するグループのリストを取得する

データベース・サーバーで、許可 ID が所属するグループのリストを取得する必要がある場合、DB2 エージェント・プロセスまたはスレッドは以下のセキュリティー・プラグイン API を、許可 ID のみを入力として呼び出します。

- db2secGetGroupsForUser();

他のセキュリティー・プラグインからのトークンはありません。

第 9 章 セキュリティー・プラグイン API

ユーザーが DB2 データベース・システムの認証およびグループ・メンバーシップの検索の動作をカスタマイズできるように、既存のプラグイン・モジュールの変更や、新規セキュリティー・プラグイン・モジュールの作成の際に使用できる API が DB2 データベース・システムに用意されています。

セキュリティー・プラグイン・モジュールを作成するときは、DB2 データベース・マネージャーが呼び出す標準の認証またはグループ・メンバーシップの検索関数をインプリメントする必要があります。使用できる 3 つのタイプのプラグイン・モジュールに関してインプリメントする必要がある機能は、以下のとおりです。

グループ検索
特定のユーザーのグループ・メンバーシップ情報を検索し、指定されたストリングが有効なグループ名を表しているかどうかを判別します。

ユーザー ID/パスワード認証
この認証は、デフォルトのセキュリティー・コンテキストを識別し (クライアントのみ)、パスワードを確認して必要があれば変更し、指定されたストリングが有効なユーザーを表しているかどうか判別し (サーバーのみ)、クライアントで規定されているユーザー ID またはパスワードをサーバーへの送信の前に変更し (クライアントのみ)、指定されたユーザーに関連付けられた DB2 許可 ID を戻します。

GSS-API 認証
この認証は、必要な GSS-API 関数をインプリメントし、デフォルトのセキュリティー・コンテキストを識別し (クライアント・サイドのみ)、ユーザー ID およびパスワードを基に初期証明書を生成し、必要があればパスワードを変更し (クライアント・サイドのみ)、セキュリティー・チケットを作成して受け入れ、指定された GSS-API セキュリティー・コンテキストに関連付けられた DB2 許可 ID を戻します。

以下のリストは、プラグイン API の説明に使用される用語の定義を示しています。

プラグイン
DB2 が、ユーザー作成の認証またはグループ・メンバーシップの検索関数にアクセスするためにロードする動的にロード可能なライブラリー。

暗黙的な認証
ユーザー ID またはパスワードが指定されないデータベースへの接続。

明示的な認証
ユーザー ID とパスワードの両方が指定されるデータベースへの接続。

Authid データベース内での権限および特権が付与された個人またはグループを表す内部 ID。内部では、DB2 authid は大文字に変換されます。これは、8 文字以上です (8 文字になるようブランクが埋め込まれます)。現在のところ、DB2 は、7 ビット ASCII で表記できる authid、ユーザー ID、パスワード、グループ名、ネーム・スペース、およびドメイン名を必要とします。

ローカル許可
許可をインプリメントしているサーバーまたはクライアントでのローカルな許可です。これは、データベース・マネージャーの開始と停止、DB2 トレースのオン/オフ、データベース・マネージャー構成の更新などのアクション (データベース接続以外のアクション) を実行する権限がユーザーにあるかどうかを検査します。

ネーム・スペース
ユーザーの集合またはグループ。この中で個々のユーザー ID はユニークでなければなりません。一般的な例としては、Windows ドメインと Kerberos レルムがあります。例えば、Windows ドメイン "usa.company.com" では、すべてのユーザー名がユニークでなければなりません。例えば、"user1@usa.company.com" などとなります。他のドメインにある同一のユーザー ID (例えば、"user1@canada.company.com") は、別のユーザーを表します。完全修飾ユーザー ID には、ユーザー ID とネーム・スペースのペア (例えば "user@domain.name" または "domain¥user") が含まれます。

入力 DB2 が、セキュリティー・プラグイン API パラメーターに値を入力することを示します。

出力 セキュリティー・プラグイン API が API パラメーターの値を指定することを示します。

グループ検索プラグイン用の API

グループ検索プラグイン・モジュール用には、以下の API をインプリメントする必要があります。

- db2secGroupPluginInit

  注: db2secGroupPluginInit API は、以下のプロトタイプを持つ API を指すポインター *logMessage_fn を入力としてとります。

  ```
  SQL_API_RC (SQL_API_FN db2secLogMessage)
  (
  db2int32 level,
  void    *data,
  db2int32 length
  );
  ```

  db2secLogMessage API により、プラグインはデバッグまたは通知の目的で、メッセージを **db2diag** ログ・ファイルに記録することができます。この API は DB2 データベース・システムによって提供されるため、インプリメントする必要はありません。
- db2secPluginTerm
- db2secGetGroupsForUser
- db2secDoesGroupExist
- db2secFreeGroupListMemory
- db2secFreeErrormsg
- 外部で解決できなければならない唯一の API は、db2secGroupPluginInit です。この API は、void * パラメーターをとり、それは以下のタイプにキャストする必要があります。

```
typedef struct db2secGroupFunctions_1
{
db2int32 version;
db2int32 plugintype;
SQL_API_RC (SQL_API_FN * db2secGetGroupsForUser)
(
const char *authid,
db2int32    authidlen,
const char *userid,
db2int32    useridlen,
const char *usernamespace,
db2int32    usernamespacelen,
db2int32    usernamespacetype,
const char *dbname,
db2int32    dbnamelen,
const void *token,
db2int32    tokentype,
db2int32    location,
const char *authpluginname,
db2int32    authpluginnamelen,
void      **grouplist,
db2int32   *numgroups,
char      **errormsg,
db2int32   *errormsglen
);

SQL_API_RC (SQL_API_FN * db2secDoesGroupExist)
(
const char *groupname,
db2int32    groupnamelen,
char      **errormsg,
db2int32   *errormsglen
);

SQL_API_RC (SQL_API_FN * db2secFreeGroupListMemory)
(
void     *ptr,
char    **errormsg,
db2int32 *errormsglen
);

SQL_API_RC (SQL_API_FN * db2secFreeErrormsg)
(
char *msgtobefree
);

SQL_API_RC (SQL_API_FN * db2secPluginTerm)
(
char    **errormsg,
db2int32 *errormsglen
);

} db2secGroupFunctions_1;
```

db2secGroupPluginInit API は、外部で使用できる残りの関数のアドレスを割り当てます。

注: _1 はこれが API のバージョン 1 に対応する構造であることを示します。後続のインターフェース・バージョンの拡張子は _2、_3 というようになります。

db2secDoesGroupExist API - グループの存在のチェック

authid がグループを表すかどうかを判断します。

groupname が存在する場合、API は、正常に完了したことを示すために、値 DB2SEC_PLUGIN_OK を返せる必要があります。グループ名が有効でない場合は、値 DB2SEC_PLUGIN_INVALIDUSERORGROUP も戻されなければなりません。入力が有効なグループかどうか判別できない場合は、API が値 DB2SEC_PLUGIN_GROUPSTATUSNOTKNOWN を戻すこともできます。無効なグループ (DB2SEC_PLUGIN_INVALIDUSERORGROUP) や不明なグループ (DB2SEC_PLUGIN_GROUPSTATUSNOTKNOWN) の値が戻される場合、 DB2 for Linux, UNIX, and Windows は USER キーワードおよび GROUP キーワードのない GRANT ステートメントを発行するときに、 authid がグループかユーザーかを判別できない可能性があり、その結果 SQLCODE -569、SQLSTATE 56092 のエラーがユーザーに戻されます。

API およびデータ構造構文

```
SQL_API_RC ( SQL_API_FN *db2secDoesGroupExist)
                      ( const char *groupname,
                        db2int32 groupnamelen,
                        char      **errormsg,
                        db2int32 *errormsglen );
```

db2secDoesGroupExist API パラメーター

groupname
入力。末尾ブランクなしの大文字の authid。

groupnamelen
入力。 **groupname** パラメーター値のバイト単位の長さ。

errormsg
出力。db2secDoesGroupExist API が正常に実行されない場合にこのパラメーターで戻されることのある、プラグインによって割り振られた ASCII エラー・メッセージ・ストリングのアドレスを指すポインター。

errormsglen
出力。**errormsg** パラメーターのエラー・メッセージ・ストリングのバイト単位の長さを示す整数を指すポインター。

db2secFreeErrormsg API - エラー・メッセージのメモリーの解放

直前の API 呼び出しのエラー・メッセージを保持するために使用されているメモリーを解放します。これは、エラー・メッセージを一緒に戻さない唯一の API です。この API がエラーを戻す場合、DB2 はそれをログに記録して続行します。

API およびデータ構造構文

```
SQL_API_RC ( SQL_API_FN *db2secFreeErrormsg)
                      ( char *errormsg );
```

db2secFreeErrormsg API パラメーター

errormsg
入力。以前の API 呼び出しで割り振られたエラー・メッセージを指すポインター。

db2secFreeGroupListMemory API - グループ・リストのメモリーの解放

直前の db2secGetGroupsForUser API の呼び出しのグループのリストを保持するのに使用されているメモリーを解放します。

API およびデータ構造構文

```
SQL_API_RC ( SQL_API_FN *db2secFreeGroupListMemory)
                      ( void *ptr,
                        char      **errormsg,
                        db2int32 *errormsglen );
```

db2secFreeGroupListMemory API パラメーター

ptr 入力。解放されるメモリーを指すポインター。

errormsg
出力。db2secFreeGroupListMemory API が正常に実行されない場合にこのパラメーターで戻されることのある、プラグインによって割り振られた ASCII エラー・メッセージ・ストリングのアドレスを指すポインター。

errormsglen
出力。**errormsg** パラメーターのエラー・メッセージ・ストリングのバイト単位の長さを示す整数を指すポインター。

db2secGetGroupsForUser API - ユーザーのグループのリストの取得

ユーザーが所属するグループのリストを戻します。

API およびデータ構造構文

```
SQL_API_RC ( SQL_API_FN *db2secGetGroupsForUser)
                      ( const char *authid,
                        db2int32 authidlen,
                        const char *userid,
                        db2int32 useridlen,
                        const char *usernamespace,
                        db2int32 usernamespacelen,
                        db2int32 usernamespacetype,
                        const char *dbname,
                        db2int32 dbnamelen,
                        void *token,
                        db2int32 tokentype,
                        db2int32 location,
                        const char *authpluginname,
                        db2int32 authpluginnamelen,
                        void      **grouplist,
                        db2int32 *numgroups,
                        char      **errormsg,
                        db2int32 *errormsglen );
```

db2secGetGroupsForUser API パラメーター

authid 入力。このパラメーター値は SQL authid です。これは、その値が DB2 for Linux, UNIX, and Windows により大文字ストリングに変換され、末尾ブランクは付かないという意味です。 DB2 for Linux, UNIX, and Windows は常に、**authid** パラメーターに対して非ヌル値を提供します。 API は、他の入力パラメーターに関係なく、 authid が所属するグループのリストを戻せ

なければなりません。これが判別できない場合は、短縮されたリストまたは空のリストを戻しても差し支えありません。

ユーザーが存在しない場合、この API は戻りコード DB2SEC_PLUGIN_BADUSER を戻す必要があります。 authid には関連するグループがなくても差し支えないため、 DB2 for Linux, UNIX, and Windows は存在しないユーザーのケースをエラーとして扱いません。これには、db2secGetAuthids API がオペレーティング・システムに存在しない authid を戻す可能性があります。この authid にはグループが関連付けられていませんが、それでもこれには直接特権を割り当てることができます。

API がその authid を使用するだけでは完全なグループのリストを戻せない場合、グループ・サポートに関連した特定の SQL 関数になんらかの制限が生じる可能性があります。考えられる問題シナリオのリストについて詳しくは、このトピックの「使用上の注意」セクションを参照してください。

authidlen
入力。 **authid** パラメーター値のバイト単位の長さ。DB2 データベース・マネージャーは常に、**authidlen** パラメーターに対してゼロ以外の値を提供します。

userid 入力。これは **authid** に対応するユーザー ID です。非接続のシナリオで、サーバー上でこの API が呼び出されたときは、DB2 for Linux, UNIX, and Windows はこのパラメーターに値を入れません。

useridlen
入力。 **userid** パラメーター値のバイト単位の長さ。

usernamespace
入力。取得されたユーザー ID が属するネーム・スペース。ユーザー ID が使用できない場合、DB2 データベース・マネージャーはこのパラメーターに値を入れません。

usernamespacelen
入力。 **usernamespace** パラメーター値のバイト単位の長さ。

usernamespacetype
入力。ネーム・スペースのタイプ。 **usernamespacetype** パラメーターの有効な値 (db2secPlugin.h で定義されている) は以下のとおりです。

- DB2SEC_NAMESPACE_SAM_COMPATIBLE は domain¥myname などのユーザー名スタイルに対応します。
- DB2SEC_NAMESPACE_USER_PRINCIPAL は myname@domain.ibm.com などのユーザー名スタイルに対応します。

現在のところ、DB2 データベース・システムは値 DB2SEC_NAMESPACE_SAM_COMPATIBLE しかサポートしていません。ユーザー ID がない場合、**usernamespacetype** パラメーターの値は DB2SEC_USER_NAMESPACE_UNDEFINED (db2secPlugin.h で定義された) に設定されます。

dbname
入力。接続先のデータベースの名前。このパラメーターは、非接続シナリオでは NULL にすることができます。

dbnamelen

入力。 **dbname** パラメーター値のバイト単位の長さ。非接続シナリオでは、**dbname** パラメーターが NULL の場合、このパラメーターは 0 に設定されます。

token 入力。認証プラグインによって提供されるデータを指すポインター。これは DB2 for Linux, UNIX, and Windows では使用されません。これを使用することにより、プラグイン作成者はユーザーおよびグループ情報を調整することができるようになります。このパラメーターは、必ずしもすべての事例で使用できない可能性があり (例えば、非接続シナリオで)、その場合のパラメーターの値は NULL になります。使用されている認証プラグインが GSS-API ベースの場合、このトークンには GSS-API コンテキスト・ハンドル (gss_ctx_id_t) が設定されます。

tokentype

入力。認証プラグインによって提供されるデータのタイプを示します。使用されている認証プラグインが GSS-API ベースの場合、このトークンには GSS-API コンテキスト・ハンドル (gss_ctx_id_t) が設定されます。使用されている認証プラグインがユーザー ID/パスワード・ベースの場合、これは汎用タイプになります。 **tokentype** パラメーターの有効な値 (db2secPlugin.h で定義されている) は以下のとおりです。

- DB2SEC_GENERIC: トークンがユーザー ID/パスワード・ベースのプラグインからのものであることを示します。
- DB2SEC_GSSAPI_CTX_HANDLE: トークンが GSS-API (Kerberos を含む) ベースのプラグインからのものであることを示します。

location

入力。 DB2 for Linux, UNIX, and Windows がクライアント・サイドとサーバー・サイドのどちらでこの API を呼び出すかを示します。 location パラメーターの有効な値 (db2secPlugin.h で定義されている) は以下のとおりです。

- DB2SEC_SERVER_SIDE: API はデータベース・サーバーで呼び出されます。
- DB2SEC_CLIENT_SIDE: API はクライアントで呼び出されます。

authpluginname

入力。トークンのデータを提供した認証プラグインの名前。db2secGetGroupsForUser API は、正しいグループ・メンバーシップを判別するためにこの情報を使用することがあります。 **authid** が認証されない場合 (例えば、**authid** が現行接続ユーザーと一致しない場合) には、このパラメーターには DB2 for Linux, UNIX, and Windows によって値が入力されないことがあります。

authpluginnamelen

入力。 **authpluginname** パラメーター値のバイト単位の長さ。

grouplist

出力。ユーザーが所属するグループのリスト。グループのリストは、連結された varchar が含まれている、プラグインによって割り振られたメモリーのセクションを指すポインターとして返される必要があります (varchar は、最初のバイトが後続のバイト数を示す文字配列です)。長さは unsigned char (1 バイト) であり、このためグループ名の最大長は 255 文字までに制限さ

れます。例えば、「¥006GROUP1¥007MYGROUP¥008MYGROUP3」などです。各グループ名は、有効な DB2 authid でなければなりません。この配列のメモリーは、プラグインによって割り振られる必要があります。したがって、プラグインは、 DB2 for Linux, UNIX, and Windows がメモリーを解放するために呼び出す db2secFreeGroupListMemory API などの API を備えている必要があります。

numgroups
出力。**grouplist** パラメーターに含まれるグループの数。

errormsg
出力。db2secGetGroupsForUser API が正常に実行されない場合にこのパラメーターで戻されることのある、プラグインによって割り振られた ASCII エラー・メッセージ・ストリングのアドレスを指すポインター。

errormsglen
出力。**errormsg** パラメーターのエラー・メッセージ・ストリングのバイト単位の長さを示す整数を指すポインター。

使用上の注意

以下のリストは、この API によって DB2 for Linux, UNIX, and Windows に不完全なグループのリストが返された場合に、問題が生じる可能性のあるシナリオについて説明しています。

- CREATE SCHEMA ステートメントで代替許可が提供される。 CREATE SCHEMA ステートメント内にネストされた CREATE ステートメントがある場合、 AUTHORIZATION NAME パラメーターに対してグループ検索が実行されます。
- MPP 環境での jar ファイルの処理。 MPP 環境では、jar 処理要求が、セッション authid とともにコーディネーター・ノードから送信されます。カタログ・ノードは要求を受信すると、セッション authid (jar 処理要求を実行するユーザー) の特権に基づいて jar ファイルを処理します。
  - jar ファイルのインストール。セッション authid は、DBADM、または CREATEIN のいずれかの (jar スキーマに対する暗黙的または明示的な) 権限を有している必要があります。セッション authid の含まれるグループに対しては前述の権限が付与されているが、セッション authid に明示的には付与されていない場合は、操作は失敗します。
  - jar ファイルの除去。セッション authid は、DBADM、または DROPIN のいずれかの (jar スキーマに対する暗黙的または明示的な) 権限を有しているか、jar ファイルの定義者である必要があります。セッション authid の含まれるグループに対しては前述の権限が付与されているが、セッション authid に明示的には付与されておらず、セッション authid が Jar ファイルの定義者でもない場合は、操作は失敗します。
  - jar ファイルの置き換え。これは、jar ファイルを除去した後に、jar ファイルをインストールするのと同じことです。前述のシナリオの両方が当てはまります。
- SET SESSION_USER ステートメントが発行される場合。その後の DB2 操作は、このステートメントで指定された authid のコンテキストの下で実行されま

す。必要な特権が SESSION_USER のグループのいずれかによって所有されているものの、 SESSION_USER authid に明示的に付与されてはいない場合、それらの操作は失敗します。

db2secGroupPluginInit API - グループ・プラグインの初期化

プラグインのロードの直後に DB2 データベース・マネージャーが呼び出す、グループ検索プラグイン用の初期化 API。

API およびデータ構造構文

```
SQL_API_RC SQL_API_FN db2secGroupPluginInit
                              ( db2int32 version,
                                void *group_fns,
                                db2secLogMessage *logMessage_fn,
                                char     **errormsg,
                                db2int32 *errormsglen );
```

db2secGroupPluginInit API パラメーター

version

入力。そのプラグインをロードするインスタンスによってサポートされる API の最上位バージョン。値 DB2SEC_API_VERSION (db2secPlugin.h 内) には、DB2 データベース・マネージャーが現在サポートしている API の最新のバージョン番号が含まれます。

group_fns

出力。db2secGroupFunctions_<version_number> (group_functions_<version_number> としても知られる) 構造を指すポインター。db2secGroupFunctions_<version_number> 構造には、グループ検索プラグイン用にインプリメントされた API を指すポインターが含まれます。将来、これらの API には異なるバージョンが存在する可能性があるので (例えば、db2secGroupFunctions_<version_number>)、**group_fns** パラメーターは、プラグインがインプリメントしているバージョンに対応する db2secGroupFunctions_<version_number> 構造を指すポインターとしてキャストされます。 group_functions_<version_number> 構造の最初のパラメーターは、プラグインがインプリメントしている API のバージョンを DB2 for Linux, UNIX, and Windows に知らせます。注: DB2 のバージョンが、プラグインがインプリメントしている API のバージョンと同じかそれより大きい場合に限り、キャストが行われます。バージョン番号は、プラグインがインプリメントしている API のバージョンを表しており、**pluginType** は DB2SEC_PLUGIN_TYPE_GROUP に設定されていなければなりません。

logMessage_fn

入力。 DB2 データベース・システムによってインプリメントされる db2secLogMessage API を指すポインター。db2secGroupPluginInit API は、db2secLogMessage API を呼び出して、デバッグまたは通知の目的でメッセージを **db2diag** ログ・ファイルに記録することができます。db2secLogMessage API の最初のパラメーター (**level**) は、**db2diag** ログ・ファイルに記録される診断エラーのタイプを指定し、最後の 2 つのパラメーターはメッセージ・ストリングとその長さです。(db2secPlugin.h で定義された) db2secLogMessage API の最初のパラメーターの有効な値は以下のとおりです。

- DB2SEC_LOG_NONE: (0) ロギングなし
- DB2SEC_LOG_CRITICAL: (1) 重大エラーを検出した
- DB2SEC_LOG_ERROR: (2) エラーを検出した
- DB2SEC_LOG_WARNING: (3) 警告
- DB2SEC_LOG_INFO: (4) 通知

メッセージ・テキストが db2diag ログ・ファイルに表示されるのは、db2secLogMessage API の **level** パラメーターの値が **diaglevel** データベース・マネージャー構成パラメーターの値以下である場合だけです。そのため、例えば DB2SEC_LOG_INFO 値を使用する場合、メッセージ・テキストは **diaglevel** データベース・マネージャー構成パラメーターに 4 が設定されている場合にのみ **db2diag** ログ・ファイルに表示されます。

errormsg
出力。db2secGroupPluginInit API が正常に実行されない場合にこのパラメーターで戻されることのある、プラグインによって割り振られた ASCII エラー・メッセージ・ストリングのアドレスを指すポインター。

errormsglen
出力。**errormsg** パラメーターのエラー・メッセージ・ストリングのバイト単位の長さを示す整数を指すポインター。

db2secPluginTerm - グループ・プラグイン・リソースのクリーンアップ

グループ検索プラグインによって使用されるリソースを解放します。

この API は、DB2 データベース・マネージャーがグループ検索プラグインをアンロードする直前に呼び出されます。これは、プラグイン・ライブラリーが保持しているリソースの適切なクリーンアップを実行する、という方法でインプリメントされる必要があります。例えば、プラグインによって割り振られたメモリーを解放し、まだオープンしているファイルをクローズし、ネットワーク接続をクローズします。これらのリソースを解放するためにその記録を保持することは、プラグインが行います。この API は Windows オペレーティング・システムでは呼び出されません。

API およびデータ構造構文

```
SQL_API_RC ( SQL_API_FN *db2secPluginTerm)
                      ( char      **errormsg,
                        db2int32 *errormsglen );
```

db2secPluginTerm API parameters

errormsg
出力。db2secPluginTerm API が正常に実行されない場合にこのパラメーターで戻されることのある、プラグインによって割り振られた ASCII エラー・メッセージ・ストリングのアドレスを指すポインター。

errormsglen
出力。**errormsg** パラメーターのエラー・メッセージ・ストリングのバイト単位の長さを示す整数を指すポインター。

ユーザー ID/パスワード認証プラグインの API

ユーザー ID/パスワード・プラグイン・モジュール用には、以下のクライアント・サイド API をインプリメントする必要があります。

- db2secClientAuthPluginInit

  注: db2secClientAuthPluginInit API は、以下のプロトタイプを持つ API を指すポインター *logMessage_fn を入力としてとります。

  ```
  SQL_API_RC (SQL_API_FN db2secLogMessage)
  (
  db2int32 level,
  void    *data,
  db2int32 length
  );
  ```

  db2secLogMessage API により、プラグインはデバッグまたは通知の目的で、メッセージを **db2diag** ログ・ファイルに記録することができます。この API は DB2 データベース・システムによって提供されるため、インプリメントする必要はありません。

- db2secClientAuthPluginTerm
- db2secGenerateInitialCred (gssapi 専用)
- db2secRemapUserid (オプション)
- db2secGetDefaultLoginContext
- db2secValidatePassword
- db2secProcessServerPrincipalName (これは GSS-API 専用)
- db2secFreeToken (DLL で保持されているメモリーを解放するための関数)
- db2secFreeErrormsg
- db2secFreeInitInfo
- 外部で解決できなければならない唯一の API は、db2secClientAuthPluginInit です。この API は void * パラメーターをとり、それは以下のいずれかにキャストする必要があります。

  ```
  typedef struct db2secUseridPasswordClientAuthFunctions_1
  {
  db2int32 version;
  db2int32 plugintype;

  SQL_API_RC (SQL_API_FN * db2secGetDefaultLoginContext)
  (
  char        authid[DB2SEC_MAX_AUTHID_LENGTH],
  db2int32   *authidlen,
  char        userid[DB2SEC_MAX_USERID_LENGTH],
  db2int32   *useridlen,
  db2int32    useridtype,
  char        usernamespace[DB2SEC_MAX_USERNAMESPACE_LENGTH],
  db2int32   *usernamespacelen,
  db2int32   *usernamespacetype,
  const char *dbname,
  db2int32    dbnamelen,
  void      **token,
  char      **errormsg,
  db2int32   *errormsglen
  );
  /* Optional */
  SQL_API_RC (SQL_API_FN * db2secRemapUserid)
  ```

```
(
char        userid[DB2SEC_MAX_USERID_LENGTH],
db2int32   *useridlen,
char        usernamespace[DB2SEC_MAX_USERNAMESPACE_LENGTH],
db2int32   *usernamespacelen,
db2int32   *usernamespacetype,
char        password[DB2SEC_MAX_PASSWORD_LENGTH],
db2int32   *passwordlen,
char        newpassword[DB2SEC_MAX_PASSWORD_LENGTH],
db2int32   *newpasswordlen,
const char *dbname,
db2int32    dbnamelen,
char      **errormsg,
db2int32   *errormsglen
);

SQL_API_RC (SQL_API_FN * db2secValidatePassword)
(
const char *userid,
db2int32    useridlen,
const char *usernamespace,
db2int32    usernamespacelen,
db2int32    usernamespacetype,
const char *password,
db2int32    passwordlen,
const char *newpassword,
db2int32    newpasswordlen,
const char *dbname,
db2int32    dbnamelen,
db2Uint32   connection_details,
void      **token,
char      **errormsg,
db2int32   *errormsglen
);

SQL_API_RC (SQL_API_FN * db2secFreeToken)
(
void     **token,
char     **errormsg,
db2int32  *errormsglen
);

SQL_API_RC (SQL_API_FN * db2secFreeErrormsg)
(
char *errormsg
);

SQL_API_RC (SQL_API_FN * db2secClientAuthPluginTerm)
(
char     **errormsg,
db2int32  *errormsglen
);
}
```

または

```
typedef struct db2secGssapiClientAuthFunctions_1
{
db2int32 version;
db2int32 plugintype;

SQL_API_RC (SQL_API_FN * db2secGetDefaultLoginContext)
(
char        authid[DB2SEC_MAX_AUTHID_LENGTH],
db2int32   *authidlen,
char        userid[DB2SEC_MAX_USERID_LENGTH],
db2int32   *useridlen,
```

```
db2int32    useridtype,
char        usernamespace[DB2SEC_MAX_USERNAMESPACE_LENGTH],
db2int32   *usernamespacelen,
db2int32   *usernamespacetype,
const char *dbname,
db2int32    dbnamelen,
void      **token,
char      **errormsg,
db2int32   *errormsglen
);

SQL_API_RC (SQL_API_FN * db2secProcessServerPrincipalName)
(
const void *data,
gss_name_t *gssName,
char      **errormsg,
db2int32   *errormsglen
);

SQL_API_RC (SQL_API_FN * db2secGenerateInitialCred)
(
const char    *userid,
db2int32       useridlen,
const char    *usernamespace,
db2int32       usernamespacelen,
db2int32       usernamespacetype,
const char    *password,
db2int32       passwordlen,
const char    *newpassword,
db2int32       newpasswordlen,
const char    *dbname,
db2int32       dbnamelen,
gss_cred_id_t *pGSSCredHandle,
void         **initInfo,
char         **errormsg,
db2int32      *errormsglen
);

SQL_API_RC (SQL_API_FN * db2secFreeToken)
(
void      *token,
char     **errormsg,
db2int32  *errormsglen
);

SQL_API_RC (SQL_API_FN * db2secFreeErrormsg)
(
char *errormsg
);


SQL_API_RC (SQL_API_FN * db2secFreeInitInfo)
(
void      *initInfo,
char     **errormsg,
db2int32  *errormsglen
);

SQL_API_RC (SQL_API_FN * db2secClientAuthPluginTerm)
(
char     **errormsg,
db2int32  *errormsglen
);

/* GSS-API specific functions -- refer to db2secPlugin.h
   for parameter list*/
```

```
  OM_uint32 (SQL_API_FN * gss_init_sec_context )(<parameter list>);
  OM_uint32 (SQL_API_FN * gss_delete_sec_context )(<parameter list>);
  OM_uint32 (SQL_API_FN * gss_display_status )(<parameter list>);
  OM_uint32 (SQL_API_FN * gss_release_buffer )(<parameter list>);
  OM_uint32 (SQL_API_FN * gss_release_cred )(<parameter list>);
  OM_uint32 (SQL_API_FN * gss_release_name )(<parameter list>);
  }
```

ユーザー ID/パスワード・プラグインを作成する場合は、db2secUseridPasswordClientAuthFunctions_1 構造を使用する必要があります。GSS-API (Kerberos を含む) プラグインを作成する場合は、db2secGssapiClientAuthFunctions_1 構造を使用する必要があります。

ユーザー ID/パスワード・プラグイン・ライブラリー用には、以下のサーバー・サイド API をインプリメントする必要があります。

- db2secServerAuthPluginInit

  db2secServerAuthPluginInit API は、以下のプロトタイプを持つ、db2secLogMessage API を指すポインター *logMessage_fn、および db2secGetConDetails API を指すポインター *getConDetails_fn を入力としてとります。

```
SQL_API_RC (SQL_API_FN db2secLogMessage)
(
db2int32 level,
void    *data,
db2int32 length
);

SQL_API_RC (SQL_API_FN db2secGetConDetails)
(
db2int32    conDetailsVersion,
const void *pConDetails
);
```

  db2secLogMessage API により、プラグインはデバッグまたは通知の目的で、メッセージを **db2diag** ログ・ファイルに記録することができます。db2secGetConDetails API により、プラグインは、データベースに接続しようとしているクライアントに関する詳細を取得することができます。 db2secLogMessage API と db2secGetConDetails API はどちらも DB2 データベース・システムによって提供されるので、インプリメントする必要はありません。同様に、db2secGetConDetails API は、その 2 番目のパラメーター **pConDetails** として、以下の構造の 1 つを指すポインターをとります。

  db2sec_con_details_1:

```
typedef struct db2sec_con_details_1
{
  db2int32  clientProtocol;
  db2Uint32 clientIPAddress;
  db2Uint32 connect_info_bitmap;
  db2int32  dbnameLen;
  char      dbname[DB2SEC_MAX_DBNAME_LENGTH + 1];
} db2sec_con_details_1;
```

  db2sec_con_details_2:

```
typedef struct db2sec_con_details_2
{
  db2int32  clientProtocol;     /* See SQL_PROTOCOL_ in sqlenv.h */
```

```
  db2Uint32 clientIPAddress;    /* Set if protocol is TCPIP4     */
  db2Uint32 connect_info_bitmap;
  db2int32  dbnameLen;
  char dbname[DB2SEC_MAX_DBNAME_LENGTH + 1];
  db2Uint32 clientIP6Address[4];/* Set if protocol is TCPIP6     */
} db2sec_con_details_2;
```

db2sec_con_details_3:

```
typedef struct db2sec_con_details_3
{
  db2int32 clientProtocol; /* See SQL_PROTOCOL_ in sqlenv.h */
  db2Uint32 clientIPAddress; /* Set if protocol is TCPIP4 */
  db2Uint32 connect_info_bitmap;
  db2int32 dbnameLen;
  char dbname[DB2SEC_MAX_DBNAME_LENGTH + 1];
  db2Uint32 clientIP6Address[4];/* Set if protocol is TCPIP6 */
  db2Uint32 clientPlatform; /* SQLM_PLATFORM_* from sqlmon.h */
  db2Uint32 _reserved[16];
} db2sec_con_details_3;
```

conDetailsVersion の考えられる値は、API のバージョンを表す DB2SEC_CON_DETAILS_VERSION_1、DB2SEC_CON_DETAILS_VERSION_2、および DB2SEC_CON_DETAILS_VERSION_3 です。

注: db2sec_con_details_1、db2sec_con_details_2、または db2sec_con_details_3 を使用しているときには、以下の事柄を考慮してください。

- db2sec_con_details_1 構造と DB2SEC_CON_DETAILS_VERSION_1 値を使用している既存のプラグインは、db2GetConDetails API を呼び出すと、バージョン 8.2 で行っていたように作業を続けます。この API が IPv4 プラットフォームで呼び出される場合、クライアント IP アドレスが構造の **clientIPAddress** フィールドで戻されます。この API が IPv6 プラットフォームで呼び出される場合、値 0 が **clientIPAddress** フィールドで戻されます。クライアント IP アドレスを IPv6 プラットフォームで取り出すには、db2sec_con_details_2 構造と DB2SEC_CON_DETAILS_VERSION_2 値を使用するように、または db2sec_con_details_3 構造と DB2SEC_CON_DETAILS_VERSION_3 値を使用するようにセキュリティー・プラグイン・コードを変更する必要があります。
- 新規プラグインは db2sec_con_details_3 構造と DB2SEC_CON_DETAILS_VERSION_3 値を使用する必要があります。 db2secGetConDetails API が IPv4 プラットフォームで呼び出される場合、クライアント IP アドレスが db2sec_con_details_3 構造の **clientIPAddress** フィールドで戻され、この API が IPv6 プラットフォームで呼び出される場合、クライアント IP アドレスが db2sec_con_details_3 構造の **clientIP6Address** フィールドで戻されます。接続詳細構造の **clientProtocol** フィールドは、SQL_PROTOCOL_TCPIP (IPv4 で v1 の構造を持つ)、SQL_PROTOCOL_TCPIP4 (IPv4 で v2 または v3 の構造を持つ)、または SQL_PROTOCOL_TCPIP6 (IPv6 で v2 の構造を持つ) のいずれかに設定されます。
- 構造 db2sec_con_details_3 は、SQLM_PLATFORM_AIX のような sqlmon.h で定義されるプラットフォーム・タイプ定数を使用するクライアント・プラットフォーム・タイプ (通信層でレポートされる) を特定する追加フィールド (**clientPlatform**) を含んでいる点を除けば、構造 db2sec_con_details_2 と同じです。

