Oracle® Database
セキュリティ・ガイド
19c

F16130-13(原本部品番号:E96299-44)

2023年11月

タイトルおよび著作権情報

Oracle Databaseセキュリティ・ガイド 19c

F16130-13

原本著者: Patricia Huey

原本協力著者: Sumit Jeloka

原本協力者: Suraj Adhikari、Thomas Baby、Tammy Bednar、Todd Bottger、Sanjay Bharadwaj、Leo Cloutier、Sudha Duraiswamy、Naveen Gopal、Rishabh Gupta、Yong Hu、Srinidhi Kayoor 、Peter Knaggs、Andre Kruklikov、Sanjay Kulhari、Anup A.Kumar、Bryn Llewellyn、Dah-Yoh Lim、Rahil Mir、Hari Mohankumar、Gopal Mulagund、Abhishek Munnolimath、Paul Needham、Robert Pang、Dilip Raj、Kumar Rajamani、Kathy Rich、Saikat Saha、Vipin Samar、Saravana Soundararajan、James Spiller、Srividya Tata、Kamal Tbeileh、Can Tuzla、Anand Verma、Patrick Wheeler、Peter H.Wong

目次

表一覧

はじめに

『Oracle Databaseセキュリティ・ガイド』へようこそ。このマニュアルでは、デフォルトのデータベース機能を使用してOracle Databaseのセキュリティを構成する方法について説明します。

対象読者

『Oracle Databaseセキュリティ・ガイド』は、データベース管理者(DBA)、セキュリティ管理者、アプリケーション開発者、そして他にも次の業務を安全かつ効率的に行う任務を負う担当者を対象としています。

次の領域があります。

- 組織のデータ、ユーザーおよびアプリケーションを不慮、不適切または不正な操作から保護するセキュリティ・ポリシーの設計および実装
- 不適切または不正な操作に対する監査とアカウンタビリティに関するポリシーと手続きの作成および実施
- ユーザー・アカウント、パスワード、ロールおよび権限の作成、メンテナンス、停止
- 様々なコンピューティング・モデルに必要なサービスを安全に提供するアプリケーションの開発、およびデータベースやディレクトリ・サービスを活用した、効率と操作性の最大化

このマニュアルの読者は、データベースの使用方法や使用する理由を基本的に理解していることを前提としています。また、SQLに関する基本的な知識も必要です。

親トピック: はじめに

ドキュメントのアクセシビリティ

オラクルのアクセシビリティについての詳細情報は、Oracle Accessibility ProgramのWebサイト(http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc)を参照してください。

Oracle Supportへのアクセス

サポートをご契約のお客様には、My Oracle Supportを通して電子支援サービスを提供しています。詳細情報は(http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info)か、聴覚に障害のあるお客様は(http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs)を参照してください。

親トピック: はじめに

関連ドキュメント

セキュリティに関連する情報の詳細は、次のOracleドキュメントを参照してください。

- Oracle Database管理者ガイド
- Oracle Database概要

- Oracle Databaseリファレンス
- Oracle Multitenant管理者ガイド

このマニュアルの多くの例では、Oracle Databaseのインストール時に作成できるシードPDBのサンプル・スキーマを使用しています。スキーマの作成方法および使用方法の詳細は、『Oracle Databaseサンプル・スキーマ』を参照してください。

Oracleテクニカル・サービス

製品データ・シート、よくある質問、最新の製品ドキュメントへのリンク、製品のダウンロードおよびその他の関連ドキュメントは、Oracle Technical Resources (旧称: Oracle Technology Network)からダウンロードできます。Oracleテクニカル・サービスを使用するには、オンライン登録が必要です。登録は、次の場所から無償で行えます。

```
https://www.oracle.com/technical-resources/
```

My Oracle Support

セキュリティ・パッチ、動作要件およびサポート・ナレッジ・ベースに関する情報を確認するには、My Oracle Support(旧OracleMetaLink)に接続してください。場所は次のとおりです。

https://support.oracle.com

親トピック: はじめに

表記規則

このドキュメントでは、次のテキスト表記規則が使用されます:

規則	意味
boldface	太字体は、アクションに関連付けられたグラフィカル・ユーザー・インタフェース要素や、本文または用語集で定義されている用語を示します。
italic	イタリック体は、ブック・タイトル、強調、またはユーザーが特定の値を指定するプレースホルダー変数を示します。
`monospace`	等幅体は、段落内のコマンド、URL、サンプル内のコード、画面に表示されるテキスト、またはユーザーが入力するテキストを示します。

親トピック: はじめに

Oracle Databaseセキュリティ・ガイドのこのリリースの変更

この章の内容は次のとおりです。

Oracle Database Security 19cでの変更点

Oracle Database19cのOracle Databaseセキュリティ・ガイドには、新しいセキュリティ機能が記載されています。

- LOBロケータの署名ベース・セキュリティ
  このリリース以降、ラージ・オブジェクト(LOB)ロケータの署名ベース・セキュリティを構成できます。
- デフォルト・ユーザー・アカウントがスキーマ限定になりました
  Oracle Databaseリリース18cからスキーマ限定アカウント機能を使用すると、ほとんどのOracle Database提供スキーマ(users)で、ユーザーがこれらのアカウントに対して認証できないようにそのパスワードが削除されるようになりました。
- 権限分析ドキュメントのOracle Databaseセキュリティ・ガイドへの移動
  権限分析のドキュメントは、『Oracle Database Vault管理者ガイド』から『*Oracle Database*セキュリティ・ガイド』に移動しました。
- スキーマ限定アカウントに対して管理権限を付与または取り消す機能
  SYSOPERやSYSBACKUPなどの管理権限をスキーマ限定(パスワードなし)アカウントに付与できるようになりました。
- SASLおよびSASL以外の両方のActive Directory接続の自動サポート
  このリリース以降、Microsoft Active Directory接続では、Simple Authentication and Security Layer (SASL)およびTransport Layer Security (TLS)バインドの両方がサポートされています。
- 異なるユーザーに対するOracleネイティブ暗号化とTLS認証の同時サポート
  以前のリリースのOracle Databaseでは、Oracleネイティブ暗号化(Advanced Networking Option (ANO)暗号化とも呼ばれる)とTransport Layer Security (TLS)認証を併用できませんでした。
- サーバー証明書の照合におけるホスト名ベースの部分DN一致のサポート
  部分DN一致に対するこの新しいサポートでは、クライアントがサーバー証明書をさらに検証する機能が追加されます。
- トップレベルのSQL文のみを監査する機能
  統合監査のトップレベルの文機能を使用すると、データベース内のトップ・レベル・ユーザー(または直接ユーザー)のアクティビティを監査できますが、間接ユーザー・アクティビティ監査データは収集されません。
- 統合監査証跡の読取りパフォーマンスの改善
  統合監査証跡レコードを格納するAUDSYS.AUD$UNIFIEDシステム表は、読取りパフォーマンスを向上するためにパーティション・プルーニングを使用して再設計されました。
- 共通統合監査ポリシーのSYSLOG宛先
  Oracle Databaseリリース19.9で使用でき、共通統合監査ポリシーからの統合監査レコードの特定の事前定義済列をUNIX SYSLOG宛先に書き込むことができます。
- SYSLOGおよびWindowsイベントビューア用の監査レコード・フィールド名としてのPDB_GUID
  SYSLOGおよびWindowsイベント・ビューアの監査レコード・フィールドに、統合監査証跡レコードに関連付けられたプラガブル・データベースを識別するために新しいフィールドPDB_GUIDが追加されました。

親トピック: このリリースでのOracle Databaseセキュリティ・ガイドの変更

LOBロケータの署名ベース・セキュリティ

このリリース以降、ラージ・オブジェクト(LOB)ロケータの署名ベース・セキュリティを構成できます。

この機能は、特にLOBデータ型のインスタンス(CLOBおよびBLOB)が分散環境で使用される場合に、Oracle Database LOBのセキュリティを強化します。

LOB署名キーは、マルチテナントPDBまたはスタンドアロンの非マルチテナント・データベースのどちらでも使用できます。`ALTER DATABASE DICTIONARY ENCRYPT CREDENTIALS` SQL文を実行することで、LOB署名キー資格証明の暗号化を有効にできます。それ以外の場合、資格証明は不明瞭化された形式で格納されます。LOB署名キーを暗号化された形式で格納する場合、データベースまたはPDBにオープンTDEキーストアが必要です。

関連トピック

- LOBロケータの署名を使用したLOBの保護

親トピック: Oracle Database Security 19cでの変更点

デフォルトのユーザー・アカウントがスキーマ限定になりました

Oracle Databaseリリース18cからスキーマ限定アカウント機能を使用すると、ほとんどのOracle Database提供スキーマ(users)で、ユーザーがこれらのアカウントに対して認証できないようにそのパスワードが削除されるようになりました。

この拡張機能はサンプル・スキーマには影響しません。サンプル・スキーマは、引き続きデフォルトのパスワードとともにインストールされます。

スキーマ限定であるデフォルト・スキーマの場合、管理者は、スキーマに対して認証する必要がある場合でもこれらのアカウントを引き続きパスワードで変更できますが、後でスキーマをスキーマ限定アカウントに戻すことをお薦めします。

この機能の利点は、管理者がこれらのOracle Database提供のスキーマのパスワードを定期的にローテーションする必要がなくなったことです。この機能により、デフォルトのパスワードを使用してこれらのアカウントにハッキングする攻撃者のセキュリティ・リスクも削減されます。

関連トピック

- Oracle Databaseから提供される事前定義済のスキーマ・ユーザー・アカウント
- スキーマ限定アカウント

親トピック: Oracle Database Security 19cでの変更点

権限分析ドキュメントのOracle Databaseセキュリティ・ガイドへの移動

権限分析のドキュメントは、『Oracle Database Vault管理者ガイド』から『*Oracle Database*セキュリティ・ガイド』に移動しました。

権限分析のライセンス情報については、『*Oracle Database*ライセンス情報ユーザー・マニュアル』を参照してください。

関連トピック

- 権限分析の実行による権限使用の特定
- Oracle Databaseライセンス情報ユーザー・マニュアル

親トピック: Oracle Database Security 19cでの変更点

スキーマ限定アカウントに対して管理権限を付与または取り消す機能

SYSOPERやSYSBACKUPなどの管理権限をスキーマ限定(パスワードなし)アカウントに付与できるようになりました。

現在管理権限を付与されている既存のユーザー・アカウント(アクティブ、ほとんどアクセスされていない、使用されていないユーザー)は、スキーマ限定アカウントになるように変更できます。この拡張により、管理者はこれらのアカウントのパスワードを管理する必要がなくなります。

関連トピック

- スキーマ限定アカウントについて

親トピック: Oracle Database Security 19cでの変更点

SASLおよびSASL以外の両方のActive Directory接続の自動サポート

このリリース以降、Microsoft Active Directory接続では、Simple Authentication and Security Layer (SASL)およびTransport Layer Security (TLS)バインドの両方がサポートされています。

集中管理されたユーザーの場合、Oracleデータベースは最初にSASLバインドを使用してActive Directoryに接続しようとします。Active DirectoryサーバーがSASLバインド接続を拒否した場合、OracleデータベースはSASLバインドなしで接続を自動的に再試行しますが、引き続きTLSで保護されます。

Active Directory管理者はActive Directoryサーバーの接続パラメータを構成しますが、この新しいActive Directory接続拡張に一致するようにデータベースを構成する必要はありません。データベースは、SASLを使用するからSASLバインドを使用しないに、自動的に調整されます。

関連トピック

- Oracle DatabaseとMicrosoft Active Directoryの接続の構成について

親トピック: Oracle Database Security 19cでの変更点

異なるユーザーに対するOracleネイティブ暗号化とTLS認証の同時サポート

以前のリリースのOracle Databaseでは、Oracleネイティブ暗号化(Advanced Networking Option (ANO)暗号化とも呼ばれる)とTransport Layer Security (TLS)認証を併用できませんでした。

このリリース以降、新しいパラメータ`SQLNET.IGNORE_ANO_ENCRYPTION_FOR_TCPS`を`TRUE`に設定して、TCPSクライアントの使用とこれら2つのパラメータのいずれかが`required`に設定されていることに競合がある場合、`SQLNET.ENCRYPTION_CLIENT`または`SQLNET.ENCRYPTION_SERVER`を無視できます。

関連トピック

- 異なるユーザーに対するOracleネイティブ暗号化とSSL認証の両方の同時有効化

親トピック: Oracle Database Security 19cでの変更点

サーバー証明書の照合におけるホスト名ベースの部分DN一致のサポート

部分DN一致に対するこの新しいサポートでは、クライアントがサーバー証明書をさらに検証する機能が追加されます。

Transport Layer Security (TLS)ハンドシェイク中に、サーバー証明書との完全DN一致を実行する以前の機能もサポートされています。クライアントは完全DN一致と部分DN一致の両方をサポートします。サーバーDN一致を有効にすると、部分DN一致がデフォルトになります。

証明書の検証に部分および完全DN一致を許可すると、証明書の作成方法に基づいて柔軟性が向上します。

関連トピック

- サーバーDN一致の構成とクライアントでのTLS付きTCP/IPの使用について

親トピック: Oracle Database Security 19cでの変更点

トップレベルのSQL文のみを監査する機能

統合監査のトップレベルの文機能を使用すると、データベース内のトップ・レベル・ユーザー(または直接ユーザー)のアクティビティを監査できますが、間接ユーザー・アクティビティ監査データは収集されません。

この機能を使用すると、トップレベルのユーザーが直接発行したイベントのみを監査できます。間接SQL文のオーバーヘッドはありません。 トップレベルの文は、ユーザーが直接発行するSQL文です。これらの文は、セキュリティとコンプライアンスの両方にとって重要になる可能性があります。 PL/SQLプロシージャ内またはファンクション内から実行されるSQL文は、トップレベルとはみなされないため、監査目的にはあまり関係がない可能性があります。

関連トピック

- トップレベルの文のみの監査

親トピック: Oracle Database Security 19cでの変更点

統合監査証跡の読取りパフォーマンスの改善

統合監査証跡レコードを格納するAUDSYS.AUD$UNIFIEDシステム表は、読取りパフォーマンスを向上するためにパーティション・プルーニングを使用して再設計されました。

この再設計により、AUDSYS.AUD$UNIFIED表への新しい列の追加が必要になります。AUDSYS.AUD$UNIFIED表監査レコードの問合せを可能にするUNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューでは、新しいAUDSYS.AUD$UNIFIED表列に対応するEVENT_TIMESTAMP_UTC列が追加されました。この拡張に伴い、GV$UNIFIED_AUDIT_TRAILビューのEVENT_TIMESTAMP列のデータ型はTIMESTAMP(6)に変更されました。

UNIFIED_AUDIT_TRAILビューを問い合せる場合は、パーティション化プルーニングを実現するためにWHERE句にEVENT_TIMESTAMP_UTC列を含めることをお薦めします。

関連トピック

- UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューのためのベスト・プラクティス

親トピック: Oracle Database Security 19cでの変更点

共通統合監査ポリシーのSYSLOG宛先

Oracle Databaseリリース19.9で使用でき、共通統合監査ポリシーからの統合監査レコードの特定の事前定義済列をUNIX SYSLOG宛先に書き込むことができます。

この機能を有効にするには、新しいCDBレベルのinit.oraパラメータであるUNIFIED_AUDIT_COMMON_SYSTEMLOGを設定します。この拡張によって、共通統合監査ポリシーのすべての監査レコードを1つの宛先に統合できます。

この機能は、UNIXプラットフォームのみで使用できます。Windowsでは使用できません。

関連トピック

- SYSLOGおよびWindowsイベントビューアでの統合監査証跡の取得の有効化

親トピック: Oracle Database Security 19cでの変更点

SYSLOGおよびWindowsイベントビューアの監査レコード・フィールドとしてのPDB_GUID

SYSLOGおよびWindowsイベントビューアの監査レコード・フィールドに、統合監査証跡レコードに関連付けられたプラガブル・データベースを識別するために新しいフィールドPDB_GUIDが追加されました。

マルチテナント・データベース・デプロイメントでは、統合監査証跡レコードを生成したプラガブル・データベースが監査証跡で識別される必要があります。このリリース以降、SYSLOGおよびWindowsイベントビューアには、この情報を取得するために新しいフィールドPDB_GUIDがあります。データ型はVARCHAR2です。

関連トピック

- SYSLOGまたはWindowsイベントビューアへの統合監査証跡レコードの書込みについて

親トピック: Oracle Database Security 19cでの変更点

Oracle Database Security 19cの更新

Oracle Databaseリリース19cには、リリース19cの最終更新からの更新がいくつかあります。

- Oracle Autonomous Database on Dedicated Exadata Infrastructureに対するIAMユーザーの認証と認可
  Oracle Databaseリリース19.14より、Oracle Autonomous Database on Dedicated Exadata Infrastructureに対してIAMユーザーを認証および認可できるようになりました。
- Identity and Access ManagementとOracle Database環境との統合の拡張機能
  Oracle Databaseリリース19.16では、Identity and Access Management (IAM)ユーザーとOracle Database環境との統合の拡張機能を使用できます。
- Oracle Autonomous Cloud DatabaseとIdentity and Access Managementの統合
  Oracle Databaseリリース19.13で使用でき、Identity and Access Management (IAM)ユーザーは、パスワードまたはトークンベースの認証のいずれかを使用して、Oracle Autonomous Database on Shared Exadata Infrastructureにログインできます。
- データベース統合でのアイデンティティ・ドメインを含むIdentity and Access Managementの非デフォルト・ドメインのサポート
  このリリース以降、Oracle Databaseでは、アイデンティティ・ドメインを含むIdentity and Access Management (IAM)でのテナンシ内の非デフォルト・ドメインがサポートされています。
- Microsoft Azure Active Directoryとオンプレミスを含むその他のOracle Database環境との統合
  Oracle Databaseリリース19.16で使用でき、Microsoft Azure Active Directory (Azure AD)ユーザーは、Azure AD OAuth2アクセス・トークンを使用してその他のOracle Database環境にログインできます。
- Oracle Cloud Infrastructure Autonomous DatabaseとのMicrosoft Azure Active Directory統合
  2022年6月にOracle Autonomous Databaseで利用可能となり、Microsoft Azure Active Directory (Azure AD)ユーザーは、Azure AD OAuth2アクセス・トークンを使用してOracle Cloud Infrastructure (OCI) Autonomous Databaseにログインできます。
- アプリケーションの段階的データベース・パスワード・ロールオーバー
  Oracle Databaseリリース19.12で使用でき、管理者が停止時間をスケジュールしなくても、アプリケーションのデータベース・パスワードを変更できます。

- 単一のデータベース・クライアントで複数のKerberosプリンシパルを使用する機能
  Oracle Databaseリリース19.10で使用でき、Oracle DatabaseクライアントのKerberos認証を構成するときに、単一のOracle Databaseクライアントで複数のKerberosプリンシパルを指定できます。
- FIPS 140.2用のMicro Edition Suite (MES)のサポートの更新
  Oracle Databaseリリース19.10で使用でき、Oracle DatabaseではFIPS 140.2用のMicro Edition Suite (MES)バージョン4.5がサポートされます。
- DBMS_CRYPTO非対称キー操作のサポート
  Oracle Databaseリリース19.9で使用でき、DBMS_CRYPTO PL/SQLパッケージでは、対称キー操作に対する既存のサポートに加えて、非対称キー操作がサポートされます。
- 共通統合監査ポリシーのSYSLOG宛先
  Oracle Databaseリリース19.9で使用でき、共通統合監査ポリシーからの統合監査レコードの特定の事前定義済列をUNIX SYSLOG宛先に書き込むことができます。
- ネイティブ暗号化のセキュリティ更新
  Oracleでは、Oracle Databaseリリース11.2以降における、ネイティブ・ネットワーク暗号化環境に影響を与える必要なセキュリティ機能強化に対応するために、ダウンロード可能なパッチを提供しています。
- クライアント・ウォレットを使用しないTransport Layer Security接続を構成する機能
  Oracle Databaseリリース19.14で使用でき、Oracle Databaseクライアントを必要とせずに、ローカル・システムの他の場所でCAルート証明書が使用可能な場合に、その既知の証明書を保持するウォレットを提供します。

親トピック: このリリースでのOracle Databaseセキュリティ・ガイドの変更

Oracle Autonomous Database on Dedicated Exadata Infrastructureに対するIAMユーザーの認証と認可

Oracle Databaseリリース19.14より、Oracle Autonomous Database on Dedicated Exadata Infrastructureに対してIAMユーザーを認証および認可できるようになりました。

追加の機能強化は次のとおりです。

- アプリケーションは、エンド・ユーザー、インスタンスおよびリソース・プリンシパルを使用してAutonomous Databaseインスタンスに接続できるようになりました。
- IAMユーザーは、データベース・ユーザー・スキーマを使用してAutonomous Databaseにプロキシできるようになりました。
- データベース・リンクがIAM接続でサポートされます。

関連トピック

- Oracle DBaaSデータベースに対するIAMユーザーの認証と認可

親トピック: Oracle Database Security 19cの更新

Identity and Access ManagementとOracle Database環境との統合の拡張機能

Oracle Databaseリリース19.16では、Identity and Access Management (IAM)ユーザーとOracle Database環境との統合の拡張機能を使用できます。

- その他のOracle Database環境:サポートされているOracle Database環境のリストは、次のとおりです。
  - Oracle Autonomous Database on Dedicated Exadata Infrastructure

    - Oracle Autonomous Database on Shared Exadata Infrastructure
    - Oracle Base Database Service
- IAMのユーザー名とパスワードを使用してIAMトークンを取得する機能: IAMのユーザー名とパスワードまたは安全性の高い外部パスワード・ストア(SEPS)を使用してトークンを取得する方が、データベース・アクセスのパスワード・ベリファイアを使用するよりも安全です。

関連トピック

- Oracle DBaaSデータベースに対するIAMユーザーの認証と認可

親トピック: Oracle Database Security 19cの更新

Oracle Autonomous Cloud DatabaseとIdentity and Access Managementの統合

Oracle Databaseリリース19.13で使用でき、Identity and Access Management (IAM)ユーザーは、パスワードまたはトークンベースの認証のいずれかを使用して、Oracle Autonomous Database on Shared Exadata Infrastructureにログインできます。

IAMのADMINユーザーは、IAMユーザーおよびIAMグループの認証と認可の両方を構成できます。IAMユーザーは、SQL*PlusやSQLclなどのツールを使用して、Oracle Autonomous Databaseにログインできます。

この拡張により、IAMとOracle Databaseの両方のセキュリティ上のメリットが得られます。たとえば、この構成でOracle Databaseの段階的なパスワード・ロールオーバー機能を使用し、停止時間なしでアプリケーション・パスワードを更新できます。

関連トピック

- Oracle DBaaSデータベースに対するIAMユーザーの認証と認可

親トピック: Oracle Database Security 19cの更新

データベース統合でのアイデンティティ・ドメインを含むIdentity and Access Managementの非デフォルト・ドメインのサポート

このリリース以降、Oracle Databaseでは、アイデンティティ・ドメインを含むIdentity and Access Management (IAM)でのテナンシ内の非デフォルト・ドメインがサポートされています。

次のリリースがサポートされています。

- Oracle Autonomous Database on Shared Exadata Infrastructure
- Oracle Autonomous Database on Dedicated Exadata Infrastructure

この更新により、デフォルト以外のIAMドメイン内のIAMユーザーが、IAMデータベース・パスワード・ベリファイアまたはIAMアクセス・トークンを使用してデータベースにアクセスできつようになります。デフォルト・ドメイン内のIAMユーザーはすでにサポートされています。

以前のリリースでは、IAM統合は、デフォルト・ドメインからのユーザーおよびグループの場合のみ機能し、デフォルト以外のカスタム・ドメインからのユーザーおよびグループはサポートしていませんでした。

関連トピック

- Oracle DBaaSデータベースに対するIAMユーザーの認証と認可

親トピック: Oracle Database Security 19cの更新

Microsoft Azure Active Directoryとオンプレミスを含むその他のOracle Database環境との統合

Oracle Databaseリリース19.16で使用でき、Microsoft Azure Active Directory (Azure AD)ユーザーは、Azure AD OAuth2アクセス・トークンを使用してその他のOracle Database環境にログインできます。

以前のリリースでは、Oracle Cloud Infrastructure (OCI) Autonomous DatabaseのAzure AD統合がサポートされていました。このリリースでは、Azure AD統合サポートがオンプレミスOracle Databaseリリース19.16以降(Oracle Database 21cを除く)に拡張されました。

Azure AD OAuth2トークンを使用してデータベースにアクセスできます。Azure ADユーザーはAzure ADトークンを使用してデータベースに直接アクセスでき、アプリケーションはサービス・トークンを使用してデータベースにアクセスできます。

関連トピック

- Oracle DatabaseのMicrosoft Azure Active Directoryユーザーの認証および認可

親トピック: Oracle Database Security 19cの更新

Oracle Cloud Infrastructure Autonomous DatabaseとのMicrosoft Azure Active Directory統合

2022年6月にOracle Autonomous Databaseで利用可能となり、Microsoft Azure Active Directory (Azure AD)ユーザーは、Azure AD OAuth2アクセス・トークンを使用してOracle Cloud Infrastructure (OCI) Autonomous Databaseにログインできます。

OCI Oracle Autonomous Databaseは、Azure AD OAuth2トークンを受け入れてデータベースにアクセスできるようになりました。Azure ADユーザーはAzure ADトークンを使用してデータベースに直接アクセスでき、アプリケーションはサービス・トークンを使用してデータベースにアクセスできます。

Azure AD OAuth2トークンを使用してデータベースにアクセスできます。Azure ADユーザーはAzure ADトークンを使用してデータベースに直接アクセスでき、アプリケーションはサービス・トークンを使用してデータベースにアクセスできます。

関連トピック

- Oracle DatabaseのMicrosoft Azure Active Directoryユーザーの認証および認可

親トピック: Oracle Database Security 19cの更新

アプリケーションの段階的データベース・パスワード・ロールオーバー

Oracle Databaseリリース19.12で使用でき、管理者が停止時間をスケジュールしなくても、アプリケーションのデータベース・パスワードを変更できます。

これを行うために、データベース管理者は、このリリースで新しく追加されたPASSWORD_ROLLOVER_TIMEパスワード・プロファイル・パラメータにゼロ以外の制限を持つプロファイルをアプリケーション・スキーマに関連付けることができます。これにより、PASSWORD_ROLLOVER_TIME制限で指定された期間、古いパスワードを有効なままにしながら、アプリケーション・ユーザーのデータベース・パスワードを変更できます。ロールオーバー期間中、アプリケーション・インスタンスは古いパスワードまたは新しいパスワードのいずれかを使用してデータベース・サーバーに接続できます。ロールオーバー時間が経過すると、新しいパスワードのみが許可されます。

この拡張の前は、管理者は通常、アプリケーション・データベースのパスワードのローテーション中にアプリケーションを停止していま

した。これは、パスワードの更新にデータベース側とアプリケーション側の両方での変更が必要なためです。段階的なデータベース・パスワード・ロールオーバーの拡張により、アプリケーションで新しいパスワードが構成されるまで、アプリケーションは古いパスワードを引き続き使用できます。

CREATE PROFILEおよびALTER PROFILE文の新しい句PASSWORD_ROLLOVER_TIMEに加えて、ALTER USER文には新しい句EXPIRE PASSWORD ROLLOVER PERIODがあります。DBA_USERSおよびUSER_USERSデータ・ディクショナリ・ビューのACCOUNT_STATUS列には、ロールオーバー・ステータスを示す値を示す新しいステータスがいくつかあります。

関連トピック

- アプリケーションの段階的データベース・パスワード・ロールオーバーの管理

親トピック: Oracle Database Security 19cの更新

単一のデータベース・クライアントで複数のKerberosプリンシパルを使用する機能

Oracle Databaseリリース19.10で使用でき、Oracle DatabaseクライアントのKerberos認証を構成するときに、単一のOracle Databaseクライアントで複数のKerberosプリンシパルを指定できます。

この機能を有効にするには、クライアントのユーザーごとに個別の資格証明キャッシュを作成してから、接続文字列を使用してユーザーを指定する必要があります。

以前のリリースでは、Oracle Databaseクライアントごとに1つのKerberosプリンシパルに制限されていました。

関連トピック

- ステップ1C: tnsnames.oraを使用した追加のKerberosプリンシパルの指定(オプション)
- 『Oracle Database Net Servicesリファレンス・ガイド』

親トピック: Oracle Database Security 19cの更新

FIPS 140.2用のMicro Edition Suite (MES)のサポートの更新

Oracle Databaseリリース19.10で使用でき、Oracle DatabaseではFIPS 140.2用のMicro Edition Suite (MES)バージョン4.5がサポートされます。

Micro Edition Suite (MES)バージョン4.5の更新には、RSA BSAFE MESライブラリ内の4つの新しいCVE、FIPS 140.2で必要なルールのサポート、Crypto Foundationからの更新されたNZ/ZTライブラリへのアクセスが含まれます。

この拡張により、Oracle Database FIPS 140.2構成では、最新のRSA BSAFE MESライブラリの新機能およびセキュリティの向上を活用できます。

関連トピック

- 透過的データ暗号化およびDBMS_CRYPTO用のFIPS 140-2の構成

親トピック: Oracle Database Security 19cの更新

DBMS_CRYPTO非対称キー操作のサポート

Oracle Databaseリリース19.9で使用でき、DBMS_CRYPTO PL/SQLパッケージでは、対称キー操作に対する既存のサポートに加えて、非対称キー操作がサポートされます。

非対称キー操作のサポートを実装するために、次のプロシージャがDBMS_CRYPTOパッケージに追加されています。

- PKENCRYPT

- PKDECRYPT
- SIGN
- VERIFY

関連トピック

- DBMS_CRYPTOパッケージを使用した非対称キー操作
- Oracle Database PL/SQLパッケージ・プロシージャおよびタイプ・リファレンス

親トピック: Oracle Database Security 19cの更新

共通統合監査ポリシーのSYSLOG宛先

Oracle Databaseリリース19.9で使用でき、共通統合監査ポリシーからの統合監査レコードの特定の事前定義済列をUNIX SYSLOG宛先に書き込むことができます。

この機能を有効にするには、新しいCDBレベルの`init.ora`パラメータである`UNIFIED_AUDIT_COMMON_SYSTEMLOG`を設定します。この拡張によって、共通統合監査ポリシーのすべての監査レコードを1つの宛先に統合できます。

この機能は、UNIXプラットフォームのみで使用できます。Windowsでは使用できません。

関連トピック

- SYSLOGおよびWindowsイベントビューアでの統合監査証跡の取得の有効化

親トピック: Oracle Database Security 19cの更新

ネイティブ暗号化のセキュリティ更新

Oracleでは、Oracle Databaseリリース11.2以降における、ネイティブ・ネットワーク暗号化環境に影響を与える必要なセキュリティ機能強化に対応するために、ダウンロード可能なパッチを提供しています。

このパッチは、My Oracle Supportノート2118136.2で入手できます。

改善されたサポート対象アルゴリズムは次のとおりです。

- 暗号化アルゴリズム: AES128、AES192およびAES256
- チェックサム・アルゴリズム: SHA1、SHA256、SHA384およびSHA512

非推奨であり、使用をお薦めしないアルゴリズムは次のとおりです。

- 暗号化アルゴリズム: DES、DES40、3DES112、3DES168、RC4_40、RC4_56、RC4_128およびRC4_256
- チェックサム・アルゴリズム: MD5

サイトでネットワーク・ネイティブ暗号化を使用する必要がある場合は、My Oracle Supportノート2118136.2で説明されているパッチをダウンロードする必要があります。Oracle Databaseインストールの円滑な移行を可能にするために、このパッチでは、脆弱なアルゴリズムを無効にしより強力なアルゴリズムの使用を開始できる、2つのパラメータが提供されます。このパッチは、サーバー上とクライアント上の両方のOracle Databaseインストールにインストールする必要があります。

ネットワーク・ネイティブの暗号化の代替となるのは、TLS (Transport Layer Security)であり、中間者攻撃からの保護を実現します。

関連トピック

- ネイティブ・ネットワーク暗号化とTransport Layer Securityの間の選択
- ネイティブ・ネットワーク暗号化のセキュリティの向上

親トピック: Oracle Database Security 19cの更新

クライアント・ウォレットを使用しないTransport Layer Security接続を構成する機能

Oracle Databaseリリース19.14で使用でき、Oracle Databaseクライアントを必要とせずに、ローカル・システムの他の場所でCAルート証明書が使用可能な場合に、その既知の証明書を保持するウォレットを提供します。

Transport Layer Security (TLS)暗号化には、一方向認証または双方向認証のいずれかが必要です。HTTPS接続に一般的に使用される一方向認証(デフォルト)では、ローカル・システムですでに使用可能な既知のルートCA証明書を使用して、サーバー証明書が検証されます。このリリースから、ルート証明書がすでにローカル・システムで使用可能である場合は、その既知の証明書を保持するウォレットをインストールして構成する必要がなくなります。

この拡張により、Oracle Databaseクライアントのインストールと、Oracle Databaseクライアント/サーバー通信を暗号化するTLSプロトコルの使用が大幅に簡略化されます。

関連トピック

- クライアント・ウォレットを使用しないTransport Layer Security接続

親トピック: Oracle Database Security 19cの更新

1 Oracle Databaseセキュリティの概要

Oracle Databaseには、デフォルトのセキュリティ機能が豊富に用意されています。これらを使用して、ユーザー・アカウント、認証、権限、アプリケーション・セキュリティ、暗号化、ネットワーク・トラフィックおよび監査を管理できます。

- Oracle Databaseセキュリティについて
  デフォルトのOracle Database機能を使用すると、Oracle Databaseインストールのいくつかの領域でセキュリティを構成できます。
- その他のOracle Databaseセキュリティ製品
  デフォルトのデータベース・インストールで利用可能なセキュリティ・リソースに加え、Oracle Databaseには、他のいくつかのデータベース・セキュリティ製品が用意されています。

1.1 Oracle Databaseセキュリティについて

デフォルトのOracle Database機能を使用すると、Oracle Databaseインストールのいくつかの領域でセキュリティを構成できます。

セキュリティを構成できる領域は次のとおりです。

- ユーザー・アカウント。作成したユーザー・アカウントは様々な方法で保護できます。サイトのパスワード・ポリシーを強化するために、パスワード・プロファイルを作成することもできます。Oracle Databaseユーザーのセキュリティの管理では、ユーザー・アカウントの管理方法について説明します。
- 認証方式。Oracle Databaseには、ユーザーおよびデータベース管理者用の認証を構成する方法がいくつかあります。たとえば、ユーザーは、データベース・レベル、オペレーティング・システムおよびネットワークで認証できます。認証の構成では、Oracle Databaseにおける認証の機能について説明します。Microsoft Active Directoryによる集中管理ユーザーの構成も参照してください。
- 権限とロール。権限とロールを使用すると、データに対するユーザー・アクセスを制限できます。次の章では、権限およびロールを管理する方法について説明します。
  - 権限とロール認可の構成
  - 権限分析の実行による権限使用の特定
  - 定義者権限および実行者権限のセキュリティの管理
  - PL/SQLパッケージおよびタイプでのファイングレイン・アクセスの管理
  - Enterprise Managerによるマルチテナント環境のセキュリティの管理
- アプリケーション・セキュリティ。データベース・アプリケーションを作成する最初のステップは、データベース・アプリケーションが適切に保護されるようにすることです。アプリケーション開発者のセキュリティの管理では、アプリケーション・セキュリティをアプリケーション・セキュリティ・ポリシーに組み込む方法について説明します。
- アプリケーション・コンテキストを使用したユーザー・セッション情報。アプリケーション・コンテキストは、セッション情報を保持する名前と値のペアです。この情報に基づいて、ユーザーの名前や端末などのユーザーに関するセッション情報を取得し、そのユーザーのデータベース・アクセスおよびアプリケーション・アクセスを制限できます。「アプリケーション・コンテキストを使用したユーザー情報の取得」では、アプリケーション・コンテキストの使用方法について説明します。
- 仮想プライベート・データベースを使用した行および列レベルでのデータベース・アクセス。仮想プライベート・データベース・ポリシーは、ユーザーが発行したSQL文にWHERE述語を動的に埋め込みます。Oracle Virtual Private

Databaseを使用したデータ・アクセスの制御では、仮想プライベート・データベース・ポリシーの作成方法と管理方法について説明します。

- 異なるカテゴリのデータの分類および保護。機密データ(クレジット・カードや社会保障番号など)を保持するデータベースのすべての表の列を確認し、このデータを分類して、指定されたクラスのこのデータ全体を保護するポリシーを作成できます。透過的機密データ保護の使用では、透過的機密データ保護ポリシーの作成方法について説明します。
- ネットワーク・データの暗号化。手動によるデータ暗号化では、ネットワーク・データへの不正アクセスを防止するために、DBMS_CRYPTO PL/SQLパッケージを使用して、ネットワーク・データを暗号化する方法について説明します。Oracle Databaseのネイティブ・ネットワーク・データ暗号化とデータ整合性の構成で説明するように、サーバーとクライアントの両方でOracle Net Servicesのネイティブなデータの暗号化と整合性を構成できます。
- シンJDBCクライアント・ネットワーク構成。シンJava Database Connectivity (JDBC)クライアントを構成して、Oracleデータベースに安全に接続できます。シンJDBCクライアント・ネットワークの構成では、詳細情報について示します。
- 厳密認証。データベースを構成して、デジタル証明書を使用するSSLを含む様々なサード・パーティ認証サービスをサポートするOracle認証アダプタを使用した厳密認証を使用できます。Oracle Databaseには、次の厳密認証サポートが用意されています。
    - 集中化された認証とシングル・サインオン。
    - Kerberos
    - Remote Authentication Dial-in User Service(RADIUS)
    - Transport Layer Security (TLS) (旧称: Secure Sockets Layer)

  次の章では、厳密認証を扱います。
    - 厳密認証の概要
    - 厳密認証の管理ツール
    - Kerberos認証の構成
    - Transport Layer Security認証の構成
    - RADIUS認証の構成
    - 厳密認証の使用のカスタマイズ
- データベース・アクティビティの監査。データベース・アクティビティは、すべてのSQL文、SQL権限、スキーマ・オブジェクト、ネットワーク・アクティビティの監査など、一般的な条件で監査できます。または、社内ネットワーク外部のIPアドレスが使用されているような場合は、きめ細かい方法で監査できます。この章では、データベース監査証跡の削除方法についても説明します。次の章では、データベース監査の構成および管理方法について説明します。
    - 監査の概要
    - 監査ポリシーの構成
    - 監査証跡の管理

さらに、Oracle Databaseの安全性の維持では、Oracle Databaseインストールを保護する際に従う必要のあるガイドラインを示します。

親トピック: Oracle Database Securityの概要

1.2 その他のOracle Databaseセキュリティ製品

デフォルトのデータベース・インストールで利用可能なセキュリティ・リソースに加え、Oracle Databaseには、他のいくつかのデータベース・セキュリティ製品が用意されています。

これらの製品は次のとおりです。

- Oracle Advanced Securityでは、透過的データ暗号化およびOracle Data Redactionを使用して機密データを保護できます。
- Oracle Label Securityは、分類ラベルをデータに適用して、行レベルのデータのユーザー・アクセスをフィルタ処理できます。
- Oracle Database Vaultには、権限を持つユーザーからのデータ保護など、機密データに対するファイングレイン・アクセス・コントロールが用意されています。たとえば、給与などの従業員情報へのアクセスをデータベース管理者に制限できます。
- Oracle Data Safeでは、Oracleデータベース内のデータの機密性とリスクを分析し、その結果に基づいて、機密データをマスクするポリシーを作成し、セキュリティ制御を作成およびモニターし、ユーザー・セキュリティを評価し、ユーザー・アクティビティをモニターできます。
- Oracle Enterprise User Securityを使用すると、ユーザー・セキュリティをエンタープライズ・レベルで管理できます。
- Oracle Enterprise Manager Data Masking and Subsetting Pack では、元の機密データを架空のデータに不可逆的に置き換え、本番データをIT開発者またはオフショア・ビジネス・パートナと安全に共有できます。
- Oracle Audit Vault and Database Firewallでは、Oracle Databaseの監査証跡表、データベース・オペレーティング・システム監査ファイル、データベースのREDOログなどのソースからデータベース監査データを収集します。
  Oracle Audit Vault and Database Firewallを使用すると、不審なアクティビティに対するアラートを作成したり、権限を持つユーザーの変更、スキーマの変更およびデータ・レベルのアクセスに関する履歴のレポートを作成できます。
- Oracle Key Vaultを使用すると、暗号化キー、Oracleウォレット、Javaキーストアおよび資格証明ファイルの一元管理によってセキュリティおよび暗号化のデプロイメントを促進できます。これは、Oracleウォレット、JavaキーストアおよびOracle Advanced Security透過的データ暗号化(TDE)マスター・キー用に最適化されています。Oracle Key Vaultでは、OASIS KMIP標準がサポートされています。このフルスタックの、セキュリティが強化されたソフトウェア・アプライアンスは、セキュリティ、可用性およびスケーラビリティのためにOracle LinuxとOracle Databaseのテクノロジを使用しており、互換性のあるハードウェアのうちご希望のものにデプロイできます。

これらの製品に加えて、新製品およびセキュリティ・パッチやアラートに関する重要な情報など、Oracle Databaseセキュリティに関する最新情報を入手するには、Oracle Technology Networkの「Security Technology Center」を参照してください。次の場所でアクセスできます。

http://www.oracle.com/technetwork/topics/security/whatsnew/index.html

親トピック: Oracle Database Securityの概要

第I部 ユーザー認証および認可の管理

第I部では、ユーザー認証および認可の管理方法について説明します。

- Oracle Databaseユーザーのセキュリティの管理
  パスワード作成時に制限を適用するなど、Oracle Databaseユーザーのセキュリティは様々な方法で管理できます。
- 認証の構成
  認証とは、データベースに接続するユーザーや他のエンティティのアイデンティティを検証することです。
- 権限とロール認可の構成
  権限とロールの認可によって、ユーザーが毎日のタスクを実行するために保持する権限が制御されます。
- 権限分析の実行による権限使用の特定
  権限分析では、ユーザーが使用中および未使用の権限とロールが動的に分析されます。
- Microsoft Active Directoryによる集中管理ユーザーの構成
  Oracle Databaseでは、中間ディレクトリまたはOracle Enterprise User Securityを使用せずにデータベースでMicrosoft Active Directoryユーザーを直接認証および認可できます。
- Oracle DBaaSデータベースに対するIAMユーザーの認証と認可
  Oracle Database as a Service (Oracle DBaaS)インスタンスに接続するように、Identity and Access Management (IAM)ユーザーを構成できます。
- Oracle Databaseに対するMicrosoft Azure Active Directoryユーザーの認証および認可
  Oracle Databaseは、Microsoft Azure ADユーザーがシングル・サインオンを使用して接続できるように構成できます。
- 定義者権限および実行者権限のセキュリティの管理
  ユーザー定義プロシージャの実行中に権限へのアクセス制御で実行者権限および定義者権限を使用すると、セキュリティ上のメリットが得られます。
- PL/SQLパッケージおよびタイプでのファイングレイン・アクセスの管理
  Oracle Databaseにはファイングレイン・アクセスのためのPL/SQLパッケージとタイプが用意されており、これにより外部ネットワーク・サービスやウォレットへのアクセスを制御できます。
- Enterprise Managerによるマルチテナント環境のセキュリティの管理
  Oracle Enterprise Managerを使用して、マルチテナント環境の共通およびローカルのユーザーとロールを管理できます。

2 Oracle Databaseユーザーのセキュリティの管理

パスワード作成時に制限を適用するなど、Oracle Databaseユーザーのセキュリティは様々な方法で管理できます。

- ユーザー・セキュリティについて
  厳密なパスワードおよびユーザー用の特別な制限を指定することで、ユーザー・アカウントを保護できます
- ユーザー・アカウントの作成
  プロファイル、デフォルト・ロール、表領域制限など、ユーザー・アカウントに制限を設定できます。
- ユーザー・アカウントの変更
  ALTER USER文で、デフォルトの表領域やプロファイル、ユーザー・パスワードの変更など、ユーザー・アカウントを変更します。
- ユーザー・リソース制限の構成
  リソースの制限により、ユーザーに利用可能なシステム・リソースの量が決まります。
- ユーザー・アカウントの削除
  ユーザーのスキーマにオブジェクトがある場合、そのユーザーがセッション中でなければユーザー・アカウントを削除できます。
- Oracle Databaseから提供される事前定義済のスキーマ・ユーザー・アカウント
  Oracle Databaseのインストール・プロセスでは、事前定義済の管理アカウント、非管理ユーザー・アカウント、およびサンプル・スキーマ・ユーザー・アカウントがデータベースに作成されます。
- データベース・ユーザーおよびプロファイルのデータ・ディクショナリ・ビュー
  Oracle Databaseには、ユーザーとプロファイルの作成に使用した設定の情報を提供する、一連のデータ・ディクショナリ・ビューがあります。

親トピック: ユーザー認証および認可の管理

2.1 ユーザー・セキュリティについて

厳密なパスワードおよびユーザー用の特別な制限を指定することで、ユーザー・アカウントを保護できます。

各Oracleデータベースには、有効なデータベース・ユーザーのリストがあります。データベースにアクセスするには、ユーザーは、データベース・アプリケーションを実行し、データベースに定義されている有効なユーザー名を使用して、データベース・インスタンスに接続する必要があります。

ユーザー・アカウントの作成時に、ユーザー・アカウントに対して制限を指定できます。ユーザーのセキュリティ・ドメインの一部として、各ユーザーが使用できる各種のシステム・リソースの容量に制限を設定することもできます。Oracle Databaseには、リソースやセッションなどの情報を検索する際に問い合せることができる一連のデータベース・ビューが用意されています。この章では、プロファイルについても説明します。プロファイルとは、ユーザーに適用される属性の集合です。それらの属性を共有する複数ユーザーの中の任意のユーザーに関する単一の参照になります。

Oracle Databaseには、事前定義された管理ユーザー、非管理ユーザーおよびサンプル・スキーマのアカウントのセットが提供されています。これらのアカウントのリストは、Oracle Databaseのインストレーション・ガイドを参照してください。これらのアカウントのステータスを検索するには、DBA_USERSデータ・ディクショナリ・ビューのUSERNAMEおよびACCOUNT_STATUS列を問い合せます。

関連トピック

- 権限とロール認可の構成

親トピック: Oracle Databaseユーザーのセキュリティの管理

2.2 ユーザー・アカウントの作成

プロファイル、デフォルト・ロール、表領域制限など、ユーザー・アカウントに制限を設定できます。

- 共通ユーザーおよびローカル・ユーザーについて
  マルチテナント環境では、CDB共通ユーザーとアプリケーション共通ユーザーはそれぞれのコンテナに対するアクセス権を持ち、ローカル・ユーザーはPDB固有のユーザーです。
- ユーザー・アカウントの作成者とは
  CREATE USERシステム権限を付与されているユーザーは、ユーザー・アカウント(プロキシ・ユーザーとして使用されるユーザー・アカウントを含む)を作成できます。
- 最小限のデータベース権限を持つ新しいユーザー・アカウントの作成
  新しいユーザー・アカウントを作成する場合、このユーザーがデータベースにアクセスできるようにする必要があります。
- 新しいアカウントのユーザー名の作成に関する制限事項
  ユーザー・アカウントの名前を指定する場合は、ネーミング規則や名前が一意かどうかなどの制限事項に注意してください。
- ユーザーへのパスワードの割当て
  CREATE USER文のIDENTIFIED BY句で、ユーザーにパスワードを割り当てます。
- ユーザーのデフォルト表領域
  デフォルト表領域は、ユーザーが作成するオブジェクトを格納します。
- ユーザーへの表領域の割当て制限
  表領域の割当て制限により、ユーザーの表領域に提供される領域の量が決まります。
- ユーザーの一時表領域
  一時表領域には、ユーザー・セッションの存続期間中のみ保持される一時データが含まれています。
- ユーザーのプロファイル
  プロファイルとは、属性によって定義される、データベース・リソースとそのデータベースへのパスワード・アクセスに関する一連の制限です。
- 共通ユーザーまたはローカル・ユーザーの作成
  CREATE USER SQL文を使用して、共通(CDBおよびアプリケーション)ユーザーとローカル・ユーザーの両方を作成できます。
- ユーザーのデフォルト・ロールの作成
  デフォルト・ロールは、ユーザーがセッションを作成したときに、自動的にそのユーザーに対して使用可能になります。

親トピック: Oracle Databaseユーザーのセキュリティの管理

2.2.1 共通ユーザーおよびローカル・ユーザーについて

マルチテナント環境では、CDB共通ユーザーとアプリケーション共通ユーザーはそれぞれのコンテナに対するアクセス権を持ち、ローカル・ユーザーはPDB固有のユーザーです。

- 共通ユーザーについて
  共通ユーザーには、CDB共通ユーザーとアプリケーション共通ユーザーの2つのタイプがあります。
- PDBへの接続によるCDB共通ユーザーへの影響
  CDB以外をPDBとしてCDBに差し込むと、Oracle付属の管理アカウント、ユーザー作成アカウントおよびそれらの権限に影響があります。
- ローカル・ユーザーについて
  マルチテナント環境では、ローカル・ユーザーとは、単一のPDBにのみ存在するデータベース・ユーザーです。

2.2.1.1 共通ユーザーについて

共通ユーザーには、CDB共通ユーザーとアプリケーション共通ユーザーの2つのタイプがあります。

CDB共通ユーザーとは、CDBルートならびに既存および将来のプラガブル・データベース(PDB)によって単一のIDとパスワードが認識されているデータベース・ユーザーです。Oracle提供のSYSおよびSYSTEMなどのすべての管理ユーザー・アカウントはCDB共通ユーザーであり、システム・コンテナを操作できます。CDB共通ユーザーは、異なるPDBで異なる権限を持つことができます。たとえば、ユーザーSYSTEMは、PDBを切り替えて、現在のPDBのSYSTEMに付与されている権限を使用できます。ただし、PDBのいずれかでOracle Database Vaultが有効な場合、SYSTEMがこのPDBに接続している間は、SYSTEMによるユーザー・アカウントの作成が許可されないといったDatabase Vaultの制限が同ユーザーに適用されます。Oracle提供のCDB共通ユーザーの権限を変更することはお薦めしません。

CDB共通ユーザーは、そのユーザーに適切な権限が付与されている場合にかぎり、アプリケーション共通ユーザーが実行できるすべてのタスクを実行できます。

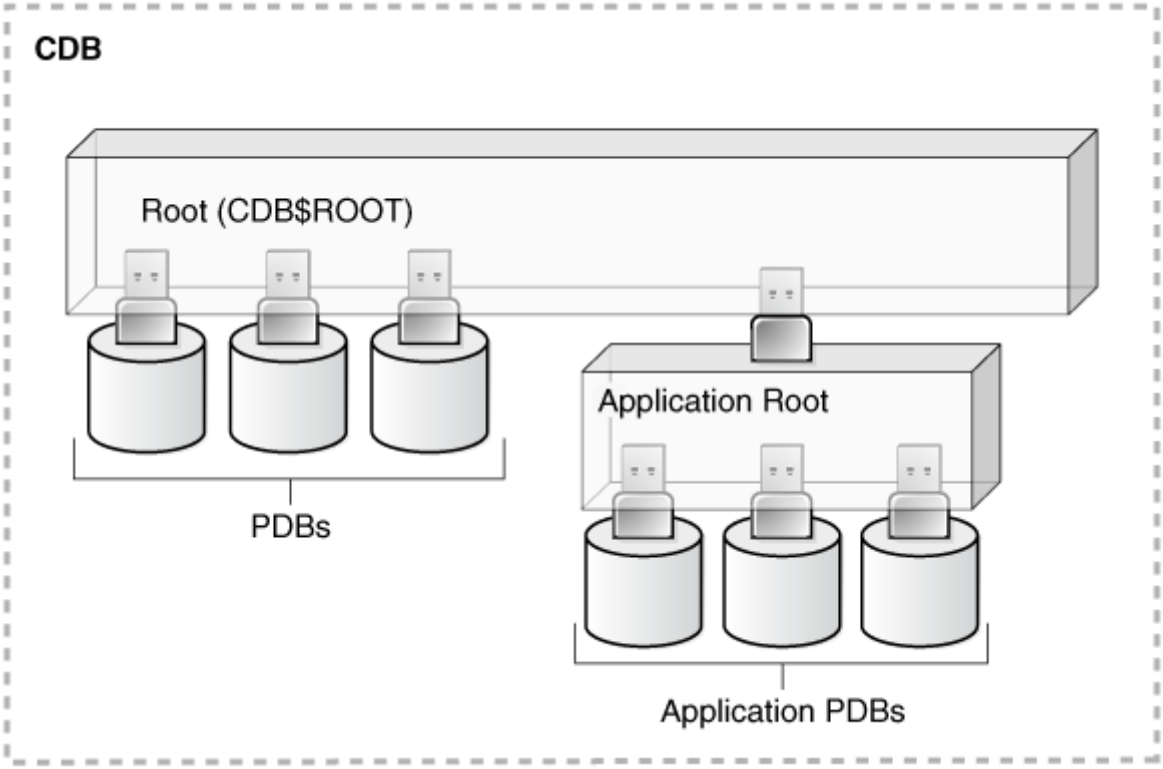
アプリケーション共通ユーザーはアプリケーション・ルートで作成されるユーザー・アカウントで、このアプリケーション・コンテナ内のみで共通です。つまり、アプリケーション共通ユーザーは、CDB共通ユーザーのようにCDB環境全体にアクセスすることはできません。アプリケーション共通ユーザーは、アプリケーションPDBの作成(接続を含む)、オープン、クローズ、切断および削除などのアクティビティに対する責任を負います。このユーザーは、アプリケーション・ルートでアプリケーション共通オブジェクトを作成できます。アプリケーション共通ユーザーを作成できるのは、アプリケーション・ルートに接続しているときに限定されます。ユーザーがアプリケーション共通オブジェクトにアクセスする機能は、ローカルおよびCDB共通オブジェクトと同じ権限に依存します。たとえば、アプリケーション・ルートに関連付けられたPDBのローカル・ユーザーは、そのユーザーが権限を持つPDB内のオブジェクトにのみアクセスできます。アプリケーション・ルート自体では、アプリケーション・コンテナ間で適用される、CDB共通オブジェクトに対する権限を共通に付与できます。

いずれのタイプの共通ユーザーも、それぞれのルート内の共通オブジェクトを管理する責任を負います。CDB共通ユーザーまたはアプリケーション共通ユーザーに適切な権限がある場合、ローカル・ユーザーへの権限の付与など、PDBでの操作も実行できます。これらのユーザーは、共通ユーザーに対してコンテナごとの異なる権限をローカルに付与することもできます。

CDBおよびアプリケーション共通ユーザーは、次のアクティビティを実行できます。

- 共通ユーザーまたは共通ロールへの権限の付与。つまり、CDB共通ユーザーは共通ユーザーまたは共通ロールに権限を付与でき、この権限が適用される範囲は、文が発行されるコンテナ(CDBルート、アプリケーション・ルートまたはPDB)および権限が(CDBルートまたはアプリケーション・ルートで)共通に付与されたかどうかによって決まります。アプリケーション・ルートに接続しているCDB共通ユーザーは、CDB共通オブジェクトに対する権限を共通に付与でき、その権限はアプリケーション・コンテナ間で適用されます。

  次の図に、CDB共通ユーザー、アプリケーション共通ユーザーおよびローカル・ユーザーのアクセス階層を示します。



CDB共通ユーザーはCDBルートで定義され、CDB内のすべてのPDB(アプリケーション・ルートおよびそのアプリケーションPDBを含む)にアクセスできます。アプリケーション共通ユーザーはアプリケーション・ルートで定義され、アプリケーション・コンテナに属するPDBにアクセスできます。CDB PDBまたはアプリケーションPDBのローカル・ユーザーは、そのローカル・ユーザーが存在するPDBにのみアクセスできます。

- PDBの状態は、CDBルート、アプリケーション・ルート(PDBがアプリケーション・コンテナに属するアプリケーションPDBである場合)またはPDB自体からALTER PLUGGABLE DATABASE文を発行するための適切な権限を持つユーザーが変更できます。

CDB共通ユーザーとアプリケーション共通ユーザーの違いの1つは、CDB全体に適用されるリカバリ句を指定するALTER DATABASE文を実行できるのはCDB共通ユーザーのみであることです。

関連項目:

- 共通ユーザー・アカウントの作成について
- PDBでの権限の使用方法の詳細は、共通およびローカルに付与される権限についてを参照してください
- CDB共通ユーザーおよびアプリケーション共通ユーザーの概念については、Oracle Database概要を参照してください

親トピック: 共通ユーザーおよびローカル・ユーザーについて

2.2.1.2 PDBへの接続によるCDB共通ユーザーへの影響

CDB以外をPDBとしてCDBに差し込むと、Oracle付属の管理アカウント、ユーザー作成アカウントおよびそれらの権限に影響があります。

これは、これらのCDB共通ユーザー・アカウントのパスワードと、新しく接続したデータベース内のすべてのアカウントの権限に影響します。

次のアクションが実行されます。

- Oracleから提供される管理アカウントが既存の共通ユーザー・アカウントにマージされます。
- ユーザー作成のアカウントが既存のユーザー作成の共通ユーザー・アカウントにマージされます。
- 既存のCDB共通ユーザー・アカウントのパスワードは、非CDBのアカウントのパスワードよりも優先されます。
- 元のCDB以外のユーザー・アカウントの権限を変更した場合、これらの権限は保存されますが、ローカルに付与された権限としてCDBにPDBを接続した場合に作成されたPDBにのみ適用されます。たとえば、ユーザーSYSTEMにCDB以外のdb1のhr_mgrと呼ばれるロールを付与したとします。db1データベースがCDBに追加された後、SYSTEMは、他のPDBではなくdb1 PDBのhr_mgrロールのみ使用できます。

あるCDB (cdb_1)のPDB (たとえば、pdb_1)を別のCDB (cdb_2)に接続する場合、次の2つのシナリオが可能です。

- cdb_1には共通ユーザーc##cdb1_userがいます。cdb_2にこのユーザーはいません。

  c##cdb1_userはPDB_1に残っていますが、このアカウントはロックされています。このアカウントを復活させるには、pdb_1をクローズし、cdb_2のルートで共通ユーザーc##cdb1_userを再作成し、pdb_1を再オープンします。

- cdb_1とcdb_2の両方には、共通ユーザーc##common_userがいます。

  両方のc##common_userアカウントはマージされます。c##common_userはそのパスワードをcdb_2で保持します。cdb_2で割り当てられ、cdb_1では割り当てられていない権限は、pdb_1でローカルに保持されます。

親トピック: 共通ユーザーおよびローカル・ユーザーについて

2.2.1.3 ローカル・ユーザーについて

マルチテナント環境では、ローカル・ユーザーとは、単一のPDBにのみ存在するデータベース・ユーザーです。

ローカル・ユーザーは管理権限を持つことができますが、これらの権限はローカル・ユーザー・アカウントが作成されたPDBにのみ適用されます。ローカル・ユーザー・アカウントには、共通ユーザー・アカウントと区別する次の特性があります。

- ローカル・ユーザー・アカウントは、共通ユーザー・アカウントを作成できないまたは権限をローカルに付与できません。適切な権限を持つ共通ユーザーは、共通またはローカル・ユーザー・アカウントを作成および変更し、共通またはローカルに権限を付与および取り消すことができます。ローカル・ユーザーは、ローカル・ユーザー・アカウントを作成または変更したり、指定のPDBの共通またはローカル・ユーザーに権限をローカルに付与できます。
- ローカル・ユーザー・アカウントの共通ロールを付与できます。ただし、共通ロールに関連付けられている権限は、ローカル・ユーザーのPDBにのみ適用されます。
- ローカル・ユーザーは、そのPDB内のみで一意にする必要があります。
- 適切な権限がある場合、ローカル・ユーザーは、共通ユーザーのスキーマのオブジェクトにアクセスできます。たとえば、共通ユーザーのスキーマ内の表にアクセスするためのローカル・ユーザー権限が共通ユーザーに付与されている場合、ローカル・ユーザーはその表にアクセスできます。
- 共通ユーザー・アカウントではなくローカル・ユーザー・アカウントのエディションに対応できます。

関連トピック

- ローカル・ユーザー・アカウントの作成について
- Oracle Database概要

親トピック: 共通ユーザーおよびローカル・ユーザーについて

2.2.2 ユーザー・アカウントの作成者とは

CREATE USERシステム権限を付与されているユーザーは、ユーザー・アカウント(プロキシ・ユーザーとして使用されるユーザー・アカウントを含む)を作成できます。

CREATE USERは強力なシステム権限であるため、通常、この権限を持つユーザーはデータベース管理者またはセキュリティ管理者のみです。

自らユーザーを作成する権限を持つユーザーを作成するには、GRANT文にWITH ADMIN OPTION句を含めます。たとえば:

```
GRANT CREATE USER TO lbrown WITH ADMIN OPTION;
```

権限を付与するすべてのユーザー・アカウントと同様に、これらの権限を信頼できるユーザーのみに付与します。

マルチテナント環境で共通ユーザー・アカウントを作成するには、共通に付与されるCREATE USERシステム権限が必要です。ローカル・ユーザー・アカウントを作成するには、ローカル・ユーザー・アカウントを作成するPDB内に、共通に付与されるCREATE USER権限またはローカルに付与されるCREATE USER権限が必要です。

ノート:

セキュリティ管理者は、独自のロールを作成し、必要な権限のみを割り当てる必要があります。たとえば、それまで CONNECT 権限を付与されていた多くのユーザーは、CONNECT で提供されていた追加権限を必要とはしていませんでした。かわりに、CREATE SESSION のみが実際には必要でした。デフォルトでは、SET CONTAINER 権限が CONNECT ロールに付与されます。

組織固有のロールを作成することによって、組織で割り当てる権限を詳細に制御でき、Oracle Database が定義するロールが将来のリリースで変更された場合に組織を保護できます。

関連トピック

- 権限とロール認可の構成

親トピック: ユーザー・アカウントの作成

2.2.3 最小限のデータベース権限を持つ新しいユーザー・アカウントの作成

新しいユーザー・アカウントを作成する場合、このユーザーがデータベースにアクセスできるようにする必要があります。

1. CREATE USER文を使用して新しいユーザー・アカウントを作成します。

   たとえば:

   ```
   CREATE USER jward
    IDENTIFIED BY password
    DEFAULT TABLESPACE example
    QUOTA 10M ON example
    TEMPORARY TABLESPACE temp
    QUOTA 5M ON system
    PASSWORD EXPIRE;
   ```

   「パスワードの最低要件」のガイドラインに従って、passwordを安全なパスワードに置き換えます。

   この例では、ローカル・ユーザー・アカウントを作成し、ユーザー・パスワード、デフォルト表領域、一時セグメントが作成される一時表領域、表領域の割当て制限およびプロファイルを指定します。

2. 少なくとも、データベース・インスタンスにアクセスできるように、ユーザーにCREATE SESSION権限を付与します。

```
GRANT CREATE SESSION TO jward;
```

新規作成されたユーザーは、CREATE SESSION権限を持つまでデータベースに接続できません。Oracle Enterprise Managerへのアクセスが必要なユーザーには、SELECT ANY DICTIONARY権限も付与する必要があります。

関連トピック

- 新しいアカウントのユーザー名の作成に関する制限事項
- ユーザーへのパスワードの割当て
- ユーザーのデフォルト表領域
- ユーザーへの表領域の割当て制限
- ユーザーの一時表領域
- ユーザーのプロファイル
- 共通ユーザーまたはローカル・ユーザーの作成

親トピック: ユーザー・アカウントの作成

2.2.4 新しいアカウントのユーザー名の作成に関する制限事項

ユーザー・アカウントの名前を指定する場合は、名前が一意かどうかやネーミング規則などの制限事項に注意する必要があります。

- ユーザー名の一意性
  各ユーザーにはスキーマが関連付けられています。スキーマ内の各スキーマ・オブジェクトには、必ず一意の名前を指定する必要があります。
- マルチテナント環境のユーザー名
  各PDB内のユーザー名は、そのPDB内の他のユーザー名およびロールと比較して一意である必要があります。
- ユーザー名の大/小文字の区別
  ユーザー名をどのように作成するかによって、データベースに格納されるユーザー名の大文字と小文字の区別が決まります。

親トピック: ユーザー・アカウントの作成

2.2.4.1 ユーザー名の一意性

各ユーザーにはスキーマが関連付けられています。スキーマ内の各スキーマ・オブジェクトには、必ず一意の名前を指定する必要があります。

Oracle Databaseでは、すでに存在しているユーザー名は作成できません。DBA_USERSデータ・ディクショナリ・ビューのUSERNAME列を問い合せて、既存の名前を確認できます。

親トピック: 新しいアカウントのユーザー名の作成に関する制限事項

2.2.4.2 マルチテナント環境のユーザー名

各PDB内のユーザー名は、そのPDB内の他のユーザー名およびロールと比較して一意であることが必要です。

次の制約に注意してください:

- 共通ユーザー名の場合、ユーザー作成の共通ユーザーの名前は共通のユーザー接頭辞で始める必要があります。デ

フォルトでは、CDB共通ユーザーの場合、この接頭辞はC##です。アプリケーション共通ユーザーの場合、この接頭辞は空の文字列です。これは、CDB共通ユーザー用に予約された接頭辞を使用しないかぎり、アプリケーション共通ユーザーに割り当てる名前には制限がないことを意味します。たとえば、CDB共通ユーザーにc##hr_admin、アプリケーション共通ユーザーにhr_adminという名前を付けることができます。

CDB$ROOTのCOMMON_USER_PREFIXパラメータは、共通のユーザー接頭辞を定義します。この設定は変更できますが、十分に注意して行うようにしてください。

- ローカル・ユーザー名の場合、名前をC## (またはc##)で始めることはできません。
- ユーザーとロールに同じ名前を付けることはできません。

親トピック: 新しいアカウントのユーザー名の作成に関する制限事項

2.2.4.3 ユーザー名の大/小文字の区別

ユーザー名をどのように作成するかによって、データベースに格納されるユーザー名の大文字と小文字の区別が決まります。

たとえば:

```
CREATE USER jward
 IDENTIFIED BY password
 DEFAULT TABLESPACE data_ts
 QUOTA 100M ON test_ts
 QUOTA 500K ON data_ts
 TEMPORARY TABLESPACE temp_ts
 PROFILE clerk
 CONTAINER = CURRENT;
```

ユーザーjwardはデータベースに大文字で格納されます。たとえば:

```
SELECT USERNAME FROM ALL_USERS;
USERNAME
---------
JWARD
...
```

しかし、ユーザー名を二重引用符で囲むと、このユーザー名は名前に使用した大/小文字の区別を使用して格納されます。たとえば:

```
CREATE USER "jward" IDENTIFIED BY password;
```

そのため、ALL_USERSデータ・ディクショナリ・ビューを問い合せると、ユーザー・アカウントが作成に使用した大/小文字を使用して格納されていることがわかります。

```
SELECT USERNAME FROM ALL_USERS;
USERNAME
---------
jward
...
```

ユーザーJWARDとユーザーjwardは、どちらも別々のユーザー・アカウントとしてデータベースに格納されています。後で、二重引用符を使用して作成したユーザーを変更または削除する場合は、そのユーザー名を二重引用符で囲む必要があります。

たとえば:

```
DROP USER "jward";
```

親トピック: 新しいアカウントのユーザー名の作成に関する制限事項

2.2.5 ユーザーへのパスワードの割当て

CREATE USER文のIDENTIFIED BY句で、ユーザーにパスワードを割り当てます。

安全性の高いパスワードを作成していることを確認します。

最小限のデータベース権限を持つ新しいユーザー・アカウントの作成の例では、新しいローカル・ユーザーはデータベースを使用して認証されます。この場合、接続に成功するために、接続ユーザーはデータベースに対して正しいパスワードを指定する必要があります。

```
CREATE USER jward
 IDENTIFIED BY password
 DEFAULT TABLESPACE data_ts
 QUOTA 100M ON test_ts
 QUOTA 500K ON data_ts
 TEMPORARY TABLESPACE temp_ts
 PROFILE clerk
 CONTAINER = CURRENT;
```

関連トピック

- パスワードの最低要件
- パスワードの保護に関するガイドライン

親トピック: ユーザー・アカウントの作成

2.2.6 ユーザーのデフォルト表領域

デフォルト表領域は、ユーザーが作成するオブジェクトを格納します。

- ユーザーに対するデフォルト表領域の割当てについて
  各ユーザーには、デフォルト表領域が必要です。
- デフォルト表領域を割り当てるためのDEFAULT TABLESPACE句
  CREATE USER文のDEFAULT TABLESPACE句で、ユーザーにデフォルト表領域を割り当てます。

親トピック: ユーザー・アカウントの作成

2.2.6.1 ユーザーに対するデフォルト表領域の割当てについて

各ユーザーには、デフォルト表領域が必要です。

スキーマ・オブジェクトがユーザーのスキーマ内に作成され、かつそのDDL文でそのオブジェクトを格納する表領域が指定されていない場合、そのオブジェクトはユーザーのデフォルト表領域に格納されます。

表領域を使用すると、ユーザー・データとシステム・データ(SYSTEM表領域に格納されるデータなど)を区別できます。CREATE USER文またはALTER USER文を使用して、ユーザーにデフォルト表領域を割り当てます。すべてのユーザーのデフォルト表領域に対するデフォルト設定は、SYSTEM表領域です。ユーザーがオブジェクトを作成せず、オブジェクトを作成するための権限も持っていない場合は、デフォルト設定のままで問題ありません。ただし、任意のタイプのオブジェクトを作成する可能性があるユーザーには、デフォルト表領域(USERS表領域など)を明示的に割り当てる必要があります。SYSTEM以外の表領域を使用すると、同じデータ・ファイルに対するデータ・ディクショナリ・オブジェクトとユーザー・オブジェクト間の競合が解消されます。通常、ユーザー・データはSYSTEM表領域に格納しないでください。

CREATE TABLESPACE SQL文を使用して、データベースの作成時に、データベースの永続オブジェクトのデフォルトとして使用される、SYSTEM以外のデフォルト永続表領域を作成できます。ユーザー・データをシステム・データと区切ることにより、状況次第ではデータベース全体が機能不全になる場合もある、SYSTEM表領域に関する問題が生じる可能性を減らすことができ

ます。このデフォルトの永続表領域は、システム・ユーザー、つまりデフォルトの永続表領域がSYSTEMであるSYS、SYSTEM、およびOUTLNが使用するものではありません。デフォルト永続表領域として指定されている表領域は削除できません。目標を達成するためには、最初にデフォルトの永続表領域として他の表領域を指定する必要があります。ALTER TABLESPACE SQL文を使用して、デフォルトの永続表領域を他の表領域に変更できます。この変更はALTER DDL文の実行後に作成されたすべてのユーザーまたはオブジェクトに影響を及ぼすことに注意してください。

ユーザー作成時に、そのユーザーのデフォルト表領域を設定しておき、作成後にALTER USER文を使用して変更することもできます。ユーザーのデフォルト表領域を変更すると、設定の変更後に作成されたオブジェクトのみがこの変更の影響を受けます。

ユーザーのデフォルト表領域を指定するときは、その表領域に対する割当て制限もあわせて指定してください。

親トピック: ユーザーのデフォルト表領域

2.2.6.2 デフォルト表領域を割り当てるためのDEFAULT TABLESPACE句

CREATE USER文のDEFAULT TABLESPACE句で、ユーザーにデフォルト表領域を割り当てます。

次のCREATE USER文では、ローカル・ユーザーjwardのデフォルト表領域はdata_tsです。

```
CREATE USER jward
 IDENTIFIED BY password
 DEFAULT TABLESPACE data_ts
 QUOTA 100M ON test_ts
 QUOTA 500K ON data_ts
 TEMPORARY TABLESPACE temp_ts
 PROFILE clerk
 CONTAINER = CURRENT;
```

関連トピック

- ユーザーへの表領域の割当て制限

親トピック: ユーザーのデフォルト表領域

2.2.7 ユーザーの表領域の割当て制限

表領域の割当て制限により、ユーザーの表領域に提供される領域の量が決まります。

- ユーザーへの表領域割当て制限の割当てについて
  各ユーザーには、任意の表領域(一時表領域を除く)に対する表領域割当て制限を設定できます。
- 表領域の割当て制限を割り当てるためのCREATE USER文
  CREATE USER文のQUOTA句は、表領域の割当て制限を割り当てます。
- 表領域でのユーザー・オブジェクトに対する割当て限度の制限
  現在の割当てがゼロになるように、表領域内のユーザー・オブジェクトに割当て制限を設定できます。
- ユーザーへのUNLIMITED TABLESPACEシステム権限の付与
  データベース内の表領域を無制限に使用することをユーザーに許可するには、そのユーザーにUNLIMITED TABLESPACEシステム権限を付与します。

親トピック: ユーザー・アカウントの作成

2.2.7.1 ユーザーへの表領域割当て制限の割当てについて

各ユーザーには、任意の表領域(一時表領域を除く)に対する表領域割当て制限を設定できます。

割当て制限による影響は、次のとおりです。

- 特定タイプのオブジェクトを作成する権限があるユーザーは、指定した表領域内にオブジェクトを作成できます。
- Oracle Databaseでは、指定した表領域内にあるユーザーのオブジェクトの記憶域に対して割当て可能な領域は、割当て制限以内に制限されます。

デフォルトでは、ユーザーに対するデータベースの表領域の割当て制限はありません。ユーザーがスキーマ・オブジェクトを作成する権限を持っている場合は、このユーザーがオブジェクトを作成できるようにするための割当て制限を割り当てる必要があります。最低でも、ユーザーにはデフォルト表領域の割当て制限と、ユーザーがオブジェクトを作成する他の表領域の追加割当て制限を割り当てます。表領域に割り当てることができる最大領域は2 TBです。より多くの領域が必要な場合は、QUOTA句にUNLIMITEDを指定します。

ユーザーの割当て制限では、各表領域の一定量のディスク領域を個別に割り当てるか、またはすべての表領域のディスク領域を無制限に割り当てるかのどちらかを選択できます。一定量の割当て制限を設定すると、ユーザーのオブジェクトによってデータベースの領域が大量に使用されるのを防止できます。

ユーザーの表領域に対する割当て制限は、ユーザーを作成するときに割り当てることができ、後で割当て制限の追加または変更もできます。(USER_TS_QUOTASビューを問い合せることで、既存ユーザーの割当て制限を確認できます。)新しい割当て制限が古い割当て制限より少ない場合は、次の状況が当てはまります。