- db2secServerAuthPluginTerm

- db2secValidatePassword
- db2secGetAuthIDs
- db2secDoesAuthIDExist
- db2secFreeToken
- db2secFreeErrormsg
- 外部で解決できなければならない唯一の API は、db2secServerAuthPluginInit です。この API は void * パラメーターをとり、それは以下のいずれかにキャストする必要があります。

```
typedef struct db2secUseridPasswordServerAuthFunctions_1
{
db2int32 version;
db2int32 plugintype;

  /* parameter lists left blank for readability
    see above for parameters */
SQL_API_RC (SQL_API_FN * db2secValidatePassword)(<parameter list>);
SQL_API_RC (SQL_API_FN * db2secGetAuthIDs)(<parameter list);
SQL_API_RC (SQL_API_FN * db2secDoesAuthIDExist)(<parameter list>);
SQL_API_RC (SQL_API_FN * db2secFreeToken)(<parameter list>);
SQL_API_RC (SQL_API_FN * db2secFreeErrormsg)(<parameter list>);
SQL_API_RC (SQL_API_FN * db2secServerAuthPluginTerm)();
} userid_password_server_auth_functions;
```

または

```
typedef struct db2secGssapiServerAuthFunctions_1
{
db2int32 version;
db2int32 plugintype;
gss_buffer_desc serverPrincipalName;
gss_cred_id_t ServerCredHandle;
SQL_API_RC (SQL_API_FN * db2secGetAuthIDs)(<parameter list);
SQL_API_RC (SQL_API_FN * db2secDoesAuthIDExist)(<parameter list>);
SQL_API_RC (SQL_API_FN * db2secFreeErrormsg)(<parameter list>);
SQL_API_RC (SQL_API_FN * db2secServerAuthPluginTerm)();

/* GSS-API specific functions
refer to db2secPlugin.h for parameter list*/
OM_uint32 (SQL_API_FN * gss_accept_sec_context )(<parameter list>);
OM_uint32 (SQL_API_FN * gss_display_name )(<parameter list>);
OM_uint32 (SQL_API_FN * gss_delete_sec_context )(<parameter list>);
OM_uint32 (SQL_API_FN * gss_display_status )(<parameter list>);
OM_uint32 (SQL_API_FN * gss_release_buffer )(<parameter list>);
OM_uint32 (SQL_API_FN * gss_release_cred )(<parameter list>);
OM_uint32 (SQL_API_FN * gss_release_name )(<parameter list>);

} gssapi_server_auth_functions;
```

ユーザー ID/パスワード・プラグインを作成する場合は、db2secUseridPasswordServerAuthFunctions_1 構造を使用する必要があります。GSS-API (Kerberos を含む) プラグインを作成する場合は、db2secGssapiServerAuthFunctions_1 構造を使用する必要があります。

db2secClientAuthPluginInit API - クライアント認証プラグインの初期化

プラグインのロードの直後に DB2 データベース・マネージャーが呼び出す、クライアント認証プラグイン用の初期化 API。

API およびデータ構造構文

```
SQL_API_RC SQL_API_FN db2secClientAuthPluginInit
                              ( db2int32 version,
                                void *client_fns,
                                db2secLogMessage *logMessage_fn,
                                char    **errormsg,
                                db2int32 *errormsglen );
```

db2secClientAuthPluginInit API パラメーター

version
入力。 DB2 データベース・マネージャーが現在サポートしている API の最大のバージョン番号。 DB2SEC_API_VERSION 値 (db2secPlugin.h 内) には、DB2 for Linux, UNIX, and Windowsが現在サポートしている API の最新のバージョン番号が含まれます。

client_fns
出力。DB2 データベース・マネージャーによって提供されるメモリーを指すポインター。このメモリーは、GSS-API 認証が使用される場合は、db2secGssapiClientAuthFunctions_<version_number> 構造 (gssapi_client_auth_functions_<version_number> としても知られる) のために提供され、ユーザー ID/パスワード認証が使用される場合は、db2secUseridPasswordClientAuthFunctions_<version_number> 構造 (userid_password_client_auth_functions_<version_number> としても知られる) のために提供されます。 db2secGssapiClientAuthFunctions_<version_number> 構造と db2secUseridPasswordClientAuthFunctions_<version_number> 構造には、GSS-API 認証プラグイン用にインプリメントされた API を指すポインターとユーザー ID/パスワード認証プラグイン用にインプリメントされた API を指すポインターが含まれています。 DB2 for Linux, UNIX, and Windows の将来のバージョンでは、異なるバージョンの API が存在している可能性があるので、**client_fns** パラメーターは、プラグインがインプリメントしているバージョンに対応する gssapi_client_auth_functions_<version_number> 構造を指すポインターとしてキャストします。

gssapi_client_auth_functions_<version_number> 構造または userid_password_client_auth_functions_<version_number> 構造の最初のパラメーターは、プラグインがインプリメントしている API のバージョンを DB2 データベース・マネージャーに知らせます。

注: DB2 のバージョンが、プラグインがインプリメントしている API のバージョンと同じかそれより大きい場合に限り、キャストが行われます。

gssapi_server_auth_functions_<version_number> または userid_password_server_auth_functions_<version_number> 構造内では、**plugintype** パスワードを DB2SEC_PLUGIN_TYPE_USERID_PASSWORD、DB2SEC_PLUGIN_TYPE_GSSAPI、または **DB2SEC_PLUGIN_TYPE_KERBEROS** のいずれかに設定する必要があります。将来のバージョンの API では、他の値も定義される可能性があります。

logMessage_fn
入力。 DB2 データベース・マネージャーによってインプリメントされる db2secLogMessage API を指すポインター。db2secClientAuthPluginInit API は、db2secLogMessage API を呼び出して、デバッグまたは通知の目的でメッセージ

を **db2diag** ログ・ファイルに記録することができます。db2secLogMessage API の最初のパラメーター (**level**) は、**db2diag** ログ・ファイルに記録される診断エラーのタイプを指定し、最後の 2 つのパラメーターはメッセージ・ストリングとその長さです。(db2secPlugin.h で定義された) db2secLogMessage API の最初のパラメーターの有効な値は以下のとおりです。

- DB2SEC_LOG_NONE (0) ロギングなし
- DB2SEC_LOG_CRITICAL (1) 重大エラーを検出した
- DB2SEC_LOG_ERROR (2) エラーを検出した
- DB2SEC_LOG_WARNING (3) 警告
- DB2SEC_LOG_INFO (4) 通知

メッセージ・テキストが **db2diag** ログ・ファイルに表示されるのは、db2secLogMessage API の 'level' パラメーターの値が **diaglevel** データベース・マネージャー構成パラメーターの値以下である場合だけです。例えば DB2SEC_LOG_INFO 値を使用する場合、メッセージ・テキストは **diaglevel** データベース・マネージャー構成パラメーターに 4 が設定されている場合にのみ **db2diag** ログ・ファイルに表示されます。

errormsg
出力。db2secClientAuthPluginInit API が正常に実行されない場合にこのパラメーターで戻されることのある、プラグインによって割り振られた ASCII エラー・メッセージ・ストリングのアドレスを指すポインター。

errormsglen
出力。**errormsg** パラメーターのエラー・メッセージ・ストリングのバイト単位の長さを示す整数を指すポインター。

db2secClientAuthPluginTerm API - クライアント認証プラグイン・リソースのクリーンアップ

クライアント認証プラグインによって使用されるリソースを解放します。

この API は、DB2 データベース・マネージャーがクライアント認証プラグインをアンロードする直前に呼び出します。これは、プラグイン・ライブラリーが保持しているリソースの適切なクリーンアップを実行する、という方法でインプリメントされる必要があります。例えば、プラグインによって割り振られたメモリーを解放し、まだオープンしているファイルをクローズし、ネットワーク接続をクローズします。これらのリソースを解放するためにその記録を保持することは、プラグインが行います。この API は Windows オペレーティング・システムでは呼び出されません。

API およびデータ構造構文

```
SQL_API_RC ( SQL_API_FN *db2secClientAuthPluginTerm)
                      ( char     **errormsg,
                        db2int32 *errormsglen);
```

db2secClientAuthPluginTerm API パラメーター

errormsg
出力。db2secClientAuthPluginTerm API が正常に実行されない場合にこのパラメ

ーターで戻されることのある、プラグインによって割り振られた ASCII エラー・メッセージ・ストリングのアドレスを指すポインター。

errormsglen
出力。**errormsg** パラメーターのエラー・メッセージ・ストリングのバイト単位の長さを示す整数を指すポインター。

db2secDoesAuthIDExist - 認証 ID の存在の検査

authid が個々のユーザーを表しているかどうか (例えば、この API がこの **authid** を外部ユーザーにマップできるかどうか) を判別します。

この API は、これが正常 (**authid** が有効) な場合は値 DB2SEC_PLUGIN_OK を、無効な場合は DB2SEC_PLUGIN_INVALID_USERORGROUP を、**authid** の存在を判別できない場合は DB2SEC_PLUGIN_USERSTATUSNOTKNOWN を戻す必要があります。

API およびデータ構造構文

```
SQL_API_RC ( SQL_API_FN *db2secDoesAuthIDExist)
                      ( const char *authid,
                        db2int32 authidlen,
                        char       **errormsg,
                        db2int32 *errormsglen );
```

db2secDoesAuthIDExist API パラメーター

authid
入力。確認する authid。これは、末尾ブランクなしの大文字になります。

authidlen
入力。 **authid** パラメーター値のバイト単位の長さ。

errormsg
出力。db2secDoesAuthIDExist API が正常に実行されない場合にこのパラメーターで戻されることのある、プラグインによって割り振られた ASCII エラー・メッセージ・ストリングのアドレスを指すポインター。

errormsglen
出力。**errormsg** パラメーターのエラー・メッセージ・ストリングの長さを示す整数を指すポインター。

db2secFreeInitInfo API - db2secGenerateInitialCred が保持しているリソースのクリーンアップ

db2secGenerateInitialCred API によって割り振られたすべてのリソースを解放します。これには、例えば、基礎メカニズム・コンテキストのハンドルや、 GSS-API 証明書キャッシュ用に作成された証明書キャッシュが含まれます。

API およびデータ構造構文

```
SQL_API_RC ( SQL_API_FN *db2secFreeInitInfo)
                      ( void *initinfo,
                        char     **errormsg,
                        db2int32 *errormsglen);
```

db2secFreeInitInfo API パラメーター

initinfo
入力。 DB2 データベース・マネージャーに認識されていないデータを指すポインター。プラグインはこのメモリーを使用して、証明書ハンドルの生成プロセスで割り振られたリソースのリストを保守できます。これらのリソースは、この API を呼び出すことによって解放されます。

errormsg
出力。db2secFreeInitInfo API が正常に実行されない場合にこのパラメーターで戻されることのある、プラグインによって割り振られた ASCII エラー・メッセージ・ストリングのアドレスを指すポインター。

errormsglen
出力。**errormsg** パラメーターのエラー・メッセージ・ストリングのバイト単位の長さを示す整数を指すポインター。

db2secFreeToken API - トークンが保持しているメモリーの解放

トークンによって保持されたメモリーを解放します。この API は、DB2 データベース・マネージャーが token パラメーターによって保持されているメモリーを必要としなくなったときに呼び出します。

API およびデータ構造構文

```
SQL_API_RC ( SQL_API_FN *db2secFreeToken)
                      ( void *token,
                        char       **errormsg,
                        db2int32 *errormsglen );
```

db2secFreeToken API パラメーター

token
入力。解放されるメモリーを指すポインター。

errormsg
出力。db2secFreeToken API が正常に実行されない場合にこのパラメーターで戻されることのある、プラグインによって割り振られた ASCII エラー・メッセージ・ストリングのアドレスを指すポインター。

errormsglen
出力。**errormsg** パラメーターのエラー・メッセージ・ストリングのバイト単位の長さを示す整数を指すポインター。

db2secGenerateInitialCred API - 初期証明書の生成

db2secGenerateInitialCred API は、渡されるユーザー ID およびパスワードを基に初期 GSS-API 証明書を取得します。

Kerberos の場合、これは発券許可証 (TGT) になります。**pGSSCredHandle** パラメーターに戻される証明書ハンドルは、 gss_init_sec_context API で使用されるハンドルであり、INITIATE 証明書か BOTH 証明書のいずれかでなければなりません。db2secGenerateInitialCred API は、ユーザー ID、そしておそらくパスワードが指定されている場合にのみ呼び出されます。それ以外の場合、DB2 データベース・マネ

ージャーは gss_init_sec_context API を呼び出すときに値 GSS_C_NO_CREDENTIAL を指定して、現行ログイン・コンテキストから取得されるデフォルト証明書が使用されることを示します。

API およびデータ構造構文

```
SQL_API_RC ( SQL_API_FN *db2secGenerateInitialCred)
                      ( const char *userid,
                        db2int32 useridlen,
                        const char *usernamespace,
                        db2int32 usernamespacelen,
                        db2int32 usernamespacetype,
                        const char *password,
                        db2int32 passwordlen,
                        const char *newpassword,
                        db2int32 newpasswordlen,
                        const char *dbname,
                        db2int32 dbnamelen,
                        gss_cred_id_t *pGSSCredHandle,
                        void          **InitInfo,
                        char          **errormsg,
                        db2int32 *errormsglen );
```

db2secGenerateInitialCred API パラメーター

userid
入力。データベース・サーバー上でパスワードが検証されるユーザー ID。

useridlen
入力。 **userid** パラメーター値のバイト単位の長さ。

usernamespace
入力。取得されたユーザー ID が属するネーム・スペース。

usernamespacelen
入力。 **usernamespace** パラメーター値のバイト単位の長さ。

usernamespacetype
入力。ネーム・スペースのタイプ。

password
入力。検証されるパスワード。

passwordlen
入力。 **password** パラメーター値のバイト単位の長さ。

newpassword
入力。パスワードが変更される場合の新規パスワード。変更が要求されない場合、**newpassword** パラメーターは NULL に設定されます。これが非ヌルの場合、API は、旧パスワードを新規パスワードに設定する前に確認する必要があります。 API は、パスワードの変更要求を受け入れる必要はありませんが、受け入れない場合は、旧パスワードを妥当性検査せずに、即時に戻り値 DB2SEC_PLUGIN_CHANGEPASSWORD_NOTSUPPORTED を返す必要があります。

newpasswordlen
入力。 **newpassword** パラメーター値のバイト単位の長さ。

dbname
入力。接続先のデータベースの名前。この API はこのパラメーターを無視しても差し支えありません。あるいは、特定のデータベースへのアクセスを、有効な

パスワードを特別に持つユーザーのみに限定する方針をとっている場合は、この関数は値 DB2SEC_PLUGIN_CONNECTION_DISALLOWED を戻すことができます。

dbnamelen
入力。 **dbname** パラメーター値のバイト単位の長さ。

pGSSCredHandle
出力。GSS-API 証明書ハンドルを指すポインター。

InitInfo
出力。DB2 for Linux, UNIX, and Windows に認識されていないデータを指すポインター。プラグインはこのメモリーを使用して、証明書ハンドルの生成プロセスで割り振られたリソースのリストを保守できます。 DB2 データベース・マネージャーは認証プロセスの最後に db2secFreeInitInfo API を呼び出し、その時点でこれらのリソースは解放されます。db2secGenerateInitialCred API は、このようなリストを保守する必要がない場合は、NULL を戻す必要があります。

errormsg
出力。db2secGenerateInitialCred API が正常に実行されない場合にこのパラメーターで戻されることのある、プラグインによって割り振られた ASCII エラー・メッセージ・ストリングのアドレスを指すポインター。

注: この API では、戻り値が無効なユーザー ID またはパスワードを示している場合には、エラー・メッセージは作成されるべきではありません。エラー・メッセージは、API の中に、この API が正しく完了することを妨げる内部エラーがある場合にのみ戻される必要があります。

errormsglen
出力。**errormsg** パラメーターのエラー・メッセージ・ストリングのバイト単位の長さを示す整数を指すポインター。

db2secGetAuthIDs API - 認証 ID の取得

認証ユーザーの SQL authid を戻します。この API は、ユーザー ID/パスワードと GSS-API の両方の認証方式において、データベース接続時に呼び出されます。

API およびデータ構造構文

```
SQL_API_RC ( SQL_API_FN *db2secGetAuthIDs)
                      ( const char *userid,
                        db2int32 useridlen,
                        const char *usernamespace,
                        db2int32 usernamespacelen,
                        db2int32 usernamespacetype,
                        const char *dbname,
                        db2int32 dbnamelen,
                        void      **token,
                        char SystemAuthID[DB2SEC_MAX_AUTHID_LENGTH],
                        db2int32 *SystemAuthIDlen,
                        char InitialSessionAuthID[DB2SEC_MAX_AUTHID_LENGTH],
                        db2int32 *InitialSessionAuthIDlen,
                        char username[DB2SEC_MAX_USERID_LENGTH],
                        db2int32 *usernamelen,
                        db2int32 *initsessionidtype,
                        char      **errormsg,
                        db2int32 *errormsglen );
```

db2secGetAuthIDs API パラメーター

userid
入力。認証ユーザー。 GSS-API 認証では、通常は使用されません。ただし、認証なしのユーザー切り替え操作を認めるトラステッド・コンテキストが定義されている場合は例外です。そのような場合は、ユーザーの切り替え要求で指定されているユーザー名がこのパラメーターで渡されます。

useridlen
入力。 **userid** パラメーター値のバイト単位の長さ。

usernamespace
入力。取得されたユーザー ID が属するネーム・スペース。

usernamespacelen
入力。 **usernamespace** パラメーター値のバイト単位の長さ。

usernamespacetype
入力。ネーム・スペース・タイプ値。現時点でサポートされる唯一のネーム・スペース・タイプ値は、DB2SEC_NAMESPACE_SAM_COMPATIBLE です (domain¥myname などのユーザー名スタイルに相当します)。

dbname
入力。接続先のデータベースの名前。 API はこれを無視しても差し支えありません。あるいはプラグインは、同一のユーザーが異なるデータベースに接続する際に別々の authid を戻すこともできます。このパラメーターは、NULL にすることができます。

dbnamelen
入力。 **dbname** パラメーター値のバイト単位の長さ。 **dbname** パラメーターが NULL の場合、このパラメーターは 0 に設定されます。

token
入力または出力。プラグインが db2secGetGroupsForUser API に渡すデータ。GSS-API の場合、これはコンテキスト・ハンドル (gss_ctx_id_t) です。通常、トークンは入力のみのパラメーターであり、この値は db2secValidatePassword から取り込まれます。認証がクライアントで行われ、そのために db2secValidatePassword API が呼び出されない場合には、これは出力パラメーターにもなります。認証なしのユーザー切り替え操作を認めるトラステッド・コンテキストが定義されている環境では、db2secGetAuthIDs API でこのトークン・パラメーターの NULL 値を受け付け、前述の **userid** 入力パラメーターと **useridlen** 入力パラメーターに基づいて、システム許可 ID を派生させることができるように設定しなければなりません。

SystemAuthID
出力。認証ユーザーの ID に対応するシステム許可。サイズは 255 バイトですが、DB2 データベース・マネージャーは現在最大 30 バイトまで使用します。

SystemAuthIDlen
出力。**SystemAuthID** パラメーター値のバイト単位の長さ。

InitialSessionAuthID
出力。この接続セッションに使用される authid。これは通常 **SystemAuthID** パラメーターと同じですが、SET SESSION AUTHORIZATION ステートメントを発

行する場合などのある特定の場合には異なることがあります。サイズは 255 バイトですが、DB2 データベース・マネージャーは現在最大 30 バイトまで使用します。

InitialSessionAuthIDlen
出力。**InitialSessionAuthID** パラメーター値のバイト単位の長さ。

username
出力。認証ユーザーと authid に対応するユーザー名。これは監査のためにのみ使用され、CONNECT ステートメントの監査記録内の「ユーザー ID」フィールドに記録されます。 API で **username** パラメーターを指定している場合、DB2 データベース・マネージャーは userid からそれをコピーします。

usernamelen
出力。**username** パラメーター値のバイト単位の長さ。

initsessionidtype
出力。**InitialSessionAuthid** パラメーターがロールか authid かを示すセッション authid タイプ。 API は、以下の値のいずれか (db2secPlugin.h で定義された) を戻さなければなりません。

- DB2SEC_ID_TYPE_AUTHID (0)
- DB2SEC_ID_TYPE_ROLE (1)

errormsg
出力。db2secGetAuthIDs API が正常に実行されない場合にこのパラメーターで戻されることのある、プラグインによって割り振られた ASCII エラー・メッセージ・ストリングのアドレスを指すポインター。

errormsglen
出力。**errormsg** パラメーターのエラー・メッセージ・ストリングのバイト単位の長さを示す整数を指すポインター。

db2secGetDefaultLoginContext API - デフォルト・ログイン・コンテキストの取得

デフォルト・ログイン・コンテキストに関連したユーザーを判別します。つまり、ユーザー ID を明示的に指定せずに (データベースに対する暗黙的な認証か、ローカル許可のいずれかで)、DB2 コマンドを呼び出すユーザーの DB2 authid を判別します。この API は、authid とユーザー ID の両方を戻さなければなりません。

API およびデータ構造構文

```
SQL_API_RC ( SQL_API_FN *db2secGetDefaultLoginContext)
                     ( char authid[DB2SEC_MAX_AUTHID_LENGTH],
                       db2int32 *authidlen,
                       char userid[DB2SEC_MAX_USERID_LENGTH],
                       db2int32 *useridlen,
                       db2int32 useridtype,
                       char usernamespace[DB2SEC_MAX_USERNAMESPACE_LENGTH],
                       db2int32 *usernamespacelen,
                       db2int32 *usernamespacetype,
                       const char *dbname,
                       db2int32 dbnamelen,
                       void      **token,
                       char      **errormsg,
                       db2int32 *errormsglen );
```

db2secGetDefaultLoginContext API パラメーター

authid
出力。authid が戻されるパラメーター。戻り値は DB2 authid の命名規則に準拠していなければなりません。そうでなければ、ユーザーは要求されたアクションの実行を許可されません。

authidlen
出力。**authid** パラメーター値のバイト単位の長さ。

userid
出力。デフォルト・ログイン・コンテキストに関連したユーザー ID を戻すパラメーター。

useridlen
出力。**userid** パラメーター値のバイト単位の長さ。

useridtype
入力。プロセスの実ユーザー ID、または有効ユーザー ID が指定されているかどうかを示します。 Windowsの場合、実ユーザー ID のみ存在します。 UNIX および Linux では、アプリケーションの uid ユーザー ID がプロセスを実行しているユーザーの ID と異なる場合、実ユーザー ID と有効ユーザー ID が異なることがあります。 **userid** パラメーターの有効な値 (db2secPlugin.h で定義されている) は以下のとおりです。

DB2SEC_PLUGIN_REAL_USER_NAME
実ユーザー ID が指定されていることを示します。

DB2SEC_PLUGIN_EFFECTIVE_USER_NAME
有効ユーザー ID が指定されていることを示します。

注: 一部のプラグイン・インプリメンテーションによっては、実ユーザー ID と有効ユーザー ID を区別しないものがあります。特に、DB2 許可 ID を設定するためにユーザーの UNIX または Linux の ID を使用しないプラグインは、この区別を無視しても支障がありません。

usernamespace
出力。ユーザー ID のネーム・スペース。

usernamespacelen
出力。**usernamespace** パラメーター値のバイト単位の長さ。**usernamespacetype** パラメーターが値 DB2SEC_NAMESPACE_SAM_COMPATIBLE (db2secPlugin.h で定義されている) に設定されていなければならないという制限のもとでは、現在サポートされる最大長は 15 バイトになります。

usernamespacetype
出力。ネーム・スペース・タイプ値。現時点でサポートされる唯一のネーム・スペース・タイプは、DB2SEC_NAMESPACE_SAM_COMPATIBLE です (domain¥myname などのユーザー名スタイルに相当します)。

dbname
入力。データベース接続のコンテキストでこの呼び出しが使用される場合に、接続先のデータベースの名前が入ります。ローカル許可アクションやインスタンス接続の場合、このパラメーターは NULL に設定されます。

dbnamelen
入力。 **dbname** パラメーター値のバイト単位の長さ。

token
出力。これはプラグインが、そのプラグインでの後の認証呼び出しや、またはグループ検索プラグインに渡す、プラグインによって割り振られたデータを指すポインターです。このデータの構造は、プラグイン作成者によって決定されます。

errormsg
出力。db2secGetDefaultLoginContext API が正常に実行されない場合にこのパラメーターで戻されることのある、プラグインによって割り振られた ASCII エラー・メッセージ・ストリングのアドレスを指すポインター。

errormsglen
出力。**errormsg** パラメーターのエラー・メッセージ・ストリングのバイト単位の長さを示す整数を指すポインター。

db2secProcessServerPrincipalName API - サーバーから戻されたサービス・プリンシパル名の処理

db2secProcessServerPrincipalName API は、サーバーから戻されたサービス・プリンシパル名を処理し、gss_init_sec_context API で使用される gss_name_t 内部形式のプリンシパル名を戻します。

db2secProcessServerPrincipalName API は、Kerberos 認証の使用時に、データベース・ディレクトリーでカタログされたサービス・プリンシパル名も処理します。通常、この変換では gss_import_name API が使用されます。コンテキストが確立されると、gss_name_t オブジェクトは gss_release_name API の呼び出しによって解放されます。 db2secProcessServerPrincipalName API は、**gssName** パラメーターが有効な GSS 名を指していれば値 **DB2SEC_PLUGIN_OK** を戻します。プリンシパル名が無効な場合は DB2SEC_PLUGIN_BAD_PRINCIPAL_NAME エラー・コードが戻されます。

API およびデータ構造構文

```
SQL_API_RC ( SQL_API_FN *db2secProcessServerPrincipalName)
                      ( const char *name,
                        db2int32 namelen,
                        gss_name_t *gssName,
                        char       **errormsg,
                        db2int32 *errormsglen );
```

db2secProcessServerPrincipalName API パラメーター

name
入力。 GSS_C_NT_USER_NAME 形式のサービス・プリンシパルのテキスト名 (例: service/host@REALM)。

namelen
入力。 **name** パラメーター値のバイト単位の長さ。

gssName
出力。GSS-API 内部形式の出力サービス・プリンシパル名を指すポインター。

errormsg
出力。db2secProcessServerPrincipalName API が正常に実行されない場合にこの

パラメーターで戻されることのある、プラグインによって割り振られた ASCII エラー・メッセージ・ストリングのアドレスを指すポインター。

errormsglen
出力。**errormsg** パラメーターのエラー・メッセージ・ストリングのバイト単位の長さを示す整数を指すポインター。

db2secRemapUserid API - ユーザー ID およびパスワードの再マップ

この API は DB2 データベース・マネージャーによってクライアント・サイドで呼び出され、特定のユーザー ID およびパスワード (そしておそらく新規パスワードおよび usernamespace) を、接続時に指定された値とは異なる値に再マップします。

DB2 データベース・マネージャーは、接続時にユーザー ID およびパスワードが指定されている場合にのみ、この API を呼び出します。これは、プラグインがユーザー ID を自らユーザー ID/パスワードのペアに再マップすることを防止します。この API はオプションであり、セキュリティー・プラグインによって提供あるいはインプリメントされていなければ呼び出されません。

API およびデータ構造構文

```
SQL_API_RC ( SQL_API_FN *db2secRemapUserid)
                      ( char userid[DB2SEC_MAX_USERID_LENGTH],
                        db2int32 *useridlen,
                        char usernamespace[DB2SEC_MAX_USERNAMESPACE_LENGTH],
                        db2int32 *usernamespacelen,
                        db2int32 *usernamespacetype,
                        char password[DB2SEC_MAX_PASSWORD_LENGTH],
                        db2int32 *passwordlen,
                        char newpasswd[DB2SEC_MAX_PASSWORD_LENGTH],
                        db2int32 *newpasswdlen,
                        const char *dbname,
                        db2int32 dbnamelen,
                        char      **errormsg,
                        db2int32 *errormsglen);
```

db2secRemapUserid API パラメーター

userid
入力または出力。再マップされるユーザー ID。入力ユーザー ID 値がある場合は、API は出力ユーザー ID 値を提供しなければならず、それは入力ユーザー ID 値と同じか、あるいは異なる値になる可能性があります。入力ユーザー ID 値がない場合、API は出力ユーザー ID 値を戻すべきではありません。

useridlen
入力または出力。**userid** パラメーター値のバイト単位の長さ。

usernamespace
入力または出力。ユーザー ID のネーム・スペース。この値は、オプションで再マップすることができます。入力パラメーター値が指定されず、出力値が戻されている場合、**usernamespace** は CLIENT タイプ認証の場合にのみ DB2 データベース・マネージャーによって使用され、他の認証タイプでは無視されます。

usernamespacelen
入力または出力。**usernamespace** パラメーター値のバイト単位の長さ。
usernamespacetype パラメーターが値 DB2SEC_NAMESPACE_SAM_COMPATIBLE

(db2secPlugin.h で定義されている) に設定されていなければならないという制限のもとでは、現在サポートされる最大長は 15 バイトになります。

usernamespacetype
入力または出力。namespace タイプの古い値と新しい値。現時点でサポートされる唯一のネーム・スペース・タイプ値は、DB2SEC_NAMESPACE_SAM_COMPATIBLE です (domain¥myname などのユーザー名スタイルに相当します)。

password
入力または出力。入力の場合、これは再マップ対象のパスワードになります。出力の場合、これは再マップ済みパスワードになります。このパラメーターで入力値が指定されている場合、API は入力値とは異なる出力値を戻すことができなければなりません。入力値が指定されていない場合、API は出力 password 値を戻してはなりません。

passwordlen
入力または出力。**password** パラメーター値のバイト単位の長さ。

newpasswd
入力または出力。入力の場合、これは設定される新規パスワードになります。出力の場合、これは確認済みの新規パスワードになります。

注: これは、DB2 データベース・マネージャーが、クライアント上またはサーバー上 (**authentication** データベース・マネージャー構成パラメーターの値によって異なる) の db2secValidatePassword API の **newpassword** パラメーターに渡す新規パスワードです。新規パスワードが入力として渡された場合は、API は出力値を戻すことができなければなりませんが、出力値が別の新規パスワードである可能性もあります。入力として渡された新規パスワードがない場合、API は出力の新規パスワードを戻すべきではありません。

newpasswdlen
入力または出力。**newpasswd** パラメーター値のバイト単位の長さ。

dbname
入力。クライアントの接続先のデータベースの名前。

dbnamelen
入力。 **dbname** パラメーター値のバイト単位の長さ。

errormsg
出力。db2secRemapUserid API が正常に実行されない場合にこのパラメーターで戻されることのある、プラグインによって割り振られた ASCII エラー・メッセージ・ストリングのアドレスを指すポインター。

errormsglen
出力。**errormsg** パラメーターのエラー・メッセージ・ストリングのバイト単位の長さを示す整数を指すポインター。

db2secServerAuthPluginInit - サーバー認証プラグインの初期化

db2secServerAuthPluginInit API は、サーバー認証プラグインのロードの直後に DB2 データベース・マネージャーが呼び出す、プラグインの初期化 API です。

GSS-API の場合は、プラグインが、初期化時に gssapi_server_auth_functions 構造内部の **serverPrincipalName** パラメーターにサーバーのプリンシパル名を入れ、

gssapi_server_auth_functions 構造内部の **serverCredHandle** パラメーターにサーバーの証明書ハンドルを提供します。プリンシパル名および証明書ハンドルを保持するために割り振られているメモリーを解放するのは、gss_release_name および gss_release_cred API を呼び出すことによって、db2secServerAuthPluginTerm API が行うべき事柄です。

API およびデータ構造構文

```
SQL_API_RC SQL_API_FN db2secServerAuthPluginInit
                              ( db2int32 version,
                                void *server_fns,
                                db2secGetConDetails *getConDetails_fn,
                                db2secLogMessage *logMessage_fn,
                                char               **errormsg,
                                db2int32 *errormsglen );
```

db2secServerAuthPluginInit API パラメーター

version
入力。 DB2 データベース・マネージャーが現在サポートしている API の最大のバージョン番号。 DB2SEC_API_VERSION 値 (db2secPlugin.h 内) には、DB2 データベース・マネージャーが現在サポートしている API の最新のバージョン番号が含まれます。

server_fns
出力。DB2 データベース・マネージャーによって提供されるメモリーを指すポインター。このメモリーは、GSS-API 認証が使用される場合は、db2secGssapiServerAuthFunctions_<version_number> 構造 (gssapi_server_auth_functions_<version_number> としても知られる) のために提供され、ユーザー ID/パスワード認証が使用される場合は db2secUseridPasswordServerAuthFunctions_<version_number> 構造 (userid_password_server_auth_functions_<version_number> としても知られる) のために提供されます。 db2secGssapiServerAuthFunctions_<version_number> 構造と db2secUseridPasswordServerAuthFunctions_<version_number> 構造には、GSS-API 認証プラグイン用にインプリメントされた API を指すポインターとユーザー ID/パスワード認証プラグイン用にインプリメントされた API を指すポインターが含まれています。

server_fns パラメーターは、プラグインがインプリメントしているバージョンに対応する gssapi_server_auth_functions_<version_number> 構造を指すポインターとしてキャストします。 gssapi_server_auth_functions_<version_number> 構造または userid_password_server_auth_functions_<version_number> 構造の最初のパラメーターは、プラグインがインプリメントしている API のバージョンを DB2 データベース・マネージャーに知らせます。

注: DB2 のバージョンが、プラグインがインプリメントしている API のバージョンと同じかそれより大きい場合に限り、キャストが行われます。

gssapi_server_auth_functions_<version_number> または userid_password_server_auth_functions_<version_number> 構造内では、**plugintype** パスワードを DB2SEC_PLUGIN_TYPE_USERID_PASSWORD、DB2SEC_PLUGIN_TYPE_GSSAPI、または DB2SEC_PLUGIN_TYPE_KERBEROS のいずれかに設定する必要があります。将来のバージョンの API では、他の値も定義される可能性があります。

getConDetails_fn

入力。 DB2 によってインプリメントされる db2secGetConDetails API を指すポインター。 db2secServerAuthPluginInit API は、いずれかの他の認証 API で db2secGetConDetails API を呼び出して、データベース接続に関する詳細を取得することができます。これらの詳細には、接続に関連した通信メカニズム (TCP/IP の場合は IP アドレスなど) についての情報が含まれ、これは、プラグイン作成者が認証についての決定をする際に参照しなければならない可能性があります。例えば、プラグインは、特定のユーザーが特定の IP アドレスから接続しようとしているのでない場合、そのユーザーの接続を禁止できます。db2secGetConDetails API の使用はオプションです。

データベース接続に関係しない状況で db2secGetConDetails API が呼び出された場合、これは値 DB2SEC_PLUGIN_NO_CON_DETAILS を戻し、それ以外の場合は、正常なら 0 を戻します。

db2secGetConDetails API は、db2sec_con_details_<version_number> 構造を指すポインターである **pConDetails** と、使用される db2sec_con_details 構造を示すバージョン番号である **conDetailsVersion** という 2 つの入力パラメーターをとります。可能な値は、db2sec_con_details1 が使用される場合は DB2SEC_CON_DETAILS_VERSION_1 で、db2sec_con_details2 が使用される場合は DB2SEC_CON_DETAILS_VERSION_2 です。使用するように推奨されているバージョン番号は DB2SEC_CON_DETAILS_VERSION_2 です。

正常に戻る場合、db2sec_con_details 構造 (db2sec_con_details1 または db2sec_con_details2) には以下の情報が含まれます。