- ユーザーがすでに新しい表領域割当て制限を超過している場合は、これらのオブジェクトをあわせた領域が新しい割当て制限より少なくならないかぎり、その表領域のユーザー・オブジェクトに追加の領域を割り当てることはできません。
- ユーザーが新しい表領域割当て制限を超過していない場合、つまり表領域内でユーザーのオブジェクトが使用している領域が新しい表領域割当て制限よりも少ない場合は、そのユーザーのオブジェクトに新しい割当て制限までの領域を割り当てることができます。

親トピック: ユーザーへの表領域の割当て制限

2.2.7.2 表領域の割当て制限を割り当てるためのCREATE USER文

CREATE USER文のQUOTA句は、表領域の割当て制限を割り当てます。

次のCREATE USER文では、test_tsおよびdata_ts表領域に対して割当て制限を割り当てています。

```
CREATE USER jward
 IDENTIFIED BY password
 DEFAULT TABLESPACE data_ts
 QUOTA 500K ON data_ts
 QUOTA 100M ON test_ts
 TEMPORARY TABLESPACE temp_ts
 PROFILE clerk
 CONTAINER = CURRENT;
```

親トピック: ユーザーへの表領域の割当て制限

2.2.7.3 表領域でのユーザー・オブジェクトに対する割当て限度の制限

現在の割当てがゼロになるように、表領域内のユーザー・オブジェクトに割当て制限を設定できます。

割当て制限を指定するには、ALTER USER SQL文を使用します。

ゼロの割当て制限が割り当てられると、表領域内のユーザーのオブジェクトはそのまま残り、ユーザーは引き続き新規オブジェクトを作成できますが、既存のオブジェクトには新しい領域が割り当てられなくなります。たとえば、このユーザーの既存の表の1つにデータを挿入することはできません。操作は失敗し、 ORA-1536表に対する領域割当て制限を使い果たしました。エラーが発生します。

親トピック: ユーザーへの表領域の割当て制限

2.2.7.4 ユーザーへのUNLIMITED TABLESPACEシステム権限の付与

データベース内の表領域を無制限に使用することをユーザーに許可するには、そのユーザーにUNLIMITED TABLESPACEシステム権限を付与します。

UNLIMITED TABLESPACE権限によって、そのユーザーに対する明示的な表領域の割当て制限がすべて置き換えられます。後で権限を取り消す場合、個々の表領域に対して割当て制限を明示的に付与する必要があります。この権限は、ロールに対してではなく、ユーザーに対してのみ付与できます。

UNLIMITED TABLESPACEシステム権限を付与する前に、この方法のメリットとデメリットを考慮してください。

メリット:

- データベースのすべての表領域に無制限にアクセスできる権限を1つの文でユーザーに付与できます。

デメリット:

- この権限によって、そのユーザーに対する明示的な表領域割当て制限がすべて置き換えられます。
- UNLIMITED TABLESPACEシステム権限を持つユーザーから表領域へのアクセス権を選択的に取り消すことはできません。この権限を取り消した後でのみ、選択的または制限付きアクセス権を付与できます。

親トピック: ユーザーへの表領域の割当て制限

2.2.8 ユーザーの一時表領域

一時表領域には、ユーザー・セッションの存続期間中のみ保持される一時データが含まれています。

- ユーザーに対する一時表領域の割当てについて
  各ユーザーには、一時表領域を割り当てる必要があります。
- 一時表領域を割り当てるためのTEMPORARY TABLESPACE句
  CREATE USER文のTEMPORARY TABLESPACE句で、ユーザーに一時表領域を割り当てます。

親トピック: ユーザー・アカウントの作成

2.2.8.1 ユーザーに対する一時表領域の割当てについて

各ユーザーには、一時表領域を割り当てる必要があります。

ユーザーが一時セグメントを必要とするSQL文を実行すると、このセグメントはそのユーザーの一時表領域に格納されます。これらの一時セグメントは、ソート操作または結合操作の実行時にシステムによって作成されます。一時セグメントは、すべての表領域のリソースに関する権限を持つSYSが所有します。

一時表領域を作成するには、CREATE TEMPORARY TABLESPACE SQL文を使用できます。

ユーザーの一時表領域を明示的に割り当てない場合、そのユーザーには、データベース作成時に指定された、または作成後にALTER DATABASE文によって指定されたデフォルト一時表領域が割り当てられます。デフォルトの一時表領域が明示的に割り当てられていない場合、デフォルトはSYSTEM表領域またはシステム管理者が設定した別のデフォルト永続表領域になります。一時表領域として使用する表領域を明示的に割り当てることによって、一時セグメントとそれ以外のタイプのセグメントとの間で発生するファイルの競合が解消されます。

ノート:

SYSTEM 表領域がローカル管理の場合、ユーザーには特定のデフォルト(ローカル管理)一時表領域を割り当てる必要があります。永続的なローカル管理表領域には、一時オブジェクトを格納できないため、SYSTEM 表領域の使用はデフォルトで禁止されます。

ユーザー作成時に、そのユーザーの一時表領域を設定しておき、作成後にALTER USER文を使用して変更できます。個々の一時表領域を割り当てるかわりに、表領域グループを設定することもできます。

関連トピック

- Oracle Database管理者ガイド

親トピック: ユーザーの一時表領域

2.2.8.2 一時表領域を割り当てるためのTEMPORARY TABLESPACE句

CREATE USER文のTEMPORARY TABLESPACE句で、ユーザーに一時表領域を割り当てます。

次の例では、jwardの一時表領域はtemp_tsです。これは一時セグメントのみを格納するために明示的に作成された表領域です。

```
CREATE USER jward
 IDENTIFIED BY password
 DEFAULT TABLESPACE data_ts
 QUOTA 100M ON test_ts
 QUOTA 500K ON data_ts
 TEMPORARY TABLESPACE temp_ts
 PROFILE clerk
 CONTAINER = CURRENT;
```

親トピック: ユーザーの一時表領域

2.2.9 ユーザーのプロファイル

プロファイルとは、データベース・リソースとそのデータベースへのパスワード・アクセスについて属性によって定義される一連の制限です。

プロファイルは複数のユーザーに適用でき、それらのユーザーはそうした属性を共有できます。

ユーザーの作成時に、プロファイルを指定できます。CREATE USER文のPROFILE句は、ユーザーにプロファイルを割り当てます。プロファイルを指定しない場合、ユーザーにはデフォルト・プロファイルが割り当てられます。

たとえば:

```
CREATE USER jward
 IDENTIFIED BY password
 DEFAULT TABLESPACE data_ts
 QUOTA 100M ON test_ts
 QUOTA 500K ON data_ts
 TEMPORARY TABLESPACE temp_ts
 PROFILE clerk
 CONTAINER = CURRENT;
```

マルチテナント環境では、ルートおよびPDBの共通ユーザーに異なるプロファイルを割り当てることができます。共通ユーザーがPDBにログインすると、設定がセッションに適用されるプロファイルは、設定がパスワード関連であるかリソース関連であるかに応じて異なります。

- パスワード関連のプロファイル設定は、ルートの共通ユーザーに割り当てられたプロファイルからフェッチされます。たとえば、共通プロファイルc##prof(FAILED_LOGIN_ATTEMPTSが1に設定されている)をルートの共通ユーザーc##adminに割り当てるとします。PDBでは、そのユーザーにはローカル・プロファイルlocal_prof(FAILED_LOGIN_ATTEMPTSを6に設定)が割り当てられるとします。loc_profが割り当てられているPDBにログインする場合、共通ユーザーc##adminは、1回のログインの失敗のみ許可されます。
- ルートの共通ユーザーに割り当てられたプロファイルのリソース関連の設定を参照せずに、PDBのユーザーに割り当てられたプロファイルに指定されたリソース関連のプロファイル設定が使用されます。たとえば、PDBのユーザーc##adminに割り当てられたプロファイルlocal_profでSESSIONS_PER_USERが2に設定されている場合、c##adminは、loc_profが割り当てられたPDBにログインすると、ルートで割り当てられたプロファイルにおけるこの設定の値に関係なく、許可される同時セッション数は2つのみになります。

関連トピック

- プロファイルによるリソースの管理

親トピック: ユーザー・アカウントの作成

2.2.10 共通ユーザーまたはローカル・ユーザーの作成

CREATE USER SQL文を使用して、共通(CDBおよびアプリケーション)ユーザーとローカル・ユーザーの両方を作成できます。

- 共通ユーザー・アカウントの作成について
  作成できる場所、ネーミング規則、スキーマに保存されるオブジェクトなど、共通ユーザー・アカウントの制限事項に注意してください。
- 共通ユーザー・アカウントを作成するためのCREATE USER文
  CREATE USER文のCONTAINER=ALL句を使用して、共通ユーザー・アカウントを作成できます。
- ローカル・ユーザー・アカウントの作成について
  作成できる場所、ネーミング規則、スキーマに保存されるオブジェクトなど、ローカル・ユーザー・アカウントの制限事項に注意してください。
- ローカル・ユーザー・アカウントを作成するためのCREATE USER文
  CREATE USER文のCONTAINER句を使用して、ローカル・ユーザー・アカウントを作成できます。

親トピック: ユーザー・アカウントの作成

2.2.10.1 共通ユーザー・アカウントの作成について

作成できる場所、ネーミング規則、スキーマに保存されるオブジェクトなど、共通ユーザー・アカウントの制限事項に注意してください。

共通ユーザー・アカウントを作成するには、次の規則に従います。

- CDB共通ユーザーを作成するには、CDBルートに接続し、共通に付与されるCREATE USERシステム権限を持っていることが必要です。
- アプリケーション共通ユーザーを作成するには、アプリケーション・ルートに接続し、共通に付与されるCREATE USERシステム権限を持っていることが必要です。
- CREATE USER ... CONTAINER = ALL文を実行して、アプリケーション・ルートでアプリケーション共通ユーザーを作成できます。その後、このユーザーがアプリケーションPDBで表示されるように、アプリケーションを同期する必要があります。たとえば、saas_sales_appというアプリケーションでは次のようになります。

```
ALTER PLUGGABLE DATABASE APPLICATION saas_sales_app SYNC;
```

- CDBルートに接続する共通ユーザーに付ける名前は、CDBルートのCOMMON_USER_PREFIXパラメータで定義された接頭辞(デフォルトでは、C##)で始まる必要があります。(このパラメータは変更できますが、十分に注意して行うようにしてください。)名前に使用できるのは、ASCIIまたはEBCDIC文字のみです。このネーミング要件は、SYSやSYSTEMなど、Oracle提供の既存のユーザー・アカウントの名前には適用されません。既存のユーザー・アカウントの名前を確認するには、ALL_USERS、CDB_USERS、DBA_USERSおよびUSER_USERSデータ・ディクショナリ・ビューを問い合せます。
- アプリケーション・ルートに接続する共通ユーザーに付ける名前は、標準ユーザー・アカウントのネーミング規則に従う必要があります。デフォルトでは、アプリケーション・ルートでCOMMON_USER_PREFIXパラメータが空の文字列に設定されています。つまり、アプリケーション・ルートでhr_adminというユーザーを作成できますが、c##hr_adminというユーザーは作成できません。
- CDBまたはアプリケーション共通ユーザーとしてユーザー・アカウントを明示的に指定するには、CREATE USER文で、CONTAINER=ALL句を指定します。CDBルートまたはアプリケーション・ルートにログインしている場合に、CREATE USER文からCONTAINER句を省略すると、CONTAINER=ALL句が暗黙のうちに入れられます。
- CDBの共通ユーザーのスキーマにはオブジェクトを作成しないでください。かわりに、アプリケーション共通オブジェクトを作成できます。これらは、アプリケーション・コンテナに属するすべてのアプリケーションPDB間でメタデータおよび(データ・リンクまたは拡張データ・リンクの場合は)データが共有されるオブジェクトです。アプリケーション共通オブジェクトは、アプリケーション・コンテナのルートで作成する必要があります。
- CDBまたはアプリケーション共通ユーザー・アカウントのCREATE USER文でDEFAULT TABLESPACE、TEMPORARY TABLESPACE、QUOTA...ONおよびPROFILE句を指定する場合、CDB共通ユーザーのCDBのすべてのコンテナに、またはアプリケーション共通ユーザーのアプリケーション・コンテナのアプリケーション・ルートおよびすべてのPDBに、これらのオブジェクト(表領域、表領域グループおよびプロファイル)が存在することを確認する必要があります。

親トピック: 共通ユーザーまたはローカル・ユーザーの作成

2.2.10.2 共通ユーザー・アカウントを作成するためのCREATE USER文

CREATE USER文のCONTAINER=ALL句を使用して、共通ユーザー・アカウントを作成できます。

CDB共通ユーザー・アカウントを作成するにはCDBルートに、アプリケーション共通ユーザー・アカウントを作成するにはアプリケーション・ルートにいる必要があります。

次の例は、CONTAINER句を使用してCDBルートからCDB共通ユーザー・アカウントを作成し、そのユーザーにSET CONTAINERおよびCREATE SESSION権限を付与する方法を示しています。共通ユーザーは、コンテナ間を移動するためにSET CONTAINERシステム権限を必要とします。このアカウントを作成すると、すべてのコンテナでこの共通ユーザーに1つの共通パスワードが作成されます。

```
CONNECT SYSTEM
Enter password: password
Connected.
CREATE USER c##hr_admin
IDENTIFIED BY password
DEFAULT TABLESPACE data_ts
QUOTA 100M ON test_ts
QUOTA 500K ON data_ts
TEMPORARY TABLESPACE temp_ts
CONTAINER = ALL;
GRANT SET CONTAINER, CREATE SESSION TO c##hr_admin
CONTAINER = ALL;
```

次の例は、CONTAINER句を使用してアプリケーション・ルート(app_root)でアプリケーション共通ユーザーを作成し、そのユーザーにSET CONTAINERおよびCREATE SESSIONシステム権限を付与する方法を示しています。最後に、このユーザーがアプリケーションPDBで表示されるように同期するため、ALTER PLUGGABLE DATABASE APPLICATION APP$CON SYNC文が実行されています。

```
CONNECT SYSTEM@app_root
Enter password: password
Connected.
CREATE USER app_admin
IDENTIFIED BY password
DEFAULT TABLESPACE data_ts
QUOTA 100M ON temp_ts
QUOTA 500K ON data_ts
TEMPORARY TABLESPACE temp_ts
CONTAINER = ALL;
GRANT SET CONTAINER, CREATE SESSION TO app_admin CONTAINER = ALL;
CONNECT SYSTEM@app_hr_pdb
Enter password: password
Connected.
ALTER PLUGGABLE DATABASE APPLICATION APP$CON SYNC;
```

関連トピック

- 共通ユーザーについて
- Enterprise Managerの共通ユーザー・アカウントの作成

親トピック: 共通ユーザーまたはローカル・ユーザーの作成

2.2.10.3 ローカル・ユーザー・アカウントの作成について

作成できる場所、ネーミング規則、スキーマに保存されるオブジェクトなど、ローカル・ユーザー・アカウントの制限事項に注意してください。

ローカル・ユーザー・アカウントを作成するには、次の規則に従います。

- ローカル・ユーザー・アカウントを作成するには、アカウントを作成するPDBに接続して、CREATE USER権限を持つ必要があります。
- ローカル・ユーザーに付ける名前は、共通ユーザー用に予約された接頭辞(CDB共通ユーザーの場合、デフォルトではC##)で始めることはできません。
- CREATE USER文にCONTAINER=CURRENTを含め、ローカル・ユーザーとしてユーザーを指定できます。PDBに接続しており、この句を省略すると、CONTAINER=CURRENT句が含まれます。
- 共通ユーザーとローカル・ユーザーの名前を同じにすることはできません。ただし、異なるPDBのローカル・ユーザーに同じ名前を使用できます。既存のユーザー・アカウントの名前を確認するには、ALL_USERS、CDB_USERS、DBA_USERSおよびUSER_USERSデータ・ディクショナリ・ビューを問い合せます。
- 適切な権限があれば、PDBに接続されている共通ユーザーとローカル・ユーザーの両方で、ローカル・ユーザー・アカウントを作成できます。

親トピック: 共通ユーザーまたはローカル・ユーザーの作成

2.2.10.4 ローカル・ユーザー・アカウントを作成するためのCREATE USER文

CREATE USER文のCONTAINER句を使用して、ローカル・ユーザー・アカウントを作成できます。

ローカル・ユーザー・アカウントは、そのアカウントを配置するPDBに作成する必要があります。

次の例は、CONTAINER句を使用したローカル・ユーザー・アカウントの作成方法を示しています。

```
CONNECT SYSTEM@hrpdb
Enter password: password
Connected.
CREATE USER kmurray
 IDENTIFIED BY password
 DEFAULT TABLESPACE data_ts
 QUOTA 100M ON test_ts
 QUOTA 500K ON data_ts
 TEMPORARY TABLESPACE temp_ts
 PROFILE hr_profile
 CONTAINER = CURRENT;
```

関連トピック

- Enterprise Managerの共通ユーザー・アカウントの作成
- ローカル・ユーザーについて

親トピック: 共通ユーザーまたはローカル・ユーザーの作成

2.2.11 ユーザーのデフォルト・ロールの作成

デフォルト・ロールは、ユーザーがセッションを作成したときに、自動的にそのユーザーに対して使用可能になります。

ユーザーには複数のデフォルト・ロールを割り当てることができますが、割り当てなくてもかまいません。CREATE USER文ではユーザーのデフォルト・ロールを設定できません。ユーザーを最初に作成すると、ユーザーのデフォルト・ロール設定はALLであり、ユーザーにその後に付与されるすべてのロールがデフォルト・ロールになります。

- ALTER USER文を使用して、ユーザーのデフォルト・ロールを変更します。

たとえば:

```
GRANT USER rdale clerk_mgr;
ALTER USER rdale DEFAULT ROLE clerk_mgr;
```

ロールをユーザーのデフォルト・ロールにするには、そのロールがユーザーに付与されている必要があります。

関連トピック

- ユーザー・ロールの管理

親トピック: ユーザー・アカウントの作成

2.3 ユーザー・アカウントの変更

ALTER USER文で、デフォルトの表領域やプロファイル、ユーザー・パスワードの変更など、ユーザー・アカウントを変更します。

- ユーザー・アカウントの変更について
  ユーザーのセキュリティ設定の変更は、現行セッションではなく、それより後のユーザー・セッションから反映されます。
- 共通またはローカル・ユーザー・アカウントの変更方法
  ALTER USER文またはPASSWORDコマンドを使用して、共通ユーザー・アカウントとローカル・ユーザー・アカウントの両方を変更できます。
- SYS以外のユーザー・パスワードの変更
  ユーザーは、自身のパスワードを変更できますが、他のユーザーのパスワードを変更するには、正しい権限が必要です。
- SYSユーザーのパスワードの変更

SYSユーザーのパスワードを変更するには、ALTER USER文、PASSWORDコマンドまたはORAPWDコマンドライン・ユーティリティを使用できます。

親トピック: Oracle Databaseユーザーのセキュリティの管理

2.3.1 ユーザー・アカウントの変更について

ユーザーのセキュリティ設定の変更は、現行セッションではなく、それより後のユーザー・セッションから反映されます。

ほとんどの場合、ユーザーのセキュリティ設定を変更するには、ALTER USER SQL文を使用します。ユーザーは、自分のパスワードを変更できます。ただし、ユーザーのセキュリティ・ドメインの他のオプションを変更するには、ALTER USERシステム権限が必要です。通常は、セキュリティ管理者のみがこのシステム権限を持ちます。これは、この権限によってすべてのユーザーのセキュリティ・ドメインを変更できるためです。この権限には、データベースの任意の表領域に対するユーザーの表領域割当て制限を設定する許可が含まれています。これは、変更を実行するユーザーが、指定した表領域に対する割当て制限を持っていない場合も同じです。

マルチテナント環境で共通ユーザー・アカウントを変更するには、共通に付与されるALTER USERシステム権限が必要です。ローカル・ユーザー・アカウントを変更するには、ローカル・ユーザー・アカウントが存在するPDB内で、共通に付与されるALTER USER権限またはローカルに付与されるALTER USER権限が必要です。

親トピック: ユーザー・アカウントの変更

2.3.2 共通ユーザー・アカウントまたはローカル・ユーザー・アカウントの変更方法

ALTER USER文またはPASSWORDコマンドを使用して、共通ユーザー・アカウントとローカル・ユーザー・アカウントの両方を変更できます。

既存の共通ユーザー・アカウントをローカル・ユーザー・アカウントに変更したり、ローカル・ユーザー・アカウントを共通ユーザー・アカウントに変更することはできません。この場合は、共通ユーザー・アカウントまたはローカル・ユーザー・アカウントとして、新しいアカウントを作成する必要があります。

次の例は、ALTER USER文を使用して、ユーザーc##hr_adminがV$SESSION行を表示する機能を、接続先がCDB$ROOTならびにemp_dbおよびhr_db PDBであるセッションに関連するものに制限する方法を示しています。

```
CONNECT SYSTEM
Enter password: password
Connected.
ALTER USER c##hr_admin
 DEFAULT TABLESPACE data_ts
 TEMPORARY TABLESPACE temp_ts
 QUOTA 100M ON data_ts
 QUOTA 0 ON test_ts
 SET CONTAINER_DATA = (emp_db, hr_db) FOR V$SESSION
 CONTAINER = CURRENT;
```

このALTER USER文によって、ユーザーc##hr_adminのセキュリティ設定は次のように変更されます。

- DEFAULT TABLESPACEおよびTEMPORARY TABLESPACEは、data_tsおよびtemp_tsにそれぞれ明示的に設定されます。
- QUOTA 100Mは、data_ts表領域に100MBを提供します。
- QUOTA 0は、temp_ts表領域の割当て制限を取り消します。
- SET CONTAINER_DATAは、ユーザーc##hr_adminがemp_dbおよびhr_db PDBに関するデータにアクセスすることを可能にし、このユーザーがルートからV$SESSIONビューを問い合せる場合はルートに関するデータにアクセスす

ることを可能にします。

パスワードを変更するにはALTER USERを使用できますが、SYS以外のユーザー・アカウントとSYSユーザー・アカウントの両方で、PASSWORDコマンドを使用してパスワードを変更することをお薦めします。

関連トピック

- Oracle Database SQL言語リファレンス
- SYS以外のユーザーのパスワードの変更について
- SYSユーザーのパスワードの変更について

親トピック: ユーザー・アカウントの変更

2.3.3 SYS以外のユーザー・パスワードの変更

ユーザーは、自身のパスワードを変更できますが、他のユーザーのパスワードを変更するには、正しい権限が必要です。

- SYS以外のユーザーのパスワードの変更について
  ユーザーは、PASSWORDコマンドまたはALTER USER文を使用してパスワードを変更できます。
- PASSWORDコマンドまたはALTER USER文を使用したパスワードの変更
  ほとんどのユーザーが、SQL*PlusのPASSWORDコマンドまたはALTER USER SQL文を使用して自身のパスワードを変更できます。

親トピック: ユーザー・アカウントの変更

2.3.3.1 SYS以外のユーザーのパスワードの変更について

ユーザーは、PASSWORDコマンドまたはALTER USER文を使用してパスワードを変更できます。

ユーザーが自分のパスワードを変更するには、特別な権限(データベースへの接続およびセッションの作成権限以外)は不要です。ユーザーには、自分のパスワードを頻繁に変更することを薦めてください。ALL_USERSビューを問い合せることで、現行データベース・インスタンスの既存のユーザーを検索できます。

セキュリティを強化するには、PASSWORDコマンドを使用してアカウントのパスワードを変更します。ALTER USER文では、新規パスワードが画面に表示されるため、他者に見られる可能性があります。PASSWORDコマンドでは新規パスワードが表示されないため、自分のみが把握し、他者に知られることはありません。PASSWORDコマンドは、ネットワーク上のパスワードも暗号化します。ALTER USERはパスワードをクリア・テキストで送信するため、クライアントとデータベース間のネットワーク接続が暗号化されているか、セッションがネットワークを介してルーティングされないローカル・セッションである場合を除き、パスワードを使用しないでください。

ユーザーが認証方式を切り替えるには、PASSWORDとALTER USER権限が必要です。通常、この権限を持つのは管理者のみです。

関連トピック

- パスワードの最低要件
- パスワードの保護に関するガイドライン
- 認証の構成

親トピック: SYS以外のユーザー・パスワードの変更

2.3.3.2 PASSWORDコマンドまたはALTER USER文を使用したパスワードの変更

ほとんどのユーザーが、SQL*PlusのPASSWORDコマンドまたはALTER USER SQL文を使用して自身のパスワードを変更でき

ます。

マルチテナント環境では、CDB共通ユーザーはCDBルートで自分のパスワードを変更する必要があり、アプリケーション共通ユーザーはアプリケーション・ルートで自分のパスワードを変更する必要があります。

- 次のいずれかの方法を使用して、ユーザーのパスワードを変更します。
  - SQL*PlusのPASSWORDコマンドを使用してパスワードを変更するには、ユーザーの名前を指定し、プロンプトが表示された場合は新しいパスワードを入力します。

    たとえば:

    ```
    PASSWORD andy
    Changing password for andy
    New password: password
    Retype new password: password
    ```

  - ALTER USER SQL文を使用してパスワードを変更するには、IDENTIFIED BY句を指定します。

    たとえば:

    ```
    ALTER USER andy IDENTIFIED BY password;
    ```

親トピック: SYS以外のユーザー・パスワードの変更

2.3.4 SYSユーザー・パスワードの変更

SYSユーザーのパスワードを変更するには、ALTER USER文、PASSWORDコマンドまたはORAPWDコマンドライン・ユーティリティを使用できます。

- SYSユーザー・パスワードの変更について
  選択したSYSパスワードを変更する方法は、データベースの構成方法(たとえば、REMOTE_LOGIN_PASSWORDFILE初期化パラメータの設定方法)によって異なります。
- SYSユーザーのパスワードを変更するためのORAPWDユーティリティ
  ORAPWDユーティリティでは、SYSユーザーのパスワードを変更できます。

親トピック: ユーザー・アカウントの変更

2.3.4.1 SYSユーザーのパスワードの変更について

選択したSYSパスワードを変更する方法は、データベースの構成方法(たとえば、REMOTE_LOGIN_PASSWORDFILE初期化パラメータの設定方法)によって異なります。

PASSWORDコマンド、ALTER USER文またはORAPWDユーティリティを使用して、SYSパスワードを変更できます。

SYS以外のユーザー・アカウントと同様に、PASSWORDを使用してSYSユーザー・アカウントを変更するのには十分な理由があります。PASSWORDは画面に新しいパスワードを表示せず、さらにPASSWORDはネットワーク上のパスワードを暗号化します。ALTER USERはパスワードをクリア・テキストで送信するため、クライアントとデータベース間のネットワーク接続が暗号化されているか、セッションがネットワークを介してルーティングされないローカル・セッションである場合を除き、パスワードを使用しないでください。したがって、リモート接続にはPASSWORDを使用する必要があります。

ALTER USER文には、ORAPWDを使用するよりも次の利点があります。

- Oracleデータベース・インスタンス内からSYSユーザーのパスワードを変更できます。
- Oracle Data Guard環境では、SYSパスワードの変更をOracle Data Guardインスタンスに伝播します。

共有パスワード・ファイルを使用するOracle Real Application Clusters (Oracle RAC)データベースには REMOTE_LOGIN_PASSWORDFILE = SHAREDが設定されるため、ALTER USERがSYSパスワードを更新できないことに注意してください。パスワード・ファイルが共有されておらず、パスワードが変更されている場合は、Oracle RACクラスタのすべてのノードにパスワード・ファイルをコピーする必要があります。

REMOTE_LOGIN_PASSWORDFILE初期化パラメータが設定されており、ALTER USERを使用してSYSパスワードを変更する場合は、次の点に注意してください:

- REMOTE_LOGIN_PASSWORDFILE初期化パラメータがEXCLUSIVEに設定されていることを確認します。そうしないと、SYSユーザー・パスワードの変更(または管理ユーザー・パスワードの変更)の試行は失敗します。
- REMOTE_LOGIN_PASSWORDFILEがnullの場合、またはNONEに設定されている場合は、パスワードを変更しようとすると失敗し、「ORA-01994: パスワード・ファイルが欠落しているか、無効です」というエラーが発生します。
- REMOTE_LOGIN_PASSWORDFILEをSHAREDに設定した場合、パスワードを変更するALTER USER文は、ORA-28046: Password change for SYS disallowedエラーで失敗します。

ORAPWDを使用してSYSパスワードを変更する場合は、次の点に注意してください:

- SYSユーザー・アカウントのパスワードを変更するには、このアカウントのパスワード・ファイルが存在している必要があります。
- インスタンス初期化パラメータREMOTE_LOGIN_PASSWORDFILEがSHAREDに設定されている場合またはnullの場合は、ORAPWDを使用してSYSパスワードを変更する必要があります。

SYSユーザー・パスワードを変更するALTER USERメソッドとORAPWDメソッドの両方に、次が適用されます。

- 新しいアカウントは、SHA-2 (SHA-512)ベリファイアで作成されます。SYSユーザー・ベリファイアは、sqlnet.oraパラメータALLOWED_LOGON_VERSION_SERVERに基づいて生成されます。DBA_USERSデータ・ディクショナリ・ビューのPASSWORD_VERSIONS列を問い合せて、これらのアカウントを識別できます。(これらのベリファイアは、DBA_USERSビュー出力のPASSWORD_VERSIONS列に12Cとしてリストされます。)
- Oracle Real Application Clusters (Oracle RAC)環境では、複数のOracle RACインスタンスで共有できるように、パスワードをASMディスク・グループに格納してください。

関連トピック

- パスワードのセキュリティへの脅威からの12Cパスワード・バージョンによる保護
- Oracle Database管理者ガイド

親トピック: SYSユーザー・パスワードの変更

2.3.4.2 SYSユーザーのパスワードを変更するためのORAPWDユーティリティ

ORAPWDユーティリティでは、SYSユーザーのパスワードを変更できます。

ORAPWDユーティリティでINPUT_FILEパラメータを使用して、SYSユーザーのパスワードを変更できます。パスワード・ファイルを特定の形式に移行するには、FORMATオプションを指定します。デフォルトでは、FORMATオプションが指定されていない場合の形式は12.2です。

ORAPWDユーティリティを使用してSYSユーザーの新しいパスワードを設定するには、SYSオプションをY (はい)に設定し、INPUT_FILEパラメータを使用して現在のパスワード・ファイル名を指定し、FILEパラメータを使用して元のパスワード・ファイルの移行先のパスワード・ファイルを作成します。たとえば:

```
ORAPWD INPUT_FILE='orapworcl' FILE='orapwd' SYS=Y
```

```
Enter password for SYS: new_password
```

new_passwordを安全なパスワードに置き換えます。パスワード・ファイルを別の形式に移行しない場合、input_fileと同じ形式を指定できます。たとえば、入力ファイルorapworclの形式が12で、SYSユーザー・パスワードを変更する場合、次のようになります。

```
ORAPWD INPUT_FILE='orapworcl' FILE='orapwd' FORMAT=12 SYS=Y
Enter password for SYS: new_password
```

関連トピック

- Oracle Database管理者ガイド
- パスワードの保護に関するガイドライン

親トピック: SYSユーザー・パスワードの変更

2.4 ユーザー・リソース制限の構成

リソースの制限により、ユーザーに利用可能なシステム・リソースの量が決まります。

- ユーザー・リソース制限について
  ユーザーのセキュリティ・ドメインの一部として、各ユーザーが使用できるシステム・リソースの容量に制限を設定できます。
- システム・リソースのタイプと制限
  CPU時間や論理読取りなど、各種のシステム・リソースについて、セッション・レベル、コール・レベルまたはその両方のレベルで制限を課すことができます。
- プロファイルのリソース制限の値
  プロファイルを作成しリソース制限を設定する前に、各リソース制限について適切な値を決定する必要があります。
- プロファイルによるリソースの管理
  プロファイルとは、ユーザーのデータベース使用とインスタンス・リソースを制限する一連のリソース制限およびパスワード・パラメータに名前を付けたものです。

親トピック: Oracle Databaseユーザーのセキュリティの管理

2.4.1 ユーザー・リソース制限について

ユーザーのセキュリティ・ドメインの一部として、各ユーザーが使用できるシステム・リソースの容量に制限を設定できます。

そうすることで、CPUタイムなど貴重なシステム・リソースが無制限に消費されることを防ぐことができます。

このリソース制限機能は、システム・リソースに多額の費用がかかる大規模なマルチ・ユーザー・システムでは非常に有効です。1人以上のユーザーが過度にリソースを使用すると、データベースの他のユーザーに有害な影響を与える可能性があります。シングル・ユーザー・データベースや小規模なマルチ・ユーザー・データベースの場合は、ユーザーがシステム・リソースを消費しても影響は少ないため、システム・リソース機能はそれほど重要ではありません。

ユーザーのリソース制限は、データベース・リソース・マネージャを使用して管理します。パスワード管理の作業環境は、ユーザーごとに個別にプロファイルを使用して設定するか、または多数のユーザー用のデフォルト・プロファイルを使用して設定できます。それぞれのOracleデータベースに指定できるプロファイルの数に、制限はありません。Oracle Databaseでは、セキュリティ管理者が、プロファイルによるリソース制限の規定を全体的に使用可能または使用禁止に設定できます。

リソース制限を設定すると、ユーザーによるセッション作成時に、パフォーマンスがわずかに低下します。これは、各ユーザーがデータベースに接続した時点で、そのユーザーのすべてのリソース制限データがロードされるためです。

関連トピック

- Oracle Database管理者ガイド

親トピック: ユーザー・リソース制限の構成

2.4.2 システム・リソースのタイプと制限

CPU時間や論理読取りなど、各種のシステム・リソースについて、セッション・レベル、コール・レベルまたはその両方のレベルで制限を課すことができます。

- ユーザー・セッション・レベルの制限
  ユーザーがデータベースに接続すると、セッションが作成されます。セッションではCPU時間とメモリーが使用されますが、これらに制限を設定できます。
- データベース・コール・レベルの制限
  ユーザーがSQL文を実行するたびに、Oracle Databaseでは、いくつかのステップが実行され文が処理されます。
- CPU時間の制限
  SQL文やその他のコールがOracle Databaseに発行されると、そのコールを処理するためにCPU時間が必要になります。
- 論理読取りの制限
  入出力(I/O)は、データベース・システムで最もリソースの使用量が多い操作の1つです。
- その他のリソースの制限
  ユーザーの同時セッション数やアイドル時間に制限を設定できます。

親トピック: ユーザー・リソース制限の構成

2.4.2.1 ユーザー・セッション・レベルの制限

ユーザーがデータベースに接続すると、セッションが作成されます。セッションではCPU時間とメモリーが使用されますが、これらに制限を設定できます。

複数のリソース制限をセッション・レベルで設定できます。ユーザーがセッション・レベルのリソース限度を超えると、Oracle Databaseは現在の文を終了(ロールバック)して、セッション限度に達したことを示すメッセージを返します。この時点で、カレント・トランザクションのそれ以前の文のすべてがそのまま残り、ユーザーが実行できる操作のみが`COMMIT`、`ROLLBACK`、または切断になります(この場合、カレント・トランザクションはコミットされます)。他のすべての操作はエラーになります。トランザクションがコミットまたはロールバックされた後も、ユーザーはこれ以上の作業をカレント・セッション中は完了できません。

親トピック: システム・リソースのタイプと制限

2.4.2.2 データベース・コール・レベルの制限

ユーザーがSQL文を実行するたびに、Oracle Databaseでは、いくつかのステップが実行され文が処理されます。

SQL文の処理では、データベースに対して複数のコールが異なる実行フェーズの一部として発行されます。1回のコールで過度にシステムが使用されないように、Oracle Databaseでは、複数のリソース制限をコール・レベルで設定できます。

ユーザーがコール・レベルのリソース制限を超えると、Oracle Databaseは文の処理を停止してその文をロールバックし、エラーを戻します。ただし、カレント・トランザクションのそれ以前の文の結果はそのまま残り、そのユーザー・セッションは接続されたままになります。

親トピック: システム・リソースのタイプと制限

2.4.2.3 CPU時間の制限

SQL文やその他のコールがOracle Databaseに発行されると、そのコールを処理するためにCPU時間が必要になります。

平均的なコールであれば、わずかなCPUタイムですみます。ただし、大量のデータや冗長な問合せを伴うSQL文はCPUタイムを大量に使用することがあるため、他の処理に使用できるCPUタイムが少なくなります。

CPUタイムが無制限に消費されないようにするため、1回のコール当たりのCPUタイムと、1つのセッション中にOracle Databaseコールに使用されるCPUタイムの合計に対して、固定した制限または動的な制限を設定できます。これらの制限は、コールやセッションに使用される1/100秒(0.01秒)単位のCPUタイムで設定し、測定されます。

親トピック: システム・リソースのタイプと制限

2.4.2.4 論理読取りの制限

入出力(I/O)は、データベース・システムで最もリソースの使用量が多い操作の1つです。

I/Oを集中的に実行するSQL文は、メモリーとディスクの使用を独占することがあるため、他のデータベース操作がこれらのリソースをめぐって競合する原因になる可能性があります。

単一の原因による過度のI/Oが発生しないようにするために、1コール当たりおよび1セッション当たりの論理データ・ブロック読取り数を制限できます。論理データ・ブロック読取りには、メモリーとディスクの両方からの論理データ・ブロック読取りが含まれます。これらの制限は、1コールまたは1セッション中に実行されるブロック読取りの数として設定し、測定されます。

親トピック: システム・リソースのタイプと制限

2.4.2.5 その他のリソースの制限

ユーザーの同時セッション数やアイドル時間に制限を設定できます。

その他のリソースの制限を次に示します。

- ユーザー当たりの同時実行セッション数の制限。各ユーザーは、事前に定義された数まで同時実行セッションを作成できます。
- セッションのアイドル時間の制限。1つのセッションでのコール間の時間がアイドル制限時間に達すると、カレント・トランザクションがロールバックされてセッションは終了し、そのセッションのリソースはシステムに戻されます。次のコールは、ユーザーがインスタンスから切断されたことを示すエラーを受け取ります。この制限は、分単位の経過時間として設定します。

  ノート:

  セッションがアイドル時間の制限を超えたために終了すると、その少し後に、終了したセッションの後処理としてプロセス・モニター(PMON)・バックグラウンド・プロセスがクリーン・アップを実行します。PMON がこのプロセスを完了するまでは、終了したセッションも、セッションまたはユーザー・レベルのリソース制限に加算されます。

- セッション当たりの経過接続時間の制限。セッションの持続時間が経過制限時間を超えると、カレント・トランザクションがロールバックされてセッションが削除され、そのセッションのリソースがシステムに戻されます。この制限は、分単位の経過時間として設定します。

ノート:

Oracle Database は、経過アイドル時間や経過接続時間を絶えず監視しているわけではありません。絶えず監視した場合、システム・パフォーマンスが低下します。そのかわり数分ごとにチェックします。このため、Oracle Database がこの制限を規定してからセッションを終了させるまでの間に、セッションはこの制限をわずかに(5 分など)超える可能性があります。

- セッションのプライベート・システム・グローバル領域(SGA)(プライベートSQL領域に使用)の容量の制限。この制限が重要になるのは、共有サーバーの構成を使用するシステムの場合のみです。それ以外のシステムの場合、プライベートSQL領域はプログラム・グローバル領域(PGA)内にあります。この制限は、インスタンスのSGAに使用するメモリーのバイト数として設定します。KBまたはMBで指定するには、KまたはMの文字を使用します。

親トピック: システム・リソースのタイプと制限

2.4.3 プロファイルのリソース制限の値

プロファイルを作成しリソース制限を設定する前に、各リソース制限について適切な値を決定する必要があります。

リソース制限の値は、典型的なユーザーが実行する操作のタイプを基準として決定できます。たとえば、あるクラスのユーザーが通常は大量の論理データ・ブロック読取りを実行しない場合は、`ALTER RESOURCE COST` SQL文を使用して、`LOGICAL_READS_PER_SESSION`設定を控えめに設定します。

通常、ユーザー・プロファイルの適切なリソース制限値を決定するには、それぞれのタイプのリソースの使用状況について履歴情報を収集するのが最善です。たとえば、データベース管理者やセキュリティ管理者は、`AUDIT SESSION`句を使用して、`CONNECT_TIME`および`LOGICAL_READS_PER_SESSION`の制限値についての情報を収集できます。

Oracle Data Guard環境では、アクティブ・スタンバイ・データベースが読取り専用モードでオープンされます。これにより、プライマリ・データベースに対してと同じようにそれに対するユーザー接続が可能になります。したがって、指定されたユーザー・プロファイルのパスワード・リソース関連の制限はすべて、単独で機能します(スタンバイ・データベースでのユーザー・パスワード変更を意味するか必要とするものを除く)。このタスクは、読取り専用モードでオープンされているデータベースでは実行できません。

その他の制限値の統計情報は、Oracle Enterprise Manager(またはSQL*Plus)のモニター機能、特に統計モニターを使用して収集できます。

親トピック: ユーザー・リソース制限の構成

2.4.4 プロファイルによるリソースの管理

プロファイルとは、ユーザーのデータベース使用とインスタンス・リソースを制限する一連のリソース制限およびパスワード・パラメータに名前を付けたものです。

- プロファイルについて
  プロファイルとは、ユーザーに適用される属性の集合です。
- ora_stig_profileユーザー・プロファイル
  `ora_stig_profile`ユーザー・プロファイルは、セキュリティ技術導入ガイドに準拠するように設計されています。
- プロファイルの作成
  プロファイルには、パスワードの制限やリソースの制限など、特定のカテゴリに対する制限が含まれます。
- CDBプロファイルまたはアプリケーション・プロファイルの作成
  `CREATE PROFILE`または`ALTER PROFILE`文の`CONTAINER=ALL`句で、CDBまたはアプリケーション・ルート内

にプロファイルを作成できます。

- ユーザーへのプロファイルの割当て
  プロファイルの作成後、ユーザーにプロファイルを割り当てることができます。
- プロファイルの削除
  プロファイルは、現在ユーザーに割り当てられている場合でも削除できます。

親トピック: ユーザー・リソース制限の構成

2.4.4.1 プロファイルについて

プロファイルとは、ユーザーに適用される属性の集合です。

プロファイルは、これらの属性を複数のユーザーで共有する場合に参照される単一ポイントです。

profileを各ユーザーに割り当てる必要があります。各ユーザーに割り当てることができるプロファイルは1つのみで、新しいプロファイルを作成すると、以前の割当てと置き換えられます。

ユーザー・プロファイルを作成および管理できるのは、リソース制限がデータベースのセキュリティ・ポリシーの要件である場合のみです。プロファイルを使用するには、最初にデータベース内のユーザーの関連タイプを分類します。ロールを使用して関連ユーザーの権限を管理するのと同様に、プロファイルを使用して関連ユーザーのリソース制限を管理します。データベース内のすべてのカテゴリのユーザーを含めるために必要なプロファイル数を決定してから、プロファイルごとに適切なリソース制限を決定します。

Oracle Internet Directoryのユーザー・プロファイルには、各ユーザーのディレクトリ使用と認証に関連した属性が含まれています。同様に、Oracle Label Securityのプロファイルには、Oracle Label Securityのユーザー管理や操作管理に役立つ属性が含まれています。プロファイル属性にはシステム・リソースに関する制限を含めることができます。データベース・リソース・マネージャを使用すると、これらのタイプのリソース制限を設定できます。

マルチテナント環境では、プロファイルは、コンテナ・データベース(CDB)およびアプリケーション・コンテナのほか、それらが関連付けられたプラガブル・データベース(PDB)で行われる管理および操作に役立ちます。CDBとアプリケーション・コンテナのいずれの場合でも、共通プロファイルを定義すると、そのプロファイルはコンテナ全体に適用され、コンテナ外には適用されません。ローカル・プロファイルを作成した場合、プロファイルはそのPDBにのみ適用されます。

プロファイルのリソース制限が適用されるのは、対応するデータベースのリソース制限が使用可能な場合のみです。このリソース制限が使用可能になるのは、データベースの起動前(RESOURCE_LIMIT初期化パラメータを使用)またはオープン中(ALTER SYSTEM文を使用)のいずれかです。

パスワード・パラメータはプロファイル内にありますが、RESOURCE_LIMITまたはALTER SYSTEMの影響は受けず、パスワード管理は常に使用可能です。Oracle Databaseでは、主にデータベース・リソース・マネージャによって、リソースの割当てと制限が処理されます。

認可されたデータベース・ユーザーは、プロファイルの作成、ユーザーへの割当て、変更および削除を(CREATE USERまたはALTER USER文を使用して)随時実行できます。プロファイルは、ロールや他のプロファイルではなく、ユーザーにのみ割り当てることができます。プロファイルの割当ては現行のセッションには影響を与えず、それより後のセッションにのみ有効です。

現行のプロファイルに関する情報を検索するには、DBA_PROFILESビューを問い合せます。

関連項目:

リソース管理の詳細は、『Oracle Database管理者ガイド』を参照してください

親トピック: プロファイルによるリソースの管理

2.4.4.2 ora_stig_profileユーザー・プロファイル

ora_stig_profileユーザー・プロファイルは、セキュリティ技術導入ガイドに準拠するように設計されています。

ora_stig_profileユーザー・プロファイルは、パスワード複雑度ファンクション、ログインの最大失敗回数、最大再利用回数などの要件を求めるSTIG要件に対応します。このプロファイルの定義は次のとおりです。

```
CREATE PROFILE ora_stig_profile
  password_life_time         60
  password_grace_time        5
  password_reuse_time        365
  password_reuse_max         10
  failed_login_attempts      3
  password_lock_time         unlimited
  inactive_account_time      35
  idle_time                  15
  password_verify_function   ora12c_stig_verify_function;
```

親トピック: プロファイルによるリソースの管理

2.4.4.3 プロファイルの作成

プロファイルには、パスワードの制限やリソースの制限など、特定のカテゴリに対する制限が含まれます。

プロファイルを作成する場合、CREATE PROFILEシステム権限が必要です。すべての既存のプロファイルを検索するには、DBA_PROFILESビューを問い合せます。

- プロファイルを作成するには、CREATE PROFILE文を使用します。

たとえば、パスワードの制限を定義するプロファイルを作成するとします。

```
CREATE PROFILE password_prof LIMIT
  FAILED_LOGIN_ATTEMPTS 6
  PASSWORD_LIFE_TIME 60
  PASSWORD_REUSE_TIME 60
  PASSWORD_REUSE_MAX 5
  PASSWORD_LOCK_TIME 1/24
  PASSWORD_GRACE_TIME 10
  PASSWORD_VERIFY_FUNCTION DEFAULT;
```

次の例は、リソース制限プロファイルの作成方法を示しています。

```
CREATE PROFILE app_user LIMIT
  SESSIONS_PER_USER          UNLIMITED
  CPU_PER_SESSION            UNLIMITED
  CPU_PER_CALL               3500
  CONNECT_TIME               50
  LOGICAL_READS_PER_SESSION  DEFAULT
  LOGICAL_READS_PER_CALL     1200
  PRIVATE_SGA                20K
  COMPOSITE_LIMIT            7500000;
```

関連トピック

- Oracle Database SQL言語リファレンス

親トピック: プロファイルによるリソースの管理

2.4.4.4 CDBプロファイルまたはアプリケーション・プロファイルの作成

CREATE PROFILEまたはALTER PROFILE文のCONTAINER=ALL句で、CDBまたはアプリケーション・ルート内にプロファイルを作成できます。

ローカル・プロファイルは、CDBルートまたはアプリケーション・ルートでは作成できません。作成するプロファイルは、CDBルートまたはアプリケーション・ルートに関連付けられたすべてのPDBに適用されます。非マルチテナント環境と同様のパラメータを使用してプロファイルを作成します。

- CDBルートまたはアプリケーション・ルートでプロファイルを作成するには、CREATE PROFILEまたはALTER PROFILE文にオプションとしてCONTAINER=ALL句を含めます。

  CONTAINER=ALL句をオプションとするのは、文が処理される場合のデフォルトであるためです。

たとえば:

```
CREATE PROFILE password_prof LIMIT
  FAILED_LOGIN_ATTEMPTS 6
  PASSWORD_LIFE_TIME 60
  PASSWORD_REUSE_TIME 60
  PASSWORD_REUSE_MAX 5
  PASSWORD_LOCK_TIME 1/24
  PASSWORD_GRACE_TIME 10
  PASSWORD_VERIFY_FUNCTION DEFAULT
  CONTAINER=ALL;
```

親トピック: プロファイルによるリソースの管理

2.4.4.5 ユーザーへのプロファイルの割当て

プロファイルの作成後、ユーザーにプロファイルを割り当てることができます。

すでにプロファイルを割り当てられているユーザーにプロファイルを割り当てることはできますが、直近に割り当てられたプロファイルが優先されます。外部ユーザーまたはグローバル・ユーザーにプロファイルを割り当てる場合、パスワードのパラメータは、そのユーザーに対して有効ではありません。

ユーザーに現在割り当てられているプロファイルを検索するには、DBA_USERSビューを問い合せます。

- ALTER USER文を使用して、ユーザーにプロファイルを割り当てます。

たとえば:

```
ALTER USER psmith PROFILE app_user;
```

親トピック: プロファイルによるリソースの管理

2.4.4.6 プロファイルの削除

プロファイルは、現在ユーザーに割り当てられている場合でも削除できます。

プロファイルを削除しても、現在アクティブなセッションに影響はありません。プロファイルの削除後に作成されたセッションのみが、変更されたプロファイル割当てを使用します。プロファイルを削除するには、DROP PROFILEシステム権限が必要です。デフォルトのプロファイルは削除できません。

- プロファイルを削除するには、SQL文のDROP PROFILEを使用します。ユーザーに現在割り当てられているプロファイルを削除するには、CASCADEオプションを使用します。

たとえば:

```
DROP PROFILE clerk CASCADE;
```

削除するプロファイルに現在割り当てられているユーザーは、自動的にDEFAULTプロファイルに割り当てられます。DEFAULTプロファイルは削除できません。

関連トピック

- Oracle Database SQL言語リファレンス

親トピック: プロファイルによるリソースの管理

2.5 ユーザー・アカウントの削除

ユーザーのスキーマにオブジェクトがある場合、そのユーザーがセッション中でなければユーザー・アカウントを削除できます。

- ユーザー・アカウントの削除について
  ユーザー・アカウントを削除する前に、削除に適した権限があることを確認する必要があります。
- ユーザー・セッションの終了
  データベースに接続されているユーザーは削除できません。
- ユーザーがデータベースから切断した後のユーザーの削除について
  ユーザーがデータベースから切断した後に、DROP USER文を使用してユーザーを削除できます。
- スキーマにオブジェクトが含まれるユーザーの削除
  スキーマ内にオブジェクトを格納しているユーザーを削除する前に、スキーマ・オブジェクトを削除した場合の影響を徹底的に調べます。

親トピック: Oracle Databaseユーザーのセキュリティの管理

2.5.1 ユーザー・アカウントの削除について

ユーザー・アカウントを削除する前に、削除に適した権限があることを確認する必要があります。

いずれの環境でユーザー・アカウントを削除する場合でも、DROP USERシステム権限が必要です。マルチテナント環境で共通ユーザー・アカウントを削除するには、共通に付与されるDROP USERシステム権限が必要です。ローカル・ユーザー・アカウントを削除するには、ローカル・ユーザー・アカウントが存在するPDB内で、共通に付与されるDROP USER権限またはローカルに付与されるDROP USER権限が必要です。

ユーザー・アカウントを削除すると、Oracle Databaseではこのユーザー・アカウントおよび対応するスキーマがデータ・ディクショナリから削除されます。さらに、ユーザー・スキーマに含まれているすべてのスキーマ・オブジェクトも(存在する場合)削除されます。

ノート:

- ユーザーのスキーマとそれに対応するオブジェクトは残したままで、データベースへのアクセスを拒否する場合は、そのユーザーから CREATE SESSION 権限を取り消してください。
- SYS ユーザーまたは SYSTEM ユーザーは削除しないでください。これらのユーザーを削除すると、データベースが破損します。

親トピック: ユーザー・アカウントの削除

2.5.2 ユーザー・セッションの終了

データベースに接続されているユーザーは削除できません。

ユーザーを削除する前に、まずユーザー・セッションを終了する必要があります(または、ユーザーがセッションを終了できます)。

1. V$SESSION動的ビューを問い合せて、セッションを終了するユーザーのセッションIDを確認します。

   たとえば:

```
SELECT SID, SERIAL#, USERNAME FROM V$SESSION;
    SID         SERIAL# USERNAME
------- --------------- ----------------------
    127           55234  ANDY
...
```

2. ALTER SYSTEM SQL文を使用して、V$SESSIONビューのSID設定とSERIAL#設定に基づき、ユーザーのセッションを終了します。

   たとえば:

```
ALTER SYSTEM KILL SESSION '127, 55234';
```

親トピック: ユーザー・アカウントの削除

2.5.3 ユーザーがデータベースから切断した後のユーザーの削除について

ユーザーがデータベースから切断した後に、DROP USER文を使用してユーザーを削除できます。

ユーザーとそのユーザーのスキーマ・オブジェクト(ある場合)をすべて削除するには、DROP USERシステム権限が必要です。DROP USERシステム権限は強力な権限であるため、通常はセキュリティ管理者のみがこの権限を持ちます。

ユーザーのスキーマに依存型のスキーマ・オブジェクトが含まれている場合に、ユーザーと対応付けられているすべてのオブジェクト、およびそのユーザーの表に依存している外部キーをすべて削除するには、CASCADEオプションを使用します。CASCADEを指定していない場合は、ユーザーのスキーマに依存型のオブジェクトが含まれていると、エラー・メッセージが戻され、ユーザーは削除されません。

親トピック: ユーザー・アカウントの削除

2.5.4 スキーマにオブジェクトが含まれるユーザーの削除

スキーマ内にオブジェクトを格納しているユーザーを削除する前に、スキーマ・オブジェクトを削除した場合の影響を徹底的に調べます。

1. DBA_OBJECTSデータ・ディクショナリ・ビューを問い合せて、ユーザーが所有するオブジェクトを確認します。

   たとえば:

```
SELECT OWNER, OBJECT_NAME FROM DBA_OBJECTS WHERE OWNER LIKE 'ANDY';
```

   ユーザー名を大文字で入力します。事前に認識できない連鎖的な影響に注意してください。たとえば、表を所有するユーザーを削除する場合は、ビューまたはプロシージャがその表に依存していないかどうかを確認してください。

2. ユーザーとすべての関連オブジェクト、およびそのユーザーが所有する表に依存する外部キーを削除するには、DROP USER SQL文とCASCADE句を使用します。

   たとえば:

```
DROP USER andy CASCADE;
```

親トピック: ユーザー・アカウントの削除

2.6 Oracle Databaseから提供される事前定義済のスキーマ・ユーザー・アカウント

Oracle Databaseのインストール・プロセスでは、事前定義された管理アカウント、非管理ユーザー・アカウント、およびサンプル・スキーマ・ユーザー・アカウントがデータベースに作成されます。

- 事前定義済のスキーマ・ユーザー・アカウントについて
  事前定義済のスキーマ・アカウントは、標準のOracleスクリプトを実行するか、架空の会社を表すアカウントである場合に自動的に作成されます。
- 事前定義済の管理アカウント
  デフォルトのOracle Databaseインストールでは、監査などのよく使用される機能を管理するための事前定義済の管理アカウントが提供されます。
- 事前定義済の非管理ユーザー・アカウント
  デフォルトのOracle Databaseインストールでは、Oracle Spatialなどの機能を管理するための非管理ユーザー・アカウントが提供されます。
- 事前定義されたサンプル・スキーマ・ユーザー・アカウント
  Oracle Databaseでは、サンプル・スキーマをインストールした場合にサンプル・ユーザー・アカウントのセットが作成されます。

親トピック: Oracle Databaseユーザーのセキュリティの管理

2.6.1 事前定義済のスキーマ・ユーザー・アカウントについて

事前定義済のスキーマ・アカウントは、標準Oracleスクリプトの実行時に自動的に作成されるか、架空の会社を表すアカウントです。

事前定義済のスキーマ・アカウントは、次の2つのカテゴリに分類されます。

- 事前定義済の管理および非管理スキーマ・アカウントは、様々な`cat.*sql`スクリプトなどの標準スクリプトを実行すると自動的に作成されます。これらのアカウントは、`ALL_USERS`データ・ディクショナリ・ビューの`USERNAME`列および`ORACLE_MAINTAINED`列を問い合せることで検索できます。`ORACLE_MAINTAINED`の出力がYの場合、そのユーザー・アカウントは、アカウントの作成に使用したスクリプトを実行するという方法以外で変更しないでください。
- HRサンプル・スキーマ・ユーザー・アカウントはデフォルトでインストールされます。一連の追加のスキーマ・ユーザー・アカウント(`OE`、`PM`、`IX`および`SH`と`HR`)がGitHubで使用できます。これらのスキーマ・アカウントは、様々な製品を製造する架空の会社の異なる部門を表します。これらのアカウントのステータスは、`DBA_USERS`データ・ディクショナリ・ビューを問い合せることで確認できます。これらのアカウントに対する`ORACLE_MAINTAINED`列の出力はNになるため、これらのアカウントの作成に使用したスクリプトを再実行することなくアカウントの変更が可能です。

デフォルトでは、これらのアカウントのほとんどがスキーマ限定アカウントとして認証されます。ただし、サンプル・スキーマ・アカウントを除いて、データベース・インストール・プロセス中にロックされて期限切れになります。これらのアカウントを使用する場合、他の方法(パスワード認証など)で認証するように構成できますが、セキュリティを向上させるためにこれらのアカウントをスキーマ限定アカウントとして維持することをお薦めします。

関連トピック

- Oracle Databaseサンプル・スキーマ
- スキーマ限定アカウント

親トピック: Oracle Databaseで提供される事前定義済のスキーマ・ユーザー・アカウント

2.6.2 事前定義済の管理アカウント

デフォルトのOracle Databaseインストールでは、監査などのよく使用される機能を管理するための事前定義された管理アカウントが提供されます。

これらのアカウントには、SYSスキーマが所有するパッケージのEXECUTE権限、CREATE ANY TABLE権限、またはALTER SESSIONのような、データベースの領域の管理に必要となる特別な権限を持ちます。管理アカウントのデフォルトの表領域は、SYSTEMかSYSAUXです。マルチテナント環境では、事前定義された管理アカウントはルート・データベースにあります。

これらのアカウントを無許可アクセスから保護するため、インストール・プロセスにより、次の表に示されたアカウントを除くほとんどのアカウントが期限切れにされ、ロックされます。データベース管理者には、これらのアカウントのロック解除とリセットを行う責任があります。

表2-1に、事前定義済の管理ユーザー・アカウントを示します。このアカウントは、標準スクリプト(各種のcat*.sqlスクリプトなど)の実行時に、Oracleデータベースによって自動的に作成されます。Oracleによって作成および維持されるユーザー・アカウントの完全なリストは、ALL_USERSデータ・ディクショナリ・ビューのUSERNAME列とORACLE_MAINTAINED列を問い合せることで見つかります。ORACLE_MAINTAINEDの出力がYの場合、そのユーザー・アカウントは、アカウントの作成に使用したスクリプトを実行するという方法以外で変更しないでください。

オープン、ロック、または期限切れなど、アカウントのステータスを確認するには、DBA_USERSデータ・ディクショナリ・ビューのACCOUNT_STATUS列を問い合せます。アカウントがスキーマ限定の場合、ステータスはNONEです。

表2-1 事前定義されたOracle Databaseの管理ユーザー・アカウント

ユーザー・アカウント	説明
ANONYMOUS	Oracle XML DB に HTTP アクセス可能なアカウント。EPG (Embedded PL/SQL Gateway)をデータベースにインストールするときに APEX_PUBLIC_USER アカウントのかわりに使用されます。 EPG は Oracle Database とともに使用される Web サーバーです。動的アプリケーションの作成に必要なインフラストラクチャを提供します。
APPQOSSYS	Oracle Quality of Service Management で必要なすべてのデータおよびメタデータの格納および管理に使用されます。
AUDSYS	統合監査機能によって統合監査証跡レコードの格納に使用される内部アカウント。 監査レコードが作成されるときと場所を参照してください。
CTXSYS	Oracle Text を管理するためのアカウント。Oracle Text を使用すると、テキスト問合せアプリケーションおよびドキュメント分類アプリケーションを作成できます。Oracle Text は、テキスト用の索引付け、語とテーマの検索および表示機能を提供します。 Oracle Text アプリケーション開発者ガイドを参照してください。
DBSNMP	Oracle Enterprise Manager の管理エージェント・コンポーネントによりデータベースの監視およ

ユーザー・アカウント	説明
	び管理に使用されるアカウント。 『Enterprise Manager Cloud Control 管理者ガイド』を参照してください。
DBSFWUSER	DBMS_SFW_ACL_ADMIN パッケージの実行に使用されるアカウント。 『Oracle Database PL/SQL パッケージ・プロシージャおよびタイプ・リファレンス』を参照してください。
DVF	Oracle Database Vault が所有するアカウントで、Database Vault のファクタ値を取得するためのパブリック・ファンクションが含まれます。 『Oracle Database Vault 管理者ガイド』を参照してください
DVSYS	DV_OWNER(管理構成用)ロールと DV_ACCTMGR ロール(アカウント管理用)に関連付けられている Oracle Database Vault アカウント。 『Oracle Database Vault 管理者ガイド』を参照してください
GGSYS	Oracle GoldenGate が使用する内部アカウント。ロック解除したり、データベース・ログインに使用したりしないでください。 『Oracle Database Global Data Services 概念および管理ガイド』を参照してください
GSMADMIN_INTERNAL	Global Data Services スキーマを所有する内部アカウント。ロック解除したり、データベース・ログインに使用したりしないでください。 『Oracle Database Global Data Services 概念および管理ガイド』を参照してください
GSMCATUSER	グローバル・サービス・マネージャが Global Data Services カタログへの接続に使用するアカウント。 『Oracle Database Global Data Services 概念および管理ガイド』を参照してください
GSMROOTUSER	シャーディング構成内の CDB の CDB$ROOT にログインするために使用されるアカウント。このユーザーは GDS 構成では使用されません。CDB 内の CDB$ROOT へのすべての接続で、GSMROOTUSER が使用されます。
GSMUSER	グローバル・サービス・マネージャがデータベースへの接続に使用するアカウント。 『Oracle Database Global Data Services 概念および管理ガイド』を参照してください

ユーザー・アカウント	説明
LBACSYS	Oracle Label Security (OLS)を管理するためのアカウント。Label Security オプションをインストールするときのみ作成されます。 Oracle Label Security 管理者ガイドを参照してください。
MDSYS	Oracle Spatial および Oracle Multimedia Locator の管理者アカウント。 『Oracle Spatial and Graph 開発者ガイド』を参照してください。
OJVMSYS	Oracle JVM サポートによる Java Naming and Directory Interface (JNDI)のサポートで使用されるアカウント。このアカウントは、JVM オブジェクトに関する詳細(ネームスペース・メタデータ、バインドされた名前、属性、権限およびストアド・オブジェクト表現)を格納するデータベース表を所有します。 『Oracle Database Java 開発者ガイド』を参照してください。
OLAPSYS	OLAP カタログ(CWMLite)を所有するアカウント。このアカウントは、非推奨となりましたが、下位互換性のために保持されています。
ORDDATA	このアカウントには、Oracle Multimedia DICOM データ・モデルが含まれます。詳細は、『Oracle Multimedia DICOM 開発者ガイド』を参照してください。
ORDPLUGINS	Oracle Multimedia ユーザー。Oracle およびサード・パーティにより提供されたプラグイン(フォーマット・プラグイン)はこのスキーマにインストールされています。 Oracle Multimedia により Oracle Database で画像、音声、動画、DICOM フォーマットの医療用画像などのオブジェクトや、その他の企業情報と統合された異機種間のメディア・データを格納、管理および取得できます。 『Oracle Multimedia ユーザーズ・ガイド』 を参照してください。
ORDSYS	Oracle Multimedia 管理者アカウント。 『Oracle Multimedia ユーザーズ・ガイド』 を参照してください。
OUTLN	プラン・スタビリティをサポートするアカウント。プラン・スタビリティは、同じ SQL 文の同じ実行計画の保守を可能にします。OUTLN は、ストアド・アウトラインに関連付けられたメタデータを集中管理するロールを実行します。
REMOTE_SCHEDULER_AGENT	データベースのリモート・ジョブを無効化するアカウント。このアカウントはリモート・スケジューラ・エージェントの構成時に作成されます。データベースでリモート・ジョブを実行する機能は、このユーザーを

ユーザー・アカウント	説明
	削除することで、無効(使用禁止)にできます。 Oracle Database 管理者ガイドを参照してください。
SI_INFORMTN_SCHEMA	SQL/MM Still Image Standard の情報ビューを格納するアカウント。 『Oracle Multimedia ユーザーズ・ガイド』 を参照してください。 ノート: SI_INFORMTN_SCHEMA アカウントは、Oracle Database 12c リリース 2 (12.2)では非推奨です。
SYS	データベース管理タスクの実行に使用されるアカウント。 Oracle Database 2 日でデータベース管理者を参照してください。
SYS$UMF	リモートの自動ワークロード・リポジトリ(AWR)などのリモート管理フレームワークの管理に使用されるアカウント。 『Oracle Database パフォーマンス・チューニング・ガイド』を参照してください。
SYSBACKUP	Oracle Recovery Manager のリカバリとバックアップの実行に使用されるアカウント。 『Oracle Database バックアップおよびリカバリ・ユーザーズ・ガイド』を参照してください。
SYSDG	Oracle Data Guard の操作の実行に使用されるアカウント。 Oracle Data Guard 概要および管理を参照してください。
SYSKM	透過的データ暗号化の管理に使用される内部アカウント。 『Oracle Database Advanced Security ガイド』を参照してください。
SYSRAC	Oracle Real Application Clusters の管理に使用するアカウント。 詳細は、Oracle Real Application Clusters 管理およびデプロイメント・ガイドを参照してください。
SYSTEM	Oracle Database のデフォルトの汎用データベース管理者アカウント。 本番システムでは、データベース管理操作用に汎用 SYSTEM アカウントを使用せずに、個々のデータベース管理者アカウントを作成することをお薦めします。

ユーザー・アカウント	説明
	Oracle Database 2 日でデータベース管理者を参照してください。
WMSYS	Oracle Workspace Manager 用のメタデータ情報の格納に使用されるアカウント。 Oracle Database Workspace Manager 開発者ガイドを参照してください。
XDB	Oracle XML DB のデータおよびメタデータの格納に使用されるアカウント。セキュリティを高めるには、XDB ユーザー・アカウントのロックを解除しないでください。 Oracle XML DB は Oracle Database のデータに対し、パフォーマンスの高い XML の格納および取得を提供します。 Oracle XML DB 開発者ガイドを参照してください。

ノート:

Oracle 自動ストレージ管理(Oracle ASM)インスタンスを作成すると、ASMSNMP アカウントが作成されます。Oracle Enterprise Manager はこのアカウントを使用して、ASM インスタンスを監視し、ASM 関連のデータ・ディクショナリ・ビューからデータを取得します。ASMSNMP アカウントのステータスはアカウントの作成時に OPEN に設定され、SYSDBA 管理権限が付与されます。

親トピック: Oracle Databaseで提供される事前定義済のスキーマ・ユーザー・アカウント

2.6.3 事前定義された非管理ユーザー・アカウント

デフォルトのOracle Databaseインストールでは、Oracle Spatialなどの機能を管理するための非管理ユーザー・アカウントが提供されます。

表2-2に、事前定義済の非管理ユーザー・アカウントを示します。このアカウントは、標準スクリプト(各種の`cat*.sql`スクリプトなど)の実行時に、Oracleデータベースによって自動的に作成されます。Oracleによって作成および維持されるユーザー・アカウントの完全なリストは、`ALL_USERS`データ・ディクショナリ・ビューの`USERNAME`列と`ORACLE_MAINTAINED`列を問い合せることで見つかります。`ORACLE_MAINTAINED`の出力がYの場合、そのユーザー・アカウントは、アカウントの作成に使用したスクリプトを実行するという方法以外で変更しないでください。

非管理ユーザー・アカウントはジョブの実行に最低限必要な権限のみ所有します。デフォルトの表領域は`USERS`です。マルチテナント環境では、事前定義された非管理アカウントはルート・データベースに存在します

これらのアカウントを無許可アクセスから保護するため、インストール・プロセスにより、次の表に示されたアカウントを除くほとんどのアカウントがインストール後すぐにロックされ、期限切れになります。データベース管理者には、これらのアカウントのロック解除とリセットを行う責任があります。

オープン、ロック、または期限切れなど、アカウントのステータスを確認するには、`DBA_USERS`データ・ディクショナリ・ビューの`ACCOUNT_STATUS`列を問い合せます。アカウントがスキーマ限定の場合、ステータスは`NONE`です。

表2-2 事前定義済のOracle Databaseの非管理ユーザー・アカウント

ユーザー・アカウント	説明
DIP	Oracle Label Security とともにインストールされる Oracle Directory Integration and Provisioning(DIP)のアカウント。このプロファイルは、Oracle Internet Directory 対応 Oracle Label Security のインストール・プロセスの一部として自動的に作成されます。 Oracle Label Security 管理者ガイドを参照してください。
MDDATA	格納されるジオコーダおよびルーターのデータ用に Oracle Spatial に使用されるスキーマ。 Oracle Spatial は SQL スキーマおよびファンクションを提供し、これにより Oracle Database の Spatial 機能の格納、取得、更新、問合せができます。 『Oracle Spatial and Graph 開発者ガイド』を参照してください。
ORACLE_OCM	Oracle Configuration Manager と使用するアカウント。この機能により現在の Oracle Database インスタンスの構成情報を My Oracle Support と関連付けることができます。サービス・リクエストを記録すると、データベース・インスタンスの構成情報と関連付けられます。 使用しているプラットフォームの『Oracle Database インストレーション・ガイド』を参照してください。
XS$NULL	セッションにデータベース・ユーザーが存在せず、実際のセッション・ユーザーは、Oracle Real Application Security でサポートされるアプリケーション・ユーザーであることを示す内部アカウント。XS$NULL には権限はなく、データベース・オブジェクトを所有しません。XS$NULL として認証されることはなく、認証資格証明に XS$NULL が割り当てられることもありません。

親トピック: Oracle Databaseで提供される事前定義済のスキーマ・ユーザー・アカウント

2.6.4 事前定義されたサンプル・スキーマ・ユーザー・アカウント

Oracle Databaseでは、サンプル・スキーマをインストールした場合にサンプル・ユーザー・アカウントのセットが作成されます。

サンプル・スキーマ・ユーザー・アカウントはすべて非管理アカウントで、表領域は USERS です。

これらのアカウントを無許可アクセスから保護するため、インストール・プロセスによりこれらのアカウントがインストール直後にロックされ、期限切れになります。データベース管理者には、これらのアカウントのロック解除とリセットを行う責任があります。

表2-3には、様々な製品を製造している架空の企業の個別の部門を表すサンプル・スキーマ・ユーザー・アカウントがリストされています。これらのアカウントのステータスは、DBA_USERSデータ・ディクショナリ・ビューを問い合せることで確認できます。これらのアカウントに対するORACLE_MAINTAINED列の出力はNになるため、これらのアカウントの作成に使用したスクリプトを再実行することなくアカウントの変更が可能です。

オープン、ロック、または期限切れなど、アカウントのステータスを確認するには、DBA_USERSデータ・ディクショナリ・ビューのACCOUNT_STATUS列を問い合せます。アカウントがスキーマ限定の場合、ステータスはNONEです。

表2-3 デフォルトのサンプル・スキーマ・ユーザー・アカウント

ユーザー・アカウント	説明
HR	HR(Human Resources)スキーマを管理するためのアカウント。このスキーマには企業の従業員および施設に関する情報が格納されます。
OE	OE(Order Entry)スキーマを管理するためのアカウント。このスキーマには製品のインベントリや、様々なチャネルによる製品の売上が格納されます。
PM	PM(Product Media)スキーマを管理するためのアカウント。このスキーマには企業が販売した各製品の説明と詳細情報が含まれます。
IX	IX(Information Exchange)スキーマを管理するためのアカウント。このスキーマにより B2B(Business-to-Business)アプリケーションを介した発送が管理されます。
SH	SH(Sales)スキーマを管理するためのアカウント。このスキーマにはビジネス上の決断を容易にするビジネス戦略が格納されます。

例にあるスキーマ・アカウントに加えて、Oracle Databaseでは他のスキーマ・アカウントであるSCOTTの例も用意しています。SCOTTスキーマには、表EMP、DEPT、SALGRADEおよびBONUSが含まれています。SCOTTアカウントはOracle Databaseのドキュメント・セット全体の例で使用されます。Oracle Databaseをインストールすると、SCOTTアカウントはロックされ、期限が切れます。

関連トピック

- Oracle Databaseサンプル・スキーマ

親トピック: Oracle Databaseで提供される事前定義済のスキーマ・ユーザー・アカウント

2.7 データベース・ユーザーおよびプロファイルのデータ・ディクショナリ・ビュー

Oracle Databaseには、ユーザーとプロファイルの作成に使用した設定の情報を提供する、一連のデータ・ディクショナリ・ビューがあります。

- ユーザーとプロファイルに関する情報を表示するデータ・ディクショナリ・ビュー
  Oracle Databaseには、データベース・ユーザーおよびプロファイルに関する情報を含むデータ・ディクショナリ・ビュー一式が用意されています。
- すべてのユーザーと関連情報を検索する問合せ
  DBA_USERSデータ・ディクショナリ・ビューには、データベースで定義されているすべてのユーザーとその関連情報が表示されます。
- すべての表領域の割当て制限を表示する問合せ
  DBA_TS_QUOTASデータ・ディクショナリ・ビューには、各ユーザーに割り当てられているすべての表領域割当てが表示されます。
- すべてのプロファイルと割り当てられている制限を表示する問合せ
  DBA_PROFILEビューには、データベース内のすべてのプロファイル、および各プロファイルの制限ごとの関連設定がリストされます。

- 各ユーザー・セッションのメモリー使用量を表示する問合せ
  V$SESSION動的ビューには、各ユーザー・セッションのメモリー使用量が表示されます。

親トピック: Oracle Databaseユーザーのセキュリティの管理

2.7.1 ユーザーとプロファイルに関する情報を表示するデータ・ディクショナリ・ビュー

Oracle Databaseには、データベース・ユーザーおよびプロファイルに関する情報を含むデータ・ディクショナリ・ビュー一式が用意されています。

表2-4に、これらのデータ・ディクショナリ・ビューを示します。

表2-4 ユーザーとプロファイルに関する情報を表示するデータ・ディクショナリ・ビュー

ビュー	説明
ALL_OBJECTS	現行ユーザーがアクセス可能なすべてのオブジェクトが表示されます。
ALL_USERS	現行ユーザーに対して表示可能なユーザーがリストされますが、それらの記述は表示されません。
DBA_PROFILES	すべてのプロファイルとそれぞれの制限が表示されます。
DBA_TS_QUOTAS	ユーザーの表領域割当て制限が表示されます。
DBA_OBJECTS	データベース内のすべてのオブジェクトが表示されます。
DBA_USERS	データベースのすべてのユーザーの記述が表示されます。
DBA_USERS_WITH_DEFPWD	デフォルト・パスワードが設定されているすべてのユーザー・アカウントがリストされます。
PROXY_USERS	他のユーザーの識別情報を引き継ぐことができるユーザーの記述が表示されます。
RESOURCE_COST	セッション当たりの CPU、セッション当たりの読取り、接続時間および SGA の観点から各リソースのコストがリストされます。
USER_PASSWORD_LIMITS	ユーザーに割り当てられているパスワード・プロファイル・パラメータが表示されます。
USER_RESOURCE_LIMITS	現行ユーザーのリソース制限が表示されます。
USER_TS_QUOTAS	ユーザーの表領域割当て制限が表示されます。
USER_OBJECTS	現行ユーザーが所有するすべてのオブジェクトが表示されます。

ビュー	説明
USER_USERS	現行ユーザーの記述のみが表示されます。
V$SESSION	現在のデータベース・セッションのセッション情報がリストされます。
V$SESSTAT	ユーザー・セッションの統計が表示されます。
V$STATNAME	V$SESSTAT ビューに表示される統計のデコードされた統計名が表示されます。

次の各項では、これらのビューの使用例を示します。各例では、次の文がすでに実行されていることを前提としています。ユーザーはすべてローカル・ユーザーです。

```
CREATE PROFILE clerk LIMIT
 SESSIONS_PER_USER 1
 IDLE_TIME 30
 CONNECT_TIME 600;
CREATE USER jfee
 IDENTIFIED BY password
 DEFAULT TABLESPACE example
 TEMPORARY TABLESPACE temp
 QUOTA 500K ON example
 PROFILE clerk
 CONTAINER = CURRENT;
CREATE USER dcranney
 IDENTIFIED BY password
 DEFAULT TABLESPACE example
 TEMPORARY TABLESPACE temp
 QUOTA unlimited ON example
 CONTAINER = CURRENT;
CREATE USER userscott
 IDENTIFIED BY password
 CONTAINER = CURRENT;
```

関連トピック

- Oracle Databaseリファレンス

親トピック: データベース・ユーザーおよびプロファイルのデータ・ディクショナリ・ビュー

2.7.2 すべてのユーザーと関連情報を検索する問合せ

DBA_USERSデータ・ディクショナリ・ビューには、データベースで定義されているすべてのユーザーとその関連情報が表示されます。

DBA_USERSビューの詳細は、『Oracle Databaseリファレンス』を参照してください。

たとえば:

```
col username format a11
col profile format a10
col account_status format a19
col authentication_type format a29
SELECT USERNAME, PROFILE, ACCOUNT_STATUS, AUTHENTICATION_TYPE FROM DBA_USERS;

USERNAME        PROFILE         ACCOUNT_STATUS   AUTHENTICATION_TYPE
--------------- --------------- ---------------  -------------------
SYS             DEFAULT         OPEN             PASSWORD
SYSTEM          DEFAULT         OPEN             PASSWORD
```

```
USERSCOTT       DEFAULT         OPEN            PASSWORD
JFEE            CLERK           OPEN            GLOBAL
DCRANNEY        DEFAULT         OPEN            EXTERNAL
```

親トピック: データベース・ユーザーおよびプロファイルのデータ・ディクショナリ・ビュー

2.7.3 すべての表領域の割当て制限を表示する問合せ

DBA_TS_QUOTASデータ・ディクショナリ・ビューには、各ユーザーに割り当てられているすべての表領域割当てが表示されます。

このビューの詳細は、『Oracle Databaseリファレンス』を参照してください。

たとえば:

```
SELECT * FROM DBA_TS_QUOTAS;
TABLESPACE    USERNAME    BYTES     MAX_BYTES    BLOCKS    MAX_BLOCKS
----------    ---------  --------   ----------   -------   ----------
EXAMPLE       JFEE               0       512000         0          250
EXAMPLE       DCRANNEY           0           -1         0           -1
```

固有の割当て制限が割り当てられている場合は、正確な数値がMAX_BYTES列に示されます。この数値は常にデータベース・ブロック・サイズの倍数となるため、倍数でない表領域割当て制限を指定すると、適切な値に切り上げられます。無制限割当ての場合は、-1が表示されます。

親トピック: データベース・ユーザーおよびプロファイルのデータ・ディクショナリ・ビュー

2.7.4 すべてのプロファイルと割り当てられている制限を表示する問合せ

DBA_PROFILEビューには、データベース内のすべてのプロファイル、および各プロファイルの制限ごとの関連設定がリストされます。

このビューの詳細は、『Oracle Databaseリファレンス』を参照してください。

たとえば:

```
SELECT * FROM DBA_PROFILES
   ORDER BY PROFILE;
PROFILE              RESOURCE_NAME              RESOURCE_TYPE   LIMIT
-----------------    ------------------------   -------------  --------------
CLERK                COMPOSITE_LIMIT            KERNEL         DEFAULT
CLERK                FAILED_LOGIN_ATTEMPTS      PASSWORD       DEFAULT
CLERK                PASSWORD_LIFE_TIME         PASSWORD       DEFAULT
CLERK                PASSWORD_REUSE_TIME        PASSWORD       DEFAULT
CLERK                PASSWORD_REUSE_MAX         PASSWORD       DEFAULT
CLERK                PASSWORD_VERIFY_FUNCTION   PASSWORD       DEFAULT
CLERK                PASSWORD_LOCK_TIME         PASSWORD       DEFAULT
CLERK                PASSWORD_GRACE_TIME        PASSWORD       DEFAULT
CLERK                PRIVATE_SGA                KERNEL         DEFAULT
CLERK                CONNECT_TIME               KERNEL         600
CLERK                IDLE_TIME                  KERNEL         30
CLERK                LOGICAL_READS_PER_CALL     KERNEL         DEFAULT
CLERK                LOGICAL_READS_PER_SESSION  KERNEL         DEFAULT
CLERK                CPU_PER_CALL               KERNEL         DEFAULT
CLERK                CPU_PER_SESSION            KERNEL         DEFAULT
CLERK                SESSIONS_PER_USER          KERNEL         1
DEFAULT              COMPOSITE_LIMIT            KERNEL         UNLIMITED
DEFAULT              PRIVATE_SGA                KERNEL         UNLIMITED
DEFAULT              SESSIONS_PER_USER          KERNEL         UNLIMITED
DEFAULT              CPU_PER_CALL               KERNEL         UNLIMITED
DEFAULT              LOGICAL_READS_PER_CALL     KERNEL         UNLIMITED
DEFAULT              CONNECT_TIME               KERNEL         UNLIMITED
DEFAULT              IDLE_TIME                  KERNEL         UNLIMITED
```

```
DEFAULT             LOGICAL_READS_PER_SESSION  KERNEL          UNLIMITED
DEFAULT             CPU_PER_SESSION            KERNEL          UNLIMITED
DEFAULT             FAILED_LOGIN_ATTEMPTS      PASSWORD        10
DEFAULT             PASSWORD_LIFE_TIME         PASSWORD        180
DEFAULT             PASSWORD_REUSE_MAX         PASSWORD        UNLIMITED
DEFAULT             PASSWORD_LOCK_TIME         PASSWORD        1
DEFAULT             PASSWORD_GRACE_TIME        PASSWORD        7
DEFAULT             PASSWORD_VERIFY_FUNCTION   PASSWORD        UNLIMITED
DEFAULT             PASSWORD_REUSE_TIME        PASSWORD        UNLIMITED
32 rows selected.
```

デフォルト・プロファイル値を検索するには、次の問合せを実行します。

```
SELECT * FROM DBA_PROFILES WHERE PROFILE = 'DEFAULT';
PROFILE             RESOURCE_NAME              RESOURCE_TYPE  LIMIT
-----------------   -------------------------  -------------  --------------
DEFAULT             COMPOSITE_LIMIT            KERNEL         UNLIMITED
DEFAULT             SESSIONS_PER_USER          KERNEL         UNLIMITED
DEFAULT             CPU_PER_SESSION            KERNEL         UNLIMITED
DEFAULT             CPU_PER_CALL               KERNEL         UNLIMITED
DEFAULT             LOGICAL_READS_PER_SESSION  KERNEL         UNLIMITED
DEFAULT             LOGICAL_READS_PER_CALL     KERNEL         UNLIMITED
DEFAULT             IDLE_TIME                  KERNEL         UNLIMITED
DEFAULT             CONNECT_TIME               KERNEL         UNLIMITED
DEFAULT             PRIVATE_SGA                KERNEL         UNLIMITED
DEFAULT             FAILED_LOGIN_ATTEMPTS      PASSWORD       10
DEFAULT             PASSWORD_LIFE_TIME         PASSWORD       180
DEFAULT             PASSWORD_REUSE_TIME        PASSWORD       UNLIMITED
DEFAULT             PASSWORD_REUSE_MAX         PASSWORD       UNLIMITED
DEFAULT             PASSWORD_VERIFY_FUNCTION   PASSWORD       NULL
DEFAULT             PASSWORD_LOCK_TIME         PASSWORD       1
DEFAULT             PASSWORD_GRACE_TIME        PASSWORD       7
16 rows selected.
```

親トピック: データベース・ユーザーおよびプロファイルのデータ・ディクショナリ・ビュー

2.7.5 各ユーザー・セッションのメモリー使用量を表示する問合せ

V$SESSION動的ビューには、各ユーザー・セッションのメモリー使用量が表示されます。

このビューの詳細は、『Oracle Databaseリファレンス』を参照してください。

次の問合せを実行するとすべてのカレント・セッションがリストされ、各セッションのOracle Databaseユーザーと現在のユーザー・グローバル領域(UGA)メモリー使用が示されます。

```
SELECT USERNAME, VALUE || 'bytes' "Current UGA memory"
   FROM V$SESSION sess, V$SESSTAT stat, V$STATNAME name
WHERE sess.SID = stat.SID
   AND stat.STATISTIC# = name.STATISTIC#
   AND name.NAME = 'session uga memory';
USERNAME                       Current UGA memory
------------------------------ ----------------------------------------------
                               18636bytes
                               17464bytes
                               19180bytes
                               18364bytes
                               39384bytes
                               35292bytes
                               17696bytes
                               15868bytes
USERSCOTT                      42244bytes
SYS                            98196bytes
SYSTEM                         30648bytes
11 rows selected.
```

インスタンスの起動以降、各セッションに割り当てられた最大のUGAメモリーを表示するには、この問合せの'session uga memory'を'session uga memory max'に置き換えてください。

親トピック: データベース・ユーザーおよびプロファイルのデータ・ディクショナリ・ビュー

3 認証の構成

認証とは、データベースに接続するユーザーや他のエンティティを認証することです。

- 認証について
  認証とは、データ、リソースまたはアプリケーションの使用を希望するユーザーやデバイスなどのエンティティの身元を検証することです。
- パスワード保護の構成
  ユーザーのパスワードは様々な方法で保護できます。たとえば、パスワードの作成要件の制御、パスワード管理ポリシーの使用などの方法があります。
- データベース管理者の認証
  データベース管理者を認証するには、強力な認証を使用するか、オペレーティング・システムから行うか、パスワードを使用してデータベースから行います。
- ユーザーのデータベース認証
  ユーザーのデータベース認証では、認証を実行するためにデータベース自体の情報を使用する必要があります。
- スキーマ限定アカウント
  スキーマ限定アカウントを作成できます。つまり、スキーマ・ユーザーにパスワードはありません。
- ユーザーのオペレーティング・システム認証
  Oracle Databaseではオペレーティング・システムで管理されている情報を使用した認証が可能です。
- ユーザーのネットワーク認証
  ネットワークでのユーザーの認証は、サード・パーティ・サービスでTransport Layer Securityを使用して行うことができます。
- PDBのオペレーティング・システム・ユーザーの構成
  `DBMS_CREDENTIAL.CREATE_CREDENTIAL`プロシージャで、ユーザー・アカウントをPDBのオペレーティング・システム・ユーザーに設定します。
- ユーザーのグローバル認証とグローバル認可
  ユーザーのグローバル認証とグローバル認可を使用すると、ユーザー関連情報を一元管理できます。
- ユーザーとパスワード認証のための外部サービスの構成
  外部サービス(オペレーティング・システムまたはネットワーク)でパスワードを管理し、ユーザーを認証できます。
- 複数層の認証と認可
  中間層アプリケーションを保護するために、Oracle Databaseは権限を制限し、すべての層のクライアントの識別情報を保持し、クライアントによるアクションを監査します。
- クライアント、アプリケーション・サーバーおよびデータベース・サーバーの管理とセキュリティ
  複数層環境では、アプリケーション・サーバーはクライアントにデータを提供し、1つ以上のデータベース・サーバーとのインタフェースとして機能します。
- 複数層環境でのユーザー識別情報の保持
  中間層サーバーを使用してプロキシ認証を行ったり、クライアント識別子を使用してデータベースが認識しないアプリケーション・ユーザーを識別したりできます。
- ユーザー認証のデータ・ディクショナリ・ビュー
  Oracle Databaseには、ユーザーのロールや使用しているプロファイルなど、ユーザー認証に関する情報を表示するデータ・ディクショナリ・ビューが用意されています。

親トピック: ユーザー認証および認可の管理

3.1 認証について

認証とは、データ、リソースまたはアプリケーションの使用を希望するユーザーやデバイスなどのエンティティの身元を検証することです。

アイデンティティを検証することで、その後の対話に関する信頼関係が確立されます。また、認証により、アクセスとアクションを特定のアイデンティティにリンクでき、アカウンタビリティが有効化されます。認証後は、認可プロセスでそのエンティティが実行できるアクセスとアクションのレベルを許可または制限できます。

Oracle Databaseのデータベース・ユーザーと非データベース・ユーザーの両方を認証できます。簡潔性を考慮して、すべてのデータベース・ユーザーに同じ認証方式を使用するのが一般的ですが、Oracle Databaseでは1つのデータベース・インスタンスで一部またはすべての方法を使用できます。Oracle Databaseでは、データベース管理者が特別なデータベース操作を実行するため、そのための特別な認証プロシージャが必要です。また、Oracle Databaseでは、ネットワーク認証のセキュリティを確保するために送信時にパスワードの暗号化も実行されます。

認証後は、認可プロセスでそのエンティティが実行できるアクセスとアクションのレベルを許可または制限できます。

関連トピック

- 権限とロール認可の構成

親トピック: 認証の構成

3.2 パスワード保護の構成

ユーザーのパスワードは様々な方法で保護できます。たとえば、パスワードの作成要件の制御、パスワード管理ポリシーの使用などの方法があります。

- Oracle Databaseの組込みパスワード保護の概要
  Oracle Databaseでは、ユーザーのパスワードを保護するように設計された一連の組込みパスワード保護が提供されています。
- パスワードの最低要件
  パスワードに関する最低要件のセットが提供されています。
- IDENTIFIED BY句を使用したパスワードの作成
  `IDENTIFIED BY`句を受け入れるSQL文でもパスワードを作成できます。
- パスワード管理ポリシーの使用
  パスワード管理ポリシーで、ユーザーのパスワードの安全性を強化できる一連の制限事項を作成して実施できます。
- アプリケーションの段階的データベース・パスワード・ロールオーバーの管理
  段階的なデータベース・パスワード・ロールオーバーを行うと、古いパスワードを指定した期間有効なままにすることで、アプリケーションの停止時間が発生しないようにしながら、新しいパスワードがアプリケーション・クライアントに伝播されている間にアプリケーションのデータベース・パスワードを更新できます。
- パスワードの複雑度の管理
  Oracle Databaseには、パスワードの複雑度の管理に使用できる一連の関数があります。
- パスワードでの大/小文字の区別の管理
  以前のリリースのユーザー・アカウントのパスワードに対して、パスワードの大/小文字の区別を管理できます。
- パスワードのセキュリティへの脅威からの12Cパスワード・バージョンによる保護
  12Cパスワード・バージョンを使用すると、ユーザーは、コンプライアンス基準を満たす複雑なパスワードを作成できます。
- パスワード資格証明用の安全性の高い外部パスワード・ストアの管理

安全性の高い外部パスワード・ストアは、パスワード資格証明の格納に使用されるクライアント側のウォレットになります。

- 管理ユーザーのパスワードの管理
  パスワード・ファイルやパスワード複雑度ファンクションなど、管理ユーザーのパスワードには特別な保護機能があります。

親トピック: 認証の構成

3.2.1 Oracle Databaseの組込みパスワード保護の概要

Oracle Databaseでは、ユーザーのパスワードを保護するように設計された一連の組込みパスワード保護が提供されています。

提供されるパスワード保護は、次のとおりです。

- パスワード暗号化。Oracle Databaseでは、Advanced Encryption Standard(AES)を使用して、ネットワーク(クライアントとサーバー間、双方のサーバー間)接続中にパスワードを自動的かつ透過的に暗号化し、その後ネットワーク経由で送信します。ただし、SQL文内で指定されたパスワード(例: `CREATE USER user_name IDENTIFIED BY password;`)は、ネットワーク・トレース・ファイル内のクリアテキストでネットワーク経由で送信されます。このため、ネイティブ・ネットワーク暗号化を有効にするか、Transport Layer Security (TLS)暗号化を構成する必要があります。

- パスワードの複雑度のチェック。デフォルトのインストールでは、Oracle Databaseには、`ora12c_verify_function`および`ora12c_strong_verify_function`パスワード検証ファンクションがあります。これらの機能では、新しいパスワードや変更されたパスワードの複雑性が十分であり、パスワードを推測してシステムに侵入しようとする侵入者を防ぐことができるかどうかを確認します。パスワードの複雑性チェックを手動で有効にする必要があります。さらに、ユーザーのパスワードの複雑度はカスタマイズできます。

- パスワード突破の防止。ユーザーが誤ったパスワードを使用してOracle Databaseへのログインを複数回試行した場合、Oracle Databaseによってログインが1秒ずつ遅延されます。この保護は、異なるIPアドレスまたは複数のクライアント接続からのログインに適用されます。この機能により、侵入者がログインを試みるときに一定時間内に試すことができるパスワードの数が大幅に減少します。ログイン失敗による遅延によって、各ログイン試行の失敗にかかる時間が延び、(通常、こうした攻撃では非常に多くの試行に失敗せざるを得ないため)パスワード推測攻撃全体にかかる時間が増加します。

  非管理ログインの場合、Oracle Databaseは、ログイン失敗による遅延に対して排他ロックを設定することによって、同時パスワード推測攻撃からデータを保護します。これによって、侵入者がログイン失敗による遅延を回避しようとする(最初の推測に失敗して遅延された後すぐに別のデータベース・セッションで次の同時推測を試行した場合)のを阻止します。

  攻撃対象のアカウントに対して排他ロックを保持することによって、Oracle Databaseでは同時パスワード推測攻撃が緩和されますが、同時にそのアカウントはサービス拒否(DoS)攻撃を受けやすいままになります。この問題に対処するには、`FAILED_LOGIN_ATTEMPTS`パラメータが`UNLIMITED`に設定されたパスワード・プロファイルを作成し、このパスワード・プロファイルをそのユーザー・アカウントに適用する必要があります。`FAILED_LOGIN_ATTEMPTS`パラメータに値`UNLIMITED`を設定することで、ログイン失敗による遅延が無効になり、ログイン試行失敗回数は制限されません。このようなタイプのアカウントについては、長いランダムなパスワードを使用することをお薦めします。

  同時パスワード推測攻撃の保護は、管理ユーザー接続には適用されません。これは、このような接続は常に使用可能な状態である必要があり、サービス拒否攻撃の影響を受けない必要があるためです。したがって、すべての管理権限アカウントに対して長いパスワードを選択することをお薦めします。

- パスワードでの大/小文字の区別の規定。パスワードは大/小文字が区別されます。たとえば、パスワードが`hPP5620qr`の場合、`hpp5620QR`または`hPp5620Qr`と入力すると失敗します。大/小文字の区別は、パスワード・

ファイルおよびデータベース・リンクに影響します。

- 12Cパスワード・バージョンを使用したパスワードのハッシュ化。ユーザーのパスワードを検証し、パスワード作成で大/小文字の区別を適用するために、12Cパスワード・バージョンが使用されていて、これは、パスワードベースのキー導出関数(PBKDF2)およびSHA-512暗号化ハッシュ関数を含む非最適化されたアルゴリズムに基づいています。

関連トピック

- パスワードの保護に関するガイドライン

親トピック: パスワード保護の構成

3.2.2 パスワードの最低要件

パスワードに関する最低要件のセットが提供されています。

パスワードは30バイト以内にする必要があります。パスワードを保護するには、パスワードが適切な長さになるよう要求することから、サイトで適用されるパスワード複雑度ポリシー要件を強制するカスタムのパスワード複雑度検証スクリプトの作成まで、様々な方法があります。

関連トピック

- パスワードの保護に関するガイドライン

親トピック: パスワード保護の構成

3.2.3 IDENTIFIED BY句を使用したパスワードの作成

`IDENTIFIED BY`句を受け入れるSQL文でもパスワードを作成できます。

- ユーザーのパスワードを作成するには、`CREATE USER`、`ALTER USER`、`GRANT CREATE SESSION`または`CREATE DATABASE LINK` SQL文を使用します。

次のSQL文は、`IDENTIFIED BY`句でパスワードを作成します。

```
CREATE USER psmith IDENTIFIED BY password;
GRANT CREATE SESSION TO psmith IDENTIFIED BY password;
ALTER USER psmith IDENTIFIED BY password;
CREATE DATABASE LINK AUTHENTICATED BY psmith IDENTIFIED BY password;
```

関連トピック

- パスワードの複雑度検証について

親トピック: パスワード保護の構成

3.2.4 パスワード管理ポリシーの使用

パスワード管理ポリシーで、ユーザーのパスワードの安全性を強化できる一連の制限事項を作成して実施できます。

- パスワード管理について
  パスワードに依存しているデータベース・セキュリティ・システムでは、パスワードの機密を常に保つ必要があります。
- デフォルト・パスワードが設定されているユーザー・アカウントの検索
  `DBA_USERS_WITH_DEFPWD`データ・ディクショナリ・ビューで、デフォルトのパスワードを使用するユーザー・アカウントを確認できます。
- デフォルト・プロファイルのパスワード設定

プロファイルとは、データベース・リソースに関する制限を設定するパラメータの集合です。

- ALTER PROFILE文を使用したプロファイル制限の設定
  ログイン試行の失敗、パスワードのロック回数、パスワードの再利用、その他の設定などのプロファイルの制限を変更できます。
- デフォルトのパスワード・セキュリティ設定の有効化および無効化
  デフォルトのパスワード・セキュリティ設定を無効または有効にするスクリプトが用意されています。
- 非アクティブなデータベース・ユーザー・アカウントの自動ロック
  INACTIVE_ACCOUNT_TIMEプロファイル・パラメータで、指定した日数の間データベース・インスタンスにログインしていないユーザー・アカウントをロックします。
- ログイン試行に指定の回数だけ失敗した後のユーザー・アカウントの自動ロック
  Oracle Databaseでは、ログイン試行に指定の回数だけ連続して失敗した後、ユーザーのアカウントをロックできます。
- 例: CREATE PROFILE文を使用したアカウントのロック
  CREATE PROFILE文で、ユーザーのログイン試行がCREATE PROFILE設定に違反した場合にユーザー・アカウントをロックできます。
- ユーザー・アカウントの明示的ロック
  ユーザー・アカウントを明示的にロックした場合、アカウントのロックは自動的には解除されません。セキュリティ管理者のみがアカウントのロックを解除できます。
- ユーザーによる以前のパスワードの再利用の制御
  一定の期間、または一定のパスワード変更回数を経過するまで、ユーザーが前のパスワードを再利用しないようにできます。
- パスワード・エイジングおよび期限切れの制御について
  パスワードの存続期間を指定して、この期間を過ぎるとパスワードを期限切れにできます。
- CREATE PROFILE文またはALTER PROFILE文を使用したパスワード存続期間の設定
  パスワードの存続期間を設定した場合、存続期間の終了時にユーザーは新しいパスワードを作成する必要があります。
- ユーザー・アカウントのステータスの確認
  アカウント・ステータスは、オープン、猶予期間、期限切れのいずれの場合も確認できます。
- パスワード変更のライフ・サイクル
  パスワードが作成されると、そのパスワードは、次の4つのフェーズのライフ・サイクルと猶予期間をたどります。
- PASSWORD_LIFE_TIMEプロファイル・パラメータの低い値
  CREATE PROFILEまたはALTER PROFILEのPASSWORD_LIFE_TIMEパラメータを低い値(たとえば1日)に設定する場合は、注意が必要です。

親トピック: パスワード保護の構成

3.2.4.1 パスワード管理について

パスワードに依存しているデータベース・セキュリティ・システムでは、パスワードの機密を常に保つ必要があります。

パスワードは盗難や悪用などの被害を受けやすいため、Oracle Databaseではパスワード管理ポリシーが使用されています。データベース管理者およびセキュリティ管理者がユーザー・プロファイルを介してパスワード管理ポリシーを制御することで、データベース・セキュリティの管理を強化できます。

CREATE PROFILE文を使用して、ユーザー・プロファイルを作成できます。プロファイルは、CREATE USERまたはALTER USER文を使用してユーザーに割り当てます。

親トピック: パスワード管理ポリシーの使用

3.2.4.2 デフォルト・パスワードが設定されているユーザー・アカウントの検索

DBA_USERS_WITH_DEFPWDデータ・ディクショナリ・ビューで、デフォルトのパスワードを使用するユーザー・アカウントを確認できます。

データベースを作成すると、ほとんどのデフォルト・アカウントは、パスワードが期限切れのためロックされます。以前のリリースのOracle Databaseからアップグレードした場合は、デフォルト・パスワードが設定されているユーザー・アカウントが存在していることがあります。それらは、データベースの作成時に作成されたデフォルト・アカウント(HR、OE、SCOTTアカウントなど)です。

セキュリティを強化するために、それらのアカウントのパスワードを変更してください。周知されているデフォルト・パスワードを使用すると、データベースが侵入者から攻撃を受けやすくなります。

1. SYSDBA管理権限を持つSQL*Plusを使用して、データベース・インスタンスにログインします。

   たとえば:

   ```
   sqlplus sys as sysdba
   Enter password: password
   ```

2. DBA_USERS_WITH_DEFPWDデータ・ディクショナリ・ビューを問い合せます。

   たとえば、デフォルトのパスワードが設定されているアカウントの名前とステータスの両方を検索するには:

   ```
   SELECT d.username, u.account_status
   FROM DBA_USERS_WITH_DEFPWD d, DBA_USERS u
   WHERE d.username = u.username
   ORDER BY 2,1;
   USERNAME  ACCOUNT_STATUS
   --------- --------------------------
   SCOTT     EXPIRED & LOCKED
   ```

3. DBA_USERS_WITH_DEFPWDビューにリストされたアカウントのパスワードを変更します。

   これらのアカウントに、旧リリースのOracle Databaseで指定されていた可能性のあるパスワードを割り当てないことをお薦めします。

   たとえば:

   ```
   ALTER USER SCOTT ACCOUNT UNLOCK IDENTIFIED BY password;
   ```

   「パスワードの最低要件」のガイドラインに従って、passwordを安全なパスワードに置き換えます。

親トピック: パスワード管理ポリシーの使用

3.2.4.3 デフォルト・プロファイルのパスワード設定

プロファイルとは、データベース・リソースに関する制限を設定するパラメータの集合です。

プロファイルをユーザーに割り当てた場合、そのユーザーはそれらの制限を超えることはできません。プロファイルを使用すると、ユーザーごとのセッション数、ロギングやトレースの機能など、データベース設定を構成できます。また、プロファイルによってユーザー・パスワードも制御できます。プロファイル内の現行のパスワード設定に関する情報は、DBA_PROFILESデータ・ディクショナリ・ビューを問い合せることで確認できます。

表3-1に、デフォルト・プロファイルのパスワード固有のパラメータ設定を示します。

表3-1 デフォルト・プロファイルのパスワード固有の設定

パラメータ	デフォルト設定	説明

パラメータ	デフォルト設定	説明
INACTIVE_ACCOUNT_TIME	UNLIMITED	指定した日数の間データベース・インスタンスにログインしていないデータベース・ユーザーのアカウントをロックします。
FAILED_LOGIN_ATTEMPTS	10	ユーザーがログインを試行して失敗する最大回数。この回数を超えるとアカウントがロックされます。 ノート: ● このパラメータを設定する場合、CONNECT THROUGH 権限を使用してログインするユーザーを考慮します。 ● 未認可ユーザー(侵入者の可能性があります)がOracle Call Interface(OCI)アプリケーションにログインを試行する回数の制限を設定するには、SEC_MAX_FAILED_LOGIN_ATTEMPTS 初期化パラメータを使用できます。
PASSWORD_GRACE_TIME	7	パスワードが期限切れになる前に、ユーザーがパスワードを変更するための日数を設定します。
PASSWORD_LIFE_TIME	180	ユーザーが現行のパスワードを使用できる日数を設定します。
PASSWORD_LOCK_TIME	1	ログインを指定の回数だけ連続して失敗した後に、アカウントがロックされる日数を設定します。この期間が経過すると、アカウントのロックは解除されます。このユーザー・プロファイル・パラメータは、管理者のメンテナンス負荷を高めることなく、ユーザー・パスワードに対する総当り攻撃を容易に防止するのに役立ちます。 PASSWORD_LOCK_TIME で設定された値がパスワードの有効期限が切れていることを示した後でも、DBA_USERS データ・ディクショナリ・ビューにはアカウントがロックされていることが示されます。ただし、ユーザーが接続すると、DBA_USERS の情報が正しい OPEN ステータスで更新されます。
PASSWORD_REUSE_MAX	UNLIMITED	現行のパスワードを再利用できるようになるまでに必要なパスワード変更の回数を設定します。

パラメータ	デフォルト設定	説明
PASSWORD_REUSE_TIME	UNLIMITED	パスワードを再利用できない日数を設定します。

関連トピック

- プロファイルによるリソースの管理
- 非アクティブなデータベース・ユーザー・アカウントの自動ロック
- 認証の最大試行回数の構成
- ログイン試行に指定の回数だけ失敗した後のユーザー・アカウントの自動ロック
- パスワード・エイジングおよび期限切れの制御について
- ユーザーによる以前のパスワードの再利用の制御

親トピック: パスワード管理ポリシーの使用

3.2.4.4 ALTER PROFILE文を使用したプロファイル制限の設定

ログイン試行の失敗、パスワードのロック回数、パスワードの再利用、その他の設定などのプロファイルの制限を変更できます。

これらの設定については、表3-1で説明されています。セキュリティを強化するために、必要に応じて、この表に示すデフォルト設定を使用してください。

- ALTER PROFILE文を使用して、ユーザーのプロファイル制限を変更します。

たとえば:

```
ALTER PROFILE prof LIMIT
 FAILED_LOGIN_ATTEMPTS 9
 PASSWORD_LOCK_TIME 10
 INACTIVE_ACCOUNT_TIME 21;
```

親トピック: パスワード管理ポリシーの使用

3.2.4.5 デフォルトのパスワード・セキュリティ設定の有効化および無効化

デフォルトのパスワード・セキュリティ設定を無効または有効にするスクリプトが用意されています。

アプリケーションでOracle Database 10g リリース2 (10.2)のデフォルトのパスワード・セキュリティ設定を使用している場合、Oracle Databaseリリース11g以降のデフォルトのパスワード・セキュリティ設定を使用するようにアプリケーションを変更しないかぎり、それらの設定の復元が可能です。

1. Oracle Database 11g以降のパスワード・セキュリティ設定に準拠するように、アプリケーションを変更します。
2. 次のいずれかの方法で、ビジネス・ニーズに合うセキュリティ構成を使用するようにデータベースを更新します。
    - データベース・セキュリティ構成を手動で更新します。
    - secconf.sqlスクリプトを実行して、Oracle Database 11g以降のデフォルトのパスワード設定を適用します。必要に応じて、異なるセキュリティ設定を使用するようにこのスクリプトをカスタマイズできますが、元のスクリプトにリストされている設定は、Oracle推奨の設定であることに注意してください。

データベースを手動で作成した場合は、secconf.sqlスクリプトを実行して、Oracleのデフォルトのパスワード設定をデータベースに適用する必要があります。Database Configuration Assistant(DBCA)を使用して作成されたデータベースではこの設定が使用されますが、手動で作成したデータベースでは使用されません。

secconf.sqlスクリプトは$ORACLE_HOME/rdbms/adminディレクトリにあります。secconf.sqlスクリプトはパスワー

ド設定と監査設定の両方に影響を与えます。他のセキュリティ設定には影響しません。

親トピック: パスワード管理ポリシーの使用

3.2.4.6 非アクティブなデータベース・ユーザー・アカウントの自動ロック

INACTIVE_ACCOUNT_TIMEプロファイル・パラメータで、指定した日数の間データベース・インスタンスにログインしていないユーザー・アカウントをロックします。

定期的にログインしているユーザーはアクティブであると見なされます。INACTIVE_ACCOUNT_TIMEの計時は、最後にユーザーが正常にログインしてからの日数に基づきます。

- 指定した日数が経過した後で自動的にユーザー・アカウントをロックするには、CREATE PROFILEまたはALTER PROFILE文でINACTIVE_ACCOUNT_TIMEプロファイル・パラメータを設定します。
  次のことに注意してください。
  - INACTIVE_ACCOUNT_TIMEのデフォルト値はUNLIMITEDです。
  - 日数には整数を指定する必要があります。最小値は15、最大値は24855です。
  - ユーザーのアカウントの非アクティブ時間を無制限に設定するには、INACTIVE_ACCOUNT_TIMEをUNLIMITEDに設定します。
  - デフォルト・プロファイルで指定された時間を使用するようにユーザーのアカウントを設定するには、INACTIVE_ACCOUNT_TIMEをDEFAULTに設定します。
  - このパラメータは、管理ユーザーを含むすべてのデータベース認証ユーザーに対して設定できますが、外部またはグローバル認証ユーザーには設定できません。
  - 読取り専用のデータベースでは、最後の正常なログインはINACTIVE_ACCOUNT_TIMEの計時において考慮されません。読取り専用のデータベースでユーザー・アカウントをロックすることはできません(ログイン失敗の連続回数がアカウントのFAILED_LOGIN_ATTEMPTSパスワード・プロファイル設定の値に達した場合を除きます)。
  - 新規作成したユーザー・アカウントの場合、計時はアカウント作成時に開始されます。このユーザーがログアウトして再度ログインした場合、計時はユーザーが正常にログインしたときに開始されます。
  - マルチテナント環境では、INACTIVE_ACCOUNT_TIME設定は、共通ユーザーが最後にルートにログインした時点で適用されます。PDBのいずれかまたはルートにログインしている共通ユーザーはアクティブであると見なされます。
  - プロキシ・ユーザー・アカウントのログインの場合、INACTIVE_ACCOUNT_TIMEの計時は、プロキシ・ユーザーが正常にログインしたときに開始されます。

たとえば、アクティブでない状態で60日間経過すると、アカウントをロックするプロファイルを作成するとします。

```
CREATE PROFILE time_limit LIMIT
 INACTIVE_ACCOUNT_TIME 60;
```

親トピック: パスワード管理ポリシーの使用

3.2.4.7 ログイン試行に指定の回数だけ失敗した後のユーザー・アカウントの自動ロック

Oracle Databaseでは、ログイン試行に指定の回数だけ連続して失敗した後、ユーザーのアカウントをロックできます。

- 指定した時間間隔が経過した後で自動的にユーザー・アカウントをロックするか、またはロックを解除するためにデータ

ベース管理者の介入を必要とするには、CREATE PROFILE文またはALTER PROFILE文でユーザーのPASSWORD_LOCK_TIMEプロファイル・パラメータを設定します。

たとえば、時間間隔を10日に設定するには、次のようにします。

```
PASSWORD_LOCK_TIME = 10
```

次のことに注意してください。

- データベース管理者による明示的な解除を強制するため、手動でアカウントをロックできます。
- ログインの失敗が許容される回数は、CREATE PROFILE文を使用して指定します。また、アカウントがロックされる時間の長さも指定できます。
- ユーザーがログインに失敗するたびに、Oracle Databaseでは次第に遅延時間が長くなります。
- アカウントのロック解除の間隔を指定しない場合、PASSWORD_LOCK_TIMEはデフォルト・プロファイルに指定されている値を想定します。(推奨値は1日です。)PASSWORD_LOCK_TIMEをUNLIMITEDとして指定すると、ALTER USER文を使用してアカウントを明示的にロック解除する必要があります。たとえば、PASSWORD_LOCK_TIME UNLIMITEDがjohndoeに指定されていると想定して、次の文を使用してjohndoeアカウントをロック解除します。

  ```
  ALTER USER johndoe ACCOUNT UNLOCK;
  ```

- ユーザーが正常にアカウントにログインすると、Oracle Databaseでは、そのユーザーが失敗したログインの回数がリセットされます。ゼロでない場合、回数はゼロに設定されます。
- マルチテナント環境では、ロック対象のCDB共通ユーザー・アカウントは、CDB内のすべてのPDBにわたってロックされます。ロック対象のアプリケーション共通ユーザー・アカウントは、アプリケーション・ルートに関連付けられたすべてのPDBにわたってロックされます。

親トピック: パスワード管理ポリシーの使用

3.2.4.8 例: CREATE PROFILE文を使用したアカウントのロック

CREATE PROFILE文で、ユーザーのログイン試行がCREATE PROFILE設定に違反した場合にユーザー・アカウントをロックできます。

例3-1では、ユーザーjohndoeに対して許容されているログイン失敗の最大回数は10回(デフォルト)、アカウントがロックされる時間の長さは30日です。アカウントのロックは、30日が経過すると自動的に解除されます。

例3-1 CREATE PROFILE文を使用したアカウントのロック

```
CREATE PROFILE prof LIMIT
 FAILED_LOGIN_ATTEMPTS 10
 PASSWORD_LOCK_TIME 30
ALTER USER johndoe PROFILE prof;
```

親トピック: パスワード管理ポリシーの使用

3.2.4.9 ユーザー・アカウントの明示的ロック

ユーザー・アカウントを明示的にロックした場合、アカウントのロックは自動的には解除されません。セキュリティ管理者のみがアカウントのロックを解除できます。

マルチテナント環境では、CDBルートでCDB共通ユーザー・アカウントをロックすると、このユーザーはこのルートに関連付けられたすべてのPDBにログインできなくなり、PDBでこのアカウントのロックを解除することもできなくなります。また、CDB共通アカウントをPDBでローカルにロックできます。これにより、CDB共通ユーザーがそのPDBにログインできなくなります。同様に、アプリケー

ション・ルートでロックされたアプリケーション共通ユーザー・アカウントは、アプリケーション・ルートに関連付けられたすべてのPDBにログインできず、アプリケーションPDBでこのアプリケーション共通ユーザーのロックを解除することもできません。アプリケーションPDB内でローカルに、アプリケーション共通ユーザーを明示的にロックすることもできます。

- ユーザー・アカウントを明示的にロックするには、CREATE USER文またはALTER USER文を使用します。

たとえば、次の文はユーザー・アカウントsusanをロックします。

```
ALTER USER susan ACCOUNT LOCK;
```

親トピック: パスワード管理ポリシーの使用

3.2.4.10 ユーザーによる以前のパスワードの再利用の制御

一定の期間、または一定のパスワード変更回数を経過するまで、ユーザーが前のパスワードを再利用しないようにできます。

- 指定した期間、ユーザーが以前のパスワードを再利用できないようにするには、CREATE PROFILE文またはALTER PROFILE文を使用してパスワードの再利用のルールを構成できます。

次の表に、ユーザーによる前のパスワードの再利用を制御するCREATE PROFILEとALTER PROFILEのパラメータを示します。

表3-2 前のパスワードの再利用を制御するパラメータ

パラメータ名	説明および使用方法
PASSWORD_REUSE_TIME	次のいずれかを指定する必要があります。 ● 以前に使用していた同じパスワードを次に使用できるようになるまでの日数(または 1 日の一部分)を示す数字 ● UNLIMITED の文字
PASSWORD_REUSE_MAX	次のいずれかを指定する必要があります。 ● パスワードを再利用できるようになるまでに必要なパスワード変更の回数を示す整数 ● UNLIMITED の文字

パラメータを指定しない場合は、ユーザーがいつでもパスワードを再利用できる状況になります。これはセキュリティの方法としては適切ではありません。

どちらもUNLIMITEDではない場合、両方の条件に一致した場合にのみ再利用できます。つまり、そのパスワードを最後に使用してから、指定された回数のパスワード変更を行っていること、および指定された日数が経過している必要があります。

たとえば、ユーザーAのプロファイルでPASSWORD_REUSE_MAXが10、PASSWORD_REUSE_TIMEが30に指定されている場合を考えます。ユーザーAのパスワードは、そのパスワードを最後に使用してから30日が経過し、パスワードを10回再設定するまでは再利用できません。

一方のパラメータがUNLIMITEDに指定されている場合、ユーザーはパスワードを再利用できません。

両方のパラメータをUNLIMITEDに設定している場合は、両方が無視され、ユーザーはいつでもパスワードを再利用できます。

ノート:

いずれかのパラメータに DEFAULT を指定すると、DEFAULT のプロファイルに定義されている値が使用されます。このプロファイルでは、すべてのパラメータが UNLIMITED に設定されています。したがって、DEFAULT のプロファイルでそのパラメータの設定を変更していない場合は、DEFAULT として指定されているパラメータには UNLIMITED が使用されます。

関連トピック

- Oracle Database SQL言語リファレンス

親トピック: パスワード管理ポリシーの使用

3.2.4.11 パスワード・エイジングおよび期限切れの制御について

パスワードの存続期間を指定して、この期間を過ぎるとパスワードを期限切れにできます。

つまり、ユーザーは現行の正しいパスワードを使用して次回ログインする際に、パスワードの変更を求められるということです。デフォルトでは、複雑度チェックやパスワード履歴チェックは行われないため、ユーザーは以前のパスワードや脆弱なパスワードを再利用できます。PASSWORD_REUSE_TIME、PASSWORD_REUSE_MAXおよびPASSWORD_VERIFY_FUNCTIONパラメータを設定することによって、これらの要素を制御します。

さらに、猶予期間を設定できます。この期間中は、データベース・アカウントへのログインを試行するたびに、パスワードの変更を求める警告メッセージが発行されます。ユーザーがこの期間内にパスワードを変更しないと、Oracle Databaseではアカウントを期限切れにします。

データベース管理者は、手動でパスワードを期限切れ状態に設定できます(アカウント・ステータスをEXPIREDに設定します)。この場合、ユーザーはログオンを続行する前に、プロンプトに従ってパスワードを変更する必要があります。

たとえば、SQL*Plusにおいて、ユーザーSCOTTは正しい資格証明を使用してログインを試行しますが、パスワードが期限切れだとします。続いて、ユーザーSCOTTに「ORA-28001: パスワードが期限切れです。」エラーが表示され、次のようにパスワードの変更を求められます。

```
Changing password for scott
New password: new_password
Retype new password: new_password
Password changed.
```

関連トピック

- ユーザーによる以前のパスワードの再利用の制御
- パスワードの複雑度検証について

親トピック: パスワード管理ポリシーの使用

3.2.4.12 CREATE PROFILE文またはALTER PROFILE文を使用したパスワード存続期間の設定

パスワードの存続期間を設定した場合、存続期間の終了時にユーザーは新しいパスワードを作成する必要があります。

- パスワードの存続期間を指定するには、CREATE PROFILE文またはALTER PROFILE文を使用します。

次の例は、プロファイルを作成してユーザーjohndoeに割り当てる方法を示しています。PASSWORD_LIFE_TIME句によって、johndoeはパスワードの期限が切れるまで180日間同じパスワードを使用できることが指定されています。

```
CREATE PROFILE prof LIMIT
 FAILED_LOGIN_ATTEMPTS 4
 PASSWORD_GRACE_TIME 3
 PASSWORD_LIFE_TIME 180;
ALTER USER johndoe PROFILE prof;
```

関連トピック

- パスワード変更のライフ・サイクル

親トピック: パスワード管理ポリシーの使用

3.2.4.13 ユーザー・アカウントのステータスの確認

アカウント・ステータスは、オープン、猶予期間、期限切れのいずれの場合も確認できます。

- ユーザー・アカウントのステータスをチェックするには、DBA_USERSデータ・ディクショナリ・ビューのACCOUNT_STATUS列を問い合せます。

たとえば:

```
SELECT ACCOUNT_STATUS FROM DBA_USERS WHERE USERNAME = 'username';
```

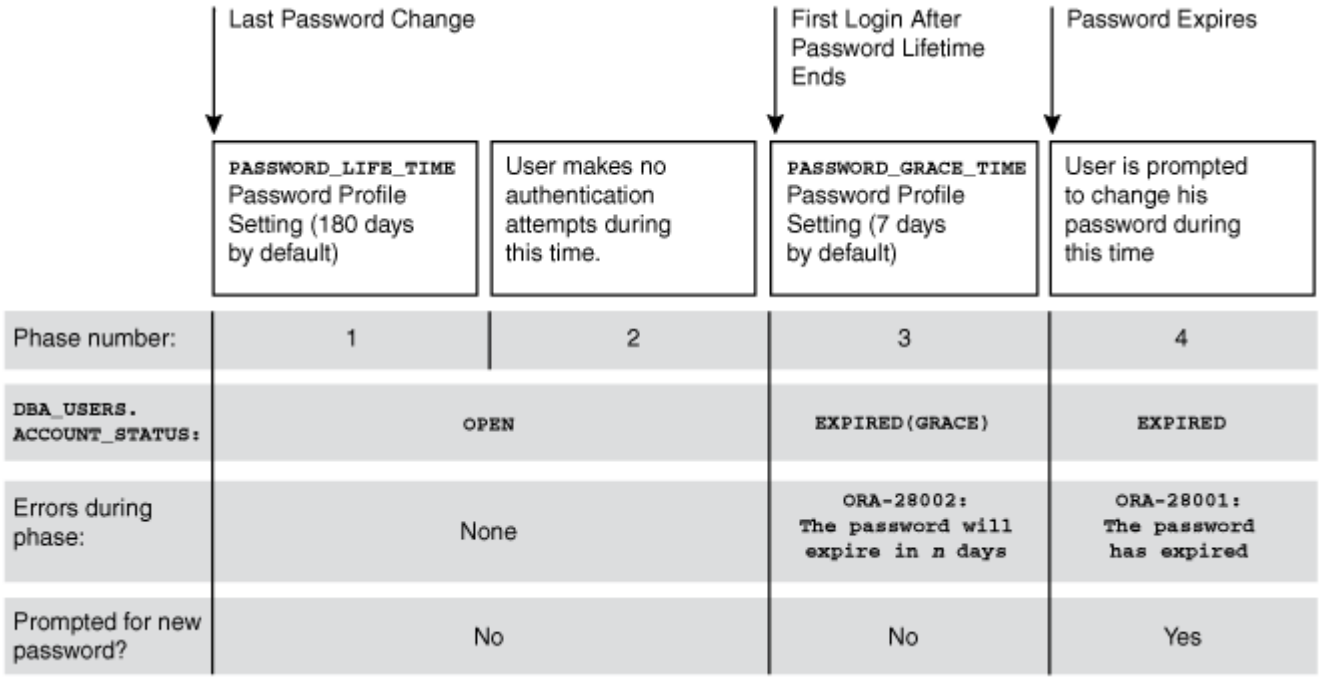
親トピック: パスワード管理ポリシーの使用

3.2.4.14 パスワード変更のライフ・サイクル

パスワードが作成されると、そのパスワードは、次の4つのフェーズのライフ・サイクルと猶予期間をたどります。

次の図は、パスワードの存続期間および猶予期間のライフ・サイクルを示しています。

図3-1 パスワード変更のライフ・サイクル



この図の内容は次のとおりです。

- フェーズ1: ユーザー・アカウントが作成されたか、既存のアカウントのパスワードが変更された後、パスワードの存続期間が開始されます。
- フェーズ2: このフェーズは、パスワードの存続期間が終了した後で、正しいパスワードを使用してユーザーが再度ログインする前の期間を表しています。Oracle Databaseによってアカウント・ステータスが更新されるためには、正しい資格

証明が必要です。それ以外の場合、アカウント・ステータスは変更されません。Oracle Databaseには、アカウント・ステータスを更新するためのバックグラウンド・プロセスは存在しません。アカウント・ステータスの変更はすべて、認証されたユーザーにかわって、Oracle Databaseのサーバー・プロセスによって実行されます。

- フェーズ3: 最終的にユーザーがログインすると、猶予期間が開始されます。Oracle Databaseによって、現在の時間にそのアカウントのパスワード・プロファイルのPASSWORD_GRACE_TIME設定の値を加えた値が使用されて、DBA_USERS.EXPIRY_DATE列の値が更新されます。この時点でユーザーは、近い将来にパスワードが期限切れするというORA-28002警告メッセージ(たとえば、PASSWORD_GRACE_TIMEが7日に設定されている場合は、「ORA-28002 パスワードは、7日以内に期限切れになります。」)を受け取りますが、依然としてパスワードを変更することなくログインできます。DBA_USERS.EXPIRY_DATE列には、ユーザーがパスワードを変更するよう求められる将来の時間が示されます。
- フェーズ4: 猶予期間(フェーズ3)が終了した後、ユーザーが現行の正しいパスワードを入力すると、認証が続行される前に「ORA-28001: パスワードが期限切れです。」エラーが表示されて、パスワードを変更するよう求められます。ユーザーが、Oracle Active Data Guard構成を使用しており(その場合は、プライマリ・データベースとスタンバイ・データベースが存在します)、認証がスタンバイ・データベース(読取り専用データベース)で試行された場合は、「ORA-28032: パスワードの期限が切れており、データベースは読取り専用に設定されています」エラーが表示されます。ユーザーは、プライマリ・データベースにログインして、そこでパスワードを変更する必要があります。

これら4フェーズのいずれの間も、DBA_USERSデータ・ディクショナリ・ビューに問い合せて、DBA_USERS.ACCOUNT_STATUS列でユーザーのアカウント・ステータスを検索できます。

次の例では、johndoeに割り当てられたプロファイルに、猶予期間PASSWORD_GRACE_TIME = 3(推奨値)が指定されています。johndoeは90日後に初めてデータベースにログインしようとすると(これは90日目より後の任意の日、つまり91日目や100日目でも構いません)、パスワードが3日で期限切れになるという警告メッセージを受け取ります。3日経過してもパスワードを変更しない場合は、パスワードの期限が切れます。その後、このユーザーがログインしようとするたびに、パスワードの変更を求めるプロンプトが表示されます。

```
CREATE PROFILE prof LIMIT
 FAILED_LOGIN_ATTEMPTS 4
 PASSWORD_LIFE_TIME 90
 PASSWORD_GRACE_TIME 3;
ALTER USER johndoe PROFILE prof;
```

データベース管理者またはALTER USERシステム権限を持つユーザーは、CREATE USER文およびALTER USER文を使用して、任意のパスワードを明示的に期限切れにできます。次の文は、期限切れのパスワードを持つユーザーを作成します。この設定によって、ユーザーがデータベースにログインする前に、強制的にパスワードを変更させることができます。

```
CREATE USER jbrown
 IDENTIFIED BY password
 ...
 PASSWORD EXPIRE;
```

CREATE USER文にパスワードの期限切れを解除する句はありませんが、アカウントのパスワードを変更することによって、期限切れは解除されます。

親トピック: パスワード管理ポリシーの使用

3.2.4.15 PASSWORD_LIFE_TIMEプロファイル・パラメータの低い値

CREATE PROFILEまたはALTER PROFILEのPASSWORD_LIFE_TIMEパラメータを低い値(たとえば1日)に設定する場合は、注意が必要です。

プロファイルのPASSWORD_LIFE_TIME制限は、アカウントのパスワードが最後に変更された時点、またはパスワードが一度も変更されていない場合はアカウントの作成時点から測定されます。これらの日付は、SYS.USER$システム表のPTIME (パスワード変更時間)列およびCTIME (アカウント作成時間)列に記録されています。PASSWORD_LIFE_TIME制限の測定は、PASSWORD_LIFE_TIMEプロファイル・パラメータが最後に変更された時点のタイムスタンプから開始されるのではありません(最初はそう考えがちです)。そのため、変更されたプロファイルによって影響を受けるアカウントで、パスワードの最終変更時間がPASSWORD_LIFE_TIME日前より以前のアカウントは、ただちに期限切れとなり、次回の接続時点で猶予期間に入り、「ORA-28002: パスワードは、n日以内に期限切れになります。」警告が発行されます。

データベース管理者は、次の方法で、アカウントのパスワードの最終変更時間を調べることができます。

```
ALTER SESSION SET NLS_DATE_FORMAT='DD-MON-YYYY HH24:MI:SS';
SELECT PTIME FROM SYS.USER$ WHERE NAME = 'user_name'; -- Password change time
```

アカウントの作成時間とパスワードの有効期限を調べるには、次の問合せを発行します。

```
SELECT CREATED, EXPIRY_DATE FROM DBA_USERS WHERE USERNAME = 'user_name';
```

管理者がPASSWORD_LIFE_TIMEパラメータを設定した時点で、このプロファイルを割り当てられているユーザーがログイン中であり、そのままログインし続ける場合、現在記載されている有効期限を過ぎても、このユーザーのアカウント・ステータスは、OPENからEXPIRED(GRACE)に変更されません。時間測定が開始されるのは、ユーザーがデータベースにログインした時点のみです。ユーザーの最終ログイン時間を確認する方法は次のとおりです。

```
SELECT LAST_LOGIN FROM DBA_USERS WHERE USERNAME = 'user_name';
```

パスワードのプロファイルを変更する際にデータベース管理者が注意すべきなのは、このプロファイルの対象となるユーザーの一部が、管理者がパスワードのプロファイルを更新している時点でOracle Databaseにログイン中であれば、これらのユーザーは、パスワードの有効期限を越えてシステムにログインし続けることが可能だということです。現在ログインしているユーザーを見つけるには、V$SESSIONビューのUSERNAME列を問い合せます。

これは、ユーザーのパスワードの有効期限が、パスワード最終変更時点のタイムスタンプに、管理者の設定したPASSWORD_LIFE_TIMEパスワード・プロファイル・パラメータの値を加えたものに基づいているためです。パスワード・プロファイル自体の最終変更時点のタイムスタンプに基づいているのではありません。

次のことに注意してください。

- PASSWORD_LIFE_TIMEを低い値に設定しているときにユーザーがログインしていない場合、ユーザーのアカウント・ステータスはユーザーがログインするまで変わりません。
- PASSWORD_LIFE_TIMEパラメータをUNLIMITEDに設定できますが、この設定が作用するのは猶予期間に入っていないアカウントのみです。猶予期間に入っているユーザーは、その期間を過ぎるとパスワードを変更する必要があります。

親トピック: パスワード管理ポリシーの使用

3.2.5 アプリケーションの段階的データベース・パスワード・ロールオーバーの管理

段階的データベース・パスワード・ロールオーバーを行うと、古いパスワードを指定した期間有効なままにすることで、アプリケーションの停止時間が発生しないようにしながら、新しいパスワードがアプリケーション・クライアントに伝播されている間にアプリケーションのデータベース・パスワードを更新できます。

- アプリケーションの段階的データベース・パスワード・ロールオーバーの管理について
  データベース管理者がアプリケーションのデータベース・パスワードを変更するときに、データベース・アプリケーション・クライアントに対して開始するように段階的なデータベース・パスワード・ロールオーバー・プロセスを構成できます。

- 段階的データベース・パスワード・ロールオーバー中のパスワード変更ライフ・サイクル
  パスワードが作成または変更されると、そのパスワードは、4つのフェーズのライフ・サイクルと猶予期間をたどります。
- 段階的データベース・パスワード・ロールオーバーの有効化
  段階的データベース・パスワード・ロールオーバーを有効にするには、PASSWORD_ROLLOVER_TIMEユーザー・プロファイル・パラメータを構成する必要があります。
- 段階的データベース・パスワード・ロールオーバー期間を開始するためのパスワードの変更
  ゼロ以外のPASSWORD_ROLLOVER_TIME値を設定した後、ユーザーのパスワードを変更し、すべてのアプリケーションでパスワードを更新します。
- 段階的データベース・パスワード・ロールオーバー期間中のパスワードの変更
  ロールオーバー期間の開始後も、パスワードを変更できます。
- パスワード・ロールオーバー期間の終了
  パスワード・ロールオーバー期間を終了するには、複数の方法があります。
- 段階的パスワード・ロールオーバー期間中のデータベース動作
  ユーザーは、パスワード・ロールオーバー期間中に標準のパスワード変更およびログインを実行できます。
- パスワード・ロールオーバー期間終了後のデータベース・サーバーの動作
  Oracle Databaseは、段階的なデータベース・パスワード・ロールオーバー期間の終了後にクリーン・アップ操作を実行します。
- 漏えいしたパスワードの処理に関するガイドライン
  データベース・アカウント・パスワードが漏えいしたと思われる場合は、すぐにパスワードを変更する必要があります。
- Oracle Data Pumpのエクスポート時の段階的データベース・パスワード・ロールオーバーの動作
  パスワード・ロールオーバー期間中のユーザーがエクスポートされた場合、そのユーザーの新しいパスワードに対応するベリファイアのみがエクスポートされます。
- Oracle Data Guard環境での段階的データベース・パスワード・ロールオーバーの使用
  Oracle Data Guard環境では、段階的データベース・パスワード・ロールオーバーを使用するには、ADG_ACCOUNT_INFO_TRACKING環境変数をGLOBALに設定する必要があります。
- 古いパスワードをまだ使用しているユーザーの検索
  LOGIN監査レコードのAUTHENTICATION_TYPEフィールドを使用する問合せを実行して、古いパスワードをまだ使用しているユーザーを検索できます。

親トピック: パスワード保護の構成

3.2.5.1 アプリケーションの段階的データベース・パスワード・ロールオーバーの管理について

データベース管理者がアプリケーションのデータベース・パスワードを変更するときに、データベース・アプリケーション・クライアントに対して開始するように段階的なデータベース・パスワード・ロールオーバー・プロセスを構成できます。

データベース管理者またはアプリケーション管理者がデータベース内のアプリケーションのパスワードを変更するときに、アプリケーションを新しいデータベース・パスワードで更新する必要があります。ユーザーのプロファイルにPASSWORD_ROLLOVER_TIMEパラメータを設定すると、アプリケーションが古いパスワードを使用しようとした結果発生する可能性のある停止時間やアプリケーション停止のリスクを回避しながら、パスワード変更を実行できます。パスワード・ロールオーバーはサーバーからシームレスに実行され、サポートされている既存のすべてのクライアント・バージョンで実行されます。

段階的なデータベース・パスワード・ロールオーバー機能は、アプリケーションのデータベース・アカウント(サービス・アカウント)用に設計されています。アプリケーションは、単一のサーバー(データベース・クライアント)にすることも、複数のデータベース・クライアントを持つ複数のサーバーにスケール・アウトすることもできます。管理ユーザー向けには設計されていないため、管理ユーザーは、関連付けられているプロファイルに関係なく、この機能の使用が制限されます。パスワード・ロールオーバーが有効なプロファイルを

持つユーザーには管理権限を付与できません。

ネイティブ・パスワード認証ユーザー接続の段階的データベース・パスワード・ロールオーバーを構成できます。パスワード・データベース・アカウントをNO AUTHENTICATIONアカウントに変換すると、Oracle Databaseはこのアカウントに関連付けられているパスワードおよびベリファイアを削除します。パスワード認証済ユーザー・アカウントがGLOBAL、EXTERNALまたはNO AUTHENTICATIONアカウントに変換されると、ユーザーはパスワードのロールオーバー期間を暗黙的に終了します。段階的パスワード・ロールオーバーは、11gパスワード・バージョン以降をサポートしています。

また、接続されたユーザー・データベース・リンクを使用する環境について、段階的データベース・パスワード・ロールオーバーを構成することもできます。この場合、段階的データベース・パスワード・ロールオーバーを構成するときに、接続されているユーザー・データベース・リンクのターゲットでターゲット・アカウントをロールオーバーにすることを確認してから、これらのリンクのターゲット・アカウントもロールオーバーします。ターゲット・アカウントをロールオーバーにするには、次の構文を使用します。

```
ALTER USER username IDENTIFIED BY same_new_rollover_password;
```

段階的データベース・パスワード・ロールオーバーは、次の接続には構成できません。

- Oracle Real Application Securityユーザーの直接ログイン
- Kerberosベース、証明書ベースまたはRADIUSベースの外部認証接続
- 集中管理ユーザー(CMU)接続
- 外部パスワード・ファイルを使用する管理接続
- プライマリとスタンバイ間のOracle Data Guard接続

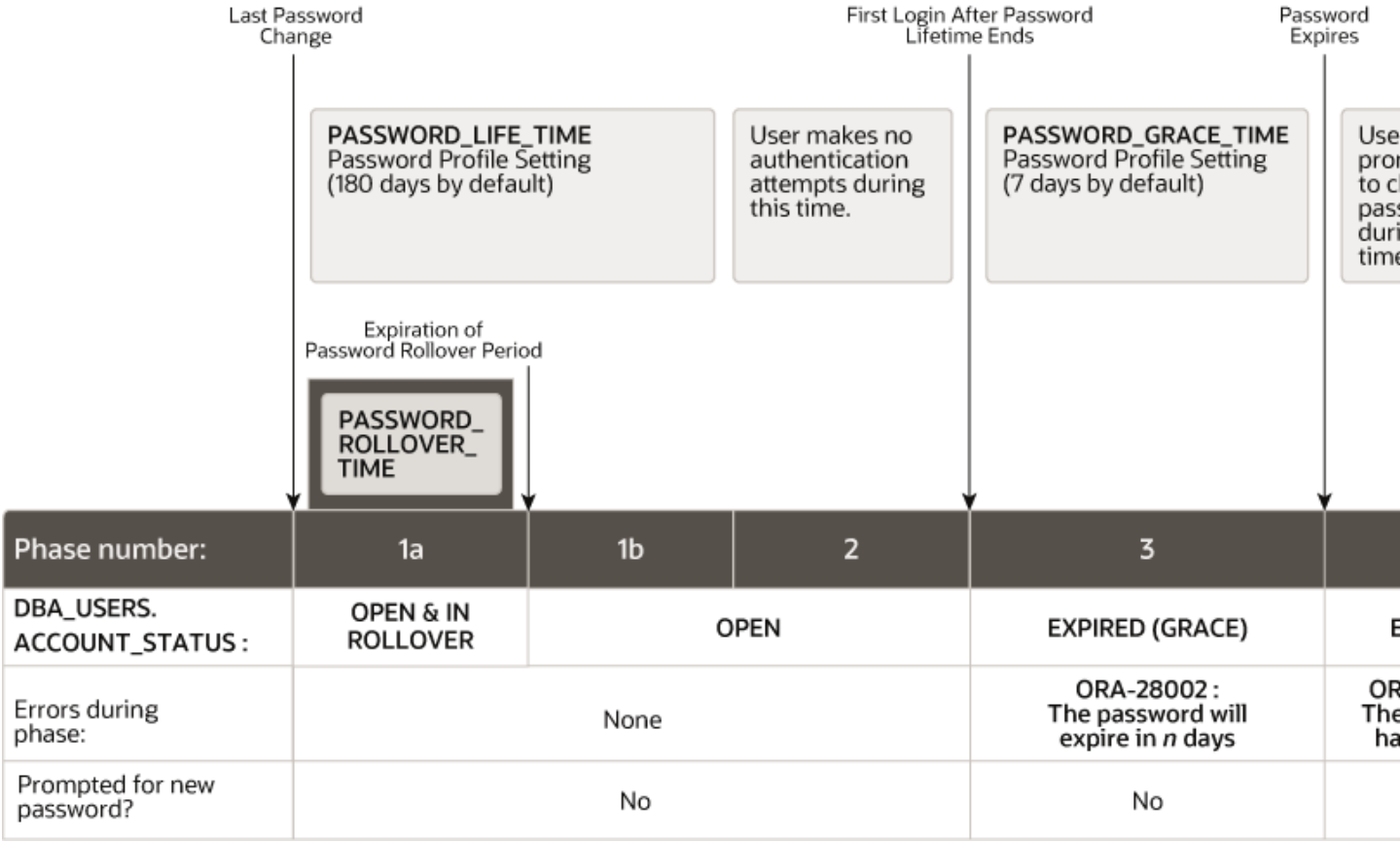
親トピック: アプリケーションの段階的データベース・パスワード・ロールオーバーの管理

3.2.5.2 段階的データベース・パスワード・ロールオーバー中のパスワード変更ライフ・サイクル

パスワードが作成または変更されると、そのパスワードは、4つのフェーズのライフ・サイクルと猶予期間をたどります。

次の図は、パスワードの存続期間および猶予期間のライフ・サイクルを示しています。

図3-2 段階的データベース・パスワード・ロールオーバー中のパスワード変更ライフ・サイクル



この図の内容は次のとおりです。

- フェーズ1: パスワードの存続期間は、ユーザー・アカウントが作成された後、またはパスワードが変更されたときに始まります。既存のアカウントのパスワードが変更され、ユーザーのプロファイルのPASSWORD_ROLLOVER_TIME値がゼロ以外の場合、パスワードの存続期間は1aと1bの2つのフェーズで構成されます。
    - フェーズ1aはパスワード変更から始まります。フェーズ1aでは、ユーザーは旧パスワードまたは新規パスワードのいずれかを使用してログインできます。フェーズ1aの期間は通常PASSWORD_ROLLOVER_TIMEですが、管理者がこれより早くすべてのクライアント・アプリケーションでパスワードを更新できる場合は、次のコマンドを発行してパスワード・ロールオーバー期間を早く終了できます。これにより、新しいパスワードが受け入れられる唯一のパスワードになります。

      ```
      ALTER USER username EXPIRE PASSWORD ROLLOVER PERIOD;
      ```

    - フェーズ1bは、パスワード・ロールオーバー期間が終了してからPASSWORD_LIFE_TIMEが終了するまでの残り時間に対応します。フェーズ1bでは、ユーザーは新しいパスワードのみを使用してログインできます。
- フェーズ2: このフェーズは、パスワードの存続期間が終了した後で、正しいパスワードを使用してユーザーが再度ログインする前の期間を表しています。Oracle Databaseによってアカウント・ステータスが更新されるためには、正しい資格証明が必要です。それ以外の場合、アカウント・ステータスは変更されません。Oracle Databaseには、アカウント・ステータスを更新するためのバックグラウンド・プロセスは存在しません。アカウント・ステータスの変更はすべて、認証されたユーザーにかわって、Oracle Databaseのサーバー・プロセスによって実行されます。
- フェーズ3: 最終的にユーザーがログインすると、猶予期間が開始されます。Oracle Databaseによって、現在の時間にそのアカウントのパスワード・プロファイルのPASSWORD_GRACE_TIME設定の値を加えた値が使用されて、DBA_USERS.EXPIRY_DATE列の値が更新されます。この時点でユーザーは、近い将来にパスワードが期限切れするというORA-28002警告メッセージ(たとえば、PASSWORD_GRACE_TIMEが7日に設定されている場合は、「ORA-28002 パスワードは、7日以内に期限切れになります。」)を受け取りますが、依然としてパスワー

ドを変更することなくログインできます。DBA_USERS.EXPIRY_DATE列には、ユーザーがパスワードを変更するよう求められる将来の時間が示されます。

- フェーズ4: 猶予期間(フェーズ3)が終了した後、ユーザーが現行の正しいパスワードを入力すると、認証が続行される前に「ORA-28001: パスワードが期限切れです。」エラーが表示されて、パスワードを変更するよう求められます。ユーザーが、Oracle Active Data Guard構成を使用しており(その場合は、プライマリ・データベースとスタンバイ・データベースが存在します)、認証がスタンバイ・データベース(読取り専用データベース)で試行された場合は、「ORA-28032: パスワードの期限が切れており、データベースは読取り専用に設定されています」エラーが表示されます。ユーザーは、プライマリ・データベースにログインして、そこでパスワードを変更する必要があります。

これら4フェーズのいずれの間も、DBA_USERSデータ・ディクショナリ・ビューに問い合せて、DBA_USERS.ACCOUNT_STATUS列でユーザーのアカウント・ステータスを検索できます。

次の例では、johndoeに割り当てられたプロファイルに、猶予期間PASSWORD_GRACE_TIME = 3(推奨値)が指定されています。johndoeは90日後に初めてデータベースにログインしようとすると(これは90日目より後の任意の日、つまり91日目や100日目でも構いません)、パスワードが3日で期限切れになるという警告メッセージを受け取ります。3日経過してもパスワードを変更しない場合は、パスワードの期限が切れます。その後、このユーザーがログインしようとするたびに、パスワードの変更を求めるプロンプトが表示されます。

```
CREATE PROFILE prof LIMIT
 FAILED_LOGIN_ATTEMPTS 4
 PASSWORD_LIFE_TIME 90
 PASSWORD_GRACE_TIME 3;
ALTER USER johndoe PROFILE prof;
```

データベース管理者またはALTER USERシステム権限を持つユーザーは、CREATE USER文およびALTER USER文を使用して、任意のパスワードを明示的に期限切れにできます。次の文は、期限切れのパスワードを持つユーザーを作成します。この設定によって、ユーザーがデータベースにログインする前に、強制的にパスワードを変更させることができます。

```
CREATE USER jbrown
 IDENTIFIED BY password
 ...
 PASSWORD EXPIRE;
```

CREATE USER文にパスワードの期限切れを解除する句はありませんが、アカウントのパスワードを変更することによって、期限切れは解除されます。

親トピック: アプリケーションの段階的データベース・パスワード・ロールオーバーの管理

3.2.5.3 段階的データベース・パスワード・ロールオーバーの有効化

段階的データベース・パスワード・ロールオーバーを有効にするには、PASSWORD_ROLLOVER_TIMEユーザー・プロファイル・パラメータを構成する必要があります。

- 段階的データベース・パスワード・ロールオーバーを構成するには、CREATE PROFILE文またはALTER PROFILE文でPASSWORD_ROLLOVER_TIMEパラメータを設定します。
  たとえば、段階的パスワード・ロールオーバー期間を1日に設定するには、次のようにします。

  ```
  CREATE PROFILE prof LIMIT
   ...
   PASSWORD_ROLLOVER_TIME 1;
  ```

  次のことに注意してください。

  - ロールオーバー期間は日数で指定しますが、必要に応じて時間を指定できます。たとえば、1/24と入力して1

時間を指定したり、6/24(または1/4)と入力して6時間を指定します。

- アクティブなロールオーバー時間の最小値は1時間です。最大値は60日か、PASSWORD_LIFE_TIMEまたはPASSWORD_GRACE_TIMEパラメータの小さいほうの値です。PASSWORD_GRACE_TIMEが0 (ゼロ)に設定されている場合、PASSWORD_ROLLOVER_TIMEによる制限に関して無視されます。次の表に、これらの制限を示します。

  表3-3 パスワード・ロールオーバー時間制限

プロファイル名	PASSWORD_LIFE_TIME	PASSWORD_GRACE_TIME	PASSWORD_ROLLOVER_TIME
デフォルト	180	7	● 最小: 1/24 (1時間) ● 最大: 7 (日)
ORA_STIG_PROFILE	60	5	● 最小: 1/24 (1時間) ● 最大: 5 (日)
ユーザー・カスタム・プロファイル	365	90	● 最小: 1/24 (1時間) ● 最大: 60 (日)

- PASSWORD_ROLLOVER_TIMEのデフォルト設定は0またはNULLで、これは無効化です。
- パスワード・ロールオーバー・プロセスに現在存在するデータベース・アカウントを確認するには、DBA_USERSデータ・ディクショナリ・ビューのACCOUNT_STATUS列を問い合せます。ステータスはIN ROLLOVERになります。
- パスワード・ロールオーバー期間は、管理者がデータベース・アカウントのパスワードを変更した時点から始まります。

親トピック: アプリケーションの段階的データベース・パスワード・ロールオーバーの管理

3.2.5.4 段階的データベース・パスワード・ロールオーバー期間を開始するためのパスワードの変更

ゼロ以外のPASSWORD_ROLLOVER_TIME値を設定した後、ユーザーのパスワードを変更し、すべてのアプリケーションでパスワードを更新します。

ALTER USER文を使用して、アプリケーションの新しいロールオーバー・パスワードをプロビジョニングします。ユーザーの新規パスワードがデータベースにプロビジョニングされたら、アプリケーション・サーバー上のパスワードを更新できます。PASSWORD_ROLLOVER_TIME期間が終了する前に、パスワードの更新を完了しておく必要があります。DBA_USERSデータ・ディクショナリ・ビューのACCOUNT_STATUS列を問い合せて、ユーザーのパスワード・ロールオーバー・ステータスを確認できます。ロールオーバー期間内のユーザー・アカウントのステータスはIN ROLLOVERになります。

- CREATE USER文およびALTER USER文を使用して、ユーザー、関連付けられたプロファイルおよびパスワード・ロールオーバー期間を構成します。CREATE USERを使用すると、管理者はパスワード・ロールオーバー付きのプロファイルに関連付けられた新しいアプリケーション・サービス・アカウントを作成できます。ALTER USERは、既存のユーザーに新規または変更されたプロファイルを関連付けるケースに適しています。プロファイルを変更するには、ALTER PROFILE

文を使用します。

次の例のCREATE USERでは、パスワードp1とプロファイルprof1を使用して、新しいユーザーu1をPASSWORD_ROLLOVER_TIME付きで構成します。ALTER USER文は、ユーザーのパスワードを変更してパスワードのロールオーバー期間を開始します。ユーザー・ステータスを確認するには、DBA_USERSデータ・ディクショナリ・ビューを問い合せます。

1. プロファイルprof1を作成します。

```
CREATE PROFILE prof1
LIMIT
PASSWORD_ROLLOVER_TIME 1;
```

2. ユーザーu1を作成し、このユーザーをprof1プロファイルに関連付けます。

```
CREATE USER u1 IDENTIFIED BY p1 PROFILE prof1;
```