- サーバーへの接続に使用されるプロトコル。プロトコル定義のリストは、ファイル sqlenv.h (include ディレクトリーにある) (SQL_PROTOCOL_*) にあります。この情報は **clientProtocol** パラメーターに書き込まれます。
- **clientProtocol** が SQL_PROTOCOL_TCPIP または SQL_PROTOCOL_TCPIP4 の場合、サーバーへのインバウンド接続の TCP/IP アドレス。この情報は **clientIPAddress** パラメーターに書き込まれます。
- クライアントが接続しようとしているデータベースの名前。これは、インスタンス接続の場合は設定されません。この情報は **dbname** および **dbnameLen** パラメーターに書き込まれます。
- db2secValidatePassword API の **connection_details** パラメーター内に記述されるのと同じ詳細を含む、接続情報のビットマップ。この情報は **connect_info_bitmap** パラメーターに書き込まれます。
- **clientProtocol** が SQL_PROTOCOL_TCPIP6 の場合、サーバーへのインバウンド接続の TCP/IP アドレス。この情報は **clientIP6Address** パラメーターに書き込まれ、db2secGetConDetails API 呼び出しに DB2SEC_CON_DETAILS_VERSION_2 が使用される場合にのみ使用できます。

logMessage_fn

入力。 DB2 データベース・マネージャーによってインプリメントされる db2secLogMessage API を指すポインター。db2secClientAuthPluginInit API は、db2secLogMessage API を呼び出して、デバッグまたは通知の目的でメッセージを **db2diag** ログ・ファイルに記録することができます。db2secLogMessage API の最初のパラメーター (**level**) は、**db2diag** ログ・ファイルに記録される診断エラーのタイプを指定し、最後の 2 つのパラメーターはメッセージ・ストリン

グとその長さです。(db2secPlugin.h で定義された) db2secLogMessage API の最初のパラメーターの有効な値は以下のとおりです。

- DB2SEC_LOG_NONE (0): ロギングなし
- DB2SEC_LOG_CRITICAL (1): 重大エラーを検出した
- DB2SEC_LOG_ERROR (2): エラーを検出した
- DB2SEC_LOG_WARNING (3): 警告
- DB2SEC_LOG_INFO (4): 通知

メッセージ・テキストが **db2diag** ログ・ファイルに表示されるのは、db2secLogMessage API の **level** パラメーターの値が **diaglevel** データベース・マネージャー構成パラメーターの値以下である場合のみです。

そのため、例えば DB2SEC_LOG_INFO 値を使用する場合、メッセージ・テキストは **diaglevel** データベース・マネージャー構成パラメーターに 4 が設定されている場合にのみ **db2diag** ログ・ファイルに表示されます。

errormsg
出力。db2secServerAuthPluginInit API が正常に実行されない場合にこのパラメーターで戻されることのある、プラグインによって割り振られた ASCII エラー・メッセージ・ストリングのアドレスを指すポインター。

errormsglen
出力。**errormsg** パラメーターのエラー・メッセージ・ストリングのバイト単位の長さを示す整数を指すポインター。

db2secServerAuthPluginTerm API - サーバー認証プラグイン・リソースのクリーンアップ

db2secServerAuthPluginTerm API は、サーバー認証プラグインによって使用されるリソースを解放します。

この API は、DB2 データベース・マネージャーがサーバー認証プラグインをアンロードする直前に呼び出します。これは、プラグイン・ライブラリーが保持しているリソースの適切なクリーンアップを実行する、という方法でインプリメントされる必要があります。例えば、プラグインによって割り振られたメモリーを解放し、まだオープンしているファイルをクローズし、ネットワーク接続をクローズします。これらのリソースを解放するためにその記録を保持することは、プラグインが行います。この API は Windows オペレーティング・システムでは呼び出されません。

API およびデータ構造構文

```
SQL_API_RC ( SQL_API_FN *db2secServerAuthPluginTerm)
                      ( char     **errormsg,
                        db2int32 *errormsglen );
```

db2secServerAuthPluginTerm API パラメーター

errormsg
出力。db2secServerAuthPluginTerm API が正常に実行されない場合にこのパラメーターで戻されることのある、プラグインによって割り振られた ASCII エラー・メッセージ・ストリングのアドレスを指すポインター。

errormsglen
出力。**errormsg** パラメーターのエラー・メッセージ・ストリングのバイト単位の長さを示す整数を指すポインター。

db2secValidatePassword API - パスワードの確認

データベース接続の操作時に、ユーザー ID およびパスワードのスタイル認証を実行する方法について説明します。

注: API がクライアント・サイドで実行されている場合、API コードは、CONNECT ステートメントを実行するユーザーの特権で実行されます。この API は、**authentication** 構成パラメーターが CLIENT に設定されている場合にのみ、クライアント・サイドで呼び出されます。

API がサーバー・サイドで実行されている場合、API コードはインスタンス所有者の特権で実行されます。

認証に特別な特権 (UNIX 上のルート・レベルのシステム・アクセスなど) が必要な場合、プラグイン作成者は、上記のシナリオを考慮する必要があります。

この API は、パスワードが正常な場合は値 DB2SEC_PLUGIN_OK (正常) を、また、パスワードが無効な場合は DB2SEC_PLUGIN_BADPWD などのエラー・コードを戻す必要があります。

API およびデータ構造構文

```
SQL_API_RC ( SQL_API_FN *db2secValidatePassword)
                      ( const char *userid,
                        db2int32 useridlen,
                        const char *usernamespace,
                        db2int32 usernamespacelen,
                        db2int32 usernamespacetype,
                        const char *password,
                        db2int32 passwordlen,
                        const char *newpasswd,
                        db2int32 newpasswdlen,
                        const char *dbname,
                        db2int32 dbnamelen,
                        db2Uint32 connection_details,
                        void      **token,
                        char      **errormsg,
                        db2int32 *errormsglen );
```

db2secValidatePassword API パラメーター

userid
入力。パスワードが検証されるユーザー ID。

useridlen
入力。 **userid** パラメーター値のバイト単位の長さ。

usernamespace
入力。取得されたユーザー ID が属するネーム・スペース。

usernamespacelen
入力。 **usernamespace** パラメーター値のバイト単位の長さ。

usernamespacetype
入力。ネーム・スペースのタイプ。 **usernamespacetype** パラメーターの有効な値 (db2secPlugin.h で定義されている) は以下のとおりです。

- DB2SEC_NAMESPACE_SAM_COMPATIBLE は domain¥myname などのユーザー名スタイルに対応します。
- DB2SEC_NAMESPACE_USER_PRINCIPAL は myname@domain.ibm.com などのユーザー名スタイルに対応します。

現在のところ、DB2 データベース・システムは値 DB2SEC_NAMESPACE_SAM_COMPATIBLE しかサポートしていません。ユーザー ID がない場合、**usernamespacetype** パラメーターの値は DB2SEC_USER_NAMESPACE_UNDEFINED (db2secPlugin.h で定義された) に設定されます。

password
入力。検証されるパスワード。

passwordlen
入力。 **password** パラメーター値のバイト単位の長さ。

newpasswd
入力。パスワードが変更される場合の新規パスワード。変更が要求されない場合、このパラメーターは NULL に設定されます。このパラメーターが非ヌルの場合、この API は、旧パスワードを新規パスワードに変更する前に確認する必要があります。 API は、パスワードの変更要求を受け入れなくても構いませんが、受け入れない場合は、旧パスワードを確認せずに即時に戻り値 DB2SEC_PLUGIN_CHANGEPASSWORD_NOTSUPPORTED を戻す必要があります。

newpasswdlen
入力。 **newpasswd** パラメーター値のバイト単位の長さ。

dbname
入力。接続先のデータベースの名前。この API は **dbname** パラメーターを無視しても差し支えありません。あるいは、特定のデータベースへのアクセスを、有効なパスワードを特別に持つユーザーのみに限定する方針をとっている場合は、この関数は値 DB2SEC_PLUGIN_CONNECTIONREFUSED を戻すことができます。このパラメーターは、NULL にすることができます。

dbnamelen
入力。 **dbname** パラメーター値のバイト単位の長さ。 **dbname** パラメーターが NULL の場合、このパラメーターは 0 に設定されます。

connection_details
入力。 32 ビットのパラメーター。そのうちの 3 ビットが、以下の情報を保管するために現在使用されています。

- 右端のビットは、ユーザー ID のソースが db2secGetDefaultLoginContext API のデフォルトであるか、それとも接続時に明示的に指定されているかを示します。
- 右から 2 番目のビットは、接続がローカル (Inter Process Communication (IPC) を使用しているか、あるいはパーティション・データベース環境の db2nodes.cfg 内にあるノードのいずれかからの接続である) か、リモート (ネットワークまたはループバックを経由) かを示します。これによって、API

は同一のマシン上のクライアントがパスワードなしで DB2 サーバーに接続できるかどうか判別できます。デフォルトのオペレーティング・システム・ベースのユーザー ID/パスワード・プラグインにより、同一のマシン上のクライアントからのパスワードなしのローカル接続が常時許可されます (ユーザーが接続特権を持つ場合)。

- 右から 3 番目のビットは、DB2 データベース・マネージャーがサーバー・サイドまたはクライアント・サイドのどちらで API を呼び出しているかを示します。

ビット値は、db2secPlugin.h 内で次のように定義されています。

- DB2SEC_USERID_FROM_OS は、ユーザー ID は OS から取得され、接続ステートメントで明示的には指定されていないことを示します。
- DB2SEC_CONNECTION_ISLOCAL はローカル接続を示します。
- DB2SEC_VALIDATING_ON_SERVER_SIDE は、 DB2 データベース・マネージャーがサーバー・サイドまたはクライアント・サイドのどちらからパスワードの確認を呼び出しているかを示します。このビット値が設定されている場合、DB2 データベース・マネージャーはサーバー・サイドから呼び出しています。それ以外の場合は、クライアント・サイドから呼び出しています。

暗黙的な認証での DB2 データベース・システムのデフォルト動作では、パスワード確認なしの接続が許可されます。しかし、プラグイン開発者は、DB2SEC_PLUGIN_BADPASSWORD エラーを返すことによって、暗黙的な認証を禁止することができます。

token
入力。現行接続中の後続の API 呼び出しに渡されるデータを指すポインター。呼び出される可能性のある API は、db2secGetAuthIDs API と db2secGetGroupsForUser API です。

errormsg
出力。db2secValidatePassword API が正常に実行されない場合にこのパラメーターで戻されることのある、プラグインによって割り振られた ASCII エラー・メッセージ・ストリングのアドレスを指すポインター。

errormsglen
出力。**errormsg** パラメーターのエラー・メッセージ・ストリングのバイト単位の長さを示す整数を指すポインター。

GSS-API 認証プラグインに必要な API および定義

以下の表は、DB2 セキュリティー・プラグイン・インターフェースに必要な GSS-API の完全なリストです。

サポートされている API は、*Generic Security Service Application Program Interface, Version 2* (IETF RFC2743) および *Generic Security Service API Version 2: C-Bindings* (IETF RFC2744) の仕様に準拠しています。 GSS-API ベース・プラグインをインプリメントする前に、これらの仕様について完全に理解しておく必要があります。

表 39. GSS-API 認証プラグインに必要な API および定義

API タイプ	API 名	説明
クライアント・サイド API	gss_init_sec_context	対等アプリケーションとのセキュリティー・コンテキストを開始します。
サーバー・サイド API	gss_accept_sec_context	対等アプリケーションによって開始されたセキュリティー・コンテキストを受け入れます。
サーバー・サイド API	gss_display_name	内部フォーマットの名前をテキストに変換します。
共通 API	gss_delete_sec_context	確立されたセキュリティー・コンテキストを削除します。
共通 API	gss_display_status	GSS-API 状況コードに関連したテキスト・エラー・メッセージを取得します。
共通 API	gss_release_buffer	バッファーを削除します。
共通 API	gss_release_cred	GSS-API 証明書に関連したローカル・データ構造を解放します。
共通 API	gss_release_name	内部フォーマットの名前を削除します。
必要な定義	GSS_C_DELEG_FLAG	委任を要求します。
必要な定義	GSS_C_EMPTY_BUFFER	gss_buffer_desc にデータが含まれていないことを意味します。
必要な定義	GSS_C_GSS_CODE	GSS メジャー状況コードを示します。
必要な定義	GSS_C_INDEFINITE	メカニズムがコンテキストの有効期限をサポートしていないことを示します。
必要な定義	GSS_C_MECH_CODE	GSS マイナー状況コードを示します。
必要な定義	GSS_C_MUTUAL_FLAG	相互認証が要求されました。
必要な定義	GSS_C_NO_BUFFER	gss_buffer_t 変数が有効な gss_buffer_desc 構造を指していないことを示します。
必要な定義	GSS_C_NO_CHANNEL_BINDINGS	通信チャネル・バインディングがありません。
必要な定義	GSS_C_NO_CONTEXT	gss_ctx_id_t 変数が有効なコンテキストを指していないことを示します。
必要な定義	GSS_C_NO_CREDENTIAL	gss_cred_id_t 変数が有効な証明書ハンドルを指していないことを示します。
必要な定義	GSS_C_NO_NAME	gss_name_t 変数が有効な内部名を指していないことを示します。
必要な定義	GSS_C_NO_OID	デフォルト認証メカニズムを使用します。
必要な定義	GSS_C_NULL_OID_SET	デフォルト・メカニズムを使用します。
必要な定義	GSS_S_COMPLETE	API が正常に完了しました。
必要な定義	GSS_S_CONTINUE_NEEDED	プロセスが完了しておらず、対等機能から受け取った応答トークンを指定して API をもう一度呼び出す必要があります。

GSS-API 認証プラグインに関する制約事項

以下のリストは、GSS-API 認証プラグインに関する制約事項について説明しています。

- デフォルト・セキュリティー・メカニズムが常時採用されるため、OID についての考慮事項はありません。

- gss_init_sec_context() で要求される GSS サービスは、相互認証および委任だけです。 DB2 データベース・マネージャーは常に委任のためのチケットを要求しますが、そのチケットを使用して新規チケットを生成することはありません。
- デフォルト・コンテキスト時刻のみが要求されます。
- gss_delete_sec_context() からのコンテキスト・トークンは、クライアントからサーバー、またはその逆には送信されません。
- 匿名はサポートされていません。
- チャネル・バインディングはサポートされていません。
- 初期証明書の有効期限が切れた場合、DB2 データベース・マネージャーはそれを自動的に更新しません。
- GSS-API 仕様は、gss_init_sec_context() または gss_accept_sec_context() が失敗しても、対等機能に送信するトークンがいずれかの関数によって戻されなければならないことを規定しています。しかし、DRDA の制限のため、DB2 データベース・マネージャーは gss_init_sec_context() が失敗し、かつ最初の呼び出しでトークンを生成した場合にのみトークンを送信できます。

第 10 章 通信バッファー出口ライブラリー

DB2 for Linux, UNIX, and Windows のデータベース・マネージャーは、顧客およびベンダーが通信バッファーを検討できるようにします。クライアントとデータベース・サーバーの間を流れる通信バッファーに送受信の前にアクセスするため、信頼できる外部の共有ライブラリーが使用されます。それらの外部のライブラリーは、通信バッファー出口ライブラリーとして知られています。

通信バッファー出口ライブラリーを使用することで、監査などのソリューションやバッファー内容に基づく他のセキュリティー・ソリューションを提供するために、通信バッファーを検査することができます。 DB2 for Linux, UNIX, and Windows は、クライアントに送信する直前の各バッファーとクライアントから受信した各バッファーにアクセスする機能を備えています。バッファーは、DATA_ENCRYPT 認証または SSL のいずれかを使用して暗号化される前に用意されます。 DB2 for Linux, UNIX, and Windows は、クライアントとサーバーの間の通信に、DRDA プロトコルを使用します。通信バッファー出口ライブラリーに渡される通信バッファーは、DRDA プロトコルに従ってフォーマットされます。通信バッファー出口ライブラリーは、通信に使用される DRDA プロトコルに対応するものでなければなりません。

DB2 for Linux, UNIX, and Windows は、通信プロトコルに関係なくバッファーを提供します。通信バッファー出口ライブラリーは、TCPIP (IPv4 および IPv6)、SSL、プロセス間通信 (IPC)、および名前付きパイプに対応しています。

バッファーに加えて、DB2 for Linux, UNIX, and Windows では ID 情報も取得できます。これには、データベースとの接続用に設定されたユーザー名およびセッション許可 ID が含まれます。この情報は、Kerberos などの GSS-API プラグインが関与するシナリオにおいて有用です。そのようなシナリオでは、標準のユーザー名は存在せず、データベース・マネージャーは汎用のチケットからユーザー名を派生させます。この詳細は、通信バッファーを調べるだけでは取得できません。

データベース・マネージャーは、信頼できるライブラリーのみがロードされるようにします。ライブラリーは、インスタンスの所有者のみが変更できる特定の場所にインストールする必要があります。さらに、SYSADM 権限を持つユーザーのみが、そのライブラリーを使用可能にすることができます。この権限レベルは、暗号化 (DATA_ENCRYPT または SSL) を使用可能にするために必要な権限レベルと同じです。

通信バッファー出口ライブラリーは、提供されたバッファーのいずれかに、このライブラリーによって有害と見なされるデータが含まれている場合、接続を終了することができます。このデータには、サーバーに送信されるデータとクライアントに返されるデータの両方が含まれます。例えば、通信バッファー出口ライブラリーは、select ステートメントから返されるデータが、クライアントが受け取るデータとして不適切であることを検出する可能性があります。ライブラリーからの戻りコードによって、その接続を終了する必要があることが、データベース・マネージャーに示されます。データベース・マネージャーは、この通信バッファーまたはそのクライアントへの他の通信バッファーを停止し、接続を終了します。

注: これらの通信バッファー出口ライブラリーは、通常、サード・パーティー・ベンダーによって提供されます。 DB2 for Linux, UNIX, and Windows には、sqllib/samples/security/commexit ディレクトリーにライブラリーのサンプルがあります。サンプルをガイドとして使用して、独自のライブラリーを作成することができます。

通信バッファー出口ライブラリーのデプロイメント

通信バッファー出口ライブラリーのデプロイメントについては、いくつかの考慮事項があります。

標準的なシナリオでは、通信バッファー出口ライブラリーはベンダーによって提供されます。そのようなシナリオでは、通信バッファー出口ライブラリーのデプロイメントは、ベンダーが提供するインストール・スクリプトによって処理されます。ここでは、独自のライブラリーをデプロイする場合のデプロイメント手順について概説します。

通信バッファー出口ライブラリーのロケーション

通信バッファー出口ライブラリーは、特定のディレクトリーに存在する必要があります。

データベース・マネージャーは、以下のディレクトリーにある通信バッファー出口ライブラリーを検索します。

Linux および UNIX 32 ビット
$DB2PATH/security32/plugin/commexit

Linux および UNIX 64 ビット
$DB2PATH/security64/plugin/commexit

Windows 32 ビットおよび 64 ビット
$DB2PATH¥security¥plugin¥instance_name¥commexit

注: Windows プラットフォームの場合、サブディレクトリー instance_name および commexit は自動的には作成されません。これらは、インスタンスの所有者が手動で作成しなければなりません。

通信バッファー出口ライブラリーの命名規則と権限

通信バッファー出口ライブラリーは、プラットフォーム固有の命名規則および権限規則に従う必要があります。

通信バッファー出口ライブラリー名の最大長 (ファイル拡張子および接尾部の 64 を含まない) は、32 バイトに制限されています。

次のリストは、ライブラリー・ファイル拡張子の命名規則の概要を、プラットフォームごとに示しています。

AIX
拡張子は .a または .so にする必要があります。

注: .a と .so の両方の拡張子が存在する場合は、.a が使用されます。

Linux、HP IPF、および Solaris
拡張子は .so にする必要があります。

Windows
拡張子は .dll にする必要があります。

次のリストは、ライブラリー・ファイルの権限の概要を、プラットフォームごとに示しています。

UNIX および Linux
インスタンス所有者が所有し、そのインスタンス所有者のみが読み取りおよび実行できます。

Windows
DB2AMINS グループのメンバーが所有し、DB2ADMINS グループのメンバーが読み取りおよび実行できます。

例

以下の例は、すべてのプラットフォームにおける mycommexit というライブラリーの通信バッファー出口ライブラリー拡張子を示しています。

- AIX 64 ビット: mycommexit.a または mycommexit.so
- Solaris 64 ビット、Linux 32 ビットまたは 64 ビット、IPF 上の HP 64 ビット: mycommexit.so
- Windows 32 ビット: mycommexit.dll
- Windows 64 ビット: mycommexit64.dll

  注: ファイル名の接尾部の 64 は、Windows 64 ビットにおけるライブラリー名の場合にのみ必要です。

  通信バッファー出口ライブラリーの名前を使用してデータベース・マネージャーの構成を更新する場合は、接尾部の 64 を除いたライブラリーの絶対パス名を使用します。ファイル拡張子とファイルの修飾パスは、データベース・マネージャーの構成を更新する際にはどちらも指定しないでください。

次の例では、Windows 64 ビット・システムでデータベース・マネージャーの構成を更新し、mycommexit64.dll ライブラリーを通信バッファー出口ライブラリーとして設定します。

```
UPDATE DBM CFG USING COMM_EXIT_LIST mycommexit
```

注: COMM_EXIT_LIST の名前には大/小文字の区別があるため、ライブラリー名と完全に一致している必要があります。

DB2 pureScale 環境以外で通信バッファー出口ライブラリーを使用可能にする

この操作で概略する手順は、通常、サード・パーティーが提供するインストール・スクリプトによって実行されます。この手順の概略は、独自に作成した通信バッファー出口ライブラリーを使用可能にする際に役立ちます。

始める前に

この操作の手順を実行するには、SYSADM 権限が必要です。

制約事項

通信バッファー出口ライブラリー・ファイルは、厳格なファイル権限ガイドラインに従う必要があります。これらのガイドラインについて詳しくは、関連概念を参照してください。

手順

通信バッファー出口ライブラリーを使用可能にするには、以下のようにします。

1. データベース・マネージャーを停止します。データベース・マネージャーを停止するには、**db2stop** コマンドを実行します。
2. 通信バッファー出口ライブラリー・ファイルを正しいディレクトリーにコピーします。 必要な通信バッファー出口ライブラリーの場所について詳しくは、関連概念を参照してください。このファイルは、必要に応じて別のロケーションへのシンボリック・リンクにすることができます。
3. データベース・マネージャー構成パラメーター **COMM_EXIT_LIST** をライブラリーの名前で更新します。 構成パラメーターを更新するには、**UPDATE DBM CFG** コマンドを使用します。
4. データベース・マネージャーを開始します。データベース・マネージャーを開始するには、**db2start** コマンドを実行します。

タスクの結果

ライブラリーがロードされ、初期化されます。

DB2 pureScale 環境で通信バッファー出口ライブラリーを使用可能にする

この操作で概略する手順は、通常、サード・パーティーが提供するインストール・スクリプトによって実行されます。この手順の概略は、独自に作成した通信バッファー出口ライブラリーを使用可能にする際に役立ちます。

このタスクについて

ファイル名にバージョン番号が含まれる通信バッファー出口ライブラリーを使用することで、また、バージョン番号を持たないライブラリーの場合はこのファイルへのシンボリック・リンクを使用することで、ライブラリーをメンバーごとにデプロイすることができます。このシナリオでは、インスタンス全体を停止する必要はありません。個々のメンバーを停止するだけで実行できます。

制約事項

通信バッファー出口ライブラリー・ファイルは、厳格なファイル権限ガイドラインに従う必要があります。これらのガイドラインについて詳しくは、関連概念を参照してください。

手順

通信バッファー出口ライブラリーを使用可能にするには、以下のようにします。

1. ファイル名にバージョン番号が含まれる通信バッファー出口ライブラリー・ファイルを、正しいディレクトリーにコピーします。 必要な通信バッファー出口ライブラリーの場所について詳しくは、関連概念を参照してください。
2. バージョンを持たないライブラリーからファイル名にバージョンが含まれるライブラリーへのシンボリック・リンクを作成します。
3. データベース・マネージャー構成パラメーター **comm_exit_list** をライブラリーの名前で更新します。 構成パラメーターを更新するには、**UPDATE DBM CFG** コマンドを使用します。
4. 各メンバーを個別に停止します。 各メンバーを停止するには、メンバーごとに **db2stop** コマンドを実行します。
5. 停止したメンバーを再始動します。 停止したメンバーを始動するには、**db2start** コマンドを実行します。

タスクの結果

ライブラリーがロードされ、初期化されます。

通信バッファー出口ライブラリーの問題判別

通信バッファー出口ライブラリーの問題の診断に役立ついくつかのオプションが使用可能です。

通信バッファー出口ライブラリーは、DB2 for Linux, UNIX, and Windows の一部としては組み込まれていません。これは、自分でインストールするライブラリーです。使用しているツールまたはアプリケーションによって自動的にインストールおよび構成される場合と、自分で作成する場合があります。

データベース・マネージャーの構成パラメーター **comm_exit_list** で指定されるライブラリー名には、そのライブラリーのソースに関する何らかのヒントが含まれています。

ライブラリーに問題がある場合は、そのツールまたはアプリケーションの資料を参照して、行う必要のある問題判別手順を確認する必要があります。

通信バッファー出口ライブラリーは、db2diag ログ・ファイルに書き込むためのインターフェースを使用することができます。ライブラリーがどのように機能しているかについて気になる点がある場合は、db2diag ログ・ファイルを調べることができます。

通信バッファー出口ライブラリーのパフォーマンスについて気になる点がある場合は、待ち時間のモニターを使用して、ライブラリーの処理時間を調べることができます。これらのモニター・ツールについて詳しくは、関連資料を参照してください。

通信バッファー出口ライブラリーの作成

通信バッファー出口ライブラリーの作成については、いくつかの考慮事項があります。

標準的なシナリオでは、通信バッファー出口ライブラリーはベンダーによって提供されます。そのようなシナリオでは、通信バッファー出口ライブラリーの作成は、ベンダーによって扱われます。自分でライブラリーを作成することもできます。

通信バッファー出口ライブラリーのロード方法

データベース・マネージャーが開始されると、通信バッファー出口ライブラリーが動的にロードされ、初期化されます。ライブラリーには初期化関数 db2commexitInit が含まれている必要があります。この関数は、ライブラリー初期化関数と呼ばれます。

ライブラリーの初期化関数は、指定されている通信バッファー出口ライブラリーを初期化します。初期化によって、ライブラリー関数を呼び出すために必要な情報が、データベース・マネージャーに提供されます。ライブラリー初期化関数は、以下のパラメーターを受け入れます。

- ライブラリーを呼び出すデータベース・インスタンスがサポートできる関数ポインター構造の最高バージョン番号。
- インプリメンテーションを必要とするすべての API を指すポインターを収めた構造を指すポインター。
- db2diag ログ・ファイルにログ・メッセージを追加する関数を指すポインター。
- エラー・メッセージ・ストリングを指すポインター。
- エラー・メッセージの長さ。

初期化関数の関数シグニチャーは以下のとおりです。

```
SQL_API_RC SQL_API_FN db2commexitInit
 ( db2int32 version,
   void *commexit_fns,
   db2commexitLogMessage *logMessage_fn,
   char     **errormsg,
   db2int32 *errormsglen );
```

初期化関数は、規定の関数名を使用しなければならない、ライブラリー内の唯一の関数です。その他のライブラリー関数は、初期化関数から戻された関数ポインターを通して参照されます。

この関数固有のタスクは、以下のとおりです。

- 関数ポインターを、該当する関数構造を指すポインターにキャストする。
- ライブラリー内の他の関数を指すポインターを割り当てる。
- 返される関数ポインター構造のバージョン番号を割り当てる。

ライブラリーが C++ としてコンパイルされている場合、関数 db2commexitInit は extern "C" として宣言する必要があります。

通信バッファー出口ライブラリー API

通信バッファー出口ライブラリーには、API がインプリメントされています。

db2commexitInit API - 初期化

db2start コマンドを使用してデータベース・マネージャーが開始されると、通信バッファー出口ライブラリーがロードされます。ライブラリーのロード後すぐに、この関数が呼び出されます。この関数は、通信バッファー出口ライブラリーの初期化を行います。またこの関数は、インプリメントされたすべての関数をデータベース・マネージャーに返します。

ライブラリーが C++ としてコンパイルされている場合、この関数は extern "C" として宣言する必要があります。

この関数が呼び出されるのは 1 回のみであるため、この関数はスレッド・セーフである必要はありません。

API ヘッダー・ファイル

db2commexit.h

API およびデータ構造構文

```
SQL_API_RC ( SQL_API_FN * db2commexitInit )
(
      db2int32 version,
      void *commexit_fns,
      db2commexitLogMessage *logMessage_fn,
      char     **errormsg,
      db2int32 *errormsglen
);
```

db2commexiInit API パラメーター

version
入力。そのライブラリーをロードするインスタンスによってサポートされる API の最上位バージョン。db2commexit.h の中の値 DB2COMMEXIT_API_VERSION には、データベース・マネージャーが現在サポートしている API の最新のバージョン番号が入ります。

commexit_fns
出力。db2commexitFunctions_<version_number> 構造を指すポインター。この構造には、通信バッファー出口ライブラリー用にインプリメントされた API を指すポインターが格納されます。 API には別のバージョンが存在する可能性があるため、ライブラリーによってインプリメントされたバージョンに対応する db2commexitFunctions_<version_number> 構造に **commexit_fns** パラメーターがキャストされます。 db2commexitFunctions_<version_number> 構造の最初のパラメーターは、プラグインによってインプリメントされた API のバージョンを示します。

logMessage_fn
入力。 DB2 データベース・システムによってインプリメントされる db2commexitLogMessage API を指すポインター。db2commexitInit API は、db2commexitLogMessage API を呼び出して、デバッグまたは通知の目的でメッセージを db2diag ログ・ファイルに記録することができます。
db2commexitLogMessage API の最初のパラメーターは、db2diag ログ・ファイルに記録される診断エラーのタイプを指定します。最後の 2 つのパラメーター

は、メッセージ・ストリングとその長さです。 db2commexit.h で定義される db2commexitLogMessage API の最初のパラメーターの有効な値は、以下のとおりです。

- DB2COMMEXIT_LOG_NONE: (0) ロギングなし
- DB2COMMEXIT_LOG_CRITICAL: (1) 重大エラーを検出した
- DB2COMMEXIT_LOG_ERROR: (2) エラーを検出した
- DB2COMMEXIT_LOG_WARNING: (3) 警告
- DB2COMMEXIT_LOG_INFO: (4) 通知

メッセージ・テキストが db2diag ログ・ファイルに記録されるのは、db2commexitLogMessage API の「level」パラメーターの値が **diaglevel** データベース・マネージャー構成パラメーターの値以下である場合のみです。例えば DB2SEC_LOG_INFO 値を使用する場合、メッセージ・テキストは、**diaglevel** データベース・マネージャー構成パラメーターが 4 に設定されている場合にのみ、ログに記録されます。

errormsg
出力。関数が正常に実行されない場合にこのパラメーターで返すことができる、プラグインによって割り振られた ASCII エラー・メッセージ・ストリングのアドレスを指すポインター。このメモリーは、db2commexitFreeErrormsg を呼び出しても解放されません。

errormsglen
出力。**errormsg** パラメーターのエラー・メッセージ・ストリングのバイト単位の長さを示す整数を指すポインター。

db2commexitTerm API - 終了

この関数は、通信バッファー出口ライブラリーによって使用されるリソースを解放します。

この API は、データベース・マネージャーが **db2stop** の処理中に通信バッファー出口ライブラリーをアンロードする直前に、そのデータベース・マネージャーによって呼び出されます。この API は、ライブラリーが保持しているすべてのリソースを適切にクリーンアップするようにインプリメントされている必要があります。例えば、この API は、ライブラリーによって割り振られたすべてのメモリーを解放し、開かれたままになっているファイルを閉じ、ネットワーク接続を閉じる必要があります。これらのリソースを解放するために、ライブラリーはこれらのリソースをトラッキングする必要があります。

この関数が呼び出されるのは 1 回のみであるため、この関数はスレッド・セーフである必要はありません。

API ヘッダー・ファイル

db2commexit.h

API およびデータ構造構文

```
SQL_API_RC ( SQL_API_FN * db2commexitTerm )
(
   char     **errormsg,
   db2int32 *errormsglen
);
```

db2commexitTerm API パラメーター

errormsg
出力。通信バッファー出口ライブラリーによって割り振られた ASCII エラー・メッセージ・ストリングのアドレスを指すポインター。このエラー・メッセージ・ストリングは、関数が正常に実行されない場合にこのパラメーターで返されることがあります。このメモリーは、db2commexitFreeErrormsg を呼び出しても解放されません。

errormsglen
出力。**errormsg** パラメーターのエラー・メッセージ・ストリングのバイト単位の長さを示す整数を指すポインター。

db2commexitRegister API - 登録

この関数は、エージェントを接続に登録します。

この関数は、エージェントがソケットを受け入れ、そのソケットでデータの送受信を開始すると、データベース・マネージャーによって呼び出されます。このアクティビティーは、通常、データベースへの新しい SQL 接続またはインスタンス接続に関連付けられます。

この関数はまた、クライアントからの新しい要求を処理するために、アイドル状態の接続がエージェントにディスパッチされるときにも呼び出されます。

この関数は、データベースへの SQL 接続に直接関連付けられるわけではありません。この関数の入力パラメーターには、新しいソケットであるか、それとも新しいエージェントにディスパッチされる既存のソケットであるかで区別があります。

API ヘッダー・ファイル

```
db2commexit.h
```

API およびデータ構造構文

```
SQL_API_RC ( SQL_API_FN * db2commexitRegister )
(
   void                          ** pConnectionContext,
   const db2commexitCommInfo_v1   * pCommInfo,
   db2int32                         state,
   db2int64                       * pReservedFlags,
   char                          ** errormsg,
   db2int32                       * errormsglen
);
```

db2commexitRegister API パラメーター

pConnectionContext
入出力。通信バッファー出口ライブラリー固有のデータを指すポインター。このポインターは、インバウンド接続に固有のものです。このパラメーターは、その

接続における各関数呼び出しの入力として渡されます。ライブラリーは、接続固有の情報の割り振りと保管を行うことができます。そのようにすると、それらの情報を各関数呼び出しで使用できます。パラメーターのメモリーは、db2commexitDeregister の呼び出しの中で解放される必要があります。データベース・マネージャーは、このパラメーターによって指し示されるメモリーにはアクセスできません。

pCommInfo
入力。データベース・サーバーを識別する情報、および着信接続のプロトコル固有の情報が含まれる構造を指すポインター。その構造内の一部のフィールドは、複数のバッファーがクライアントと交換されるまでセットアップされません。それらのフィールドは、db2commexitRecv と db2commexitSend を後で呼び出すときに使用できます。このシナリオは、特に **inbound_appl_id**、**outbound_appl_id**、および **connection_type** に適用されます。これらの値が判明すると、**connection_type** パラメーターによって、その接続がローカル・データベースへの接続かゲートウェイ接続かが示されます。

State
入力。この関数が呼び出される条件を示します。可能な値は以下のとおりです。

- NEW_CONNECTION - 新しい物理着信クライアント接続を示します。
- AGENT_ASSOCIATION - 既存のアイドル状態のクライアント接続を示します。この接続が再度アクティブになり、要求を処理するためにエージェントに関連付けられます。

pReservedFlags
入出力。将来の使用のために予約されています。出力の場合、この値は 0 に設定される必要があります。

errormsg
出力。通信バッファー出口ライブラリーによって割り振られた ASCII エラー・メッセージ・ストリングのアドレスを指すポインター。このエラー・メッセージ・ストリングは、関数が正常に実行されない場合にこのパラメーターで返されることがあります。このメモリーは、db2commexitFreeErrormsg を呼び出しても解放されません。

errormsglen
出力。**errormsg** パラメーターのエラー・メッセージ・ストリングのバイト単位の長さを示す整数を指すポインター。

db2commexitDeregister API - 登録解除

この関数は、エージェントを、関連付けられている接続から解放します。

この関数は、エージェントが対象接続における要求の処理を停止すると、データベース・マネージャーによって呼び出されます。この状態になるのは、クライアントとの物理接続が終了するか、クライアントがアイドル状態で、エージェントとの関連付けが解除される場合です。

API ヘッダー・ファイル

db2commexit.h

API およびデータ構造構文

```
SQL_API_RC ( SQL_API_FN * db2commexitDeregister )
(
   void                           * pConnectionContext,
   const db2commexitCommInfo_v1   * pCommInfo,
   db2int32                         state,
   db2int64                       * pReservedFlags,
   char                          ** errormsg,
   db2int32                       * errormsglen
);
```

db2commexitDeregister API パラメーター

pConnectionContext
入力。通信バッファー出口ライブラリー固有のデータを指すポインター。このポインターは、インバウンド接続に固有のものです。このパラメーターは、その接続における各関数呼び出しの入力として渡されます。データベース・マネージャーは、このパラメーターによって指し示されるメモリーにはアクセスできません。このメモリーは、この関数で割り振り解除する必要があります。

pCommInfo
入力。データベース・サーバーを識別する情報、および着信接続のプロトコル固有の情報が含まれる構造を指すポインター。

State
入力。この関数が呼び出される条件を示します。可能な値は以下のとおりです。

- CONNECTION_TERM - クライアントとの物理接続が終了されることを示します。
- AGENT_DISASSOCIATION - クライアント接続がアイドル状態で、その接続とのエージェントの関連付けが解除されることを示します。

pReservedFlags
入出力。将来の使用のために予約されています。出力の場合、この値は 0 に設定される必要があります。

errormsg
出力。通信バッファー出口ライブラリーによって割り振られた ASCII エラー・メッセージ・ストリングのアドレスを指すポインター。このエラー・メッセージ・ストリングは、関数が正常に実行されない場合にこのパラメーターで返されることがあります。このメモリーは、db2commexitFreeErrormsg を呼び出しても解放されません。

errormsglen
出力。**errormsg** パラメーターのエラー・メッセージ・ストリングのバイト単位の長さを示す整数を指すポインター。

db2commexitRecv API - 受信

この関数は、データベース・マネージャーがクライアントから受信するバッファーごとに呼び出されます。

この関数は、クライアントから通信バッファーを受信した直後に、データベース・マネージャーによって呼び出されます。この関数はバッファーが暗号化解除されてから呼び出されるため、通信バッファー出口ライブラリーは、暗号化されていないバッファーにアクセスできます。

API ヘッダー・ファイル

db2commexit.h

API およびデータ構造構文

```
SQL_API_RC ( SQL_API_FN * db2commexitRecv )
(
   void                          * pConnectionContext,
   const db2commexitCommInfo_v1  * pCommInfo,
   const db2commexitBuffer       * pBuffer,
   db2int64                      * pReservedFlags,
   char                         ** errormsg,
   db2int32                      * errormsglen
);
```

db2commexitRecv API パラメーター

pConnectionContext
入力。通信バッファー出口ライブラリー固有のデータを指すポインター。このポインターは、インバウンド接続に固有のものです。このパラメーターは、その接続における各関数呼び出しの入力として渡されます。データベース・マネージャーは、このパラメーターによって指し示されるメモリーにはアクセスできません。このメモリーは、この関数で割り振り解除する必要があります。

pCommInfo
入力。データベース・サーバーを識別する情報、および着信接続のプロトコル固有の情報が含まれる構造を指すポインター。その構造内の一部のフィールドは、複数のバッファーがクライアントと交換されるまでセットアップされません。それらのフィールドは、db2commexitRecv と db2commexitSend を後で呼び出すときに使用できます。このシナリオは、特に **inbound_appl_id**、**outbound_appl_id**、および **connection_type** に適用されます。

pBuffer
入力。データベース・マネージャーが受信したバッファーの長さと、そのバッファーを指すポインターの両方を格納する構造を指すポインター。そのバッファーが暗号化されている場合には、この関数が呼び出される前に暗号化解除されます。

pReservedFlags
入出力。ビット DB2COMMEXIT_RECV_IN_FLAG_END_DECRYPT が設定され、暗号化された DSS 用にこの関数が呼び出されるのはこれが最後であることが示されます。入力として渡される DSS の長さは、暗号化された DSS の長さを示します。ただし、DSS はその後暗号化解除され、埋め込みは削除されます。埋め込みは必ず存在するため、DSS の長さは示された長さより短くなります。 pBuffer 構造で示される長さが DSS の最終データです。ブロック・サイズ全体と同じ長さの埋め込みが追加されている場合、この値がゼロになる可能性もあります。

出力の場合、この値は将来の使用のために予約されます。出力の場合、この値は 0 に設定される必要があります。

errormsg
出力。通信バッファー出口ライブラリーによって割り振られた ASCII エラー・メッセージ・ストリングのアドレスを指すポインター。このエラー・メッセー

ジ・ストリングは、関数が正常に実行されない場合にこのパラメーターで返されることがあります。このメモリーは、db2commexitFreeErrormsg を呼び出しても解放されません。

errormsglen
出力。**errormsg** パラメーターのエラー・メッセージ・ストリングのバイト単位の長さを示す整数を指すポインター。

db2commexitSend API - 送信

この関数は、データベース・マネージャーがクライアントに送信するバッファーごとに呼び出されます。

この関数は、クライアントに通信バッファーを送信する直前に、データベース・マネージャーによって呼び出されます。この関数はバッファーが暗号化される前に呼び出されるため、通信バッファー出口ライブラリーは、暗号化されていないバッファーにアクセスできます。

API ヘッダー・ファイル

```
db2commexit.h
```

API およびデータ構造構文

```
SQL_API_RC ( SQL_API_FN * db2commexitSend )
(
   void                          * pConnectionContext,
   const db2commexitCommInfo_v1  * pCommInfo,
   const db2commexitBuffer       * pBuffer,
   db2int64                      * pReservedFlags,
   char                         ** errormsg,
   db2int32                      * errormsglen
);
```

db2commexitSend API パラメーター

pConnectionContext
入力。通信バッファー出口ライブラリー固有のデータを指すポインター。このポインターは、インバウンド接続に固有のものです。このパラメーターは、その接続における各関数呼び出しの入力として渡されます。データベース・マネージャーは、このパラメーターによって指し示されるメモリーにはアクセスできません。

pCommInfo
入力。データベース・サーバーを識別する情報、および着信接続のプロトコル固有の情報が含まれる構造を指すポインター。その構造内の一部のフィールドは、複数のバッファーがクライアントと交換されるまでセットアップされません。それらのフィールドは、db2commexitRecv と db2commexitSend を後で呼び出すときに使用できます。このシナリオは、特に **inbound_appl_id**、**outbound_appl_id**、および **connection_type** に適用されます。

pBuffer
入力。クライアントに送信されるバッファーの長さと、そのバッファーを指すポインターの両方を格納する構造を指すポインター。そのバッファーが暗号化されている場合には、この関数が呼び出される前に暗号化解除されます。

pReservedFlags
入出力。データベース・マネージャーがエラーを検出し、クライアントへの送信用に準備したいくつかのバッファーをパージする必要がある場合、ビット DB2COMMEXIT_SEND_IN_FLAG_PURGE が設定されます。これらのバッファーの一部は、通信バッファー出口ライブラリーに入力として渡されている可能性があります。

出力の場合、この値は将来の使用のために予約されます。出力の場合、この値は 0 に設定される必要があります。

errormsg
出力。通信バッファー出口ライブラリーによって割り振られた ASCII エラー・メッセージ・ストリングのアドレスを指すポインター。このエラー・メッセージ・ストリングは、関数が正常に実行されない場合にこのパラメーターで返されることがあります。このメモリーは、db2commexitFreeErrormsg を呼び出しても解放されません。

errormsglen
出力。**errormsg** パラメーターのエラー・メッセージ・ストリングのバイト単位の長さを示す整数を指すポインター。

db2commexitUserIdentity API - ユーザー ID

この関数は、現行ソケットのユーザーの ID を提供するために呼び出されます。

この関数は、接続の確立に使用されるユーザー名とセッション許可 ID を通信バッファー出口ライブラリーに通知するために呼び出されます。この関数はまた、トラステッド・コンテキストのユーザー切り替えまたは SET SESSION AUTHORIZATION が実行されることでパラメーターが変更された場合にも呼び出されます。ユーザー名およびセッション許可 ID は、データベース・マネージャーがそのユーザーを認証するまで判別されません。この関数は、db2commexitRegister 関数と複数の db2commexitSend 関数および db2commexitRecv 関数が呼び出されて認証を行うまで、呼び出されません。

API ヘッダー・ファイル

db2commexit.h

API およびデータ構造構文

```
SQL_API_RC ( SQL_API_FN * db2commexitUserIdentity )
(
   void                          * pConnectionContext,
   const db2commexitCommInfo_v1  * pCommInfo,
   db2int32                        state,
   db2int32                        usernameLen,
   const char                    * pUserame,
   db2int32                        sessionAuthidLen,
   const char                    * pSessionAuthid,
   db2int64                      * pReservedFlags,
   char                         ** errormsg,
   db2int32                      * errormsglen
);
```

db2commexitUserIdentity API パラメーター

pConnectionContext
入力。通信バッファー出口ライブラリー固有のデータを指すポインター。このポインターは、インバウンド接続に固有のものです。このパラメーターは、その接続における各関数呼び出しの入力として渡されます。データベース・マネージャーは、このパラメーターによって指し示されるメモリーにはアクセスしません。

pCommInfo
入力。データベース・サーバーを識別する情報、および着信接続のプロトコル固有の情報が含まれる構造を指すポインター。その構造内の一部のフィールドは、複数のバッファーがクライアントと交換されるまでセットアップされません。それらのフィールドは、db2commexitRecv と db2commexitSend を後で呼び出すときに使用できます。このシナリオは、特に **inbound_appl_id**、**outbound_appl_id**、および **connection_type** に適用されます。

State
入力。この関数が呼び出される条件を示します。可能な値は以下のとおりです。

- DB2COMMEXIT_USERIDENT_NEW_CONNECTION - 新規接続。
- DB2COMMEXIT_USERIDENT_TC_SWITCH_USER - トラステッド・コンテキストのユーザー切り替えが実行される。
- DB2COMMEXIT_USERIDENT_SET_SESSION_USER - SET SESSION AUTHORIZATION SQL ステートメントが実行されて、セッション許可 ID が変更される。

usernameLen
入力。 **pUsername** の長さ。

pUsername
入力。接続の確立に使用されるユーザー名。

sessionAuthidLen
入力。 **pSessionAuthid** の長さ。

pSessionAuthid
入力。この接続用に設定されたセッション許可 ID。

pReservedFlags
入出力。将来の使用のために予約されています。出力の場合、この値は 0 に設定される必要があります。

errormsg
出力。通信バッファー出口ライブラリーによって割り振られた ASCII エラー・メッセージ・ストリングのアドレスを指すポインター。このエラー・メッセージ・ストリングは、関数が正常に実行されない場合にこのパラメーターで返されることがあります。このメモリーは、db2commexitFreeErrormsg を呼び出しても解放されません。

errormsglen
出力。**errormsg** パラメーターのエラー・メッセージ・ストリングのバイト単位の長さを示す整数を指すポインター。

db2commexitFreeErrormsg API - エラー・メッセージのメモリーの解放

この関数は、直前の API 呼び出しのエラー・メッセージを保持するために、使用されているメモリーを解放します。

API ヘッダー・ファイル

db2commexit.h

API およびデータ構造構文

```
SQL_API_RC ( SQL_API_FN * db2commexitFreeErrormsg )
 ( char * errormsg );
```

db2commexitFreeErrormsg API パラメーター

errormsg
入力。以前の API 呼び出しで返されたエラー・メッセージを指すポインター。

通信バッファー出口ライブラリーの関数構造

db2commexitInit 関数は、void * commexit_fns パラメーターを使用します。このパラメーターは、通信バッファー出口ライブラリーによってインプリメントされたすべての関数が含まれる、バージョン固有の構造にキャストされます。
db2commexitInit 関数は、関数ポインターを割り当てる必要があります。そうすることで、データベース・マネージャーは、それらの関数を呼び出すことができます。

作成する必要のある構造 (各 API を指す関数ポインターを含む) は、以下のとおりです。

```
struct db2commexitFunctions_v1
{
    db2int32 version;

    SQL_API_RC ( SQL_API_FN * db2commexitTerm )
    (
       char     **errormsg,
       db2int32 *errormsglen
    );

    SQL_API_RC ( SQL_API_FN * db2commexitRegister )
    (
       void                          ** ppConnectionContext,
       const db2commexitCommInfo_v1  * pCommInfo,
       db2int32                        state,
       db2int64                      * pReservedFlags,
       char                          ** errormsg,
       db2int32                      * errormsglen
    );

    SQL_API_RC ( SQL_API_FN * db2commexitDeregister )
    (
       void                          * pConnectionContext,
       const db2commexitCommInfo_v1  * pCommInfo,
       db2int32                        state,
       db2int64                      * pReservedFlags,
       char                          ** errormsg,
       db2int32                      * errormsglen
```

```
    );

    SQL_API_RC ( SQL_API_FN * db2commexitRecv )
    (
       void                          * pConnectionContext,
       const db2commexitCommInfo_v1  * pCommInfo,
       const db2commexitBuffer       * pBuffer,
       db2int64                      * pReservedFlags,
       char                         ** errormsg,
       db2int32                      * errormsglen
    );

    SQL_API_RC ( SQL_API_FN * db2commexitSend )
    (
       void                          * pConnectionContext,
       const db2commexitCommInfo_v1  * pCommInfo,
       const db2commexitBuffer       * pBuffer,
       db2int64                      * pReservedFlags,
       char                         ** errormsg,
       db2int32                      * errormsglen
    );

    SQL_API_RC ( SQL_API_FN * db2commexitUserIdentity )
    (
       void                          * pConnectionContext,
       const db2commexitCommInfo_v1  * pCommInfo,
       db2int32                        state,
       db2int32                        usernameLen,
       const char                    * pUserame,
       db2int32                        sessionAuthidLen,
       const char                    * pSessionAuthid,
       db2int64                      * pReservedFlags,
       char                         ** errormsg,
       db2int32                      * errormsglen
    );

    SQL_API_RC ( SQL_API_FN * db2commexitFreeErrormsg )
    (
       char * errormsg
    );

};
```

通信バッファー出口ライブラリーの情報構造

情報構造は、現行の物理接続の通信プロトコル情報を示します。

各通信バッファー出口ライブラリー関数に渡される db2commexitCommInfo_v1 構造は以下のとおりです。この構造は、db2commexit.h ファイルに組み込まれています。

```
struct db2commexitIPV4Info
{
   sockaddr_in client_sockaddr;
   sockaddr_in server_sockaddr;
};

struct db2commexitIPV6Info
{
   sockaddr_in6 client_sockaddr;
   sockaddr_in6 server_sockaddr;
};

struct db2commexitIPCInfo
{
   void * pSharedMemSegmentHandle;
```

```
};

struct db2commexitNamedPipeInfo
{
   void * handle;
};


struct db2commexitCommInfo_v1
{
   db2int32 clientProtocol;   // SQL_PROTOCOL_ ...
   db2int32 connectionType; // unknown, local or gateway

   db2int32 hostnameLen;
   db2int32 instanceLen;
   db2int32 dbnameLen;
   db2int32 dbaliasLen;
   db2int32 inbound_appl_id_len;
   db2int32 outbound_appl_id_len;

   db2int32 reserved1;
   db2int32 reserved2;

   db2NodeType member;

   char hostname[SQL_HOSTNAME_SZ+1];
   char instance[DB2COMMEXIT_INSTANCE_SZ + 1];
   char dbname[DB2COMMEXIT_DBNAME_SZ + 1];
   char dbalias[DB2COMMEXIT_DBNAME_SZ + 1];
   char inbound_appl_id[SQLM_APPLID_SZ + 1];
   char outbound_appl_id[SQLM_APPLID_SZ + 1];

   char reservedChar1[128];

   union
   {
      db2commexitIPV4Info ipv4Info;
      db2commexitIPV6Info ipv6Info;
      db2commexitIPCInfo  ipcInfo;
      db2commexitNamedPipeInfo namedPipeInfo;
   }
};
```

通信バッファー出口ライブラリーのバッファー構造

バッファー構造は、db2commexitSend 関数および db2commexitRecv 関数に入力として渡される構造です。

バッファー構造は以下のとおりです。

```
struct db2commexitBuffer
{
   const unsigned char * pBuffer;
   db2int64 buffer_len;

   db2int32 reserved1;
   db2int32 reserved2;
};
```

通信バッファー出口ライブラリーによる接続の制御

通信バッファー出口ライブラリーは、クライアントへの接続をいつでも強制的にドロップできます。

通信バッファー出口ライブラリーが db2commexitUserIdentity、db2commexitRegister、db2commexitDeregister、db2commexitRecv、または db2commexitSend のいずれかの呼び出しに関して該当するエラー戻りコードを返すと、データベース・マネージャーは直ちにそのクライアントとの接続をクローズします。

この機能を使用すると、通信バッファー出口ライブラリーが、検討対象バッファーに基づいて、何らかの不適切なアクティビティーが行われているかどうかを判断できます。そのように判断されると、それ以降、その接続に対するデータベース・マネージャーによるアクションをすべて回避できます。

通信バッファー出口ライブラリー API のバージョン

DB2 データベース・システムは、通信バッファー出口ライブラリー API のバージョン番号付け機能をサポートしています。そのようなバージョン番号は、1 で始まる整数になります。

データベース・マネージャーがセキュリティー・ライブラリーの初期化関数に渡すバージョン番号は、その API のサポートされている最高のバージョン番号になります。ライブラリーがそれより高い API バージョンをサポートできる場合は、データベース・マネージャーが要求するバージョンの関数ポインターを返す必要があります。ライブラリーがそれより低い API バージョンしかサポートしていない場合、そのライブラリーは、サポートしているバージョン用の関数ポインターを定義する必要があります。どちらのケースでも、ライブラリーの初期化関数は、サポートしている API のバージョン番号を、関数構造のバージョン・フィールドに入れて返す必要があります。

通信バッファー出口ライブラリー API のバージョン番号は、必要な場合にのみ変更されます。例えば、API のパラメーターが変更された場合などです。バージョン番号は、データベース・マネージャーのリリース番号を使って自動的に変更されることはありません。

バージョン番号があることによって、新しい、または変更された API を導入することができます。古いバージョンに対するライブラリーのサポートは維持されます。

通信バッファー出口ライブラリーのエラー処理と戻りコード

通信バッファー出口ライブラリー API でエラーが発生した場合、その API は、**errormsg** フィールドで ASCII テキスト・ストリングを返すことができます。その ASCII テキスト・ストリングには、戻りコードよりも具体的な問題の説明が示されます。データベース・マネージャーは、このストリング全体を db2diag ログ・ファイルに書き込みます。

これらのエラー・メッセージ用のメモリーは、通信バッファー出口ライブラリーによって割り振られる必要があります。したがって、ライブラリーは、このメモリーを解放するための API である db2commexitFreeErrormsg を備えていなければなりません。

errormsg フィールドに加えて、初期化時には、メッセージ・ロギング関数ポインター logMessage_fn が、通信バッファー出口ライブラリーに渡されます。ライブラリーはこの関数を使用してデバッグ情報を db2diag ログ・ファイルに記録できます。以下に例を示します。

```
// Log an message indicate init successful
  (*(logMessage_fn))(DB2COMMEXIT_LOG_CRITICAL,
                     "comm exit initialization successful",
                      strlen("comm. exit initialization successful"));
```

db2secLogMessage 関数の各パラメーターについて詳しくは、関連資料に記載されている、初期化 API db2commexitInit の説明を参照してください。

戻りコード

表 40. 通信バッファー出口ライブラリーがデータベース・マネージャーに返すことができる戻りコード

戻りコード	定義値	詳細
0	DB2COMMEXIT_SUCCESS	正常に実行されました。
-1	DB2COMMEXIT_ERR_UNKNOWN	ライブラリーで想定外のシステム・エラーが発生しました。
-2	DB2COMMEXIT_ERR_DROP_CONNECTION	呼び出されたライブラリーは、対象となる接続を終了する必要があると判断しました。

通信バッファー出口ライブラリーの作成における制約事項

通信バッファー出口ライブラリーの作成において、いくつかの制約事項および考慮事項があります。

制約事項

C-linkage

通信バッファー出口ライブラリーは C/C++ で作成し、C リンケージでリンクする必要があります。プロトタイプを提供するヘッダー・ファイル、ライブラリーのインプリメントに必要なデータ構造、およびエラー・コード定義は、C/C++ の場合のみ提供されます。ライブラリーが C++ としてコンパイルされている場合、関数 db2commexitInit は extern "C" として宣言する必要があります。

シグナル・ハンドラー

通信バッファー出口ライブラリーでは、シグナル・ハンドラーのインストールやシグナル・マスクの変更を行ってはなりません。これを行うと、データベース・マネージャーのシグナル・ハンドラーの動作を妨害してしまいます。データベース・マネージャーのシグナル・ハンドラーの動作を妨害すると、エラーを報告してその状態から復旧する機能の重大な妨げとなる可能性があります。

例外

通信バッファー出口ライブラリー API では、C++ 例外がスローされないようにしてください。それらの例外は、データベース・マネージャーのエラー処理の妨げになる可能性があります。

スレッド・セーフ

通信バッファー出口ライブラリーの関数は、スレッド・セーフになっている必要があります。その必要がない API は、db2commexitInit 関数および db2commexitTerm 関数のみです。

終了ハンドラー

通信バッファー出口ライブラリーでは、終了ハンドラーや pthread_atfork ハンドラーをインストールしてはなりません。通信バッファー出口ライブラリーは、データベース・マネージャーのプロセスが終了する前にアンロードされるため、終了ハンドラーの使用はサポートされていません。

フォーク/スレッド

通信バッファー出口ライブラリーでは、フォークの呼び出しや新規スレッドの作成を行ってはなりません。これを行うと、データベース・マネージャーで、トラップなどの定義されていない振る舞いが発生する可能性があります。

ライブラリーの従属関係

Linux および UNIX では、通信バッファー出口ライブラリーは、setuid または setgid のプロセスからロードされます。このライブラリーは、**LD_LIBRARY_PATH**、**SHLIB_PATH**、または **LIBPATH** の各環境変数を使って従属ライブラリーを検索することはできません。したがって、そのライブラリーは、追加のライブラリーには依存しないようにする必要があります。ただし以下のように、他の方式を使ってアクセス可能な従属ライブラリーがある場合は除きます。

- /lib または /usr/lib に存在する。
- それらが常駐するディレクトリーを OS ワイド (Linux における ld.so.conf ファイル内など) で指定する。
- そのライブラリー自体の RPATH で指定する。

シンボルの重複

可能であれば、通信バッファー出口ライブラリーは、シンボルが競合する可能性を減らす方法として選択可能なオプションを用いてコンパイルおよびリンクします。その方法として、アンバインドされた外部シンボル参照を削減するオプションなどがあります。例えば、HP、Solaris、および Linux 上で "-Bsymbolic" リンカー・オプションを使用するなら、シンボルの重複に関係した問題を防ぐことができます。ただし、AIX で作成されたライブラリーの場合は、"-brtl" リンカー・オプションは明示的にも暗黙的にも使用しないでください。

32 ビットと 64 ビットに関する考慮事項

データベース・マネージャーには 32 ビットと 64 ビットの両方のバージョンがあり、プラットフォームによって異なります。 32 ビットの通信バッファー出口ライブラリーは、32 ビットのデータベース・マネージャーで使用可能にする必要があり、64 ビットの通信バッファー出口ライブラリーは、64 ビットのデータベース・マネージャーで使用可能にする必要があります。これらを混用することはできません。

ストアード・プロシージャー、トリガー、およびその他の内部 SQL

サーバーと相互作用するストアード・プロシージャーは、通信バッファー出口ライブラリーに渡されます。その相互作用のほとんどは、標準の通信チャネルでは発生せず、出口ライブラリーに使用されるモデルには適合しません。同様に、トリガー、およびその他の内部 SQL のソースは、標準の通信チャネルを通過せず、通信バッファー出口ライブラリーには渡されません。

通信バッファーの操作の禁止

通信バッファー出口ライブラリーでは、それが渡されるバッファーの操作または変更を行わないようにする必要があります。

ローリング更新のサポート

DB2 for Linux, UNIX, and Windows は、DB2 pureScale 環境に含まれる個々のメンバーのフィックスパック・レベルを、他のメンバーを停止せずに更新する機能をサポートしています。これは、ローリング更新と呼ばれます。同様に、通信バッファー出口ライブラリーのデプロイメント・セクションで概説したように、個々のメンバーで使用されるライブラリーのレベルを更新することも可能です。2 つの異なるバージョンの通信バッファー出口ライブラリーを、フィックスパックのレベルが違う可能性のある 2 つの異なるメンバーで同時に実行することができます。通信バッファー出口ライブラリーは、エラーを発生させずに、こうした状態を許容する必要があります。

通信バッファー出口ライブラリー API 呼び出しシーケンス

API 呼び出しシーケンスは、特定のシナリオごとに異なる可能性があります。

以下のトピックでは、通信バッファー出口ライブラリーの作成時に注意する必要がある特定のシナリオについて概説しています。これらのトピックは、ご使用の環境に最も適した呼び出しシーケンスを決める際に役立ちます。

API 呼び出しシーケンス - 単一エージェントの通常接続

最も典型的なシナリオは、データベース・マネージャーへのクライアント接続で、SQL をいくつか発行してから切断します。

この場合、単一のエージェントまたはスレッドが接続を処理し、以下の呼び出しが行われます。

1. 新しいソケット接続用の `db2commexitRegister`。
2. 認証を処理する `db2commexitRecv` および `db2commexitSend` (複数回の場合もあります)。
3. 新しい接続用の `db2commexitUserIdentity`。
4. クライアントの SQL 要求を処理する `db2commexitRecv` および `db2commexitSend` (複数回の場合もあります)。
5. ソケット接続を終了する `db2commexitDeregister`。

API 呼び出しシーケンス - 接続の初期化を行わない接続

このシナリオは、既存のソケットを使用した接続を対象としています。クライアントは、最初に接続の初期化を実行せずに、別の SQL 接続を開始する場合があります。

データベース・マネージャーがクライアントから SQL connect ステートメントを受け取ると、接続を続行する前に、内部接続の初期化を暗黙実行します。これは通常の要求および応答で、ソケットの状況は変更されません。この場合、単一のエージェントがすべての要求を処理します。クライアントからの接続要求が含まれているバッファーは、`db2commexitRecv` によって使用可能になるため、通信バッファー出口ライブラリーは、バッファーの構文解析時に開始された新しい接続を判別することができます。以下の呼び出しが行われます。

1. 新しいソケット接続用の `db2commexitRegister`。
2. 認証を処理する `db2commexitRecv` および `db2commexitSend` (複数回の場合もあります)。

3. 新しい接続用の db2commexitUserIdentity。
4. クライアントの SQL 要求を処理する db2commexitRecv および db2commexitSend (複数回の場合もあります)。
5. 認証を処理する db2commexitRecv および db2commexitSend (複数回の場合もあります)。
6. 新しい接続用の db2commexitUserIdentity。
7. クライアントの SQL 要求を処理する db2commexitRecv および db2commexitSend (複数回の場合もあります)。
8. ソケット接続を終了する db2commexitDeregister。

注:
db2commexitRegister および db2commexitDeregister は、データベース・マネージャーが 2 つの SQL 接続を処理した場合でも、それぞれ 1 回のみ呼び出されます。

API 呼び出しシーケンス - トラステッド・コンテキストとユーザー切り替え

このシナリオは、接続の初期化を行わない接続と似ています。相違点は、クライアントが新しい SQL 接続要求を送信するのではなく、トラステッド・コンテキストのユーザー切り替えを要求することです。

以下の呼び出しが行われます。

1. 新しいソケット接続用の db2commexitRegister。
2. 認証を処理する db2commexitRecv および db2commexitSend (複数回の場合もあります)。
3. 新しい接続用の db2commexitUserIdentity。
4. クライアントの SQL 要求を処理する db2commexitRecv および db2commexitSend (複数回の場合もあります)。
5. 認証を処理する db2commexitRecv および db2commexitSend (複数回の場合もあります)。

   将来のどこかの時点で、クライアントはトラステッド・コンテキストのユーザー切り替え要求をサーバーに送信して、接続のユーザーを切り替えます。
6. トラステッド・コンテキストのユーザー切り替え用の db2commexitUserIdentity。
7. クライアントの SQL 要求を処理する db2commexitRecv および db2commexitSend (複数回の場合もあります)。
8. ソケット接続を終了する db2commexitDeregister。

API 呼び出しシーケンス - 接続コンセントレーター

このシナリオは、接続コンセントレーターが使用される場合の API 呼び出しシーケンスを対象としています。接続コンセントレーター・フィーチャーは、コーディネーター・エージェントやスレッドよりもかなり多くのクライアントをデータベース・マネージャーが処理できるようにします。

クライアントが作業単位の境界に到達してすぐに別の要求を送信しないと、クライアント・ソケットはアイドル・プールに入れられます。以前にクライアント要求を処理したエージェントは、別のクライアントに移動します。読み取るべきデータがアイドル・ソケットに入ると、ディスパッチャーはそのデータを処理するアイドル・エージェントを検索します。 1 つの SQL 接続の存続期間全体において、複数のエージェントがクライアント要求を処理する可能性があります。アイドル・プールにソケットを出し入れするたびに、db2commexitDeregister および db2commexitRegister が呼び出されます。以下の呼び出しが行われます。

1. 新しいソケット接続用の db2commexitRegister。
2. 認証を処理する db2commexitRecv および db2commexitSend (複数回の場合もあります)。
3. 新しい接続用の db2commexitUserIdentity。
4. クライアントの SQL 要求を処理する db2commexitRecv および db2commexitSend (複数回の場合もあります)。

   クライアントは直ちに別の要求を送信することはせず、ソケットはアイドル・プールに入れられます。
5. エージェントとの関連付けを解除する db2commexitDeregister。

   将来のどこかの時点で、クライアントは次のような別の要求を送信します。ディスパッチャーはそのときにアイドル・エージェントを選択しますが、そのエージェントはおそらく、以前に使用されたものとは異なります。
6. エージェントを関連付ける db2commexitRegister。
7. クライアントの SQL 要求を処理する db2commexitRecv および db2commexitSend (複数回の場合もあります)。
8. ソケット接続を終了する db2commexitDeregister。

注: 1 つの SQL 接続に対して、db2commexitRegister および db2commexitDeregister の呼び出しは複数存在します。

API 呼び出しシーケンス - SET SESSION AUTHORIZATION ステートメント

このシナリオは、SET SESSION AUTHORIZATION ステートメントが使用される場合の API 呼び出しシーケンスを対象としています。

SET SESSION AUTHORIZATION ステートメントは、現行の接続に使用されているセッション許可 ID を変更します。現行の接続用の ID 情報が変更されたことを通信バッファー出口ライブラリーに通知するために、Db2commexitUserIdentity が呼び出されます。以下の呼び出しが行われます。

1. 新しいソケット接続用の db2commexitRegister。
2. 認証を処理する db2commexitRecv および db2commexitSend (複数回の場合もあります)。
3. 新しい接続用の db2commexitUserIdentity。
4. クライアントの SQL 要求を処理する db2commexitRecv および db2commexitSend (複数回の場合もあります)。

ユーザーが SET SESSION AUTHORIZATION ステートメントを発行します。この要求は、db2commexitRecv に渡されます。これと他の SQL ステートメントとの違いはありません。

5. SET SESSION AUTHORIZATION 用の db2commexitUserIdentity。
6. クライアントの SQL 要求を処理する db2commexitRecv および db2commexitSend (複数回の場合もあります)。
7. ソケット接続を終了する db2commexitDeregister。

ターゲットの論理ノードの設定に関する考慮事項

DB2NODE 変数または **SET CLIENT** コマンドを使用してターゲットの論理ノードを設定する場合には、注意の必要な考慮事項があります。

パーティション・データベース環境では、クライアントにおいて、その接続先として構成されていないメンバーを *DB2NODE* 変数で指定すると、データベース・マネージャーは、その変数で指定された新規メンバーに接続先を切り替えます。クライアント接続は、接続されたメンバーを介してリモート・メンバーに転送されます。この場合、通信バッファー出口ライブラリーは、両方のメンバーで呼び出されます。注意の必要なフィーチャーがいくつかあります。

- 接続先のメンバーにおけるクライアント・アドレスには、実際のクライアントが反映されます。
- リモート・メンバーにおけるクライアント・アドレスには、接続先のメンバーが反映されます。
- 接続先のメンバーにおけるアウトバウンド・アプリケーション ID は、リモート・メンバーにおけるインバウンド・アプリケーション ID と同じになります。

アプリケーション ID が設定されると、db2commexitCommInfo_v1 構造内の **connectionType** は、GATEWAY に設定されます。

接続ゲートウェイに関する考慮事項

データベース・マネージャーが別の DRDA データベース・サーバーへの接続ゲートウェイとして機能する場合には、注意の必要な考慮事項があります。

DB2 for Linux, UNIX, and Windows が接続ゲートウェイとして機能する場合、通信バッファー出口ライブラリーは、標準的な接続と同じ方法で呼び出されます。認証が完了し、アプリケーション ID が設定されると、db2commexitCommInfo_v1 構造内の **connectionType** は、GATEWAY に設定されます。 outbound_application_id は、DRDA データベース・サーバーにおける、その接続用のアプリケーション ID と一致します。

DATA_ENCRYPT に関する考慮事項

DATA_ENCRYPT 認証タイプが使用される場合には、注意の必要な考慮事項があります。

認証タイプ DATA_ENCRYPT で保護された通信の処理については、特別な考慮事項があります。 SSL とは異なり、DATA_ENCRYPT をサポートするために必要な暗号化および暗号化解除は、クライアントからの受信後、クライアントに応答を送信する前に、データベース・マネージャーによって行われます。

受信と DATA_ENCRYPT

暗号化された DSS をクライアントから受信すると、バッファーは必要に応じてデータベース・マネージャーによって暗号化解除されます。つまり、バッファー全体が一度に暗号化解除されるわけではありません。通信バッファー出口ライブラリーは、データの暗号化解除時に、暗号化解除されたデータを使用して呼び出されます。

DSS の長さ (DSS が論理レコードより長い場合は DSS の継続の長さ) には、暗号化された DSS は含まれ、暗号化解除されたバッファーの長さは含まれません。暗号化では常に埋め込みが追加されるため、この長さは必ず、プレーン・テキストの長さよりも大きくなります。 DSS への埋め込みの長さは、最大 8 バイトです。

db2CommexitRecv への最後の呼び出しが行われると、
DB2COMMEXIT_RECV_IN_FLAG_END_DECRYPT フラグが入力として渡され、暗号化された DSS の終わりであることが示されます。