3. ユーザーのパスワードを変更します。

```

ALTER USER u1 IDENTIFIED BY p2;
```

4. DBA_USERデータ・ディクショナリ・ビューを問い合せて、ユーザーのロールオーバー・ステータスを確認します。

```
SELECT USERNAME, ACCOUNT_STATUS FROM DBA_USERS WHERE USERNAME = 'U1';
USERNAME ACCOUNT_STATUS
-------- ------------------
U1       OPEN & IN ROLLOVER
```

親トピック: アプリケーションの段階的データベース・パスワード・ロールオーバーの管理

3.2.5.5 段階的データベース・パスワード・ロールオーバー期間中のパスワードの変更

ロールオーバー期間の開始後も、パスワードを変更できます。

たとえば、誤ってパスワードを入力したとします。次の手順では、ロールオーバー・プロセスがすでに開始されている場合でも、パスワードを修正できます。

- ロールオーバー・プロセスの開始後にパスワードを変更するには、REPLACE句を指定して、または指定せずにALTER USER文を使用します。
  たとえば、ユーザーu1が元のパスワードp1を使用しており、ロールオーバー・プロセスを開始した新しいパスワードがp2のときに、パスワードp2のかわりに別のパスワードp3を使用するように切り替えるとします。次の文のいずれかが機能します。

```
ALTER USER u1 IDENTIFIED BY p3;
ALTER USER u1 IDENTIFIED BY p3 REPLACE p1;
ALTER USER u1 IDENTIFIED BY p3 REPLACE p2;
```

パスワードをp3に変更した後、ユーザーはp1またはp3を使用してログインできます。p2を使用してログインしようとすると、「ORA-1017 無効なユーザー名/パスワード」エラーが返され、失敗したログイン試行として記録されます。同様に、ロールオーバー期間中にパスワードが次にp3からp4に変更された後、ユーザーはp1またはp4のいずれかを使用してログインできます。p2またはp3を使用してログインしようとすると、「ORA-1017 無効なユーザー名/パスワード」エラーが返され、失敗したログイン試行として記録されます。

ロールオーバーの開始時間は、ユーザーがパスワードを初めて変更したときに固定されます。パスワードのロールオーバー期間中にさらにパスワードを変更しても、開始時間は影響を受けません。この設計により、パスワードがパスワード・ロールオーバー期間外に変更された後、古いパスワードをPASSWORD_ROLLOVER_TIME期間に使用できる期間が制限されます。

親トピック: アプリケーションの段階的データベース・パスワード・ロールオーバーの管理

3.2.5.6 パスワード・ロールオーバー期間の終了

パスワード・ロールオーバー期間を終了するには、複数の方法があります。

たとえば、p1がユーザーu1の元のパスワードであり、p2がすべてのクライアントに更新された新しいパスワードであるとします。

- 次のいずれかの方法で、パスワード・ロールオーバー期間を終了します。
  - パスワード・ロールオーバー期間をそのまま期限切れにします。たとえば、パスワード・ロールオーバー期間が1日の場合、1日待機すると、パスワード・ロールオーバー期間は自動的に期限切れになります。
  - ユーザーまたは管理者として、次の文を実行して、パスワード・ロールオーバー期間を手動で終了します。

    ```
    ALTER USER u1 EXPIRE PASSWORD ROLLOVER PERIOD;
    ```

  - 管理者として、ALTER USER username PASSWORD EXPIRE文を実行してパスワードを期限切れにします。ユーザーが次にログインすると、パスワードの変更を求められます。

パスワード・ロールオーバー期間が経過した後の最初の接続試行以降、Oracle Databaseは以前のパスワードp1を削除します。古いパスワードp1を使用してログインしようとすると、「ORA-1017 無効なユーザー名/パスワード」エラーが返され、失敗したログイン試行として記録されます。実際には、ロールオーバー期間後の接続は新しいパスワードでのみ認証され、古いパスワードで試行された接続は失敗したログイン試行として記録されます。ログイン試行の失敗が、古いパスワードを使用した連続ログオン試行が一定回数を超えると、アカウントがロックされる可能性があります。

PASSWORD_ROLLOVER_TIMEの有効期限が切れた後に読取り専用データベース・サーバーへの接続を試行するには、新しいパスワード(p2)が必要です。p2へのパスワード変更は、すべてのデータベース・クライアントに対して有効になります。

親トピック: アプリケーションの段階的データベース・パスワード・ロールオーバーの管理

3.2.5.7 段階的パスワード・ロールオーバー期間中のデータベース動作

ユーザーは、パスワード・ロールオーバー期間中に標準のパスワード変更およびログインを実行できます。

ロールオーバー期間中には、次のデータベース動作が実装されます。

- ユーザーは、新しいパスワードまたは古いパスワードのいずれかを使用してデータベースにログインできます。これにより、PASSWORD_ROLLOVER_TIMEで設定された時間だけ、古いパスワードの存続期間が事実上長くなります。
- パスワードは、次の方法で変更できます。
  - 管理者またはユーザーは、ALTER USER文を使用して自分のパスワードを変更します。
  - ユーザーは、SQL*Plusのpasswordコマンドを使用して自分のパスワードを変更します。
  - ユーザーのパスワードは、Oracle Call Interface (Oracle OCI)のOCIPasswordChange関数の実行時にプログラムによって変更されます。
- Oracle Databaseは、ユーザー・アカウントがパスワード・ロールオーバー期間内であることを示す特別なメッセージをデータベース・クライアントに送信しません。この設計により、ユーザーがログインしたときのエラーおよび警告メッセージを処理するように設定されていない可能性のあるアプリケーションからのエラーを回避できます。
- 失敗したログイン試行が多すぎると、プロファイル制限PASSWORD_LOCK_TIMEの値に応じて、ユーザー・アカウントがタイム・ロック状態に移行します。タイム・ロック期間が経過すると、パスワード・ロールオーバー期間の状態によって、ユーザーがログインを試行したときの処理が決まります。
- ユーザー管理者は、ACCOUNT LOCK、ACCOUNT UNLOCK、EXPIRE PASSWORD操作など、他のパスワード・ライフサイクル関連のアクションを通常どおりに実行できます。
- ユーザー・プロファイルのPASSWORD_REUSE_TIMEおよびPASSWORD_REUSE_MAXによって設定されたパスワード

制限は、ロールオーバー期間中も引き続き適用されます。ロールオーバー期間中のパスワード変更は、パスワード変更履歴に対して検証され、パスワード変更履歴に追加されます。

- ユーザー・アカウントを期限切れにしても、パスワードのロールオーバー・ステータスには影響しません。ロックされたアカウントと同様に、Oracle Databaseはパスワード・ベリファイアを現在の状態で保持します。ユーザーは、古いパスワードまたは新しいパスワード(p1またはp2)を使用してログインできます。ただし、ユーザーがパスワードを正常に(p3に)変更した後、ユーザーは最新のパスワード(p3)のみを使用してログインできます。古いパスワードは両方とも期限切れとして扱われます。
- Oracle Data Pumpは、パスワード・ロールオーバー期間中のユーザー・アカウントの最新パスワードのパスワード・ハッシュ(ベリファイアとも呼ばれる)をエクスポートします。たとえば、ユーザーu1に古いパスワードp1と新しいパスワードp2がある場合、Oracle Data Pumpはパスワードp2のパスワード・ハッシュのみをエクスポートします。

親トピック: アプリケーションの段階的データベース・パスワード・ロールオーバーの管理

3.2.5.8 パスワード・ロールオーバー期間終了後のデータベース・サーバーの動作

Oracle Databaseは、段階的データベース・パスワード・ロールオーバー期間の終了後にクリーン・アップ操作を実行します。

パスワード・ロールオーバー期間が終了すると、新しいパスワードのみが許可され、古いパスワードは機能しなくなります。古いパスワードを使用しようとすると、「ORA-1017 無効なユーザー名/パスワード」エラーが返され、失敗したログイン試行として記録されます。パスワード・ロールオーバー期間後の接続では新しいパスワードのみが使用され、以前のパスワードを使用しようとすると、読取り専用データベースと読取り/書込みデータベースの両方で失敗します。ログイン試行に失敗すると、パスワード・プロファイルのFAILED_LOGIN_ATTEMPTS制限に基づいて、古いパスワードを使用した連続するログイン試行回数に応じて、ユーザー・アカウントがロックされる可能性があります。

親トピック: アプリケーションの段階的データベース・パスワード・ロールオーバーの管理

3.2.5.9 漏えいしたパスワードの処理に関するガイドライン

データベース・アカウント・パスワードが漏えいしたと思われる場合は、すぐにパスワードを変更する必要があります。

2つのコマンドを順番に実行するのではなく、1回の実行でALTER USER文を使用して、古いパスワードを変更および期限切れにすることによって、パスワード・ロールオーバー期間を経由せずにこの変更を実行できます。このオプションは、他のアカウントが影響を受けるため、関連するユーザー・プロファイルのPASSWORD_ROLLOVER_TIMEの変更よりも優先されます。

次の構文を使用して、古いパスワードを変更および期限切れにします。

```
ALTER USER user_name IDENTIFIED BY new_password EXPIRE PASSWORD ROLLOVER PERIOD;
```

親トピック: アプリケーションの段階的データベース・パスワード・ロールオーバーの管理

3.2.5.10 Oracle Data Pumpのエクスポート時の段階的データベース・パスワード・ロールオーバーの動作

パスワード・ロールオーバー期間中のユーザーがエクスポートされた場合、そのユーザーの新しいパスワードに対応するベリファイアのみがエクスポートされます。

古いパスワードに対応するベリファイアは、Oracle Data Pumpダンプ・ファイルには含まれません。ユーザーがインポートされた後は、認証には新しいパスワードのみを使用できます。

親トピック: アプリケーションの段階的データベース・パスワード・ロールオーバーの管理

3.2.5.11 Oracle Data Guard環境での段階的データベース・パスワード・ロールオーバーの使用

Oracle Data Guard環境で、段階的データベース・パスワード・ロールオーバーを使用するには、ADG_ACCOUNT_INFO_TRACKING環境変数をGLOBALに設定する必要があります。

```
ADG_ACCOUNT_INFO_TRACKING=GLOBAL
```

それ以外の場合、PASSWORD_ROLLOVER_TIMEの有効期限後にロールオーバー・パスワードを使用しているユーザーによってOracle Data Guardスタンバイで初期ログオンが実行されると、「ORA-16000: データベースまたはプラガブル・データベースは読取り専用アクセスでオープンされています」エラーになります。

親トピック: アプリケーションの段階的データベース・パスワード・ロールオーバーの管理

3.2.5.12 古いパスワードをまだ使用しているユーザーの検索

LOGIN監査レコードのAUTHENTICATION_TYPEフィールドを使用する問合せを実行して、古いパスワードをまだ使用しているユーザーを検索できます。

統合監査証跡では、古いパスワードを使用してデータベースにまだ接続しているユーザーを識別できます。LOGON監査レコードのAUTHENTICATION_TYPEフィールドは、古いベリファイアが使用されたかどうかを示すことができます。この情報を使用すると、段階的なデータベース・パスワードのロールオーバーで更新されていないアプリケーションを検索して、新しいパスワードを使用できます。LOGON監査レコードは、どのアプリケーション・サーバーを更新する必要があるかを示します。

1. AUDIT_VIEWERまたはAUDIT_MGMTロールを持つユーザーとしてデータベースに接続します。
2. 次の問合せを実行します。

```
SELECT DBUSERNAME, AUTHENTICATION_TYPE, OS_USERNAME, USERHOST, EVENT_TIMESTAMP
FROM UNIFIED_AUDIT_TRAIL
WHERE ACTION_NAME='LOGON' AND EVENT_TIMESTAMP > SYSDATE-1
AND REGEXP_LIKE(AUTHENTICATION_TYPE, '¥(VERIFIER=.*?¥-OLD¥)');
```

古いパスワードをまだ使用しているユーザーがいる場合は、次のような出力が表示されます。

```
DBUSERNAME     AUTHENTICATION_TYPE
OS_USERNAME     USERHOST     EVENT_TIMESTAMP
_____________
_____________________________________________________________________________
_____________________________________________________________________________
__ _____________ ___________ ____________________________________________
APP_USER       (TYPE=(DATABASE));(CLIENT
ADDRESS=((PROTOCOL=tcp)(HOST=192.0.2.225)(PORT=24938)));(LOGON_INFO=((VERIFIER=
12C-OLD)(CLIENT_CAPABILITIES=O5L_NP,O7L_MR,O8L_LI)));    oracle          db211
14-JAN-21 08.56.34.724172000 PM
APP_USER       (TYPE=(DATABASE));(CLIENT
ADDRESS=((PROTOCOL=tcp)(HOST=192.0.2.225)(PORT=24983)));(LOGON_INFO=((VERIFIER=
12C-OLD)(CLIENT_CAPABILITIES=O5L_NP,O7L_MR,O8L_LI)));    oracle          db211
14-JAN-21 09.01.18.938008000 PM
APP_USER       (TYPE=(DATABASE));(CLIENT
ADDRESS=((PROTOCOL=tcp)(HOST=192.0.2.226)(PORT=48727)));(LOGON_INFO=((VERIFIER=
12C-OLD)(CLIENT_CAPABILITIES=O5L_NP,O7L_MR,O8L_LI)));    oracle          db212
14-JAN-21 10.10.48.042817000 PM
APP_USER       (TYPE=(DATABASE));(CLIENT
ADDRESS=((PROTOCOL=tcp)(HOST=192.0.2.226)(PORT=48745)));(LOGON_INFO=((VERIFIER=
12C-OLD)(CLIENT_CAPABILITIES=O5L_NP,O7L_MR,O8L_LI)));    oracle          db212
14-JAN-21 10.12.53.609965000 PM
APP_USER       (TYPE=(DATABASE));(CLIENT
ADDRESS=((PROTOCOL=tcp)(HOST=192.0.2.226)(PORT=48751)));(LOGON_INFO=((VERIFIER=
12C-OLD)(CLIENT_CAPABILITIES=O5L_NP,O7L_MR,O8L_LI)));    oracle          db212
14-JAN-21 10.13.41.112194000 PM
```

親トピック: アプリケーションの段階的データベース・パスワード・ロールオーバーの管理

3.2.6 パスワードの複雑度の管理

Oracle Databaseには、パスワードの複雑度の管理に使用できる一連の関数があります。

- パスワードの複雑度検証について
  複雑度検証では、パスワードを推定してシステムに入ろうとする侵入者から保護できるだけの複雑性が各パスワードに備わっているかがチェックされます。
- Oracle Databaseによるパスワードの複雑度のチェック方法
  Oracle Databaseにはパスワードの複雑度をチェックするパスワード検証機能が4つ用意されています。
- パスワード複雑度ファンクションを使用できるユーザー
  パスワード複雑度ファンクションを使用すると、ユーザーがデータにアクセスする方法をカスタマイズできます。
- verify_function_11Gファンクションのパスワード要件
  `verify_function_11G`ファンクションは、Oracle Databaseリリース11gから開始されました。
- ora12c_verify_functionのパスワード要件
  `ora12c_verify_function`ファンクションは、Department of Defense Database Security Technical Implementation Guideの要件を満たしています。
- ora12c_strong_verify_function関数のパスワード要件
  `ora12c_strong_verify_function`関数は、厳密なパスワード検証関数です。
- ora12c_stig_verify_functionのパスワード要件
  `ora12c_stig_verify_function`ファンクションは、『Security Technical Implementation Guides (STIG)』の要件を満たしています。
- パスワード複雑度検証のカスタマイズについて
  Oracle Databaseを使用すると、サイトのパスワード複雑度をカスタマイズできます。
- パスワード複雑度検証の有効化
  `catpvf.sql`スクリプトをカスタマイズして、パスワードの複雑度検証を有効にできます。

親トピック: パスワード保護の構成

3.2.6.1 パスワードの複雑度検証について

複雑度検証では、パスワードを推定してシステムに入ろうとする侵入者から保護できるだけの複雑性が各パスワードに備わっているかがチェックされます。

複雑度検証ファンクションを使用すると、データベース・ユーザー・アカウントに対して強力で安全性の高いパスワードが強制的に作成されます。ユーザーのパスワードは、パスワードを推定してシステムに入ろうとする侵入者に対して、十分保護可能な複雑なものである必要があります。

親トピック:パスワードの複雑度の管理

3.2.6.2 Oracle Databaseによるパスワードの複雑度のチェック方法

Oracle Databaseにはパスワードの複雑度をチェックするパスワード検証機能が4つ用意されています。

これらの関数は、(`$ORACLE_HOME/rdbms/admin`にある)`catpvf.sql` PL/SQLスクリプト内にあります。これらの関数が有効になっている場合、ユーザーがパスワードを正しく作成または変更したかどうかを確認できます。有効にすると、パスワードの複雑度のチェックはユーザーSYSに対して適用されず、SYS以外のユーザーにのみ適用されます。パスワードのセキュリティを強化するには、パスワード検証ファンクションとデフォルト・プロファイルを関連付けることをお薦めします。パスワード複雑度検証のカ

スタマイズについては、これを実行する方法の例が示されています。

親トピック:パスワードの複雑度の管理

3.2.6.3 パスワード複雑度ファンクションを使用できるユーザー

パスワード複雑度ファンクションを使用すると、ユーザーがデータにアクセスする方法をカスタマイズできます。

パスワード複雑度検証ファンクションをCREATE PROFILE文またはALTER PROFILE文で使用する前に、そのファンクションに対するEXECUTE権限が付与される必要があります。

パスワード検証ファンクションは、SYSスキーマにあります。

親トピック:パスワードの複雑度の管理

3.2.6.4 verify_function_11G関数のパスワード要件

verify_function_11Gファンクションは、Oracle Databaseリリース11gから開始されました。

> ノート:
>
> verify_function_11G ファンクションは、Oracle Database の以前のリリースの弱いパスワード制限が強制されるため、非推奨となりました。かわりに、より強力かつ最新のパスワード検証制限を強制する ORA12C_VERIFY_FUNCTION、ORA12C_STRONG_VERIFY_FUNCTION、ORA12C_STIG_VERIFY_FUNCTION ファンクションを使用することが望まれます。

このファンクションでは、ユーザーがパスワードを作成または変更したときに、次の要件をチェックします。

- パスワードの長さが8文字以上であり、少なくとも数字が1つと英字が1つ含まれていること。
- パスワードがユーザー名と同一でないこと。ユーザー名のスペルを逆にしたり、ユーザー名に数字1から100を追加したパスワードでないこと。
- サーバー名と同一であったり、サーバー名に数字1から100を追加したパスワードでないこと。
- パスワードにoracleが含まれていないこと(たとえば、oracleに数字1から100を追加したものなど)。
- パスワードを単純にしすぎないこと(たとえば、welcome1、database1、account1、user1234、password1、oracle123、computer1、abcdefg1またはchange_on_install)。
- 以前のパスワードとの違いが3文字以上あること。

次の内部チェックも適用されます。

- パスワードが二重引用符文字(")を含まないこと。ただし、二重引用符で囲むことができます。

親トピック:パスワードの複雑度の管理

3.2.6.5 ora12c_verify_functionのパスワード要件

ora12c_verify_functionファンクションは、Department of Defense Database Security Technical Implementation Guideの要件を満たしています。

このファンクションでは、ユーザーがパスワードを作成または変更したときに、次の要件をチェックします。

- パスワードの長さが8文字以上であり、少なくとも数字が1つと英字が1つ含まれていること。

- パスワードがユーザー名またはユーザー名のスペルを逆にしたものと同一でないこと。
- パスワードがデータベース名と同一でないこと。
- パスワードにoracle (oracle123など)の語が含まれていないこと。
- 以前のパスワードとの違いが3文字以上あること。
- パスワードに少なくとも1つの特殊文字が含まれていること。

次の内部チェックも適用されます。

- パスワードが二重引用符文字(")を含まないこと。ただし、二重引用符で囲むことができます。

親トピック:パスワードの複雑度の管理

3.2.6.6 ora12c_strong_verify_function関数のパスワード要件

ora12c_strong_verify_function関数は、厳密なパスワード検証関数です。

このファンクションでは、ユーザーがパスワードを作成または変更したときに、次の要件をチェックします。

- パスワードが9文字以上である。
- パスワードに大文字が2つ以上含まれている。
- パスワードに小文字が2つ以上含まれている。
- パスワードに少なくとも2つの数字が含まれている。
- パスワードに少なくとも2つの特殊文字が含まれている。これらの特殊文字は次のとおりです。

```
‘ ~ ! @ # $ % ^ & * ( ) _ - + = { } [ ] ¥ / < > , . ; ? ' : | (space)
```

- 以前のパスワードとの違いが4文字以上あること。

次の内部チェックも適用されます。

- パスワードが二重引用符文字(")を含まないこと。ただし、二重引用符で囲むことができます。

親トピック:パスワードの複雑度の管理

3.2.6.7 ora12c_stig_verify_functionのパスワード要件

ora12c_stig_verify_functionファンクションは、Security Technical Implementation Guides (STIG)の要件を満たしています。

このファンクションでは、ユーザーがパスワードを作成または変更したときに、次の要件をチェックします。

- パスワードが15文字以上であること。
- パスワードに少なくとも1文字以上の小文字と、1文字以上の大文字が含まれていること。
- パスワードに少なくとも1つの数字が含まれていること。
- パスワードに少なくとも1つの特殊文字が含まれていること。
- 以前のパスワードとの違いが8文字以上あること。

次の内部チェックも適用されます。

- パスワードが二重引用符文字(")を含まないこと。ただし、二重引用符で囲むことができます。

ora12c_stig_verify_functionファンクションはORA_STIG_PROFILEプロファイルのデフォルト・ハンドラであり、新しく作成されたOracleデータベースまたはアップグレードされたOracleデータベースで使用できます。

親トピック:パスワードの複雑度の管理

3.2.6.8 パスワード複雑度検証のカスタマイズについて

Oracle Databaseを使用すると、サイトのパスワード複雑度をカスタマイズできます。

admin/catpvf.sqlに定義される関数と同様、SYSスキーマに独自のパスワードの複雑度検証関数を作成できます。サイトのパスワードの保護を強化するために、独自のパスワード複雑度検証関数を作成することをお薦めします。

次のことに注意してください。

- データ定義言語(DDL)文は、カスタムのパスワード複雑度検証関数に含めないでください。パスワード複雑度検証関数の実行中、DDLは使用できません。
- admin/catpvf.sqlスクリプトまたはOracle提供のパスワード複雑度関数は変更しないでください。これらのファイルの内容に基づいて、独自の関数を作成できます。
- utlpwdmg.sqlスクリプトを変更しない場合、デフォルト関数としてora12c_verify_function関数を使用します。

関連項目:

パスワードの作成に関するガイドラインは、「パスワードの保護に関するガイドライン」のガイドライン1を参照してください。

親トピック:パスワードの複雑度の管理

3.2.6.9 パスワード複雑度検証の有効化

catpvf.sqlスクリプトをカスタマイズして、パスワードの複雑度検証を有効にできます。

パスワード複雑度検証を有効にするには、必要なパスワード検証ファンクションを使用するようにcatpvf.sqlスクリプトを編集し、そのスクリプトを実行して検証を有効にする必要があります。

1. 管理者権限を使用してSQL*Plusにログインします。

   たとえば:

   ```
   CONNECT SYSTEM
   Enter password: password
   ```

2. catpvf.sqlスクリプト(またはこのスクリプトの変更されたバージョン)を実行して、SYSスキーマのパスワード複雑度関数を作成します。

   ```
   @$ORACLE_HOME/rdbms/admin/catpvf.sql
   ```

3. このファンクションを使用する必要があるユーザーに、ファンクションに対するEXECUTE権限を付与します。

   たとえば:

   ```
   GRANT pmsith EXECUTE ON ora12c_strong_verify_function;
   ```

4. デフォルト・プロファイルまたはユーザー・プロファイルで、PASSWORD_VERIFY_FUNCTION設定を、catpvf.sqlスクリプトのサンプルのパスワード複雑度関数か、カスタマイズした関数に設定します。次のいずれかの方法を使用します。

- 管理者権限でSQL*Plusにログインし、CREATE PROFILE文またはALTER PROFILE文を使用して関数を使用可能にします。ファンクションに対するEXECUTE権限があることを確認します。

  たとえば、デフォルト・プロファイルを更新してora12c_strong_verify_function関数を使用するには、次のようにします。

```
ALTER PROFILE default LIMIT
 PASSWORD_VERIFY_FUNCTION ora12c_strong_verify_function;
```

- Oracle Enterprise Manager Cloud Controlで、「管理」メニューから、「セキュリティ」を選択し、「プロファイル」を選択します。「パスワード」タブを選択します。「複雑なパスワード検証」で、「複雑なパスワード検証のための関数」リストから、使用する複雑度関数の名前を選択します。「適用」をクリックします。

パスワード複雑度検証は、使用可能にするとすぐに有効になります。無効にする必要がある場合、次の文を実行します。

```
ALTER PROFILE DEFAULT LIMIT PASSWORD_VERIFY_FUNCTION NULL;
```

ノート:

ALTER USER 文で REPLACE 句を使用できます。ユーザーは、この句を使用して、自身を認証するために以前のパスワードを指定し、期限が切れていない自分のパスワードを変更できます。

パスワードが期限切れになると、ユーザーは SQL にログインして ALTER USER コマンドを発行できません。かわりに OCIPasswordChange( )関数を使用しますが、この場合は以前のパスワードも必要になります。

ALTER ANY USER 権限が付与されているデータベース管理者は、古いパスワードを指定せずにユーザーのパスワードを変更(新しいパスワードを適用)できます。

親トピック:パスワードの複雑度の管理

3.2.7 パスワードでの大/小文字の区別の管理

以前のリリースのユーザー・アカウントのパスワードに対して、パスワードの大/小文字の区別を管理できます。

- SEC_CASE_SENSITIVE_LOGONパラメータおよびパスワードの大/小文字の区別
  SEC_CASE_SENSITIVE_LOGON初期化パラメータは、パスワードでの大/小文字の区別の使用を制御します。
- ALTER SYSTEM文を使用したパスワードの大/小文字の区別の有効化
  パスワードの大/小文字の区別が無効化されている場合、有効化するには、SEC_CASE_SENSITIVE_LOGONパラメータをTRUEに設定します。
- 安全性の高いロール・パスワードの大/小文字の区別の管理
  セキュリティを強化するために、安全性の高いロールのためのパスワードは、大/小文字が区別されるようにする必要があります。
- ユーザーのパスワード・バージョンの管理
  デフォルトでは、Oracle Databaseはパスワード・バージョンの管理に排他モード(大/小文字を区別しないパスワードは許可されない)を使用します。
- 10Gパスワード・バージョンを使用するユーザー・パスワードの確認と再設定
  よりセキュアなパスワード認証を行うためには、10Gパスワード・バージョンを使用するユーザー・アカウントのパスワードを確認して再設定し、より新しい、よりセキュアなパスワード・バージョンを使用するようにします。

- 大/小文字の区別がパスワード・ファイルに与える影響
  デフォルトでは、パスワード・ファイルは大/小文字を区別します。ORAPWDコマンドライン・ユーティリティのIGNORECASE引数は、パスワード・ファイルの大/小文字の区別を制御します。
- 大/小文字の区別がデータベース・リンク接続で使用されるパスワードに与える影響
  データベース・リンク接続を作成する場合は、接続用のユーザー名とパスワードを定義する必要があります。

親トピック: パスワード保護の構成

3.2.7.1 SEC_CASE_SENSITIVE_LOGONパラメータおよびパスワードの大/小文字の区別

SEC_CASE_SENSITIVE_LOGON初期化パラメータは、パスワードでの大/小文字の区別の使用を制御します。

SEC_CASE_SENSITIVE_LOGONパラメータを設定できるのは、ALTER SYSTEM権限を持つユーザーのみです。このパラメータがTRUEに設定されていて、ユーザーがパスワードを入力するときに大/小文字の区別が適用されることを確認する必要があります。(ただし、SEC_CASE_SENSITIVE_LOGONパラメータが非推奨ですが、下位互換性のために現在残されていることに注意してください。)

ユーザー・アカウントを作成または変更するとき、パスワードはデフォルトで大/小文字の区別があります。大/小文字の区別は、ユーザーが手動で入力するパスワードのみでなく、パスワード・ファイルにも影響を与えます。

SQLNET.ALLOWED_LOGON_VERSION_SERVERパラメータが12または12aに設定されている場合は、SEC_CASE_SENSITIVE_LOGONパラメータがFALSEに設定されていないことを確認します。これは、このモードで使用されているセキュリティ度の強いパスワード・バージョンでは、大文字/小文字を区別するパスワード・チェックのみがサポートされているためです。互換性上の理由により、Oracle DatabaseではSQLNET.ALLOWED_LOGON_VERSION_SERVERが12または12aに設定されているときにSEC_CASE_SENSITIVE_LOGONでFALSEの使用が禁止されてはいません。SQLNET.ALLOWED_LOGON_VERSION_SERVERが12または12aに設定されている場合にSEC_CASE_SENSITIVE_LOGONをFALSEに設定すると、すべてのアカウントがアクセス不能になります。SQLNET.ALLOWED_LOGON_VERSION_SERVERが11以下の値に設定されている場合は、Oracle Database 12cの排他モード(SQLNET.ALLOWED_LOGON_VERSION_SERVERが12または12aの場合)で使用されるセキュリティ度の強いパスワード・バージョンが大文字/小文字を区別しないパスワード照合をサポートしないため、SEC_CASE_SENSITIVE_LOGONをTRUEに設定することをお薦めします。

サーバー側の設定に加えて、ユーザーが接続しているクライアント・ソフトウェアにO5L_NP機能フラグがあることを確認する必要があります。Oracle Databaseリリース11.2.0.3以上のすべてのクライアントにO5L_NP機能があります。以前のクライアントがある場合は、CPUOct2012パッチをインストールする必要があります。

親トピック: パスワードの大/小文字の区別の管理

3.2.7.2 ALTER SYSTEM文を使用したパスワードの大/小文字の区別の有効化

パスワードの大/小文字の区別が無効化されている場合、有効化するには、SEC_CASE_SENSITIVE_LOGONパラメータをTRUEに設定します。

1. パスワード・ファイルを使用している場合、ORAPWDユーティリティのIGNORECASEパラメータがNに設定されて作成され、FORMATパラメータが12に設定されていることを確認します。

   IGNORECASEパラメータによりSEC_CASE_SENSITIVE_LOGONパラメータが上書きされます。デフォルトでは、IGNORECASEはパスワードの大/小文字が区別されることを意味するNに設定されます。

   IGNORECASEパラメータおよびSEC_CASE_SENSITIVE_LOGONシステム・パラメータは非推奨なので注意してください。IGNORECASEをNに設定するか、IGNORECASE設定全体を省略することをお薦めします。

2. 次のALTER SYSTEM文を入力します。

```
ALTER SYSTEM SET SEC_CASE_SENSITIVE_LOGON = TRUE;
```

関連トピック

- Oracle Database管理者ガイド

親トピック: パスワードの大/小文字の区別の管理

3.2.7.3 安全性の高いロール・パスワードの大/小文字の区別の管理

セキュリティを強化するために、安全性の高いロールのためのパスワードは、大/小文字が区別されるようにする必要があります。

Oracle Database 12cリリース2 (12.2)にアップグレードする前に、CREATE ROLE文のIDENTIFIED BY句を使用して安全性の高いロールを作成した場合、およびOracle Database 12cリリース12.2へのアップグレード時にSQLNET.ALLOWED_LOGON_VERSION_SERVERパラメータを排他モード12または12aのいずれかに設定した場合、これらの安全性の高いロールを引き続き使用可能にするには、パスワードを変更する必要があります。現在排他モードがデフォルトであるため、以前のリリース(10Gパスワード・バージョンがデフォルトのOracle Database 10gなど)で作成された安全性の高いロールについては、そのパスワードを変更する必要があります。

DBA_ROLESデータ・ディクショナリ・ビューのPASSWORD_REQUIREDおよびAUTHENTICATION_TYPE列を問い合せて、再度使用可能にするためにOracle Database 12cへのアップグレード後にパスワードを変更する必要がある安全性の高いロールを確認できます。

それ以外の場合、SQLNET.ALLOWED_LOGON_VERSION_SERVERパラメータを8に設定しないかぎり、これらの安全性の高いロールのパスワード・バージョンを使用できません。このパラメータが12または12aに設定されている場合、次のSQL文を実行して、大/小文字の区別が有効であることを確認する必要があります。そうしないと、パスワードを変更した後でも、安全性の高いロールを使用できません。

```
ALTER SYSTEM SET SEC_CASE_SENSITIVE_LOGON = "TRUE";
```

親トピック: パスワードの大/小文字の区別の管理

3.2.7.4 ユーザーのパスワード・バージョンの管理

デフォルトでは、Oracle Databaseはパスワード・バージョンの管理に排他モード(大/小文字を区別しないパスワードは許可されない)を使用します。

デフォルトのインストールでは、排他モードを有効にするため、SQLNET.ALLOWED_LOGON_VERSION_SERVERパラメータが12に設定されています。排他モードでは、パスワードベースの認証プロトコルで大/小文字を区別するパスワード・バージョン(11Gまたは12C)のいずれかを使用してアカウントを認証することが必要です。排他モードでは、以前のリリースで使用されていた10Gパスワード・バージョンの使用が除外されます。Oracle Database 12cリリース2 (12.2)にアップグレードすると、10Gパスワード・バージョンを使用するアカウントはアクセス不能になります。これは、サーバーはデフォルトで排他モードで実行され、排他モードではクライアントの認証に古い10Gパスワード・バージョンを使用できないためです。サーバーにはそのクライアントの認証に使用するパスワード・バージョンがありません。

リリース10gのユーザー・アカウントは、10Gパスワード・バージョンを使用します。したがって、10Gパスワード・バージョンを使用するユーザー・アカウントを特定して、これらのアカウントのパスワードを再設定する必要があります。これにより、SQLNET.ALLOWED_LOGON_VERSION_SERVERパラメータの設定に基づく適切なパスワード・バージョンが次のように生成されます。

- SQLNET.ALLOWED_LOGON_VERSION_SERVER=8は、3つのすべてのパスワード・バージョン10G、11Gおよび

12Cを生成します。

- SQLNET.ALLOWED_LOGON_VERSION_SERVER=12は、11Gおよび12Cパスワード・バージョンを生成し、10Gパスワード・バージョンを削除します。
- SQLNET.ALLOWED_LOGON_VERSION_SERVER=12aは、12Cパスワード・バージョンのみ生成します。

最初にSQLNET.ALLOWED_LOGON_VERSION_SERVER設定をより緩やかな値(SQLNET.ALLOWED_LOGON_VERSION_SERVER=8など)に緩和してから、Oracle Databaseリリース10g (またはそれ以前の)リリースから現在のデータベース・リリースにユーザー・アカウントをインポートする場合、(古いリリースで使用されていた)10Gパスワード・バージョンは大/小文字を区別しないため、これらのユーザーは引き続き大/小文字を区別しないパスワードを使用してデータベースにログインできます。ただし、そうしたユーザーが自分のパスワードを変更する場合は、インスタンス初期化パラメータSEC_CASE_SENSITIVE_LOGONのデフォルト値がTRUEのため、新しい11Gおよび12Cパスワード・バージョンが自動的に生成され、そのユーザーのパスワードは自動的に大/小文字が区別されるようになります。
(SEC_CASE_SENSITIVE_LOGONは非推奨ですが、下位互換性のため現在も維持されていることに注意してください。)

次の例は、SEC_CASE_SENSITIVE_LOGONパラメータをTRUEに設定した場合の影響を示しています。このシナリオでは、ユーザーrtaylorはOracle Database release 10gからインポートされたため、このアカウントのパスワード・バージョンは10Gのみです。サーバーでは、SQLNET.ALLOWED_LOGON_VERSION_SERVERが8に設定されています。そうしないと、rtaylorがログインできなくなるためです。また、SEC_CASE_SENSITIVE_LOGONパラメータはTRUEに設定され、11Gおよび12Cパスワード・バージョン用に大/小文字の区別が有効になっています。

1. ユーザーrtaylorのパスワード・バージョンを確認します。

```
SELECT PASSWORD_VERSIONS FROM DBA_USERS WHERE USERNAME='RTAYLOR';
PASSWORD VERSIONS
-----------------
10G
```

2. ユーザーrtaylorとして接続します。

```
CONNECT rtaylor
Enter password: "MaresEatOats"
Connected.
```

ユーザーrtaylorは、まだ自分のパスワードに大/小文字を区別しない10Gパスワード・バージョンを使用しているため、データベースに接続できます。ここで、実際のパスワードはすべて小文字のmareseatoatsですが、大文字と小文字を使用してパスワードを入力しています。

3. デフォルト・ユーザーの1人であるSCOTTのパスワード・バージョンを確認します。

```
SELECT PASSWORD_VERSIONS FROM DBA_USERS WHERE USERNAME='SCOTT';
PASSWORD VERSIONS
-----------------
11G 12C
```

4. 実際のパスワードはすべて小文字のluv2walkmyk9ですが、パスワードに大文字と小文字を使用してユーザーSCOTTとして接続を試みます。

```
CONNECT SCOTT
Enter password: "LuvToWalkMyK9"
ERROR: ORA-01017: invalid username/password; logon denied
Warning: You are no longer connected to ORACLE.
```

ユーザーSCOTTのパスワード・バージョンは11Gおよび12Gであるため、パスワードは大/小文字が区別されます。この例で入力されたパスワードは合っていますが、大文字と小文字が間違っています。

5. rtaylorのパスワードをgrumble_mumble2workに変更します。

```
ALTER USER rtaylor IDENTIFIED BY grumble_mumble2work;
User altered.
```

6. SYSDBA管理権限を使用して接続します。

```
CONNECT / AS SYSDBA
```

7. ユーザーrtaylorのパスワード・バージョンを確認します。

```
SELECT PASSWORD_VERSIONS FROM DBA_USERS WHERE USERNAME='RTAYLOR';
PASSWORD VERSIONS
-----------------
10G 11G 12C
```

パスワードを変更したため、SQLNET.ALLOWED_LOGON_VERSION_SERVERおよびSEC_CASE_SENSITIVE_LOGON設定で構成済の認証プロトコルによってrtaylorのパスワードの大/小文字の区別が強制されます。

8. パスワードに大文字と小文字を使用してrtaylorとして接続を試みます。

```
CONNECT rtaylor
Enter password: "Grumble_Mumble2Work"
ERROR: ORA-01017: invalid username/password; logon denied
Warning: You are no longer connected to ORACLE.
```

入力したパスワードは、パスワード作成時のものと大/小文字が異なるため失敗します。

9. 大/小文字を正しく使用して、rtaylorとして再度接続を試みます

```
CONNECT rtaylor
Enter password: "grumble_mumble2work"
Connected.
```

ユーザーrtaylorは接続できました。

rtaylorアカウントの大/小文字の区別は、サーバーでSEC_CASE_SENSITIVE_LOGONのデフォルト設定がTRUEになっていることによるものです。この設定がFALSEの場合は、rtaylorアカウントのパスワード・バージョンには10Gもあるため、大/小文字を区別しない一致を使用できます。ただし、この設定はお薦めしません。SEC_CASE_SENSITIVE_LOGONパラメータは、この理由のため非推奨となりました。セキュリティ向上のため、大/小文字を区別するパスワード認証を有効にしたままにすることをお薦めします。

親トピック: パスワードの大/小文字の区別の管理

3.2.7.5 10Gパスワード・バージョンを使用するユーザーのパスワードの確認と再設定

セキュリティを向上するため、10Gバージョンのパスワードを使用しているユーザー・アカウントを確認しパスワードをリセットして、より安全なバージョンのパスワードが今後使用されるようにします。

現行ユーザーのすべてのパスワード・バージョンの確認

DBA_USERSデータ・ディクショナリ・ビューを問い合せて、ユーザー・アカウントに構成されているすべてのパスワード・バージョンのリストを確認できます。

たとえば:

```
SELECT USERNAME,PASSWORD_VERSIONS FROM DBA_USERS;
USERNAME                       PASSWORD_VERSIONS
------------------------------ -----------------
```

```
JONES                          10G 11G 12C
ADAMS                          10G 11G
CLARK                          10G 11G
PRESTON                        11G
BLAKE                          10G
```

PASSWORD_VERSIONS列は、アカウントに存在するパスワード・バージョンのリストを示しています。10Gは以前の大/小文字を区別しないOracleパスワード・バージョン、11GはSHA-1ベースのパスワード・バージョン、12CはSHA-2ベースのSHA-512パスワード・バージョンを表します。

- ユーザーjones: このユーザーのパスワードは、SQLNET.ALLOWED_LOGON_VERSION_SERVERパラメータ設定が8のときに、Oracle Database 12cリリース12.1で再設定されました。これにより、3つのすべてのパスワード・バージョンを作成できます。
- ユーザーadamsおよびclark: これらのアカウントのパスワードは最初にOracle Database 10gで作成され、Oracle Database 11gで再設定されました。Oracle Database 11gソフトウェアは、その時点でSQLNET.ALLOWED_LOGON_VERSIONのデフォルト設定8を使用していました。大/小文字の区別がデフォルトで有効になっているため、これらのパスワードは、prestonのパスワードと同様に大/小文字が区別されます。
- ユーザーpreston: このアカウントは、排他モード(SQLNET.ALLOWED_LOGON_VERSION = 12)で実行されていたOracle Database 11gデータベースからインポートされました。
- ユーザーblake: このアカウントは、Oracle Database 10gパスワード・バージョンをまだ使用しています。この段階では、ユーザーblakeはログインできません。

10Gパスワード・バージョンを使用するユーザーのパスワードの再設定

セキュリティを強化するには、すべてのユーザーのアカウントから10Gパスワード・バージョンを削除します。次の手順では、10Gパスワード・バージョンを使用しているユーザーのパスワードを再設定するために、クライアントのログインの許可に必要な機能レベルを制御するSQLNET.ALLOWED_LOGON_VERSION_SERVER設定を一時的に緩和する必要があります。設定を緩和することで、それらのユーザーがログインしてパスワードを変更し、10Gパスワード・バージョンに加えてより新しいパスワード・バージョンを生成できるようになります。その後にデータベースが排他モードを使用するように設定して、クライアントがO5L_NP機能を使用できるようにします。その後、ユーザーはパスワードを再設定して、パスワード・バージョンに10G,を含めずに、より安全な11Gと12Cのパスワード・バージョンのみを含めるようにすることができます。

1. DBA_USERSビューを問い合せて、10Gパスワード・バージョンのみを使用しているユーザーを特定します。

```
SELECT USERNAMEFROM DBA_USERS
WHERE ( PASSWORD_VERSIONS = '10G '
OR PASSWORD_VERSIONS = '10G HTTP ')
AND USERNAME <> 'ANONYMOUS';
```

2. データベースを排他モードで実行しないように、次のように構成します。
   a. sqlnet.oraファイルのSQLNET.ALLOWED_LOGON_VERSION_SERVER設定を編集して、デフォルトより緩やかな設定にします。たとえば、次のようにします。

```
SQLNET.ALLOWED_LOGON_VERSION_SERVER=11
```

   b. データベースを再起動します。
3. DBA_USERSビューに問い合せて、10Gパスワード・バージョンのみを使用するユーザーを期限切れにします。

   10Gパスワード・バージョンのみを使用し、11Gまたは12Cパスワード・バージョンのいずれかまたは両方を使用していないユーザーを期限切れにする必要があります。

   たとえば:

```
ALTER USER username PASSWORD EXPIRE;
```

4. パスワードを期限切れにしたユーザーにログインするよう依頼します。

   ユーザーがログインすると、パスワードを変更するよう求められます。データベースは、10Gパスワード・バージョンに加えて、アカウントで欠けている11Gおよび12Cパスワード・バージョンを生成します。データベースは許可モードで実行されているため、10Gパスワード・バージョンはそのまま存在します。

5. ユーザーが接続しているクライアント・ソフトウェアにO5L_NP機能があることを確認します。

   Oracle Databaseリリース11.2.0.3以上のすべてのクライアントにO5L_NP機能があります。以前のOracle Databaseクライアントがある場合は、CPUOct2012パッチをインストールする必要があります。

6. すべてのクライアントにO5L_NP機能があれば、次のようにサーバーのセキュリティを排他モードに設定しなおします。
   a. インスタンス初期化ファイルからSEC_CASE_SENSITIVE_LOGONパラメータ設定を削除するか、SEC_CASE_SENSITIVE_LOGONをTRUEに設定します。

      ```
      SEC_CASE_SENSITIVE_LOGON = TRUE
      ```

   b. サーバーsqlnet.oraファイルからSQLNET.ALLOWED_LOGON_VERSION_SERVERパラメータを削除するか、または、サーバーsqlnet.oraファイルのSQLNET.ALLOWED_LOGON_VERSION_SERVERの値を元の12に設定して、排他モードに設定します。

      ```
      SQLNET.ALLOWED_LOGON_VERSION_SERVER = 12
      ```

   c. データベースを再起動します。
7. まだ10Gパスワード・バージョンを使用しているアカウントを特定します。

```
SELECT USERNAME FROM DBA_USERS
WHERE PASSWORD_VERSIONS LIKE '%10G%'
AND USERNAME <> 'ANONYMOUS';
```

8. まだ10Gパスワード・バージョンを使用しているアカウントを期限切れにします。

```
ALTER USER username PASSWORD EXPIRE;
```

9. これらのユーザーに自分のアカウントにログインするよう依頼します。

   ユーザーがログインすると、パスワードを再設定するよう求められます。次に、データベースは、アカウントに対して11Gおよび12Cパスワード・バージョンのみを生成します。データベースは排他モードで実行されているため、10Gパスワード・バージョンは生成されません。

10. 次の問合せを再実行します。

```
SELECT USERNAME FROM DBA_USERS
WHERE PASSWORD_VERSIONS LIKE '%10G%'
AND USERNAME <> 'ANONYMOUS';
```

    この問合せに何も応答がなければ、10Gパスワード・バージョンを使用しているアカウントはありません。これで、データベースは以前のバージョンよりも安全なモードで稼働しています。

親トピック: パスワードの大/小文字の区別の管理

3.2.7.6 大/小文字の区別がパスワード・ファイルに与える影響

デフォルトでは、パスワード・ファイルは大/小文字を区別します。ORAPWDコマンドライン・ユーティリティのIGNORECASE引数は、パスワード・ファイルの大/小文字の区別を制御します。

IGNORECASEのデフォルト値はN(no)で、大/小文字の区別が適用されます。セキュリティを強化するために、IGNORECASEをNに設定するか、ignorecase引数全体を省略します。IGNORECASEは非推奨なので注意してください。

次の例は、パスワード・ファイルで大/小文字の区別を有効にする方法を示しています。

```
orapwd file=orapw entries=100
Enter password for SYS: password
```

このコマンドは、orapwと呼ばれる大/小文字を区別するパスワード・ファイルを作成します。デフォルトでは、パスワードでは大文字と小文字が区別されます。その後、このパスワードを使用して接続する場合、パスワードの作成時と大/小文字を同じにして入力すれば成功します。同じパスワードでも異なる大/小文字を入力した場合、そのパスワードを使用した認証試行は失敗します。

または、ある形式から別の形式へのパスワード・ファイル移行を使用してIGNORECASEパラメータを無効にすることもできます。たとえば:

```
orapwd input_file=input_password_file file=output_password_file
```

以前のリリースからユーザー・アカウントをインポートし、そのアカウントがSYSDBAまたはSYSOPER管理権限を使用して作成されている場合、アカウントはパスワード・ファイルに格納されます。この時点で、アカウントのパスワードは大/小文字が区別されません。大/小文字の区別が有効な場合、次にユーザーがパスワードを変更すると、パスワードは大/小文字が区別されます。セキュリティを強化するために、そのユーザーに対してパスワードを変更するように要請してください。

関連トピック

- Oracle Database管理者ガイド

親トピック: パスワードの大/小文字の区別の管理

3.2.7.7 大/小文字の区別がデータベース・リンク接続で使用されるパスワードに与える影響

データベース・リンク接続を作成する場合は、接続用のユーザー名とパスワードを定義する必要があります。

データベース・リンク接続を作成すると、パスワードは大/小文字が区別されます。ユーザーが接続用のパスワードをどのように入力するかは、データベース・リンクが作成されたリリースによって決まります。

- ユーザーはOracle Database 12cより前のデータベースからOracle Database 12cデータベースに接続できます。大/小文字の区別が有効なため、ユーザーは、アカウントの作成時に使用された大文字/小文字を使用して、パスワードを入力する必要があります。
- ユーザーがOracle Database 12cデータベースから Oracle Database 12cより前のリリースのデータベースに接続する場合、およびリリース12cより前のデータベースのSEC_CASE_SENSITIVE_LOGONパラメータがFALSEに設定されていた場合、大文字/小文字を使用してこのデータベース・リンクのパスワードを指定できます。

既存のデータベース・リンクのユーザー・アカウントを検索するには、V$DBLINKビューを問い合せます。たとえば:

```
SELECT DB_LINK, OWNER_ID FROM V$DBLINK;
```

V$DBLINKビューの詳細は、『Oracle Databaseリファレンス』を参照してください。

親トピック: パスワードの大/小文字の区別の管理

3.2.8 パスワードのセキュリティへの脅威からの12Cパスワード・バージョンによる保護

12Cパスワード・バージョンを使用すると、ユーザーは、コンプライアンス基準を満たす複雑なパスワードを作成できます。

- 12Cバージョンのパスワード・ハッシュについて
  12Cパスワード・ハッシュはパスワードベースのセキュリティ脅威に対する保護を、大文字小文字混在のパスワードをサポートすることで実現します。
- Oracle Database 12Cパスワード・バージョン構成のガイドライン
  デフォルトでは、Oracle Databaseは、11Gおよび12Cという2つのバージョンのパスワード・ハッシュを生成します。
- 12Cパスワード・バージョンを排他的に使用するためのOracle Databaseの構成
  SQLNET.ALLOWED_LOGON_VERSION_SERVERパラメータを12aに設定し、12Cのパスワード・ハッシュ・バージョンのみが使用されるようにしてください。
- サーバーとクライアントのログオン・バージョンのデータベース・リンクへの影響
  SQLNET.ALLOWED_LOGON_VERSION_SERVERとSQLNET.ALLOWED_LOGON_VERSION_CLIENTパラメータで、リリースの異なるデータベースとクライアント間の接続に対応できます。
- 12Cパスワード・バージョンを排他的に使用するためのOracle Databaseクライアントの構成
  侵入者が偽のサーバーを用意し、認証をダウングレードしてクライアントがより脆弱なパスワード・ハッシュ・バージョンを使用するよう試みることが多くあります。

親トピック: パスワード保護の構成

3.2.8.1 12Cバージョンのパスワード・ハッシュについて

12Cパスワード・ハッシュはパスワードベースのセキュリティ脅威に対する保護を、大文字小文字混在のパスワードをサポートすることで実現します。

12Cバージョンのパスワード・ハッシュの生成に使用される暗号化ハッシュ関数は、パスワードベースのキー導出関数(PBKDF2)およびSHA-512暗号化ハッシュ関数を含む非最適化されたアルゴリズムに基づいています。12Cバージョンのパスワード・ハッシュがある場合、PBKDF2アルゴリズムによって計算上の非対称性が生じ、侵入者が元のパスワードに戻すことが難しくなります。12Cパスワードの生成では、その最後のステップとしてPBKDF2出力のSHA-512ハッシュを実行します。12Cパスワード・バージョンの生成で使用されるこの2段階のアプローチでは、クライアントにO7L_MR機能がある場合、サーバーのCPUリソースの使用が抑えられます。これは、O5LOGON認証のパスワード検証フェーズ中に、パスワード自体でPBKDF2計算全体を繰り返すのではなく、サーバーがO7L_MR対応クライアントによって送信される値の1つのSHA-512ハッシュのみを実行すればよいためです。

さらに、12Cパスワード・バージョンにより、ハッシュ時にパスワードにsaltが追加され、さらなる保護が提供されます。12Cパスワード・バージョンでは、ユーザーはより複雑なパスワードを作成できます。12Cパスワード・バージョンでsaltおよびPBKDF2非最適化が使用され、大/小文字混在のパスワードがサポートされることで、侵入者が12Cパスワード・バージョンに対して辞書攻撃や総当り攻撃を行ってユーザーのパスワードに戻すことがより難しくなります。12Cバージョンのパスワード・ハッシュを使用することをお薦めします。

パスワード・ハッシュ値は、サーバーとログインしているユーザー間の「共有秘密」として使用されるため、非常に機密性が高いとみなされます。侵入者にこの秘密を知られると、認証の保護はただちに重大な危険にさらされます。アカウント管理権限を持つ管理ユーザー、SYSDBA管理権限を持つ管理ユーザーまたはEXP_FULL_DATABASEロールを持つユーザーがパスワード・ハッシュ値に直接アクセスできることに注意してください。したがって、データベースのパスワードベースの認証の整合性を保持するには、このタイプの管理ユーザーは信頼できるユーザーである必要があります。これらの管理者が信頼できない場合、パスワード・ハッシュ値が「エンタープライズ・ユーザー・セキュリティ」ディレクトリ内に保持され、エンタープライズ・ユーザー・セキュリティ管理者以外はアクセスできないように、ディレクトリ・サーバー(Oracle Databaseエンタープライズ・ユーザー・セキュリティなど)をデプロイすることをお薦めします。

関連トピック

- 『Oracle Database Net Servicesリファレンス・ガイド』

親トピック: パスワードのセキュリティへの脅威からの12Cパスワード・バージョンによる保護

3.2.8.2 Oracle Database 12Cパスワード・バージョン構成のガイドライン

デフォルトでは、Oracle Databaseは、11Gおよび12Cという2つのバージョンのパスワード・ハッシュを生成します。

Oracle Databaseが特定のクライアントの認証に使用するパスワード・ハッシュのバージョンは、クライアントの機能、ならびにSQLNET.ALLOWED_LOGON_VERSION_CLIENTおよびSQLNET.ALLOWED_LOGON_VERSION_SERVERパラメータの設定に依存します。パスワード・バージョンによるクライアント認証の動作の詳細は、Oracle Database Net ServicesリファレンスのSQLNET.ALLOWED_LOGON_VERSION_SERVERパラメータの説明にあるSQLNET.ALLOWED_LOGON_VERSION_SERVERの設定の表で、クライアントに必要な機能の列を参照してください。

Oracle Database 10gで生成された10Gパスワード・バージョンは大/小文字を区別しません。11Gと12Cの両パスワード・バージョンでは大/小文字は区別されます。

Oracle Database 12g リリース2 (12.2)では、sqlnet.oraのパラメータSQLNET.ALLOWED_LOGON_VERSION_SERVERのデフォルトは12 (これは排他モードで、10Gパスワード・バージョンを使用できません)、SQLNET.ALLOWED_LOGON_VERSION_CLIENTパラメータのデフォルトは11です。新しいアカウントについては、クライアントがOracle Database 12cの場合、Oracle Databaseは、Oracle Database 12cリリース・ソフトウェアを実行しているクライアントで12Cパスワード・バージョンを排他的に使用します。Oracle Databaseリリース12cより前に作成されたアカウントの場合、クライアントにO5L_NP機能があるかぎり、ログインは成功します。これは、Oracle Databaseリリース11gなどの以前のリリースで作成されたアカウント用に11Gパスワード・バージョンが通常存在しているためです。非常に古いアカウント(Oracle Databaseリリース10gのものなど)の場合、そのアカウントにSHA-1パスワード・バージョンを作成するために、ユーザーのパスワードの再設定が必要になる可能性があります。新しいアカウントを作成する際、または既存のアカウント・パスワードを変更する際に、12Cパスワード・バージョンのみを生成するようこのサーバーを構成するには、SQLNET.ALLOWED_LOGON_VERSION_SERVERパラメータを12aに設定します。ただし、古いクライアントに対してアプリケーションに互換性を持たせる場合は、SQLNET.ALLOWED_LOGON_VERSION_SERVERをデフォルトの12に設定するようにします。

SQLNET.ALLOWED_LOGON_VERSION_SERVERパラメータの設定は、セキュリティと、システムに必要な古いクライアントとの相互運用性とのバランスによって決まります。セキュリティのレベルを次のように制御できます。

- 最高レベルの互換性: 新規アカウントの作成時または既存のアカウントのパスワードの変更時に3つのバージョンのパスワード・ハッシュ(12Cパスワード・バージョン、11Gパスワード・バージョンおよびDESベースの10Gパスワード・バージョン)をすべて生成するようにサーバーを構成するには、SQLNET.ALLOWED_LOGON_VERSION_SERVERパラメータを値11以下に設定します。(以前のリリースでは値8がデフォルトで使用されていたことに注意してください。)
- 推奨レベルのセキュリティ: 新規アカウントの作成時または既存のアカウントのパスワードの変更時に12Cパスワード・バージョンと11Gパスワード・バージョンの両方を生成するように(ただし、10Gパスワード・バージョンは除く)サーバーを構成するには、SQLNET.ALLOWED_LOGON_VERSION_SERVERパラメータを値12に設定します。
- 最高レベルのセキュリティ: 新しいアカウントを作成する際、または既存のアカウント・パスワードを変更する際に、12Cパスワード・バージョンのみを生成するようサーバーを構成するには、SQLNET.ALLOWED_LOGON_VERSION_SERVERパラメータを値12aに設定します。

認証中に、アカウントに存在するパスワード・バージョンの種類および使用するクライアント・ソフトウェアのバージョンに基づき、次のシナリオが可能です。

- 10Gバージョンのパスワード・ハッシュのみを使用したアカウント: サーバーで古いアカウントに対して新しいバージョンのパ

スワード・ハッシュを生成するよう強制する場合、管理者は10Gパスワード・バージョンのみを持つ(11Gや12Cといったよりセキュアなパスワード・バージョンを持たない)アカウントのパスワードを期限切れにする必要があります。データベースはこれらのパスワード・バージョンを使用してより強固なセキュリティを提供しているため、これらのパスワード・バージョンを生成する必要があります。これらのユーザーは次のようにして検出できます。

```
SELECT USERNAMEFROM DBA_USERS
WHERE PASSWORD_VERSIONS LIKE '%10G%
AND USERNAME <> 'ANONYMOUS';
```

次に、各アカウントを次のようにして期限切れにします。

```
ALTER USER username PASSWORD EXPIRE;
```

各アカウントを期限切れにした後、これらのユーザーにログインするよう通知します。ログインの際、パスワードを変更するよう求められます。クライアントのバージョンによって、使用されるパスワード・バージョンが決まります。SQLNET.ALLOWED_LOGON_VERSION_SERVERパラメータの設定によって、生成されるパスワード・バージョンが決まります。クライアントにO7L_MR機能(Oracle Databaseリリース12c)がある場合、12Cパスワード・バージョンが認証に使用されます。クライアントにO5L_NP機能があり、O7L_MR機能がない場合(Oracle Databaseリリース11gクライアントなど)、11Gパスワード・バージョンが認証に使用されます。認証に12Cパスワード・バージョンを排他的に使用するには、すべてのクライアントをOracle Databaseリリース12cにアップグレードする必要があります。(デフォルトでは、Oracle Databaseリリース11.2.0.3以上のクライアントにO5L_NP機能があります。これにより、11Gパスワード・バージョンを排他的に使用できるようになります。以前のOracle Databaseクライアントを使用している場合は、CPUOct2012パッチをインストールする必要があります。)

アカウント・パスワードの期限が切れ、ALLOWED_LOGON_VERSION_SERVERパラメータが12または12aに設定されている場合、10Gパスワード・バージョンは削除され、パラメータの設定に応じて次のように1つまたは2つの新しいパスワード・バージョンが作成されます。

- ALLOWED_LOGON_VERSION_SERVERが12 (デフォルト)に設定されている場合、11Gと12Cの両方のバージョンのパスワード・ハッシュが生成されます。
- ALLOWED_LOGON_VERSION_SERVERが12aに設定されている場合、12Cバージョンのパスワード・ハッシュのみが生成されます。

詳細は、Oracle Database Net ServicesリファレンスのSQLNET.ALLOWED_LOGON_VERSION_SERVERパラメータの使用上のノートに関する項の表内で、生成されるパスワードのバージョンの列を参照してください。

- 10Gおよび11Gバージョンのパスワード・ハッシュを使用したアカウント: リリース10g以上のクライアントを使用しているユーザーは、11Gバージョンのパスワード・ハッシュが使用されるため、ユーザー・ログインは成功します。ただし、最新のバージョンを使用するには、前述のアカウントに関する箇条書き項目での説明のように、これらのパスワードを期限切れにします。
- 11Gバージョンのパスワード・ハッシュのみを使用したアカウント: 認証で11Gバージョンのパスワード・ハッシュが使用されます。最新のバージョンを使用するには、1つ目の箇条書き項目での説明のように、パスワードを期限切れにします。

Oracle Database 12cのデフォルト構成(SQLNET.ALLOWED_LOGON_VERSION_SERVERは12)は、Oracle Database 12cリリース2 (12.2)の認証プロトコルおよびOCIベースのドライバを使用する以降の製品(SQL*Plus、ODBC、Oracle .NET、Oracle Forms、および様々なサード・パーティ製Oracle Databaseアダプタなど)と互換性があることを意味します。JDBCタイプ4 (シン)バージョン(CPUOct2012バンドルパッチが適用されたもの、またはOracle Database 11g以降のもの)およびOracle Database 10gリリース10.2以降のOracle Database Clientインタフェース(OCI)ベースのドライバとも互換性があります。OCIクライアント・ドライバの以前のリリースでは、パスワードベース認証を使用してOracle

Databaseデータベースに対して認証することができません。

親トピック: パスワードのセキュリティへの脅威からの12Cパスワード・バージョンによる保護

3.2.8.3 12Cパスワード・バージョンを排他的に使用するためのOracle Databaseの構成

SQLNET.ALLOWED_LOGON_VERSION_SERVERパラメータを12aに設定し、12Cのパスワード・ハッシュ・バージョンのみが使用されるようにしてください。

12Cパスワード・バージョンは、パスワードハッシュ・バージョンの中で最も制限が厳しく安全性が高いものであるため、このパスワード・バージョンのみを使用することをお薦めします。デフォルトでは、SQLNET.ALLOWED_LOGON_VERSION_SERVERが12に設定されているため、11Gと12Cの両方のパスワード・バージョンを使用できます。
(SQLNET.ALLOWED_LOGON_VERSION_SERVERの12と12aの両方の値は排他モードと見なされるため、以前の10Gパスワード・バージョンは使用できません。)以前のリリースからアップグレードした場合、または SQLNET.ALLOWED_LOGON_VERSION_SERVERが12または以前のリリースで使用されていた別の設定に設定されている場合、このパラメータを再構成する必要があります。これは、侵入者がより脆弱なパスワード・バージョンを使用するために認証のダウングレードを試みる可能性があるためです。このトピックの後述の表では、パスワード・バージョンの生成に対する SQLNET.ALLOWED_LOGON_VERSION_SERVER設定の効果を示しています。

12Cパスワード・バージョンを排他的に使用できるのは、Oracle Database 12cリリース12.1.0.2以降のクライアントを使用している場合に限られます。SQLNET.ALLOWED_LOGON_VERSION_SERVERパラメータを12aに変更する前に、サーバーに接続しているデータベース・クライアントのバージョンを確認します。

1. ALTER USERシステム権限を持つ管理ユーザーとしてSQL*Plusにログインします。
2. 次のSQL問合せを実行して、ユーザーのパスワード・バージョンを確認します。

   ```
   SELECT USERNAME,PASSWORD_VERSIONS FROM DBA_USERS;
   ```

3. 12Cパスワード・バージョンを持たない各ユーザーのアカウントを期限切れにします。

   たとえば、ユーザーblakeがまだ10Gパスワード・バージョンを使用しているとします。

   ```
   ALTER USER blake PASSWORD EXPIRE;
   ```

   これらのユーザーが次にログインする際、パスワードの変更が強制され、サーバーが排他モードに必要なパスワード・バージョンを生成できます。

4. しかるべき期間(30日間など)内にログインするようユーザーに通知します。

   ログイン時、パスワードを変更するよう求められ、排他モードでの認証に必要なパスワード・バージョンがサーバーによって生成されます。(排他モードの機能の詳細は、『Oracle Database Net Servicesリファレンス』の SQLNET.ALLOWED_LOGON_VERSION_SERVERパラメータの使用上のノートに関する項を参照してください。)

5. パスワードの大/小文字を含めてこれらのテスト・スクリプトまたはバッチ・ジョブで使用されるパスワードと正確に一致するように、テスト・スクリプトまたはバッチ・ジョブで使用されるアカウントのパスワードを手動で変更します。

6. 次のようにして排他モード構成を有効にします。

   a. sqlnet.oraパラメータ・ファイルのバックアップ・コピーを作成します。

      デフォルトでは、このファイルは、UNIXオペレーティング・システムでは$ORACLE_HOME/network/adminディレクトリに、Microsoft Windowsオペレーティング・システムでは%ORACLE_HOME%¥network¥adminディレクトリにあります。

マルチテナント環境では、sqlnet.oraファイルの設定がすべてのPDBに適用されることに注意してください。

b. 表3-4をガイダンスに使用して、SQLNET.ALLOWED_LOGON_VERSION_SERVERパラメータを設定します。

c. sqlnet.oraファイルを保存します。

パスワード・バージョン生成のSQLNET.ALLOWED_LOGON_VERSION_SERVER設定の効果を次の表に示します。

表3-4 パスワード・バージョンの生成に対するSQLNET.ALLOWED_LOGON_VERSION_SERVERの影響

SQLNET.ALLOWED_LOGON_VERSION_SERVER設定	8	11	12	12a
サーバーを排他モードで実行しますか	いいえ	いいえ	はい	はい
10G パスワード・バージョンを生成しますか	はい	はい	いいえ	いいえ
11G パスワード・バージョンを生成しますか	はい	はい	はい	いいえ
12C パスワード・バージョンを生成しますか	はい	はい	はい	はい

Oracle Database 12cリリース12.1.0.2以降のクライアントを使用する場合、SQLNET.ALLOWED_LOGON_VERSION_SERVERを12aに設定します。

設定を高くすると、次のようにパスワード・バージョンの使用がさらに制限されます。

- 12a (最も制限が厳しく、安全性の高い設定)に設定すると、12Cパスワード・バージョンのみが許可されます。
- 12に設定すると、11Gおよび12Cパスワード・バージョンが許可され、認証に使用されます。
- 8に設定すると、ほとんどのパスワード・バージョン(10G、11Gおよび12C)が許可されます。

SQLNET.ALLOWED_LOGON_VERSION_SERVERパラメータの詳細は、Oracle Database Net Servicesリファレンスを参照してください。

ノート:

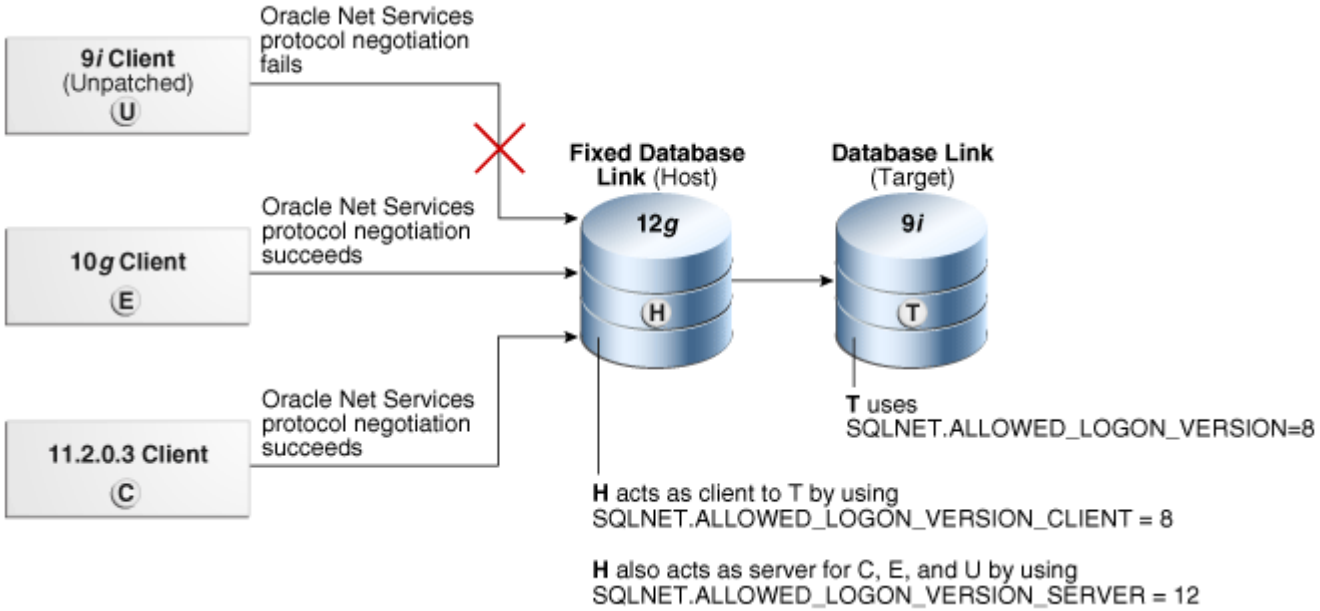
以前のリリースを実行しているターゲット・データベースへの固定データベース・リンクをホストするシステムの場合、サーバーとクライアントのログオン・バージョンのデータベース・リンクへの影響で説明しているように、SQLNET.ALLOWED_LOGON_VERSION_CLIENT パラメータを設定できます。

親トピック: パスワードのセキュリティへの脅威からの12Cパスワード・バージョンによる保護

3.2.8.4 サーバーとクライアントのログオン・バージョンのデータベース・リンクへの影響

SQLNET.ALLOWED_LOGON_VERSION_SERVERとSQLNET.ALLOWED_LOGON_VERSION_CLIENTパラメータで、リリースの異なるデータベースとクライアント間の接続に対応できます。

次の図に、異なるリリースのデータベースとクライアントの接続の仕組みを示します。
SQLNET.ALLOWED_LOGON_VERSION_CLIENTパラメータは、データベース・リンクHをホストするサーバーにおいて、クライアントに許可されたログオン・バージョンという側面に影響します。この設定により、Hはデータベース・リンクを介して、Oracle 9iを実行しているサーバー(T)などの古いサーバーに接続できますが、パッチが適用されていない古いクライアント(U)からの接続を引き続き拒否します。この場合、Oracle Net Servicesプロトコル・ネゴシエーションが失敗し、Oracle 9Iソフトウェアを使用して認証を試行しているこのクライアントに「ORA-28040: 一致する認証プロトコルがありません」エラー・メッセージが表示されます。Oracle Database 10gリリース10.2クライアントEのOracle Net Servicesプロトコル・ネゴシエーションは、このリリースにクリティカル・パッチ・アップデートCPUOct2012が組み込まれているため成功します。リリース11.2.0.3クライアントCのOracle Net Servicesプロトコル・ネゴシエーションは、安全なパスワード・バージョンが使用されているため成功します。



このシナリオでは、データベース・リンクHをホストするシステムに次の設定を使用します。

```
SQLNET.ALLOWED_LOGON_VERSION_CLIENT=8
SQLNET.ALLOWED_LOGON_VERSION_SERVER=12
```

リモートOracle Database Tには次の設定があることに注意してください。

```
SQLNET.ALLOWED_LOGON_VERSION=8
```

リモートのOracle Database TのリリースがホストHに設定されるSQLNET.ALLOWED_LOGON_VERSION_CLIENTパラメータで定義された値を満たしていないまたは超えていない場合、データベース・リンク・ユーザーの認証中に固定データベース・リンクの問合せに失敗し、エンドユーザーがデータベース・リンクで表にアクセスしようとすると、「ORA-28040: 一致する認証プロトコルがありません」エラーになります。

ノート:

古い Oracle Database クライアント(Oracle Database 11g リリース 11.1.0.7 など)を使用している場合は、アップグレードしてクリティカル・パッチ・アップデート CPUOct2012 を使用することをお薦めします。

関連項目:

- SQLNET.ALLOWED_LOGON_VERSION_CLIENTパラメータの詳細は、『Oracle Database Net Servicesリファレンス』を参照してください。

- CPUOct2012の詳細は、http://www.oracle.com/technetwork/topics/security/cpuoct2012-1515893.htmlを参照してください。

親トピック: パスワードのセキュリティへの脅威からの12Cパスワード・バージョンによる保護

3.2.8.5 12Cパスワード・バージョンを排他的に使用するためのOracle Databaseクライアントの構成

侵入者が偽のサーバーを用意し、認証をダウングレードしてクライアントがより脆弱なパスワード・ハッシュ・バージョンを使用するよう試みることが多くあります。

- 10Gパスワード・バージョンの使用、または10Gおよび11Gパスワード・バージョンの使用を防ぐには、サーバーの構成後、次のように、排他モードで稼働するようクライアントを構成します。
  - クライアント排他モード設定を使用して、11Gおよび12Cの2つのパスワード・バージョンを許可するには:

    ```
    SQLNET.ALLOWED_LOGON_VERSION_CLIENT = 12
    ```

  - より制限の厳しいクライアント排他モード設定を使用して、12Cパスワード・バージョンのみを許可するには (この設定では、クライアントはOracle Database 12cリリース1 (12.1.0.2)以上のサーバーにのみ接続できます):

    ```
    SQLNET.ALLOWED_LOGON_VERSION_CLIENT = 12a
    ```

サーバーとクライアントの両方が同じコンピュータにインストールされている場合、それぞれのTNS_ADMIN環境変数が各Oracle Net Services構成ファイルの正しいディレクトリを指していることを確認します。変数が両方で同じ場合、サーバーがクライアントのSQLNET.ALLOWED_LOGON_VERSION_CLIENT設定をかわりに使用することがあります。

古いOracle Databaseクライアント(Oracle Database 11gリリース11.1.0.7など)を使用している場合は、これらのクライアントにCPUOct2012以上を適用する必要があります。このパッチは、O5L_NP機能を提供します。このパッチを適用しないと、ユーザーはログインできなくなります。

関連項目:

- SQLNET.ALLOWED_LOGON_VERSION_CLIENTパラメータの詳細は、『Oracle Database Net Servicesリファレンス』を参照してください。
- CPUOct2012の詳細は、次のOracle Technology Networkサイトを参照してください。

  http://www.oracle.com/technetwork/topics/security/cpuoct2012-1515893.html

親トピック: パスワードのセキュリティへの脅威からの12Cパスワード・バージョンによる保護

3.2.9 パスワード資格証明用の安全性の高い外部パスワード・ストアの管理

安全性の高い外部パスワード・ストアは、パスワード資格証明の格納に使用されるクライアント側のウォレットになります。

- 安全性の高い外部パスワード・ストアについて
  パスワード資格証明データベース接続は、クライアント側のOracleウォレットを使用して格納できます。
- 安全性の高い外部パスワード・ストアの機能
  ユーザー(アプリケーション、バッチ・ジョブ、スクリプトを含む)は、データベース接続文字列を指定した標準のCONNECT文を使用してデータベースに接続します。
- 安全性の高い外部パスワード・ストアの使用を目的とするクライアントの構成について
  クライアントがWindowsネイティブ認証やSSLなどの外部認証を使用するように構成されている場合、Oracle