注: その場合、長さは 0 になって、ブロック・サイズ全体と同じ長さの埋め込みの追加が示される可能性があります。

送信と DATA_ENCRYPT

クライアントへの DSS 応答を暗号化する場合、クライアントに送信されるバッファーは、複数のプレーン・テキスト DSS と暗号化された DSS によって構成される可能性があります。これらの DSS が準備されると、入力として db2commexitSend ルーチンに渡されます。暗号化前の入力にはプレーン・テキスト・データを使用する必要があるため、これらの受け渡しは一度に 1 つずつ行われます。データベース・マネージャーは、準備済みではあるが送信されていない DSS のパージが必要となるようなエラー状態に遭遇する場合があります。通信バッファー出口ライブラリーは、既にこれらのライブラリーを認識している可能性があります。
db2CommexitSend が長さ 0 で呼び出され、パージが行われたことがフラグ
DB2COMMEXIT_SEND_IN_FLAG_PURGE 関数によって示されます。

第 11 章 監査機能のレコード・レイアウト

監査ログから監査レコードを抽出する際、各レコードは、以下の表に示されているいずれかのフォーマットになります。各表の前には、サンプル・レコードを示します。

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。表の中で、各項目は、抽出操作後に区切りファイルに出力されるときと同じ順序で示してあります。

注:

1. 監査イベントによっては、監査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。フィールドに値がない場合、そのフィールドは監査出力に表示されません。
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。

監査レコード・オブジェクト・タイプ

次の表は、監査レコード・オブジェクト・タイプ別に、各タイプで CHECKING、OBJMAINT、および SECMAINT イベントが生成される可能性があるかどうかを示しています。

表 41. 監査イベントに基づく監査レコード・オブジェクト・タイプ

オブジェクト・タイプ	CHECKING イベント	OBJMAINT イベント	SECMAINT イベント
ACCESS_RULE			X
ALIAS	X	X	
ALL	X		
AUDIT_POLICY	X	X	
BUFFERPOOL	X	X	
CHECK_CONSTRAINT		X	
DATABASE	X		X
DATA TYPE		X	
EVENT_MONITOR	X	X	
FOREIGN_KEY		X	
FUNCTION	X	X	X
FUNCTION MAPPING	X	X	
GLOBAL_VARIABLE	X	X	X
HISTOGRAM TEMPLATE	X	X	
INDEX	X	X	X
INDEX EXTENSION		X	

表 41. 監査イベントに基づく監査レコード・オブジェクト・タイプ (続き)

オブジェクト・タイプ	CHECKING イベント	OBJMAINT イベント	SECMAINT イベント
INSTANCE	X		
JAR_FILE		X	
METHOD_BODY	X	X	X
MODULE	X	X	X
NICKNAME	X	X	X
NODEGROUP	X	X	
NONE	X	X	X
OPTIMIZATION PROFILE	X		
PACKAGE	X	X	X
PACKAGE CACHE	X		
PRIMARY_KEY		X	
REOPT_VALUES	X		
ROLE	X	X	X
SCHEMA	X	X	X
SECURITY LABEL		X	X
SECURITY LABEL COMPONENT		X	
SECURITY POLICY		X	X
SEQUENCE	X	X	
SERVER	X	X	X
SERVER OPTION	X	X	
SERVICE CLASS	X	X	
STORED_PROCEDURE	X	X	X
SUMMARY TABLES	X	X	X
TABLE	X	X	X
TABLESPACE	X	X	X
THRESHOLD	X	X	
TRIGGER		X	
TRUSTED CONTEXT	X	X	X
TYPE MAPPING	X	X	
TYPE&TRANSFORM	X	X	
UNIQUE_CONSTRAINT		X	
USER MAPPING	X	X	
USER_TEMPORARY_TABLE	X	X	X
VIEW	X	X	X
WORK ACTION SET	X	X	
WORK CLASS SET	X	X	
WORKLOAD	X	X	X
WRAPPER	X	X	
XSR オブジェクト	X	X	X

AUDIT イベントの監査レコード設計

次の表は、AUDIT イベントの監査レコード設計を示しています。

以下に監査レコードのサンプルを示します。

```
timestamp=2007-04-10-08.29.52.000001;
category=AUDIT;
audit event=START;
event correlator=0;
event status=0;
userid=newton;
authid=NEWTON;
application id=*LOCAL_APPLICATION;
application name=db2audit.exe;
```

表 42. AUDIT イベントの監査レコード設計

名前	フォーマット	説明
Timestamp	CHAR(26)	監査イベントの日付と時刻。
カテゴリー	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。 可能な値のリストについては、 380 ページの『監査イベント』の AUDIT カテゴリーのセクションを参照してください。
Event Correlator	INTEGER	監査対象の操作の相関 ID。単一イベントにどの監査レコードが関連している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。 成功イベント > = 0 失敗イベント < 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。
Original User ID	VARCHAR(1024)	監査イベントが発生した時刻の CLIENT_ORIGUSERID グローバル変数の値。
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であった場合にはブランクとなります。
Origin Node Number	SMALLINT	監査イベントが発生したメンバー番号。
Coordinator Node Number	SMALLINT	コーディネーター・メンバーのメンバー番号。
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	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。
Package Version	VARCHAR(64)	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のバージョン。

表 42. AUDIT イベントの監査レコード設計 (続き)

名前	フォーマット	説明
Local Transaction ID	VARCHAR(10) FOR BIT DATA	監査イベントが発生した時刻で使用していたローカル・トランザクション ID。これは、トランザクション・ログの一部となる SQLU_TID 構造体です。
Global Transaction ID	VARCHAR(30) FOR BIT DATA	監査イベントが発生した時刻で使用していたグローバル・トランザクション ID。これは、トランザクション・ログの一部となる SQLP_GXID 構造体のデータ・フィールドです。
Client User ID	VARCHAR(255)	監査イベントが発生した時刻の CURRENT CLIENT USERID 特殊レジスターの値。
Client Workstation Name	VARCHAR(255)	監査イベントが発生した時刻の CURRENT CLIENT_WRKSTNNAME 特殊レジスターの値。
Client Application Name	VARCHAR(255)	監査イベントが発生した時刻の CURRENT CLIENT_APPLNAME 特殊レジスターの値。
Client Accounting String	VARCHAR(255)	監査イベントが発生した時刻の CURRENT CLIENT_ACCTNG 特殊レジスターの値。
Trusted Context Name	VARCHAR(128)	トラステッド接続に関連付けられたトラステッド・コンテキストの名前。
Connection Trust Type	INTEGER	可能な値は以下のとおりです。 IMPLICIT_TRUSTED_CONNECTION EXPLICIT_TRUSTED_CONNECTION
Role Inherited	VARCHAR(128)	トラステッド接続を介して継承したロール。
Policy Name	VARCHAR(128)	監査ポリシー名。
Policy Association Object Type	CHAR(1)	監査ポリシーを関連付けるオブジェクトのタイプ。可能な値は以下のとおりです。 • N = ニックネーム • S = MQT • T = 表 (非型付き) • i = 許可 ID • g = 権限 • x = トラステッド・コンテキスト • ブランク = データベース
Policy Association Subobject Type	CHAR(1)	監査ポリシーを関連付けるサブオブジェクトのタイプ。オブジェクト・タイプが i (許可 ID) の場合、可能な値は以下のとおりです。 • U = ユーザー • G = グループ • R = ロール
Policy Association Object Name	VARCHAR(128)	監査ポリシーを関連付けるオブジェクトの名前。
Policy Association Object Schema	VARCHAR(128)	監査ポリシーを関連付けるオブジェクトのスキーマ名。「Policy Association Object Type」で、スキーマが適用されないオブジェクトが識別されている場合は、NULL になります。

表 42. AUDIT イベントの監査レコード設計 (続き)

名前	フォーマット	説明
Audit Status	CHAR(1)	監査ポリシーの AUDIT 区分の状況。可能な値は以下のとおりです。 • B- 両方 • F- 失敗 • N- なし • S- 成功
Checking Status	CHAR(1)	監査ポリシーの CHECKING 区分の状況。可能な値は以下のとおりです。 • B- 両方 • F- 失敗 • N- なし • S- 成功
Context Status	CHAR(1)	監査ポリシーの CONTEXT 区分の状況。可能な値は以下のとおりです。 • B- 両方 • F- 失敗 • N- なし • S- 成功
Execute Status	CHAR(1)	監査ポリシーの EXECUTE 区分の状況。可能な値は以下のとおりです。 • B- 両方 • F- 失敗 • N- なし • S- 成功
Execute With Data	CHAR(1)	監査ポリシーの EXECUTE 区分の WITH DATA オプション。可能な値は以下のとおりです。 • Y - WITH DATA • N - WITHOUT DATA
Objmaint Status	CHAR(1)	監査ポリシーの OBJMAINT 区分の状況。可能な値は以下のとおりです。 • B- 両方 • F- 失敗 • N- なし • S- 成功
Secmaint Status	CHAR(1)	監査ポリシーの SECMAINT 区分の状況。可能な値については、「Audit Status」フィールドを参照してください。

表 42. AUDIT イベントの監査レコード設計 (続き)

名前	フォーマット	説明
Sysadmin Status	CHAR(1)	監査ポリシーの SYSADMIN 区分の状況。可能な値は以下のとおりです。 • B- 両方 • F- 失敗 • N- なし • S- 成功
Validate Status	CHAR(1)	監査ポリシーの VALIDATE 区分の状況。可能な値は以下のとおりです。 • B- 両方 • F- 失敗 • N- なし • S- 成功
Error Type	CHAR(8)	監査ポリシーのエラー・タイプ。可能な値は AUDIT および NORMAL です。
Data Path	VARCHAR(1024)	**db2audit configure** コマンドで指定されたアクティブ監査ログのパス。
Archive Path	VARCHAR(1024)	**db2audit configure** コマンドで指定されたアーカイブされた監査ログのパス。

CHECKING イベントの監査レコード設計

次の表は、CHECKING イベントの監査レコードの形式を示しています。

以下に監査レコードのサンプルを示します。

```
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;
package schema=NULLID;
package name=SYSSH200;
package section=0;
object schema=GSTAGER;
object name=NONE;
object type=REOPT_VALUES;
access approval reason=DBADM;
access attempted=STORE;
```

表 43. CHECKING イベントの監査レコード設計

名前	フォーマット	説明
Timestamp	CHAR(26)	監査イベントの日付と時刻。
カテゴリー	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

表 43. CHECKING イベントの監査レコード設計 (続き)

名前	フォーマット	説明
Audit Event	VARCHAR(32)	特定の監査イベント。 可能な値のリストについては、 380 ページの『監査イベント』の CHECKING カテゴリーのセクションを参照してください。
Event Correlator	INTEGER	監査対象の操作の相関 ID。単一イベントにどの監査レコードが関連している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。 成功イベント > = 0 失敗イベント < 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であった場合に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。
Original User ID	VARCHAR(1024)	監査イベントが発生した時刻の CLIENT_ORIGUSERID グローバル変数の値。
Origin Node Number	SMALLINT	監査イベントが発生したメンバー番号。
Coordinator Node Number	SMALLINT	コーディネーター・メンバーのメンバー番号。
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)	監査イベントの生成対象となったオブジェクトのスキーマ。
Object Name	VARCHAR(128)	監査イベントの生成対象となったオブジェクトの名前。
Object Type	VARCHAR(32)	監査イベントの生成対象となったオブジェクトのタイプ。有効な値は、『監査レコード・オブジェクト・タイプ』というトピックに示されています。
Access Approval Reason	CHAR(34)	アクセスがこの監査イベントで承認された理由を示します。可能な値については、『有効な CHECKING アクセス承認理由のリスト』というトピックに示されています。
Access Attempted	CHAR(34)	試みられたアクセスのタイプを示します。可能な値については、『有効な CHECKING アクセス試行タイプのリスト』というトピックに示されています。
Package Version	VARCHAR (64)	監査イベントが発生した時点で使用されていたパッケージのバージョン。
Checked Authorization ID	VARCHAR(128)	許可 ID は監査イベント時の許可 ID と異なる場合にチェックされます。例えば、これは TRANSFER OWNERSHIP ステートメントのターゲット所有者などです。 監査イベントが SWITCH_USER である場合、このフィールドには、切り替え後の許可 ID が表示されます。

表 43. CHECKING イベントの監査レコード設計 (続き)

名前	フォーマット	説明
Local Transaction ID	VARCHAR(10) FOR BIT DATA	監査イベントが発生した時刻で使用していたローカル・トランザクション ID。これは、トランザクション・ログの一部となる SQLU_TID 構造体です。
Global Transaction ID	VARCHAR(30) FOR BIT DATA	監査イベントが発生した時刻で使用していたグローバル・トランザクション ID。これは、トランザクション・ログの一部となる SQLP_GXID 構造体のデータ・フィールドです。
Client User ID	VARCHAR(255)	監査イベントが発生した時刻の CURRENT CLIENT USERID 特殊レジスターの値。
Client Workstation Name	VARCHAR(255)	監査イベントが発生した時刻の CURRENT CLIENT_WRKSTNNAME 特殊レジスターの値。
Client Application Name	VARCHAR(255)	監査イベントが発生した時刻の CURRENT CLIENT_APPLNAME 特殊レジスターの値。
Client Accounting String	VARCHAR(255)	監査イベントが発生した時刻の CURRENT CLIENT_ACCTNG 特殊レジスターの値。
Trusted Context Name	VARCHAR(255)	トラステッド接続に関連付けられたトラステッド・コンテキストの名前。
Connection Trust Type	INTEGER	可能な値は以下のとおりです。 IMPLICIT_TRUSTED_CONNECTION EXPLICIT_TRUSTED_CONNECTION
Role Inherited	VARCHAR(128)	トラステッド接続を介して継承したロール。

CHECKING アクセス承認理由

次のリストは、有効な CHECKING アクセス承認理由を示しています。

監査レコードには複数のアクセス承認理由が含まれることがあります。例: `access approval reason=DATAACCESS,ACCESSCTRL;` 複数のアクセス承認理由が存在する場合、試行されたアクセスが承認されるには、ユーザーは定められたすべての権限と特権を持っていなければなりません。

0x00000000000000000000000000000001 ACCESS DENIED
アクセスは承認されません。その上、拒否されました。

0x00000000000000000000000000000002 SYSADM
アクセスは承認されます。アプリケーションまたはユーザーは SYSADM 権限を持ちます。

0x00000000000000000000000000000004 SYSCTRL
アクセスは承認されます。アプリケーションまたはユーザーは SYSCTRL 権限を持ちます。

0x00000000000000000000000000000008 SYSMAINT
アクセスは承認されます。アプリケーションまたはユーザーは SYSMAINT 権限を持ちます。

0x00000000000000000000000000000010 DBADM
アクセスは承認されます。アプリケーションまたはユーザーは DBADM 権限を持ちます。

0x00000000000000000000000000000020 DATABASE
アクセスは承認されます。アプリケーションまたは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。

0x00000000000000000000000000000040 OBJECT
アクセスは承認されます。アプリケーションまたはユーザーは、オブジェクトまたは関数に関する特権を持ちます。

0x00000000000000000000000000000080 DEFINER
アクセスは承認されます。アプリケーションまたはユーザーは、オブジェクトまたは関数の定義をするものとなります。

0x00000000000000000000000000000100 OWNER
アクセスは承認されます。アプリケーションまたはユーザーは、オブジェクトまたは関数の所有者となります。

0x00000000000000000000000000000200 CONTROL
アクセスは承認されます。アプリケーションまたはユーザーは、オブジェクトまたは関数に関する CONTROL 権限を持ちます。

0x00000000000000000000000000000400 BIND
アクセスは承認されます。アプリケーションまたはユーザーは、パッケージに関するバインド権限を持ちます。

0x00000000000000000000000000000800 SYSQUIESCE
アクセスは承認されます。インスタンスまたはデータベースが静止モードにある場合は、アプリケーションまたはユーザーは接続またはアタッチを行うことができます。

0x00000000000000000000000000001000 SYSMON
アクセスは承認されます。アプリケーションまたはユーザーは SYSMON 権限を持ちます。

0x00000000000000000000000000002000 SECADM
アクセスは承認されます。アプリケーションまたはユーザーは SECADM 権限を持ちます。

0x00000000000000000000000000004000 SETSESSIONUSER
アクセスは承認されます。アプリケーションまたはユーザーは SETSESSIONUSER 権限を持ちます。

0x00000000000000000000000000008000 TRUSTED_CONTEXT_MATCH
接続属性が、DB2 サーバーで定義されている固有のトラステッド・コンテキストの属性と一致しました。

0x00000000000000000000000000010000 TRUSTED_CONTEXT_USE
トラステッド・コンテキストを使用するためのアクセスが承認されます。

0x00000000000000000000000000020000 SQLADM
アクセスは承認されます。アプリケーションまたはユーザーは SQLADM 権限を持ちます。

0x00000000000000000000000000040000 WLMADM
アクセスは承認されます。アプリケーションまたはユーザーは WLMADM 権限を持ちます。

0x00000000000000000000000000080000 EXPLAIN
アクセスは承認されます。アプリケーションまたはユーザーは EXPLAIN 権限を持ちます。

0x00000000000000000000000000100000 DATAACCESS
アクセスは承認されます。アプリケーションまたはユーザーは DATAACCESS 権限を持ちます。

0x00000000000000000000000000200000 ACCESSCTRL
アクセスは承認されます。アプリケーションまたはユーザーは ACCESSSCTRL 権限を持ちます。

CHECKING アクセス試行タイプ

次のリストは、有効な CHECKING アクセス試行タイプを示しています。

監査イベントが CHECKING_TRANSFER の場合、監査項目は特権が保留されるかどうかを反映します。

0x00000000000000000000000000000001 CONTROL
CONTROL 特権が保留されるかどうかを検査しようとします。

0x00000000000000000000000000000002 ALTER
オブジェクトを変更しようとするか、または監査イベントが CHECKING_TRANSFER の場合に ALTER 特権が保留されるかどうかを検査しようとします。

0x00000000000000000000000000000004 DELETE
オブジェクトを削除しようとするか、または監査イベントが CHECKING_TRANSFER の場合に DELETE 特権が保留されるかどうかを検査しようとします。

0x00000000000000000000000000000008 INDEX
索引を使用しようとするか、または監査イベントが CHECKING_TRANSFER の場合に INDEX 特権が保留されるかどうかを検査しようとします。

0x00000000000000000000000000000010 INSERT
オブジェクトに挿入しようとするか、または監査イベントが CHECKING_TRANSFER の場合に INSERT 特権が保留されるかどうかを検査しようとします。

0x00000000000000000000000000000020 SELECT
表またはビューを照会しようとするか、または監査イベントが CHECKING_TRANSFER の場合に SELECT 特権が保留されるかどうかを検査しようとします。

0x00000000000000000000000000000040 UPDATE
オブジェクト内のデータを更新しようとするか、または監査イベントが CHECKING_TRANSFER の場合に UPDATE 特権が保留されるかどうかを検査しようとします。

0x00000000000000000000000000000080 REFERENCE
オブジェクト間の参照制約を確立しようとするか、または監査イベントが CHECKING_TRANSFER の場合に REFERENCE 特権が保留されるかどうかを検査しようとします。

0x00000000000000000000000000000100 CREATE
オブジェクトを作成しようとします。

0x00000000000000000000000000000200 DROP
オブジェクトをドロップしようとします。

0x00000000000000000000000000000400 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。

0x00000000000000000000000000000800 DROPIN
別のスキーマ内に見いだされるオブジェクトをドロップしようとします。

0x00000000000000000000000000001000 ALTERIN
別のスキーマ内に見いだされるオブジェクトを変更しようとします。

0x00000000000000000000000000002000 EXECUTE
アプリケーションを実行、またはルーチンを呼び出そうとします。ルーチンからのソース関数を作成する (関数のみに適用されます) か、何らかの DDL ステートメントのルーチンを参照するか、または監査イベントが CHECKING_TRANSFER の場合に EXECUTE 特権が保留されるかどうかを検査しようとします。

0x00000000000000000000000000004000 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。

0x00000000000000000000000000008000 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。

0x00000000000000000000000000010000 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。

0x00000000000000000000000000020000 COMMENT_ON
オブジェクトに関する注釈を作成しようとします。

0x00000000000000000000000000040000 GRANT
あるオブジェクトに対する特権またはロールを別の許可 ID に付与しようとします。

0x00000000000000000000000000080000 REVOKE
あるオブジェクトに対する特権またはロールを許可 ID から取り消そうとします。

0x00000000000000000000000000100000 LOCK
オブジェクトをロックしようとします。

0x00000000000000000000000000200000 RENAME
オブジェクトを名前変更しようとします。

0x00000000000000000000000000400000 CONNECT
データベースに接続しようとします。

0x00000000000000000000000000800000 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。

0x00000000000000000000000001000000 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)。

0x00000000000000000000000002000000 DROP ALL
複数のオブジェクトをドロップしようとします。

0x00000000000000000000000004000000 LOAD
表スペースに表をロードしようとします。

0x00000000000000000000000008000000 USE
表スペースに表を作成しようとするか、または監査イベントが CHECKING_TRANSFER の場合に USE 特権が保留されるかどうかを検査しようとします。

0x00000000000000000000000010000000 SET_SESSION_USER
SET SESSION_USER ステートメントを実行しようとします。

0x00000000000000000000000020000000 FLUSH
FLUSH ステートメントを実行しようとします。

0x00000000000000000000000040000000 STORE
EXPLAIN_PREDICATE 表内の再び最適化されたステートメントの値を表示しようとします。

0x00000000000000000000000100000000 SET_OWNER
パッケージのバインド時に、現行ユーザーと一致しない所有者を設定しようとします。

0x00000000000000000000000200000000 SET_PASSTHRU
SET PASSTHRU ステートメントを発行しようとします。

0x00000000000000000000000400000000 TRANSFER
オブジェクトを転送しようとします。

0x00000000000000000000000800000000 ALTER_WITH_GRANT
GRANT 特権の付いた ALTER が保留かどうかを検査しようとします。

0x00000000000000000000001000000000 DELETE_WITH_GRANT
GRANT 特権の付いた DELETE が保留かどうかを検査しようとします。

0x00000000000000000000002000000000 INDEX_WITH_GRANT
GRANT 特権の付いた INDEX が保留かどうかを検査しようとします。

0x00000000000000000000004000000000 INSERT_WITH_GRANT
GRANT 特権の付いた INSERT が保留かどうかを検査しようとします。

0x00000000000000000000008000000000 SELECT_WITH_GRANT
GRANT 特権の付いた SELECT が保留かどうかを検査しようとします。

0x00000000000000000000010000000000 UPDATE_WITH_GRANT
GRANT 特権の付いた UPDATE が保留かどうかを検査しようとします。

0x00000000000000000000020000000000 REFERENCE_WITH_GRANT
GRANT 特権の付いた REFERENCE が保留かどうかを検査しようとします。

0x00000000000000000000040000000000 USAGE
シーケンスまたは XSR オブジェクトを使用しようとするか、または監査イベントが CHECKING_TRANSFER の場合に USAGE 特権が保留されるかどうかを検査しようとします。

0x00000000000000000000080000000000 SET ROLE
ロールを設定しようとします。

0x00000000000000000000100000000000 EXPLICIT_TRUSTED_CONNECTION
明示的なトラステッド接続を確立しようとします。

0x00000000000000000000200000000000 IMPLICIT_TRUSTED_CONNECTION
暗黙的なトラステッド接続を確立しようとします。

0x00000000000000000000400000000000 READ
グローバル変数を読み取ろうとします。

0x00000000000000000000800000000000 WRITE
グローバル変数を書き込もうとします。

0x00000000000000000001000000000000 SWITCH_USER
明示的なトラステッド接続でユーザー ID を切り替えようとします。

0x00000000000000000002000000000000 AUDIT_USING
オブジェクトに監査ポリシーを関連付けようとします。

0x00000000000000000004000000000000 AUDIT_REPLACE
オブジェクトへの監査ポリシーの関連付けを置き換えようとします。

0x00000000000000000008000000000000 AUDIT_REMOVE
オブジェクトへの監査ポリシーの関連付けを除去しようとします。

0x00000000000000000010000000000000 AUDIT_ARCHIVE
監査ログをアーカイブしようとします。

0x00000000000000000020000000000000 AUDIT_EXTRACT
監査ログを抽出しようとします。

0x00000000000000000040000000000000 AUDIT_LIST_LOGS
監査ログをリストしようとします。

0x00000000000000000080000000000000 IGNORE_TRIGGERS
データベース・オブジェクトに関連付けられたトリガーを無視しようとします。

0x00000000000000000100000000000000 PREPARE
SQL ステートメントを準備しようとしますが、ユーザーは必要なオブジェクト・レベル特権または DATAACCESS 権限を保持していません。

0x00000000000000000200000000000000 DESCRIBE
ステートメントを記述しようとしますが、ユーザーは必要なオブジェクト・レベル特権または DATAACCESS 権限を保持していません。

0x00000000000000000400000000000000 SET_USAGELIST
使用量リスト状態を設定しようとします。

OBJMAINT イベントの監査レコード設計

次の表は、OBJMAINT イベントの監査レコードの形式を示しています。

以下に監査レコードのサンプルを示します。

```
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;
```

表 44. OBJMAINT イベントの監査レコード設計

名前	フォーマット	説明
Timestamp	CHAR(26)	監査イベントの日付と時刻。
カテゴリー	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。 可能な値のリストについては、 380 ページの『監査イベント』の OBJMAINT カテゴリーのセクションを参照してください。
Event Correlator	INTEGER	監査対象の操作の相関 ID。単一イベントにどの監査レコードが関連している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。 成功イベント > = 0 失敗イベント < 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であった場合に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。
Original User ID	VARCHAR(1024)	監査イベントが発生した時刻の CLIENT_ORIGUSERID グローバル変数の値。
Origin Node Number	SMALLINT	監査イベントが発生したメンバー番号。
Coordinator Node Number	SMALLINT	コーディネーター・メンバーのメンバー番号。
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(256)	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 44. *OBJMAINT* イベントの監査レコード設計 (続き)

名前	フォーマット	説明
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)	監査イベントの生成対象となったオブジェクトのスキーマ。
Object Name	VARCHAR(128)	監査イベントの生成対象となったオブジェクトの名前。
Object Type	VARCHAR(32)	監査イベントの生成対象となったオブジェクトのタイプ。有効な値は、『監査レコード・オブジェクト・タイプ』というトピックに示されています。
Package Version	VARCHAR(64)	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のバージョン。
Security Policy Name	VARCHAR(128)	オブジェクト・タイプが TABLE でその表がセキュリティー・ポリシーに関連している場合、そのセキュリティー・ポリシーの名前。
Alter Action	VARCHAR(32)	特定の変更操作 可能な値は以下のとおりです。 • ADD_PROTECTED_COLUMN • ADD_COLUMN_PROTECTION • DROP_COLUMN_PROTECTION • ADD_ROW_PROTECTION • ADD_SECURITY_POLICY • ADD_ELEMENT • ADD COMPONENT • USE GROUP AUTHORIZATIONS • IGNORE GROUP AUTHORIZATIONS • USE ROLE AUTHORIZATIONS • IGNORE ROLE AUTHORIZATIONS • OVERRIDE NOT AUTHORIZED WRITE SECURITY LABEL • RESTRICT NOT AUTHORIZED WRITE SECURITY LABEL
Protected Column Name	VARCHAR(128)	Alter Action が ADD_COLUMN_PROTECTION または DROP_COLUMN_PROTECTION の場合、これは影響される列の名前です。
Column Security Label	VARCHAR(128)	フィールド Column Name で指定された列を保護するセキュリティー・ラベル。
Security Label Column Name	VARCHAR(128)	行を保護するセキュリティー・ラベルを含む列の名前。
Local Transaction ID	VARCHAR(10) FOR BIT DATA	監査イベントが発生した時刻で使用していたローカル・トランザクション ID。これは、トランザクション・ログの一部となる SQLU_TID 構造体です。
Global Transaction ID	VARCHAR(30) FOR BIT DATA	監査イベントが発生した時刻で使用していたグローバル・トランザクション ID。これは、トランザクション・ログの一部となる SQLP_GXID 構造体のデータ・フィールドです。
Client User ID	VARCHAR(255)	監査イベントが発生した時刻の CURRENT CLIENT USERID 特殊レジスターの値。

表 44. *OBJMAINT イベントの監査レコード設計 (続き)*

名前	フォーマット	説明
Client Workstation Name	VARCHAR(255)	監査イベントが発生した時刻の CURRENT CLIENT_WRKSTNNAME 特殊レジスターの値。
Client Application Name	VARCHAR(255)	監査イベントが発生した時刻の CURRENT CLIENT_APPLNAME 特殊レジスターの値。
Client Accounting String	VARCHAR(255)	監査イベントが発生した時刻の CURRENT CLIENT_ACCTNG 特殊レジスターの値。
Trusted Context Name	VARCHAR(128)	トラステッド接続に関連付けられたトラステッド・コンテキストの名前。
Connection Trust Type	INTEGER	可能な値は以下のとおりです。 IMPLICIT_TRUSTED_CONNECTION EXPLICIT_TRUSTED_CONNECTION
Role Inherited	VARCHAR(128)	トラステッド接続を介して継承したロール。
Object Module	VARCHAR(128)	オブジェクトが所属するモジュールの名前。

SECMAINT イベントの監査レコード設計

次の表は、SECMAINT イベントの監査レコードの形式を示しています。

以下に監査レコードのサンプルを示します。

```
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;
```

表 45. *SECMAINT イベントの監査レコード設計*

名前	フォーマット	説明
Timestamp	CHAR(26)	監査イベントの日付と時刻。
カテゴリー	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。 可能な値のリストについては、 380 ページの『監査イベント』の SECMAINT カテゴリーのセクションを参照してください。

表 45. SECMAINT イベントの監査レコード設計 (続き)

名前	フォーマット	説明
Event Correlator	INTEGER	監査対象の操作の相関 ID。単一イベントにどの監査レコードが関連している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。 成功イベント > = 0 失敗イベント < 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であった場合に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。
Original User ID	VARCHAR(1024)	監査イベントが発生した時刻の CLIENT_ORIGUSERID グローバル変数の値。
Origin Node Number	SMALLINT	監査イベントが発生したメンバー番号。
Coordinator Node Number	SMALLINT	コーディネーター・メンバーのメンバー番号。
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)	監査イベントの生成対象となったオブジェクトのスキーマ。 オブジェクト・タイプ・フィールドが ACCESS_RULE なら、このフィールドには規則に関連したセキュリティー・ポリシー名が含まれます。 規則の名前はオブジェクト名のフィールドに格納されます。 オブジェクト・タイプ・フィールドが SECURITY_LABEL なら、このフィールドにはセキュリティー・ラベルがその一部であるセキュリティー・ポリシーの名前が含まれます。 セキュリティー・ラベルの名前はオブジェクト名のフィールドに格納されます。

表 45. SECMAINT イベントの監査レコード設計 (続き)

名前	フォーマット	説明
Object Name	VARCHAR(128)	監査イベントの生成対象となったオブジェクトの名前。 監査イベントが以下のいずれかの場合の役割名を表します。 • ADD_DEFAULT_ROLE • DROP_DEFAULT_ROLE • ALTER_DEFAULT_ROLE • ADD_USER • DROP_USER • ALTER_USER_ADD_ROLE • ALTER_USER_DROP_ROLE • ALTER_USER_AUTHENTICATION オブジェクト・タイプ・フィールドが ACCESS_RULE なら、このフィールドには規則の名前が含まれます。 規則に関連するセキュリティー・ポリシーの名前はオブジェクト・スキーマのフィールドに格納されます。 オブジェクト・タイプ・フィールドが SECURITY_LABEL なら、このフィールドにはセキュリティー・ラベルの名前が含まれます。 セキュリティー・ポリシーの一部である名前はオブジェクト・スキーマのフィールドに格納されます。
Object Type	VARCHAR(32)	監査イベントの生成対象となったオブジェクトのタイプ。有効な値は、『監査レコード・オブジェクト・タイプ』というトピックに示されています。 監査イベントが以下のいずれかの場合、値は ROLE です。 • ADD_DEFAULT_ROLE • DROP_DEFAULT_ROLE • ALTER_DEFAULT_ROLE • ADD_USER • DROP_USER • ALTER_USER_ADD_ROLE • ALTER_USER_DROP_ROLE • ALTER_USER_AUTHENTICATION
Grantor	VARCHAR(128)	特権や権限の付与または取り消しを行う ID。

表 45. SECMAINT イベントの監査レコード設計 (続き)

名前	フォーマット	説明
Grantee	VARCHAR(128)	特権または権限が付与または取り消された被認可者の ID。 監査イベントが以下のいずれかの場合のトラステッド・コンテキスト・オブジェクトを表します。 • ADD_DEFAULT_ROLE • DROP_DEFAULT_ROLE • ALTER_DEFAULT_ROLE • ADD_USER、DROP_USER • ALTER_USER_ADD_ROLE • ALTER_USER_DROP_ROLE • ALTER_USER_AUTHENTICATION
Grantee Type	VARCHAR(32)	付与または取り消された被認可者のタイプ。可能な値は USER、GROUP、ROLE、AMBIGUOUS、または監査イベントが以下のいずれかの場合には TRUSTED_CONTEXT です。 • ADD_DEFAULT_ROLE • DROP_DEFAULT_ROLE • ALTER_DEFAULT_ROLE • ADD_USER • DROP_USER • ALTER_USER_ADD_ROLE • ALTER_USER_DROP_ROLE • ALTER_USER_AUTHENTICATION
Privilege または Authority	CHAR(34)	付与または取り消された特権または権限のタイプを示します。有効な値は、『有効な SECMAINT 特権または権限のリスト』というトピックに示されています。 監査イベントが以下のいずれかの場合、値は ROLE MEMBERSHIP です。 • ADD_DEFAULT_ROLE、DROP_DEFAULT_ROLE • ALTER_DEFAULT_ROLE • ADD_USER • DROP_USER • ALTER_USER_ADD_ROLE • ALTER_USER_DROP_ROLE • ALTER_USER_AUTHENTICATION
Package Version	VARCHAR(64)	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のバージョン。

表 45. SECMAINT イベントの監査レコード設計 (続き)

名前	フォーマット	説明
Access Type	VARCHAR(32)	セキュリティー・ラベルが付与されるアクセス・タイプ。 可能な値: • READ • WRITE • ALL セキュリティー・ポリシーが変更されるアクセス・タイプ。可能な値: • USE GROUP AUTHORIZATIONS • IGNORE GROUP AUTHORIZATIONS • USE ROLE AUTHORIZATIONS • IGNORE ROLE AUTHORIZATIONS • OVERRIDE NOT AUTHORIZED WRITE SECURITY LABEL • RESTRICT NOT AUTHORIZED WRITE SECURITY LABEL
Assumable Authid	VARCHAR(128)	付与される特権が SETSESSIONUSER 特権のとき、これは被認可者がセッション・ユーザーとして設定されることが可能な許可 ID です。
Local Transaction ID	VARCHAR(10) FOR BIT DATA	監査イベントが発生した時刻で使用していたローカル・トランザクション ID。これは、トランザクション・ログの一部となる SQLU_TID 構造体です。
Global Transaction ID	VARCHAR(30) FOR BIT DATA	監査イベントが発生した時刻で使用していたグローバル・トランザクション ID。これは、トランザクション・ログの一部となる SQLP_GXID 構造体のデータ・フィールドです。
Grantor Type	VARCHAR(32)	付与者のタイプ。可能な値は、USER です。
Client User ID	VARCHAR(255)	監査イベントが発生した時刻の CURRENT CLIENT USERID 特殊レジスターの値。
Client Workstation Name	VARCHAR(255)	監査イベントが発生した時刻の CURRENT CLIENT_WRKSTNNAME 特殊レジスターの値。
Client Application Name	VARCHAR(255)	監査イベントが発生した時刻の CURRENT CLIENT_APPLNAME 特殊レジスターの値。
Client Accounting String	VARCHAR(255)	監査イベントが発生した時刻の CURRENT CLIENT_ACCTNG 特殊レジスターの値。
Trusted Context User	VARCHAR(128)	監査イベントが ADD_USER または DROP_USER であるときに、トラステッド・コンテキスト・ユーザーを識別する。
Trusted Context User Authentication	INTEGER	監査イベントが ADD_USER、DROP_USER、または ALTER_USER_AUTHENTICATION であるときに、トラステッド・コンテキスト・ユーザーの認証設定を示す。 1 : 認証が必要 0 : 認証は不要
Trusted Context Name	VARCHAR(128)	トラステッド接続に関連付けられたトラステッド・コンテキストの名前。

表 45. SECMAINT イベントの監査レコード設計 (続き)

名前	フォーマット	説明
Connection Trust Type	INTEGER	可能な値は以下のとおりです。 IMPLICIT_TRUSTED_CONNECTION EXPLICIT_TRUSTED_CONNECTION
Role Inherited	VARCHAR(128)	トラステッド接続を介して継承したロール。

SECMAINT 特権または権限

次のリストは、有効な SECMAINT 特権または権限を示しています。

0x00000000000000000000000000000001 Control Table
表またはビューに関して付与または取り消されたコントロール特権。

0x00000000000000000000000000000002 ALTER
表またはシーケンスを変更するために付与または取り消された特権。

0x00000000000000000000000000000004 ALTER with GRANT
特権の付与が許可されている表またはシーケンスを変更するために付与または取り消された特権。

0x00000000000000000000000000000008 DELETE TABLE
表またはビューをドロップするために付与または取り消された特権。

0x00000000000000000000000000000010 DELETE TABLE with GRANT
特権の付与が許可されている表をドロップするために付与または取り消された特権。

0x00000000000000000000000000000020 Table Index
索引に関して付与または取り消された特権。

0x00000000000000000000000000000040 Table Index with GRANT
特権の付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。

0x00000000000000000000000000000080 Table INSERT
表またはビューへの挿入に関して付与または取り消された特権。

0x00000000000000000000000000000100 Table INSERT with GRANT
特権の付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。

0x00000000000000000000000000000200 Table SELECT
表での選択に関して付与または取り消された特権。

0x00000000000000000000000000000400 Table SELECT with GRANT
特権の付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。

0x00000000000000000000000000000800 Table UPDATE
表またはビューの更新に関して付与または取り消された特権。

0x00000000000000000000000000001000 Table UPDATE with GRANT
特権の付与が許可されている表またはビューの更新に関して付与または取り消された特権。

0x00000000000000000000000000002000 Table REFERENCE
表への参照に関して付与または取り消された特権。

0x00000000000000000000000000004000 Table REFERENCE with GRANT
特権の付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。

0x00000000000000000000000000020000 CREATEIN Schema
スキーマに関して付与または取り消された CREATEIN 特権。

0x00000000000000000000000000040000 CREATEIN Schema with GRANT
特権の付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 CREATEIN 特権。

0x00000000000000000000000000080000 DROPIN Schema
スキーマに関して付与または取り消された DROPIN 特権。

0x00000000000000000000000000100000 DROPIN Schema with GRANT
特権の付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 DROPIN 特権。

0x00000000000000000000000000200000 ALTERIN Schema
スキーマに関して付与または取り消された ALTERIN 特権。

0x00000000000000000000000000400000 ALTERIN Schema with GRANT
特権の付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。

0x00000000000000000000000000800000 DBADM Authority
付与または取り消された DBADM 権限。

0x00000000000000000000000001000000 CREATETAB Authority
付与または取り消された CREATETAB 権限。

0x00000000000000000000000002000000 BINDADD Authority
付与または取り消された BINDADD 権限。

0x00000000000000000000000004000000 CONNECT Authority
付与または取り消された CONNECT 権限。

0x00000000000000000000000008000000 Create not fenced Authority
付与または取り消された fenced でない作成権限。

0x00000000000000000000000010000000 Implicit Schema Authority
付与または取り消された暗黙的スキーマ権限。

0x00000000000000000000000020000000 Server PASSTHRU
このサーバー (フェデレーテッド・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。

0x00000000000000000000000040000000 ESTABLISH TRUSTED CONNECTION
トラステッド接続が作成された。

0x00000000000000000000000100000000 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。

0x00000000000000000000000200000000 Table Space USE with GRANT
特権の付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。

0x00000000000000000000000400000000 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。

0x00000000000000000000000800000000 Column UPDATE with GRANT
特権の付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。

0x00000000000000000000001000000000 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。

0x00000000000000000000002000000000 Column REFERENCE with GRANT
特権の付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。

0x00000000000000000000004000000000 LOAD Authority
付与または取り消された LOAD 権限。

0x00000000000000000000008000000000 Package BIND
パッケージに関して付与または取り消された BIND 特権。

0x00000000000000000000010000000000 Package BIND with GRANT
特権の付与が許可されているパッケージに関して付与または取り消された BIND 特権。

0x00000000000000000000020000000000 EXECUTE
パッケージまたはルーチンに関して付与または取り消された EXECUTE 特権。

0x00000000000000000000040000000000 EXECUTE with GRANT
特権の付与が許可されているパッケージまたはルーチンに関して付与または取り消された EXECUTE 特権。

0x00000000000000000000080000000000 EXECUTE IN SCHEMA
スキーマ内のすべてのルーチンに関して付与または取り消された EXECUTE 特権。

0x00000000000000000000100000000000 EXECUTE IN SCHEMA with GRANT
特権の付与が許可されているスキーマ内のすべてのルーチンに関して付与または取り消された EXECUTE 特権。

0x00000000000000000000200000000000 EXECUTE IN TYPE
タイプ内のすべてのルーチンに関して付与または取り消された EXECUTE 特権。

0x00000000000000000000400000000000 EXECUTE IN TYPE with GRANT
特権の付与が許可されているタイプ内のすべてのルーチンに関して付与または取り消された EXECUTE 特権。

0x00000000000000000000800000000000 CREATE EXTERNAL ROUTINE
付与または取り消された CREATE EXTERNAL ROUTINE 特権。

0x00000000000000000001000000000000 QUIESCE_CONNECT
付与または取り消された QUIESCE_CONNECT 特権。

0x00000000000000000004000000000000 SECADM Authority
付与または取り消された SECADM 権限。

0x00000000000000000008000000000000 USAGE Authority
シーケンスに関して付与または取り消された USAGE 特権。

0x00000000000000000010000000000000 USAGE with GRANT Authority
特権の付与が許可されているシーケンスに関して付与または取り消された USAGE 特権。

0x00000000000000000020000000000000 WITH ADMIN Option
ロールに関して WITH ADMIN オプションが付与または取り消された。

0x00000000000000000040000000000000 SETSESSIONUSER Privilege
付与または取り消された SETSESSIONUSER 特権。

0x00000000000000000080000000000000 Exemption
付与または取り消された免除。

0x00000000000000000100000000000000 Security label
付与または取り消されたセキュリティー・ラベル。

0x00000000000000000200000000000000 WRITE with GRANT
特権の付与が許可されているグローバル変数を書き込むために付与または取り消された特権。

0x00000000000000000400000000000000 Role Membership
付与または取り消されたロールのメンバーシップ。

0x00000000000000000800000000000000 Role Membership with ADMIN Option
付与または取り消された ADMIN オプション付きのロールのメンバーシップ。

0x00000000000000001000000000000000 READ
グローバル変数を読み取るために付与または取り消された特権。

0x00000000000000002000000000000000 READ with GRANT
特権の付与が許可されているグローバル変数を読み取るために付与または取り消された特権。

0x00000000000000004000000000000000 WRITE
グローバル変数を書き込むために付与または取り消された特権。

0x00000000000000010000000000000000 SQLADM
付与または取り消された SQLADM 権限。

0x00000000000000020000000000000000 WLMADM
付与または取り消された WLMADM 権限。

0x00000000000000040000000000000000 EXPLAIN
付与または取り消された EXPLAIN 権限。

0x00000000000000080000000000000000 DATAACCESS
付与または取り消された DATAACCESS 権限。

0x00000000000000100000000000000000 ACCESSCTRL
付与または取り消された ACCESSCTRL 権限。

SYSADMIN イベントの監査レコード設計

次の表は、SYSADMIN イベントの監査レコード設計を示しています。

以下に監査レコードのサンプルを示します。

```
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;
```

表 46. SYSADMIN イベントの監査レコード設計

名前	フォーマット	説明
Timestamp	CHAR(26)	監査イベントの日付と時刻。
カテゴリー	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。 可能な値のリストについては、 380 ページの『監査イベント』の SYSADMIN カテゴリーのセクションを参照してください。
Event Correlator	INTEGER	監査対象の操作の相関 ID。単一イベントにどの監査レコードが関連している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。 成功イベント > = 0 失敗イベント < 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であった場合に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。
Original User ID	VARCHAR(1024)	監査イベントが発生した時刻の CLIENT_ORIGUSERID グローバル変数の値。
Origin Node Number	SMALLINT	監査イベントが発生したメンバー番号。
Coordinator Node Number	SMALLINT	コーディネーター・メンバーのメンバー番号。
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。
Package Version	VARCHAR(64)	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のバージョン。

表 46. SYSADMIN イベントの監査レコード設計 (続き)

名前	フォーマット	説明
Local Transaction ID	VARCHAR(10) FOR BIT DATA	監査イベントが発生した時刻で使用していたローカル・トランザクション ID。これは、トランザクション・ログの一部となる SQLU_TID 構造体です。
Global Transaction ID	VARCHAR(30) FOR BIT DATA	監査イベントが発生した時刻で使用していたグローバル・トランザクション ID。これは、トランザクション・ログの一部となる SQLP_GXID 構造体のデータ・フィールドです。
Client User ID	VARCHAR(255)	監査イベントが発生した時刻の CURRENT CLIENT USERID 特殊レジスターの値。
Client Workstation Name	VARCHAR(255)	監査イベントが発生した時刻の CURRENT CLIENT_WRKSTNNAME 特殊レジスターの値。
Client Application Name	VARCHAR(255)	監査イベントが発生した時刻の CURRENT CLIENT_APPLNAME 特殊レジスターの値。
Client Accounting String	VARCHAR(255)	監査イベントが発生した時刻の CURRENT CLIENT_ACCTNG 特殊レジスターの値。
Trusted Context Name	VARCHAR(128)	トラステッド接続に関連付けられたトラステッド・コンテキストの名前。
Connection Trust Type	INTEGER	可能な値は以下のとおりです。 IMPLICIT_TRUSTED_CONNECTION EXPLICIT_TRUSTED_CONNECTION
Role Inherited	VARCHAR(128)	トラステッド接続を介して継承したロール。

VALIDATE イベントの監査レコード設計

次の表は、VALIDATE イベントの監査レコードの形式を示しています。

以下に監査レコードのサンプルを示します。

```
timestamp=2007-05-07-10.30.51.585626;
category=VALIDATE;
audit event=AUTHENTICATION;
event correlator=1;
event status=0;
userid=newton;
authid=NEWTON;
execution id=gstager;
application id=*LOCAL.gstager.070507143051;
application name=db2bp;
auth type=SERVER;
plugin name=IBMOSauthserver;
```

表 47. VALIDATE イベントの監査レコード設計

名前	フォーマット	説明
Timestamp	CHAR(26)	監査イベントの日付と時刻。
カテゴリー	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

表 47. *VALIDATE* イベントの監査レコード設計 (続き)

名前	フォーマット	説明
Audit Event	VARCHAR(32)	特定の監査イベント。 可能な値は以下のとおりです。GET_GROUPS、GET_USERID、AUTHENTICATE_PASSWORD、VALIDATE_USER、AUTHENTICATION および GET_USERMAPPING_FROM_PLUGIN。
Event Correlator	INTEGER	監査対象の操作の相関 ID。単一イベントにどの監査レコードが関連している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。 成功イベント > = 0 失敗イベント < 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であった場合に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。
Original User ID	VARCHAR(1024)	監査イベントが発生した時刻の CLIENT_ORIGUSERID グローバル変数の値。
Origin Node Number	SMALLINT	監査イベントが発生したメンバー番号。
Coordinator Node Number	SMALLINT	コーディネーター・メンバーのメンバー番号。
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。
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。
Package Version	VARCHAR(64)	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のバージョン。
Plug-in Name	VARCHAR(32)	監査イベントが発生した時点で使用されていたプラグインの名前。
Local Transaction ID	VARCHAR(10) FOR BIT DATA	監査イベントが発生した時刻で使用していたローカル・トランザクション ID。これは、トランザクション・ログの一部となる SQLU_TID 構造体です。
Global Transaction ID	VARCHAR(30) FOR BIT DATA	監査イベントが発生した時刻で使用していたグローバル・トランザクション ID。これは、トランザクション・ログの一部となる SQLP_GXID 構造体のデータ・フィールドです。
Client User ID	VARCHAR(255)	監査イベントが発生した時刻の CURRENT CLIENT USERID 特殊レジスターの値。
Client Workstation Name	VARCHAR(255)	監査イベントが発生した時刻の CURRENT CLIENT_WRKSTNNAME 特殊レジスターの値。
Client Application Name	VARCHAR(255)	監査イベントが発生した時刻の CURRENT CLIENT_APPLNAME 特殊レジスターの値。

表 47. VALIDATE イベントの監査レコード設計 (続き)

名前	フォーマット	説明
Client Accounting String	VARCHAR(255)	監査イベントが発生した時刻の CURRENT CLIENT_ACCTNG 特殊レジスターの値。
Trusted Context Name	VARCHAR(128)	トラステッド接続に関連付けられたトラステッド・コンテキストの名前。
Connection Trust Type	INTEGER	可能な値は以下のとおりです。 IMPLICIT_TRUSTED_CONNECTION EXPLICIT_TRUSTED_CONNECTION
Role Inherited	VARCHAR(128)	トラステッド・コンテキストを介して継承したロールの名前。

CONTEXT イベントの監査レコード設計

次の表は、CONTEXT イベントの監査レコード設計を示しています。

以下に監査レコードのサンプルを示します。

```
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);
```

表 48. CONTEXT イベントの監査レコード設計

名前	フォーマット	説明
Timestamp	CHAR(26)	監査イベントの日付と時刻。
カテゴリー	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。 可能な値のリストについては、 380 ページの『監査イベント』の CONTEXT カテゴリーのセクションを参照してください。
Event Correlator	INTEGER	監査対象の操作の相関 ID。単一イベントにどの監査レコードが関連している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であった場合に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。 監査イベントが SWITCH_USER である場合、 このフィールドには、切り替え後のユーザー ID が表示されます。

表 48. CONTEXT イベントの監査レコード設計 (続き)

名前	フォーマット	説明
Authorization ID	VARCHAR(128)	監査イベントの時刻での許可 ID。 監査イベントが SWITCH_USER である場合、 このフィールドには、切り替え後の許可 ID が表示されます。
Original User ID	VARCHAR(1024)	監査イベントが発生した時刻の CLIENT_ORIGUSERID グローバル変数の値。
Origin Node Number	SMALLINT	監査イベントが発生したメンバー番号。
Coordinator Node Number	SMALLINT	コーディネーター・メンバーのメンバー番号。
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。
Statement Text	CLOB(8M)	適用できる場合には、SQL または XQuery ステートメントのテキストです。 SQL または XQuery ステートメントのテキストが使用可能でない場合、NULL となります。
Package Version	VARCHAR(64)	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のバージョン。
Local Transaction ID	VARCHAR(10) FOR BIT DATA	監査イベントが発生した時刻で使用していたローカル・トランザクション ID。これは、トランザクション・ログの一部となる SQLU_TID 構造体です。
Global Transaction ID	VARCHAR(30) FOR BIT DATA	監査イベントが発生した時刻で使用していたグローバル・トランザクション ID。これは、トランザクション・ログの一部となる SQLP_GXID 構造体のデータ・フィールドです。
Client User ID	VARCHAR(255)	監査イベントが発生した時刻の CURRENT CLIENT USERID 特殊レジスターの値。
Client Workstation Name	VARCHAR(255)	監査イベントが発生した時刻の CURRENT CLIENT_WRKSTNNAME 特殊レジスターの値。
Client Application Name	VARCHAR(255)	監査イベントが発生した時刻の CURRENT CLIENT_APPLNAME 特殊レジスターの値。
Client Accounting String	VARCHAR(255)	監査イベントが発生した時刻の CURRENT CLIENT_ACCTNG 特殊レジスターの値。
Trusted Context Name	VARCHAR(128)	トラステッド接続に関連付けられたトラステッド・ コンテキストの名前。
Connection Trust Type	INTEGER	可能な値は以下のとおりです。 IMPLICIT_TRUSTED_CONNECTION EXPLICIT_TRUSTED_CONNECTION
Role Inherited	VARCHAR(128)	トラステッド接続を介して継承したロール。

EXECUTE イベントの監査レコード設計

次の表は、EXECUTE 区分の一部として監査されるすべてのフィールドを説明しています。

以下に監査レコードのサンプルを示します。

注: 他の監査区分とは異なり、EXECUTE 区分の場合は、監査ログが表形式で表示される際に 1 つのイベントの記述が複数行にわたって示されることがあります。1 行目のレコードは主要なイベントについて記述しており、イベント列にはキーワード STATEMENT が含まれています。残りの行は、パラメーター・マーカーやホスト変数について、パラメーターごとに 1 つの行を使用して記述します。これらの行のイベント列には、キーワード DATA が含まれています。監査ログがレポート形式で表示される際は、レコードは 1 つになりますが、「Statement Value」には複数の項目が表示されます。DATA キーワードは、表形式の場合にのみ表示されます。

```
timestamp=2006-04-10-13.20.51.029203;
  category=EXECUTE;
  audit event=STATEMENT;
  event correlator=1;
  event status=0;
  database=SAMPLE;
  userid=smith;
  authid=SMITH;
  session authid=SMITH;
  application id=*LOCAL.prodrig.060410172044;
  application name=myapp;
  package schema=NULLID;
  package name=SQLC2F0A;
  package section=201;
  uow id=2;
  activity id=3;
  statement invocation id=0;
  statement nesting level=0;
  statement text=SELECT * FROM DEPARTMENT WHERE DEPTNO = ? AND DEPTNAME = ?;
  statement isolation level=CS;
  compilation environment=
    isolation level=CS
    query optimization=5
    min_dec_div_3=NO
    degree=1
    sqlrules=DB2
    refresh age=+00000000000000.000000
    schema=SMITH
    maintained table type=SYSTEM
    resolution timestamp=2006-04-10-13.20.51.000000
    federated asynchrony=0;
  value index=0;
  value type=CHAR;
  value data=C01;
  value index=1;
  value type=VARCHAR;
  value extended indicator=-1;
  value index=INFORMATION CENTER;
  local_start_time=2006-04-10-13.20.51.021507
```

表 49. EXECUTE イベントの監査レコード設計

NAME	フォーマット	説明
Timestamp	CHAR(26)	監査イベントの日付と時刻。

表 49. EXECUTE イベントの監査レコード設計 (続き)

NAME	フォーマット	説明
カテゴリー	CHAR(8)	監査イベントの区分。可能な値は EXECUTE です。
Audit Event	VARCHAR(32)	特定の監査イベント。 可能な値のリストについては、 380 ページの『監査イベント』の EXECUTE カテゴリーのセクションを参照してください。
Event Correlator	INTEGER	監査対象の操作の相関 ID。単一イベントにどの監査レコードが関連している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。成功イベント > = 0 失敗イベント < 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であった場合に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ステートメント許可 ID。
Session Authorization ID	VARCHAR(128)	監査イベントの時刻でのセッション許可 ID。
Origin Node Number	SMALLINT	監査イベントが発生したメンバー番号。
Coordinator Node Number	SMALLINT	コーディネーター・メンバーのメンバー番号。
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。
Client User ID	VARCHAR(255)	監査イベントが発生した時刻の CURRENT CLIENT USERID 特殊レジスターの値。
Client Accounting String	VARCHAR(255)	監査イベントが発生した時刻の CURRENT CLIENT_ACCTNG 特殊レジスターの値。

表 49. EXECUTE イベントの監査レコード設計 (続き)

NAME	フォーマット	説明
Client Workstation Name	VARCHAR(255)	監査イベントが発生した時刻の CURRENT CLIENT_WRKSTNNAME 特殊レジスターの値。
Client Application Name	VARCHAR(255)	監査イベントが発生した時刻の CURRENT CLIENT_APPLNAME 特殊レジスターの値。
Trusted Context Name	VARCHAR(128)	トラステッド接続に関連付けられたトラステッド・コンテキストの名前。
Connection Trust type	INTEGER	可能な値は以下のとおりです。 IMPLICIT_TRUSTED_CONNECTION および EXPLICIT_TRUSTED_CONNECTION
Role Inherited	VARCHAR(128)	トラステッド接続を介して継承したロール。
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	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。
Package Version	VARCHAR(164)	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のバージョン。
Local Transaction ID	VARCHAR(10) FOR BIT DATA	監査イベントが発生した時刻で使用していたローカル・トランザクション ID。これは、トランザクション・ログの一部となる SQLU_TID 構造体です。
Global Transaction ID	VARCHAR(30) FOR BIT DATA	監査イベントが発生した時刻で使用していたグローバル・トランザクション ID。これは、トランザクション・ログの一部となる SQLP_GXID 構造体のデータ・フィールドです。

表 49. *EXECUTE* イベントの監査レコード設計 (続き)

NAME	フォーマット	説明
UOW ID	BIGINT	アクティビティーが発生した作業単位の ID。この値は、作業単位ごとにアプリケーション ID 内で固有です。
Activity ID	BIGINT	作業単位内で固有のアクティビティー ID。
Statement Invocation ID	BIGINT	作業単位内の同じネスティング・レベルにあるルーチン呼び出しを区別する ID。これは、作業単位内の特定のネスティング・レベルにおいて固有です。
Statement Nesting Level	BIGINT	ステートメントが実行されていたときに有効であったネスティングまたは再帰のレベル。ネスティングの各レベルは、ストアード・プロシージャーやユーザー定義関数 (UDF) のネストされた、または再帰可能な呼び出しに対応しています。
Activity Type	VARCHAR(32)	アクティビティーのタイプ。 可能な値は以下のとおりです。 • READ_DML • WRITE_DML • DDL • CALL • NONE
Statement Text	CLOB(8M)	適用できる場合には、SQL または XQuery ステートメントのテキストです。
Statement Isolation Level	CHAR(8)	ステートメントが実行されていたときに有効であった分離の値。 可能な値は以下のとおりです。 • NONE (分離の指定なし) • UR (非コミット読み取り) • CS (カーソル固定) • RS (読み取り固定) • RR (反復可能読み取り)

表 49. *EXECUTE* イベントの監査レコード設計 (続き)

NAME	フォーマット	説明
Compilation Environment Description	BLOB(8K)	SQL ステートメントのコンパイル時に使用されたコンパイル環境。このエレメントは、COMPILATION_ENV 表関数または SET COMPILATION ENVIRONMENT SQL ステートメントに入力として渡すことができます。
Rows Modified	INTEGER	以下の両方の結果として削除、挿入、または更新された行の総数。 • 削除操作成功後の制約の強制 • アクティブにされたインライン・トリガーからのトリガー SQL ステートメントの処理 コンパウンド SQL が呼び出される場合は、すべてのサブステートメントの、これに該当する行の数の集計が含まれます。場合によっては、エラーが発生したときに、内部エラーを示す負の値がこのフィールドに表示されることがあります。この値は、SQLCA の sqlerrd(5) フィールドと等価です。
Rows Returned	BIGINT	ステートメントによって戻される行の総数。
Savepoint ID	BIGINT	ステートメントが実行されていたときにそのステートメントで有効であったセーブポイント ID。「Audit Event」が SAVEPOINT、RELEASE_SAVEPOINT、または ROLLBACK_SAVEPOINT である場合、「Savepoint ID」は、設定、解放、またはロールバックされるセーブポイントになります。

表 49. EXECUTE イベントの監査レコード設計 (続き)

NAME	フォーマット	説明
Statement Value Index	INTEGER	SQL ステートメントで使用される入力パラメーター・マーカーまたはホスト変数の位置。
Statement Value Type	CHAR(16)	SQL ステートメントに関連付けられているデータ値のタイプのストリング表現。可能な値の例としては、INTEGER や CHAR が挙げられます。
Statement Value Data	CLOB(128K)	SQL ステートメントへのデータ値のストリング表現。LOB、LONG、XML、および構造化タイプのパラメーターは表示されません。日付、時刻、およびタイム・スタンプのフィールドは、ISO 形式で記録されます。
Statement Value Extended Indicator	INTEGER	このステートメント値に関して指定される拡張標識の値。可能な値は次のとおりです。 • 0 (標識値による割り当てと同じ値にステートメント値が指定された場合) • -1 (標識値によって NULL が指定された場合) • -5 (標識値によって DEFAULT が指定された場合) • -7 (標識値によって UNASSIGNED が指定された場合)
Local Start Time	CHAR(26)	このアクティビティーがパーティションを処理し始めた時間。アクティビティーがパッケージを必要としない場合 (例えば CONNECT、CONNECT RESET、COMMIT、ROLLBACK などの場合) には、このフィールドを空ストリングにすることができます。値は現地時間でログに記録されます。

監査イベント

監査カテゴリーごとに、特定のタイプのイベントが監査レコードを作成できます。

AUDIT カテゴリーのイベント

- ALTER_AUDIT_POLICY
- ARCHIVE
- AUDIT_REMOVE
- AUDIT_REPLACE
- AUDIT_USING
- CONFIGURE
- CREATE_AUDIT_POLICY
- DB2AUD
- DROP_AUDIT_POLICY
- EXTRACT
- FLUSH
- LIST_LOGS
- PRUNE (バージョン 9.5 以降では生成されません)
- START
- STOP
- UPDATE_DBM_CFG

CHECKING カテゴリーのイベント

- CHECKING_FUNCTION
- CHECKING_MEMBERSHIP_IN_ROLES
- CHECKING_OBJECT
- CHECKING_TRANSFER

CONTEXT カテゴリーのイベント

- ADD_NODE
- ATTACH
- BACKUP_DB
- BIND
- CLOSE_CONTAINER_QUERY
- CLOSE_CURSOR
- CLOSE_HISTORY_FILE
- CLOSE_TABLESPACE_QUERY
- COMMIT
- CONNECT
- CONNECT_RESET
- CREATE_DATABASE
- DARI_START

- DARI_STOP
- DBM_CFG_OPERATION
- DESCRIBE
- DESCRIBE_DATABASE
- DETACH
- DISCOVER
- DROP_DATABASE
- ENABLE_MULTIPAGE
- ESTIMATE_SNAPSHOT_SIZE
- EXECUTE
- EXECUTE_IMMEDIATE
- EXTERNAL_CANCEL
- FETCH_CONTAINER_QUERY
- FETCH_CURSOR
- FETCH_HISTORY_FILE
- FETCH_TABLESPACE
- FORCE_APPLICATION
- GET_DB_CFG
- GET_DFLT_CFG
- GET_SNAPSHOT
- GET_TABLESPACE_STATISTIC
- IMPLICIT_REBIND
- LOAD_MSG_FILE
- LOAD_TABLE
- OPEN_CONTAINER_QUERY
- OPEN_CURSOR
- OPEN_HISTORY_FILE
- OPEN_TABLESPACE_QUERY
- PREPARE
- PRUNE_RECOVERY_HISTORY
- QUIESCE_TABLESPACE
- READ_ASYNC_LOG_RECORD
- REBIND
- REDISTRIBUTE
- REORG
- REQUEST_ROLLBACK
- RESET_DB_CFG
- RESET_MONITOR
- RESTORE_DB
- ROLLBACK

- ROLLFORWARD_DB
- RUNSTATS
- SET_APPL_PRIORITY
- SET_MONITOR
- SET_RUNTIME_DEGREE
- SET_TABLESPACE_CONTAINERS
- SINGLE_TABLESPACE_QUERY
- SWITCH_USER
- UNLOAD_TABLE
- UNQUIESCE_TABLESPACE
- UPDATE_AUDIT
- UPDATE_DBM_CFG
- UPDATE_RECOVERY_HISTORY

EXECUTE カテゴリーのイベント

- COMMIT: COMMIT ステートメントの実行。
- CONNECT: データベース接続の確立。
- CONNECT RESET: データベース接続の終了。
- DATA: ステートメント用のホスト変数またはパラメーター・マーカーのデータ値。

  このイベントは、ステートメントに含まれる各ホスト変数やパラメーター・マーカーごとに繰り返されます。DATA は、区切りファイルから監査ログを抽出するときにのみ表示されます。
- GLOBAL COMMIT: グローバル・トランザクション内での COMMIT の実行。
- GLOBAL ROLLBACK: グローバル・トランザクション内での ROLLBACK の実行。
- RELEASE SAVEPOINT: RELEASE SAVEPOINT ステートメントの実行。
- ROLLBACK: ROLLBACK ステートメントの実行。
- SAVEPOINT: SAVEPOINT ステートメントの実行。
- STATEMENT: SQL ステートメントの実行。
- SWITCH USER: トラステッド接続内でのユーザーの切り替え。

OBJMAINT カテゴリーのイベント

- ALTER_OBJECT (保護されている表を変更する場合、およびモジュールを変更する場合に生成されます)
- CREATE_OBJECT
- DROP_OBJECT
- RENAME_OBJECT

SECMAINT カテゴリーのイベント

- ADD_DEFAULT_ROLE
- ADD_USER

- ALTER_DEFAULT_ROLE
- ALTER SECURITY POLICY
- ALTER_USER_ADD_ROLE
- ALTER_USER_AUTHENTICATION
- ALTER_USER_DROP_ROLE
- DROP_DEFAULT_ROLE
- DROP_USER
- GRANT
- IMPLICIT_GRANT
- IMPLICIT_REVOKE
- REVOKE
- SET_SESSION_USER
- TRANSFER_OWNERSHIP
- UPDATE_DBM_CFG

SYSADMIN カテゴリーのイベント

- ACTIVATE_DB
- ADD_NODE
- ALTER_BUFFERPOOL
- ALTER_DATABASE
- ALTER_NODEGROUP
- ALTER_TABLESPACE
- ATTACH_DEBUGGER
- BACKUP_DB
- CATALOG_DB
- CATALOG_DCS_DB
- CATALOG_NODE
- CHANGE_DB_COMMENT
- CLOSE_CONTAINER_QUERY
- CLOSE_TABLESPACE_QUERY
- CREATE_BUFFERPOOL
- CREATE_DATABASE
- CREATE_DB_AT_NODE
- CREATE_EVENT_MONITOR
- CREATE_INSTANCE
- CREATE_NODEGROUP
- CREATE_TABLESPACE
- DB2AUD
- DB2AUDIT
- DB2REMOT

- DB2SET
- DB2TRC
- DEACTIVATE_DB
- DELETE_INSTANCE
- DESCRIBE_DATABASE
- DROP_BUFFERPOOL
- DROP_DATABASE
- DROP_EVENT_MONITOR
- DROP_NODEGROUP
- DROP_NODE_VERIFY
- DROP_TABLESPACE
- ENABLE_MULTIPAGE
- ESTIMATE_SNAPSHOT_SIZE
- FETCH_CONTAINER_QUERY
- FETCH_TABLESPACE
- FORCE_APPLICATION
- GET_SNAPSHOT
- GET_TABLESPACE_STATISTIC
- GRANT_DBADM (V97 では生成されません)
- GRANT_DB_AUTH (V97 では生成されません)
- KILLDBM
- LIST_DRDA_INDOUBT_TRANSACTIONS
- LOAD_TABLE
- MERGE_DBM_CONFIG_FILE
- MIGRATE_DB
- MIGRATE_DB_DIR
- MIGRATE_SYSTEM_DIRECTORY
- OPEN_CONTAINER_QUERY
- OPEN_TABLESPACE_QUERY
- PRUNE_RECOVERY_HISTORY
- QUIESCE_TABLESPACE
- READ_ASYNC_LOG_RECORD
- REDISTRIBUTE_NODEGROUP
- RENAME_TABLESPACE
- RESET_ADMIN_CFG
- RESET_DBM_CFG
- RESET_DB_CFG
- RESET_MONITOR
- RESTORE_DB
- REVOKE_DBADM (V97 では生成されません)

- REVOKE_DB_AUTH (V97 では生成されません)
- ROLLFORWARD_DB
- SET_APPL_PRIORITY
- SET_EVENT_MONITOR_STATE
- SET_RUNTIME_DEGREE
- SET_TABLESPACE_CONTAINERS
- SINGLE_TABLESPACE_QUERY
- START_DB2
- STOP_DB2
- UNCATALOG_DB
- UNCATALOG_DCS_DB
- UNCATALOG_NODE
- UNLOAD_TABLE
- UPDATE_ADMIN_CFG
- UPDATE_CLI_CONFIGURATION
- UPDATE_DB_VERSION
- UPDATE_DBM_CFG
- UPDATE_DB_CFG
- SET_MONITOR
- UPDATE_RECOVERY_HISTORY

VALIDATE カテゴリーのイベント

- AUTHENTICATE
- CHECK_GROUP_MEMBERSHIP (バージョン 9.5 以降では生成されません)
- GET_USERMAPPING_FROM_PLUGIN
- GET_GROUPS (バージョン 9.5 以降では生成されません)
- GET_USERID (バージョン 9.5 以降では生成されません)

第 12 章 オペレーティング・システム・セキュリティーの操作

オペレーティング・システムは、データベース・インストールのセキュリティーをサポートするのに使用できるセキュリティー・フィーチャーを提供します。

DB2 および Windows セキュリティー

Windows ドメインは、特定の名前および固有の名前で参照されるクライアント・コンピューターおよびサーバー・コンピューターの配置であり、 Security Access Manager (SAM) と呼ばれる単一のユーザー・アカウント・データベースを共有します。ドメイン内のコンピューターのうちの 1 つが、ドメイン・コントローラーです。ドメイン・コントローラーは、ユーザー・ドメインの対話のすべての面を管理します。

ドメイン・コントローラーは、ドメイン・アカウントにログオンするユーザーを認証するために、ドメイン・ユーザー・アカウント・データベース内の情報を使用します。ドメインごとに、1 つのドメイン・コントローラーが 1 次ドメイン・コントローラー (PDC) になります。ドメイン内には、1 次ドメイン・コントローラーが存在しない場合、または 1 次ドメイン・コントローラーが使用不可の場合に、ユーザー・アカウントを認証するバックアップ・ドメイン・コントローラー (BDC) が存在する場合があります。バックアップ・ドメイン・コントローラーは、PDC のマスター・コピーと定期的に同期化される Windows Security Account Manager (SAM) データベースのコピーを保持しています。