Databaseではその認証方式が使用されます。

- 安全性の高い外部パスワード・ストアの使用を目的とするクライアントの構成
  安全性の高い外部パスワード・ストア機能を使用するようにクライアントを構成するには、mkstoreコマンドライン・ユーティリティを使用します。
- 例: ウォレット・パラメータが設定されたサンプルsqlnet.oraファイル
  sqlnet.oraファイルに特別なパラメータを設定すると、ウォレットの管理方法を制御できます。
- 外部パスワード・ストア資格証明の管理
  mkstoreコマンドライン・ユーティリティで、外部パスワード・ストアからの資格証明を管理します。

親トピック: パスワード保護の構成

3.2.9.1 安全性の高い外部パスワード・ストアについて

パスワード資格証明データベース接続は、クライアント側のOracleウォレットを使用して格納できます。

Oracleウォレットは、認証および署名用資格証明を格納する安全性の高いソフトウェア・コンテナです。このウォレットの使用方法により、データベースに接続する際にパスワード資格証明に依存する大規模な配置を簡素化できます。この機能が構成されている場合、アプリケーション・コードおよびスクリプトにユーザー名とパスワードを埋め込む必要がありません。これによりパスワードが危険にさらされることがなくなるため、リスクが軽減します。また、ユーザー名またはパスワードが変更されるたびにアプリケーション・コードを変更する必要がなくなるため、パスワード管理ポリシーの適用が容易になります。

ノート:

ウォレットの外部パスワード・ストアは、公開キー・インフラストラクチャ(PKI)資格証明が格納されている領域とは別の場所にあります。そのため、ウォレットの外部パスワード・ストアの資格証明管理には、Oracle Wallet Managerを使用できません。かわりに、コマンドライン・ユーティリティ mkstore を使用して資格証明を管理します。

関連トピック

- 安全性の高い外部パスワード・ストアとプロキシ認証の使用
- Oracle Databaseエンタープライズ・ユーザー・セキュリティ管理者ガイド

親トピック: パスワード資格証明用の安全性の高い外部パスワード・ストアの管理

3.2.9.2 安全性の高い外部パスワード・ストアの機能

ユーザー(アプリケーション、バッチ・ジョブ、スクリプトを含む)は、データベース接続文字列を指定した標準のCONNECT文を使用してデータベースに接続します。

この文字列には、ユーザー名、パスワード、およびOracle Databaseネットワーク上のデータベースを識別するOracle Netサービス名が含まれています。パスワードを省略すると、ユーザーは接続時にパスワードが要求されます。

たとえば、このサービス名は、データベースを識別するURL、またはデータベースのtnsnames.oraファイルに入力したTNS別名になります。または、host:port:sidの文字列となる場合もあります。

次の例は、外部パスワード・ストアを使用するように構成されていないクライアントにも使用できる標準CONNECT文です。

```
CONNECT salesapp@sales_db.us.example.com
Enter password: password
CONNECT salesapp@orasales
Enter password: password
CONNECT salesapp@ourhost37:1527:DB17
Enter password: password
```

これらの例では、salesappがユーザー名で、一意のデータベースの接続文字列が3つの方法で示されています。URL形式のsales_db.us.example.com、tnsnames.oraファイルでのTNS別名orasales、またはhost:port:sid形式の文字列を使用できます。

ただし、クライアントが安全性の高い外部パスワード・ストアを使用するように構成されている場合は、アプリケーションは、データベースのログイン接続情報を指定せずに、次のCONNECT文の構文を使用してデータベースに接続できます。

```
CONNECT /@db_connect_string
CONNECT /@db_connect_string AS SYSDBA
CONNECT /@db_connect_string AS SYSOPER
```

ここでdb_connect_stringは、前述の例にあったように、サービス名、URL、別名など、対象のデータベースにアクセスするための有効な接続文字列です。各ユーザー・アカウントにはそれぞれ専用の一意の接続文字列が必要です。複数のユーザー用に1つの接続文字列を作成することはできません。

この場合、データベースの資格証明、ユーザー名およびパスワードが、安全性を目的に作成されたOracleウォレットに格納されています。このウォレットの自動ログイン機能が使用状態になるため、システムにはウォレットを開くためのパスワードは不要です。ウォレットから、システムはデータベースにアクセスするための資格証明を、資格を証明するユーザーのために取得します。

関連トピック

- Oracle Databaseエンタープライズ・ユーザー・セキュリティ管理者ガイド

親トピック: パスワード資格証明用の安全性の高い外部パスワード・ストアの管理

3.2.9.3 安全性の高い外部パスワード・ストアの使用を目的とするクライアントの構成について

クライアントがWindowsネイティブ認証やSSLなどの外部認証を使用するように構成されている場合、Oracle Databaseではその認証方式が使用されます。

通常は、そのタイプの認証で使用されるものと同じ資格証明が、データベースへのログインにも使用されます。データベース認証として、その認証方式を使用しないかまたは変更するクライアントには、sqlnet.oraのSQLNET.WALLET_OVERRIDEパラメータをTRUEに設定できます。SQLNET.WALLET_OVERRIDEのデフォルト値はFALSEで、今までと同様に認証資格証明の標準的な使用が許可されます。

親トピック: パスワード資格証明用の安全性の高い外部パスワード・ストアの管理

3.2.9.4 安全性の高い外部パスワード・ストアの使用を目的とするクライアントの構成

安全性の高い外部パスワード・ストア機能を使用するようにクライアントを構成するには、mkstoreコマンドライン・ユーティリティを使用します。

1. コマンドラインで次の構文を使用して、クライアント上にウォレットを作成します。

   ```
   mkstore -wrl wallet_location -create
   ```

   たとえば:

   ```
   mkstore -wrl c:¥oracle¥product¥19.1.0¥db_1¥wallets -create
   Enter password: password
   ```

   wallet_locationは、ウォレットを作成して格納するディレクトリのパスです。このコマンドにより、指定した場所にOracleウォレットが作成され、自動ログイン機能が使用可能になります。この自動ログイン機能により、クライアントは、パスワードを指定しなくてもウォレットの内容にアクセスできます。

   mkstoreユーティリティの-createオプションを指定すると、パスワードの複雑度検証が使用されます。詳細は、「パ

スワードの複雑度検証について」を参照してください。

2. コマンドラインで次の構文を使用して、ウォレットにデータベース接続の資格証明を作成します。

```
mkstore -wrl wallet_location -createCredential db_connect_string username
Enter password: password
```

たとえば:

```
mkstore -wrl c:¥oracle¥product¥19.1.0¥db_1¥wallets -createCredential orcl
system
Enter password: password
```

詳細は、次のとおりです。

- `wallet_location`は、ステップ1でウォレットを作成したディレクトリのパスです。
- `db_connect_string`は、`tnsnames.ora`ファイルでデータベースを指定するために使用するTNS別名、またはOracleネットワーク上のデータベースを識別するために使用するサービス名です。デフォルトで、`tnsnames.ora`は`$ORACLE_HOME/network/admin`ディレクトリ(UNIXシステムの場合)または`ORACLE_HOME¥network¥admin`(Windowsの場合)にあります。
- `username`は、データベース・ログイン資格証明です。プロンプトが表示された後に、このユーザーのパスワードを入力します。

アクセス可能にする各データベースに対し、`CONNECT /@db_connect_string`構文を使用してこのステップを繰り返します。`CONNECT /@db_connect_string`文で使用する`db_connect_string`は、`-createCredential`コマンドで指定する`db_connect_string`と同じにする必要があります。

3. クライアントの`sqlnet.ora`ファイルに、`WALLET_LOCATION`パラメータを入力し、ステップ1で作成したウォレットのディレクトリの場所に設定します。

たとえば、`$ORACLE_HOME/network/admin`にウォレットを作成し、Oracleホームが`/private/ora11`に設定されている場合、クライアントの`sqlnet.ora`ファイルには次のように指定する必要があります。

```
WALLET_LOCATION =
  (SOURCE =
    (METHOD = FILE)
    (METHOD_DATA =
  (DIRECTORY = /private/ora11/network/admin)
  )
 )
```

4. クライアントの`sqlnet.ora`ファイルに`SQLNET.WALLET_OVERRIDE`パラメータを入力し、それを次のように`TRUE`に設定します。

```
SQLNET.WALLET_OVERRIDE = TRUE
```

この設定により、すべての`CONNECT /@db_connect_string`文で、データベースへの認証に、指定された場所にあるウォレットの情報が使用されます。

外部認証が使用されている場合、そのウォレットによる認証ユーザーは`CONNECT /@db_connect_string`構文を使用し、前述の手順で指定したデータベースにユーザー名およびパスワードを使用せずにアクセスできます。ただし、ユーザーがこの外部認証に失敗した場合は、これらの接続文の実行も失敗します。

ノート:

アプリケーションが暗号化に SSL を使用する場合、sqlnet.ora パラメータ SQLNET.AUTHENTICATION_SERVICES により SSL が指定され、SSL ウォレットが作成されます。このアプリケーションが、データベースへの認証に SSL 証明書ではなく秘密のストア資格証明を使用する場合、これらの資格証明を SSL ウォレットに格納する必要があります。SSL 認証の後、SQLNET.WALLET_OVERRIDE = TRUE に設定されている場合は、ウォレットのユーザー名およびパスワードがデータベースへの認証に使用されます。SQLNET.WALLET_OVERRIDE = FALSE の場合は、SSL 証明書が使用されます。

親トピック: パスワード資格証明用の安全性の高い外部パスワード・ストアの管理

3.2.9.5 例: ウォレット・パラメータが設定されたサンプルsqlnet.oraファイル

sqlnet.oraファイルに特別なパラメータを設定すると、ウォレットの管理方法を制御できます。

例3-2では、「安全性の高い外部パスワード・ストアの使用を目的とするクライアントの構成」のステップ3およびステップ4で説明したWALLET_LOCATIONおよびSQLNET.WALLET_OVERRIDEパラメータが指定されている、サンプルsqlnet.oraファイルを示します。

例3-2 ウォレット・パラメータが設定されたサンプルsqlnet.oraファイル

```
WALLET_LOCATION =
  (SOURCE =
    (METHOD = FILE)
      (METHOD_DATA =
        (DIRECTORY = /private/ora_db/network/admin)
     )
  )
SQLNET.WALLET_OVERRIDE = TRUE
SSL_CLIENT_AUTHENTICATION = FALSE
SSL_VERSION = 0
```

親トピック: パスワード資格証明用の安全性の高い外部パスワード・ストアの管理

3.2.9.6 外部パスワード・ストア資格証明の管理

mkstoreコマンドライン・ユーティリティで、外部パスワード・ストアからの資格証明を管理します。

- 外部パスワード・ストアの内容のリスト表示
  クライアント・ウォレットの外部パスワード・ストアの内容(資格証明など)を表示できます。
- 外部パスワード・ストアへの資格証明の追加
  1つのクライアント・ウォレットに複数の資格証明を格納できます。
- 外部パスワード・ストアの資格証明の変更
  データベース接続文字列が変わった場合は、ウォレットに格納されているデータベース・ログイン資格証明を変更できます。
- 外部パスワード・ストアからの資格証明の削除
  データベースが現存しない場合や、特定のデータベースへの接続を無効にする場合は、データベースのログイン資格証明をウォレットから削除できます。

親トピック: パスワード資格証明用の安全性の高い外部パスワード・ストアの管理

3.2.9.6.1 外部パスワード・ストアの内容のリスト表示

クライアント・ウォレットの外部パスワード・ストアの内容(資格証明など)を表示できます。

外部パスワード・ストアの内容をリスト表示することによって、ストアに資格証明を追加または削除するかどうかの判断に使用できる情報が提供されます。

- 外部パスワード・ストアの内容をリスト表示するには、コマンドラインで次のコマンドを入力します。

```
mkstore -wrl wallet_location -listCredential
```

たとえば:

```
mkstore -wrl c:¥oracle¥product¥19.1.0¥db_1¥wallets -listCredential
```

`wallet_location`では、表示する外部パスワード・ストアの内容が格納されているウォレットのディレクトリ・パスを指定します。このコマンドにより、資格証明にあるデータベース・サービス名(別名)および対応するユーザー名(スキーマ)がすべてリスト表示されます。パスワードはリスト表示されません。

親トピック: 外部パスワード・ストア資格証明の管理

3.2.9.6.2 外部パスワード・ストアへの資格証明の追加

1つのクライアント・ウォレットに複数の資格証明を格納できます。

たとえば、クライアントのバッチ・ジョブが`hr_database`に接続し、スクリプトが`sales_database`に接続する場合、同じクライアント・ウォレットにこのログイン資格証明を格納できます。ただし、同じウォレット内の同じデータベースに対する、(複数のスキーマにログインするための)複数の資格証明を格納することはできません。同じデータベースに対する複数のログイン資格証明がある場合、別のウォレットに格納する必要があります。

- 既存のクライアント・ウォレットにデータベース・ログイン資格証明を追加するには、コマンド行で次のコマンドを指定します。

```
mkstore -wrl wallet_location -createCredential db_alias username
```

たとえば:

```
mkstore -wrl c:¥oracle¥product¥19.1.0¥db_1¥wallets -createCredential orcl system
Enter password: password
```

詳細は、次のとおりです。

- `wallet_location`は、資格証明を追加するクライアント・ウォレットが格納されるディレクトリのパスです。
- `db_alias`は、`tnsnames.ora`ファイルでデータベースを指定するために使用するTNS別名、またはOracleネットワーク上のデータベースを識別するために使用するサービス名です。
- `username`は、アプリケーションが接続するスキーマに対するデータベース・ログイン資格証明です。プロンプトが表示された後に、このユーザーのパスワードを入力します。

親トピック: 外部パスワード・ストア資格証明の管理

3.2.9.6.3 外部パスワード・ストアの資格証明の変更

データベース接続文字列が変わった場合は、ウォレットに格納されているデータベース・ログイン資格証明を変更できます。

- ウォレット内のデータベース・ログイン資格証明を変更するには、コマンドラインで次のコマンドを入力します。

```
mkstore -wrl wallet_location -modifyCredential db_alias username
```

たとえば:

```
mkstore -wrl c:¥oracle¥product¥19.1.0¥db_1¥wallets -modifyCredential sales_db
Enter password: password
```

詳細は、次のとおりです。

- `wallet_location`は、ウォレットが格納されているディレクトリのパスです。
- `db_alias`は、データベースの識別に使用する新規または個別の別名です。これは、`tnsnames.ora`ファイルでデータベースを指定するために使用するTNS別名、もしくはOracleネットワークのデータベースを識別するために使用する任意のサービス名になります。
- `username`は、新規または他のデータベース・ログイン資格証明です。プロンプトが表示された後に、このユーザーのパスワードを入力します。

親トピック: 外部パスワード・ストア資格証明の管理

3.2.9.6.4 外部パスワード・ストアからの資格証明の削除

データベースが現存しない場合や、特定のデータベースへの接続を無効にする場合は、データベースのログイン資格証明をウォレットから削除できます。

- ウォレットのデータベース・ログイン資格証明を削除するには、コマンドラインで次のコマンドを入力します。

  ```
  mkstore -wrl wallet_location -deleteCredential db_alias
  ```

たとえば:

```
mkstore -wrl c:¥oracle¥product¥19.1.0¥db_1¥wallets -deleteCredential orcl
```

詳細は、次のとおりです。

- `wallet_location`は、ウォレットが格納されているディレクトリのパスです。
- `db_alias`は、`tnsnames.ora`ファイルでデータベースを指定するために使用するTNS別名、またはOracle Databaseネットワーク上のデータベースを識別するために使用するサービス名です。

親トピック: 外部パスワード・ストア資格証明の管理

3.2.10 管理ユーザーのパスワードの管理

パスワード・ファイルやパスワード複雑度ファンクションなど、管理ユーザーのパスワードには特別な保護機能があります。

- 管理ユーザーのパスワード管理について
  管理ユーザーのパスワードはデータベースの外部に保存されるため、データベースが開かれていないときでも認証が可能です。
- 管理ユーザーのLOCKおよびEXPIREDステータスの設定
  アカウントがロックされている管理ユーザーはデータベースに接続できません。
- 管理ユーザーのパスワード・プロファイル設定
  管理ユーザーに強制される、いくつかのユーザー・プロファイル・パスワード設定があります。
- 管理ユーザーの最後の正常なログイン時間
  パスワード・ファイル・ベースの認証を使用する管理ユーザー接続の最後の正常なログイン時間が取得されます。
- 管理ユーザーのパスワード・ファイルの管理
  ORAPWDユーティリティのFORMATパラメータを12.2に設定すると、管理ユーザーのパスワード・プロファイル・パラメータを管理できます。

- 管理ユーザーのパスワード・ファイルの移行
  ORAPWDユーティリティのinput_fileパラメータまたはDBUAを使用して、以前のパスワード・ファイルの形式を12または12.2形式に移行できます。
- 管理ユーザーのパスワード・ファイルに対するマルチテナント・オプションの影響
  マルチテナント環境では、ローカル管理ユーザーと共通管理ユーザーのパスワード情報はそれぞれ異なる場所に保存されます。
- 管理ユーザーのパスワードの複雑性の検証機能
  セキュリティ向上のため、管理ユーザーのパスワードには、パスワードの複雑性の検証機能を使用します。

親トピック: パスワード保護の構成

3.2.10.1 管理ユーザーのパスワード管理について

管理ユーザーのパスワードはデータベースの外部に保存されるため、データベースが開かれていないときでも認証が可能です。

パスワード・ファイルには特別な保護機能はありません。データベースが開いていないときでも認証ができるように、パスワード・ベリファイアはデータベースの外部に保存する必要があります。これまでのリリースでは、パスワードの複雑性の検証機能は管理者以外のユーザーにしか使用できませんでした。Oracle Databaseリリース12c (12.2)以降では、パスワードの複雑性の検証機能を非管理ユーザーと管理ユーザーの両方に使用できます。

親トピック: 管理ユーザーのパスワードの管理

3.2.10.2 管理ユーザーのLOCKおよびEXPIREDステータスの設定

アカウントがロックされている管理者ユーザーはデータベースに接続できません。

- ロックされた管理アカウントまたは期限切れの管理アカウントをロック解除するには、ALTER USER文を使用します。

たとえば:

```
ALTER USER hr_admin ACCOUNT UNLOCK;
```

管理ユーザーのパスワードが期限切れになると、次にユーザーがログインを試みるときに、新しいパスワードの作成を求められます。

親トピック: 管理ユーザーのパスワードの管理

3.2.10.3 管理ユーザーのパスワード・プロファイル設定

管理ユーザーに強制される、いくつかのユーザー・プロファイル・パスワード設定があります。

これらのパスワード・プロファイル・パラメータは次のとおりです。

- FAILED_LOGIN_ATTEMPT
- INACTIVE_ACCOUNT_TIME
- PASSWORD_LOCK_TIME
- PASSWORD_LIFE_TIME
- PASSWORD_GRACE_TIME

関連トピック

- プロファイルによるリソースの管理

親トピック: 管理ユーザーのパスワードの管理

3.2.10.4 管理ユーザーの最後の正常なログイン時間

パスワード・ファイル・ベースの認証を使用する管理ユーザー接続の最後の正常なログイン時間が取得されます。

このログイン時間を確認するには、V$PWFILE_USERS動的パフォーマンス・ビューのLAST_LOGIN列を問い合せます。

親トピック: 管理ユーザーのパスワードの管理

3.2.10.5 管理ユーザーのパスワード・ファイルの管理

ORAPWDユーティリティのFORMATパラメータを12.2に設定すると、管理ユーザーのパスワード・プロファイル・パラメータを管理できます。

パスワード・ファイルは、データベース自体ではなく、外部ファイルに管理ユーザーの資格証明を格納するため、管理ユーザーにとって特に重要です。これにより、管理ユーザーはオープンしていないデータベースにログインして、データ・ディクショナリ・ビューの問合せなどのタスクを実行できます。パスワード・ファイルを作成するには、ORAPWDユーティリティを使用する必要があります。

FORMATパラメータをデフォルトの12.2に設定すると、パスワード・ファイルが管理ユーザー用のパスワード・プロファイル情報に対応するようになります。

たとえば:

```
orapwd file=orapworcl input_file=orapwold format=12.2
...
```

FORMATを12.2に設定すると、次のルールが適用されます。

- パスワードの長さが8文字以上であり、少なくとも数字が1つと英字が1つ含まれていること。
- パスワードに、ユーザー名、またはユーザー名のスペルを逆にしたものが含まれていないこと。
- パスワードにoracle (oracle123など)の語が含まれていないこと。
- パスワードに少なくとも1つの特殊文字が含まれていること。

FORMAT=12.2により次の内部チェックも適用されます。

- パスワードの長さが30バイト以内であること。
- パスワードが二重引用符文字(")を含まないこと。ただし、二重引用符で囲むことができます。

次のユーザー・プロファイル・パスワード設定が管理ユーザーに適用されます。

- FAILED_LOGIN_ATTEMPT
- INACTIVE_ACCOUNT_TIME
- PASSWORD_GRACE_TIME
- PASSWORD_LIFE_TIME
- PASSWORD_LOCK_TIME

パスワード・ファイルに含まれている管理ユーザーおよびその管理権限を確認するには、V$PWFILE_USERS動的ビューを問い合せます。

親トピック: 管理ユーザーのパスワードの管理

3.2.10.6 管理ユーザーのパスワード・ファイルの移行

ORAPWDユーティリティのinput_fileパラメータまたはDBUAを使用して、以前のパスワード・ファイルの形式を12または12.2形式に移行できます。

ORAPWDユーティリティのfileおよびinput_fileパラメータを使用して、またはOracle Database Upgrade Assistant (DBUA)を使用して、以前のパスワード・ファイル形式から12または12.2形式に移行できます。

- ORAPWDのFILEおよびINPUT_FILEパラメータ: ORAPWDユーティリティを使用して移行するには、FILEパラメータに新しいパスワード・ファイルの名前、INPUT_FILEパラメータに以前のパスワード・ファイルの名前を設定します。

  たとえば:

  ```
  orapwd file=orapworcl input_file=orapwold format=12.2
  ```

- DBUA: パスワード・ファイルの以前の形式(FORMAT = LEGACYおよびFORMAT = 12)から移行するには、以前のデータベースを現在のリリースにアップグレードするときにDBUAを使用できます。ただし、データベースが読取り専用モードでオープンしていることを確認します。データベースの読取り専用ステータスを確認するには、V$DATABASE動的ビューのOPEN_MODE列を問い合せます。

関連トピック

- Oracle Database管理者ガイド

親トピック: 管理ユーザーのパスワードの管理

3.2.10.7 管理ユーザーのパスワード・ファイルに対するマルチテナント・オプションの影響

マルチテナント環境では、ローカル管理ユーザーと共通管理ユーザーのパスワード情報はそれぞれ異なる場所に保存されます。

- CDB管理ユーザーの場合: CDBルートで管理権限が付与されたCDB共通管理ユーザーのパスワード情報(パスワードのハッシュ)は、パスワード・ファイルに格納されます。
- CDBルート外で管理権限が付与された、CDB内のすべてのユーザーの場合: これらのユーザーのパスワード・ハッシュ情報に関する情報を確認するには、$PWFILE_USERS動的ビューを問い合せます。

親トピック: 管理ユーザーのパスワードの管理

3.2.10.8 管理ユーザーのパスワードの複雑性の検証機能

セキュリティ向上のため、管理ユーザーのパスワードには、パスワードの複雑性の検証機能を使用します。

次のことに注意してください。

- プロファイル: SYSユーザーにパスワードの複雑性の検証機能を指定するには、CREATE PROFILEまたはALTER PROFILE文のPASSWORD_VERIFY_FUNCTION句を使用します。管理ユーザーのパスワードをより適切に保護するために、パスワードの検証ファンクションを使用することをお薦めします。
- ORAPWDパスワード・ファイル: ORAPWDユーティリティを使用してパスワード・ファイルを作成した場合、Oracle Databaseでは、SYSユーザーに加えて、SYSDBA、SYSBACKUP、SYSDGおよびSYSKM管理権限を使用してログインした管理ユーザーに対してパスワードの複雑性のチェックが強制されます。

  パスワードは、次の要件についてチェックされます。

  - パスワードの長さが8文字以上であり、数字、英字および特殊文字が少なくとも1つずつ含まれていること。
  - パスワードがユーザー名またはユーザー名のスペルを逆にしたものと同一でないこと。
  - パスワードにoracle (oracle123など)の語が含まれていないこと。
  - 以前のパスワードとの違いが3文字以上あること。

  次の内部チェックも適用されます。

- パスワードの長さが30バイト以内であること。
- パスワードが二重引用符文字(")を含まないこと。ただし、二重引用符で囲むことができます。

関連トピック

- パスワードの複雑度の管理

親トピック: 管理ユーザーのパスワードの管理

3.3 データベース管理者の認証

データベース管理者を認証するには、強力な認証を使用するか、オペレーティング・システムから行うか、パスワードを使用してデータベースから行います。

- データベース管理者の認証について
  データベース管理者は、データベースの起動や停止などの特別な管理操作を実行します。
- 管理者の厳密認証と集中管理
  一元管理するデータベースの厳密認証方式として、ディレクトリ認証、Kerberos認証、SSL認証があります。
- オペレーティング・システムを使用したデータベース管理者の認証
  WindowsおよびUNIXの両システムで、DBA権限のあるグループを使用してオペレーティング・システムに対して認証します。
- パスワードを使用したデータベース管理者の認証
  パスワード・ファイルを使用してデータベース管理者を認証します。
- データベース管理者認証のパスワード・ファイルを使用するリスク
  パスワード・ファイルの使用は、セキュリティ上のリスクを伴う可能性があることに注意してください。

親トピック: 認証の構成

3.3.1 データベース管理者の認証について

データベース管理者は、データベースの起動や停止などの特別な管理操作を実行します。

Oracle Databaseには、SYSDBA、SYSOPER、SYSBACKUP、SYSDGまたはSYSKM管理権限を持つデータベース管理者の認証を保護するための方式が用意されています。

親トピック: データベース管理者の認証

3.3.2 管理者の厳密認証と集中管理

一元管理するデータベースの厳密認証方式として、ディレクトリ認証、Kerberos認証、SSL認証があります。

- データベース管理者の厳密認証について
  厳密認証を使用すると、複数のデータベースに対するSYSDBAおよびSYSOPERのアクセスを集中管理できます。
- 管理ユーザーのディレクトリ認証の構成
  Oracle Internet Directoryで、管理ユーザーのディレクトリ認証を構成します。
- 管理ユーザーのKerberos認証の構成
  Oracle Internet Directoryを使用して、管理ユーザーのKerberos認証を構成できます。
- Transport Layer Securityを使用したユーザー認証の構成
  Transport Layer Security (TLS)を使用して、クライアント側とサーバー側の両方で管理ユーザーを認証できます。

親トピック: データベース管理者の認証

3.3.2.1 データベース管理者の厳密認証について

厳密認証を使用すると、複数のデータベースに対するSYSDBAおよびSYSOPERのアクセスを集中管理できます。

データベース管理のこのような認証は、次の状況で使用を検討してください。

- パスワード・ファイルの脆弱性が懸念される場合。
- サイトで非常に強固なセキュリティが要求される場合。
- アイデンティティ管理をデータベースから分離する必要がある場合。たとえば、Oracle Internet Directory(OID)などのディレクトリ・サーバーを使用すると、そのサーバーを個別にメンテナンス、保護および管理できます。

Oracle Internet Directoryサーバーを使用してSYSDBAおよびSYSOPERの接続を認可するには、使用している環境に応じて、この項で説明する次のいずれかの方式を使用します。

親トピック: 管理者の厳密認証と集中管理

3.3.2.2 管理ユーザーのディレクトリ認証の構成

Oracle Internet Directoryで、管理ユーザーのディレクトリ認証を構成します。

1. 通常のユーザーを構成するのと同じ手順で、管理ユーザーを構成します。
2. Oracle Internet Directoryで、このユーザーが管理するデータベースに対して、SYSDBAまたはSYSOPER管理権限をユーザーに付与します。

   SYSDBAまたはSYSOPERは信頼できるユーザーにのみ付与してください。
3. LDAP_DIRECTORY_SYSAUTH初期化パラメータをYESに設定します。

   ```
   ALTER SYSTEM SET LDAP_DIRECTORY_SYSAUTH = YES;
   ```

   LDAP_DIRECTORY_SYSAUTHパラメータをYESに設定すると、SYSDBAおよびSYSOPERユーザーは、厳密認証方式を使用してデータベースへの認証を行うことができます。
4. LDAP_DIRECTORY_ACCESSパラメータをPASSWORDまたはSSLのいずれかに設定します。たとえば、次のようにします。

   ```
   ALTER SYSTEM SET LDAP_DIRECTORY_ACCESS = PASSWORD;
   ```

   LDAP_DIRECTORY_ACCESS初期化パラメータをNONEに設定しないでください。このパラメータをPASSWORDまたはSSLに設定すると、Oracle Internet DirectoryからSYSDBAまたはSYSOPER管理権限を使用してユーザーを認証できます。

   Oracle Real Application Clusters (Oracle RAC)環境では、ALTER SYSTEM文またはinit.oraファイルで、すべてのインスタンスのLDAP_DIRECTORY_ACCESS設定が同じであることを確認します。

   Oracle Data GuardまたはActive Data Guard環境では、スタンバイ・データベースのLDAP_DIRECTORY_ACCESS設定がプライマリ・データベースと同じであることを確認します。この環境では、ALTER SYSTEM文を使用すると、その設定がプライマリ・データベースからスタンバイ・データベースに伝播されます。init.oraのパラメータはプライマリ・データベースとスタンバイ・データベースの両方で使用されているので、init.oraファイルを更新する場合に、この設定をデータベース間で手動で伝播する必要はありません。

この結果、ユーザーは、SQL*PlusでCONNECT文にネット・サービス名を指定してログインできるようになります。たとえば、ネット・サービス名がorclの場合、SYSDBAとしてログインするには、次のように入力します。

```
CONNECT someuser@orcl AS SYSDBA
```

```
Enter password: password
```

リモート認証でパスワード・ファイルを使用するようにデータベースが構成されている場合、Oracle Databaseでは最初にパスワード・ファイルをチェックします。

関連トピック

- ユーザー・アカウントと権限の保護に関するガイドライン
- Oracle Databaseリファレンス
- Oracle Databaseリファレンス

親トピック: 管理者の厳密認証と集中管理

3.3.2.3 管理ユーザーのKerberos認証の構成

Oracle Internet Directoryを使用して、管理ユーザーのKerberos認証を構成できます。

1. 通常のユーザーを構成するのと同じ手順で、管理ユーザーを構成します。

   詳細は、Kerberos認証の構成を参照してください。

2. Kerberos認証用にOracle Internet Directoryを構成します。

   詳細は、『Oracle Databaseエンタープライズ・ユーザー・セキュリティ管理者ガイド』を参照してください。

3. Oracle Internet Directoryで、このユーザーが管理するデータベースに対して、SYSDBAまたはSYSOPER管理権限をユーザーに付与します。

   SYSDBAまたはSYSOPERは信頼できるユーザーにのみ付与してください。この項の内容に関するガイドラインは、ユーザー・アカウントと権限の保護に関するガイドラインを参照してください。

4. LDAP_DIRECTORY_SYSAUTH初期化パラメータをYESに設定します。

   ```
   ALTER SYSTEM SET LDAP_DIRECTORY_SYSAUTH = YES;
   ```

   LDAP_DIRECTORY_SYSAUTHパラメータをYESに設定すると、SYSDBAおよびSYSOPERユーザーは、厳密認証方式を使用してデータベースへの認証を行うことができます。LDAP_DIRECTORY_SYSAUTHの詳細は、『Oracle Databaseリファレンス』を参照してください。

5. LDAP_DIRECTORY_ACCESSパラメータをPASSWORDまたはSSLのいずれかに設定します。たとえば、次のようにします。

   ```
   ALTER SYSTEM SET LDAP_DIRECTORY_ACCESS = SSL;
   ```

   LDAP_DIRECTORY_ACCESS初期化パラメータをNONEに設定しないでください。このパラメータをPASSWORDまたはSSLに設定すると、Oracle Internet DirectoryからSYSDBAまたはSYSOPERを使用してユーザーを認証できます。LDAP_DIRECTORY_ACCESSの詳細は、『Oracle Databaseリファレンス』を参照してください。

   Oracle Real Application Clusters (Oracle RAC)環境では、ALTER SYSTEM文またはinit.oraファイルで、すべてのインスタンスのLDAP_DIRECTORY_ACCESS設定が同じであることを確認します。

   Oracle Data GuardまたはActive Data Guard環境では、スタンバイ・データベースのLDAP_DIRECTORY_ACCESS設定がプライマリ・データベースと同じであることを確認します。この環境では、ALTER SYSTEM文を使用すると、その設定がプライマリ・データベースからスタンバイ・データベースに伝播されます。init.oraのパラメータはプライマリ・データベースとスタンバイ・データベースの両方で使用されているので、init.oraファイルを更新する場合に、この設定をデータベース間で手動で伝播する必要はありません。

この結果、ユーザーは、SQL*PlusでCONNECT文にネット・サービス名を指定してログインできるようになります。たとえば、ネット・サービス名がorclの場合、SYSDBAとしてログインするには、次のように入力します。

```
CONNECT /@orcl AS SYSDBA
```

親トピック: 管理者の厳密認証と集中管理

3.3.2.4 Transport Layer Securityを使用したユーザー認証の構成

Transport Layer Security (TLS)を使用して、クライアント側とサーバー側の両方で管理ユーザーを認証できます。

1. クライアントとサーバーの両方は、同じルート認証局(CA)証明書で署名されたユーザー証明書(公開または自己署名)を取得します。
2. TLSを使用するようにクライアントを構成します。
   a. クライアント・ウォレットに署名付きユーザー証明書を追加します。CAルート信頼証明書は、クライアント・ウォレットに存在している必要があります。ユーザー証明書の追加前に、ユーザー証明書に必要な中間証明書がウォレットに追加されていることを確認します。
   orapkiを使用して、クライアント・ウォレットおよびユーザー証明書を構成できます。
   b. sqlnet.oraファイルで認証サービスとしてTLSを設定します。

   ```
   SSL_CLIENT_AUTHENTICATION=TRUE
   ```

   c. オプションで、セキュリティを向上させるために、完全または部分的なDN一致を使用するようにクライアントを設定します。
   DN一致が有効になっている場合、クライアントはサーバー証明書をチェックして、ホスト名がクライアントで一致するように構成されたものと一致することを確認します。このステップは、Oracle Internet DirectoryでTLSの使用を有効にする場合に実行します。

   > ノート:
   >
   > データベース・クライアントとサーバーは、最強の TLS プロトコルと暗号スイートを使用して接続を確立するようになります。そのため、この TLS バージョンと暗号スイートは、それが必要になる特定のセキュリティ要件がないかぎり、指定する必要はありません。特定の TLS バージョンと暗号スイートを設定する場合は、古いバージョンが使用されなくなったときに構成の更新が必要になる点に注意してください。

3. クライアント、リスナーおよびサーバーでTLSのリスナーを構成します。
   a. 安全なデータベース・ポート1522を使用して、TLS接続用に個別のリスナー・エントリを作成します。
   たとえば:

   ```
   LISTENER =
     (DESCRIPTION_LIST =
       (DESCRIPTION =
         (ADDRESS = (PROTOCOL = TCP)(HOST = example.com)(PORT = 1521))
         (ADDRESS = (PROTOCOL = TCPS)(HOST = example.com)(PORT = 1522))
       )
     )
   ```

   b. TLS以外のリスナー・エントリ(PROTOCOL = TCPの行など)をコメント・アウトするか、TLS以外の必須接続の場合はそのままにします。
   c. データベース・サーバーがリスナーではなくクライアントを認証するように、sqlnet.oraファイルにSSL_CLIENT_AUTHENTICATION = FALSEを追加します。
   サーバーが使用するものと同じウォレットを、同じサーバー証明書とともにリスナーで使用できます。リスナーは、

標準のOracle Databaseウォレット検索順序を使用してウォレットを検索するようになります。または、WALLET_LOCATIONパラメータを設定すると、リスナーのウォレットの場所を指定できます。(リスナーでは使用できないため、この目的にWALLET_ROOTパラメータは使用できません)。

4. TLSを使用するようにサーバーを構成します。
   a. sqlnet.oraファイルで、SSL_CLIENT_AUTHENTICATIONをFALSE (またはOFF)に設定して一方向TLSを有効にします。
   b. TLSサーバー・ウォレットの場合は、次を実行します。
      - WALLET_ROOTパラメータをTLSサーバーの場所に設定します。
      - WALLET_ROOT/pdb_guidの下にtlsディレクトリを作成します。
      - TLSサーバー・ウォレットをWALLET_ROOT/pdb_guid/tlsディレクトリに移動します。
   c. sqlnet.oraファイルで、次のパラメータを追加します。

```
SSL_CLIENT_AUTHENTICATION=TRUE
```

   認証をTCPSのみに制限する場合は、AUTHENTICATION_SERVICESをTCPSに設定します。

5. 新しいスキーマを作成するか、ユーザーにマップするように既存のスキーマを変更します。

```
CREATE USER user_name IDENTIFIED EXTERNALLY AS 'user DN on certificate';
```

6. SYSDBAやSYSOPERなどの適切な管理権限にデータベース・スキーマを付与します。
   TLS認証を使用する管理ユーザーはTLSで認証できます。こうしたユーザーを有効にするには、適切な管理権限をユーザー・スキーマに付与します。管理ユーザーは、この管理権限を使用してログインする必要があります。SYSOPER管理権限を付与されたユーザーの例:

```
CONNNECT /@pdb_name AS SYSOPER
```

この結果、ユーザーは、SQL*PlusでCONNECT文にネット・サービス名を指定してログインできるようになります。たとえば、ネット・サービス名がorclの場合、SYSDBAとしてログインするには、次のように入力します。

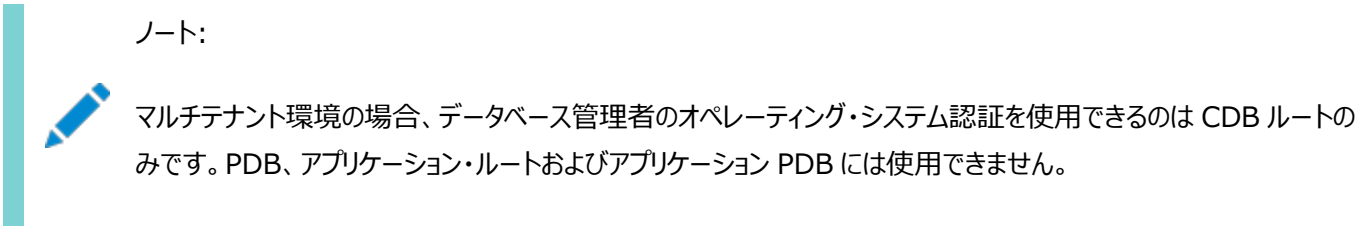
```
CONNECT /@orcl AS SYSDBA
```

親トピック: 管理者の厳密認証と集中管理

3.3.3 オペレーティング・システムを使用したデータベース管理者の認証

WindowsおよびUNIXの両システムで、DBA権限のあるグループを使用してオペレーティング・システムに対して認証します。

通常、データベース管理者のオペレーティング・システム認証には、オペレーティング・システムにグループを作成すること、そのグループにDBA権限を付与すること、および権限を付与する管理者の名前をグループに追加することが含まれます。(UNIXシステムでは、このグループはdbaグループです。)



ノート:

マルチテナント環境の場合、データベース管理者のオペレーティング・システム認証を使用できるのは CDB ルートのみです。PDB、アプリケーション・ルートおよびアプリケーション PDB には使用できません。

Microsoft Windowsシステムの場合:

- SYSDBA管理権限で接続するユーザーはWindowsネイティブ認証を利用できます。このユーザーがドメイン・アカウントを使用してOracle Databaseを操作する場合は、ローカル管理権限およびORA_DBAメンバーシップを明示的に

付与する必要があります。

- Microsoft Windows組込みのアカウントではなく、権限の低いMicrosoft Windowsのユーザー・アカウントを使用して、Oracle Databaseサービスを実行することをお薦めします。

関連項目:

データベース管理者のオペレーティング・システム認証の構成の詳細は、オペレーティング・システム固有のOracle Databaseマニュアルを参照してください。

親トピック: データベース管理者の認証

3.3.4 パスワードを使用したデータベース管理者の認証

パスワード・ファイルを使用してデータベース管理者を認証します。

つまり、SYSDBA、SYSOPER、SYSASM、SYSBACKUP、SYSDGおよびSYSKM管理権限を付与されたOracle Databaseユーザーは、データベース固有のパスワード・ファイルを使用して最初に認証されます。

これらの権限によって、次のアクティビティが使用可能になります。

- SYSOPERシステム権限によって、データベース管理者はSTARTUP、SHUTDOWN、ALTER DATABASE OPEN/MOUNT、ALTER DATABASE BACKUP、ARCHIVE LOGおよびRECOVERの各操作を実行できます。また、SYSOPER権限には、RESTRICTED SESSION権限も含まれます。
- SYSDBA管理権限には、ADMIN OPTIONおよびSYSOPER管理権限も含めて、すべてのシステム権限が含まれます。CREATE DATABASEと時間ベースのリカバリが許可されます。
- SYSDBA、SYSOPER、SYSASM、SYSBACKUP、SYSDGおよびSYSKM管理権限を持つユーザーが含まれるパスワード・ファイルは、異なるデータベース間で共有できます。また、このタイプのパスワード・ファイル認証は、Transport Layer Security (TLS)またはKerberos構成で、マルチテナント環境の共通管理ユーザーに対して使用できます。SYSユーザー以外のユーザーが含まれた共有パスワード・ファイルも保持できます。異なるデータベース間でパスワード・ファイルを共有するには、init.oraファイルのREMOTE_LOGIN_PASSWORDFILEパラメータをSHAREDに設定します。

  REMOTE_LOGIN_PASSWORDFILE初期化パラメータの設定をNONEからEXCLUSIVEまたはSHAREDに変更する場合は、パスワード・ファイルとディクショナリ・パスワードを必ず同期してください。詳細は、Oracle Database管理者ガイドを参照してください。
- 自動ストレージ管理(ASM)環境では、共有ASMパスワード・ファイルを作成できます。ASMパスワード・ファイルを作成するSYSASMシステム権限が必要なことに注意してください。詳細は、『Oracle Automatic Storage Management管理者ガイド』を参照してください。
- SYSDG シャーディング管理者がファイル転送およびOracle Recovery Manager (RMAN)のアクティビティを伴うタスクを実行できるように、パスワード・ファイルに管理権限を含める必要があります。
- パスワード・ファイル・ベースの認証が、デフォルトで使用可能です。これは、SYSDBA、SYSOPER、SYSASM、SYSBACKUP、SYSDGおよびSYSKM管理権限を持つユーザーを認証するために、データベースでパスワード・ファイルを使用する準備が整っていることを意味します。パスワード・ファイル・ベースの認証は、ORAPWDユーティリティを使用してパスワード・ファイルを作成すると、アクティブになります。

  $ORACLE_HOME/dbsディレクトリに対してEXECUTE権限および書込み権限を持つユーザーがORAPWDユーティリ

ティを実行できます。

- Oracle Database 12cリリース2 (12.2)形式でパスワード・ファイルが作成されると、FAILED_LOGIN_ATTEMPTSやPASSWORD_LIFE_TIMEなどのパスワード制限が管理ログインに対して強制されます。

ノート:

- パスワード・ファイルに含まれているユーザーのリストを検索するには、V$PWFILE_USERS データ・ディクショナリ・ビューを問い合せることができます。
- AS SYSDBA または AS SYSOPER で要求された接続は、これらのフレーズを使用する必要があります。使用していない場合、接続は失敗します。

親トピック: データベース管理者の認証

3.3.5 データベース管理者認証のパスワード・ファイルを使用するリスク

パスワード・ファイルの使用は、セキュリティ上のリスクを伴う可能性があることに注意してください。

このため、管理者の厳密認証と集中管理で説明した認証方式を使用することを検討してください。

パスワードによるセキュリティのリスクの例は、次のとおりです。

- 侵入者がパスワード・ファイルを盗んだり攻撃する可能性があります。
- 多くのユーザーがデフォルト・パスワードを変更しない場合があります。
- パスワードが簡単に推定される場合があります。
- パスワードがディレクトリ内に存在すると、無防備になります。
- 短かすぎたり簡単に入力できるパスワードは、侵入者がパスワードの暗号化ハッシュを取得した場合に無防備になります。

ノート:

パスワード・ファイルの作成およびメンテナンスの詳細は、『Oracle Database 管理者ガイド』を参照してください。

親トピック: データベース管理者の認証

3.4 ユーザーのデータベース認証

ユーザーのデータベース認証では、認証を実行するためにデータベース自体の情報を使用する必要があります。

- ユーザーのデータベース認証について
  Oracle Databaseでは、データベース自体に格納されている情報を使用して、データベースに接続しようとするユーザーを認証できます。
- データベース認証の利点

データベースを使用したユーザーの認証では、次の3つのメリットがあります。

- データベースで認証されるユーザーの作成
  データベースで認証されるユーザーを作成する場合、このユーザーにパスワードを割り当てます。

親トピック: 認証の構成

3.4.1 ユーザーのデータベース認証について

Oracle Databaseでは、データベース自体に格納されている情報を使用して、データベースに接続しようとするユーザーを認証できます。

データベース認証を使用するようにOracle Databaseを構成するには、対応するパスワードを指定して各ユーザーを作成する必要があります。ユーザー名にはNational Language Support (NLS)の文字書式を使用できますが、パスワードに二重引用符文字を含めることはできません。ユーザーは、接続の確立時にそのユーザー名およびパスワードを入力する必要があります。

Oracle Databaseは、ユーザーのパスワードの一方向ハッシュを生成し、入力されたログイン・パスワードの検証時に使用するため格納します。古いクライアントをサポートするために、様々なハッシング・アルゴリズムを使用してユーザーのパスワードの一方向ハッシュを生成するようにOracle Databaseを構成できます。生成されたパスワード・ハッシュはパスワード・バージョンと呼ばれ、その略称は10G、11Gおよび12Cです。略称10G、11Gおよび12Cは、一方向パスワード・ハッシング・アルゴリズムの詳細を略したものに相当します。その詳細は、ドキュメントとしてDBA_USERSビューのPASSWORD_VERSIONS列で説明されています。指定したユーザーのパスワード・バージョンのリストを確認するには、DBA_USERSビューのPASSWORD_VERSIONS列を問い合せます。

デフォルトでは、Oracle Databaseで現在使用されている一方向ハッシング・アルゴリズムには、salt処理済のSHA-1ハッシング・アルゴリズムとsalt処理済のPKBDF2 SHA-2 SHA-512ハッシング・アルゴリズムの2つのバージョンがあります。salt処理済のSHA-1ハッシング・アルゴリズムでは、11Gパスワード・バージョンに使用されるハッシュが生成されます。salt処理済のPKBDF2 SHA-2 SHA-512ハッシング・アルゴリズムでは、12Cパスワード・バージョンに使用されるハッシュが生成されます。このハッシュ生成は同じパスワードに対して行われます。つまり、両方のアルゴリズムが同じパスワードに対して実行されます。Oracle Databaseでは、これらのパスワード・バージョンがDBA_USERSデータ・ディクショナリ・ビューに記録されます。このビューを問い合せると、2つのパスワード・バージョンが表示されます。例:

```
SELECT USERNAME, PASSWORD_VERSIONS FROM DBA_USERS;
USERNAME  PASSWORD_VERSIONS
--------  -----------------
ADAMS     11G, 12C
SYS       11G, 12C
...
```

クライアントまたはクライアントとして機能するデータベース・サーバーの認証中に許可する認証プロトコルを指定するには、サーバーsqlnet.oraファイルのSQLNET.ALLOWED_LOGON_VERSION_SERVERパラメータを明示的に設定できます。(このパラメータのクライアント・バージョンは、SQLNET.ALLOWED_LOGON_VERSION_CLIENTです。)各接続がテストされ、パートナが指定するクライアント機能要件をクライアントまたはサーバーが満たしていない場合は、認証に失敗して、『Oracle Database Net Servicesリファレンス』のSQLNET.ALLOWED_LOGON_VERSION_SERVERパラメータの説明の下にあるSQLNET.ALLOWED_LOGON_VERSION_SERVER設定の表のクライアントに必要な機能の列に示されている「ORA-28040 一致する認証プロトコルがありません」エラーが発生します。このパラメータの値には、12a、12、11、10、9または8を指定できます。デフォルト値は12で、排他モードです。これらの値は、認証プロトコルのバージョンを表します。値12をお薦めします。ただし、SQLNET.ALLOWED_LOGON_VERSION_SERVERおよびSQLNET.ALLOWED_LOGON_VERSION_CLIENTを11に設定した場合、JDBCシン・クライアントを含むOracle Databaseリリース11.1以前のクライアント・アプリケーションは、パスワード・ベースの認証を使用してOracleデータベースへの

認証を行うことができない点に注意してください。

データベース認証使用時のセキュリティを高めるために、アカウントのロック、パスワード・エイジングと期限切れ、パスワード履歴およびパスワードの複雑度検証も含めたパスワード管理の使用をお薦めします。

外部認証を使用せず、ローカル・データベース・パスワード認証のみを使用する場合は、クライアントの`sqlnet.ora`ファイルで`AUTHENTICATION_SERVICES=(none)`を設定します。この設定では、この値のデフォルトは`ALL`であり、クライアントが外部認証およびデータベース・パスワード認証を確認するように強制されるため、パフォーマンスが向上します。

関連トピック

- SQLNET.ALLOWED_LOGON_VERSION_CLIENT
- SQLNET.ALLOWED_LOGON_VERSION_SERVER
- SQLNET.AUTHENTICATION_SERVICES
- パスワードの複雑度検証について
- パスワード管理ポリシーの使用
- ユーザーのパスワード・バージョンの管理

親トピック: ユーザーのデータベース認証

3.4.2 データベース認証の利点

データベースを使用したユーザーの認証では、次の3つのメリットがあります。

これらのメリットは次のとおりです。

- ユーザー・アカウントとすべての認証がデータベースによって制御されます。データベースの外部のものには依存しません。
- Oracle Databaseには、データベース認証使用時のセキュリティを高めるために、強力なパスワード管理機能が組み込まれています。
- 小規模なユーザー・コミュニティがある場合の管理が容易になります。

親トピック: ユーザーのデータベース認証

3.4.3 データベースで認証されるユーザーの作成

データベースで認証されるユーザーを作成する場合、このユーザーにパスワードを割り当てます。

- データベースで認証されるユーザーを作成するには、ユーザーの作成時に`IDENTIFIED BY`句を指定します。

たとえば、次のSQL文は、Oracle Databaseによって識別および認証されるユーザーを作成します。ユーザー`sebastian`は、Oracle Databaseに接続するたびに割り当てられたパスワードを指定する必要があります。

```
CREATE USER sebastian IDENTIFIED BY password;
```

関連トピック

- ユーザー・アカウントの作成

親トピック: ユーザーのデータベース認証

3.5 スキーマ限定アカウント

スキーマ限定アカウントを作成できます。つまり、スキーマ・ユーザーにはパスワードがありません。

- スキーマ限定アカウントについて
  スキーマ限定アカウントはデータベースにログインできませんが、単一のセッション・プロキシでプロキシできます。
- スキーマ限定アカウントの作成
  CREATE USER SQL文を使用して、スキーマ限定アカウントを作成します。
- スキーマ限定アカウントの変更
  ALTER USER SQL文を使用して、スキーマ限定アカウントを変更できます。

親トピック: 認証の構成

3.5.1 スキーマ限定アカウントについて

スキーマ限定アカウントはデータベースにログインできませんが、単一セッション・プロキシでプロキシとなることができます。

このタイプのアカウントは、Oracle提供のスキーマおよび一部のユーザー作成スキーマ用に設計され、パスワードや認証タイプを指定せずに作成できます。これは、ALTER USER文を使用して認証方式が割り当てられない限り認証できません。スキーマ限定アカウントでは、DBA_USERS_WITH_DEFPWDデータ・ディクショナリ・ビューにエントリが含まれていません。

デフォルトでは、サンプル・スキーマ・ユーザー・アカウント(たとえばHR)など、Oracle Databaseで使用可能な事前定義済のスキーマ・ユーザー・アカウントのほとんどがスキーマ限定アカウントです。これらのアカウントのパスワードは、必要に応じて割り当てることができますが、セキュリティを強化するために、後でスキーマ限定に戻すように設定することをお薦めします。スキーマ・ユーザー・アカウントがスキーマ限定かどうかを確認するには、DBA_USERSデータ・ディクショナリ・ビューのAUTHENTICATION_TYPE列を問い合せます。NONEは、アカウントがスキーマ限定であることを示します。

スキーマ限定アカウントの使用に関する次のルールに注意してください。

- スキーマ限定アカウントは管理者アカウントおよび非管理者アカウントの両方に使用できます。
- スキーマ限定アカウントはデータベース・インスタンスでのみ作成する必要があり、Oracle Automatic Storage Management (ASM)環境では作成できません。
- スキーマ限定アカウントにはシステム権限(CREATE ANY TABLEなど)や管理者ロール(DBAなど)を付与できます。スキーマ限定アカウントは、付与された権限を修正する必要があることを前提として、表またはプロシージャのようなオブジェクトを作成できます。
- スキーマ限定アカウントは、単一セッション・プロキシのプロキシ認証でクライアント・ユーザーとして使用されるように構成できます。これは単一セッション・プロキシの場合、プロキシ・ユーザーの資格証明のみが検証され、クライアント・ユーザーの資格証明が検証されないためです。したがって、スキーマ限定アカウントがクライアント・ユーザーになることができます。ただし、2つのプロキシが存在するシナリオの場合はクライアントの資格証明を検証する必要があるため、スキーマ限定アカウントを構成することはできません。したがって、スキーマ限定アカウントの認証は失敗します。
- スキーマ限定アカウントは、接続されたユーザー・リンク、固定ユーザー・リンク、または現行ユーザー・リンクのいずれかを使用してデータベース・リンク経由で接続することはできません。

関連トピック

- 事前定義されたサンプル・スキーマ・ユーザー・アカウント

親トピック: スキーマ限定アカウント

3.5.2 スキーマ限定アカウントの作成

CREATE USER SQL文でスキーマ限定アカウントを作成します。

NO AUTHENTICATION句を指定したCREATE USER文は、データベース・インスタンス上でのみ実行できます。Oracle Automatic Storage Management(ASM)インスタンス上ではこれを実行できません。

- NO AUTHENTICATION句を指定してCREATE USER文を使用します。
  たとえば:

```
CREATE USER psmith NO AUTHENTICATION;
```

親トピック: スキーマ限定アカウント

3.5.3 スキーマ限定アカウントの変更

ALTER USER SQL文でスキーマ限定アカウントを変更できます。

1. スキーマ・ユーザーが管理者権限を持っているかどうかを確認します。
   V$PWFILE_USERSを問い合せてスキーマ・ユーザーが管理者権限を持っているかどうかを確認できます。
2. スキーマ・ユーザーが管理者権限を持っている場合、REVOKE文を使用して、これらの権限を取り消します。
3. NO AUTHENTICATION句を指定してALTER USER SQL文を使用して、スキーマ・アカウントが認証されないように修正します。
   たとえば:

```
ALTER USER psmith NO AUTHENTICATION;
```

ALTER USERを使用してスキーマ限定アカウントの認証を有効にできます。

親トピック: スキーマ限定アカウント

3.6 ユーザーのオペレーティング・システム認証

Oracle Databaseではオペレーティング・システムで管理されている情報を使用した認証が可能です。

オペレーティング・システムを使用したユーザーの認証には、メリットとデメリットの両方があります。

この機能には次の利点があります。

- オペレーティング・システムから認証を受けたユーザーは、より簡単にOracle Databaseに接続できます。ユーザー名やパスワードを指定する必要はありません。たとえば、オペレーティング・システムにより認証されたユーザーはSQL*Plusを起動して、コマンドラインで次のコマンドを入力することによりユーザー名とパスワードのプロンプトを省略できます。

```
SQLPLUS /
```

  SQL*Plusで、次のように入力します。

```
CONNECT /
```

- ユーザー認証はオペレーティング・システムで集中管理されるため、ユーザー・パスワードの暗号化されたハッシュ値(ベリファイアとも呼ばれる)をOracle Databaseが格納または管理する必要がなくなります。ただし、ユーザー名は引き続きデータベース内で管理されます。
- 監査証跡は、オペレーティング・システムのユーザー名とデータベース・ユーザー名を取得します。この場合データベース・ユーザー名には、OS_AUTHENT_PREFIXインスタンス初期化パラメータの値がオペレーティング・システムのユーザー名に接頭辞として付加されます。たとえば、OS_AUTHENT_PREFIXがOPS$に設定されていて、オペレーティング・システムのユーザー名がpsmithである場合、データベース・ユーザー名はOPS$PSMITHになります。

- 同じシステムで、オペレーティング・システム・ユーザーと非オペレーティング・システム・ユーザーの両方を認証できます。たとえば:
    - オペレーティング・システムによってユーザーを認証します。CREATE USER文のIDENTIFIED EXTERNALLY句を使用してユーザー・アカウントを作成し、OS_AUTHENT_PREFIX初期化パラメータを設定して、サーバーに接続しようとするユーザーをOracle Databaseが認証するために使用する接頭辞を指定します。
    - 非オペレーティング・システム・ユーザーを認証します。これは、パスワードが割り当てられ、データベースによって認証されるユーザーです。
    - Oracle Database Enterprise User Securityユーザーを認証します。このユーザー・アカウントはCREATE USER文のIDENTIFIED GLOBALLY句を使用して作成され、現行の同じデータベースでOracle Internet Directory(OID)によって認証されます。

ただし、ユーザーの認証にオペレーティング・システムを使用する場合は特別な注意が必要です。

- ユーザーには、アクセスが必要なコンピュータのオペレーティング・システム・アカウントが必要です。必ずしもすべてのユーザー(特に管理ユーザー以外のユーザー)がオペレーティング・システム・アカウントを持っているわけではありません。
- ユーザーがこの方式を使用してログインし、端末の前から離れた場合、別のユーザーはパスワードや資格証明が必要ないため簡単にログインできます。これは、深刻なセキュリティ問題になる可能性があります。
- データベース・ユーザーの認証にオペレーティング・システムを使用する場合は、分散データベース環境とデータベース・リンクの管理に特別な注意が必要です。オペレーティング・システム認証のデータベース・リンクは、セキュリティ上の弱点を生み出す可能性があります。そのため、これらのリンクは使用しないことをお薦めします。
- マルチテナント環境の場合、データベース管理者のオペレーティング・システム認証を使用できるのはCDBルートのみです。PDB、アプリケーション・ルートおよびアプリケーションPDBには使用できません。

関連項目:

- 認証、オペレーティング・システム、分散データベースの概要および分散データ管理の詳細は、『Oracle Database管理者ガイド』を参照してください。
- オペレーティング・システムによる認証の詳細は、そのオペレーティング・システム固有のOracle Databaseマニュアルを参照してください。

親トピック: 認証の構成

3.7 ユーザーのネットワーク認証

ネットワークでのユーザーの認証は、サード・パーティ・サービスでTransport Layer Securityを使用して行うことができます。

- Transport Layer Securityを使用した認証
  Transport Layer Security (TLS)プロトコルは、アプリケーション・レイヤー・プロトコルです。
- サード・パーティ・サービスを使用した認証
  サード・パーティのサービス(Kerberos、RADIUS、ディレクトリベース・サービス、公開キー・インフラストラクチャ)を使用して、ネットワーク経由でOracle Databaseを認証できます。

親トピック: 認証の構成

3.7.1 Transport Layer Securityを使用した認証

Transport Layer Security (TLS)プロトコルは、アプリケーション・レイヤー・プロトコルです。

TLSは、Oracle Internet Directoryでのグローバル・ユーザー管理とは関係なく、データベースに対するユーザー認証に使用できます。つまり、ユーザーは、ディレクトリ・サーバーを指定しなくても、TLSを使用してデータベースへの認証を行うことができます。

関連トピック

- Transport Layer Security認証の構成

親トピック: ユーザーのネットワーク認証

3.7.2 サード・パーティ・サービスを使用した認証

サード・パーティのサービス(Kerberos、RADIUS、ディレクトリベース・サービス、公開キー・インフラストラクチャ)を使用して、ネットワーク経由でOracle Databaseを認証できます。

- サード・パーティ・サービスを使用した認証について
  ネットワーク上のOracle Databaseユーザーを認証する場合は、サード・パーティ・ネットワーク認証サービスを使用する必要があります。
- Kerberosを使用した認証
  Kerberosは、共有秘密を使用するサード・パーティの認証システムです。
- RADIUSを使用した認証
  Remote Authentication Dial-In User Service (RADIUS)はユーザー認証、認可、およびアカウンティングに使用される標準のライトウェイト・プロトコルです。
- ディレクトリベース・サービスを使用した認証
  中核となるディレクトリを使用すると、認証とその管理が効率的になります。
- 公開キー・インフラストラクチャを使用した認証
  公開キー・インフラストラクチャ(PKI)に基づく認証システムでは、ユーザー・クライアントに電子証明書が発行されます。

親トピック: ユーザーのネットワーク認証

3.7.2.1 サード・パーティ・サービスを使用した認証について

ネットワーク上のOracle Databaseユーザーを認証する場合は、サード・パーティ・ネットワーク認証サービスを使用する必要があります。

よく知られている例としては、Kerberos、PKI (公開キー・インフラストラクチャ)、RADIUS (Remote Authentication Dial-In User Service)、およびディレクトリ・ベース・サービスがあります。

ネットワーク認証サービスを使用できる場合、Oracle Databaseはこれらのネットワーク・サービスによる認証を受け入れることができます。ネットワーク認証サービスを使用する場合は、ネットワーク・ロールとデータベース・リンクについて特別な考慮事項があります。

親トピック: サード・パーティ・サービスを使用した認証

3.7.2.2 Kerberosを使用した認証

Kerberosは、共有秘密を使用するサード・パーティの認証システムです。

Kerberosは、サード・パーティがセキュアであることを保証し、シングル・サインオン機能、集中化されたパスワード・ストレージ、

データベース・リンク認証、拡張されたPCセキュリティを提供します。これは、Kerberos認証サーバーまたはCybersafe Active Trust (Kerberosをベースとした商用の認証サーバー)を介して提供されます。

関連トピック

- Kerberos認証の構成

親トピック: サード・パーティ・サービスを使用した認証

3.7.2.3 RADIUSを使用した認証

Remote Authentication Dial-In User Service (RADIUS)はユーザー認証、認可、およびアカウンティングに使用される標準のライトウェイト・プロトコルです。

RADIUSにより、ユーザーはRSA One-Time Password Specifications (OTPS)を使用してOracleデータベースに対する認証もできるようになります。

関連項目:

- RADIUSの構成の詳細は、「RADIUS認証の構成」を参照してください。
- OTPSに関するRSAのドキュメント

親トピック: サード・パーティ・サービスを使用した認証

3.7.2.4 ディレクトリベース・サービスを使用した認証

中核となるディレクトリを使用すると、認証とその管理が効率的になります。

次のようなディレクトリベース・サービスがあります。

- Lightweight Directory Access Protocol (LDAP)を使用するOracle Internet Directoryにより、中央リポジトリを使用してユーザー(エンタープライズ・ユーザーと呼ばれます)に関する情報を格納および管理できます。エンタープライズ・ユーザーのアカウントは、分散環境で作成されます。データベース・ユーザーの場合は、アクセスするデータベースごとにパスワードとともに作成する必要がありますが、エンタープライズ・ユーザーの情報にはOracle Internet Directoryで集中的にアクセスできます。このディレクトリをMicrosoft Active DirectoryやSunOneと統合することもできます。
- Oracle Enterprise Security Managerを使用すると、Oracle Internet Directoryからのロールの取得と保管ができ、これにより権限の集中管理が可能になることで管理が容易になり、セキュリティ・レベルが向上します。

親トピック: サード・パーティ・サービスを使用した認証

3.7.2.5 公開キー・インフラストラクチャを使用した認証

公開キー・インフラストラクチャ(PKI)に基づく認証システムでは、ユーザー・クライアントに電子証明書が発行されます。

これらのクライアントはこの証明書を使用して直接企業内のサーバーに身分を証明し、認証に直接的に関与しません。Oracle Databaseで提供されている、公開キーと証明書を使用するためのPKIは、次のコンポーネントで構成されています。

- SSLによる認証および保護セッション・キー管理。詳細は、Transport Layer Securityを使用した認証を参照してください。
- 信頼できる証明書識別情報を検証するときに、ユーザー証明書の署名者として信頼するサード・パーティ・エンティティを識別するために使用します。ユーザーの証明書が確認されるとき、署名者は、検証システムに格納されている認証

局のトラスト・ポイントまたは信頼できる証明連鎖を使用してチェックされます。この連鎖内に複数レベルの信頼できる証明書がある場合は、下位レベルの証明書を信頼するため、それより上のレベルの証明書をすべて再検証する必要はありません。

- Oracle Wallet Manager。Oracleウォレットは、ユーザーの秘密キー、ユーザー証明書および一連のトラスト・ポイント(信頼できる認証局)を含むデータ構造です。Oracleウォレットの管理の詳細は、『Oracle Databaseエンタープライズ・ユーザー・セキュリティ管理者ガイド』を参照してください。

  Oracle Wallet Managerを使用してOracleウォレットを管理できます。これは、Oracleウォレットのセキュリティ資格証明を管理および編集するために使用するスタンドアロンのJavaアプリケーションです。次の操作を実行します。

  - 公開キーと秘密キーのペアを生成し、認証局に提出する証明書要求を作成して、ウォレットを作成します。
  - エンティティの証明書をインストールします。
  - Oracle Databaseのクライアントとサーバー上でX.509v3証明書を管理します。
  - エンティティの信頼できる証明書を構成します。
  - ウォレットをオープンして、PKIベースのサービスにアクセスできるようにします。

- 信頼できるエンティティ、認証局から取得された(署名された) X.509バージョン3証明書。認証局は信頼されているため、これらの証明は要求側エンティティの情報が正確であることと証明書上の公開キーが認定されるエンティティに属していることを証明します。証明書はOracleウォレットにロードされるため、今後の認証が可能になります。

親トピック: サード・パーティ・サービスを使用した認証

3.8 PDBのオペレーティング・システム・ユーザーの構成

DBMS_CREDENTIAL.CREATE_CREDENTIALプロシージャで、ユーザー・アカウントをPDBのオペレーティング・システム・ユーザーに設定します。

- PDBのオペレーティング・システム・ユーザーの構成について
  oracleオペレーティング・システム・ユーザーのかわりに、特定のユーザー・アカウントをそのPDBのオペレーティング・システム・ユーザーに設定できます。
- PDBのオペレーティング・システム・ユーザーの構成
  DBMS_CREDENTIAL.CREATE_CREDENTIALプロシージャで、PDBのオペレーティング・システム・ユーザーを設定できます。

親トピック: 認証の構成

3.8.1 PDBのオペレーティング・システム・ユーザーの構成について

oracleオペレーティング・システム・ユーザーのかわりに、特定のユーザー・アカウントをそのPDBのオペレーティング・システム・ユーザーに設定できます。

特定のユーザーをPDBのオペレーティング・システム・ユーザーとして設定しない場合、PDBではデフォルトでoracleオペレーティング・システム・ユーザーが使用されます。ルートについては、オペレーティング・システムと対話する必要がある場合、oracleオペレーティング・システム・ユーザーを使用できます。

セキュリティ向上のため、マルチテナント環境の各PDBに対して一意のオペレーティング・システム・ユーザーを設定することをお薦めします。そうすることで、oracleオペレーティング・システム・ユーザーより権限の低いユーザーとしてオペレーティング・システムとの対話を行えるほか、PDBに属するデータを、他のPDBに接続しているユーザーのアクセスから保護することにも役立ちます。

親トピック: PDBのオペレーティング・システム・ユーザーの構成

3.8.2 PDBのオペレーティング・システム・ユーザーの構成

DBMS_CREDENTIAL.CREATE_CREDENTIALプロシージャで、PDBのオペレーティング・システム・ユーザーを設定できます。

1. DBMS_CREDENTIAL PL/SQLパッケージに対するEXECUTE権限およびALTER SYSTEMシステム権限を持つユーザーとして、データベース・インスタンスのルートにログインします。
   たとえば:

```
sqlplus c##sec_admin
Enter password: password
```

2. DBMS_CREDENTIAL.CREATE_CREDENTIALプロシージャを実行して、オペレーティング・システム・ユーザーのOracle資格証明を作成します。
   たとえば、os_adminという名前のユーザーの資格証明を設定するには、次のようにします。

```
BEGIN
 DBMS_CREDENTIAL.CREATE_CREDENTIAL (
    credential_name => 'PDB1_OS_USER',
    username        => 'os_admin',
    password        => 'password');
END;
/
```

3. オペレーティング・システム・ユーザーが使用されるPDBに接続します。
   たとえば:

```
CONNECT cc##sec_admin@hrpdb
Enter password: password
```

   使用可能なPDBを検索するには、show pdbsコマンドを実行します。現在のPDBを確認するには、show con_nameコマンドを実行します。

4. ステップ2で資格証明を設定したユーザーに対して、PDB_OS_CREDENTIAL初期化パラメータを設定します。
   たとえば:

```
ALTER SYSTEM SET PDB_OS_CREDENTIAL = PDB1_OS_USER SCOPE = SPFILE;
```

   PDB_OS_CREDENTIALパラメータは静的なパラメータであるため、SCOPE = SPFILE句を使用して設定する必要があります。

5. データベース・インスタンスを再起動します。

```
SHUTDOWN IMMEDIATE
STARTUP
```

関連トピック

- パスワードの最低要件

親トピック: PDBのオペレーティング・システム・ユーザーの構成

3.9 ユーザーのグローバル認証とグローバル認可

ユーザーのグローバル認証とグローバル認可を使用すると、ユーザー関連情報を一元管理できます。

- グローバルなユーザー認証と認可の構成について
  LDAPベースのディレクトリ・サービスで、認可も含めたユーザー関連情報を集中管理します。
- ディレクトリ・サービスで認可されるユーザーの構成
  ディレクトリ・サービスで認可されるようにグローバル・ユーザーまたは複数のエンタープライズ・ユーザーを構成できます。
- グローバル認証とグローバル認可の利点
  ユーザーのグローバル認証とグローバル認可には、複数のメリットがあります。

親トピック: 認証の構成

3.9.1 グローバルなユーザー認証と認可の構成について

LDAPベースのディレクトリ・サービスで、認可も含めたユーザー関連情報を集中管理します。

これにより、ユーザーおよび管理者はデータベース内でグローバル・ユーザーとして識別されます。これは、そのユーザーがTLSによって認証され、ユーザーの管理がデータベースの外部で集中化されたディレクトリ・サービスによって行われることを意味します。グローバル・ロールはデータベース内で定義され、そのデータベースに対してのみ認識されますが、グローバル・ロールに対する認可はディレクトリ・サービスによって行われます。

ノート:

ユーザーは Transport Layer Security (TLS)で認証されるユーザーであっても構いません。この場合、認可はディレクトリで管理されておらず、これらのユーザーが持っているのはローカル・データベース・ロールのみです。

このような集中管理によって、エンタープライズ・ユーザーとエンタープライズ・ロールの作成が可能になります。エンタープライズ・ユーザーの定義と管理は、ディレクトリ内で行います。エンタープライズ・ユーザーには企業内で一意の識別情報があり、複数データベースにまたがるアクセス権限を決定するエンタープライズ・ロールを割り当てることができます。エンタープライズ・ロールは1つ以上のグローバル・ロールで構成されているため、グローバル・ロールのコンテナとみなすことができます。

また、集中管理されたユーザーを使用して、Microsoft Active Directoryなどのディレクトリ・サービスを介してユーザーを認証および認可することもできます。

関連トピック

- Transport Layer Security認証の構成
- 管理者の厳密認証と集中管理
- Microsoft Active Directoryによる集中管理ユーザーの構成

親トピック: ユーザーのグローバル認証とグローバル認可

3.9.2 ディレクトリ・サービスで認可されるユーザーの構成

ディレクトリ・サービスで認可されるようにグローバル・ユーザーまたは複数のエンタープライズ・ユーザーを構成できます。

- プライベート・スキーマを持つグローバル・ユーザーの作成
  プライベート・スキーマを持つユーザー・アカウントを作成するには、エンタープライズ・ディレクトリにとって有意義な識別子(識別名またはDN)を指定します。
- スキーマを共有する複数のエンタープライズ・ユーザーの作成
  複数のエンタープライズ・ユーザーがデータベース内の1つのスキーマを共有する可能性があります。

親トピック: ユーザーのグローバル認証とグローバル認可

3.9.2.1 プライベート・スキーマを持つグローバル・ユーザーの作成

プライベート・スキーマを持つユーザー・アカウントを作成するには、エンタープライズ・ディレクトリにとって有意義な識別子(識別名またはDN)を指定します。

ただし、ユーザーによるアクセスが必要なすべてのデータベースとディレクトリにこのユーザーを作成する必要があることに注意してください。

- プライベート・スキーマを持つグローバル・ユーザーを作成するには、`CREATE USER ... IDENTIFIED GLOBALLY` SQL文を使用します。

  標準のLDAPデータ交換形式(LDIF)フィールドを含めることができます。たとえば、SSLによって認証され、エンタープライズ・ディレクトリ・サービスによって認可される、プライベート・スキーマを持つグローバル・ユーザー(`psmith_gl`)を作成するには、次のようにします。

  ```
  CREATE USER psmith_gl IDENTIFIED GLOBALLY AS
  'CN=psmith,OU=division1,O=example,C=US';
  ```

  詳細は、次のとおりです。

  - `CN`は、このユーザーの共通名`psmith_gl`を指します。
  - `OU`は、ユーザーの組織単位`division1`を指します。
  - `O`は、ユーザーの組織`Example`を指します。
  - `C`は、組織Exampleが存在する国USを指します。

親トピック: ディレクトリ・サービスで認可されるユーザーの構成

3.9.2.2 スキーマを共有する複数のエンタープライズ・ユーザーの作成

複数のエンタープライズ・ユーザーがデータベース内の1つのスキーマを共有できます

これらのユーザーは、エンタープライズ・ディレクトリ・サービスによって認可されますが、データベース内に個々のプライベート・スキーマを持ちません。また、ユーザーはデータベース内に個別に作成されません。ユーザーは、データベース内の共有スキーマに接続します。

1. 次の例を使用して、データベースに共有スキーマを作成します。

   ```
   CREATE USER appschema IDENTIFIED GLOBALLY AS '';
   ```

2. ディレクトリに、複数のエンタープライズ・ユーザーとマッピング・オブジェクトを作成します。

   このマッピング・オブジェクトは、ユーザーのDNを共有スキーマにマップする方法をデータベースに伝えます。完全な識別名(DN)マッピング(一意のDN 1つに対して1つのディレクトリ・エントリが対応する)を作成するか、または、ユーザーごとに複数のDNコンポーネントを1つのスキーマにマップできます。たとえば:

   ```
   OU=division1,O=Example,C=US
   ```

   関連項目:

   これらのマッピングの詳細は、『Oracle Databaseエンタープライズ・ユーザー・セキュリティ管理者ガイド』を参照してください。

ほとんどのユーザーは専用スキーマを必要としないため、スキーマに依存しないユーザーを実装することで、ユーザーをデータベー

スから切り離すことができます。データベース内で同じスキーマを共有する複数のユーザーを作成すると、各ユーザーは他のデータベース内の共有スキーマにもエンタープライズ・ユーザーとしてアクセスできます。

親トピック: ディレクトリ・サービスで認可されるユーザーの構成

3.9.3 グローバル認証とグローバル認可の利点

ユーザーのグローバル認証とグローバル認可には、複数のメリットがあります。

- SSL、KerberosまたはWindowsネイティブ認証を使用して、厳密な認証が行われます。
- ユーザーと権限を全社規模で集中管理できます。
- 管理が容易です。ユーザーごとに、社内の各データベースにスキーマを作成する必要がありません。
- シングル・サインオンが容易になります。ユーザーは1回のサインオンのみで複数のデータベースおよびサービスにアクセスできます。さらに、パスワードを使用しているユーザーは、パスワード認証されたエンタープライズ・ユーザーを受け入れる複数データベースにアクセスするための単一パスワードを持つことができます。
- グローバルなユーザー認証と認可はパスワード・ベースのアクセスを提供するため、以前に定義されたパスワード認証方式のデータベース・ユーザーを、集中管理されているディレクトリに(ユーザー移行ユーティリティを使用して)移行できます。これによって、以前のリリースのOracle Databaseクライアントで使用可能だったグローバル認証と認可が引き続きサポートされます。
- CURRENT_USERデータベース・リンクはグローバル・ユーザーとして接続します。ローカル・ユーザーはストアド・プロシージャとの関連においてグローバル・ユーザーとして、グローバル・ユーザー・パスワードをリンク定義に保管することなく、接続できます。

関連トピック

- Oracle Databaseエンタープライズ・ユーザー・セキュリティ管理者ガイド

親トピック: ユーザーのグローバル認証とグローバル認可

3.10 ユーザーとパスワード認証のための外部サービスの構成

外部サービス(オペレーティング・システムまたはネットワーク)でパスワードを管理し、ユーザーを認証できます。

- 外部認証について
  外部認証を使用する場合、ユーザー・アカウントはOracle Databaseでメンテナンスされますが、パスワード管理とユーザー認証は外部サービスによって実行されます。
- 外部認証の利点
  外部認証には、複数のメリットがあります。
- 外部認証の有効化
  外部認証を使用可能にするには、初期化パラメータOS_AUTHENT_PREFIXを設定し、この接頭辞をOracle Databaseユーザー名で使用します。
- 外部認証されるユーザーの作成
  外部認証されるユーザーは、オペレーティング・システムやネットワーク・サービスで認証されます。
- オペレーティング・システムを使用したユーザー・ログインの認証
  Oracle Databaseで許可されるオペレーティング・システム認証ログインは、保護された接続のみを介したログインであるため、Oracle Netおよび共有サーバー構成を使用したログインは含まれません。
- ネットワーク認証を使用したユーザー・ログインの認証

Oracle厳密認証が実行するネットワーク認証は、Kerberosなどのサード・パーティ・サービスを使用するよう構成できます。

親トピック: 認証の構成

3.10.1 外部認証について

外部認証を使用する場合、ユーザー・アカウントはOracle Databaseでメンテナンスされますが、パスワード管理とユーザー認証は外部サービスによって実行されます。

この外部サービスは、オペレーティング・システムでもOracle Netのようなネットワーク・サービスでもかまいません。パスワード・ファイルを使用してユーザーを認証する場合、SYSDBA、SYSOPER、SYSASM、SYSBACKUP、SYSDGおよびSYSKM管理権限を付与されたユーザーに対して外部認証を構成できます。

外部認証の場合、データベースはデータベース・アカウントへのアクセス制限を、その基礎となるオペレーティング・システムまたはネットワーク認証サービスに依存します。データベース・パスワードは、このタイプのログインには使用されません。オペレーティング・システムまたはネットワーク・サービスで許可されている場合は、それにより、ユーザーがデータベースにログインする前にユーザーを認証できます。

また、集中管理されたユーザーを使用して、Microsoft Active Directoryなどのディレクトリ・サービスを介してユーザーを認証および認可することもできます。

関連トピック

- 管理ユーザーのパスワード・ファイルの管理
- Microsoft Active Directoryによる集中管理ユーザーの構成

親トピック: ユーザーとパスワード認証のための外部サービスの構成

3.10.2 外部認証の利点

外部認証には、複数のメリットがあります。

これらのメリットは次のとおりです。

- スマートカード、指紋、Kerberos、オペレーティング・システムなど、使用可能な認証メカニズムの選択肢が増えます。
- Kerberosなどのネットワーク認証サービスの多くがシングル・サインオンをサポートしているため、ユーザーは多数のパスワードを記憶する必要がありません。
- 前述の外部認証メカニズムのいずれかをすでに使用している場合は、そのメカニズムをデータベースで使用することで、管理費用を節減できます。

親トピック: ユーザーとパスワード認証のための外部サービスの構成

3.10.3 外部認証の有効化

外部認証を使用可能にするには、初期化パラメータOS_AUTHENT_PREFIXを設定し、この接頭辞をOracle Databaseユーザー名で使用します。

このOS_AUTHENT_PREFIXパラメータは、Oracle Databaseで全ユーザーのオペレーティング・システム・アカウント名の先頭に追加する接頭辞を定義します。Oracle Databaseは、ユーザーが接続しようとすると、接頭辞付きのユーザー名をデータベース内のOracle Databaseユーザー名と比較します。

1. OS_AUTHENT_PREFIXをNULL文字列(空の二重引用符""で指定)に設定します。NULL文字列を使用すると、オペレーティング・システム・アカウント名に接頭辞は追加されないため、Oracle Databaseユーザー名とオペレーティング・システム・ユーザー名は完全に一致します。

   たとえば:

   ```
   OS_AUTHENT_PREFIX=" "
   ```

2. OS_AUTHENT_PREFIXはデータベースの存続期間中は必ず同じままにします。接頭辞を変更した場合、古い接頭辞を含むデータベース・ユーザー名は、パスワード認証を使用するように変更しないかぎり、接続に使用できません。

OS_AUTHENT_PREFIXパラメータのデフォルト値はOPS$であり、これによって古いバージョンのOracle Databaseとの下位互換性を維持しています。たとえば、OS_AUTHENT_PREFIXを次のように設定する場合を想定します。

```
OS_AUTHENT_PREFIX=OPS$
```

オペレーティング・システム・アカウント名tsmithを持つユーザーが、Oracleデータベース・インストールに接続する際にオペレーティング・システムによって認証された場合、Oracle Databaseは対応するデータベース・ユーザーOPS$tsmithの存在をチェックし、存在している場合はこのユーザーを接続できます。オペレーティング・システムによって認証されたユーザーへの参照には、OPS$tsmithのように、必ず接頭辞OPS$が含まれる必要があります。

ノート:

OS_AUTHENT_PREFIX 初期化パラメータに指定する文字列は、オペレーティング・システムによって大/小文字が区別される場合があります。この初期化パラメータの詳細は、使用しているオペレーティング・システム固有のOracle Database マニュアルを参照してください。

親トピック: ユーザーとパスワード認証のための外部サービスの構成

3.10.4 外部認証されるユーザーの作成

外部認証されるユーザーは、オペレーティング・システムやネットワーク・サービスで認証されます。

外部認証されるユーザーを作成できます。Oracle Databaseがこの外部ログイン認証に依存するのは、特定のユーザーのデータベース・リソースへのアクセス権を特定のオペレーティング・システム・ユーザーに付与する場合です。

- CREATE USER文のIDENTIFIED EXTERNALLY句を使用して、外部認証されるユーザーを作成します。

次の例では、Oracle Databaseによって識別され、オペレーティング・システムまたはネットワーク・サービスによって認証されるユーザーを作成します。この例では、OS_AUTHENT_PREFIXパラメータは空白(" ")に設定されていると想定しています。

```
CREATE USER psmith IDENTIFIED EXTERNALLY;
```

親トピック: ユーザーとパスワード認証のための外部サービスの構成

3.10.5 オペレーティング・システムを使用したユーザー・ログインの認証

Oracle Databaseで許可されるオペレーティング・システム認証ログインは、保護された接続のみを介したログインであるため、Oracle Netおよび共有サーバー構成を使用したログインは含まれません。

このタイプのオペレーティング・システム認証がデフォルトです。この制限によって、リモート・ユーザーが、ネットワーク接続を介して別のオペレーティング・システムのユーザーになりすますことを防止します。

データベース初期化パラメータ・ファイルでREMOTE_OS_AUTHENTパラメータをTRUEに設定すると、データベースは保護されていない接続を介して受け取ったクライアント・オペレーティング・システム・ユーザー名を受け入れてアカウント・アクセスに使用します。一般にPCなどのクライアントは、オペレーティング・システムの認証を適切に実行していない場合があるため、この機能を有効にするとセキュリティが非常に低下します。

デフォルトの設定REMOTE_OS_AUTHENT = FALSEを使用すると、安全性の高い構成となり、Oracleデータベースに接続するクライアントがサーバーベースで適切に認証されます。

REMOTE_OS_AUTHENTパラメータは、Oracle Database 11g リリース1(11.1)では非推奨となっており、下位互換性のためにのみ保持されている点に注意してください。

このパラメータに対する変更は、次回インスタンスを起動して、データベースをマウントしたときに有効となります。一般的に、ホスト・オペレーティング・システムを介したユーザー認証では、個別のデータベース・ユーザー名やパスワードを指定せずに、Oracle Databaseに迅速かつ簡便に接続できます。ユーザー・エントリも、データベースとオペレーティング・システムの各監査証跡で互いに対応します。

親トピック: ユーザーとパスワード認証のための外部サービスの構成

3.10.6 ネットワーク認証を使用したユーザー・ログインの認証

Oracle厳密認証が実行するネットワーク認証は、Kerberosなどのサード・パーティ・サービスを使用するよう構成できます。

Oracle厳密認証を唯一の外部認証サービスとして使用している場合、Oracle厳密認証で可能になるのは保護された接続のみであるため、REMOTE_OS_AUTHENTパラメータの設定は無意味になります。

親トピック: ユーザーとパスワード認証のための外部サービスの構成

3.11 複数層の認証と認可

中間層アプリケーションを保護するために、Oracle Databaseは権限を制限し、すべての層のクライアントの識別情報を保持し、クライアントによるアクションを監査します。

トランザクション処理モニターのようにタスクの非常に多い中間層を使用するアプリケーションでは、中間層に接続しているクライアントの識別情報が保持される必要があります。中間層を使用することの1つの利点が接続プーリングであり、これにより複数のユーザーは、それぞれが個別の接続を必要とせずに、データベース・サーバーにアクセスできるようになります。このような環境では、接続を非常に迅速に設定および停止できる必要があります。

この種の環境では、Oracle Call Interfaceを使用して、各ユーザーのデータベース・パスワード認証を可能にする軽量セッションを作成できます。この方法によって、中間層を介して実際のユーザーの識別性が保たれるため、各ユーザーの個別のデータベース接続によるオーバーヘッドは生じません。

パスワードあり、またはパスワードなしで軽量セッションを作成できます。ただし、中間層がファイアウォールの外部またはファイアウォールにある場合は、軽量セッションごとに専用パスワードを設定する方がセキュリティが向上します。内部アプリケーション・サーバーの場合は、パスワードなしの軽量セッションの方が適している場合があります。

親トピック: 認証の構成

3.12 クライアント、アプリケーション・サーバーおよびデータベース・サーバーの管理とセキュリティ

複数層環境では、アプリケーション・サーバーはクライアントにデータを提供し、1つ以上のデータベース・サーバーとのインタフェース

として機能します。

アプリケーション・サーバーではWebブラウザなどのクライアントの資格証明を検証でき、データベース・サーバーではアプリケーション・サーバーで実行される操作を監査できます。監査対象の操作には、クライアントで表示する情報の要求など、クライアントのためにアプリケーション・サーバーが実行する操作が含まれます。特定のクライアントに関連しないアプリケーション・サーバー操作の例には、データベース・サーバーへの接続要求があります。

複数層環境における認証は、トラスト領域に基づいています。クライアント認証は、アプリケーション・サーバーのドメインで実行されます。アプリケーション・サーバー自身は、データベース・サーバーによって認証されます。次の操作が行われます。

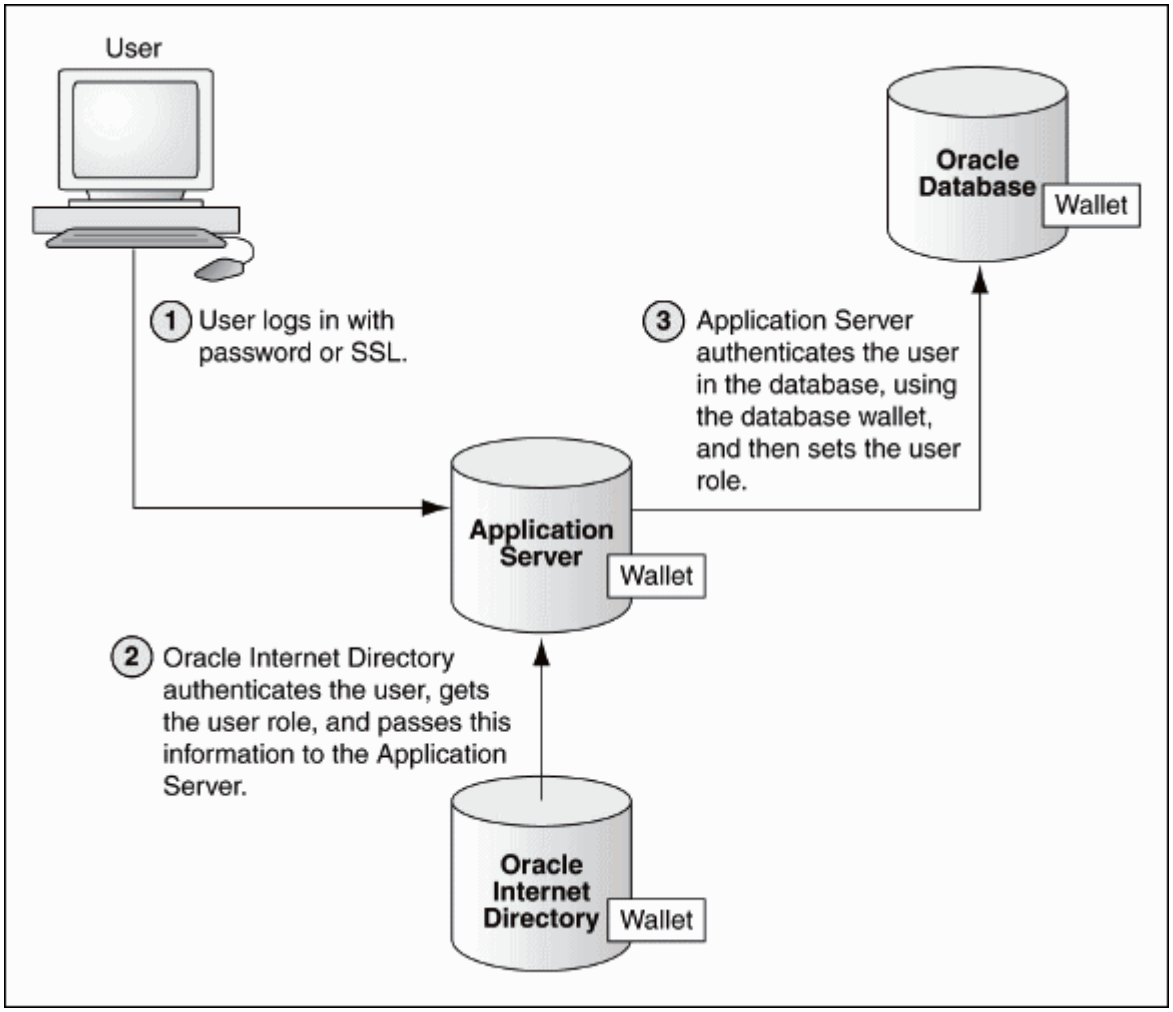
- エンド・ユーザーは通常、パスワードまたはX.509証明書を使用して、アプリケーション・サーバーに認証の証明を提供します。
- アプリケーション・サーバーは、エンド・ユーザーを認証してから、それ自体をデータベース・サーバーに対して認証します。
- データベース・サーバーは、アプリケーション・サーバーを認証し、エンド・ユーザーの存在を検証して、そのエンド・ユーザーへの接続権限がアプリケーション・サーバーにあることを検証します。

アプリケーション・サーバーでは、かわりに接続するエンド・ユーザーのロールを使用可能にすることもできます。アプリケーション・サーバーは、これらのロールを認証リポジトリとして機能するディレクトリから取得できます。アプリケーション・サーバーが要求できるのは、これらのロールを使用可能にすることのみです。データベースは、次の要件を検証します。

- クライアント内部のロール・リポジトリをチェックして、そのクライアントにこれらのロールがあることを検証します。
- アプリケーション・サーバーに、ユーザーのために接続し、これらのロールをユーザーのように使用できる権限があることを検証します。

次の図は、複数層認証の例を示しています。

図3-3 複数層認証



次のアクションが実行されます。

1. ユーザーは、パスワードまたはTransport Layer Securityを使用してログインします。認証情報はOracle Application Serverを介して渡されます。
2. Oracle Internet Directoryはユーザーを認証し、そのユーザーに対応付けられたロールをウォレットから取得して、この情報をOracle Application Serverに戻します。
3. Oracle Application Serverは、ユーザーの識別情報が格納されているウォレットを含むOracle Databaseでこの情報をチェックし、そのユーザーのロールを設定します。

中間層アプリケーションのセキュリティでは、次のような重要な問題に対処する必要があります。

- アカウンタビリティ。データベース・サーバーは、アプリケーションのアクションとアプリケーションがクライアントのかわりに行うアクションを識別できる必要があります。この2種類のアクションを監査できる必要があります。
- 最低限の権限。不慮または不正による無許可のアクティビティの危険性を排除するため、ユーザーと中間層には、それぞれのアクションを実行するために必要最小限の権限を与える必要があります。

親トピック: 認証の構成

3.13 複数層環境でのユーザー識別情報の保持

中間層サーバーを使用してプロキシ認証を行ったり、クライアント識別子を使用してデータベースが認識しないアプリケーション・ユーザーを識別したりできます。

- プロキシ認証に対する中間層サーバーの使用

Oracle Call Interface (OCI)、JDBC/OCIまたはJDBCシン・ドライバでは、データベース・ユーザーまたはエンタープライズ・ユーザーのプロキシ認証に対する中間層がサポートされます。

- データベースに認識されないアプリケーション・ユーザーの識別でのクライアント識別子の使用
  クライアント識別子は、中間層システムでユーザー識別情報を保持します。また、グローバル・アプリケーション・コンテキストとは独立してこれらを使用することもできます。

親トピック: 認証の構成

3.13.1 プロキシ認証に対する中間層サーバーの使用

Oracle Call Interface (OCI)、JDBC/OCIまたはJDBCシン・ドライバでは、データベース・ユーザーまたはエンタープライズ・ユーザーのプロキシ認証に対する中間層がサポートされます。

- プロキシ認証について
  Oracle Databaseでは、Oracle Call Interface(OCI)、JDBC/OCIまたはJDBCシン・ドライバによって、データベース・ユーザーまたはエンタープライズ・ユーザーにプロキシ認証を提供します。
- プロキシ認証の利点
  複数層環境では、プロキシ認証を使用すると、中間層アプリケーションのすべての層を通じてクライアント識別情報と権限が保持され、クライアントのアクションが監査されます。
- プロキシ・ユーザー・アカウントの作成者とは
  プロキシ・ユーザー・アカウントを作成するためには、ユーザーには特別な権限が必要です。
- プロキシ・ユーザー・アカウントの作成のガイドライン
  プロキシ・ユーザー・アカウントを作成する際の特別なガイドラインがあります。
- プロキシ・ユーザー・アカウントの作成と、作成したプロキシ・ユーザー・アカウントを介したユーザー接続の認可
  CREATE USER文およびALTER USER文を使用して、プロキシ・ユーザーを作成し、このユーザーを介して接続するユーザーを認可できます。
- プロキシ・ユーザー・アカウントと、そのアカウントを介して接続するユーザーの認可
  CREATE USER文を使用すると、次のタイプのユーザー・アカウントを作成でき、これらのすべてがプロキシ・アカウントとして使用できます。
- 安全性の高い外部パスワード・ストアとプロキシ認証の使用
  プロキシ認証に使用しているパスワードが不正なユーザーによって取得される懸念がある場合は、安全性の高い外部パスワード・ストアを使用します。
- プロキシ認証を使用した実際のユーザーの識別情報の引渡し
  エンタープライズ・ユーザーまたはデータベース・ユーザーにOracle Call Interface、JDBC/OCIまたはシン・ドライバを使用できます。
- 中間層の権限の制限
  「最低限の権限」とは、ユーザーの権限は各自の職務を行うのに必要な最小限の権限までにする必要があるという原則です。
- ユーザーのプロキシとして機能し、ユーザーを認証する中間層を認可する方法
  ユーザーとして接続する中間層サーバーを認可できます。
- 他の方式で認証されたユーザーのプロキシとして機能するために、中間層を認可する方法
  他の方式で認証されたユーザーのプロキシとして機能するための中間層を認可できます。
- 中間層を介したデータベースへのユーザーの再認証
  ALTER USER SQL文でAUTHENTICATION REQUIREDプロキシ句を使用して、認証が必要であることを指定できます。

- パスワードベースのプロキシ認証の使用
  パスワードベースのプロキシ認証を使用すると、Oracle Databaseはクライアントのパスワードを中間層サーバーに渡します。
- エンタープライズ・ユーザーでのプロキシ認証の使用
  プロキシ認証に対する中間層の応答方法は、ユーザーがエンタープライズ・ユーザーとして認証されているか、パスワード認証ユーザーとして認証されているかによって異なります。

親トピック: 複数層環境でのユーザー識別情報の保持

3.13.1.1 プロキシ認証について

Oracle Databaseでは、Oracle Call Interface(OCI)、JDBC/OCIまたはJDBCシン・ドライバによって、データベース・ユーザーまたはエンタープライズ・ユーザーにプロキシ認証を提供します。

エンタープライズ・ユーザーは、Oracle Internet Directoryで管理されるユーザーで、データベースの共有スキーマにアクセスします。

次の3つの形式のプロキシ認証を使用して、クライアントを認証する中間層サーバーを安全な方法で設計できます。

- 中間層サーバーは、データベース・サーバーを使用してそれ自体を認証し、クライアント(この場合はアプリケーション・ユーザーまたは別のアプリケーション)は、この中間層サーバーを使用してそれ自体を認証します。クライアントの識別情報は、データベースに到達するまで確実に保持されます。
- クライアント(この場合はデータベース・ユーザー)は、中間層サーバーによって認証されません。クライアントの識別情報とデータベース・パスワードは、中間層サーバーを経由してデータベース・サーバーに渡され、そこで認証されます。
- クライアント(この場合はグローバル・ユーザー)は中間層サーバーによって認証され、中間層を介して次のいずれかを渡します。クライアントのユーザー名はそこから取得されます。
  - 識別名(DN)
  - 証明書

いずれの場合でも、中間層サーバーにクライアントの代理としての機能を与えるために、管理者は中間層サーバーを認可する必要があります。

関連トピック

- 複数層環境におけるSQL文および権限の監査
- 『Oracle Database JDBC開発者ガイド』

親トピック: プロキシ認証に対する中間層サーバーの使用

3.13.1.2 プロキシ認証の利点

複数層環境では、プロキシ認証を使用すると、中間層アプリケーションのすべての層を通じてクライアント識別情報と権限が保持され、クライアントのアクションが監査されます。

たとえば、この機能によって、Webアプリケーション(プロキシとして機能する)を使用するユーザーの識別情報を、アプリケーションを介してデータベース・サーバーに渡すことができます。

3層システムは、組織にとって次のようなメリットがあります。

- 組織は、アプリケーション・ロジックをアプリケーション・サーバーに、データ記憶域をデータベースにパーティション化することによって、アプリケーション・ロジックとデータ記憶域を分離できます。

- アプリケーション・サーバーおよびWebサーバーを使用して、データベースに格納されているデータにアクセスできます。
- ユーザーは、操作が簡単で使い慣れたブラウザ・インタフェースを使用できます。
- 組織では、多数のシック・クライアントを多数のシン・クライアントと1つのアプリケーション・サーバーに置き換えることによって、コンピューティング・コストを低く抑えることもできます。

さらに、Oracle Databaseのプロキシ認証には、次のセキュリティ上のメリットがあります。

- 中間層がかわりに接続できるユーザー、および中間層がユーザーに対して想定できるロールを制御することによって制限付きトラスト・モデルが実現します。
- OCI、JDBC/OCIまたはJDBCシン・ドライバでユーザー・セッションをサポートし、クライアント再認証のためのオーバーヘッドを排除することによってスケーラビリティが得られます。
- 実際のユーザーの識別情報をデータベースに到達するまで保持し、実際のユーザーのかわりに行われるアクションの監査を可能にすることによって、アカウンタビリティが得られます。
- ユーザーがデータベースに認識されている環境と、ユーザーが単なるアプリケーション・ユーザーでデータベースには認識されていない環境の両方をサポートすることによって柔軟性が得られます。

ノート:

Oracle Database はこのプロキシ認証機能を 3 つの層のみでサポートしています。複数の中間層を横断してのサポートはありません。

親トピック: プロキシ認証に対する中間層サーバーの使用

3.13.1.3 プロキシ・ユーザー・アカウントの作成者とは

プロキシ・ユーザー・アカウントを作成するためには、ユーザーには特別な権限が必要です。

これらの権限は次のとおりです。

- プロキシ・ユーザー・アカウントとして使用されるデータベース・ユーザー・アカウントを作成するための`CREATE USER`システム権限
- Oracle Database Vaultが有効な場合にプロキシ・ユーザー・アカウントを作成するための`DV_ACCTMGR`ロール
- `CREATE SESSION`システム権限をプロキシ・ユーザー・アカウントに付与できること
- 既存のユーザー・アカウントがプロキシ・アカウントを介してデータベースに接続できるようにするための`ALTER USER`システム権限

親トピック: プロキシ認証に対する中間層サーバーの使用

3.13.1.4 プロキシ・ユーザー・アカウントの作成のガイドライン

プロキシ・ユーザー・アカウントを作成する際の特別なガイドラインがあります。

- セキュリティを高めて最小限の権限の原則を守るために、プロキシ・ユーザー・アカウントに`CREATE SESSION`権限のみを付与します。このユーザーに他の一切の権限を付与しないでください。プロキシ・ユーザー・アカウントは、他のユーザーがプロキシ・アカウントを使用して接続できるようにする場合にのみ使用されます。接続中に実施されるすべての権限は、プロキシ・アカウントではなく、接続しているユーザーに属する必要があります。

- すべてのパスワードと同様、プロキシ・ユーザーに作成するパスワードも強力であり容易に推測されないものにしてください。複数のユーザーがプロキシ・ユーザーとして接続することになるため、このパスワードを強力にすることが特に重要であることを忘れないでください。
- Oracle厳密認証ネットワーク接続機能を使用してネットワーク傍受を防ぐことを検討してください。
- 接続ユーザーが持っている制御の量をさらに微調整する場合は、接続ユーザーがプロキシ・アカウントを介して接続しているときに使用するロールを制限することを検討してください。ALTER USER文のWITH ROLE句を使用すると、指定されたロールまたは指定されたロール以外のロールを使用して接続するユーザー、あるいはロールをまったく使用せずに接続するユーザーを構成できます。プロキシ・ユーザーは、WITH ROLE句に含まれているロールのみアクティブにできることに注意してください。プロキシ・ユーザー・セッションには、クライアント(つまり現在の)ユーザーに直接付与されたすべての権限が付与されます。
- プロキシ・セッションのプロキシ・ユーザーは、WITH ROLEまたはWITH ROLE ALL句でロールを有効にできる場合にのみ、パスワード保護されたロールまたはセキュア・アプリケーション・ロールを有効にできます。(この句を指定しない場合、WITH ROLE ALLがデフォルトになります。)WITH ROLEでセキュア・ロールを指定しない場合、正しいパスワードによってもこれらのロールを有効にできません。

関連トピック

- パスワードの保護に関するガイドライン

親トピック: プロキシ認証に対する中間層サーバーの使用

3.13.1.5 プロキシ・ユーザー・アカウントの作成と、作成したプロキシ・ユーザー・アカウントを介したユーザー接続の認可

CREATE USER文およびALTER USER文を使用して、プロキシ・ユーザーを作成し、このユーザーを介して接続するユーザーを認可できます。

プロキシ・セッションのプロキシ・ユーザーは、WITH ROLEまたはWITH ROLE ALL句でロールを有効にできる場合にのみ、パスワード保護されたロールまたはセキュア・アプリケーション・ロールを有効にできます。(この句を指定しない場合、WITH ROLE ALLがデフォルトになります。)WITH ROLEでセキュア・ロールを指定しない場合、正しいパスワードによってもこれらのロールを有効にできません。

1. CREATE USER文を使用してプロキシ・ユーザー・アカウントを作成します。

   たとえば:

   ```
   CREATE USER appuser IDENTIFIED BY password;
   ```

2. ALTER USER文のGRANT CONNECT THROUGH句を使用して、既存のユーザーがプロキシ・ユーザー・アカウントを介して接続できるようにします。

   たとえば:

   ```
   ALTER USER preston GRANT CONNECT THROUGH appuser;
   ```

   ユーザー名とプロキシの組合せは、250文字を超えないように注意してください。

   ユーザーprestonは多数のロールを持っているが、このユーザーがappuserプロキシ・アカウントを介してデータベースに接続しているときに使用するのは1つのロール(たとえばappuser_role)のみになるようにするとします。次のALTER USER文を使用できます。

   ```
   ALTER USER preston GRANT CONNECT THROUGH appuser WITH ROLE appuser_role;
   ```

ユーザーprestonが持っている他のロールはすべて、このユーザーがappuserプロキシとして接続しているかぎり使用できなくなります。

これらのステップが完了した後で、ユーザーprestonは、次のようにappuserプロキシ・ユーザーを使用して接続できます。

```
CONNECT appuser[preston]
Enter password: appuser_password
```

関連トピック

- Oracle Database SQL言語リファレンス
- Oracle Database SQL言語リファレンス

親トピック: プロキシ認証に対する中間層サーバーの使用

3.13.1.6 プロキシ・ユーザー・アカウントと、そのアカウントを介して接続するユーザーの認可

CREATE USER文を使用すると、次のタイプのユーザー・アカウントを作成でき、これらのすべてがプロキシ・アカウントとして使用できます。

これらのアカウントを次に示します。

- パスワードによって認証されるデータベース・ユーザー・アカウント
- 外部ユーザー・アカウント。Secure Socket Layer (SSL)やKerberosなどの外部ソースによって認証されます。
- グローバル・ユーザー・アカウント。エンタープライズ・ディレクトリ・サービス(Oracle Internet Directory)によって認証されます。

次のことに注意してください。

- プロキシ・ユーザーが実行できるのは、ユーザーprestonに実行権限があるアクティビティのみです。プロキシ・ユーザーのappuser自身が持っているのは、最低限の権限(CREATE SESSION)のみであることに注意してください。
- 中間層クライアントでのロールの使用。クライアントとして接続したときに、中間層でアクティブにすることが可能なロールも指定できます。中間層サーバーがクライアントのかわりに実行する操作は、監査の対象にできます。
- プロキシ・ユーザーの検索。現在中間層経由での接続が認可されているユーザーを検索するには、PROXY_USERSデータ・ディクショナリ・ビューを、たとえば次のように問い合せます。

  ```
  SELECT * FROM PROXY_USERS;
  ```

- プロキシ接続の取消し。プロキシ接続の認可を取り消すには、ALTER USER文のREVOKE CONNECT THROUGH句を使用します。たとえば、ユーザーprestonを、プロキシ・ユーザーappuserを介した接続から取り消すには、次の文を実行します。

  ```
  ALTER USER preston REVOKE CONNECT THROUGH appuser;
  ```

- パスワードの期限切れとプロキシ接続。中間層で使用されるuse ofパスワードの期限切れは、プロキシを介して認証されたアカウントに適用されません。パスワードを期限切れにするかわりに、アカウントをロックしてください。

関連トピック

- 複数層環境におけるSQL文および権限の監査
- Oracle Databaseエンタープライズ・ユーザー・セキュリティ管理者ガイド

親トピック: プロキシ認証に対する中間層サーバーの使用

3.13.1.7 安全性の高い外部パスワード・ストアとプロキシ認証の使用

プロキシ認証に使用しているパスワードが不正なユーザーによって取得される懸念がある場合は、安全性の高い外部パスワード・ストアを使用します。

これを行うには、プロキシ認証で安全性の高い外部パスワード・ストアを使用してパスワード資格証明をウォレットに格納します。

プロキシ認証と安全性の高い外部パスワード・ストアを使用したOracle Databaseへの接続は、バッチ・ファイルを実行するなどの状況に理想的です。プロキシ・ユーザーがデータベースに接続し、安全性の高い外部パスワードを使用して認証を行うと、不正なユーザーが取得しようとしてもパスワードは公開されません。

安全性の高い外部パスワード・ストアとプロキシ認証を使用するには:

1. 「プロキシ・ユーザー・アカウントと、そのアカウントを介して接続するユーザーの認可」の手順に示すように、プロキシ認証アカウントを構成します。
2. 「安全性の高い外部パスワード・ストアの使用を目的とするクライアントの構成について」で説明されているように、安全性の高い外部パスワード・ストアを構成します。

その後、ユーザーはパスワードを指定せずにプロキシを使用して接続できます。たとえば:

```
sqlplus [preston]/@db_alias
```

安全性の高い外部パスワード・ストアを使用すると、ユーザーはログイン時にユーザー名とパスワードを入力する必要がありません。指定する必要があるのは、`tnsnames.ora`ファイルの`SERVICE_NAME`値(つまり、`db_alias`)のみです。

親トピック: プロキシ認証に対する中間層サーバーの使用

3.13.1.8 プロキシ認証を使用した実際のユーザーの識別情報の引渡し

エンタープライズ・ユーザーまたはデータベース・ユーザーにOracle Call Interface、JDBC/OCIまたはシン・ドライバを使用できます。

これらのツールによって、中間層は複数のユーザー・セッションを1つのデータベース接続内で設定して各々のユーザー・セッションが接続ユーザーを一意に識別するようにできます(接続プーリング)。

これらのセッションにより、中間層からデータベースまで別個のネットワーク接続を作成することによるネットワーク・オーバーヘッドが削減されます。

クライアントから中間層を介してデータベースに対して認証する場合の完全な認証順序は次のようになります。

1. クライアントは、中間層が受け入れる任意の認証形式を使用して、中間層に対する認証を行います。たとえば、クライアントは、ユーザー名とパスワード、またはSSLによるX.509証明書を使用して、中間層に対する認証を実行できます。
2. 中間層は、データベースが受け入れる任意の認証形式を使用して、中間層自体をデータベースに対して認証します。認証形式には、パスワード、またはKerberosチケットやX.509証明書(SSL)などのOracle Databaseがサポートしている認証メカニズムがあります。
3. 次に、中間層はOCI、JDBC/OCIまたはシン・ドライバを使用して、ユーザーに対して1つ以上のセッションを作成します。
    - ユーザーがデータベース・ユーザーの場合、セッションには少なくともデータベース・ユーザー名が含まれている必要があります。データベースで必要な場合は、このセッションにパスワードを含めることができます(データベースでは、このパスワードをデータベース内のパスワード・ストアに対して検証します)。また、ユーザーに対するデータベース・ロールのリストを含めることもできます。

- ユーザーがエンタープライズ・ユーザーの場合、セッションはユーザーの認証方法に応じて異なる情報を提供します。

  例1: ユーザーがSSLを介して中間層に認証された場合、中間層は、そのユーザーのX.509証明書またはセッション内の証明書自体からDNを提供できます。データベースは、DNを使用してOracle Internet Directoryでユーザーを検索します。

  例2: ユーザーがパスワード認証方式のエンタープライズ・ユーザーの場合、中間層は、少なくともユーザーのグローバルな一意の名前を提供する必要があります。データベースは、この名前を使用してOracle Internet Directoryでユーザーを検索します。セッションがユーザーのパスワードも提供する場合、データベースでは、このパスワードをOracle Internet Directoryに対して検証します。ユーザー・ロールは、セッションが確立した後でOracle Internet Directoryから自動的に取得されます。

- 中間層は、必要に応じてクライアントに対するデータベース・ロールのリストを提供する場合があります。クライアントのかわりにロールを使用する権限がプロキシにある場合は、これらのロールが使用可能になります。

4. データベースは、ユーザーのかわりにセッションを作成する権限が中間層にあるかどうかを検証します。

   アプリケーション・サーバーがクライアントのかわりにプロキシ認証を実行することを管理者によって許可されていない場合、またはアプリケーション・サーバーが指定されたロールをアクティブにすることを許可されていない場合、`OCISessionBegin`コールは失敗します。

親トピック: プロキシ認証に対する中間層サーバーの使用

3.13.1.9 中間層の権限の制限

「最低限の権限」とは、ユーザーの権限は各自の職務を行うのに必要な最小限の権限までにする必要があるという原則です。

これを中間層アプリケーションに当てはめると、中間層は必要以上の権限を持つ必要はないということを意味します。

Oracle Databaseでは、中間層が特定のデータベース・ユーザーのかわりとしてのみ、特定のデータベース・ロールのみを使用して接続できるように、中間層を制限できます。LDAPディレクトリに保存されたエンタープライズ・ユーザーのかわりに接続するよう中間層の権限を制限するために、マップされたデータベース・ユーザーとして接続する権限を中間層に付与します。たとえば、エンタープライズ・ユーザーがAPPUSERスキーマにマップされている場合は、少なくともAPPUSERのかわりに接続する機能を中間層に付与する必要があります。そうでない場合は、エンタープライズ・ユーザーのセッションを作成しようとした場合に失敗します。

ただし、中間層がエンタープライズ・ユーザーのかわりに接続する機能は制限できません。たとえば、ユーザーSarahが中間層`appsrv`(データベース・ユーザーでもある)を介してデータベースに接続するとします。Sarahには複数のロールがありますが、Sarahのかわりに`clerk`ロールのみを使用できるように中間層を制限します。

管理者は、次のSQL文を使用して、`appsrv`に対して、Sarahの`clerk`ロールのみを使用してSarahのかわりに接続を開始する許可を付与できます。

```
ALTER USER sarah GRANT CONNECT THROUGH appsrv WITH ROLE clerk;
```

デフォルトでは、中間層はどのクライアントに対する接続も確立できません。許可はユーザーごとに付与する必要があります。

`appsrv`に対して、クライアントSarahに付与されているすべてのロールの使用を許可するには、次の文を使用します。

```
ALTER USER sarah GRANT CONNECT THROUGH appsrv;
```

中間層が別のデータベース・ユーザーのOCI、JDBC/OCIまたはシン・ドライバ・セッションを開始するたびに、データベースでは指定されたロールを使用して、そのユーザーに対する接続を開始する権限が中間層にあることを検証します。

ノート:

デフォルトのロールを使用せずに、独自のロールを作成し、そのロールに必要な権限のみを割り当ててください。独自のロールを作成すると、ロールによって付与される権限を制御でき、Oracle Database でデフォルトのロールが変更または削除された場合も保護されます。たとえば、現在、CONNECT ロールには、データベースへの接続で直接必要になる CREATE SESSION 権限のみが含まれています。しかし以前、CONNECT には、ほとんどのユーザーには不要または不適切ないくつかの追加権限がありました。余分な権限は、データベースやアプリケーションのセキュリティを危険にさらす可能性があります。これらは、CONNECT から削除されています。

プロキシ・セッションのプロキシ・ユーザーは、WITH ROLE または WITH ROLE ALL 句でロールを有効にできる場合にのみ、パスワード保護されたロールまたはセキュア・アプリケーション・ロールを有効にできます。(この句を指定しない場合、WITH ROLE ALL がデフォルトになります。)WITH ROLE でセキュア・ロールを指定しない場合、正しいパスワードによってもこれらのロールを有効にできません。

関連トピック

- 権限とロール認可の構成

親トピック: プロキシ認証に対する中間層サーバーの使用

3.13.1.10 ユーザーのプロキシとして機能し、ユーザーを認証する中間層を認可する方法

ユーザーとして接続する中間層サーバーを認可できます。

プロキシ・セッションのプロキシ・ユーザーは、WITH ROLEまたはWITH ROLE ALL句でロールを有効にできる場合にのみ、パスワード保護されたロールまたはセキュア・アプリケーション・ロールを有効にできます。(この句を指定しない場合、WITH ROLE ALLがデフォルトになります。)WITH ROLEでセキュア・ロールを指定しない場合、正しいパスワードによってもこれらのロールを有効にできません。

- ユーザーとして接続する中間層サーバーを認可するには、ALTER USER文を使用します。

次の文は、中間層サーバーappserveがユーザーbillとして接続するのを認可します。WITH ROLE句を使用して、appserveがbillに関連付けられた、payrollを除くすべてのロールをアクティブにするよう指定します。

```
ALTER USER bill
    GRANT CONNECT THROUGH appserve
    WITH ROLE ALL EXCEPT payroll;
```

ユーザーbillとして接続するための中間層サーバーの(appserve)認可を取り消すには、REVOKE CONNECT THROUGH句を使用できます。たとえば:

```
ALTER USER bill REVOKE CONNECT THROUGH appserve;
```

親トピック: プロキシ認証に対する中間層サーバーの使用

3.13.1.11 他の方式で認証されたユーザーのプロキシとして機能するために、中間層を認可する方法

他の方式で認証されたユーザーのプロキシとして機能するための中間層を認可できます。

現在サポートされている認証方式は、PASSWORDのみです。

- ALTER USER ... GRANT CONNECT THROUGH文の AUTHENTICATION REQURED句を使用して、中間層がプロキシ化するユーザーを認可するが、認証はしません。

たとえば:

```
ALTER USER mary
    GRANT CONNECT THROUGH midtier
    AUTHENTICATION REQUIRED;
```

この文の中間層サーバーmidtierは、maryとしての接続を認可されており、midtierは、認証のためにユーザー・パスワードをデータベース・サーバーにも渡す必要があります。

親トピック: プロキシ認証に対する中間層サーバーの使用

3.13.1.12 中間層を介したデータベースへのユーザーの再認証

ALTER USER SQL文でAUTHENTICATION REQUIREDプロキシ句を使用して、認証が必要であることを指定できます。

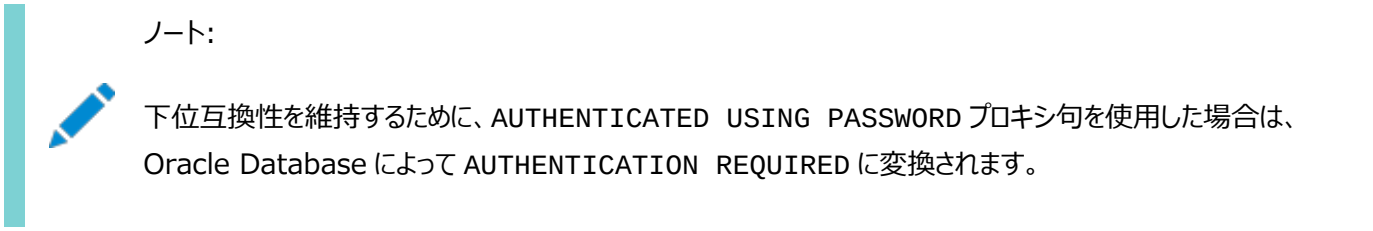
この場合、中間層はユーザーの認証資格証明を提供する必要があります。

たとえば、ユーザーSarahが中間層appsrvを介してデータベースに接続するとします。

- appsrvに対してユーザーSarahの認証資格証明を提供するように要求するには、次の構文を使用します。

  ```
  ALTER USER sarah GRANT CONNECT THROUGH appsrv AUTHENTICATION REQUIRED;
  ```

AUTHENTICATION REQUIRED句は、ユーザーが指定されたプロキシを介して認証される場合に、ユーザーの認証資格証明が提示される必要があることを示しています。



ノート:

下位互換性を維持するために、AUTHENTICATED USING PASSWORD プロキシ句を使用した場合は、Oracle Database によって AUTHENTICATION REQUIRED に変換されます。

親トピック: プロキシ認証に対する中間層サーバーの使用

3.13.1.13 パスワード・ベースのプロキシ認証の使用

パスワード・ベースのプロキシ認証を使用すると、Oracle Databaseはクライアントのパスワードを中間層サーバーに渡します。

次に、中間層サーバーは、そのパスワードを属性として、検証のためにデータ・サーバーに渡します。

この認証の主な利点は、データベース操作を実行するために、クライアント・コンピュータにOracleソフトウェアをインストールする必要がないことです。

- クライアントのパスワードを渡す場合、設定する属性のタイプとしてOCI_ATTR_PASSWORDを渡して、次のようにOCIAttrSet()関数をコールするように中間層サーバーを構成します。

  ```
  OCIAttrSet(
    session_handle,     /* Pointer to a handle whose attribute gets modified. */
    OCI_HTYPE_SESSION, /* Handle type: OCI user session handle. */
    password_ptr,       /* Pointer to the value of the password attribute. */
    0,                  /* The size of the password attribute value is already
                           known by the OCI library. */
    OCI_ATTR_PASSWORD, /* The attribute type. */
    error_handle);      /* An error handle used to retrieve diagnostic
                           information in the event of an error. */
  ```

親トピック: プロキシ認証に対する中間層サーバーの使用

3.13.1.14 エンタープライズ・ユーザーでのプロキシ認証の使用

プロキシ認証に対する中間層の応答方法は、ユーザーがエンタープライズ・ユーザーとして認証されているか、パスワード認証ユーザーとして認証されているかによって異なります。

中間層がエンタープライズ・ユーザーであるクライアントとしてデータベースに接続している場合は、識別名または識別名を含むX.509証明書のいずれかが、データベース・ユーザー名のかわりに渡されます。ユーザーがパスワード認証方式のエンタープライズ・ユーザーの場合、中間層は、少なくともユーザーのグローバルな一意の名前を提供する必要があります。データベースは、この名前を使用してOracle Internet Directoryでユーザーを検索します。

- エンタープライズ・ユーザーでプロキシ認証を構成するには、適切なOracle Call Interface設定を使用するようにアプリケーション・サーバーと中間層を構成します。
    - クライアントの識別名を渡す場合、次のように、属性タイプとして`OCI_ATTR_DISTINGUISHED_NAME`を指定して、Oracle Call Interfaceメソッド`OCIAttrSet()`をコールするようにアプリケーション・サーバーを構成します。

      ```
      OCIAttrSet(session_handle,
                 OCI_HTYPE_SESSION,
                 distinguished_name,
                 0,
                 OCI_ATTR_DISTINGUISHED_NAME,
                 error_handle);
      ```

    - 証明書全体を渡す場合、次のように、属性タイプとして`OCI_ATTR_CERTIFICATE`を指定して、`OCIAttrSet()`をコールするように中間層を構成します。

      ```
      OCIAttrSet(session_handle,
                 OCI_HTYPE_SESSION,
                 certificate,
                 certificate_length,
                 OCI_ATTR_CERTIFICATE,
                 error_handle);
      ```

証明書のタイプが指定されていない場合、データベースはデフォルトの証明書タイプX.509を使用します。

ノート:

- `OCI_ATTR_CERTIFICATE` は、Distinguished Encoding Rules(DER)でエンコードされています。
- `OCI_ATTR_CERTIFICATE` を使用する証明書ベースのプロキシ認証は、Oracle Database の将来のリリースではサポートされない予定です。かわりに、`OCI_ATTR_DISTINGUISHED_NAME` または `OCI_ATTR_USERNAME` 属性を使用してください。

パスワード認証方式のエンタープライズ・ユーザーにプロキシ認証を使用する場合は、パスワードで認証されるデータベース・ユーザーと同じOCI属性(`OCI_ATTR_USERNAME`)を使用します。Oracle Databaseでは、最初にユーザー名をデータベースに対してチェックします。ユーザーが見つからなかった場合、データベースはディレクトリ内のユーザー名をチェックします。このユーザー名はグローバルに一意である必要があります。

親トピック: プロキシ認証に対する中間層サーバーの使用

3.13.2 データベースに認識されないアプリケーション・ユーザーの識別でのクライアント識別子の使用

クライアント識別子は、中間層システムでユーザー識別情報を保持します。また、グローバル・アプリケーション・コンテキストとは独立してこれらを使用することもできます。

- クライアント識別子について
  Oracle Databaseでは、アプリケーション・ユーザーに対して、組込みアプリケーション・コンテキスト・ネームスペースUSERENVのCLIENT_IDENTIFIER属性を提供します。
- 中間層システムでのクライアント識別子の使用方法
  多くのアプリケーションがセッション・プーリングを使用して、複数のアプリケーション・ユーザーが再利用する複数のセッションを設定します。
- CLIENT_IDENTIFIER属性を使用したユーザー識別情報の保持
  組込みアプリケーション・コンテキスト・ネームスペースUSERENVの事前定義の属性CLIENT_IDENTIFIERは、グローバル・アプリケーション・コンテキストで使用するアプリケーション・ユーザー名を取得します。
- グローバル・アプリケーション・コンテキストから独立したCLIENT_IDENTIFIERの使用
  CLIENT_IDENTIFIER属性は、ユーザーがデータベースに認識されていないアプリケーションで特に役立ちます。
- グローバル・アプリケーション・コンテキストから独立したCLIENT_IDENTIFIERの設定
  グローバル・アプリケーション・コンテキストから独立するように、Oracle Call Interfaceによって CLIENT_IDENTIFIER設定を設定できます。
- DBMS_SESSION PL/SQLパッケージを使用したクライアント識別子の設定とクリア
  DBMS_SESSION PL/SQLパッケージは、中間層とデータベース自体の両方でクライアント識別子を管理します。
- システム全体でのCLIENTID_OVERWRITEイベントの有効化
  ALTER SYSTEM文は、システム全体でCLIENTID_OVERWRITEイベントを有効化できます。
- 現在のセッションに対するCLIENTID_OVERWRITEイベントの有効化
  ALTER SESSION文は、現在のセッションのみに対してCLIENTID_OVERWRITEイベントを有効化できます。
- CLIENTID_OVERWRITEイベントの無効化
  ALTER SYSTEM文は、CLIENTID_OVERWRITEイベントを無効化できます。

親トピック: 複数層環境でのユーザー識別情報の保持

3.13.2.1 クライアント識別子について

Oracle Databaseでは、アプリケーション・ユーザーに対して、組込みアプリケーション・コンテキスト・ネームスペースUSERENVのCLIENT_IDENTIFIER属性を提供します。

これらのアプリケーション・ユーザーは、アプリケーションには認識されますが、データベースには認識されません。CLIENT_IDENTIFIER属性は、アプリケーションで識別またはアクセス制御に使用する任意の値を取得し、その値をデータベースに渡すことができます。CLIENT_IDENTIFIER属性は、OCI、JDBC/OCIまたはシン・ドライバでサポートされています。

親トピック: データベースに認識されないアプリケーション・ユーザーの識別でのクライアント識別子の使用

3.13.2.2 中間層システムでのクライアント識別子の使用方法

多くのアプリケーションがセッション・プーリングを使用して、複数のアプリケーション・ユーザーが再利用する複数のセッションを設定します。

ユーザーは、単一識別情報を使用してデータベースにログイン後、すべてのユーザー接続を維持する中間層アプリケーションに対して認証します。このモデルでは、アプリケーション・ユーザーはアプリケーションの中間層に対して認証されているがデータベース

には認識されていないユーザーです。ここでは、これらのタイプのアプリケーションのアプリケーション・ユーザー・プロキシのように機能するCLIENT_IDENTIFIER属性を使用できます。

このモデルでは、中間層はセッション確立時にクライアント識別子をデータベースに渡します。クライアント識別子は、中間層に接続しているクライアント表す任意のもの(たとえばCookieやIPアドレスなど)です。アプリケーション・ユーザーを表しているクライアント識別子はユーザー・セッション情報の中にあり、(USERENVネーミング・コンテキストを使用して)アプリケーション・コンテキストによりアクセスすることもできます。このようにして、アプリケーションはセッションを設定して再利用できると同時に、セッション内でアプリケーション・ユーザーを追跡できます。アプリケーションはクライアント識別子をリセットできるため、異なるユーザーでセッションを再利用し、パフォーマンスが向上します。

親トピック: データベースに認識されないアプリケーション・ユーザーの識別でのクライアント識別子の使用

3.13.2.3 CLIENT_IDENTIFIER属性を使用したユーザー識別情報の保持

組込みアプリケーション・コンテキスト・ネームスペースUSERENVの事前定義の属性CLIENT_IDENTIFIERは、グローバル・アプリケーション・コンテキストで使用するアプリケーション・ユーザー名を取得します。

CLIENT_IDENTIFIER属性は独立して使用することもできます。

CLIENT_IDENTIFIER属性をグローバル・アプリケーション・コンテキストから独立して使用する場合、CLIENT_IDENTIFIERは、DBMS_SESSIONインタフェースを使用して設定できます。CLIENT_IDENTIFIERをデータベースに渡す機能は、Oracle Call Interface(OCI)、JDBC/OCIまたはシン・ドライバでサポートされています。

CLIENT_IDENTIFIER属性をグローバル・アプリケーション・コンテキストで使用すると、アプリケーションの作成に必要な柔軟性と高いパフォーマンスが得られます。たとえば、ビジネス・パートナに情報を提供するWebベース・アプリケーションに、ゴールド・パートナ、シルバー・パートナおよびブロンズ・パートナという3タイプのユーザーが用意されていて、それぞれが異なるレベルの使用可能な情報を表しているとします。アプリケーションでは、ユーザーごとに個別のアプリケーション・コンテキストを持つユーザー独自のセッションを設定するのではなく、ゴールド・パートナ、シルバー・パートナおよびブロンズ・パートナ用のグローバル・アプリケーション・コンテキストを設定できます。次に、CLIENT_IDENTIFIERを使用して正しいコンテキストのセッションを指すことによって、適切なタイプのデータを取得します。アプリケーションでは、この3つのグローバル・コンテキストを一度初期化すれば、CLIENT_IDENTIFIERを使用して適切なアプリケーション・コンテキストにアクセスし、データ・アクセスを制限できます。これには、セッションを再利用できるということと、セッションごとにアプリケーション・コンテキストを個別に初期化する必要がなく、一度設定したグローバル・アプリケーション・コンテキストにアクセスできるというパフォーマンス上のメリットがあります。

関連トピック

- グローバル・アプリケーション・コンテキスト
- 例: クライアント・セッションIDを使用するグローバル・アプリケーション・コンテキストの作成

親トピック: データベースに認識されないアプリケーション・ユーザーの識別でのクライアント識別子の使用

3.13.2.4 グローバル・アプリケーション・コンテキストから独立したCLIENT_IDENTIFIERの使用

CLIENT_IDENTIFIER属性は、ユーザーがデータベースに認識されていないアプリケーションで特に役立ちます。

このような場合、アプリケーションは通常、単一のデータベース・ユーザーとして接続し、すべてのアクションがそのユーザーで実行されます。

すべてのユーザー・セッションが同じユーザーとして作成されるため、このセキュリティ・モデルでは、ユーザーごとにデータを分離することが困難になります。これらのアプリケーションでは、CLIENT_IDENTIFIER属性を使用すると、実際のアプリケーション・ユーザーの識別情報をデータベースに保持できます。

この方法によると、CLIENT_IDENTIFIER属性の値を変更することで、複数のユーザーがセッションを再利用できます(この属

性は、実際のアプリケーション・ユーザーの名前を取得します)。結果として、ユーザーごとに個別のセッションと属性を設定するためのオーバーヘッドが回避され、アプリケーションによるセッションの再利用が可能になります。CLIENT_IDENTIFIER属性の値が変更されると、その変更は次のOCIコール、JDBC/OCIコールまたはシン・ドライバ・コールに伝達されるため、パフォーマンスが向上します。

たとえば、ユーザーDanielはWeb Expenseアプリケーションに接続します。Danielはデータベース・ユーザーではなく、一般的なWeb Expenseアプリケーション・ユーザーです。アプリケーションは組込みアプリケーション・コンテキスト・ネームスペースにアクセスして、DANIELをCLIENT_IDENTIFIER属性値として設定します。DanielはWeb Expenseフォームを記入し終わるとアプリケーションを終了します。その後に、AjitがWeb Expenseアプリケーションに接続します。Ajitのために新しいセッションを設定するかわりに、アプリケーションはCLIENT_IDENTIFIERをAJITに変更することにより、現在Danielに存在しているセッションを再利用します。これによりデータベースに新しい接続を設定するオーバーヘッドと、グローバル・アプリケーション・コンテキストを設定するオーバーヘッドを回避できます。CLIENT_IDENTIFIER属性は、アプリケーションがアクセス制御のベースにする任意の値に設定できます。アプリケーション・ユーザー名である必要はありません。

親トピック: データベースに認識されないアプリケーション・ユーザーの識別でのクライアント識別子の使用

3.13.2.5 グローバル・アプリケーション・コンテキストから独立したCLIENT_IDENTIFIERの設定

グローバル・アプリケーション・コンテキストから独立するように、Oracle Call InterfaceによってCLIENT_IDENTIFIER設定を設定できます。

- CLIENT_IDENTIFIER属性をOCIによって設定する場合は、OCIAttrSet()のコールでOCI_ATTR_CLIENT_IDENTIFIER属性を使用します。この結果、サーバーに対する次のリクエスト時にその情報が伝播され、サーバー・セッションに格納されます。

たとえば:

```
OCIAttrSet (session,

OCI_HTYPE_SESSION,
(dvoid *) "appuser1",
(ub4)strlen("appuser1"),
OCI_ATTR_CLIENT_IDENTIFIER,
*error_handle);
```

JDBCを使用するアプリケーションの場合、JDBCではクライアント識別子が設定されないことに注意してください。クライアント識別子を接続プール環境で設定するには、Dynamic Monitoring Service (DMS)メトリックを使用します。DMSを使用できない場合は、connection.setClientInfoメソッドを使用してください。たとえば:

```
connection.setClientInfo("E2E_CONTEXT.CLIENT_IDENTIFIER", "appuser");
```

関連項目:

- OCI_ATTR_CLIENT_IDENTIFIERユーザー・セッション・ハンドル属性の中間層アプリケーションでの使用方法は、『Oracle Call Interfaceプログラマーズ・ガイド』を参照してください。
- JDBCおよびDMSメトリックを使用してクライアント接続を構成する方法の詳細は、『Oracle Database JDBC開発者ガイド』を参照してください。
- setClientInfoメソッドの詳細は、『Oracle Database JDBC開発者ガイド』を参照してください。

親トピック: データベースに認識されないアプリケーション・ユーザーの識別でのクライアント識別子の使用

3.13.2.6 DBMS_SESSION PL/SQLパッケージを使用したクライアント識別子の設定とクリア

DBMS_SESSION PL/SQLパッケージは、中間層とデータベース自体の両方でクライアント識別子を管理します。

DBMS_SESSIONパッケージを使用して、中間層でCLIENT_IDENTIFIERの値を設定および消去するには、SET_IDENTIFIERプロシージャとCLEAR_IDENTIFIERプロシージャを使用します。

中間層では、SET_IDENTIFIERを使用してデータベース・セッションを特定のユーザーまたはグループに対応付けます。この結果、CLIENT_IDENTIFIERはセッションの属性になるため、セッション情報で確認できます。

DBMS_SESSION.SET_IDENTIFIERプロシージャを使用する場合は、次の点に注意してください。

- DBMS_SESSION.SET_IDENTIFIERのclient_idパラメータの最大バイト数は64バイトです。64を超えると、超えた分のバイトは切り捨てられます。
- DBMS_APPLICATION_INFO.SET_CLIENT_INFOプロシージャでクライアント識別子の値を上書きできます。通常、これらの値は一致する必要があるため、CLIENTID_OVERWRITEイベントがONに設定されている場合は、SET_CLIENT_INFOが設定されていれば、その値をSET_IDENTIFIERにより設定された値に自動的に伝播できます。CLIENTID_OVERWRITEイベントの状態をチェックするには、SHOW PARAMETERコマンドをEVENTパラメータで実行します。

  たとえば、CLIENTID_OVERWRITEが使用可能になっているとします。

```
SHOW PARAMETER EVENT
NAME                                 TYPE                 VALUE
------------------------------------ -------------------- ------------------
event                                string               clientid_overwrite
```

親トピック: データベースに認識されないアプリケーション・ユーザーの識別でのクライアント識別子の使用

3.13.2.7 システム全体でのCLIENTID_OVERWRITEイベントの有効化

ALTER SYSTEM文は、システム全体でCLIENTID_OVERWRITEイベントを有効化できます。

1. 次のALTER SYSTEM文を入力します。

   ```
   ALTER SYSTEM SET EVENTS 'CLIENTID_OVERWRITE';
   ```

   または、init.oraファイルに次の行を入力します。

   ```
   event="clientid_overwrite"
   ```

2. データベースを再起動します。

   たとえば:

   ```
   SHUTDOWN IMMEDIATE
   STARTUP
   ```

関連項目:

- クライアント識別子をグローバル・アプリケーション・コンテキストで使用する方法は、「グローバル・アプリケーション・コンテキスト」を参照してください。
- DBMS_SESSIONパッケージの詳細は、『Oracle Database PL/SQLパッケージおよびタイプ・リファレンス』を参照してください。

親トピック: データベースに認識されないアプリケーション・ユーザーの識別でのクライアント識別子の使用

3.13.2.8 現在のセッションに対するCLIENTID_OVERWRITEイベントの有効化

ALTER SESSION文は、現在のセッションのみに対してCLIENTID_OVERWRITEイベントを有効化できます。

1. ALTER SESSION文を使用して、セッションのみに対してCLIENTID_OVERWRITEの値を設定します。

   たとえば:

   ```
   ALTER SESSION SET EVENTS 'CLIENTID_OVERWRITE OFF';
   ```

2. DBMS_APPLICATION_INFO.SET_CLIENT_INFOプロシージャを使用してクライアント識別子を設定する場合は、クライアント識別子の設定が同一になるようにDBMS_SESSION.SET_IDENTIFIERを実行します。

   たとえば:

   ```
   DBMS_SESSION.SET_IDENTIFIER(session_id_p);
   ```

親トピック: データベースに認識されないアプリケーション・ユーザーの識別でのクライアント識別子の使用

3.13.2.9 CLIENTID_OVERWRITEイベントの無効化

ALTER SYSTEM文は、CLIENTID_OVERWRITEイベントを無効化できます。

1. 次のALTER SYSTEM文を入力します。

   ```
   ALTER SYSTEM SET EVENTS 'CLIENTID_OVERWRITE OFF';
   ```

2. データベースを再起動します。

   たとえば:

   ```
   SHUTDOWN IMMEDIATE
   STARTUP
   ```

親トピック: データベースに認識されないアプリケーション・ユーザーの識別でのクライアント識別子の使用

3.14 ユーザー認証のデータ・ディクショナリ・ビュー

Oracle Databaseには、ユーザーのロールや使用しているプロファイルなど、ユーザー認証に関する情報を表示するデータ・ディクショナリ・ビューが用意されています。

表3-5に、データ・ディクショナリ・ビューを示します。

表3-5 ユーザー認証を示すデータ・ディクショナリ・ビュー

ビュー	説明
DBA_PROFILES	設定や制限など、プロファイルに関する情報を表示します。
DBA_ROLES	データベース・ロールがデータベースにログインする際に使用する認証の種類を表示します。NONE または GLOBAL などがあります(AUTHENTICATION_TYPE 列を問い合せます)

ビュー	説明
DBA_USERS	その他のユーザー情報から、次の情報を表示します。 ● PASSWORD または EXTERNAL などの、ユーザーがデータベースにログインする際に使用する認証の種類を表示します(AUTHENTICATION_TYPE 列) ● ユーザー・アカウントに存在するパスワード・バージョン(ハッシュとも呼ばれる)のバージョンのリスト(PASSWORD_VERSIONS 列)
DBA_USERS_WITH_DEFPWD	ユーザー・アカウント・パスワードがデフォルト・パスワードかどうかを表示します
PROXY_USERS	現在中間層経由での接続が認可されているユーザーを表示します
V$DBLINK	既存のデータベース・リンク用のユーザー・アカウントを表示します(DB_LINK、OWNER_ID 列)。現在のプラガブル・データベース(PDB)に適用します
V$PWFILE	パスワード・ファイルに含まれている管理ユーザーの名前と付与された管理権限をリストします
V$SESSION	USERNAME 列を問い合せると、現在の PDB に同時ログインしているユーザーが表示されます

関連トピック

- Oracle Databaseリファレンス

親トピック: 認証の構成

4 権限とロール認可の構成

権限とロールの認可によって、ユーザーが毎日のタスクを実行するために保持する権限が制御されます。

- 権限とロールについて
  認可とは、データへのアクセス、処理または変更を特定のユーザーに許可するほか、ユーザーのアクセスやアクションに関する制限を作成するものです。
- 権限付与の対象者
  権限をユーザーに付与すると、そのユーザーはそれぞれの業務に必要な作業を実行できます。
- Oracleマルチテナント・オプションが権限に影響を与えるしくみ
  マルチテナント環境では、共通ユーザーを含むすべてのユーザーは、現在のコンテナ内でのみ権限を実行できます。
- 管理権限の管理
  一般的なデータベース操作と特定のデータベース操作の両方に管理権限を使用できます。
- システム権限の管理
  スキーマ・オブジェクトに対するアクションを実行するには、適切なシステム権限が付与されている必要があります。
- 共通およびローカルに付与される権限の管理
  マルチテナント環境では、CDB全体またはアプリケーション・コンテナに対する共通の権限を付与したり、特定のPDBに対してローカルで権限を付与したりできます。
- 共通ロールおよびローカル・ロールの管理
  共通ロールはルートで作成されるロールであり、ローカル・ロールはPDBで作成されます。
- ユーザー・ロールの管理
  ユーザー・ロールは、作成したり他のユーザーに割り当てることができる権限の名前付きコレクションです。
- PDBロックダウン・プロファイルを使用したPDBでの操作の制限
  マルチテナント環境でPDBロックダウン・プロファイルを使用して、プラガブル・データベース(PDB)の一連のユーザー操作を制限できます。
- オブジェクト権限の管理
  オブジェクト権限を使用すると、表や索引などのスキーマ・オブジェクトに対するアクションを実行できます。
- 表権限
  表に対するオブジェクト権限は、DMLまたはDDLレベルの操作に対する表セキュリティを実現します。
- ビューに対する権限
  DMLオブジェクト権限は、表の場合と同様にビューに対しても適用できます。
- プロシージャ権限
  EXECUTE権限は、スタンドアロンまたはパッケージ内でのプロシージャまたは関数の実行をユーザーに許可します。
- タイプ権限
  型、メソッドおよびオブジェクトについて、システム権限とオブジェクト権限を制御できます。
- ユーザーへの権限とロールの付与
  GRANT文は、プロシージャの実行など、特定のアクションを実行する権限をユーザーに付与します。
- ユーザーからの権限とロールの取消し
  システムまたはオブジェクトの権限を取り消す場合は、権限の取消しによる連鎖的な影響に注意してください。
- PUBLICロールに対する権限の付与と取消し
  ロールPUBLICに対して、権限とロールの付与および取消しを実行できます。
- オペレーティング・システムまたはネットワークを使用したロールの付与
  オペレーティング・システムまたはネットワークを使用してロールを管理すると、大規模エンタープライズでのロールの一元

管理が容易になります。

- SET ROLEおよびデフォルト・ロールの設定による権限の付与と取消しの機能
  権限付与およびSET ROLE文は、付与と取消しが適用されるタイミングと方法に影響します。
- ユーザー権限およびロールのデータ・ディクショナリ・ビュー
  特別な問合せを使用して、様々なタイプの権限およびロール付与に関する情報を入手できます。

親トピック: ユーザー認証および認可の管理

4.1 権限とロールについて

認可とは、データへのアクセス、処理または変更を特定のユーザーに許可するほか、ユーザーのアクセスやアクションに関する制限を作成するものです。

ユーザーに対して課せられる(または除外される)制限は、スキーマ、表全体または表の行などのオブジェクトに適用できます。

ユーザー権限とは、特定タイプのSQL文を実行する権利、別のユーザーのオブジェクトにアクセスする権利、PL/SQLパッケージを実行する権利などを指します。権限のタイプは、Oracle Databaseによって定義されています。

ロールは、ユーザー(通常は管理者)が権限や他のロールをグループ化するために作成します。ロールを使用すると、複数の権限またはロールをユーザーに簡単に付与できます。

権限は、次の一般的なカテゴリに分類されます。

- システム権限。この権限の受領者は、データベースで標準的な管理作業を実行できます。権限受領者は信頼できるユーザーのみに制限してください。権限について説明している次の項を参照してください。
  - 管理権限の管理
  - システム権限の管理
  - 共通およびローカルに付与される権限の管理
- ロール。ロールでは、複数の権限やロールがグループ化されるため、複数のユーザーに対して権限を同時に付与したり、取り消すことができます。ユーザーによるロールの使用を可能にするには、そのユーザーに対してロールを使用可能にしておく必要があります。詳細は、次の各項を参照してください。
  - 共通ロールおよびローカル・ロールの管理
  - ユーザー・ロールの管理
- オブジェクト権限。オブジェクトの各タイプには、オブジェクト権限が対応付けられています。異なるタイプのオブジェクト権限を管理する方法は、オブジェクト権限の管理を参照してください。
- 表権限。これらの権限によって、DML (データ操作言語)またはDDL (データ定義言語)レベルでセキュリティが有効になります。表権限の管理方法の詳細は、表権限を参照してください。
- 表示権限。DMLオブジェクト権限は、表の場合と同様にビューに対しても適用できます。詳細は、ビューに対する権限を参照してください。
- プロシージャ権限。スタンドアロン・プロシージャ、ファンクションを含め、プロシージャにはEXECUTE権限を付与できます。詳細は、プロシージャ権限を参照してください。
- タイプ権限。システム権限は名前付きタイプ(オブジェクト・タイプ、VARRAYおよびネストした表)に付与できます。詳細は、タイプ権限を参照してください。

関連項目:

権限の使用を分析するポリシーの作成方法の詳細は、『Oracle Database Vault管理者ガイド』を参照してください。

親トピック: 権限とロール認可の構成

4.2 権限付与の対象者

権限をユーザーに付与すると、そのユーザーはそれぞれの業務に必要な作業を実行できます。

なお、権限は、必要な作業を実行する上でその権限が必要なユーザーにのみ付与してください。必要でない権限まで付与すると、セキュリティを維持できなくなる可能性があります。たとえば、管理作業を実行しないユーザーには、SYSDBA管理権限またはSYSOPER権限を付与しないでください。

権限は、次の2つの方法でユーザーに付与できます。

- 権限を明示的にユーザーに付与します。たとえば、`employees`表にレコードを挿入する権限を、ユーザー`psmith`に明示的に付与できます。
- 権限をロール(名前付きの権限グループ)に付与した上で、そのロールを1人以上のユーザーに付与します。たとえば、`employees`表からレコードを選択、挿入、更新および削除する権限を、`clerk`という名前のロールに付与し、このロールをユーザー`psmith`や`robert`に付与できます。

ロールを使用することで権限の管理が容易になり、改善されるため、通常は権限を個々のユーザーではなくロールに付与してください。

関連項目:

- 権限を付与する際に従うベスト・プラクティスは、ユーザー・アカウントと権限の保護に関するガイドラインを参照してください
- 過度の権限付与が懸念される場合は、『Oracle Database Vault管理者ガイド』を参照してください。
- システム権限の完全なリストとその詳細は、『Oracle Database SQL言語リファレンス』を参照してください。

親トピック: 権限とロール認可の構成

4.3 Oracleマルチテナント・オプションが権限に影響を与えるしくみ

マルチテナント環境では、共通ユーザーを含むすべてのユーザーは、現在のコンテナ内でのみ権限を実行できます。

ただし、ルートに接続されているユーザーは、他のプラガブル・データベース(PDB)に影響を与える特定の操作を実行できます。これらの操作には、`ALTER PLUGGABLE DATABASE`、`CREATE USER`、`CREATE ROLE`および`ALTER USER`が含まれます。共通ユーザーは、これらの操作を可能にする、共通に付与される権限を持つ必要があります。ルートに接続されている共通ユーザーは、ビューにアクセスするために必要な権限を付与されていて、様々なPDBに関するデータを表示できるように`CONTAINER_DATA`属性が設定されている場合、ルートのコンテナ・データ・オブジェクト(たとえば、マルチテナント・コンテナ・データベース(CDB)・ビューやV$ビューなど)を介してPDBに関するメタデータを確認できます。共通ユーザーは、PDBの表またはビューに問合せできません。

共通ユーザーは、他のPDBの権限を実行できません。必要なPDBに最初に切り替えて、そこから権限を実行する必要があります。異なるコンテナに切り替えるには、共通ユーザーに`SET CONTAINER`権限が必要です。`SET CONTAINER`権限は、共通に付与するか、ユーザーが切替えを試みるコンテナに付与する必要があります。また、共通ユーザーは、そのPDBの`CREATE`