ユーザー・アカウント、ユーザー ID、およびパスワードは、 1 次ドメイン・コントローラーに定義するだけで、ドメイン・リソースにアクセスできるようになります。

注: CONNECT ステートメントと ATTACH コマンドは、2 部構成のユーザー ID をサポートしています。 SAM 互換ユーザー ID の修飾子は、最大長 15 文字の 'Domain¥User' スタイルの名前です。

Windows サーバーのインストール時のセットアップ手順の中で、以下を作成するように選択できます。

- 1 次ドメイン・コントローラー (新しいドメイン内)
- バックアップ・ドメイン・コントローラー (既知のドメイン内)
- スタンドアロン・サーバー (既知のドメイン内)

新しいドメイン内で「コントローラー」を選択すると、そのサーバーは 1 次ドメイン・コントローラーになります。

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 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。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。

この動作の方法を説明するために、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。

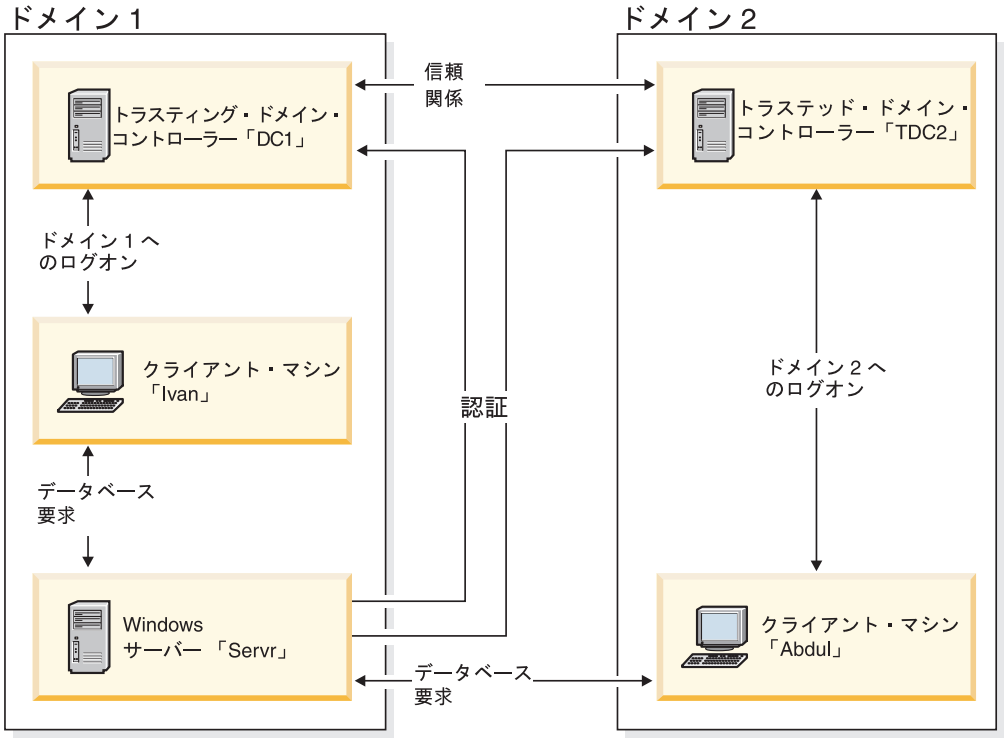


図 7. Windows のドメインを使用した認証

各マシンには、セキュリティー・データベース (セキュリティー・アクセス管理 (SAM)) があります。 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 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。

認証シナリオ

サーバー認証を使用するシナリオ (Windows)

以下の例は、サーバーによるユーザーの認証を示しています。

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。

クライアント認証および Windows クライアント・マシンを使用するシナリオ

以下の例は、クライアント・コンピューターによるユーザーの認証を示しています。

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
db2start
```

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 セキュリティー・サービスを始動してください。セキュリティー・サービスは、Windows のインストールの一部としてインストールされます。次いで DB2 がインストールされ、Windows サービスとして「登録」されますが、デフォルトでは、自動的に開始されません。DB2 セキュリティー・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。

グローバル・グループのサポート (Windows)

DB2 データベース・システムはグローバル・グループをサポートします。

グローバル・グループを使用するために、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。あるユーザーがメンバーとなっているグループを DB2 データベース・マネージャーがすべて列挙するとき、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・グループもまたリストされます (そのグループは、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・グループ内にあるため)。

グローバル・グループは、以下の 2 つの状態で使用されます。

- ローカル・グループに組み込まれた状態。ローカル・グループに許可を付与する必要があります。
- ドメイン・コントローラーに組み込まれた状態。グローバル・グループに許可を付与する必要があります。

Windows での DB2 によるユーザー認証とグループ情報

ユーザー名およびグループ名に関する制約事項 (Windows)

Windows 環境に固有の制約事項がいくつかあります。ただし、DB2 の一般的なオブジェクト命名規則も適用されます。

- Windows 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 データベース内で使用されるとき、通常は大文字に変換されます。例えば、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する場合は、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 データベース・マネージャーは単一のネーム・スペースをサポートします。つまり、トラステッド・ドメイン環境で実行する場合は、同じ名前のユーザー・アカウントを複数のドメインに置いたり、サーバー・マシンのローカル SAM や別のドメインに置いたりすることはできません。
- ユーザー名は、グループ名と同じであってはなりません。
- ローカル・グループは、ドメイン・レベル・グループと同じ名前であってはなりません。

Windows でのグループ認証およびユーザー認証

ユーザーは、「ユーザ マネージャ」という Windows 管理ツールを使用して、Windows 上に定義されます。他のアカウント (メンバーとも呼ばれる) が入っているアカウントは、グループです。

グループを使用すれば、Windows 管理者は、権利および許可をグループ内のユーザーに一度に付与できるようになり、各ユーザーを個別に保守する必要がなくなります。グループは、ユーザー・アカウントと同様、 Security Access Manager (SAM) データベース内で定義され保守されます。

グループには次の 2 つの種類があります。

- ローカル・グループ。ローカル・グループには、ローカル・アカウント・データベース内で作成されたユーザー・アカウントを入れることができます。ローカル・グループがドメインの一部であるマシン上に存在する場合は、ローカル・グループには、 Windows ドメインのドメイン・アカウントおよびグループを入れることもできます。ローカル・グループをワークステーション上に作成する場合は、ローカル・グループはそのワークステーション固有です。

- グローバル・グループ。グローバル・グループは、ドメイン・コントローラー上にのみ存在し、ドメインの SAM データベースのユーザー・アカウントが含まれています。つまり、グローバル・グループには、グローバル・グループが作成されたドメインのユーザー・アカウントだけを入れることができます。他のグループをメンバーとして入れることはできません。グローバル・グループは、グローバル・グループが所属するドメインのサーバーおよびワークステーションと、信用のあるドメインで使用できます。

Windows でのドメイン間の信頼関係

信頼関係は、2 つのドメイン間の管理および通信リンクです。 2 つのドメイン間に信頼関係があれば、ユーザー・アカウントおよびグローバル・グループを、アカウントが定義されたドメインではないドメインで使用できるようになります。

アカウント情報は、トラステッド・ドメイン内に認証されずに存在しているユーザー・アカウントおよびグローバル・グループの権利や許可を妥当性検査するために共有されます。信頼関係は、2 つ以上のドメインを単一の管理単位にまとめることにより、ユーザーの管理を単純化します。

信頼関係には次の 2 つのドメインがあります。

- トラスティング・ドメイン。このドメインは、ユーザーを認証してもらうために別のドメインを信頼しています。
- トラステッド・ドメイン。このドメインは、別のドメインに代わってユーザーを認証します。

信頼関係は他動的なものではありません。つまり、ドメイン間で双方向の信頼関係を明示的に確立する必要があります。例えば、トラスティング・ドメインは、必ずしもトラステッド・ドメインであるとは限りません。

認証でのグループおよびドメイン・セキュリティーの使用 (Windows)

DB2 データベース・システムでは、特権を付与するときまたは権限レベルを定義する時に、ローカル・グループかグローバル・グループを指定できます。

このタスクについて

ユーザーがグループのメンバーであると判断されるのは、そのユーザーのアカウントが、ローカルまたはグローバル・グループ内で明示的に定義されている場合、またはローカル・グループのメンバーになるように定義されているグローバル・グループのメンバーになることによって暗黙的に定義されている場合です。

DB2 データベース・マネージャーは、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 データベース・マネージャーは、ユーザーの情報が含まれているセキュリティー・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 データベース・システムは、ユーザー・アカウントがどこにあるかに関係なく、 DB2 データベース

がインストールされているローカル Windows サーバーでのみグループが列挙されるように設定することができます。それ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 instance_name DB2_GRP_LOOKUP=local
```

このコマンドの発行後、変更を有効にするには、 DB2 データベース・インスタンスを停止して開始する必要があります。次に、ローカル・グループを作成し、ドメイン・アカウントまたはグローバル・グループをそのローカル・グループに入れます。

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_GRP_LOOKUP プロファイル・レジストリー変数が local に設定されている場合、DB2 データベース・マネージャーはローカル・マシン上のユーザーのグループのみを列挙しようとします。そのユーザーがローカル・グループのメンバーとして定義されていないか、またはローカル・グループ内にネストされているグローバル・グループのメンバーとして定義されていない場合、グループの列挙は失敗します。DB2 データベース・マネージャーは、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のグループを列挙しようとは**しません**。

リソース・ドメイン内で 1 次ドメイン・コントローラーまたはバックアップ・ドメイン・コントローラーであるマシン上で DB2 データベース・マネージャーが実行されている場合は、任意のドメイン・コントローラーを任意のトラステッド・ドメインに置くことができます。トラステッド・ドメイン内のバックアップ・ドメイン・コントローラーのドメインの名前は、ドメイン・コントローラーでなければ知ることができないためです。

アクセス・トークンによるユーザーのグループ情報の取得 (Windows)

アクセス・トークンは、プロセスまたはスレッドのセキュリティー・コンテキストを説明するオブジェクトです。アクセス・トークン内の情報には、プロセスまたはスレッドに関連したユーザー・アカウントの識別および特権が含まれます。

ログオンすると、システムはユーザーのパスワードをセキュリティー・データベースに保管されている情報と比較して、それを検証します。パスワードが認証されると、システムはアクセス・トークンを生成します。ユーザーのために実行されるすべてのプロセスは、このアクセス・トークンのコピーを使用します。

アクセス・トークンは、キャッシュされた証明書に基づいて取得することもできます。システムに認証されると、その証明書はオペレーティング・システムによってキャッシュに入れられます。ドメイン・コントローラーにアクセスできないときは、キャッシュ内にある前回のログオン時のアクセス・トークンを参照できます。

アクセス・トークンには、ローカル・グループおよびさまざまなドメイン・グループ (グローバル・グループ、ドメイン・ローカル・グループ、およびユニバーサル・グループ) など、ユーザーが所属するすべてのグループに関する情報が含まれています。

注: アクセス・トークン・サポートは使用可能ですが、リモート接続を使用する場合、クライアント認証を使用するグループ・ルックアップはサポートされていません。

アクセス・トークン・サポートを使用可能にするには、 **db2set** コマンドを使用して **DB2_GRP_LOOKUP** レジストリー変数を更新する必要があります。 **DB2_GRP_LOOKUP** は最大で 2 つのパラメーターを持つことができ、それぞれはコンマで区切ります。

- 最初のパラメーターは従来のグループ・ルックアップ用で、取ることのできる値は " "、"LOCAL"、または "DOMAIN" です。
- 2 番目のパラメーターはトークン・スタイルのグループ・ルックアップ用で、取ることのできる値は "TOKEN"、"TOKENDOMAIN"、または "TOKENLOCAL" です。

2 番目のパラメーター (TOKEN、TOKENDOMAIN、または TOKENLOCAL) が指定されると、従来のグループの列挙より優先されます。トークンのグループ列挙が失敗する場合に最初のパラメーター **DB2_GRP_LOOKUP** が指定されると、従来のグループ・ルックアップが生じます。

値 TOKEN、TOKENDOMAIN、および TOKENLOCAL の意味は、以下のとおりです。

- TOKENLOCAL

  トークンを使用して、ローカル・マシンのグループを列挙します (これは、従来の "LOCAL" グループ・ルックアップに相当します)。

- TOKENDOMAIN

  トークンを使用して、ユーザーが定義されているロケーション (ローカル・ユーザーの場合はローカル・マシン、およびドメイン・ユーザーの場合はドメイン) のグループを列挙します。これは、従来の " "、または "DOMAIN" グループ・ルックアップに相当します。

- TOKEN

  トークンを使用して、ドメインおよびローカル・マシンのグループを列挙します。ローカル・ユーザーの場合、戻されるグループにはローカル・グループが含まれます。ドメイン・ユーザーの場合、戻されるグループにはドメインとローカル・グループの両方が含まれます。従来のグループ・ルックアップに相当するものはありません。

例えば、**DB2_GRP_LOOKUP** の以下の設定により、ローカル・グループを列挙するためのアクセス・トークン・サポートが使用可能になります。

```
db2set DB2_GRP_LOOKUP=LOCAL,TOKENLOCAL
```

次の例により、ユーザー ID が定義されたロケーションだけでなく、ローカル・マシンでもグループを列挙するためのアクセス・トークン・サポートが使用可能になります (アカウントがドメインで定義されている場合)。

```
db2set DB2_GRP_LOOKUP=,TOKEN
```

以下に挙げる最後の例により、ユーザー ID が定義されたロケーションで、ドメイン・グループを列挙するためのアクセス・トークン・サポートが使用可能になります。

```
db2set DB2_GRP_LOOKUP=DOMAIN,TOKENDOMAIN
```

注: アクセス・トークン・サポートは、CLIENT 認証を除くすべての認証タイプによって使用可能になります。

DB2_GRP_LOOKUP 環境変数と DB2 グループ列挙 (Windows)

Windows では、ドメイン・レベルで定義されたグループ、ローカル・マシンで定義されたグループ、またはその両方にユーザーは属することができます。

DB2_GRP_LOOKUP 環境変数は、ローカル・マシンでグループを列挙するかどうか、それともユーザーが定義されている場所 (ローカル・ユーザーの場合にはローカル・マシン、ドメイン・ユーザーの場合にはドメイン・レベル) でそうするかを制御します。そのため、セキュリティー管理者が権限と特権を付与する場合、**DB2_GRP_LOOKUP** が意図どおりに設定され、正しいユーザーが適正な許可を受け取るように注意する必要があります。

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には、以下のようになります。

1. DB2 データベース・システムは最初に同じマシン上で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 データベース・システムは同じドメインの中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、トラステッド・ドメインから探そうとします。

例えば、**DB2_GRP_LOOKUP** が設定されていない以下の状態について考えてみてください。

1. ドメイン・ユーザー DUSER1 は、ローカル・グループ GROUP1 のメンバーです。
2. セキュリティー管理者 (SECADM 権限を保持) は DBADM 権限をグループ GROUP1 に付与します。

   ```
   GRANT DBADM ON database TO GROUP GROUP1
   ```

3. **DB2_GRP_LOOKUP** が設定されないと、グループはユーザーが定義された場所で列挙されます。そのため、DUSER1 のグループはドメイン・レベルで列挙されます。DUSER1 はドメイン・レベルでグループ GROUP1 に属していないので、DUSER1 は DBADM 権限を受け取りません。

さらに、**DB2_GRP_LOOKUP** が設定されていない状況における、**UPGRADE DATABASE** コマンドが関係する、以下のより一層複雑なシナリオについて考えてみてください。

1. ドメイン・ユーザー DUSER2 は、ローカル Administrators グループのメンバーです。
2. **sysadm_group** 構成パラメーターが設定されていないので、ローカル Administrators グループのメンバーは自動的に SYSADM 権限を保持します。

3. ユーザー DUSER2 は **UPGRADE DATABASE** コマンドを実行できます (DUSER2 が SYSADM 権限を保持しているため)。**UPGRADE DATABASE** コマンドは、アップグレード対象のデータベースに対する DBADM 権限を、SYSADM グループ (この場合は Administrators グループ) に付与します。
4. **DB2_GRP_LOOKUP** が設定されないと、グループはユーザーが定義された場所で列挙されます。そのため、DUSER2 のグループはドメイン・レベルで列挙されます。DUSER2 はドメイン・レベルで Administrators グループに属していないので、DUSER2 は DBADM 権限を受け取りません。

このシナリオに関する可能な解決方法は、以下のいずれかの変更を行うことです。

- **DB2_GRP_LOOKUP** を `local` に設定します。
- DBADM 権限を持つ必要のあるユーザーを、ドメイン・コントローラーで Administrators グループ、または GROUP1 グループに追加します。

SYSPROC.AUTH_LIST_AUTHORITIES_FOR_AUTHID 表関数を使用すると、以下の例の DUSER1 に関して示されているように、ユーザーが保持している権限を検証できます。

```
SELECT AUTHORITY, D_USER, D_GROUP, D_PUBLIC, ROLE_USER, ROLE_GROUP, ROLE_PUBLIC, D_ROLE
   FROM TABLE (SYSPROC.AUTH_LIST_AUTHORITIES_FOR_AUTHID ('DUSER1', 'U') ) AS T
   ORDER BY AUTHORITY
```

SYSPROC.AUTH_LIST_GROUPS_FOR_AUTHID 表関数を使用すると、以下の例で DUSER1 に示されているように、DB2 データベース・マネージャーがユーザーが属していると判別したグループを検証できます。

```
SELECT * FROM TABLE (SYSPROC.AUTH_LIST_GROUPS_FOR_AUTHID ('DUSER1')) AS T
```

注: ドメイン・レベルとローカル・マシン上で同じグループ名を使用すると、DB2 データベース・マネージャーはグループを完全修飾しないので、混乱を招く可能性があります。

順序付けドメイン・リストを使用した認証

1 つのトラステッド・ドメイン・フォレスト内では、ユーザー ID が複数回にわたって定義される場合があります。トラステッド・ドメイン・フォレストとは、ネットワークを介して互いに関連している複数のドメインからなる集合です。

このタスクについて

1 つのドメインのユーザーが、別のドメイン内の別のユーザーと同じユーザー ID を持つ可能性があります。そのような場合、以下の操作が困難になる場合があります。

- 同じユーザー ID を持つ複数のユーザーをそれぞれ別のドメインで認証する。
- グループに基づいて特権を付与または取り消すための、グループ・ルックアップ。
- パスワードの確認。
- ネットワーク・トラフィックの制御。

同じユーザー ID を持つ複数のユーザーがドメイン・フォレスト内でアクセスする場合の問題を防ぐには、 **db2set** およびレジストリー変数 **DB2DOMAINLIST** を使って定義される、順序付けドメイン・リストを使用する必要があります。順序を設定す

るときには、リストに含める複数のドメインをコンマで区切ります。ユーザー認証時に複数のドメインを検索する順序を決定するときには、十分に考慮する必要があります。

ドメイン・リストの下の方にあるドメインに含まれるユーザー ID がアクセスのために認証されるには、それらを名前変更しなければなりません。

ドメイン・リストを介してアクセスを制御することができます。例えば、ユーザーのドメインがリストに含まれない場合、そのユーザーは接続を許可されません。

注: DB2DOMAINLIST レジストリー変数が有効になるのは、データベース・マネージャー構成で CLIENT 認証が設定され、 Windows ドメイン環境の Windows デスクトップからのシングル・サインオンでこの認証が必要とされる場合のみです。**DB2DOMAINLIST** は、いくつかのバージョンの DB2 サーバーでサポートされていますが、クライアントもサーバーも Windows 環境に組み込まれていなければ、**DB2DOMAINLIST** は強制されません。

ドメイン・セキュリティーのサポート (Windows)

以下の例は、DB2 データベース管理システムが Windows ドメイン・セキュリティーをどのようにサポートするかを示しています。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。

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 50. 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。

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・セキュリティーがスキャンされる。	
	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。

表 50. 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
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。	

SYSADM 権限を持つユーザーの定義 (Windows)

sysadm_group データベース・マネージャー構成パラメーターが設定されていない場合 (つまり、NULL の場合)、特定のユーザーに SYSADM 権限が与えられます。

それらのユーザーは以下のとおりです。

- ローカル Administrators グループのメンバー
- ユーザーが定義されたロケーションで、それらのユーザーのグループを列挙するよう、DB2 データベース・マネージャーを構成している場合 (**DB2_GRP_LOOKUP** 環境変数を使用して、グループの列挙を構成できます)、ドメイン・コントローラーにおける Administrators グループのメンバー
- DB2ADMNS グループのメンバー (Windows 拡張セキュリティーが有効な場合)DB2ADMNS グループのロケーションは、インストール時に決定されます。
- LocalSystem アカウント

前述のデフォルト動作では不都合な場合があります。**sysadm_group** データベース・マネージャー構成パラメーターを使用すると、以下のいずれかの方式により、この動作をオーバーライドすることができます。

- DB2 サーバー・マシン上にローカル・グループを作成し、SYSADM 権限を付与するユーザー (ドメイン・ユーザーまたはローカル・ユーザー) をそのローカル・グループに追加します。DB2 データベース・マネージャーは、ローカル・マシン上のユーザーのグループを列挙するように構成される必要があります。
- ドメイン・グループを作成し、SYSADM 権限を付与するユーザーをそのドメイン・グループに追加します。DB2 データベース・マネージャーは、ユーザーが定義されたロケーション上でそれらのユーザーのグループを列挙するように構成される必要があります。

以下のコマンドを使用して、**sysadm_group** データベース・マネージャー構成パラメーターを上記のグループに更新します。

```
DB2 UPDATE DBM CFG USING SYSADM_GROUP group_name
DB2STOP
DB2START
```

Windows LocalSystem アカウントのサポート

Windows プラットフォームで、DB2 データベース・システムは、ローカル暗黙接続があるローカル・システム・アカウント (LSA) のコンテキストで実行するアプリケーションをサポートします。LocalSystem アカウントの許可 ID は SYSTEM です。

英語以外のバージョンの Windows オペレーティング・システムを使用している場合、LocalSystem アカウントの許可 ID に、無効な文字が含まれていないことを確認する必要があります。例えば、フランス語バージョンの Windows オペレーティン

グ・システムを使用している場合、LocalSystem アカウントは Système ですが、このアカウントには無効な文字 è が含まれているため、許可 ID として使用することはできません。

sysadm_group データベース・マネージャー構成パラメーターが NULL に設定されている場合、LocalSystem アカウントは、システム管理者 (SYSADM 権限を保持) と見なされます。

LocalSystem アカウントのコンテキストで稼働するアプリケーションが、SYSADM の有効範囲内にないデータベース・アクションを実行する必要がある場合、LocalSystem アカウントに必要なデータベース特権または権限を付与する必要があります。例えば、アプリケーションでデータベース管理者機能が必要な場合、LocalSystem アカウントに、GRANT (データベース権限) ステートメントを使用して DBADM 権限を付与します。

このアカウントの下で実行するアプリケーションを作成する開発者は、「SYS」で始まるスキーマ名のオブジェクトに関して DB2 データベース・システムに制約があることに注意する必要があります。そのため、DB2 データベース・オブジェクトを作成する DDL ステートメントがアプリケーションに含まれる場合、それらのアプリケーションは以下のように作成する必要があります。

- 静的照会では、QUALIFIER オプションの値をデフォルト (SYSTEM) 以外のものにしてバインドする必要があります。
- 動的照会では、作成するオブジェクトを DB2 データベース・マネージャーによってサポートされるスキーマ名で明示的に修飾するか、CURRENT SCHEMA レジスターをDB2 データベース・マネージャーによってサポートされるスキーマ名に設定する必要があります。

LocalSystem アカウントのグループ情報は、DB2 データベース・インスタンスが開始した後の最初のグループ・ルックアップ要求で収集され、インスタンスが再起動するまで更新されません。

DB2ADMNS と DB2USERS グループの使用による拡張 Windows セキュリティー

デフォルトで拡張セキュリティーは、Windows オペレーティング・システム上のすべての DB2 データベース製品 (IBM Data Server Runtime Client および DB2 Driver を除く) で有効です。IBM Data Server Runtime Client および DB2 Driver は、Windows プラットフォームにおける拡張セキュリティーをサポートしていません。

DB2 データベース製品をインストールすると、「**DB2 オブジェクトのためにオペレーティング・システム・セキュリティーを使用可能にする**」パネルに「**オペレーティング・システム・セキュリティーを使用可能にする**」チェック・ボックスが表示されます。このオプションを使用不可にしない限りは、インストーラーは、DB2ADMNS と DB2USERS という 2 つの新規グループを作成します。DB2ADMNS と DB2USERS はデフォルトのグループ名です。任意で、インストール時にこれらのグループに別の名前を付けることもできます。サイレント・インストールを選択した場合、インストール応答ファイル内でこれらの名前を変更することができます。システム上に既に存在するグループを使用するように選択すると、

それらのグループの特権が変更されるので注意してください。必要に応じて、次の表にリストされている特権が与えられます。これらのグループは、オペレーティング・システム・レベルでの保護のため使用されるもので、SYSADM、SYSMAINT、および SYSCTRL のような DB2 権限レベルとは関連付けられていないということを理解する必要があります。しかし、インストーラーや管理者の判断により、デフォルトの Administrator グループを使用する代わりに、データベース管理者は 1 つまたはすべての DB2 権限レベルに DB2ADMNS グループを使用することができます。 SYSADM グループを指定する場合は、DB2ADMNS グループにすることが推奨されています。この設定は、インストール時、またはそれ以降に管理者が実行できます。

注: DB2 管理者グループ (DB2ADMNS、またはインストール時に選択した名前) と DB2 ユーザー・グループ (DB2USERS、またはインストール時に選択した名前) は、ローカル・グループとしても、ドメイン・グループとしても指定できます。ただし、両方のグループを同じタイプにする必要があります (つまり、両方をローカルにするか、両方をドメインにするかのどちらかです)。

コンピューター名を変更する場合に、そのコンピューターのグループ DB2ADMNS と DB2USERS がローカル・コンピューター・グループであれば、グローバル・レジストリー DB2_ADMINGROUP と DB2_USERSGROUP を更新する必要があります。レジストリー変数を更新するには、コンピューターの名前を変更し、コンピューターを再始動させた後で、次のコマンドを実行します。

1. コマンド・プロンプトを開きます。
2. **db2extsec** コマンドを実行して、セキュリティー設定を更新します。

   ```
   db2extsec -a new computer name¥DB2ADMNS -u new computer name¥DB2USERS
   ```

注: Windows Vista 上では、DB2 データベース製品で拡張セキュリティーを有効にすると、DB2ADMNS グループに属しているユーザー以外はグラフィカルの DB2 管理ツールを実行できなくなります。加えて、DB2ADMNS グループのメンバーは、完全な管理者特権を使用してツールを起動する必要があります。管理者特権を使用してツールを起動するには、ショートカットを右クリックして「管理者として実行 (Run as administrator)」を選択します。

DB2ADMNS グループと DB2USERS グループによって取得される権限

DB2ADMNS と DB2USERS グループはメンバーに、以下の機能を提供します。

- DB2ADMNS

  すべての DB2 オブジェクトに対するフル・コントロール (保護されるオブジェクトについては次のリストを参照)

- DB2USERS

  インストール・ディレクトリーとインスタンス・ディレクトリーに配置されたすべての DB2 オブジェクトに対する読み取りおよび実行アクセス。ただし、データベース・システム・ディレクトリー以下のオブジェクトにはアクセスできません。また IPC リソースに対しては限定されたアクセスになります。

特定のオブジェクトに対して、必要に応じて追加的な特権が選択可能です (例えば、書き込み特権、ファイルの追加特権、ファイルの更新特権など)。このグループのメンバーは、データベース・システム・ディレクトリー以下のオブジェクトにはアクセスできません。

注: 実行アクセスの意味はオブジェクトにより異なります。例えば、**.dll** や **.exe** ファイルに対する実行アクセスは、そのファイルを実行する権限があるという意味ですが、ディレクトリーに対する実行アクセスは、そのディレクトリーを全検索する権限があるという意味です。

すべての DB2 管理者は、DB2ADMNS グループのメンバー (かつローカル Administrators グループのメンバー) にするのが理想です。ただし、これは必須要件ではありません。 DB2 データベース・システムへのアクセス要求をする他のすべてのメンバーは、DB2USERS グループのメンバーである必要 があります。ユーザーをこれらのグループに追加するには、以下のようにします。

1. 「ユーザー/パスワード・マネージャー・ツール」を起動します。
2. ユーザー名を選択し、リストから追加します
3. 「プロパティー」をクリックします。「プロパティー」ウィンドウで、「グループ・メンバーシップ」タブをクリックします。
4. 「その他」ラジオ・ボタンを選択します。
5. ドロップダウン・リストから適切なグループを選択します。

インストール後の拡張セキュリティー追加 (db2extsec コマンド)

拡張セキュリティーを有効にせずに DB2 データベース・システムをインストールした場合、**db2extsec** コマンドを実行してこれを有効にすることができます。. **db2extsec** コマンドを実行するには、ローカルの Administrators グループのメンバーであり、保護されたオブジェクトの ACL を変更する権限を持つ必要があります。

必要に応じて、**db2extsec** コマンドを複数回実行することができます。ただしその場合、**db2extsec** を*実行*するたびに 直後に **db2extsec -r** コマンドを実行をしない限り、拡張セキュリティーを使用不可に設定することはできません。

拡張セキュリティーの削除

注意:
拡張セキュリティーを使用可能にした後に削除する操作は、絶対に必要な場合以外は実行しないでください。

db2extsec -r コマンドを実行して拡張セキュリティーを削除できますが、削除が正常に完了するのは、拡張セキュリティーを有効にした後、データベースの作成、新規インスタンスの作成、表スペースの追加などの他のデータベース操作がされていない場合に限ります。拡張セキュリティー・オプションを削除する最も安全な方法は、DB2 データベース・システムをアンインストールし、データベース・ディレクトリーを含むすべての関連する DB2 ディレクトリーをすべて削除し、それから拡張セキュリティーを有効にしないで、DB2 データベース・システムを再インストールする方法です。

保護されたオブジェクト

DB2ADMNS と DB2USERS グループを使用して保護することができる*静的* オブジェクトには次のものがあります。

- ファイル・システム
  - ファイル
  - ディレクトリー
- サービス
- レジストリー・キー

DB2ADMNS と DB2USERS グループを使用して保護することができる*動的* オブジェクトには次のものがあります。

- 以下を含む IPC リソース
  - パイプ
  - セマフォー
  - イベント
- 共有メモリー

DB2ADMNS と DB2USERS グループの所有特権

DB2ADMNS と DB2USERS グループに割り当てられた特権を次の表に掲載します。

表 51. DB2ADMNS と DB2USERS グループの特権

特権	DB2ADMNS	DB2USERS	理由
トークン・オブジェクトの作成 (SeCreateTokenPrivilege)	Y	N	トークン操作 (一定のトークン操作が要求され、認証と許可に使用されます)
処理レベル・トークンの置換 (SeAssignPrimaryTokenPrivilege)	Y	N	他のユーザーとして処理の作成
割り当て量の引き上げ (SeIncreaseQuotaPrivilege)	Y	N	他のユーザーとして処理の作成
オペレーティング・システムの一部としての活動	Y	N	LogonUser (認証目的の Windows XP より前のバージョンでは、LogonUser API を実行するために必要です)
セキュリティー監査の生成 (SeSecurityPrivilege)	Y	N	監査とセキュリティー・ログの操作
ファイルと他のオブジェクトの所有権 (SeTakeOwnershipPrivilege)	Y	N	オブジェクト ACL の変更
スケジューリング優先順位の引き上げ (SeIncreaseBasePriorityPrivilege)	Y	N	処理作業セットの変更
ファイルとディレクトリーのバックアップ (SeBackupPrivilege)	Y	N	プロファイル/レジストリー操作 (次の特定のユーザー・プロファイルとレジストリー操作ルーチンが必要です。 LoadUserProfile、RegSaveKey(Ex) 、RegRestoreKey、 RegReplaceKey、RegLoadKey(Ex))

表 51. DB2ADMNS と DB2USERS グループの特権 (続き)

特権	DB2ADMNS	DB2USERS	理由
ファイルとディレクトリーのリストア (SeRestorePrivilege)	Y	N	プロファイル/レジストリー操作 (次の特定のユーザー・プロファイルとレジストリー操作ルーチンが必要です。LoadUserProfile、RegSaveKey(Ex)、RegRestoreKey、RegReplaceKey、RegLoadKey(Ex))
デバッグ・プログラム (SeDebugPrivilege)	Y	N	トークン操作 (一定のトークン操作が要求され、認証と許可に使用されます)
監査とセキュリティー・ログの管理 (SeAuditPrivilege)	Y	N	監査ログ・エントリーの生成
サービスとしてログオン (SeServiceLogonRight)	Y	N	サービスとして DB2 を実行します。
ネットワークからコンピューターにアクセス (SeNetworkLogonRight)	Y	Y	ネットワーク・クレデンシャルを許可します。(DB2 データベース・マネージャーに LOGON32_LOGON_NETWORK オプションを認証のため使用することを許可します。これは、パフォーマンスに影響します。)
認証後クライアントの偽装 (SeImpersonatePrivilege)	Y	N	クライアントの偽装 (Windows で DB2 クライアントの偽名を使用するため、ImpersonateLoggedOnUser、ImpersonateSelf、RevertToSelf などの特定の API の使用を許可する場合に必要)
メモリー内のロックされたページ (SeLockMemoryPrivilege)	Y	N	ラージ・ページのサポート
グローバル・オブジェクトの作成 (SeCreateGlobalPrivilege)	Y	Y	端末サーバー・サポート(Windows で必要)

Windows 2008 および Windows Vista 以降に関する考慮事項: ユーザー・アクセス制御フィーチャー

Windows 2008、Windows Vista、および Windows 7 のユーザー・アクセス制御 (UAC) フィーチャーは、次のような点で DB2 データベース・システムに影響を与えます。

完全な管理特権によるアプリケーションの開始

Windows 2008、Windows Vista、および Windows 7 の場合、デフォルトでは、ユーザーがローカル管理者であっても、標準的なユーザー権限だけでアプリケーションが開始されます。さらに多くの特権を持った状態でアプリケーションを開始するには、完全な管理特権で実行するコマンド・ウィンドウからコマンドを起動する必要があります。DB2 のインストール・プロセスでは、Windows 2008、Windows Vista、および Windows 7 ユーザー専用に「コマンド・ウィンドウ - 管理者」というショートカットが作成されます。管理コマンドを実行する場合は、このショートカットを起動することをお勧めします。

Windows 2008、Windows Vista、および Windows 7 の場合、完全な管理特権を持っていない状態で、コマンド・プロンプトやグラフィック・ツールから DB2 管理タスクを実行しようとすると、さまざまなエラー・メッセージが生成される可能性があります。これらのエラー・メッセージには、アクセスが拒否され、タスクが正常に完了しないという意味が含まれています。