SESSION権限に応じて、現在の初期コンテナがこのユーザーが必要とするコンテナである新しいデータベース・セッションを開始できます。

共通に付与される権限が個々のPDBに構成されたセキュリティを妨げる場合があるので注意してください。たとえば、アプリケーションPDBのデータベース管理者がPDBのいずれのユーザーも特定のアプリケーション共通オブジェクトを変更できないようにするとします。PUBLICまたはオブジェクトの共通ユーザーもしくは共通ロールに共通に付与された権限(UPDATEなど)は、PDBのデータベース管理者の意図に反した動作をします。

関連トピック

- 共通ユーザーによるCONTAINER_DATAオブジェクトの情報の表示

親トピック: 権限とロール認可の構成

4.4 管理権限の管理

一般的なデータベース操作と特定のデータベース操作の両方に管理権限を使用できます。

- 管理権限について
  Oracle Databaseではより適切な作業分担を実現するため、一般的に実施される各管理タスク用の管理権限が用意されています。
- ユーザーへの管理権限の付与
  すべての強力な権限と同様に、管理権限は信頼できるユーザーのみに付与してください。
- 標準データベース操作のためのSYSDBAおよびSYSOPER権限
  SYSDBAおよびSYSOPER管理権限を使用すると、標準データベース操作を実行できます。
- SYSDBAとしてのログイン時のoracleユーザーに対するパスワード入力の強制
  ユーザーがSYSDBA管理権限を使用してOracleデータベースにログインするときに、oracleユーザーにパスワードの入力を強制できます。
- バックアップおよびリカバリ操作のSYSBACKUP管理権限
  Oracle Recovery Manager (RMAN)またはSQL*Plusを使用したバックアップおよびリカバリ操作を実行するには、SYSBACKUP管理権限を使用します。
- Oracle Data Guard操作のSYSDG管理権限
  SYSDG管理権限があるSYSDGユーザーとしてログインして、Data Guard操作を実行できます。
- 透過的データ暗号化のSYSKM管理権限
  SYSKM管理権限により、SYSKMユーザーは、透過的データ暗号化(TDE)のウォレット操作を管理できます。
- Oracle Real Application ClustersのSYSRAC管理権限
  SYSRAC管理権限はOracle Real Application Clusters (Oracle RAC)のClusterwareエージェントによって使用されます。

親トピック: 権限とロール認可の構成

4.4.1 管理権限について

Oracle Databaseではより適切な作業分担を実現するため、一般的に実施される各管理タスク用の管理権限が用意されています。

これらのタスクには、バックアップおよびリカバリ操作、Oracle Data GuardおよびTransparent Data Encryption (TDE)のための暗号化キーの管理などが含まれます。

ユーザーが持つ管理権限は、V$PWFILE_USERS動的ビューを問い合せるとわかります。このビューにはパスワード・ファイル内

のユーザーがリストされます。

以前のリリースでは、これらのタスクを実行するSYSDBA管理権限が必要でした。下位互換性をサポートするには、これらのタスクのSYSDBA権限を引き続き使用できますが、この項で説明する管理権限を使用することをお薦めします。

管理権限を付与されたユーザーを、スキーマ限定アカウントに変更できます。

管理権限の使用は強制的に監査されます。

関連トピック

- 管理ユーザーの監査

親トピック: 管理権限の管理

4.4.2 ユーザーへの管理権限の付与

すべての強力な権限と同様に、管理権限は信頼できるユーザーのみに付与してください。

ただし、名前に非ASCII文字(HÜBERという名前に含まれるウムラウトなど)を使用しているユーザーには制限があります。こうしたユーザーに管理権限を付与することは可能ですが、Oracle Databaseインスタンスが停止した場合、名前に非ASCII文字を使用しているユーザーには、付与されている権限を使用した認証がサポートされません。データベース・インスタンスが稼働中であれば、この認証はサポートされます。

親トピック: 管理権限の管理

4.4.3 標準データベース操作のためのSYSDBAおよびSYSOPER権限

SYSDBAおよびSYSOPER管理権限を使用すると、標準データベース操作を実行できます。

これらのデータベース操作には、データベースの起動および停止、サーバー・パラメータ・ファイル(SPFILE)の作成またはデータベース・アーカイブ・ログの変更などのタスクがあります。マルチテナント環境では、SYSDBAおよびSYSOPER管理権限をアプリケーション共通ユーザーに付与できます(CDB共通ユーザーには付与できません)。

ローカル(PDB)レベル、CDBルート、またはアプリケーション・ルートの管理権限がユーザーに付与されているかどうかを確認するには、V$PWFILE_USERS動的ビューのSCOPE列を問い合せます。

認証なしで作成されたユーザーにSYSDBAおよびSYSOPER管理権限を付与できます。

親トピック: 管理権限の管理

4.4.4 SYSDBAとしてのログイン時のoracleユーザーに対するパスワード入力の強制

ユーザーがSYSDBA管理権限を使用してOracleデータベースにログインするときに、oracleユーザーにパスワードの入力を強制できます。

1. $ORACLE_HOME/network/admin/sqlnet.oraファイルを編集します。
2. SQLNET.AUTHENTICATION_SERVICESパラメータを次のように設定します。

   ```
   sqlnet.authentication_services=none
   ```

   SQLNET.AUTHENTICATION_SERVICESは、設定されていない場合は、デフォルトでALLに設定されます。

親トピック: 管理権限の管理

4.4.5 バックアップおよびリカバリ操作のSYSBACKUP管理権限

Oracle Recovery Manager (RMAN)またはSQL*Plusを使用したバックアップおよびリカバリ操作を実行するには、SYSBACKUP管理権限を使用します。

パスワードを使用してSYSBACKUPとしてデータベースに接続するには、そのパスワード・ファイルを作成する必要があります。パスワード・ファイルの作成の詳細は、『Oracle Database管理者ガイド』を参照してください。

認証なしで作成されたユーザーにSYSBACKUP管理権限を付与できません。

この権限では、次の操作を実行できます。

- STARTUP
- SHUTDOWN
- ALTER DATABASE
- ALTER SYSTEM
- ALTER SESSION
- ALTER TABLESPACE
- CREATE CONTROLFILE
- CREATE ANY DIRECTORY
- CREATE ANY TABLE
- CREATE ANY CLUSTER
- CREATE PFILE
- CREATE RESTORE POINT(GUARANTEEDリストア・ポイントを含む)
- CREATE SESSION
- CREATE SPFILE
- DROP DATABASE
- DROP TABLESPACE
- DROP RESTORE POINT(GUARANTEEDリストア・ポイントを含む)
- FLASHBACK DATABASE
- RESUMABLE
- UNLIMITED TABLESPACE
- SELECT ANY DICTIONARY
- SELECT ANY TRANSACTION
- SELECT
    - X$表(つまり、固定表)
    - V$およびGV$ビュー(つまり、動的パフォーマンス・ビュー)
    - APPQOSSYS.WLM_CLASSIFIER_PLAN
    - SYSTEM.LOGSTDBY$PARAMETERS
- DELETE/INSERT
    - SYS.APPLY$_SOURCE_SCHEMA
    - SYSTEM.LOGSTDBY$PARAMETERS
- EXECUTE

        - SYS.DBMS_BACKUP_RESTORE
        - SYS.DBMS_RCVMAN
        - SYS.DBMS_DATAPUMP
        - SYS.DBMS_IR
        - SYS.DBMS_PIPE
        - SYS.SYS_ERROR
        - SYS.DBMS_TTS
        - SYS.DBMS_TDB
        - SYS.DBMS_PLUGTS
        - SYS.DBMS_PLUGTSP
    - SELECT_CATALOG_ROLE

また、SYSBACKUP権限では、データベースをオープンしていない場合でもデータベースに接続できます。

関連項目:

バックアップおよびリカバリ操作の詳細は、『Oracle Databaseバックアップおよびリカバリ・ユーザーズ・ガイド』を参照してください。

親トピック: 管理権限の管理

4.4.6 Oracle Data Guard操作のSYSDG管理権限

SYSDG管理権限があるSYSDGユーザーとしてログインして、Data Guard操作を実行できます。

Data Guard BrokerまたはDGMGRLコマンドライン・インタフェースでこの権限を使用できます。パスワードを使用してSYSDGとしてデータベースに接続するには、そのパスワード・ファイルを作成する必要があります。

認証なしで作成されたユーザーにSYSYSDG管理権限を付与できません。

SYSDG権限では、次の操作を実行できます。

- STARTUP
- SHUTDOWN
- ALTER DATABASE
- ALTER SESSION
- ALTER SYSTEM
- CREATE RESTORE POINT(GUARANTEEDリストア・ポイントを含む)
- CREATE SESSION
- DROP RESTORE POINT(GUARANTEEDリストア・ポイントを含む)
- FLASHBACK DATABASE
- SELECT ANY DICTIONARY
- SELECT
    - X$表(つまり、固定表)
    - V$およびGV$ビュー(つまり、動的パフォーマンス・ビュー)
    - APPQOSSYS.WLM_CLASSIFIER_PLAN
- DELETE

        - APPQOSSYS.WLM_CLASSIFIER_PLAN
- EXECUTE
    - SYS.DBMS_DRS

また、SYSDG権限では、データベースをオープンしていない場合でもデータベースに接続できます。

関連項目:

- パスワード・ファイルの作成の詳細は、『Oracle Database管理者ガイド』を参照してください。
- Oracle Data Guardの詳細は、Oracle Data Guard概要および管理を参照

親トピック: 管理権限の管理

4.4.7 透過的データ暗号化のSYSKM管理権限

SYSKM管理権限により、SYSKMユーザーは、透過的データ暗号化(TDE)のウォレット操作を管理できます。

パスワードを使用してSYSKMとしてデータベースに接続するには、そのパスワード・ファイルを作成する必要があります。

認証なしで作成されたユーザーにSYSKM管理権限を付与できません。

SYSKM管理権限では、次の操作を実行できます。

- ADMINISTER KEY MANAGEMENT
- CREATE SESSION
- SELECT(データベースをオープンしている場合のみ)
    - SYS.V$ENCRYPTED_TABLESPACES
    - SYS.V$ENCRYPTION_WALLET
    - SYS.V$WALLET
    - SYS.V$ENCRYPTION_KEYS
    - SYS.V$CLIENT_SECRETS
    - SYS.DBA_ENCRYPTION_KEY_USAGE

また、SYSKM権限では、データベースをオープンしていない場合でもデータベースに接続できます。

関連項目:

- パスワード・ファイルの作成の詳細は、『Oracle Database管理者ガイド』を参照してください。
- 透過的データ暗号化の詳細は、『Oracle Database Advanced Securityガイド』を参照してください。

親トピック: 管理権限の管理

4.4.8 Oracle Real Application ClustersのSYSRAC管理権限

SYSRAC管理権限はOracle Real Application Clusters (Oracle RAC)のClusterwareエージェントによって使用されます。

SYSRAC管理権限は、日常的なOracle RAC操作の実行に必要な最小限の権限のみを提供します。たとえば、この権限はSRVCTLなどのOracle RACユーティリティに使用します。

認証なしで作成されたユーザーにSYSRAC管理権限を付与できません。

SYSRAC管理権限では、次の操作を実行できます。

- STARTUP
- SHUTDOWN
- ALTER DATABASE MOUNT
- ALTER DATABASE OPEN
- ALTER DATABASE OPEN READ ONLY
- ALTER DATABASE CLOSE NORMAL
- ALTER DATABASE DISMOUNT
- ALTER SESSION SET EVENTS
- ALTER SESSION SET _NOTIFY_CRS
- ALTER SESSION SET CONTAINER
- ALTER SYSTEM REGISTER
- ALTER SYSTEM SET local_listener|remote_listener|listener_networks

これらの権限に加えて、SYSRACユーザーは次のビューにアクセスできます。

- V$PARAMETER
- V$DATABASE
- V$PDBS
- CDB_SERVICE$
- DBA_SERVICES
- V$ACTIVE_SERVICES
- V$SERVICES

SYSRACユーザーには、次のPL/SQLパッケージのEXECUTE権限が付与されます。

- DBMS_DRS
- DBMS_SERVICE
- DBMS_SERVICE_PRVT
- DBMS_SESSION
- DBMS_HA_ALERTS_PRVT
- メッセージのデキューSYS.SYS$SERVICE_METRICS

関連トピック

- Oracle Real Application Clusters管理およびデプロイメント・ガイド

親トピック: 管理権限の管理

4.5 システム権限の管理

スキーマ・オブジェクトに対するアクションを実行するには、適切なシステム権限が付与されている必要があります。

- システム権限について
  システム権限とは、スキーマ・オブジェクトに対して1つまたは複数の操作を実行する権限です。
- システム権限を制限することが重要な理由

システム権限はかなり強力であるため、信頼できるユーザーのみに権限を付与してください。さらに、データ・ディクショナリおよびSYSスキーマ・オブジェクトを保護する必要があります。

- システム権限の付与と取消し
  システム権限は、ユーザーとロールに対して付与したり、取り消すことができます。
- システム権限を付与したり、取り消すことができるユーザー
  他のユーザーにシステム権限を付与したり、他のユーザーのシステム権限を取り消すことができるのは、次の2つのタイプのユーザーのみです。
- ANY権限とPUBLICロールについて
  ANYキーワードを使用するシステム権限を使用すると、データベース内のオブジェクトのカテゴリ全体に対して権限を設定できます。

親トピック: 権限とロール認可の構成

4.5.1 システム権限について

システム権限とは、スキーマ・オブジェクトに対して1つまたは複数の操作を実行する権限です。

たとえば、表領域を作成する権限や、データベース内の任意の表から行を削除する権限などがシステム権限です。

システム権限には100以上の種類があります。各システム権限によって、ユーザーは特定のデータベース操作、またはあるクラスのデータベース操作を実行できます。システム権限は非常に強力な権限であることに注意してください。システム権限は、必要な場合のみ、データベースのロールと信頼できるユーザーに付与してください。ユーザーに付与されたシステム権限を検索するには、DBA_SYS_PRIVSデータ・ディクショナリ・ビューを問い合せます。

SELECT ANY TABLEなどのシステム権限は、SELECT ANY DICTIONARY権限によって保護されているSYSオブジェクトやその他のオブジェクトでは機能しません。

関連トピック

- 共通に付与されるシステム権限の使用方法
- Oracle Database SQL言語リファレンスGRANT

親トピック: システム権限の管理

4.5.2 システム権限を制限することが重要な理由

システム権限はかなり強力であるため、信頼できるユーザーのみに権限を付与してください。さらに、データ・ディクショナリおよびSYSスキーマ・オブジェクトを保護する必要があります。

- システム権限の制限の重要性について
  システム権限は非常に強力であるため、データベースは、通常のユーザー(管理者以外)がANYシステム権限を行使できないようにデフォルトで構成されています。
- SYSスキーマのオブジェクトへのユーザー・アクセス
  明示的なオブジェクト権限のあるユーザーまたは管理権限で接続しているユーザー(SYSDBA)は、SYSスキーマ内のオブジェクトにアクセスできます。

親トピック: システム権限の管理

4.5.2.1 システム権限の制限の重要性について

システム権限は非常に強力であるため、データベースは、通常のユーザー(管理者以外)がANYシステム権限を行使できないようにデフォルトで構成されています。

たとえば、ユーザーは、データ・ディクショナリに対してUPDATE ANY TABLEなどのANYシステム権限を行使できなくなっています。

関連トピック

- ユーザー・アカウントと権限の保護に関するガイドライン

親トピック: システム権限を制限することが重要な理由

4.5.2.2 SYSスキーマのオブジェクトへのユーザー・アクセス

明示的なオブジェクト権限のあるユーザーまたは管理権限で接続しているユーザー(SYSDBA)は、SYSスキーマ内のオブジェクトにアクセスできます。

次の表に、SYSスキーマ内のオブジェクトへのアクセスが必要なユーザーに対して付与できるロールをリストします。

表4-1 SYSスキーマ・オブジェクトにアクセスできるロール

ロール	説明
SELECT_CATALOG_ROLE	このロールを付与されたユーザーには、データ・ディクショナリ・ビューに対するSELECT 権限が与えられます。
EXECUTE_CATALOG_ROLE	このロールを付与されたユーザーには、データ・ディクショナリ内にあるパッケージとプロシージャに対する EXECUTE 権限が与えられます。

さらにSELECT ANY DICTIONARYシステム権限を、SYSスキーマで作成された表にアクセスが必要なユーザーに付与できます。このシステム権限により、SYSスキーマのあらゆるオブジェクト(そのスキーマに作成された表を含む)への問合せアクセスが可能になります。このシステム権限は、これを必要とする各ユーザーへ個別に付与する必要があります。これはGRANT ALL PRIVILEGESには含まれていませんが、ロールを通じて付与できます。

ノート:

これらのロールおよび SELECT ANY DICTIONARY システム権限は、悪用されるとシステムの整合性が損われる危険があるため、付与する際には十分な注意が必要です。

親トピック: システム権限を制限することが重要な理由

4.5.3 システム権限の付与と取消し

システム権限は、ユーザーとロールに対して付与したり、取り消すことができます。

システム権限をロールに付与すると、そのロールを使用してシステム権限を行使できます。たとえば、ロールを使用すると権限を選択的に使用できるようになります。ロールの保護に関するガイドラインで説明する業務分離のガイドラインに従っていることを確認してください。

ユーザーやロールに対するシステム権限の付与と取消しには、次のいずれかの方法を使用します。

- SQL文のGRANTおよびREVOKE
- Oracle Enterprise Manager Cloud Control

関連トピック

- ユーザー権限およびロールのデータ・ディクショナリ・ビュー

親トピック: システム権限の管理

4.5.4 システム権限を付与したり、取り消すことができるユーザー

他のユーザーにシステム権限を付与したり、他のユーザーのシステム権限を取り消すことができるのは、次の2つのタイプのユーザーのみです。

これらのユーザーは次のとおりです。

- ADMIN OPTIONによって特定のシステム権限を付与されているユーザー
- GRANT ANY PRIVILEGEシステム権限を付与されているユーザー

そのため、これらの権限は信頼できるユーザーにのみ付与してください。

親トピック: システム権限の管理

4.5.5 ANY権限とPUBLICロールについて

ANYキーワードを使用するシステム権限を使用すると、データベース内のオブジェクトのカテゴリ全体に対して権限を設定できます。

たとえば、CREATE ANY PROCEDUREシステム権限により、ユーザーはデータベース内のどこにでもプロシージャを作成可能になります。ANY権限を持つユーザーが作成したオブジェクトの動作は、そのオブジェクトが作成されたスキーマに限定されません。たとえば、ユーザーJSMITHにはCREATE ANY PROCEDURE権限があり、プロシージャをスキーマJONESに作成すると、そのプロシージャはJONESとして実行されることになります。ただし、JONESはJSMITHによって作成されたプロシージャがJONESとして機能していることに気付かない可能性があります。JONESにDBA権限が付与されている場合は、JSMITHがJONESとしてプロシージャを実行することで、セキュリティ違反が発生する可能性があります。

PUBLICロールは、データベース・ユーザー・アカウントが作成されるときすべてのアカウントに自動的に与えられる特別なロールです。デフォルトでは付与されている権限がありませんが、多数の付与があり、ほとんどがJavaオブジェクトに対する付与です。PUBLICロールは削除できません。また、ユーザー・アカウントは常にこのロールを前提とするため、このロールの手動の付与や取消しは意味がありません。PUBLICロールはすべてのデータベース・ユーザー・アカウントが前提とするため、DBA_ROLESおよびSESSION_ROLESデータ・ディクショナリ・ビューには表示されません。

PUBLICロールに権限を付与できますが、付与した権限はOracleデータベースのすべてのユーザーが利用できるようになることに注意してください。このため、PUBLICロールに権限を付与するとき、特にANY権限やシステム権限のように強力な権限の場合は注意してください。たとえば、JSMITHがCREATE PUBLIC SYNONYMシステム権限を持っている場合、他の誰もが使用するとわかっているインタフェースを再定義し、それから自分で作成したPUBLIC SYNONYMでこのインタフェースを指し示すことができます。ユーザーは正しいインタフェースではなくJSMITHのインタフェースにアクセスし、ユーザーのログイン資格証明が盗まれるなど、不正な行為が実行される可能性があります。

この種の権限は非常に強力であるため、不適切な個人に付与するとセキュリティ上のリスクが発生する可能性があります。ANYまたはPUBLICを使用した権限の付与には注意が必要です。他のすべての権限と同様に、これらの権限をユーザーに付与する場合は、「最低限の権限」を付与する原則に従ってください。

関連トピック

- データベースのインストールと構成の保護に関するガイドライン

親トピック: システム権限の管理

4.6 共通およびローカルに付与される権限の管理

マルチテナント環境では、CDB全体またはアプリケーション・コンテナに対する共通の権限を付与するか、または特定のPDBに対してローカルで権限を付与できます。

- 共通およびローカルに付与される権限について
  マルチテナント環境では、共通ユーザーとローカル・ユーザーの両方は、相互に権限を付与できます。
- 共通に付与されるシステム権限の使用方法
  ユーザーは、システム権限が付与されているPDB内でのみシステム権限を実行できます。
- 共通に付与されるオブジェクト権限の使用方法
  共通オブジェクトのオブジェクト権限は、オブジェクト自体とそのオブジェクト上の関連するすべてのリンクに適用されます。
- PDBへのアクセス権限の付与または取消し
  マルチテナント環境では、PDBアクセスに対する権限の付与および取消を実行できます。
- 例: マルチテナント環境での権限の付与
  マルチテナント環境で権限を付与するには、GRANT文を使用できます。
- 共通ユーザーによるCONTAINER_DATAオブジェクトの情報の表示
  共通ユーザーは、ルート内のCONTAINER_DATAオブジェクトや特定のPDB内のデータに関する情報を表示できます。

親トピック: 権限とロール認可の構成

4.6.1 共通およびローカルに付与される権限について

マルチテナント環境では、共通ユーザーとローカル・ユーザーの両方は、相互に権限を付与できます。

権限自体は、共通でもローカルでもありません。権限がどのように適用されるかは、権限が共通に付与されるか、ローカルに付与されるかによって異なります。

共通に付与される権限の場合:

- 共通に付与される権限は、既存および将来のすべてのコンテナで使用できます。
- 権限を共通に付与できるのは共通ユーザーのみで、権限受領者が共通の場合のみです。
- 共通ユーザーは、他の共通ユーザーまたは共通ロールに権限を付与できます。
- 権限付与者は、ルートに接続して、GRANT文のCONTAINER=ALLを指定する必要があります。
- システム権限とオブジェクト権限は、どちらも共通に付与できます。(オブジェクト権限は、指定したオブジェクトに関してのみ実現化します。)
- 共通ユーザーを指定されたコンテナに接続または切り替える場合、様々なアクティビティ(表の作成など)を実行するユーザーの機能は、共通に付与された権限および特定のコンテナでローカルに付与された権限によって制御されます。
- PUBLICには権限を共通に付与しないでください。

ローカルに付与される権限

- ローカルに付与された権限は、それが付与されたコンテナでのみ使用できます。権限がルートで付与されている場合は、ルートにのみ適用されます。
- 共通ユーザーおよびローカル・ユーザーは、どちらも権限をローカルに付与できます。

- 共通ユーザーおよびローカル・ユーザーは、他の共通ロールまたはローカル・ロールに権限を付与できます。
- 権限付与者は、コンテナに接続して、GRANT文のCONTAINER=CURRENTを指定する必要があります。
- ユーザーは、他のユーザーまたはロール(共通およびローカルの両方)あるいはPUBLICロールにローカルに権限を付与できます。

関連トピック

- Oracle Multitenant管理者ガイド
- マルチテナント環境でPUBLICロールが機能するしくみ

親トピック: 共通およびローカルに付与される権限の管理

4.6.2 共通に付与されるシステム権限の使用方法

ユーザーは、システム権限が付与されているPDB内でのみシステム権限を実行できます。

たとえば、PDB B内の共通ユーザーAにシステム権限がローカルに付与されている場合、ユーザーAは、PDB Bに接続されている間のみ、その権限を実行できます。

システム権限は、ルートでのみ適用可能で、次の要件を満たしている場合は、既存および将来のすべてのPDBで適用できます。

- システム権限付与者が共通ユーザーで、権限受領者が共通ユーザー、共通ロールまたはPUBLICロールです。事実上すべてのユーザーがシステム権限を使用できるようになるため、システム権限をPUBLICロールに共通に付与しないでください。
- システム権限付与者は、共通に付与される権限に対してADMIN OPTIONを所有しています。
- GRANT文には、CONTAINER=ALL句を含める必要があります。

次の例は、共通ユーザーc##hr_adminに権限を共通に付与する方法を示しています。

```
CONNECT SYSTEM
Enter password: password
Connected.
GRANT CREATE ANY TABLE TO c##hr_admin CONTAINER=ALL;
```

親トピック: 共通およびローカルに付与される権限の管理

4.6.3 共通に付与されるオブジェクト権限の使用方法

共通オブジェクトのオブジェクト権限は、オブジェクト自体とそのオブジェクト上の関連するすべてのリンクに適用されます。

このリンクには、すべてのメタデータ・リンクおよびデータ・リンク(旧称オブジェクト・リンク)のほか、ルートやコンテナに属するすべてのPDB(将来のPDBを含む)内で特定の要件を満たしたときに関連付けが行われる拡張データ・リンクが含まれます。

この要件を次に示します。

- オブジェクト権限付与者が共通ユーザーで、権限受領者が共通ユーザー、共通ロールまたはPUBLICロールです。
- オブジェクト権限付与者は、権限に対して共通に付与されるGRANT OPTIONを所有しています
- GRANT文には、CONTAINER=ALL句が含まれています。

次の例は、共通ユーザーc##hr_adminにオブジェクト権限を付与して、CDBルートまたは関連付けらているアクセス可能なPDBのいずれかでDBA_PDBSビューから選択できるようにする方法を示しています。

```
CONNECT SYSTEM
```

```
Enter password: password
Connected.
GRANT SELECT ON DBA_OBJECTS TO c##hr_admin
CONTAINER=ALL;
```

関連トピック

- Oracle Multitenant管理者ガイド
- マルチテナント環境でPUBLICロールが機能するしくみ

親トピック: 共通およびローカルに付与される権限の管理

4.6.4 PDBへのアクセス権限の付与または取消し

マルチテナント環境では、PDBアクセスに対する権限の付与および取消を実行できます。

マルチテナント環境で権限を付与するには、次のようにします。

- GRANT文またはREVOKE文にCONTAINER句を含めます。

CONTAINERをALLに設定すると、既存および将来のすべてのコンテナに権限が適用され、CURRENTに設定すると、ローカル・コンテナのみに権限が適用されます。CONTAINER句を省略すると、ローカル・コンテナに権限が適用されます。ルートからGRANT文を発行してCONTAINER句を省略すると、権限がローカルに適用されます。

関連トピック

- Oracle Database SQL言語リファレンス

親トピック: 共通およびローカルに付与される権限の管理

4.6.5 例: マルチテナント環境での権限の付与

マルチテナント環境で権限を付与するには、GRANT文を使用できます。

例4-1は、既存および将来のすべてのコンテナでこの権限を使用できるように、共通ユーザーc##hr_adminにCREATE TABLE権限を共通に付与する方法を示しています。

例4-1 マルチテナント環境での権限の付与

```
CONNECT SYSTEM
Enter password: password
Connected.
GRANT CREATE TABLE TO c##hr_admin CONTAINER=ALL;
```

親トピック: 共通およびローカルに付与される権限の管理

4.6.6 共通ユーザーによるCONTAINER_DATAオブジェクトの情報の表示

共通ユーザーは、ルート内のCONTAINER_DATAオブジェクトや特定のPDB内のデータに関する情報を表示できます。

- ルートに接続中のルート、CDBおよびPDBに関するデータの表示
  共通ユーザーが問合せを実行した場合に、X$表ならびにV$、GV$およびCDB_*ビューの表示情報を制限できます。
- 特定のPDBのデータを問い合せる共通ユーザーの有効化
  特定のPDBに関するデータへのアクセスを共通ユーザーに許可するには、ユーザーのCONTAINER_DATA属性を調整します。

親トピック: 共通およびローカルに付与される権限の管理

4.6.6.1 ルートに接続中のルート、CDBおよびPDBに関するデータの表示

共通ユーザーが問合せを実行した場合に、X$表ならびにV$、GV$およびCDB_*ビューの表示情報を制限できます。

X$表およびこれらのビューには、アプリケーション・ルートおよび関連付けられたアプリケーションPDBに関する情報(CDBルートに接続している場合は、CDB全体に関する情報)が含まれます。

この情報の制限は、他のPDBに関する機密情報を公開しない場合に役立ちます。この機能を有効にするために、Oracle Databaseでは、これらの表およびビューをコンテナ・データ・オブジェクトとして提供します。特定の表またはビューがコンテナ・データ・オブジェクトかどうかを確認するには、USER_|DBA_|ALL_VIEWS|TABLESディクショナリ・ビューのTABLE_NAME、VIEW_NAMEおよびCONTAINER_DATA 列を問い合せます。

デフォルト(ユーザー・レベル)およびオブジェクト固有のCONTAINER_DATA属性に関する情報を検索するには、次のようにします。

1. SQL*PlusまたはSQL Developerで、rootとしてログインします。
2. CDB_CONTAINER_DATAデータ・ディクショナリ・ビューを問い合せます。

   たとえば:

```
COLUMN USERNAME FORMAT A15
COLUMN DEFAULT_ATTR FORMAT A7
COLUMN OWNER FORMAT A15
COLUMN OBJECT_NAME FORMAT A15
COLUMN ALL_CONTAINERS FORMAT A3
COLUMN CONTAINER_NAME FORMAT A10
COLUMN CON_ID FORMAT A6
SELECT USERNAME, DEFAULT_ATTR, OWNER, OBJECT_NAME,
       ALL_CONTAINERS, CONTAINER_NAME, CON_ID
FROM   CDB_CONTAINER_DATA
ORDER BY OBJECT_NAME;
USERNAME        DEFAULT OWNER           OBJECT_NAME     ALL CONTAINERS CON_ID
--------------- ------- --------------- --------------- --- ---------- ------
C##HR_ADMIN     N       SYS             V$SESSION       N   CDB$ROOT        1
C##HR_ADMIN     N       SYS             V$SESSION       N   SALESPDB        1
C##HR_ADMIN     Y                                       N   HRPDB           1
C##HR_ADMIN     Y                                       N   CDB$ROOT        1
DBSNMP          Y                                       Y                   1
SYSTEM          Y                                       Y                   1
```

関連トピック

- Oracle Databaseリファレンス

親トピック: 共通ユーザーによるCONTAINER_DATAオブジェクトの情報の表示

4.6.6.2 特定のPDBのデータを問い合せる共通ユーザーの有効化

特定のPDBに関するデータへのアクセスを共通ユーザーに許可するには、ユーザーのCONTAINER_DATA属性を調整します。

共通ユーザーが特定のPDBについてのデータにアクセスできるようにするには、次のようにします。

- ルートでALTER USER文を発行します。

例4-2 CONTAINER_DATA属性の設定

この例は、ALTER USER文を発行して、共通ユーザーc##hr_adminがV$SESSIONビュー(このユーザーがこのビューを問い合せることができるものと仮定します)のCDB$ROOT、SALES_PDBおよびHRPDBコンテナに関する情報を表示できるようにする方法を示しています。

```
CONNECT SYSTEM
Enter password: password
Connected.
ALTER USER c##hr_admin
SET CONTAINER_DATA = (CDB$ROOT, SALESPDB, HRPDB)
FOR V$SESSION CONTAINER=CURRENT;
```

詳細は、次のとおりです。

- SET CONTAINER_DATAは、コンテナのほか、ユーザーがアクセスできる対象に関するデータをリストします。
- FOR V$SESSIONは、共通ユーザーc##hr_adminが問い合せるCONTAINER_DATA動的ビューを指定します。
- ルートに接続する場合にCONTAINER=ALLがALTER USER文のデフォルトのため、CONTAINER = CURRENTを指定する必要がありますが、CONTAINER_DATA属性の変更はルートに制限する必要があります。

ユーザーc##hr_adminが自身がアクセス可能なすべてのCONTAINER_DATAオブジェクト内のCDB$ROOT、SALES_PDB、HRPDBコンテナに関連する情報を表示できるようにするには、FOR V$SESSIONを省略します。たとえば:

```
ALTER USER c##hr_admin
SET CONTAINER_DATA = (CDB$ROOT, SALESPDB, HRPDB)
CONTAINER=CURRENT;
```

関連トピック

- Oracle Database SQL言語リファレンス

親トピック: 共通ユーザーによるCONTAINER_DATAオブジェクトの情報の表示

4.7 共通ロールおよびローカル・ロールの管理

共通ロールはルートで作成されるロールであり、ローカル・ロールはPDBで作成されます。

- 共通ロールおよびローカル・ロールの管理について
  マルチテナント環境では、データベース・ロールをPDBに固有にすることも、システム・コンテナまたはアプリケーション・コンテナ全体で使用することもできます。
- 共通ロールの使用方法
  共通ロールは、ルートのほか、マルチテナント環境でそれらのロールが定義されているコンテナの各PDBで表示できます。
- マルチテナント環境でPUBLICロールが機能するしくみ
  OracleによってPUBLICロールに付与されるすべての権限はローカルに付与されます。
- 共通ロールの作成、変更または削除に必要な権限
  共通に付与されるCREATE ROLE、ALTER ROLEおよびDROP ROLE権限を持つ共通ユーザーのみが、共通ロールの作成、変更または削除ができます。
- 共通ロールの作成の規則
  共通ロールを作成する場合は、特別な規則に従う必要があります。
- 共通ロールの作成
  CREATE ROLE文を使用して、共通ロールを作成できます。
- ローカル・ロールの作成の規則
  ローカル・ロールを作成するには、特別な規則に従う必要があります。
- ローカル・ロールの作成
  CREATE ROLE文を使用して、ロールを作成できます。
- 共通ユーザーとローカル・ユーザーに対するロールの付与と取消し
  ロールの付与と取消しは、共通ユーザーまたはローカル・ユーザーのアクセス範囲にのみ適用されます。

親トピック: 権限とロール認可の構成

4.7.1 共通ロールおよびローカル・ロールの管理について

マルチテナント環境では、データベース・ロールをPDBに固有にすることも、システム・コンテナまたはアプリケーション・コンテナ全体で使用することもできます。

共通ロールとは、IDと(オプションの)パスワードがコンテナのルートで作成され、ルートのほか、そのコンテナに属する既存および将来のすべてのPDBで認識されるロールです。

ローカル・ロールは、1つのPDBにのみ存在し、このPDB内でのみ使用できます。共通に付与される権限は持ちません。

次のことに注意してください。

- 共通ユーザーは、共通ロールを作成して、他の共通ユーザーおよびローカル・ユーザーに付与できます。
- ロール(ローカルまたは共通)は、ローカル・ユーザーまたはロールに対してローカルに付与できます。
- 共通ロールをローカルに付与する場合、その共通ロールの権限は、ロールが付与されるコンテナ内にのみ適用されます。
- ローカル・ユーザーは共通ロールを作成できませんが、共通ユーザーおよび他のローカル・ユーザーに共通ロールを付与できます。
- 共通ロールをCDBルートまたはアプリケーション・ルートで作成する場合、`CONTAINER = ALL`句がデフォルトです。

関連トピック

- Oracle Databaseのインストールで事前に定義されているロール

親トピック: 共通ロールおよびローカル・ロールの管理

4.7.2 共通ロールの使用方法

共通ロールは、ルートのほか、マルチテナント環境でそれらのロールが定義されているコンテナの各PDBで表示できます。

次の場合、権限は共通ロールに対して共通に付与されます。

- 付与者は、共通ユーザーである。
- 付与者は、付与される権限に対して、共通に付与される`ADMIN OPTION`を所有している。
- `GRANT`文には、`CONTAINER=ALL`句が含まれています。

共通ロールがローカルに付与された権限を含む場合、これらの権限は、共通ロールに付与されたPDB内にのみ適用されます。ローカル・ロールは共通に付与できません。

たとえば、CDB共通ユーザー`c##hr_mgr`に、`DBA`ロールが共通付与されているとします。これは、ユーザー`c##hr_mgr`は、マルチテナント環境のルートおよび各PDBで`DBA`ロールに関連付けられている権限を使用できることを意味します。一方、CDB共通ユーザー`c##hr_mgr`に、`hr_pdb` PDBに対する`DBA`ロールがローカルで付与されているのみであれば、このユーザーは、`hr_pdb` PDBでのみ`DBA`ロールの権限を使用できます。

親トピック: 共通ロールおよびローカル・ロールの管理

4.7.3 マルチテナント環境でPUBLICロールが機能するしくみ

Oracleによって`PUBLIC`ロールに付与されるすべての権限はローカルに付与されます。

この機能により、各PDBで個別に`PUBLIC`ロールに付与された権限およびロールを必要に応じて取り消すことができます。権限

をPUBLICロールに付与する必要がある場合は、ローカルに付与します。PUBLICには権限を共通に付与しないでください。

関連トピック

- 共通およびローカルに付与される権限について

親トピック: 共通ロールおよびローカル・ロールの管理

4.7.4 共通ロールの作成、変更または削除に必要な権限

共通に付与されるCREATE ROLE、ALTER ROLEおよびDROP ROLE権限を持つ共通ユーザーのみが、共通ロールの作成、変更または削除ができます。

共通ユーザーはローカル・ロールも作成できますが、作成されたPDBでのみそれらのロールを使用できます。

親トピック: 共通ロールおよびローカル・ロールの管理

4.7.5 共通ロールの作成の規則

共通ロールを作成する場合は、特別な規則に従う必要があります。

この規則は次のとおりです。

- 正しいルートにいることを確認します。共通ロールを作成するには、正しいルート(CDBルートまたはアプリケーション・ルート)にいる必要があります。PDBから共通ロールを作成することはできません。正しいルートにいることを確認するには、次のいずれかを実行します。
  - CDBルートにいることを確認するには、show_con_nameコマンドを発行します。CDB$ROOTと表示される必要があります。
  - アプリケーション・ルートにいることを確認するには、次の問合せにYESが戻されることを確認します。

    ```
    SELECT APPLICATION_ROOT FROM V$PDBS WHERE CON_ID=SYS_CONTEXT('USERENV',
    'CON_ID');
    ```

  - 共通ロールに付ける名前がCOMMON_USER_PREFIXパラメータの値(デフォルトではC##)で始まるようにします。この要件はDBAやRESOURCEなど、Oracle Databaseによって提供される既存のロールの名前に適用されないことに注意してください。
- オプションで、CONTAINER句をALLに設定します。ルートにいるかぎりは、CONTAINER = ALL句を省略しても、ロールは、デフォルトでCDBルートまたはアプリケーション・ルートの共通ロールとして作成されます。

親トピック: 共通ロールおよびローカル・ロールの管理

4.7.6 共通ロールの作成

CREATE ROLE文を使用して、共通ロールを作成できます。

1. 共通ロールを作成するCDBまたはアプリケーション・コンテナのルートに接続します。

   たとえば:

   ```
   CONNECT SYSTEM
   Enter password: password
   Connected.
   ```

2. CONTAINER句をALLに設定してCREATE ROLE文を実行します。

たとえば:

```
CREATE ROLE c##sec_admin IDENTIFIED BY password CONTAINER=ALL;
```

関連トピック

- ロールの作成
- Enterprise Managerの共通ロールの作成

親トピック: 共通ロールおよびローカル・ロールの管理

4.7.7 ローカル・ロールの作成の規則

ローカル・ロールを作成するには、次の特別な規則に従う必要があります。

これらの規則は次のとおりです。

- ロールを作成するPDBに接続する必要があり、CREATE ROLE権限がある必要があります。
- ローカル・ロールに付ける名前をCOMMON_USER_PREFIXパラメータの値(デフォルトではC##)で始めることはできません。
- CREATE ROLE文にCONTAINER=CURRENTを含め、ローカル・ロールとしてロールを指定できます。PDBに接続しており、この句を省略すると、CONTAINER=CURRENT句が含まれます。
- 共通ロールとローカル・ロールの名前を同じにすることはできません。ただし、異なるPDBのローカル・ロールに同じ名前を使用できます。既存のロールの名前を検索するには、CDB_ROLESおよびDBA_ROLESデータ・ディクショナリ・ビューを問い合せます。

親トピック: 共通ロールおよびローカル・ロールの管理

4.7.8 ローカル・ロールの作成

CREATE ROLE文を使用して、ロールを作成できます。

1. ローカル・ロールを作成するPDBに接続します。

   たとえば:

   ```
   CONNECT SYSTEM@hrpdb
   Enter password: password
   Connected.
   ```

2. CONTAINER句をCURRENTに設定してCREATE ROLE文を実行します。

   たとえば:

   ```
   CREATE ROLE sec_admin CONTAINER=CURRENT;
   ```

親トピック: 共通ロールおよびローカル・ロールの管理

4.7.9 共通ユーザーとローカル・ユーザーに対するロールの付与と取消し

ロールの付与と取消は、共通ユーザーまたはローカル・ユーザーのアクセス範囲にのみ適用されます。

共通ユーザーは、他の共通ユーザーへの共通ロールの付与および取消しを行うことができます。ローカル・ユーザーは、共通ロールを共通ユーザーを含むPDBのユーザーに付与できますが、これは、PDB内のみで適用されます。

次の例は、共通ユーザーc##sec_adminへのすべてのコンテナで使用するAUDIT_ADMIN共通ロールの付与方法を示しています。

```
CONNECT SYSTEM
Enter password: password
Connected.
GRANT AUDIT_ADMIN TO c##sec_admin CONTAINER=ALL;
```

同様に、次の例は、ローカル・ユーザーaud_adminによる共通ユーザーc##sec_adminへのhrpdb PDB内で使用するAUDIT_ADMIN共通ロールの付与方法を示しています。

```
CONNECT aud_admin@hrpdb
Enter password: password
Connected.
GRANT AUDIT_ADMIN TO c##sec_admin CONTAINER=CURRENT;
```

この例は、ローカル・ユーザーaud_adminがPDBの別のユーザーからロールを取り消す方法を示しています。CONTAINER句を省略すると、CURRENTが暗黙のうちに入れられます。

```
CONNECT aud_admin@hrpdb
Enter password: password
Connected.
REVOKE sec_admin FROM psmith CONTAINER=CURRENT;
```

関連トピック

- Enterprise Managerの共通権限付与の取消し

親トピック: 共通ロールおよびローカル・ロールの管理

4.8 ユーザー・ロールの管理

ユーザー・ロールは、作成したり他のユーザーに割り当てることができる権限の名前付きコレクションです。

- ユーザー・ロールについて
  DDLの使用を制限するなど、ユーザー・ロールは様々な目的で利用できます。
- Oracle Databaseのインストールで事前に定義されているロール
  Oracle Databaseには、データベース管理を容易にする一連の事前定義ロールが用意されています。
- ロールの作成
  パスワードの有無に関係なく、認証されるロールを作成できます。外部ロールまたはグローバル・ロールも作成できます。
- ロール認可のタイプの指定
  データベースや外部ソースなどの様々なソースを通じて認可されるように、ロールを構成できます。
- ロールの付与と取消し
  ロールに権限を付与またはロールから権限を取り消して、そのロールをユーザーまたは別のロールに付与できます。
- ロールの削除
  ロールの削除は、そのロールを付与されていたユーザーまたはロールのセキュリティ・ドメインに影響します。
- SQL*Plusユーザーによるデータベース・ロール使用の制限
  SQL*Plusユーザーによるデータベース・ロールの使用を制限することで、侵入者による攻撃からデータベースを保護できます。
- ロール権限およびセキュア・アプリケーション・ロール
  セキュア・アプリケーション・ロールを使用可能にできるのは、認可されたPL/SQLパッケージまたはプロシージャのみです。

親トピック: 権限とロール認可の構成

4.8.1 ユーザー・ロールについて

DDLの使用を制限するなど、ユーザー・ロールは様々な目的で利用できます。

- ユーザー・ロールの概要
  ユーザー・ロールは、ユーザーまたは他のロールに一括で付与できる関連権限の名前付きグループです。
- ロールの機能
  ロールとは、ユーザーに権限を素早く簡単に付与するために便利なものです。
- ロールの特性とそのメリット
  権限管理に要する労力の削減など、ロールにはその管理を容易にする特別なプロパティがあります。
- ロールの通常の使用
  通常は、権限を管理するためにロールを作成します。
- アプリケーション・ロールの一般的な使用方法
  アプリケーション・ロールを使用して、アプリケーションを使用する権限を制御できます。
- ユーザー・ロールの一般的な使用方法
  共通の権限付与要件があるデータベース・ユーザーのグループに対してユーザー・ロールを作成できます。
- ロールがユーザーの権限範囲に与える影響
  各ロールと各ユーザーには、それぞれ独自のセキュリティ・ドメインがあります。
- PL/SQLブロックでのロールの機能
  PL/SQLブロック内でのロールの動作は、ブロックのタイプと定義者権限または実行者権限によって決まります。
- ロールによるDDL使用の支援または制限
  ユーザーがDDL文を正常に実行するには、その文に応じて1つ以上の権限が必要になります。
- オペレーティング・システムによるロールの支援方法
  環境によっては、オペレーティング・システムを使用してデータベース・セキュリティを管理できます。
- 分散環境でのロールの機能
  分散データベース環境では、必要なすべてのロールを分散(リモート)セッションのデフォルト・ロールとして設定する必要があります。

親トピック: ユーザー・ロールの管理

4.8.1.1 ユーザー・ロールの概要

ユーザー・ロールは、ユーザーまたは他のロールに一括で付与できる関連権限の名前付きグループです。

権限の管理および制御は、ロールを使用すると容易になります。

データベース内では、各ロール名を一意にする必要があり、すべてのユーザー名や他のすべてのロール名とは異なる名称にする必要があります。スキーマ・オブジェクトとは異なり、ロールはいずれのスキーマにも含まれません。したがって、ロールを作成するユーザーを、ロールに影響をおよぼすことなく削除できます。

関連トピック

- 共通ロールおよびローカル・ロールの管理

関連項目:

共通ロールおよびローカル・ロールの管理

親トピック: ユーザー・ロールについて

4.8.1.2 ロールの機能

ロールとは、ユーザーに権限を素早く簡単に付与するために便利なものです。

Oracle Databaseで定義されているロールを使用することもできますが、必要な権限のみを含む独自のロールを作成すると、より継続的な制御が可能になります。Oracle Databaseで定義されているロールの権限は、Oracleによって変更または削除される場合があります。

ロールは、次の機能を備えています。

- ロールには、システム権限またはオブジェクト権限を付与できます。
- 任意のロールを任意のデータベース・ユーザーに付与できます。
- ユーザーに付与した各ロールは、任意の時点で使用可能または使用禁止にできます。ユーザーのセキュリティ・ドメインには、そのユーザーに対して現在使用可能になっているすべてのロールの権限が含まれており、ユーザーに対して現在使用禁止になっているロールの権限は除外されています。権限を選択的に使用できるように、Oracle Databaseでは、データベース・アプリケーションとユーザーがロールを使用可能または使用禁止にできます。
- 1つのロールを別のロールにも付与できます。ただし、ロールをそのロール自体に付与したり、循環的に付与することはできません。たとえば、`role2`があらかじめロール`role1`に付与されている場合、ロール`role1`をロール`role2`に付与することはできません。
- ロールがパスワード認証ロールまたはセキュア・アプリケーション・ロールでない場合は、ユーザーに間接的に付与できます。間接的に付与するロールとは、ユーザーにすでに付与されている別のロールを通じて同じユーザーに付与するロールのことです。たとえば、ユーザー`psmith`に`role1`ロールを付与するとします。その後、`role2`ロールと`role3`ロールを`role1`ロールに付与します。これで、`role2`と`role3`の2つのロールは、`role1`に含まれることになります。つまり、`psmith`には、直接付与された`role1`に加え、`role2`ロールと`role3`ロールが間接的に付与されたことになります。`psmith`に対して、直接付与されたロール`role1`を使用可能にすると、間接的に付与されたロール`role2`および`role3`も同様に使用可能になります。
- 必要に応じて、直接付与されたロールをデフォルト・ロールにできます。直接付与されたロールに対してデフォルト・ロール・ステータスを使用可能または使用禁止にするには、`ALTER USER`文の`DEFAULT ROLE`句を使用します。`DEFAULT ROLE`句は、ユーザーに直接付与されたロールのみを示すようにしてください。ユーザーに直接付与されたロールを検索するには、`DBA_ROLE_PRIVS`データ・ディクショナリ・ビューを問い合せます。このビューには、ユーザーに間接的に付与されたロールは含まれません。他のロールに付与されたロールを検索するには、`ROLE_ROLE_PRIVS`ビューを問い合せます。
- ロールがパスワード認証ロールまたはセキュア・アプリケーション・ロールの場合は、ユーザーに間接的に付与することも、デフォルト・ロールにすることもできません。このタイプのロールは、ユーザーに直接付与する必要があります。通常、パスワード認証ロールまたはセキュア・アプリケーション・ロールを使用可能にするには、`SET ROLE`文を使用します。

親トピック: ユーザー・ロールについて

4.8.1.3 ロールの特性とそのメリット

権限管理に要する労力の削減など、ロールにはその管理を容易にする特別なプロパティがあります。

表4-2では、データベース内での権限管理をさらに容易にする、ロールの特性について説明します。

表4-2 ロールの特性とその説明

プロパティ	説明

プロパティ	説明
権限管理に要する労力の削減	複数のユーザーに対して同一の権限セットを明示的に付与するかわりに、関連するユーザー・グループのための権限をまとめて 1 つのロールに付与しておき、そのグループの各メンバーにはそのロールを付与するだけですみます。
動的な権限管理	あるグループの権限を変更する必要がある場合、修正が必要なのは、そのロールの権限のみです。グループのロールを付与した全ユーザーのセキュリティ・ドメインには、そのロールに対して加えられる変更が自動的に反映されます。
権限の選択的な可用性	あるユーザーに付与したロールを、選択的に使用可能または使用禁止にできます。この機能によって、どのような状況でもユーザー権限を個々に制御できます。
アプリケーションによる認識	ロールの存在はデータ・ディクショナリに記録されます。したがって、ユーザーが特定のユーザー名でアプリケーションを実行したときに、アプリケーションがディクショナリに問い合せ、自動的に特定のロールを使用可能(または使用禁止)にするようにアプリケーションを設計できます。
アプリケーション固有のセキュリティ	ロールの使用はパスワードを使用して保護できます。正しいパスワードを入力するとロールが使用可能になるようなアプリケーションを作成できます。パスワードを知らないユーザーは、ロールを使用可能にできません。

データベース管理者は、データベース・アプリケーションのロールを頻繁に作成します。セキュア・アプリケーション・ロールに対して、アプリケーションを実行するために必要なすべての権限を付与する必要があります。それから保護アプリケーション・ロールを他のロールやユーザーに付与できます。1つのアプリケーションは複数の異なるロールを持つことができ、各ロールには異なる権限のセットが付与され、アプリケーション使用中のデータ・アクセスの可否が決定されます。

DBAはパスワード付きのロールを作成することで、そのロールに付与されている権限が無許可で使用されるのを防止できます。通常、アプリケーションは起動時に適切なロールが使用可能になるように設計されます。そのため、アプリケーション・ユーザーは、アプリケーション・ロールのパスワードを知る必要がありません。

関連トピック

- ロールによるDDL使用の支援または制限

親トピック: ユーザー・ロールについて

4.8.1.4 ロールの通常の使用

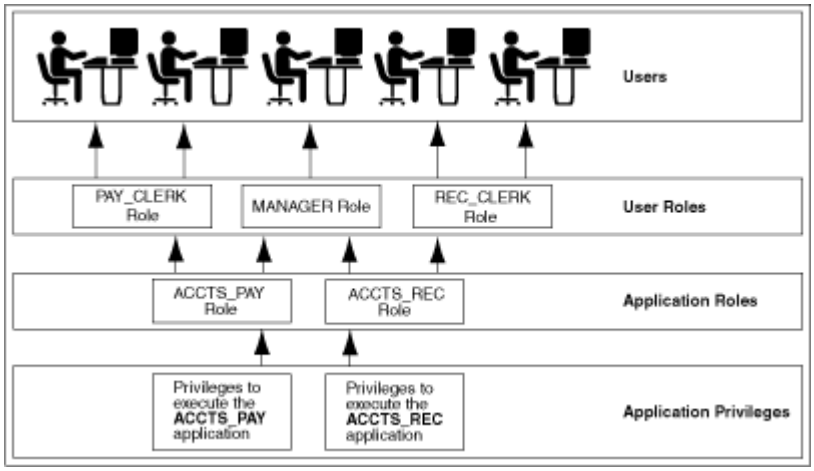
通常は、権限を管理するためにロールを作成します。

理由は次のとおりです。

- データベース・アプリケーションに対する権限の管理
- ユーザー・グループに対する権限の管理

次の図は、ロールの2つの使用方法を示しています。

図4-1 ロールの一般的な使用方法



関連トピック

- アプリケーション・ロールの一般的な使用方法
- ユーザー・ロールの一般的な使用方法

親トピック: ユーザー・ロールについて

4.8.1.5 アプリケーション・ロールの一般的な使用方法

アプリケーション・ロールを使用して、アプリケーションを使用する権限を制御できます。

アプリケーション・ロールには、特定のデータベース・アプリケーションを実行するために必要な権限をすべて付与する必要があります。次に、そのセキュア・アプリケーション・ロールを、他のロールや特定のユーザーに対して付与します。

1つのアプリケーションに対して複数の異なるロールを設定し、アプリケーション使用時のデータ・アクセスの量や範囲にあわせて異なる権限セットを各ロールに割り当てることができます。

親トピック: ユーザー・ロールについて

4.8.1.6 ユーザー・ロールの一般的な使用方法

共通の権限付与要件があるデータベース・ユーザーのグループに対してユーザー・ロールを作成できます。

ユーザーの権限は、セキュア・アプリケーション・ロールと権限をユーザー・ロールに付与し、そのユーザー・ロールを適切なユーザーに付与することで管理できます。

親トピック: ユーザー・ロールについて

4.8.1.7 ロールがユーザーの権限範囲に与える影響

各ロールと各ユーザーには、それぞれ独自のセキュリティ・ドメインがあります。

ロールのセキュリティ・ドメインには、ロール自体に付与されている権限と、そのロールに付与されたロールに対して付与されている権限が含まれます。

ユーザーのセキュリティ・ドメインには、対応するスキーマ内のすべてのスキーマ・オブジェクトに対する権限、ユーザーに付与された権限、そして現在使用可能なユーザーに付与されたロールの権限が含まれます。(ロールをあるユーザーに対して使用可能にすると同時に、他のユーザーに対して使用禁止にできます。)このドメインには、ロールPUBLICに付与された権限とロールも含まれます。PUBLICロールは、データベース内のすべてのユーザーを表します。

親トピック: ユーザー・ロールについて

4.8.1.8 PL/SQLブロックでのロールの機能

PL/SQLブロック内でのロールの動作は、ブロックのタイプと定義者権限または実行者権限によって決まります。

- 定義者権限を持つ名前付きブロックで使用されるロール
  定義者権限で実行される名前付きPL/SQLブロックでは、すべてのロールは使用禁止になっています。
- 実行者権限を持つ名前付きブロックおよび無名PL/SQLブロックで使用されるロール
  実行者権限を使用して実行する名前付きPL/SQLブロックと無名PL/SQLブロックは、使用可能なロールを通じて付与された権限に基づいて実行されます。

親トピック: ユーザー・ロールについて

4.8.1.8.1 定義者権限を持つ名前付きブロックで使用されるロール

定義者権限で実行される名前付きPL/SQLブロックでは、すべてのロールは使用禁止になっています。

名前付きPL/SQLブロックには、ストアド・プロシージャやファンクション、トリガーがあります。

ロールは権限チェックに使用されず、定義者権限プロシージャ内ではロールを設定できません。

PL/SQLブロックが定義者権限で実行する場合、SESSION_ROLESデータ・ディクショナリ・ビューには現在使用可能なすべてのロールが表示されます。定義者権限で実行される名前付きPL/SQLブロックでSESSION_ROLESを問い合せると、その問合せは行を戻しません。

関連項目:

SESSION_ROLESデータ・ディクショナリ・ビューの詳細は、Oracle Databaseリファレンスを参照してください

親トピック: PL/SQLブロックでのロールの機能

4.8.1.8.2 実行者権限を持つ名前付きブロックおよび無名PL/SQLブロックで使用されるロール

実行者権限を使用して実行する名前付きPL/SQLブロックと無名PL/SQLブロックは、使用可能なロールを通じて付与された権限に基づいて実行されます。

現行ロールは、実行者権限PL/SQLブロック内での権限チェックに使用されます。動的SQLを使用して、セッション内にロールを設定できます。

関連項目:

- 実行者権限と定義者権限を使用して名前解決と権限チェックを行う方法については、『Oracle Database PL/SQLパッケージおよびタイプ・リファレンス』を参照してください。
- PL/SQLの動的SQLの詳細は、『Oracle Database PL/SQLパッケージおよびタイプ・リファレンス』を参照してください。

親トピック: PL/SQLブロックでのロールの機能

4.8.1.9 ロールによるDDL使用の支援または制限

ユーザーがDDL文を正常に実行するには、その文に応じて1つ以上の権限が必要になります。

たとえば、表を作成するには、CREATE TABLEまたはCREATE ANY TABLEシステム権限が必要です。

別のユーザーに属する表のビューを作成するには、CREATE VIEWまたはCREATE ANY VIEWシステム権限のみでなく、その

表に対するSELECT object権限またはSELECT ANY TABLEシステム権限も必要です。

Oracle Databaseでは、特定のDDL文での特定の権限の使用を制限することで、ロールを介して受け取った権限への依存性を回避します。次の規則は、DDL文に関する権限の制限を示しています。

- DDL操作の実行をユーザーに許可するすべてのシステム権限およびオブジェクト権限は、ロールを介して受け取った場合でも使用可能。たとえば:
  - システム権限: CREATE TABLE、CREATE VIEWおよびCREATE PROCEDURE権限
  - オブジェクト権限: 表に対するALTERおよびINDEX権限

    表に対するREFERENCESオブジェクト権限は、ロールを介して付与されている場合、表の外部キー定義には使用できません。
- DDL文を発行するために必要なDML操作をユーザーが実行できるようにするすべてのシステム権限とオブジェクト権限は、ロールを通じて受け取った場合には使用できません。CREATE VIEW文が使用されているとき、セキュリティ・ドメインにはロールが含まれません。たとえばユーザーはSELECT ANY TABLEシステム権限を付与されている場合、または表に対するSELECT object権限をロールを通じて付与されている場合、これらの権限のどちらを使用しても他のユーザーに属する表でビューは作成できません。これは、ビューが定義者権限のオブジェクトであり、ビューを作成するとき、自分にロールを通じて付与された権限はいずれも(システム権限もオブジェクト権限も)使用できないためです。権限が直接自分に付与されている場合は、この権限を使用できます。しかし権限が後で取り消されると、ビュー定義は無効になり(エラーとなり)、再度使用する前に再コンパイルする必要があります。

ロールを介して受け取った権限の使用許可と使用制限について、次の例で具体的に説明します。

次のようなユーザーを想定します。

- CREATE VIEWシステム権限を持つロールが付与されています。
- employees表に対するSELECT object権限を持つロールが直接付与されています。
- departments表に対するSELECT object権限が直接付与されています。

前述の権限がこのユーザーに直接的および間接的に付与されているとすると、

- このユーザーはemployees表とdepartments表の両方に対してSELECT文を発行できます。
- このユーザーには、 employees表のCREATE VIEW権限とSELECT権限がロールを介して付与されていますが、employees表のSELECT object権限がロールを介して付与されていたため、employees表のビューは作成できません。
- このユーザーにはCREATE VIEW権限がロールを介して付与され、departments表のSELECT権限が直接付与されているため、departments表のビューは作成できます。

親トピック: ユーザー・ロールについて

4.8.1.10 オペレーティング・システムによるロールの支援方法

環境によっては、オペレーティング・システムを使用してデータベース・セキュリティを管理できます。

オペレーティング・システムを使用して、データベース・ロールの付与と取消し、パスワードの認証を管理できます。この機能は、すべてのオペレーティング・システムで利用できるとはかぎりません。

関連項目:

オペレーティング・システムによるロールの管理方法の詳細は、そのオペレーティング・システム固有のOracle Databaseマニュアルを参照してください。

親トピック: ユーザー・ロールについて

4.8.1.11 分散環境でのロールの機能

分散データベース環境では、必要なすべてのロールを分散(リモート)セッションのデフォルト・ロールとして設定する必要があります。

ローカル・データベース・セッション内からリモート・データベースに接続しているときは、これらのロールを使用可能にすることはできません。たとえば、ユーザーは、リモート・サイトでロールを使用可能にしようとするリモート・プロシージャを実行できません。

関連項目:

Oracle Database Heterogeneous Connectivityユーザーズ・ガイド

親トピック: ユーザー・ロールについて

4.8.2 Oracle Databaseのインストールで事前に定義されているロール

Oracle Databaseには、データベース管理を容易にする一連の事前定義ロールが用意されています。

これらの事前定義ロールは、データベース作成の一部である標準スクリプト(`catalog.sql`や`catproc.sql`など)の実行時に、Oracleデータベースに対して自動的に定義され、共通ロールとみなされます。他のオプションや製品をインストールすると、他の事前定義のロールが作成される場合があります。DBA_ROLESデータ・ディクショナリ・ビューのROLEおよびORACLE_MAINTAINED列を問い合せると、Oracleで作成および管理されているロールを確認できます。ORACLE_MAINTAINEDの出力がYの場合、ロールの作成時に使用したスクリプトを使用する以外の方法でロールを変更しないでください。

表4-3 Oracle Databaseの事前定義ロール

事前定義のロール	説明
ACCHK_READ	アプリケーション・コンティニュイティ保護チェック(ACCHK)を使用する権限を提供します。これには、次のデータ・ディクショナリ・ビューを問い合せる機能が含まれます。 ● DBA_ACCHK_EVENTS ● DBA_ACCHK_EVENTS_SUMMARY ● DBA_ACCHK_STATISTICS ● DBA_ACCHK_STATISTICS_SUMMARY データベース管理者および PDB 管理者は、ACCHK から結果を読み取れるようにこのロールを開発者に付与します。
ADM_PARALLEL_EXECUTE_TASK	DBMS_PARALLEL_EXECUTE PL/SQL パッケージを使用して表のデータをパラレルに更新するための権限を提供します。
AQ_ADMINISTRATOR_ROLE	アドバンスト・キューイングを管理するための権限を提供します。ENQUEUE ANY QUEUE、DEQUEUE ANY QUEUE、MANAGE ANY QUEUE、アドバンスト・キューイングの表に対する SELECT 権限、およびアドバンスト・キューイングのパッケージに

事前定義のロール	説明
	対する EXECUTE 権限が含まれます。
AQ_USER_ROLE	サポートされていませんが、主にリリース 8.0 との互換性のために残されています。DBMS_AQ および DBMS_AQIN パッケージに対する EXECUTE 権限を提供します。
AUDIT_ADMIN	統合およびファイングレイン監査ポリシーの作成、AUDIT および NOAUDIT SQL 文の使用、監査データの表示および監査証跡の管理を行う権限を提供します
AUDIT_VIEWER	監査データを表示および分析する権限を提供します
AUTHENTICATEDUSER	システムにログインしたユーザーを定義するために、XDB プロトコルで使用されます。
CAPTURE_ADMIN	権限分析ポリシーの作成および管理に必要な権限を提供します。
CDB_DBA	SET CONTAINER、SELECT ON PDB_PLUG_IN_VIOLATIONS、SELECT ON CDB_LOCAL_ADMIN_PRIVS などの CDB の管理に必要な権限を提供します。サイトで追加の権限が必要な場合、ロール(共通またはローカル)を作成してこれらの権限に対応し、このロールを CDB_DBA ロールに付与できます。
CONNECT	CREATE SESSION システム権限を提供します。 このロールは、Oracle Database の以前のリリースとの互換性を考慮して用意されています。このロールに組み込まれている権限は、DBA_SYS_PRIVS データ・ディクショナリ・ビューを問い合せることで判別できます。 ノート: このロールに依存せずに、データベース・セキュリティ用に独自のロールを設計することをお薦めします。このロールは、将来のリリースの Oracle Database では自動生成されない可能性があります。
CSW_USR_ROLE	Oracle Spatial の Catalog Services for the Web(CSW)コンポーネントを管理するユーザー権限を提供します。
CTXAPP	Oracle Text の索引および索引プリファレンスの作成権限および PL/SQL パッケージの使用権限を提供します。このロールは Oracle Text ユーザーに付与される必要があります。
CWM_USER	Oracle データ・ウェアハウスおよび意思決定支援で使用されるリポジトリ標準である Common Warehouse Metadata(CWM)の管理権限を提供します。

事前定義のロール	説明
DATAPUMP_EXP_FULL_DATABASE	Oracle Data Pump を使用して Oracle データベースからデータをエクスポートする権限を提供します。 注意: このロールは非常に強力であり、データベース内の任意のスキーマの任意のデータへのユーザー・アクセスを提供します。このロールをユーザーに付与する場合は注意が必要です。
DATAPUMP_IMP_FULL_DATABASE	Oracle Data Pump を使用して Oracle データベースにデータをインポートする権限を提供します。 注意: このロールは非常に強力であり、データベース内の任意のスキーマの任意のデータへのユーザー・アクセスを提供します。このロールをユーザーに付与する場合は注意が必要です。
DBA	ANY 権限(DELETE ANY TABLE など)や GRANT ANY PRIVILEGE 権限など、多数のシステム権限を提供します。 このロールは、Oracle Database の以前のリリースとの互換性を考慮して用意されています。このロールに組み込まれている権限は、DBA_SYS_PRIVS データ・ディクショナリ・ビューを問い合せることで確認できます。 ノート: このロールに依存せずに、データベース・セキュリティ用に独自のロールを設計することをお薦めします。このロールは、将来のリリースの Oracle Database では自動生成されない可能性があります。
DBFS_ROLE	DBFS(データベース・ファイルシステム)パッケージおよびオブジェクトへのアクセスを提供します。
EJBCLIENT	Java ストアド・プロシージャから EJB に接続する権限を提供します。
EM_EXPRESS_ALL	ユーザーは、Oracle Enterprise Manager (EM) Express に接続して、EM Express によって提供されるすべての機能(すべての EM Express 機能への読取りおよび書込みアクセス)を使用できます。EM_EXPRESS_ALL ロールは、EM_EXPRESS_BASIC ロールを含みます。
EM_EXRESS_BASIC	ユーザーは、EM Express に接続して、読取り専用モードでページを表示できます。EM_EXPRESS_BASIC ロールは、SELECT_CATALOG_ROLE ロールを含みます。
EXECUTE_CATALOG_ROLE	データ・ディクショナリ内のオブジェクトに対する EXECUTE 権限を提供します。

事前定義のロール	説明
EXP_FULL_DATABASE	エクスポート・ユーティリティ(後継は Oracle Data Pump)を使用してデータベースの全インポートおよび増分エクスポートを実行するために必要な権限を提供します。含まれる権限は、SELECT ANY TABLE、BACKUP ANY TABLE、EXECUTE ANY PROCEDURE、EXECUTE ANY TYPE、ADMINISTER RESOURCE MANAGER、そして表 SYS.INCVID、SYS.INCFIL、SYS.INCEXP に対する INSERT、DELETE、および UPDATE です。EXECUTE_CATALOG_ROLE および SELECT_CATALOG_ROLE ロールも含みます。 このロールは、エクスポート・ユーティリティおよびインポート・ユーティリティを簡単に使用できるように用意されています。 注意: このロールは非常に強力であり、データベース内の任意のスキーマの任意のデータへのユーザー・アクセスを提供します。このロールをユーザーに付与する場合は注意が必要です。
GATHER_SYSTEM_STATISTICS	DBMS_STATS.GATHER_SYSTEM_STATISTICS プロシージャを使用して収集されるシステム統計の更新権限を提供します。
GLOBAL_AQ_USER_ROLE	Oracle Database Advanced Queuing で使用される LDAP サーバーへの接続を確立する権限を提供します。
HS_ADMIN_EXECUTE_ROLE	異機種間サービス(HS)PL/SQL パッケージの使用を希望するユーザーに対して EXECUTE 権限を提供します。
HS_ADMIN_ROLE	異機種間サービス(HS)PL/SQL パッケージの使用権限および HS 関連のデータ・ディクショナリ・ビューの問合せ権限を提供します。
HS_ADMIN_SELECT_ROLE	異機種間サービスのデータ・ディクショナリ・ビューの問合せ権限を提供します。
IMP_FULL_DATABASE	インポート・ユーティリティ(後継は Oracle Data Pump)を使用してデータベースの全インポートを実行するために必要な権限を提供します。システム権限の詳細リスト(権限を表示するにはビューDBA_SYS_PRIVS を使用)と、ロール EXECUTE_CATALOG_ROLE および SELECT_CATALOG_ROLE が含まれます。 このロールは、エクスポート・ユーティリティおよびインポート・ユーティリティを簡単に使用できるように用意されています。 注意: このロールは非常に強力であり、データベース内の任意のスキーマの任意のデータへのユーザー・アクセスを提供します。このロールをユーザーに付与する場合は

事前定義のロール	説明
	注意が必要です。
JAVADEBUGPRIV	Oracle Database Java アプリケーション・デバッガの実行権限を提供します。
JAVAIDPRIV	このリリースでは非推奨となりました。
JAVASYSPRIV	Oracle JVM で保護されたパッケージの更新を含む、Java 2 を使用するための主要な権限を提供します。
JAVAUSERPRIV	Java 2 を使用するための制限された権限を提供します。
JAVA_ADMIN	Oracle Database Java アプリケーションのポリシー表を更新する管理権限を提供します。
JMXSERVER	データベース・セッションで JMX エージェントを起動およびメンテナンスする権限を提供します。
LBAC_DBA	SA_SYSDBA PL/SQL パッケージの使用権限を提供します。
LOGSTDBY_ADMINISTRATOR	SQL Apply(ロジカル・スタンバイ・データベース)環境を管理する管理権限を提供します。
OEM_ADVISOR	DBMS_SQLTUNE PL/SQL パッケージによって SQL Tuning Set を作成、削除、選択(読取り)、ロード(書込み)および削除する権限と、ADVISOR PL/SQL パッケージを使用してアドバイザ・フレームワークにアクセスする権限を提供します。
OEM_MONITOR	Oracle Enterprise Manager の管理エージェント・コンポーネントで必要とされる、データベースを監視および管理する権限を提供します。
OLAP_DBA	Oracle OLAP の異なるスキーマでディメンション・オブジェクトを作成する管理権限を提供します。
OLAP_USER	アプリケーション開発者に対し、Oracle OLAP の独自のスキーマでディメンション・オブジェクトを作成する権限を提供します。
OLAP_XS_ADMIN	Oracle OLAP のセキュリティの管理権限を提供します。
OPTIMIZER_PROCESSING_RATE	DBMS_STATS パッケージの GATHER_PROCESSING_RATE、SET_PROCESSING_RATE および DELETE_PROCESSING_RATE プロシー

事前定義のロール	説明
	ジャを実行する権限を提供します。これらのプロシージャは、自動的な並列度(Auto DOP)のシステムの処理率を管理します。Auto DOP は、これらの処理率を使用して、SQL 文の最適な並列度を決定します。
PDB_DBA	シード PDB から新しい PDB を作成する場合に作成されるローカル・ユーザーに自動的に付与されます。権限はこのロールに提供されません。
PROVISIONER	Oracle Database Real Application セッションのグローバル・コールバックを登録および更新してプリンシパルをプロビジョニングする権限を提供します。
RECOVERY_CATALOG_OWNER	リカバリ・カタログの所有者の権限を提供します。CREATE SESSION、ALTER SESSION、CREATE SYNONYM、CREATE ANY SYNONYM、DROP ANY SYNONYM、CREATE VIEW、CREATE DATABASE LINK、CREATE TABLE、CREATE CLUSTER、CREATE SEQUENCE、CREATE TRIGGER、CREATE ANY TRIGGER、QUERY REWRITE、CREATE ANY CONTEXT、EXECUTE ON DBMS_RLS、ADMINISTER DATABASE および CREATE PROCEDURE が含まれます。
RESOURCE	CREATE CLUSTER、CREATE INDEXTYPE、CREATE OPERATOR、CREATE PROCEDURE、CREATE SEQUENCE、CREATE TABLE、CREATE TRIGGER、CREATE TYPE の各システム権限を提供します。 RESOURCE が UNLIMITED TABLESPACE システム権限を提供しないことに注意してください。 このロールは、Oracle Database の以前のリリースとの互換性を考慮して用意されています。このロールに組み込まれている権限は、DBA_SYS_PRIVS データ・ディクショナリ・ビューを問い合せることで判別できます。 ノート: このロールに依存せずに、データベース・セキュリティ用に独自のロールを設計することをお薦めします。このロールは、将来のリリースの Oracle Database では自動生成されない可能性があります。
SCHEDULER_ADMIN	権限受領者が DBMS_SCHEDULER パッケージのプロシージャを実行できるようにします。すべてのジョブ・スケジューラ・システム権限が含まれ、DBA ロールに含まれます。
SELECT_CATALOG_ROLE	データ・ディクショナリ内のオブジェクトに対する SELECT 権限を付与します。
SODA_APP	SODA API を使用して、特にドキュメント・コレクションを作成、削除およびリストする

事前定義のロール	説明
	権限を提供します。
WM_ADMIN_ROLE	Oracle Workspace Manager の管理権限を提供します。これにより、ユーザーは DBMS_WM プロシージャをすべてのバージョン対応表、作業領域、およびセーブポイントで、それぞれの所有者に関係なく実行できるようになります。また、ユーザーは Workspace Manager に固有のシステム・パラメータも変更できるようになります。
XDBADMIN	権限受領者が XML スキーマを、その所有者のみが使用したりアクセスするために登録するのではなく、グローバルに登録できるようにします。また権限受領者が Oracle XML DB Repository にアクセスしているときはアクセス制御リスト(ACL)チェックを回避できるようにもします。
XDB_SET_INVOKER	権限受領者が実行者権限ハンドラを定義して、XML リポジトリのトリガーのリソース構成を作成または更新できるようにします。デフォルトでは、このロールは DBA ロールに付与されますが、XDBADMIN ロールには付与されません。
XDB_WEBSERVICES	権限受領者が HTTPS を使用して Oracle Database の Web サービスにアクセスできるようにします。ただし、パブリックなデータベース内のオブジェクトに対するユーザー・アクセスは提供されません。パブリック・アクセスを許可するには、ユーザーに XDB_WEBSERVICES_WITH_PUBLIC ロールを付与する必要があります。ユーザーがこれらの Web サービスを使用するには、SYS で、Web サービスのサーブレットを使用可能にする必要があります。
XDB_WEBSERVICES_OVER_HTTP	権限受領者が HTTP を使用して Oracle Database の Web サービスにアクセスできるようにします。ただし、パブリックなデータベース内のオブジェクトに対するユーザー・アクセスは提供されません。パブリック・アクセスを許可するには、ユーザーに XDB_WEBSERVICES_WITH_PUBLIC ロールを付与する必要があります。
XDB_WEBSERVICES_WITH_PUBLIC	権限受領者が、Oracle Database の Web サービスを介してパブリック・オブジェクトにアクセスできるようにします。
XS_CACHE_ADMIN	Oracle Database Real Application Security では、権限受領者が中間層キャッシュを管理できます。XSAccessController クラスの checkAcl(認可)メソッドの中間層レベルでのセキュリティ・ポリシーのキャッシュに必要です。このロールをアプリケーション接続ユーザーまたは Real Application Security ディスパッチャに付与します。
XS_NSATTR_ADMIN	Oracle Database Real Application Security では、権限受領者がセッションのネームスペースおよび属性を管理および操作できます。このロールを Real