実行しようとしたアクションが管理タスクと見なされるかどうかを確認するために、以下のいずれかが当てはまるかどうかをチェックしてください。

- SYSADM、SYSCTRL、SYSMAINT のいずれかの権限が必要です。
- レジストリーの HKLM ブランチにあるレジストリー・キーが変更されます。
- `Program Files` ディレクトリーの下にあるディレクトリーに書き込まれます。

例えば、以下のようなアクションはすべて管理タスクと見なされます。

- DB2 インスタンスの作成とドロップ
- DB2 インスタンスの開始と停止
- データベースの作成
- データベース・マネージャーの構成パラメーターまたは DB2 Administration Server (DAS) の構成パラメーターの更新
- CLI 構成パラメーターの更新とシステム・データ・ソース名 (DSN) の構成
- DB2 トレース機能の開始
- **db2pd** ユーティリティーの実行
- DB2 プロファイル・レジストリー変数の変更

問題を解決するには、完全な管理者特権で実行するコマンド・プロンプトやグラフィック・ツールから DB2 管理タスクを実行する必要があります。完全な管理者特権でコマンド・プロンプトやグラフィック・ツールを起動するには、前述のショートカットを右クリックして、「**管理者として実行 (Run as administrator)**」を選択します。

注: 拡張セキュリティーが有効になっている場合は、グラフィカル管理ツール (IBM Data Studio など) を起動するために、DB2ADMNS グループのメンバーになっていることも必要です。

ユーザー・データの場所

ユーザー・データ (インスタンス・ディレクトリーにあるファイルなど) は、`ProgramData¥IBM¥DB2¥`*copy_name* に格納されます (*copy_name* は DB2 コピーの名前で、デフォルトでは、インストール済みの最初のコピーの名前が DB2COPY1 になります)。Windows 2008、Windows Vista、および Windows 7 以外の Windows バージョンでは、`Documents and Settings¥All Users¥Application Data¥IBM¥DB2¥`*copy_name* にユーザー・データが格納されます。

DB2 および UNIX セキュリティー

UNIX プラットフォームに固有の、認識しておくべきセキュリティーの考慮事項が幾つかあります。

DB2 データベースは、ルートが直接データベース管理者として動作することをサポートしていません。データベース管理者としては **su - <instance owner>** を使用してください。

セキュリティーの理由で、インスタンス名を fenced ID として使用しないでください。ただし、fenced 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 fenced ID をインスタンス名に設定することができます。

推奨は、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。 fenced 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 Software Developer's Kit をインストールする場合、これは必須ではありません。

DB2 および Linux セキュリティー

Linux プラットフォームに固有の、認識しておくとよいセキュリティーの考慮事項が幾つかあります。

パスワード変更サポート (Linux)

DB2 データベース製品では、Linux オペレーティング・システムでパスワードを変更するためのサポートが用意されています。

このサポートをインプリメントするために、IBMOSchgpwdclient.so と IBMOSchgpwdserver.so というセキュリティー・プラグイン・ライブラリーが使用されています。

Linux でパスワード変更サポートを使用可能にするには、データベース・マネージャー構成パラメーター **clnt_pw_plugin** を IBMOSchgpwdclient に、**srvcon_pw_plugin** を IBMOSchgpwdserver にそれぞれ設定します。

さらに、/etc/pam.d ディレクトリーに db2 という PAM 構成ファイルを作成することも必要です。

パスワード変更プラグインのデプロイ (Linux)

Linux にインストールした DB2 データベース製品でパスワード変更サポートを有効にするには、セキュリティー・プラグイン IBMOSchgpwdclient と IBMOSchgpwdserver を使用するように DB2 インスタンスを構成する必要があります。

始める前に

プラグイン・ライブラリーは、以下のディレクトリーにあります。

- *INSTHOME*/sqllib/security*XX*/plugin/IBM/client/IBMOSchgpwdclient.so
- *INSTHOME*/sqllib/security*XX*/plugin/IBM/server/IBMOSchgpwdserver.so

INSTHOME は、インスタンス所有者のホーム・ディレクトリー、*securityXX* は、インスタンスのビット幅によって、security32 または security64 のいずれかになります。

手順

DB2 インスタンスにセキュリティー・プラグインをデプロイするには、以下の手順を実行します。

1. root 権限を持つユーザーとしてログインします。
2. PAM 構成ファイル /etc/pam.d/db2 を作成します。

   そのファイルに、システム管理者によって定義されている適切な規則のセットが含まれていることを確認します。例えば、SLES 9 でこれは以下のように使用できます。

   ```
   auth     required pam_unix2.so    nullok
   account  required pam_unix2.so
   password required pam_pwcheck.so  nullok tries=1
   password required pam_unix2.so    nullok use_authtok use_first_pass
   session  required pam_unix2.so
   ```

   そして、RHEL では以下のように使用できます。

   ```
   #%PAM-1.0
   auth     required   /lib/security/$ISA/pam_env.so
   auth     sufficient /lib/security/$ISA/pam_unix.so likeauth nullok
   auth     required   /lib/security/$ISA/pam_deny.so

   account  required   /lib/security/$ISA/pam_unix.so
   account  sufficient /lib/security/$ISA/pam_succeed_if.so uid < 100 quiet
   account  required   /lib/security/$ISA/pam_permit.so

   password requisite  /lib/security/$ISA/pam_cracklib.so retry=3 dcredit=-1
         ucredit=-1
   password sufficient /lib/security/$ISA/pam_unix.so nullok use_authtok md5
         shadow remember=3
   password required   /lib/security/$ISA/pam_deny.so

   session  required   /lib/security/$ISA/pam_limits.so
   session  required   /lib/security/$ISA/pam_unix.so
   ```

3. DB2 インスタンスでセキュリティー・プラグインを有効にします。
   a. データベース・マネージャー構成パラメーター **SRVCON_PW_PLUGIN** を値 IBMOSchgpwdserver で更新します。

      ```
      db2 update dbm cfg using srvcon_pw_plugin IBMOSchgpwdserver
      ```

   b. データベース・マネージャー構成パラメーター **CLNT_PW_PLUGIN** を値 IBMOSchgpwdclient で更新します。

      ```
      db2 update dbm cfg using CLNT_PW_PLUGIN IBMOSchgpwdclient
      ```

   c. データベース・マネージャー構成パラメーター **SRVCON_AUTH** が、 CLIENT、SERVER、SERVER_ENCRYPT、DATA_ENCRYPT、DATA_ENCRYPT_CMP のいずれかの値に設定されているか、データベース・マネージャー構成パラメーター **SRVCON_AUTH** が NOT_SPECIFIED の値に設定されていて、**AUTHENTICATION** が、 CLIENT、SERVER、SERVER_ENCRYPT、DATA_ENCRYPT、DATA_ENCRYPT_CMP のいずれかの値に設定されていることを確認します。

付録 A. DB2 技術情報の概説

DB2 技術情報は、さまざまな方法でアクセスすることが可能な、各種形式で入手できます。

DB2 技術情報は、以下のツールと方法を介して利用できます。

- DB2インフォメーション・センター
  - トピック (タスク、概念、およびリファレンス・トピック)
  - サンプル・プログラム
  - チュートリアル
- DB2 資料
  - PDF ファイル (ダウンロード可能)
  - PDF ファイル (DB2 PDF DVD に含まれる)
  - 印刷資料
- コマンド行ヘルプ
  - コマンド・ヘルプ
  - メッセージ・ヘルプ

注: DB2 インフォメーション・センターのトピックは、PDF やハードコピー資料よりも頻繁に更新されます。最新の情報を入手するには、資料の更新が発行されたときにそれをインストールするか、ibm.com にある DB2 インフォメーション・センターを参照してください。

技術資料、ホワイト・ペーパー、IBM Redbooks® 資料などのその他の DB2 技術情報には、オンライン (ibm.com) でアクセスできます。 DB2 Information Management ソフトウェア・ライブラリー・サイト (http://www.ibm.com/software/data/sw-library/) にアクセスしてください。

資料についてのフィードバック

DB2 の資料についてのお客様からの貴重なご意見をお待ちしています。DB2 の資料を改善するための提案については、db2docs@ca.ibm.com まで E メールを送信してください。DB2 の資料チームは、お客様からのフィードバックすべてに目を通しますが、直接お客様に返答することはありません。お客様が関心をお持ちの内容について、可能な限り具体的な例を提供してください。特定のトピックまたはヘルプ・ファイルについてのフィードバックを提供する場合は、そのトピック・タイトルおよび URL を含めてください。

DB2 お客様サポートに連絡する場合には、この E メール・アドレスを使用しないでください。資料を参照しても、DB2 の技術的な問題が解決しない場合は、お近くの IBM サービス・センターにお問い合わせください。

DB2 テクニカル・ライブラリー (ハードコピーまたは PDF 形式)

以下の表は、IBM Publications Center (www.ibm.com/e-business/linkweb/publications/servlet/pbi.wss) から利用できる DB2 ライブラリーについて説明しています。英語および翻訳された DB2 バージョン 10.1 のマニュアル (PDF 形式) は、www.ibm.com/support/docview.wss?rs=71&uid=swg27009474 からダウンロードできます。

この表には印刷資料が入手可能かどうかが示されていますが、国または地域によっては入手できない場合があります。

資料番号は、資料が更新される度に大きくなります。資料を参照する際は、以下にリストされている最新版であることを確認してください。

注: *DB2 インフォメーション・センター*は、PDF やハードコピー資料よりも頻繁に更新されます。

表 52. DB2 の技術情報

資料名	資料番号	印刷資料が入手可能かどうか	最終更新
管理 API リファレンス	SA88-4671-00	入手可能	2012 年 4 月
管理ルーチンおよびビュー	SA88-4672-01	入手不可	2013 年 1 月
コール・レベル・インターフェース ガイドおよびリファレンス 第 1 巻	SA88-4676-01	入手可能	2013 年 1 月
コール・レベル・インターフェース ガイドおよびリファレンス 第 2 巻	SA88-4677-01	入手可能	2013 年 1 月
コマンド・リファレンス	SA88-4673-01	入手可能	2013 年 1 月
データベース: 管理の概念および構成リファレンス	SA88-4662-01	入手可能	2013 年 1 月
データ移動ユーティリティー: ガイドおよびリファレンス	SA88-4693-01	入手可能	2013 年 1 月
データベースのモニタリング ガイドおよびリファレンス	SA88-4663-01	入手可能	2013 年 1 月
データ・リカバリーと高可用性 ガイドおよびリファレンス	SA88-4694-01	入手可能	2013 年 1 月
データベース・セキュリティー・ガイド	SA88-4695-01	入手可能	2013 年 1 月

表 52. DB2 の技術情報 (続き)

資料名	資料番号	印刷資料が入手可能かどうか	最終更新
DB2 ワークロード管理ガイドおよびリファレンス	SA88-4685-01	入手可能	2013 年 1 月
ADO.NET および OLE DB アプリケーションの開発	SA88-4665-01	入手可能	2013 年 1 月
組み込み SQL アプリケーションの開発	SA88-4666-01	入手可能	2013 年 1 月
Java アプリケーションの開発	SA88-4669-01	入手可能	2013 年 1 月
Perl、PHP、Python および Ruby on Rails アプリケーションの開発	SA88-4670-00	入手不可	2012 年 4 月
IBM データ・サーバー用の RDF アプリケーション開発	SA88-5083-00	入手可能	2013 年 1 月
SQL および外部ルーチンの開発	SA88-4667-01	入手可能	2013 年 1 月
データベース・アプリケーション開発の基礎	GI88-4279-01	入手可能	2013 年 1 月
DB2 インストールおよび管理 概説 (Linux および Windows 版)	GI88-4280-00	入手可能	2012 年 4 月
グローバリゼーション・ガイド	SA88-4696-00	入手可能	2012 年 4 月
DB2 サーバー機能 インストール	GA88-4679-01	入手可能	2013 年 1 月
IBM データ・サーバー・クライアント機能 インストール	GA88-4680-00	入手不可	2012 年 4 月
メッセージ・リファレンス 第 1 巻	SA88-4688-01	入手不可	2013 年 1 月
メッセージ・リファレンス 第 2 巻	SA88-4689-01	入手不可	2013 年 1 月
Net Search Extender 管理およびユーザーズ・ガイド	SA88-4691-01	入手不可	2013 年 1 月
パーティションおよびクラスタリングのガイド	SA88-4697-01	入手可能	2013 年 1 月
Preparation Guide for DB2 10.1 Fundamentals Exam 610	SC27-4540-00	入手不可	2013 年 1 月

表 52. DB2 の技術情報 (続き)

資料名	資料番号	印刷資料が入手可能かどうか	最終更新
Preparation Guide for DB2 10.1 DBA for Linux, UNIX, and Windows Exam 611	SC27-4541-00	入手不可	2013 年 1 月
pureXML ガイド	SA88-4686-01	入手可能	2013 年 1 月
Spatial Extender ユーザーズ・ガイドおよびリファレンス	SA88-4690-00	入手不可	2012 年 4 月
SQL プロシージャー言語: アプリケーションのイネーブルメントおよびサポート	SA88-4668-01	入手可能	2013 年 1 月
SQL リファレンス 第 1 巻	SA88-4674-01	入手可能	2013 年 1 月
SQL リファレンス 第 2 巻	SA88-4675-01	入手可能	2013 年 1 月
Text Search ガイド	SA88-4692-01	入手可能	2013 年 1 月
問題判別およびデータベース・パフォーマンスのチューニング	SA88-4664-01	入手可能	2013 年 1 月
DB2 バージョン 10.1 へのアップグレード	SA88-4678-01	入手可能	2013 年 1 月
DB2 バージョン 10.1 の新機能	SA88-4684-01	入手可能	2013 年 1 月
XQuery リファレンス	SA88-4687-01	入手不可	2013 年 1 月

表 53. DB2 Connect 固有の技術情報

資料名	資料番号	印刷資料が入手可能かどうか	最終更新
DB2 Connect Personal Edition インストールおよび構成	SA88-4681-00	入手可能	2012 年 4 月
DB2 Connect サーバー機能 インストールおよび構成	SA88-4682-01	入手可能	2013 年 1 月
DB2 Connect ユーザーズ・ガイド	SA88-4683-01	入手可能	2013 年 1 月

コマンド行プロセッサーから SQL 状態ヘルプを表示する

DB2 製品は、SQL ステートメントの結果として生じる可能性がある状態に対応した SQLSTATE 値を戻します。 SQLSTATE ヘルプは、SQL 状態および SQL 状態クラス・コードの意味を説明します。

手順

SQL 状態ヘルプを開始するには、コマンド行プロセッサーを開いて以下のように入力します。

```
? sqlstate または ? class code
```

ここで、*sqlstate* は有効な 5 桁の SQL 状態を、*class code* は SQL 状態の最初の 2 桁を表します。
例えば、`? 08003` を指定すると SQL 状態 08003 のヘルプが表示され、 `? 08` を指定するとクラス・コード 08 のヘルプが表示されます。

異なるバージョンの DB2 インフォメーション・センターへのアクセス

他のバージョンの DB2 製品の資料は、ibm.com® のそれぞれのインフォメーション・センターにあります。

このタスクについて

DB2 バージョン 10.1 のトピックを扱っている *DB2 インフォメーション・センター* の URL は、http://publib.boulder.ibm.com/infocenter/db2luw/v10r1 です。

DB2 バージョン 9.8 のトピックを扱っている *DB2 インフォメーション・センター* の URL は、http://publib.boulder.ibm.com/infocenter/db2luw/v9r8/ です。

DB2 バージョン 9.7 のトピックを扱っている *DB2 インフォメーション・センター* の URL は、http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/ です。

DB2 バージョン 9.5 のトピックを扱っている *DB2 インフォメーション・センター* の URL は、http://publib.boulder.ibm.com/infocenter/db2luw/v9r5 です。

DB2 バージョン 9.1 のトピックを扱っている *DB2 インフォメーション・センター* の URL は、http://publib.boulder.ibm.com/infocenter/db2luw/v9/ です。

DB2 バージョン 8 のトピックについては、*DB2 インフォメーション・センター* の URL (http://publib.boulder.ibm.com/infocenter/db2luw/v8/) を参照してください。

コンピューターまたはイントラネット・サーバーにインストールされた DB2 インフォメーション・センターの更新

ローカルにインストールした DB2 インフォメーション・センターは、定期的に更新する必要があります。

始める前に

DB2 バージョン 10.1 インフォメーション・センターが既にインストール済みである必要があります。詳しくは、「*DB2 サーバー機能 インストール*」の『DB2 セットアップ・ウィザードによる DB2 インフォメーション・センターのインストール』のトピックを参照してください。インフォメーション・センターのインストールに適用されるすべての前提条件と制約事項は、インフォメーション・センターの更新にも適用されます。

このタスクについて

既存の DB2 インフォメーション・センターは、自動で更新することも手動で更新することもできます。

- 自動更新は、既存のインフォメーション・センターのフィーチャーと言語を更新します。自動更新を使用すると、手動更新と比べて、更新中にインフォメーション・センターが使用できなくなる時間が短くなるというメリットがあります。さらに、自動更新は、定期的に実行する他のバッチ・ジョブの一部として実行されるように設定することができます。
- 手動更新は、既存のインフォメーション・センターのフィーチャーと言語の更新に使用できます。自動更新は更新処理中のダウン時間を減らすことができますが、フィーチャーまたは言語を追加する場合は手動処理を使用する必要があります。例えば、ローカルのインフォメーション・センターが最初は英語とフランス語でインストールされており、その後ドイツ語もインストールすることにした場合、手動更新でドイツ語をインストールし、同時に、既存のインフォメーション・センターのフィーチャーおよび言語を更新できます。しかし、手動更新ではインフォメーション・センターを手動で停止、更新、再始動する必要があります。更新処理の間はずっと、インフォメーション・センターは使用できなくなります。自動更新処理では、インフォメーション・センターは、更新を行った後に、インフォメーション・センターを再始動するための停止が発生するだけで済みます。

このトピックでは、自動更新のプロセスを詳しく説明しています。手動更新の手順については、『コンピューターまたはイントラネット・サーバーにインストールされた DB2 インフォメーション・センターの手動更新』のトピックを参照してください。

手順

コンピューターまたはイントラネット・サーバーにインストールされている DB2 インフォメーション・センターを自動更新する手順を以下に示します。

1. Linux オペレーティング・システムの場合、次のようにします。
   a. インフォメーション・センターがインストールされているパスにナビゲートします。デフォルトでは、DB2 インフォメーション・センターは、`/opt/ibm/db2ic/V10.1` ディレクトリーにインストールされています。
   b. インストール・ディレクトリーから `doc/bin` ディレクトリーにナビゲートします。
   c. 次のように `update-ic` スクリプトを実行します。

      ```
      update-ic
      ```
2. Windows オペレーティング・システムの場合、次のようにします。
   a. コマンド・ウィンドウを開きます。
   b. インフォメーション・センターがインストールされているパスにナビゲートします。デフォルトでは、DB2 インフォメーション・センターは、`<Program Files>¥IBM¥DB2 Information Center¥バージョン 10.1` ディレクトリーにインストールされています (`<Program Files>` は「Program Files」ディレクトリーのロケーション)。

c. インストール・ディレクトリーから doc¥bin ディレクトリーにナビゲートします。

d. 次のように update-ic.bat ファイルを実行します。

```
update-ic.bat
```

タスクの結果

DB2 インフォメーション・センターが自動的に再始動します。更新が入手可能な場合、インフォメーション・センターに、更新された新しいトピックが表示されます。インフォメーション・センターの更新が入手可能でなかった場合、メッセージがログに追加されます。ログ・ファイルは、doc¥eclipse¥configuration ディレクトリーにあります。ログ・ファイル名はランダムに生成された名前です。例えば、1239053440785.log のようになります。

コンピューターまたはイントラネット・サーバーにインストールされた DB2 インフォメーション・センターの手動更新

DB2 インフォメーション・センターをローカルにインストールしている場合は、IBM から資料の更新を入手してインストールすることができます。

このタスクについて

ローカルにインストールされた *DB2 インフォメーション・センター* を手動で更新するには、以下のことを行う必要があります。

1. コンピューター上の *DB2 インフォメーション・センター* を停止し、インフォメーション・センターをスタンドアロン・モードで再始動します。インフォメーション・センターをスタンドアロン・モードで実行すると、ネットワーク上の他のユーザーがそのインフォメーション・センターにアクセスできなくなります。これで、更新を適用できるようになります。DB2 インフォメーション・センターのワークステーション・バージョンは、常にスタンドアロン・モードで実行されます。を参照してください。
2. 「更新」機能を使用することにより、どんな更新が利用できるかを確認します。インストールしなければならない更新がある場合は、「更新」機能を使用してそれを入手およびインストールできます。

   注: ご使用の環境において、インターネットに接続されていないマシンに *DB2 インフォメーション・センター* の更新をインストールする必要がある場合、インターネットに接続されていて *DB2 インフォメーション・センター* がインストールされているマシンを使用して、更新サイトをローカル・ファイル・システムにミラーリングしてください。ネットワーク上の多数のユーザーが資料の更新をインストールする場合にも、更新サイトをローカルにミラーリングして、更新サイト用のプロキシーを作成することにより、個々のユーザーが更新を実行するのに要する時間を短縮できます。
   更新パッケージが入手可能な場合、「更新」機能を使用してパッケージを入手します。ただし、「更新」機能は、スタンドアロン・モードでのみ使用できます。
3. スタンドアロンのインフォメーション・センターを停止し、コンピューター上の *DB2 インフォメーション・センター* を再開します。

注: Windows 2008、Windows Vista (およびそれ以上) では、このセクションの後の部分でリストされているコマンドは管理者として実行する必要があります。完全な管理者特権でコマンド・プロンプトまたはグラフィカル・ツールを開くには、ショートカットを右クリックしてから、「**管理者として実行**」を選択します。

手順

コンピューターまたはイントラネット・サーバーにインストール済みの *DB2* インフォメーション・センター を更新するには、以下のようにします。

1. *DB2* インフォメーション・センター を停止します。
   - Windows では、「**スタート**」 > 「**コントロール パネル**」 > 「**管理ツール**」 > 「**サービス**」をクリックします。次に、「**DB2 インフォメーション・センター**」サービスを右クリックして「**停止**」を選択します。
   - Linux では、以下のコマンドを入力します。

     ```
     /etc/init.d/db2icdv10 stop
     ```
2. インフォメーション・センターをスタンドアロン・モードで開始します。
   - Windows の場合:
     a. コマンド・ウィンドウを開きます。
     b. インフォメーション・センターがインストールされているパスにナビゲートします。デフォルトでは、*DB2* インフォメーション・センター は、*Program_Files*¥IBM¥DB2 Information Center¥バージョン 10.1 ディレクトリーにインストールされています (*Program_Files* は Program Files ディレクトリーのロケーション)。
     c. インストール・ディレクトリーから doc¥bin ディレクトリーにナビゲートします。
     d. 次のように help_start.bat ファイルを実行します。

        ```
        help_start.bat
        ```
   - Linux の場合:
     a. インフォメーション・センターがインストールされているパスにナビゲートします。デフォルトでは、*DB2* インフォメーション・センター は、/opt/ibm/db2ic/V10.1 ディレクトリーにインストールされています。
     b. インストール・ディレクトリーから doc/bin ディレクトリーにナビゲートします。
     c. 次のように help_start スクリプトを実行します。

        ```
        help_start
        ```

   システムのデフォルト Web ブラウザーが開き、スタンドアロンのインフォメーション・センターが表示されます。
3. 「**更新**」ボタン () をクリックします。(ブラウザーで JavaScript が有効になっている必要があります。) インフォメーション・センターの右側のパネルで、「**更新の検索**」をクリックします。 既存の文書に対する更新のリストが表示されます。
4. インストール・プロセスを開始するには、インストールする更新をチェックして選択し、「**更新のインストール**」をクリックします。
5. インストール・プロセスが完了したら、「**完了**」をクリックします。

6. 次のようにして、スタンドアロンのインフォメーション・センターを停止します。
   - Windows の場合は、インストール・ディレクトリーの doc¥bin ディレクトリーにナビゲートしてから、次のように help_end.bat ファイルを実行します。

     ```
     help_end.bat
     ```

     注: help_end バッチ・ファイルには、help_start バッチ・ファイルを使用して開始したプロセスを安全に停止するのに必要なコマンドが含まれています。help_start.bat は、Ctrl-C や他の方法を使用して停止しないでください。
   - Linux の場合は、インストール・ディレクトリーの doc/bin ディレクトリーにナビゲートしてから、次のように help_end スクリプトを実行します。

     ```
     help_end
     ```

     注: help_end スクリプトには、help_start スクリプトを使用して開始したプロセスを安全に停止するのに必要なコマンドが含まれています。他の方法を使用して、help_start スクリプトを停止しないでください。
7. *DB2 インフォメーション・センター* を再開します。
   - Windows では、「**スタート**」 > 「**コントロール パネル**」 > 「**管理ツール**」 > 「**サービス**」をクリックします。次に、「**DB2 インフォメーション・センター**」サービスを右クリックして「**開始**」を選択します。
   - Linux では、以下のコマンドを入力します。

     ```
     /etc/init.d/db2icdv10 start
     ```

タスクの結果

更新された *DB2 インフォメーション・センター* に、更新された新しいトピックが表示されます。

DB2 チュートリアル

DB2 チュートリアルは、DB2 データベース製品のさまざまな機能について学習するための支援となります。この演習をとおして段階的に学習することができます。

はじめに

インフォメーション・センター (http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/) から、このチュートリアルの XHTML 版を表示できます。

演習の中で、サンプル・データまたはサンプル・コードを使用する場合があります。個々のタスクの前提条件については、チュートリアルを参照してください。

DB2 チュートリアル

チュートリアルを表示するには、タイトルをクリックします。

「*pureXML ガイド* 」の『**pureXML®**』
: XML データを保管し、ネイティブ XML データ・ストアに対して基本的な操作を実行できるように、DB2 データベースをセットアップします。

DB2 トラブルシューティング情報

DB2 データベース製品を使用する際に役立つ、トラブルシューティングおよび問題判別に関する広範囲な情報を利用できます。

DB2 の資料

トラブルシューティング情報は、「*問題判別およびデータベース・パフォーマンスのチューニング*」または *DB2 インフォメーション・センター* の『データベースの基本』セクションにあります。ここには、以下の情報が記載されています。

- DB2 診断ツールおよびユーティリティーを使用した、問題の切り分け方法および識別方法に関する情報。
- 最も一般的な問題のうち、いくつかの解決方法。
- DB2 データベース製品で発生する可能性のある、その他の問題の解決に役立つアドバイス。

IBM サポート・ポータル

現在問題が発生していて、考えられる原因とソリューションを見つけるには、IBM サポート・ポータルを参照してください。 Technical Support サイトには、最新の DB2 資料、TechNotes、プログラム診断依頼書 (APAR またはバグ修正)、フィックスパック、およびその他のリソースへのリンクが用意されています。この知識ベースを活用して、問題に対する有効なソリューションを探し出すことができます。

IBM サポート・ポータル (http://www.ibm.com/support/entry/portal/Overview/Software/Information_Management/DB2_for_Linux,_UNIX_and_Windows) にアクセスしてください。

ご利用条件

これらの資料は、以下の条件に同意していただける場合に限りご使用いただけます。

適用度: これらのご利用条件は、IBM Web サイトのあらゆるご利用条件に追加で適用されるものです。

個人使用: これらの資料は、すべての著作権表示その他の所有権表示をしていただくことを条件に、非商業的な個人による使用目的に限り複製することができます。ただし、IBM の明示的な承諾をえずに、これらの資料またはその一部について、二次的著作物を作成したり、配布 (頒布、送信を含む) または表示 (上映を含む) することはできません。

商業的使用: これらの資料は、すべての著作権表示その他の所有権表示をしていただくことを条件に、お客様の企業内に限り、複製、配布、および表示することができます。 ただし、IBM の明示的な承諾をえずにこれらの資料の二次的著作物を作成したり、お客様の企業外で資料またはその一部を複製、配布、または表示することはできません。

権利: ここで明示的に許可されているもの以外に、資料や資料内に含まれる情報、データ、ソフトウェア、またはその他の知的所有権に対するいかなる許可、ライセンス、または権利を明示的にも黙示的にも付与するものではありません。

資料の使用が IBM の利益を損なうと判断された場合や、上記の条件が適切に守られていないと判断された場合、IBM はいつでも自らの判断により、ここで与えた許可を撤回できるものとさせていただきます。

お客様がこの情報をダウンロード、輸出、または再輸出する際には、米国のすべての輸出入関連法規を含む、すべての関連法規を遵守するものとします。

IBM は、これらの資料の内容についていかなる保証もしません。これらの資料は、特定物として現存するままの状態で提供され、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含むすべての明示もしくは黙示の保証責任なしで提供されます。

IBM の商標: IBM、IBM ロゴおよび ibm.com は、世界の多くの国で登録された International Business Machines Corporation の商標です。他の製品名およびサービス名等は、それぞれ IBM または各社の商標である場合があります。現時点での IBM の商標リストについては、http://www.ibm.com/legal/copytrade.shtml をご覧ください。

付録 B. 特記事項

本書は米国 IBM が提供する製品およびサービスについて作成したものです。IBM 以外の製品に関する情報は、本書の最初の発行時点で入手可能な情報に基づいており、変更される場合があります。

本書に記載の製品、サービス、または機能が日本においては提供されていない場合があります。日本で利用可能な製品、サービス、および機能については、日本 IBM の営業担当員にお尋ねください。本書で IBM 製品、プログラム、またはサービスに言及していても、その IBM 製品、プログラム、またはサービスのみが使用可能であることを意味するものではありません。これらに代えて、IBM の知的所有権を侵害することのない、機能的に同等の製品、プログラム、またはサービスを使用することができます。ただし、IBM 以外の製品とプログラムの操作またはサービスの評価および検証は、お客様の責任で行っていただきます。

IBM は、本書に記載されている内容に関して特許権 (特許出願中のものを含む) を保有している場合があります。本書の提供は、お客様にこれらの特許権について実施権を許諾することを意味するものではありません。実施権についてのお問い合わせは、書面にて下記宛先にお送りください。

〒103-8510
東京都中央区日本橋箱崎町19番21号
日本アイ・ビー・エム株式会社
法務・知的財産
知的財産権ライセンス渉外

以下の保証は、国または地域の法律に沿わない場合は、適用されません。 IBM およびその直接または間接の子会社は、本書を特定物として現存するままの状態で提供し、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含むすべての明示もしくは黙示の保証責任を負わないものとします。国または地域によっては、法律の強行規定により、保証責任の制限が禁じられる場合、強行規定の制限を受けるものとします。

この情報には、技術的に不適切な記述や誤植を含む場合があります。本書は定期的に見直され、必要な変更は本書の次版に組み込まれます。 IBM は予告なしに、随時、この文書に記載されている製品またはプログラムに対して、改良または変更を行うことがあります。

本書において 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 Limited
U59/3600
3600 Steeles Avenue East
Markham, Ontario L3R 9Z7
CANADA

本プログラムに関する上記の情報は、適切な使用条件の下で使用することができますが、有償の場合もあります。

本書で説明されているライセンス・プログラムまたはその他のライセンス資料は、IBM 所定のプログラム契約の契約条項、IBM プログラムのご使用条件、またはそれと同等の条項に基づいて、IBM より提供されます。

この文書に含まれるいかなるパフォーマンス・データも、管理環境下で決定されたものです。そのため、他の操作環境で得られた結果は、異なる可能性があります。一部の測定が、開発レベルのシステムで行われた可能性がありますが、その測定値が、一般に利用可能なシステムのものと同じである保証はありません。さらに、一部の測定値が、推定値である可能性があります。実際の結果は、異なる可能性があります。お客様は、お客様の特定の環境に適したデータを確かめる必要があります。

IBM 以外の製品に関する情報は、その製品の供給者、出版物、もしくはその他の公に利用可能なソースから入手したものです。 IBM は、それらの製品のテストは行っておりません。したがって、他社製品に関する実行性、互換性、またはその他の要求については確証できません。 IBM 以外の製品の性能に関する質問は、それらの製品の供給者にお願いします。

IBM の将来の方向または意向に関する記述については、予告なしに変更または撤回される場合があり、単に目標を示しているものです。

本書には、日常の業務処理で用いられるデータや報告書の例が含まれています。より具体性を与えるために、それらの例には、個人、企業、ブランド、あるいは製品などの名前が含まれている場合があります。これらの名称はすべて架空のものであり、名称や住所が類似する企業が実在しているとしても、それは偶然にすぎません。

著作権使用許諾:

本書には、様々なオペレーティング・プラットフォームでのプログラミング手法を例示するサンプル・アプリケーション・プログラムがソース言語で掲載されています。お客様は、サンプル・プログラムが書かれているオペレーティング・プラットフォームのアプリケーション・プログラミング・インターフェースに準拠したアプリケーション・プログラムの開発、使用、販売、配布を目的として、いかなる形式においても、IBM に対価を支払うことなくこれを複製し、改変し、配布することができます。このサンプル・プログラムは、あらゆる条件下における完全なテストを経ていません。従って IBM は、これらのサンプル・プログラムについて信頼性、

利便性もしくは機能性があることをほのめかしたり、保証することはできません。サンプル・プログラムは、現存するままの状態で提供されるものであり、いかなる種類の保証も提供されません。 IBM は、これらのサンプル・プログラムの使用から生ずるいかなる損害に対しても責任を負いません。

それぞれの複製物、サンプル・プログラムのいかなる部分、またはすべての派生した創作物には、次のように、著作権表示を入れていただく必要があります。

© (*お客様の会社名*) (*西暦年*). このコードの一部は、IBM Corp. のサンプル・プログラムから取られています。 © Copyright IBM Corp. _*年を入れる*_. All rights reserved.

商標

IBM、IBM ロゴおよび ibm.com は、世界の多くの国で登録された International Business Machines Corporation の商標です。他の製品名およびサービス名等は、それぞれ IBM または各社の商標である場合があります。現時点での IBM の商標リストについては、http://www.ibm.com/legal/copytrade.shtml をご覧ください。

以下は、それぞれ各社の商標または登録商標です。

- Linux は、Linus Torvalds の米国およびその他の国における商標です。
- Java およびすべての Java 関連の商標およびロゴは Oracle やその関連会社の米国およびその他の国における商標または登録商標です。
- UNIX は The Open Group の米国およびその他の国における登録商標です。
- インテル、Intel、Intel ロゴ、Intel Inside、Intel Inside ロゴ、Celeron、Intel SpeedStep、Itanium、Pentium は、Intel Corporation または子会社の米国およびその他の国における商標または登録商標です。
- Microsoft、 Windows、 Windows NT、および Windows ロゴは、Microsoft Corporation の米国およびその他の国における商標です。

索引

日本語, 数字, 英字, 特殊文字の順に配列されています。なお, 濁音と半濁音は清音と同等に扱われています。

[ア行]

[カ行]

[サ行]

[タ行]

[ナ行]

[ハ行]

[マ行]

[ヤ行]

[ラ行]

A

B

C

D

E

F

G

H

I

K

L

N

O

P

Q

R

S

T

U

V

W

X

[特殊文字]

Printed in Japan

SA88-4695-01

日本アイ・ビー・エム株式会社
〒103-8510 東京都中央区日本橋箱崎町19-21

Spine information:

IBM DB2 10.1 for Linux, UNIX, and Windows

データベース・セキュリティー・ガイド