事前定義のロール	説明
	Application Security セッション・ユーザーに付与します。
XS_RESOURCE	Oracle Database Real Application Security では、権限受領者が XS_ACL PL/SQL パッケージを通じて付加されたスキーマのオブジェクトを管理できます。このパッケージは、アクセス制御リスト(ACL)を作成および管理するプロシージャを作成します。ADMIN SEC POLICY 権限を含みます。Oracle Database RESOURCE ロールと似ています。
XS_SESSION_ADMIN	Oracle Database Real Application Security では、権限受領者がセッションを作成、接続、切離しおよび破棄する機能を含むセッションのライフ・サイクルを管理できます。このロールをアプリケーション接続ユーザーまたは Real Application Security ディスパッチャに付与します。

ノート:

各インストールで独自のロールが作成されて、必要な権限のみが割り当てられます。このようにして、使用中の権限の詳細な制御が維持されます。このプロセスにより、Oracle Database で定義されるロールが Oracle Database で変更や削除されたとき、既存のロール、権限、またはプロシージャを調整する必要もなくなります。たとえば、CONNECT ロールが現在持っている権限は CREATE SESSION の 1 つのみです。

親トピック: ユーザー・ロールの管理

4.8.3 ロールの作成

パスワードの有無に関係なく、認証されるロールを作成できます。外部ロールまたはグローバル・ロールも作成できます。

- ロールの作成について
  ロールはCREATE ROLE文を使用して作成できます。
- パスワードを使用して認証されるロールの作成
  IDENTIFIED BY句を使用して、パスワードで認証されるロールを作成できます。
- パスワード認証のないロールの作成
  IDENTIFIED BY句を省略することで、パスワードを必要としないロールを作成できます。
- 外部またはグローバルのロールの作成
  外部ロールまたはグローバル・ロールを使用すると、データベース外部のサービスでデータベース・ロールを認証済ユーザーに関連付けることができます。
- ロールの変更
  ALTER ROLE文で、ロールの認可方法を変更できます。

親トピック: ユーザー・ロールの管理

4.8.3.1 ロールの作成について

ロールはCREATE ROLE文を使用して作成できます。

ロールを作成する場合、CREATE ROLEシステム権限が必要です。通常、このシステム権限はセキュリティ管理者のみが持っ

ています。作成した直後のロールには、権限は対応付けられていません。次のステップで、新しいロールに権限または他のロールを付与します。

作成する各ロールには、データベースの既存のユーザー名やロール名とは異なる、一意の名前を指定する必要があります。ロールはどのユーザーのスキーマ内にも格納されません。マルチバイト文字セットを使用するデータベースでは、各ロール名に少なくとも1つのシングルバイト文字を含めることをお薦めします。ロール名がマルチバイト・キャラクタのみの場合、暗号化されたロール名とパスワードの組合せの安全性は大幅に低くなります。パスワードのガイドラインは、パスワードの保護に関するガイドラインのガイドライン1を参照してください。

IDENTIFIED BY句を使用して、パスワードによってロールを認可します。この句は、このロールを付与された特定ユーザーがロールを使用をする前に、どの認可方式で認可される必要があるかを指定します。この句を指定しないか、またはNOT IDENTIFIEDを指定すると、認可がなくてもロールが使用可能になります。ロールには、必要な認可方式として次の方式を指定できます。

- パスワードを使用したデータベースによる認可
- 指定のパッケージを使用したアプリケーションによる認可
- オペレーティング・システム、ネットワークまたはその他の外部ソースによる外部認可
- エンタープライズ・ディレクトリ・サービスによるグローバル認可

パスワードで保護されたロールの作成の代替手段として、セキュア・アプリケーション・ロールの使用をお薦めします。

ロールの作成に関する次の制限に注意してください。

- ロールとユーザーに同じ名前を付けることはできません。
- このロールがCDB共通ロールでないかぎり、ロール名をCOMMON_USER_PREFIXパラメータの値(デフォルトではC##)で始めることはできません。

関連トピック

- ロール権限およびセキュア・アプリケーション・ロール
- アプリケーションへのアクセスを制御するセキュア・アプリケーション・ロールの作成
- 共通ロールの作成の規則

親トピック: ロールの作成

4.8.3.2 パスワードを使用して認証されるロールの作成

IDENTIFIED BY句を使用して、パスワードで認証されるロールを作成できます。

- パスワード認証されるロールを作成するには、IDENTIFIED BY句が指定されたCREATE ROLE文を使用します。

たとえば:

```
CREATE ROLE clerk IDENTIFIED BY password;
```

ノート:

- パスワードで保護されるロールは、プロキシ・セッションで有効にできます。セキュア・アプリケーション・ロールとパスワードで保護されるロールの両方が、セッションでロールを有効にするための安全な方法を提供します。パスワードを維持して安全でないチャネルで転送する必要がある場合、または複数の人がパスワードを知る必

要がある場合は、パスワードで保護されるロールではなく、セキュアなパスワード・ロールを使用することをお薦めします。プロキシ・セッションでパスワードで保護されるロールは、ロールを設定するために自動化が使用される状況に適しています。

- SQLNET.ALLOWED_LOGON_VERSION_SERVER パラメータを 11 以上に設定する場合、IDENTIFIED BY 句で作成されたロールを再作成する必要があります。

関連トピック

- ロール権限およびセキュア・アプリケーション・ロール
- 安全性の高いロール・パスワードの大/小文字の区別の管理

親トピック: ロールの作成

4.8.3.3 パスワード認証のないロールの作成

IDENTIFIED BY句を省略することで、パスワードを必要としないロールを作成できます。

- 句を指定せずにCREATE ROLE文を使用して、パスワード認証のないロールを作成します。

たとえば:

```
CREATE ROLE salesclerk;
```

親トピック: ロールの作成

4.8.3.4 外部またはグローバルのロールの作成

外部ロールまたはグローバル・ロールを使用すると、データベース外部のサービスでデータベース・ロールを認証済ユーザーに関連付けることができます。

データベース外部ロールは、オペレーティング・システムとRADIUSグループに関連付けられます。このように、データベース・ユーザー認可はデータベースの外部から管理できます。

外部ユーザーは、ロールを使用可能にする前に、オペレーティング・システムやサード・パーティ・サービスなどの外部サービスによって認可されている必要があります。

グローバル・ロールは、集中管理されたユーザーまたはOracle Enterprise User Securityを使用してグローバルに認証されたユーザーによって使用されます。グローバル・ユーザーは、ログイン時にロールの使用が可能になる前に、エンタープライズ・ディレクトリ・サービスによってロールの使用を認可されている必要があります。

- 外部で認可されるロールを作成するには、CREATE ROLE文にIDENTIFIED EXTERNALLY句を含めます。

  たとえば:

  ```
  CREATE ROLE clerk_external IDENTIFIED EXTERNALLY;
  ```

- グローバルに認可されるロールを作成するには、CREATE ROLE文を使用します。

  たとえば:

  ```
  CREATE ROLE clerk_global IDENTIFIED GLOBALLY;
  ```

集中管理されたユーザーなどを含むディレクトリ・サービス・マッピングを介して、ロールをユーザーにグローバルに認可できます。

関連トピック

- オペレーティング・システムまたはネットワークを使用したロールの付与

- RADIUS認証の構成
- グローバル・ロールへのディレクトリ・グループのマッピング
- Oracle Databaseエンタープライズ・ユーザー・セキュリティ管理者ガイド

親トピック: ロールの作成

4.8.3.5 ロールの変更

ALTER ROLE文で、ロールの認可方法を変更できます。

ロールの認可方式を変更するには、ALTER ANY ROLEシステム権限またはADMINオプション付きのロールが付与されている必要があります。

セキュア・アプリケーション・ロールまたはパスワード認証ロールはユーザーに直接付与する必要があることに注意してください。ルートで共通ロールを作成する場合、それをローカル・ロールに変更できないことに注意してください。

- ロールを変更するには、ALTER ROLE文を使用します。

  たとえば、ロールを有効にする前にユーザーが外部ソースによって認可されている必要があるように指定する目的でclerkロールを変更するとします。

  ```
  ALTER ROLE clerk IDENTIFIED EXTERNALLY;
  ```

親トピック: ロールの作成

4.8.4 ロール認可のタイプの指定

データベースや外部ソースなどの様々なソースを通じて認可されるように、ロールを構成できます。

- データベースを使用したロールの認可
  データベースによって認可されたロールを、ロール・パスワードを割り当てることで保護できます。
- アプリケーションを使用したロールの認可
  アプリケーション・ロールを使用可能にできるのは、認可されたPL/SQLパッケージを使用するアプリケーションのみです。
- 外部ソースを使用したロールの認可
  Oracle Databaseでは外部ロールの使用がサポートされますが、一定の制限があります。
- オペレーティング・システムを使用したロールの認可
  Oracle Databaseではオペレーティング・システムを通じたロール認証がサポートされますが、一定の制限があります。
- ネットワーク・クライアントを使用したロールの認可
  Oracle Databaseではネットワーク・クライアントによるロール認証がサポートされますが、一定のセキュリティ・リスクが伴います。
- エンタープライズ・ディレクトリ・サービスによるグローバル・ロールの認可
  グローバル・ロールを使用すると、グローバル・ユーザーの認可の取得先をエンタープライズ・ディレクトリ・サービスに限定できます。

親トピック: ユーザー・ロールの管理

4.8.4.1 データベースを使用したロールの認可

データベースによって認可されたロールを、ロール・パスワードを割り当てることで保護できます。

パスワードで保護されたロールが付与されている場合は、SET ROLE文でそのロールの正しいパスワードを指定することで、そのロールを有効または無効にできます。パスワード認証されるロールは、そのロールがデフォルト・ロールのリストに含まれている場合

でも、ログオン時に認証することはできません。SET ROLE文で必須パスワードを使用して、明示的に使用可能にする必要があります。

1. パスワード認証されるロールを作成するには、IDENTIFIED BY句が指定されたCREATE ROLE文を使用します。

   パスワードを使用して認証されるロールの作成には、clerkというロールを作成するCREATE ROLE文が示されています。このロールを使用可能にするには、パスワードを入力する必要があります。

2. パスワード認証されるロールを設定するには、SET ROLE文を使用します。

   次の例は、SET ROLE文を使用してパスワード認証ロールを設定する方法を示しています。

   ```
   SET ROLE clerk IDENTIFIED BY password;
   ```

   パスワードの保護に関するガイドラインを参照してください。

親トピック: ロール認可のタイプの指定

4.8.4.2 アプリケーションを使用したロールの認可

アプリケーション・ロールを使用可能にできるのは、認可されたPL/SQLパッケージを使用するアプリケーションのみです。

アプリケーション開発者は、アプリケーション内部にパスワードを埋め込むことによってロールを保護する必要はありません。かわりに、アプリケーション・ロール(セキュア・アプリケーション・ロール)を作成して、ロールを使用可能にすることを認可するPL/SQLパッケージを指定できます。

- 認可されたPL/SQLパッケージによって使用可能になるロールを作成するには、CREATE ROLE SQL文で IDENTIFIED USING package_name句を使用します。

たとえば、ロールadmin_roleがアプリケーション・ロールであり、このロールは、PL/SQLパッケージhr.admin内に定義されているモジュールによってのみ使用可能にできることを示すとします。

```
CREATE ROLE admin_role IDENTIFIED USING hr.admin;
```

関連トピック

- ロール権限およびセキュア・アプリケーション・ロール
- アプリケーションへのアクセスを制御するセキュア・アプリケーション・ロールの作成

親トピック: ロール認可のタイプの指定

4.8.4.3 外部ソースを使用したロールの認可

Oracle Databaseでは外部ロールの使用がサポートされますが、一定の制限があります。

外部ロールは、データベースにローカルに定義できますが、グローバル・ユーザー、グローバル・ロールまたはデータベース内の他のロールには付与できません。オペレーティング・システムまたはネットワーク・クライアントによって認可されるロールを作成できます。

- 外部ソースを使用してロールを認可するには、IDENTIFIED EXTERNALLY句を指定してCREATE ROLE文を使用します。

たとえば:

```
CREATE ROLE accts_rec IDENTIFIED EXTERNALLY;
```

親トピック: ロール認可のタイプの指定

4.8.4.4 オペレーティング・システムを使用したロールの認可

Oracle Databaseではオペレーティング・システムを通じたロール認証がサポートされますが、一定の制限があります。

オペレーティング・システムによるロール認証が有効となるのは、そのオペレーティング・システムによって、オペレーティング・システム権限をアプリケーションと動的にリンクできる場合のみです。

ユーザーがアプリケーションを開始すると、オペレーティング・システムはオペレーティング・システム権限をそのユーザーに付与します。付与されたオペレーティング・システム権限は、アプリケーションに対応付けられたロールと一致します。この時点で、アプリケーションはアプリケーション・ロールを使用可能にできます。アプリケーションが終了すると、先に付与されたオペレーティング・システム権限は、そのユーザーのオペレーティング・システム・アカウントから取り消されます。

- ロールをオペレーティング・システムによって認可する場合は、オペレーティング・システム・レベルで各ユーザーの情報を構成します。この操作は、オペレーティング・システムによって異なります。

ロールがオペレーティング・システムによって付与されている場合は、そのオペレーティング・システムによるロールの認可は不要です。

関連トピック

- オペレーティング・システムまたはネットワークを使用したロールの付与

親トピック: ロール認可のタイプの指定

4.8.4.5 ネットワーク・クライアントを使用したロールの認可

Oracle Databaseではネットワーク・クライアントによるロール認証がサポートされますが、一定のセキュリティ・リスクが伴います。

ユーザーがデータベースにOracle Net経由で接続する場合、オペレーティング・システムはデフォルトではユーザーのロールを認証できません。この接続ではOracle Netが必要となるため、共有サーバー構成を介した接続が含まれます。リモート・ユーザーはネットワーク接続を介して別のオペレーティング・システム・ユーザーになりすますおそれがあるため、デフォルトでこのような制限が適用されています。REMOTE_OS_ROLESをFALSE(デフォルト)に設定することをお薦めします。

- このようなセキュリティの危険性の心配がないときに、ネットワーク・クライアントに対してオペレーティング・システムのロール認証を使用する場合は、データベース初期化パラメータ・ファイル内の初期化パラメータREMOTE_OS_ROLESをTRUEに設定します。

この変更は、次回インスタンスを起動して、データベースをマウントするときに有効になります。

親トピック: ロール認可のタイプの指定

4.8.4.6 エンタープライズ・ディレクトリ・サービスによるグローバル・ロールの認可

グローバル・ロールを使用すると、グローバル・ユーザーの認可の取得先をエンタープライズ・ディレクトリ・サービスに限定できます。

グローバル・ロールは、権限とロールを付与することによってデータベース内でローカルに定義しますが、グローバル・ロール自体をそのデータベース内のユーザーや他のロールに付与することはできません。グローバル・ユーザーがデータベースへの接続を試みると、エンタープライズ・ディレクトリへの問合せが実行され、そのユーザーに対応付けられたグローバル・ロールが取得されます。グローバル・ロールは、エンタープライズ・ユーザー・セキュリティの構成要素の1つです。グローバル・ロールは1つのデータベースにのみ適用されますが、エンタープライズ・ディレクトリに定義されたエンタープライズ・ロールに付与できます。エンタープライズ・ロールは複数データベースのグローバル・ロールを含んだディレクトリ構造であり、エンタープライズ・ユーザーに付与できます。

- エンタープライズ・ディレクトリ・サービスによって認可されるグローバル・ロールを作成するには、IDENTIFIED GLOBALLY句を指定してCREATE ROLE文を使用します。

たとえば:

```
CREATE ROLE supervisor IDENTIFIED GLOBALLY;
```

関連項目:

- ユーザーのグローバル認証とグローバル認可、およびエンタープライズ・ユーザー管理におけるロールの概要については、ユーザーのグローバル認証とグローバル認可を参照してください
- エンタープライズ・ユーザー管理の実装の詳細は、『Oracle Databaseエンタープライズ・ユーザー・セキュリティ管理者ガイド』を参照してください。

親トピック: ロール認可のタイプの指定

4.8.5 ロールの付与と取消し

ロールに権限を付与またはロールから権限を取り消して、そのロールをユーザーまたは別のロールに付与できます。

- ロールの付与と取消しについて
  システム権限やオブジェクト権限をロールに付与できます。そしていずれのロールも任意のデータベース・ユーザーや他のロールに付与できます。
- ロールを付与したり、取り消すことができるユーザー
  GRANT ANY ROLEシステム権限を使用して、グローバル・ロール以外の任意のロールを他のユーザーまたはロールに付与したり、それらのロールを取り消したりできます。
- プログラム・ユニットに対するロールの付与と取消し
  関数、プロシージャおよびPL/SQLパッケージ・プログラム・ユニットにロールを付与できます。

親トピック: ユーザー・ロールの管理

4.8.5.1 ロールの付与と取消しについて

システム権限やオブジェクト権限をロールに付与できます。そしていずれのロールも任意のデータベース・ユーザーや他のロールに付与できます。

ただし、ロールを自身に付与したり、循環的に付与することはできません。つまり、ロールYがすでにロールXに付与されている場合は、ロールXをロールYに付与することはできません。

権限を選択的に使用できるように、Oracle Databaseでは、データベース・アプリケーションとユーザーがロールを使用可能または使用禁止にできます。ユーザーに付与した各ロールは、任意の時点で使用可能または使用禁止にできます。ユーザーのセキュリティ・ドメインには、そのユーザーに対して現在使用可能になっているすべてのロールの権限が含まれており、ユーザーに対して現在使用禁止になっているロールの権限は除外されています。

ロールに対して付与されたロールは、間接的に付与されたロールと呼ばれます。この種のロールは、ユーザーに対して明示的に使用可能または使用禁止にできます。ただし、別のロールを含んだロールを使用可能にすると、直接的に付与されたロールに含まれる間接的に付与されたロールは、すべて暗黙のうちに使用可能になります。

GRANT文を使用してロールを付与し、REVOKE文を使用して取り消します。ロールに対して権限を付与または取り消す場合にも同じ文を使用します。

セキュアなロール(つまりIDENTIFIED BYロール、IDENTIFIED USINGロールまたはIDENTIFIED EXTERNALLYロール)は、他のセキュアなロールと非セキュアなロールのどちらにも付与することはできません。SET ROLE文を使用して、セッションに対してセキュアなロールを有効化できます。

親トピック: ロールの付与と取消し

4.8.5.2 ロールを付与したり、取り消すことができるユーザー

GRANT ANY ROLEシステム権限を使用して、グローバル・ロール以外の任意のロールを他のユーザーまたはロールに付与したり、それらのロールを取り消したりできます。

グローバル・ロールはOracle Internet Directoryなどのディレクトリ内で管理されますが、その権限は単一のデータベース内に含まれます。デフォルトでは、ユーザーSYSまたはSYSTEMには、GRANT ANY ROLE権限があります。このシステム権限は非常に強力であるため、付与する場合は控えめに設定する必要があります。

ADMIN OPTION付きでロールを付与されたユーザーは、データベースの他のユーザーやロールに対してロールを付与したり、そのロールを取り消すことができます。つまり、このオプションによって、選択的なロール付与の管理が可能になります。

関連項目:

グローバル・ロールの詳細は、『Oracle Databaseエンタープライズ・ユーザー・セキュリティ管理者ガイド』を参照してください。

親トピック: ロールの付与と取消し

4.8.5.3 プログラム・ユニットに対するロールの付与と取消し

関数、プロシージャおよびPL/SQLパッケージ・プログラム・ユニットにロールを付与できます。

ロールは、プログラム・ユニットの実行中に有効になります。ただし、プログラム・ユニットのコンパイル中を除きます。これにより、ロールを直接ユーザーに付与しないで、PL/SQLコードの権限を一時的にエスカレートできます。また、アプリケーションのセキュリティが強化され、最低限の権限の原則を規定できます。

- プログラム・ユニットに対してロールを付与または取り消すには、GRANTまたはREVOKE文を使用します。

次の例は、PL/SQLパッケージcheckstats_pkgへの同じロールの付与方法を示しています。

```
GRANT clerk_admin TO package psmith.checkstats_pkg;
```

この例は、PL/SQLパッケージcheckstats_pkgのclerk_adminロールの取消し方法を示しています。

```
REVOKE clerk_admin FROM package psmith.checkstats_pkg;
```

次の例は、プロシージャpsmith.check_stats_procへのロールclerk_adminの付与方法を示しています。

```
GRANT clerk_admin TO PROCEDURE psmith.checkstats_proc;
```

関連トピック

- 定義者権限および実行者権限のコード・ベース・アクセス制御の使用

親トピック: ロールの付与と取消し

4.8.6 ロールの削除

ロールの削除は、そのロールを付与されていたユーザーまたはロールのセキュリティ・ドメインに影響します。

つまり、削除したロールを付与されていたすべてのユーザーとロールのセキュリティ・ドメインは、削除したロール権限がなくなったことを反映するために変更されます。

削除したロールによって間接的に付与されていたロールもすべて、関連するセキュリティ・ドメインから削除されます。ロールを削除

することによって、すべてのユーザーのデフォルト・ロール・リストからそのロールが自動的に削除されます。

オブジェクトの存在はロールを介して受け取った権限に依存しないため、ロールが削除されても、表や他のオブジェクトは削除されません。

ロールを削除するには、DROP ANY ROLEシステム権限またはADMINオプション付きのロールが付与されている必要があります。

- ロールを削除するには、DROP ROLE文を使用します。

たとえば、ロールCLERKを削除する方法は次のとおりです。

```
DROP ROLE clerk;
```

親トピック: ユーザー・ロールの管理

4.8.7 SQL*Plusユーザーによるデータベース・ロール使用の制限

SQL*Plusユーザーによるデータベース・ロールの使用を制限することで、侵入者による攻撃からデータベースを保護できます。

- セキュリティに関する潜在的な問題となる非定型ツールの使用
  非定型ツールは、不正なユーザーがこのようなツールにアクセスできると問題が発生する可能性があります。
- PRODUCT_USER_PROFILEシステム表がロールを制限できるしくみ
  SYSTEMスキーマPRODUCT_USER_PROFILE表で、各ユーザーのSQL*Plus環境のSQLおよびSQL*Plusコマンドを無効にできます。
- ストアド・プロシージャがビジネス・ロジックをカプセル化できるしくみ
  ストアド・プロシージャは、ビジネス・ロジックによって権限の使用をカプセル化し、適正なビジネス・トランザクションのコンテキストでのみ権限が実行されるようにします。

親トピック: ユーザー・ロールの管理

4.8.7.1 セキュリティに関する潜在的な問題となる非定型ツールの使用

非定型ツールは、不正なユーザーがこのようなツールにアクセスできると問題が発生する可能性があります。

事前作成データベース・アプリケーションは、アプリケーション使用中にユーザー・ロールを使用可能および使用禁止にすることも含めて、ユーザーのアクションを明示的に制御します。一方、SQL*Plusなどの非定型の問合せツールを使用すると、ユーザーは付与されたロールを使用可能および使用禁止にする文も含めて、あらゆるSQL文を発行できます(正常終了する場合としない場合があります)。

アプリケーションのユーザーは、そのアプリケーションに付加された権限を使用し、非定型ツールによってデータベース表に破壊的なSQL文を発行する恐れがあります。

たとえば、次の状況を想定します。

- Vacation(休暇)アプリケーションには、それに対応するvacationロールがあります。
- このvacationロールには、emp_tab表に対してSELECT、INSERT、UPDATEおよびDELETE文を発行する権限が含まれています。
- Vacationアプリケーションは、vacationロールを介して取得した権限の使用を制御します。

ここで、vacationロールを付与されたユーザーを考えてみます。このユーザーが、Vacationアプリケーションを使用するかわりに、SQL*Plusを実行するとします。この時点でユーザーが制限を受けるのは、明示的に付与された権限またはロール(vacationロールを含む)を介して付与されている権限からのみです。SQL*Plusは非定型の問合せツールであるため、設計されたデータベース・アプリケーションを使用する場合のように、ユーザーは一連の事前定義アクションに制限されることはありませ

ん。ユーザーは、emp_tab表のデータの問合せや変更を自由に実行できます。

親トピック: SQL*Plusユーザーによるデータベース・ロール使用の制限

4.8.7.2 PRODUCT_USER_PROFILEシステム表がロールを制限できるしくみ

SYSTEMスキーマPRODUCT_USER_PROFILE表で、各ユーザーのSQL*Plus環境のSQLおよびSQL*Plusコマンドを無効にできます。

Oracle DatabaseでなくSQL*Plusでこのセキュリティが実行されます。GRANT、REVOKE、およびSET ROLEコマンドへのアクセスを制限して、ユーザーによる各自のデータベース権限の変更を制御することもできます。

PRODUCT_USER_PROFILE表を使用すると、ユーザーがアプリケーションでアクティブにできないロールをリストできます。また、様々なコマンド(SET ROLEなど)の使用を明示的に禁止できます。

たとえば、PRODUCT_USER_PROFILE表にエントリを作成して、次の処理を実行できます。

- SQL*Plusで、clerkおよびmanagerロールの使用を禁止できます。
- SQL*Plusで、SET ROLEの使用を禁止できます。

ユーザーMarlaが、SQL*Plusを使用してデータベースに接続するとします。Marlaには、clerk、managerおよびanalystのロールがあります。前述のPRODUCT_USER_PROFILEのエントリによって、Marlaは、SQL*Plusでanalystロールのみを使用できます。また、GinnyがSET ROLE文を発行しようとすると、SET ROLEの使用を禁止するPRODUCT_USER_PROFILE表のエントリが原因で、発行を明示的に禁止されます。

PRODUCT_USER_PROFILE表は、様々な理由からセキュリティが完全には保証されないことに注意してください。前述の例では、SET ROLEがSQL*Plusで禁止されていますが、Marlaに直接付与されているその他の権限がある場合、MarlaはSQL*Plusを使用してそれらの権限を使用できます。

関連項目:

PRODUCT_USER_PROFILE表の詳細は、『SQL*Plusユーザーズ・ガイドおよびリファレンス』を参照してください。

親トピック: SQL*Plusユーザーによるデータベース・ロール使用の制限

4.8.7.3 ストアド・プロシージャがビジネス・ロジックをカプセル化できるしくみ

ストアド・プロシージャは、ビジネス・ロジックによって権限の使用をカプセル化し、適正なビジネス・トランザクションのコンテキストでのみ権限が実行されるようにします。

たとえば、アプリケーション開発者は、employees表にある従業員の名前および住所を、通常の勤務時間内にのみ更新するプロシージャを作成できます。

また、セキュリティ管理者は、人事部門の担当者にemployees表のUPDATE権限を付与するのではなく、プロシージャのみに権限を付与できます。これによって、人事部門の担当者が権限を使用できるのはプロシージャのコンテキスト内のみになり、employees表を直接更新できなくなります。

親トピック: SQL*Plusユーザーによるデータベース・ロール使用の制限

4.8.8 ロール権限およびセキュア・アプリケーション・ロール

セキュア・アプリケーション・ロールを使用可能にできるのは、認可されたPL/SQLパッケージまたはプロシージャのみです。

PL/SQLパッケージ自体は、アプリケーションへのアクセスを制御するために必要なセキュリティ・ポリシーを反映します。

この方法でロールを作成すると、起動対象アプリケーションに対してこのタイプのロールを使用可能にする場合に制限が加えられます。たとえば、アプリケーションはユーザーがプロキシを介して接続しているかどうかをチェックするなど、認証およびカスタマイズ認可を実行できます。

このタイプのロールでは、パスワードがアプリケーションのソース・コードに埋め込まれたり、表に格納されることがないため、セキュリティが強化されます。このように、データベースが実行する処理は、セキュリティ・ポリシーの実装に基づいており、その定義は1箇所(アプリケーション内ではなくデータベース内)に格納されます。ポリシーの変更が必要な場合は、アプリケーションを修正することなく1箇所の変更で済みます。ポリシーはロールと結び付けられているため、ユーザーがデータベースに接続する方法に関係なく、結果は常に同じです。

セキュア・アプリケーション・ロールを使用可能にするには、ユーザーがセキュア・アプリケーション・ロールによって付与された権限を行使する前のログイン時に、基礎となるパッケージをアプリケーションから直接起動して実行する必要があります。ログイン・トリガーを使用してセキュア・アプリケーション・ロールを使用可能にすることも、このタイプのロールをデフォルト・ロールにすることもできません。

セキュア・アプリケーション・ロールを使用可能にすると、認可されたPL/SQLパッケージがコール・スタックにあることが検証されます。つまり、認可されたPL/SQLパッケージがロールを使用可能にするコマンドを発行しているかどうかが検証されます。

セキュア・アプリケーション・ロールは、データベース接続の存在を保証するために使用できます。セキュア・アプリケーション・ロールはパッケージによって実装されるロールであるため、このパッケージによって、ユーザーが中間層を介して、または特定のIPアドレスからデータベースに接続できることを検証できます。この方法により、セキュア・アプリケーション・ロールは、ユーザーがアプリケーション外のデータにアクセスすることを防ぎます。ユーザーは、付与されているアプリケーション権限のフレームワーク内で作業するように規定されます。

関連トピック

- アプリケーションへのアクセスを制御するセキュア・アプリケーション・ロールの作成

親トピック: ユーザー・ロールの管理

4.9 PDBロックダウン・プロファイルを使用したPDBでの操作の制限

マルチテナント環境でPDBロックダウン・プロファイルを使用して、プラガブル・データベース(PDB)の一連のユーザー操作を制限できます。

この項では、次の項目について説明します。

- PDBロックダウン・プロファイルについて
  PDBロックダウン・プロファイルは、操作グループを制御する名前付きの機能セットです。
- PDBロックダウン・プロファイルの継承
  PDBロックダウン・プロファイルには、CDBルート、アプリケーション・ルート、およびこれらに関連付けられているPDB間の継承動作があります。
- デフォルトのPDBロックダウン・プロファイル
  Oracle Databaseには、サイト要件にあわせてカスタマイズできる一連のデフォルトのPDBロックダウン・プロファイルが用意されています。
- PDBロックダウン・プロファイルの作成
  PDBロックダウン・プロファイルを作成するには、`CREATE LOCKDOWN PROFILE`システム権限が必要です。
- PDBロックダウン・プロファイルの有効化または無効化

PDBロックダウン・プロファイルを有効化または無効化するには、PDB_LOCKDOWN初期化パラメータを使用します。

- PDBロックダウン・プロファイルの削除
  PDBロックダウン・プロファイルを削除するには、DROP LOCKDOWN PROFILEシステム権限があり、CDBルートまたはアプリケーションルートにログインすることが必要です。

親トピック: 権限とロール認可の構成

4.9.1 PDBロックダウン・プロファイルについて

PDBロックダウン・プロファイルは、操作グループを制御する名前付きの機能セットです。

場合によっては、操作を個別に有効または無効にできます。たとえば、PDBロックダウン・プロファイルに、ALTER SYSTEM文で使用する特定の句を無効にする設定を含めることができます。

PDBロックダウン・プロファイルは、ユーザーに対して定義されるリソース制限と同様、機能が提供する機能性へのユーザー・アクセスを制限します。名前が示すように、CDB内でPDBロックダウン・プロファイルを、アプリケーション・コンテナやPDBまたはアプリケーションPDBに対して使用します。カスタム・プロファイルを作成して、サイトの要件に対応することができます。PDBプロファイルを使用すると、アプリケーションのカスタム・セキュリティ・ポリシーを定義できます。さらに、**ベース・プロファイル**と呼ばれる別のプロファイルに基づくロックダウン・プロファイルを作成することができます。このプロファイルは、ベース・プロファイルが更新されたときに動的に更新されるように構成するか、ベース・プロファイルが更新されたときに静的である(変更されない)ように構成できます。ロックダウン・プロファイルは、Oracle Cloudとオンプレミスの両方の環境用に設計されています。

IDがPDB間で共有される場合、昇格する権限が存在することがあります。ロックダウン・プロファイルを使用すると、この権限の昇格を防ぐことができます。IDは、次の状況で共有できます。

- オペレーティング・システム・レベルでは、データベースがファイルやプロセスなどのオペレーティング・システム・リソースと対話するとき
- ネットワーク・レベルでは、データベースが他のシステムと通信し、ネットワークIDが重要なとき
- データベースの内部では、PDBが共通オブジェクトへのアクセスまたは作成を行うとき、またはデータベース・リンクなどの機能を使用してコンテナ境界を越えて通信するとき

共有IDを使用し、PDBロックダウン・プロファイルのメリットを受ける機能は、次のカテゴリに分かれます。

- ネットワーク・アクセス機能。これはネットワークを使用してPDB外部と通信する操作です。たとえば、PL/SQLパッケージUTL_TCP、UTL_HTTP、UTL_MAIL、UTL_SNMP、UTL_INADDRおよびDBMS_DEBUG_JDWPは、この種の操作を実行します。現在、ネットワークIDを共有してこの種のアクセスを制御するためにACLが使用されています。
- 共通ユーザーまたは共通オブジェクトのアクセス。これは、PDBのローカル・ユーザーが共通ユーザー・アカウントを介して通信したり、共通スキーマのオブジェクトにアクセスする操作です。この種の操作には、共通スキーマでのオブジェクトの追加または置換、共通オブジェクトへの権限の付与、共通ディレクトリ・オブジェクトへのアクセス、共通ユーザーへのINHERIT PRIVILEGESロールの付与、および共通ユーザーに対するユーザー・プロキシの操作が含まれます。
- オペレーティング・システム・アクセス。たとえば、UTL_FILEまたはDBMS_FILE_TRANSFER PL/SQLパッケージへのアクセスを制限できます。
- 接続。たとえば、共通ユーザーによるPDBへの接続を制限したり、SYSOPER管理権限を持つローカル・ユーザーによる制限モードでオープンしているPDBへの接続を制限することができます。

PDBロックダウン・プロファイルを作成する一般的な手順では、最初にCREATE LOCKDOWN PROFILE文を使用してプロファイルをCDBルートまたはアプリケーション・ルートに作成し、その後でALTER LOCKDOWN PROFILE文を使用してそれに制限を加えます。

PDBロックダウン・プロファイルを有効にするには、ALTER SYSTEM文を使用して、PDB_LOCKDOWNパラメータを設定します。既存のPDBロックダウン・プロファイルに関する情報を確認するには、CDBルートまたはアプリケーション・ルートに接続してDBA_LOCKDOWN_PROFILESデータ・ディクショナリ・ビューを問い合せます。ローカル・ユーザーは、V$LOCKDOWN_RULES動的データ・ディクショナリ・ビューを問い合せてPDBロックダウン・パラメータの内容を確認できます。

親トピック: PDBロックダウン・プロファイルを使用したPDBでの操作の制限

4.9.2 PDBロックダウン・プロファイルの継承

PDBロックダウン・プロファイルには、CDBルート、アプリケーション・ルート、およびこれらに関連付けられているPDB間の継承動作があります。

- PDBとそれぞれのルートの間の継承パスは次のとおりです。
    - CDB PDBのPDB_LOCKDOWNパラメータ設定は、CDBルートのPDB_LOCKDOWNパラメータ設定より優先されます。同様に、アプリケーションPDBのPDB_LOCKDOWN設定は、アプリケーション・ルートのPDB_LOCKDOWN設定より優先されます。
    - CDB PDB(またはアプリケーションPDB)にPDB_LOCKDOWNパラメータが設定されていない場合、PDBはCDBルート(またはアプリケーション・ルート)のPDB_LOCKDOWNパラメータの設定を継承します。
    - アプリケーション・ルートにPDB_LOCKDOWNパラメータが設定されていない場合、アプリケーション・ルートはCDBルートのPDB_LOCKDOWNパラメータの設定を継承します。
- CDB PDBまたはアプリケーションPDBのPDB_LOCKDOWNパラメータがCDBロックダウン・プロファイルに設定されている場合、PDBはCDBルートまたはアプリケーション・ルートのPDB_LOCKDOWNパラメータによって設定されているすべてのロックダウン・プロファイルを無視します。
- PDBロックダウン・パラメータは、自身に最も近い祖先(つまり、アプリケーション・ルートまたはCDBルート)で設定されたCDBロックダウン・プロファイルから取得された無効なルールを含む、アプリケーション・ロックダウン・プロファイルに規定されたルールを継承できます。これは、アプリケーション・ルートまたはCDBルートのPDB_LOCKDOWNパラメータがCDBロックダウン・プロファイルに設定されているときに、アプリケーションPDBのPDB_LOCKDOWNパラメータがアプリケーション・ロックダウン・プロファイルに設定されている場合に適用されます。
- 場合によっては、CDBロックダウン・プロファイルおよびアプリケーション・ロックダウン・プロファイルを構成するルールの間に競合が発生することがあります。この場合、CDBロックダウン・プロファイルのルールが優先されます。たとえば、CDBロックダウン・プロファイルのOPTION_VALUE句の設定が、アプリケーション・ロックダウン・プロファイルのOPTION_VALUE句の設定よりも優先されます。

親トピック: PDBロックダウン・プロファイルを使用したPDBでの操作の制限

4.9.3 デフォルトのPDBロックダウン・プロファイル

Oracle Databaseには、サイト要件にあわせてカスタマイズできる一連のデフォルトのPDBロックダウン・プロファイルが用意されています。

デフォルトでは、これらのプロファイルのほとんどが空です。これらは、デプロイメント要件に応じて、構成するプレースホルダまたはテンプレートとして設計されています。

これらのプロファイルに関する詳細情報は次のとおりです。

- PRIVATE_DBAASには、プライベート・クラウドDatabase-as-a-Service (DBaaS)のデプロイメントに適した制限が組み込まれています。これらの制限は次のとおりです。

- 各PDBのデータベース管理者は同じである必要があります。
- 別のユーザーがデータベースに接続することが許可されます。
- 別のアプリケーションが許可されます。

PRIVATE_DBAASは、ユーザーがPDBに接続することを許可しますが、ユーザーがOracle Databaseの管理機能を使用できないようにします。

- SAASには、Software-as-a-Service (SaaS)デプロイメントに適した制限が組み込まれています。これらの制限は次のとおりです。
  - 各PDBのデータベース管理者は同じである必要があります。
  - 別のユーザーがデータベースに接続することが許可されます。
  - 同じアプリケーションを使用する必要があります。

  SAASロックダウン・プロファイルには、PRIVATE_DBAASプロファイルよりも厳しい制限があります。ユーザーは異なってもかまいませんが、アプリケーション・コードは同一です。ユーザーは直接接続できず、アプリケーションを使用してのみ接続する必要があります。ユーザーにはすべての管理機能を実行する権限が付与されません。
- PUBLIC_DBAASには、パブリック・クラウドDatabase-as-a-Service (DBaaS)のデプロイメントに適した制限が組み込まれています。制限事項は次のとおりです。
  - PDBごとに異なるDBA
  - 異なるユーザー
  - 異なるアプリケーション

  PUBLIC_DBAASロックダウン・プロファイルは、最も制限が厳しいロックダウン・プロファイルです。

親トピック: PDBロックダウン・プロファイルを使用したPDBでの操作の制限

4.9.4 PDBロックダウン・プロファイルの作成

PDB ロックダウンプロファイルを作成する場合、CREATE LOCKDOWN PROFILEシステム権限が必要です。

作成後のロックダウン・プロファイルには、有効にする前に制限を加えることができます。

1. CREATE LOCKDOWN PROFILEシステム権限を持つユーザーとしてCDBルートまたはアプリケーション・ルートに接続します。
   たとえば、次のようにCDBルートに接続します。

```
CONNECT c##sec_admin
Enter password: password
```

2. 次の構文を使用して、CREATE LOCKDOWN PROFILE文を実行してプロファイルを作成します。

```
CREATE LOCKDOWN PROFILE profile_name
[FROM static_base_profile | INCLUDING dynamic_base_profile];
```

   詳細は、次のとおりです。

   - profile_nameはロックダウン・プロファイルを割り当てる名前です。DBA_LOCKDOWN_PROFILESデータ・ディクショナリ・ビューのPROFILE_NAMES列を問い合せて、既存の名前を確認できます。
   - FROM static_base_profileは、既存のプロファイルからの値を使用して、新しいロックダウン・プロファ

イルを作成します。ベース・プロファイルへのその後の変更は、新しいプロファイルには影響しません。

- INCLUDING dynamic_base_profileも、既存のベース・プロファイルからの値を使用して新しいロックダウン・プロファイルを作成しますが、この新しいロックダウン・プロファイルは、ベース・プロファイルを構成するDISABLE STATEMENTルール、およびベース・プロファイルへのその後の変更を継承するという点が異なります。新しいプロファイルに明示的に追加されたルールがベース・プロファイル内のルールと競合する場合、ベース・プロファイル内のルールが優先されます。たとえば、ベース・プロファイル内のOPTION_VALUE句は、新しいプロファイルのOPTION_VALUE句より優先されます。

次の2つのPDBロックダウン・プロファイル文は、継承がどのように動作するかを示しています。

```
CREATE LOCKDOWN PROFILE hr_prof INCLUDING PRIVATE_DBAAS;
CREATE LOCKDOWN PROFILE hr_prof2 FROM hr_prof;
```

最初の文では、hr_profはPRIVATE_DBAASベース・プロファイルに対して実行されるすべての変更内容を継承します。PRIVATE_DBAASに対して新しい文が有効になった場合、その文はhr_profに対しても有効になります。2番目の文はこれとは対照的に、hr_profが変更された場合、hr_prof2はそのベース・プロファイルに依存しないため変更されません。

3. ALTER LOCKDOWN PROFILE文を実行して、プロファイルに制限を加えます。
   たとえば:

```
ALTER LOCKDOWN PROFILE hr_prof DISABLE STATEMENT  = ('ALTER SYSTEM');
ALTER LOCKDOWN PROFILE hr_prof ENABLE STATEMENT = ('ALTER SYSTEM') clause =
('flush shared_pool');
ALTER LOCKDOWN PROFILE hr_prof DISABLE FEATURE = ('XDB_PROTOCOLS');
```

前述の例は、次のとおりです。

- DISABLE STATEMENT = ('ALTER SYSTEM')は、PDBに対するALTER SYSTEM文の使用をすべて無効にします。
- ENABLE STATEMENT = ('ALTER SYSTEM') clause = ('flush shared_pool')は、ALTER SYSTEMのFLUSH_SHARED_POOL句の使用のみを有効にします。
- DISABLE FEATURE = ('XDB_PROTOCOLS')は、このPDBによるXDBプロトコル(FTP、HTTP、HTTPS)の使用を禁止します

PDBロックダウン・プロファイルの作成後、ALTER SYSTEM SET PDB_LOCKDOWN SQL文を使用してプロファイルを有効にできます。

親トピック: PDBロックダウン・プロファイルを使用したPDBでの操作の制限

4.9.5 PDBロックダウン・プロファイルの有効化または無効化

PDBロックダウン・プロファイルを有効または無効にするには、PDB_LOCKDOWN初期化パラメータを使用します。

ALTER SYSTEM SET PDB_LOCKDOWNを使用して、次のすべてのコンテキストでロックダウン・プロファイルを有効にすることができます。

- CDB (すべてのPDBに影響します)
- アプリケーション・ルート(コンテナ内のすべてのアプリケーションPDBに影響します)
- アプリケーションPDB

- PDB

ノート:

プロファイルを有効にするためにインスタンスを再起動する必要はありません。ALTER SYSTEM SET PDB_LOCKDOWN 文を実行すると、プロファイル・ルールは即時に有効になります。

CDBルートでPDB_LOCKDOWNを設定すると、PDB_LOCKDOWNがコンテナ・レベルで設定されていないかぎり、すべてのPDBおよびアプリケーション・ルートがこの設定を継承します。ロックダウン・プロファイルを無効にするには、PDB_LOCKDOWNをnullに設定します。CDBルートでこのパラメータをnullに設定すると、ロックダウン・プロファイルは、PDB内でプロファイルを明示的に設定したもの以外のすべてのPDBに対して無効になります。

SYSDBA管理権限またはALTER SYSTEMシステム権限が共通付与されたCDB共通ユーザーは、CDBルートで作成されたロックダウン・プロファイルにのみPDB_LOCKDOWNを設定できます。アプリケーション共通のSYSDBA管理権限またはALTER SYSTEMシステム権限を持つアプリケーション共通ユーザーは、アプリケーション・ルートで作成されたロックダウン・プロファイルにのみPDB_LOCKDOWNを設定できます。

1. 共通に付与されるALTER SYSTEMまたは共通に付与されるSYSDBA権限を持つユーザーとして目的のコンテナにログインします。
   たとえば、すべてのPDBのプロファイルを有効化するには、CDBルートにログインします。

   ```
   CONNECT c##sec_admin
   Enter password: password
   ```

2. ALTER SYSTEM SET PDB_LOCKDOWN文を実行します。
   たとえば、次の文は、すべてのPDBについてhr_profという名前のロックダウン・プロファイルを有効にします。

   ```
   ALTER SYSTEM SET PDB_LOCKDOWN = hr_prof;
   ```

   次の文では、PDB_LOCKDOWNパラメータをリセットします。

   ```
   ALTER SYSTEM RESET PDB_LOCKDOWN;
   ```

   前の文を変形した以下の文にはSCOPE句が含まれます。

   ```
   ALTER SYSTEM RESET PDB_LOCKDOWN SCOPE = BOTH;
   ```

   次の文は、PDBレベルで明示的に設定されたものを除く、CDB内のすべてのロックダウン・プロファイルを無効にします。

   ```
   ALTER SYSTEM SET PDB_LOCKDOWN = '' SCOPE = BOTH;
   ```

   PDBロックダウン・プロファイルの名前を確認するには、DBA_LOCKDOWN_PROFILESデータ・ディクショナリ・ビューのPROFILE_NAME列を問い合せます。
3. 必要に応じて、DBA_LOCKDOWN_PROFILESを問い合せてプロファイルに関する情報を確認します。
   たとえば、次の問合せを実行します。

   ```
   SET LINESIZE 150
   COL PROFILE_NAME FORMAT a20
   COL RULE FORMAT a20
   COL CLAUSE FORMAT a25
   SELECT PROFILE_NAME, RULE, CLAUSE, STATUS FROM CDB_LOCKDOWN_PROFILES;
   ```

   出力例は次のように表示されます。

```
PROFILE_NAME         RULE                 CLAUSE                       STATUS
-------------------- -------------------- ---------------------------- -------
HR_PROF              XDB_PROTOCOLS                                     DISABLE
HR_PROF              ALTER SYSTEM                                      DISABLE
HR_PROF              ALTER SYSTEM         FLUSH SHARED_POOL            ENABLE
HR_PROF2                                                               EMPTY
PRIVATE_DBAAS                                                          EMPTY
PUBLIC_DBAAS                                                           EMPTY
SAAS                                                                   EMPTY
```

親トピック: PDBロックダウン・プロファイルを使用したPDBでの操作の制限

4.9.6 PDBロックダウン・プロファイルの削除

PDBロックダウン・プロファイルを削除するには、DROP LOCKDOWN PROFILEシステム権限があり、CDBルートまたはルートにログインすることが必要です。

DBA_LOCKDOWN_PROFILESデータ・ディクショナリ・ビューを問い合せて、既存のPDBロックダウン・プロファイルの名前を検索できます。

1. DROP LOCKDOWN PROFILEシステム権限を持つユーザーとしてCDBルートまたはアプリケーションルートに接続します。
   たとえば、次のようにCDBルートに接続します。

   ```
   CONNECT c##sec_admin
   Enter password: password
   ```

2. DROP LOCKDOWN_PROFILE文を実行します。
   たとえば:

   ```
   DROP LOCKDOWN PROFILE hr_prof2;
   ```

3. 必要に応じて、DBA_LOCKDOWN_PROFILESを問い合せてプロファイルの現在のリストを確認します。
   たとえば、次の問合せを実行します。

   ```
   SET LINESIZE 150
   COL PROFILE_NAME FORMAT a20
   COL RULE FORMAT a20
   COL CLAUSE FORMAT a25
   SELECT PROFILE_NAME, RULE, CLAUSE, STATUS FROM CDB_LOCKDOWN_PROFILES;
   ```

   出力例は次のように表示されます。

   ```
   PROFILE_NAME         RULE                 CLAUSE                       STATUS
   -------------------- -------------------- ---------------------------- -------
   HR_PROF              XDB_PROTOCOLS                                     DISABLE
   HR_PROF              ALTER SYSTEM                                      DISABLE
   HR_PROF              ALTER SYSTEM         FLUSH SHARED_POOL            ENABLE
   PRIVATE_DBAAS                                                          EMPTY
   PUBLIC_DBAAS                                                           EMPTY
   SAAS                                                                   EMPTY
   ```

親トピック: PDBロックダウン・プロファイルを使用したPDBでの操作の制限

4.10 オブジェクト権限の管理

オブジェクト権限を使用すると、表や索引などのスキーマ・オブジェクトに対するアクションを実行できます。

- オブジェクト権限について

オブジェクト権限では、特定のスキーマ・オブジェクトに対して特定のアクションを実行する権限を付与します。

- オブジェクト権限を付与できるユーザー
  ユーザーは、自分のスキーマに含まれているスキーマ・オブジェクトに関しては、すべてのオブジェクト権限が自動的に付与されています。
- オブジェクト権限の付与と取消し
  オブジェクトへの権限の付与およびオブジェクトからの権限の取消しは、ユーザーに直接またはロールを介して行うことができます。
- READオブジェクト権限とSELECTオブジェクト権限
  READおよびSELECT権限で、異なる層の問合せ権限を付与できます。
- シノニムでのオブジェクト権限の使用
  CREATE SYNONYM文で、データベース・オブジェクトのシノニムを作成します。
- アプリケーション共通オブジェクトの共有
  メタデータ・リンク、データ・リンクおよび拡張データ・リンクをアプリケーション・ルートで共有できるように、データベース・オブジェクトを構成できます。

親トピック: 権限とロール認可の構成

4.10.1 オブジェクト権限について

オブジェクト権限では、特定のスキーマ・オブジェクトに対して特定のアクションを実行する権限を付与します。

各タイプのスキーマ・オブジェクトごとに、異なるオブジェクト権限があります。オブジェクト権限の例には、departments表から行を削除する権限があります。

クラスタ、索引、トリガー、データベース・リンクなど、一部のスキーマ・オブジェクトには、対応付けられたオブジェクト権限がありません。これらのオブジェクトの使用は、システム権限によって決定されます。たとえば、クラスタを変更するには、ユーザーはそのクラスタを所有しているか、またはALTER ANY CLUSTERシステム権限が必要です。

たとえば、次のことをする権利がオブジェクト権限です。

- エディションの使用
- 表の更新
- 他のユーザーの表からの行の選択
- 他のユーザーのストアド・プロシージャの実行

関連項目:

- 共通に付与されるオブジェクト権限の使用方法
- オブジェクト権限とそれぞれで許可されている操作のリストは、『Oracle Database SQL言語リファレンス』を参照してください。

親トピック: オブジェクト権限の管理

4.10.2 オブジェクト権限を付与できるユーザー

ユーザーは、自分のスキーマに含まれているスキーマ・オブジェクトに関しては、すべてのオブジェクト権限が自動的に付与されています。

GRANT ANY OBJECT PRIVILEGEシステム権限があるユーザーは、GRANT文のWITH GRANT OPTION句を使用してもしなくても、指定したオブジェクト権限を別のユーザーに付与できます。また、GRANT ANY OBJECT PRIVILEGE権限を持つユーザーは、その権限を使用して、オブジェクト所有者またはGRANT ANY OBJECT PRIVILEGE権限を持つ他のユーザーによって付与されたオブジェクト権限を取り消すことができます。

権限受領者がGRANT ANY OBJECT PRIVILEGE権限を持っていない場合、またはGRANT文のWITH GRANT OPTION句を使用せずに権限を付与された場合、そのユーザーは権限を他のユーザーに付与できません。

WITH GRANT OPTIONは、ユーザーへのオブジェクト権限の付与でのみ使用できます。ロールへのオブジェクト権限の付与では使用できません。

関連項目:

GRANTおよびGRANT ANY OBJECT PRIVILEGEの詳細は、『Oracle Database SQL言語リファレンス』を参照してください。

親トピック: オブジェクト権限の管理

4.10.3 オブジェクト権限の付与と取消し

オブジェクトへの権限の付与およびオブジェクトからの権限の取消しは、ユーザーに直接またはロールを介して行うことができます。

- オブジェクト権限の付与と取消しについて
  オブジェクト権限は、ユーザーとロールに対して付与したり、取り消すことができます。
- ALL句がすべての使用可能なオブジェクト権限を付与または取り消すしくみ
  オブジェクトの各タイプには異なるオブジェクト権限が対応付けられていますが、これらはALL句で制御できます。

親トピック: オブジェクト権限の管理

4.10.3.1 オブジェクト権限の付与と取消しについて

オブジェクト権限は、ユーザーとロールに対して付与したり、取り消すことができます

オブジェクト権限をロールに付与した場合は、その権限を選択的に使用可能にできます。オブジェクト権限を付与するにはGRANT文を使用し、オブジェクト権限を取り消すにはREVOKE文を使用できます。

親トピック: オブジェクト権限の付与と取消し

4.10.3.2 ALL句がすべての使用可能なオブジェクト権限を付与または取り消すしくみ

オブジェクトの各タイプには異なるオブジェクト権限が対応付けられていますが、これらはALL句で制御できます。

ALL [PRIVILEGES]を指定して、オブジェクトに対するすべての使用可能なオブジェクト権限の付与または取消しが可能です。ALLは権限ではありません。ショートカットのようなもので、つまり1つのGRANT文およびREVOKE文ですべてのオブジェクト権限を付与または取り消すための方法です。すべてのオブジェクト権限がALLショートカットを使用して付与された場合でも、権限を個別に取り消すことができます。

同様に、個別に付与した権限をALLを指定してすべて取り消すこともできます。ただし、REVOKE ALLによって整合性制約が削除される場合(整合性制約は取り消そうとしているREFERENCES権限に依存しているため)は、REVOKE文にCASCADE CONSTRAINTSオプションを指定する必要があります。

例4-3では、CASCADE CONSTRAINTSを使用してHRスキーマ内の注文表に対するすべての権限を取り消しています。

例4-3 CASCADE CONSTRAINTSを使用したすべてのオブジェクト権限の取消し

```
REVOKE ALL
 ON ORDERS FROM HR
 CASCADE CONSTRAINTS;
```

親トピック: オブジェクト権限の付与と取消し

4.10.4 READオブジェクト権限とSELECTオブジェクト権限

READおよびSELECT権限で、異なる層の問合せ権限を付与できます。

- READおよびSELECTオブジェクト権限の管理について
  ユーザーにREADまたはSELECTのオブジェクト権限を付与できます。
- データベース内の任意の表を問い合せるためのREADオブジェクト権限の使用のユーザーへの許可
  READ ANY TABLEシステム権限で、データベース内の任意の表を問い合せることができるREADオブジェクト権限を付与します。
- READおよびREAD ANY TABLE権限に対する制限
  READ権限とREAD ANY TABLE権限では特別な制限があります。

親トピック: オブジェクト権限の管理

4.10.4.1 READおよびSELECTオブジェクト権限の管理について

ユーザーにREADまたはSELECTのオブジェクト権限を付与できます。

これらの権限は、ユーザーに許可するアクセス・レベルに応じて付与します。

次のガイドラインに従ってください。

- ユーザーが表、ビュー、マテリアライズド・ビューまたはシノニムの問合せのみをできるようにする場合、READオブジェクト権限を付与する必要があります。たとえば:

  ```
  GRANT READ ON HR.EMPLOYEES TO psmith;
  ```

- ユーザーが、問合せの実行に加えて次のアクションを実行できるようにする場合、ユーザーにSELECTオブジェクト権限を付与する必要があります。
  - LOCK TABLE table_name IN EXCLUSIVE MODE;
  - SELECT ... FROM table_name FOR UPDATE;

  たとえば:

  ```
  GRANT SELECT ON HR.EMPLOYEES TO psmith;
  ```

いずれの場合も、ユーザーpsmithはSELECT文を使用して問合せを実行します。

関連トピック

- READ ANY TABLEおよびSELECT ANY TABLE権限の監査

親トピック: READオブジェクト権限とSELECTオブジェクト権限

4.10.4.2 データベース内の任意の表を問い合せるためのREADオブジェクト権限の使用のユーザーへの許可

READ ANY TABLEシステム権限で、データベース内の任意の表を問い合せることができるREADオブジェクト権限を付与しま

す。

- データベース内の任意の表に対するREADオブジェクト権限をユーザーが保持できるようにするには、そのユーザーにREAD ANY TABLEシステム権限を付与します。

たとえば:

```
GRANT READ ANY TABLE TO psmith;
```

READオブジェクト権限と同様にREAD ANY TABLEシステム権限では、ユーザーは排他モードで表をロックしたり、更新操作用に表を選択したりできません。反対に、SELECT ANY TABLEシステム権限では、ユーザーは任意の表の問合せ以外に、SELECT ... FOR UPDATE文を使用して表の行をロックしたり、表全体をロックできます。

親トピック: READオブジェクト権限とSELECTオブジェクト権限

4.10.4.3 READおよびREAD ANY TABLE権限に対する制限

READ権限とREAD ANY TABLE権限では特別な制限があります。

これらの権限は次のとおりです。

- READオブジェクト権限は、SQL92_SECURITY標準の要件には影響を及ぼしません。SQL92_SECURITY初期化パラメータがTRUEに設定されている場合、UPDATEまたはDELETE文を実行するためにUPDATEまたはDELETE以外にSELECTオブジェクト権限をユーザーに付与する必要があるという要件が緩和されて、SELECTのかわりにREADで十分であるということにはなりません。
- Oracle Database Vaultが有効な場合、SQL92_SECURITY初期化パラメータは自動的にTRUEに設定されることに注意してください。したがって、ユーザーにREADオブジェクト権限またはREAD ANY TABLEシステム権限のみが付与されている場合、UPDATEおよびDELETE文は失敗します。この場合、ユーザーにSELECTオブジェクト権限を付与するか、ユーザーが信頼できるユーザーの場合はSELECT ANY TABLEシステム権限を付与する必要があります。

親トピック: READオブジェクト権限とSELECTオブジェクト権限

4.10.5 シノニムでのオブジェクト権限の使用

CREATE SYNONYM文で、データベース・オブジェクトのシノニムを作成します。

表、ビュー、順序、演算子、プロシージャ、ストアド・ファンクション、パッケージ、マテリアライズド・ビュー、Javaクラス・スキーマ・オブジェクト、ユーザー定義オブジェクト・タイプのオブジェクトのシノニムに加え、別のシノニムのシノニムを作成できます。

ユーザーにシノニムを使用する権限を付与すると、基礎オブジェクトで付与されたオブジェクト権限は、ユーザーがベース・オブジェクトを名前で参照するかシノニムを使用して参照するかに関係なく適用されます。

たとえば、ユーザーOEが次のシノニムをCUSTOMERS表に作成するとします。

```
CREATE SYNONYM customer_syn FOR CUSTOMERS;
```

それから、OEはcustomer_synシノニムに対するREAD権限をユーザーHRに付与します。

```
GRANT READ ON customer_syn TO HR;
```

ユーザーHRは、次のどちらかの問合せを試みます。

```
SELECT COUNT(*) FROM OE.customer_syn;
SELECT COUNT(*) FROM OE.CUSTOMERS;
```

どちらの問合せも同じ結果になります。

```
  COUNT(*)
----------
       319
```

シノニムを他のユーザーに権限付与すると、権限付与はシノニム自体に適用されるのでなく、シノニムが表す基礎オブジェクトに適用されることに注意してください。たとえば、ユーザーHRが自分の権限についてALL_TAB_PRIVSデータ・ディクショナリ・ビューを問い合せると、次のことを知ります。

```
SELECT TABLE_SCHEMA, TABLE_NAME, PRIVILEGE
FROM ALL_TAB_PRIVS
WHERE TABLE_SCHEMA = 'OE';
TABLE_SCHEMA  TABLE_NAME  PRIVILEGE
------------  ----------  ------------------
OE            CUSTOMER    READ
OE            OE          INHERIT PRIVILEGES
```

結果にはこのユーザーが、他の権限に加えて、customer_synシノニムの基礎オブジェクト、つまりOE.CUSTOMER表に対するREAD権限を持っていることが示されます。

この時点でユーザーOEがcustomer_synシノニムに対するREAD権限をHRから取り消した場合に、HRが自分の権限を再度調べたときの結果を次に示します。

```
TABLE_SCHEMA  TABLE_NAME  PRIVILEGE
------------  ----------  ------------------
OE            OE          INHERIT PRIVILEGES
```

ユーザーHRは、OE.CUSTOMER表に対するREAD権限を失っています。このユーザーがOE.CUSTOMERS表を問い合せると、次のエラーが表示されます。

```
SELECT COUNT(*) FROM OE.CUSTOMERS;
ERROR at line 1:
ORA-00942: table or view does not exist
```

親トピック: オブジェクト権限の管理

4.10.6 アプリケーション共通オブジェクトの共有

メタデータ・リンク、データ・リンクおよび拡張データ・リンクをアプリケーション・ルートで共有できるように、データベース・オブジェクトを構成できます。

- メタデータリンク・アプリケーション共通オブジェクト
  メタデータ・リンクを使用すると、アプリケーション・プラガブル・データベース(PDB)のデータベース・オブジェクトとアプリケーション・ルートのオブジェクトとの間でメタデータを共有できます。
- データリンク・アプリケーション共通オブジェクト
  データ・リンクは、マルチテナント環境におけるオブジェクトの参照と権限を管理します。
- 拡張データリンク・アプリケーション共通オブジェクト
  拡張データ・リンクで、アプリケーションのプラガブル・データベース(PDB)とアプリケーション・ルートからのデータを組み合せることができます。

関連項目:

アプリケーション共通オブジェクト(メタデータリンク・オブジェクト、データリンク・オブジェクトおよび拡張データリンク・オブジェクト)の作成の詳細は、Oracle Database管理者ガイドを参照してください

親トピック: オブジェクト権限の管理

4.10.6.1 メタデータリンク・アプリケーション共通オブジェクト

メタデータ・リンクを使用すると、アプリケーション・プラガブル・データベース(PDB)のデータベース・オブジェクトとアプリケーションルートのオブジェクトとの間でメタデータを共有できます。

メタデータ・リンクは、一様に定義されたオブジェクト(オラクル社提供のPL/SQLパッケージなど)について、オブジェクトのメタデータのコピー(PL/SQLパッケージのソース・コードなど)を1つのみ格納するため、ディスクとメモリーの要件を軽減するために役立ちます。これにより、このメタデータへの変更が1つの場所(アプリケーション・ルート)で行われるため、アップグレード操作のパフォーマンスが向上します。

メタデータ・リンクは、アプリケーション・ルートから構成する必要があります。DBMS_PDB.SET_MEDATADATA_LINKED PL/SQLプロシージャを使用すると、データベース・オブジェクトをメタデータ・リンクに変更できます。

次の例は、DBMS_PDB.SET_METADATA_LINKEDプロシージャを使用して、hr_mgrスキーマのupdate_emp_ratingプロシージャをメタデータリンク・アプリケーション共通オブジェクトに変更する方法を示しています。

例4-4 オブジェクトをメタデータリンク・アプリケーション共通オブジェクトに変更する方法

```
BEGIN
  DBMS_PDB.SET_METADATA_LINKED (
   SCHEMA_NAME => 'hr_mgr',
   OBJECT_NAME => 'update_emp_rating',
   NAMESPACE   => 1);
END;
/
```

いずれの共通ユーザーもメタデータ・リンクを所有できます。メタデータ・リンクは、作成者がアプリケーション・ルートで所有するアプリケーション共通オブジェクトのメタデータを共有するためにのみ使用できます。

オブジェクトにメタデータ・リンクがあるかどうかを確認するには、DBA_OBJECTSデータ・ディクショナリ・ビューのSHARING列を問い合せます。

関連項目:

DBMS_PDB.SET_METADATA_LINKEDプロシージャの詳細は、Oracle Database PL/SQLパッケージおよびタイプ・リファレンスを参照してください

親トピック: アプリケーション共通オブジェクトの共有

4.10.6.2 データリンク・アプリケーション共通オブジェクト

データ・リンクは、マルチテナント環境におけるオブジェクトの参照と権限を管理します。

データ・リンク(旧称オブジェクト・リンク)は、同じアプリケーション・コンテナに属するアプリケーション・プラガブル・データベース(PDB)からアプリケーション・ルートのオブジェクトに対する参照および権限付与を可能にします。

アプリケーション共通オブジェクトを所有するアプリケーション共通ユーザーがそのオブジェクトへのアクセス権をPDBのユーザーに付与する場合、アプリケーション共通ユーザーはその共通オブジェクトを指すデータ・リンクへの権限を付与することで、これを行うことができます。たとえば、オブジェクト(表、ビュー、クラスタ、順序またはPL/SQLパッケージなど)のデータ・リンクを作成して、この操作を参照するオブジェクトに対する操作(問合せ、DML、EXECUTE文など)が、操作が実行されるコンテナに関係なく、同じオブジェクトに影響することを確認できます。

データ・リンクは、アプリケーション・ルートから構成する必要があります。DBMS_PDB.SET_DATA_LINKED PL/SQLプロシージャを使用すると、データ・リンクを変更できます。既存のオブジェクトをデータ・リンクに変換する場合にのみ、このプロシージャを使用する必要があります。

次の例は、DBMS_PDB.SET_DATA_LINKEDプロシージャを使用して、hr_mgrスキーマのemp_ratings表をデータリンク・アプリケーション共通オブジェクトに変更する方法を示しています。

例4-5 オブジェクトをデータリンク・アプリケーション共通オブジェクトに変更する方法

```
BEGIN
  DBMS_PDB.SET_DATA_LINKED (
   SCHEMA_NAME => 'hr_mgr',
   OBJECT_NAME => 'emp_ratings',
   NAMESPACE   => 1);
END;
/
```

いずれの共通ユーザーもデータ・リンクを所有できます。

オブジェクトにデータ・リンクがあるかどうかを確認するには、DBA_OBJECTSデータ・ディクショナリ・ビューのSHARING列を問い合せます。このビューのNAMESPACE列は、ネームスペースの数値を示します。

関連項目:

DBMS_PDB.SET_DATA_LINKEDプロシージャの詳細は、Oracle Database PL/SQLパッケージおよびタイプ・リファレンスを参照してください

親トピック: アプリケーション共通オブジェクトの共有

4.10.6.3 拡張データリンク・アプリケーション共通オブジェクト

拡張データ・リンクで、アプリケーションのプラガブル・データベース(PDB)とアプリケーション・ルートからのデータを組み合せることができます。

拡張データ・リンクは、PDB内の表で見つかったデータと、アプリケーション・ルートの対応する表からのデータを組み合せることができるデータ・リンクです。

拡張データ・リンクは、メタデータ・リンクとデータ・リンクのハイブリッドと考えることができます。アプリケーションPDB内の拡張データリンク・オブジェクトは、アプリケーション・ルート内の拡張データ・リンク・オブジェクトからメタデータを継承します。オブジェクトのデータはアプリケーション・ルートに格納され、オプションで各アプリケーションPDBに格納されます。拡張データ・リンクは、表およびビューに対してのみ作成できます。拡張データ・リンク・オブジェクトのDBA_OBJECTSデータ・ディクショナリ・ビューを問い合せると、このビューは、アプリケーションPDBとアプリケーション・ルートの両方から拡張データ・リンク関連の行を戻します。

拡張データ・リンクは、アプリケーション・ルートから構成する必要があります。DBMS_PDB.SET_EXT_DATA_LINKED PL/SQLプロシージャを使用すると、データベース・オブジェクトを拡張データ・リンクに変更できます。

次の例は、DBMS_PDB.SET_EXT_DATA_LINKEDプロシージャを使用して、hr_mgrスキーマのemp_salariesデータ・ディクショナリ・ビューを拡張データリンク・アプリケーション共通オブジェクトに変更する方法を示しています。

例4-6 オブジェクトを拡張データリンク・アプリケーション共通オブジェクトに変更する方法

```
BEGIN
  DBMS_PDB.SET_EXT_DATA_LINKED (
   SCHEMA_NAME => 'hr_mgr',
   OBJECT_NAME => 'emp_salaries',
   NAMESPACE   => 1);
```

```
END;
/
```

いずれの共通ユーザーも拡張データ・リンクを所有できます。

オブジェクトに拡張データ・リンクがあるかどうかを確認するには、DBA_OBJECTSデータ・ディクショナリ・ビューのSHARING列を問い合せます。

関連項目:

DBMS_PDB.SET_EXT_DATA_LINKEDプロシージャの詳細は、Oracle Database PL/SQLパッケージおよびタイプ・リファレンスを参照してください

親トピック: アプリケーション共通オブジェクトの共有

4.11 表権限

表に対するオブジェクト権限は、DMLまたはDDLレベルの操作に対する表セキュリティを実現します。

- 表に対する権限がデータ操作言語操作に与える影響
  表およびビューでDELETE、INSERT、SELECTおよびUPDATEの各DML操作を使用する権限を付与できます。
- 表に対する権限がデータ定義言語操作に与える影響
  ALTER、INDEXおよびREFERENCESの各権限は、表に対するDDL操作の実行を許可します。

親トピック: 権限とロール認可の構成

4.11.1 表に対する権限がデータ操作言語操作に与える影響

表およびビューでDELETE、INSERT、SELECTおよびUPDATEの各DML操作を使用する権限を付与できます。

これらの権限は、表のデータの問合せや操作が必要なユーザーとロールに対してのみ付与してください。

表に対するINSERT権限とUPDATE権限は、表の特定の列に制限できます。選択的なINSERT権限を付与されたユーザーは、選択した列に値を持つ行を挿入できます。他のすべての列には、NULLまたはその列のデフォルト値が挿入されます。選択的なUPDATE権限によって、ユーザーは行の特定の列に限ってその値を更新できます。機密データに対するユーザー・アクセスを制限するには、INSERT権限とUPDATE権限を選択的に使用します。

たとえば、データ入力ユーザーにemployees表のsalary列を変更させないようにするには、そのsalary列を除外した選択的なINSERT権限またはUPDATE権限を付与できます。また、salary列を除外したビューによって、同じ制限をさらに高いセキュリティ・レベルで実現できます。

関連項目:

DML操作の詳細は、『Oracle Database SQL言語リファレンス』を参照してください。

親トピック: 表権限

4.11.2 表に対する権限がデータ定義言語操作に与える影響

ALTER、INDEXおよびREFERENCESの各権限は、表に対するDDL操作の実行を許可します。

これらの権限によって、他のユーザーは表への依存性を変更または作成できるため、権限の付与は控えめに行う必要がありま

す。表に対してDDL操作を実行するユーザーには、さらに他のシステム権限やオブジェクト権限が必要な場合があります。たとえば、表にトリガーを作成するには、その表に対するALTER TABLEオブジェクト権限とCREATE TRIGGERシステム権限の両方が必要です。

INSERT権限やUPDATE権限と同様に、REFERENCES権限は、表の特定の列を対象として付与できます。REFERENCES権限を付与されたユーザーは、付与の対象となった表を、自分の表の中に作成する外部キーの親キーとして使用できます。外部キーの存在によって、親キーに対して実行できるデータ操作と表の変更が制限されるため、このアクションは特殊な権限によって制御されます。列固有のREFERENCES権限によって、権限受領者が使用できるのは、指定された列(この列には、当然、親表の主キーまたは一意キーが最低1つ含まれている)に制限されます。

関連項目:

主キー、一意キーおよび整合性制約によるデータ整合性の仕組みの詳細は、Oracle Database概要を参照してください。

親トピック: 表権限

4.12 ビューに対する権限

DMLオブジェクト権限は、表の場合と同様にビューに対しても適用できます。

- ビューの作成に必要な権限
  ビューを作成するには、特定の権限が必要です。
- 他のスキーマのビューを問い合せるための権限
  ビューが配置されているスキーマとは異なるスキーマからユーザーがビューを問い合せるには、ビューの実表に対するSELECT WITH GRANT OPTIONがビュー所有者に付与されている必要があります。
- ビューの使用による表セキュリティの強化
  データベース・ビューでは、ユーザーが参照できるデータを制限することによって表セキュリティを強化できます。

親トピック: 権限とロール認可の構成

4.12.1 ビューの作成に必要な権限

ビューを作成するには、特定の権限が必要です。

ビューに対するオブジェクト権限は、ビューの導出元の実表に影響を与える様々なDML操作を許可します。

ビューを作成する権限は次のとおりです。

- 次のどちらかのシステム権限が、明示的に、またはロールを介して付与されている必要があります。
  - CREATE VIEWシステム権限(自分のスキーマ内にビューを作成するため)
  - CREATE ANY VIEWシステム権限(別のユーザーのスキーマ内にビューを作成するため)
- 次のいずれかの権限が明示的に付与されている必要があります。
  - ビューの基礎となるすべてのベース・オブジェクトに対するSELECT、INSERT、UPDATEまたはDELETEオブジェクト権限
  - SELECT ANY TABLE、INSERT ANY TABLE、UPDATE ANY TABLEまたはDELETE ANY TABLEシステム権限
- さらに、自分のビューへのアクセス権を他のユーザーに付与するためには、ベース・オブジェクトに対するGRANT

OPTION句付きのオブジェクト権限、またはADMIN OPTION句付きの適切なシステム権限を持っている必要があります。これらの権限を持っていない場合は、自分のビューに対するアクセス権を他のユーザーに付与できません。試行すると、ORA-01720「object_nameに対するGRANTオプションは存在しません。」エラーが発生します。object_nameはビューの基礎となるオブジェクトを参照しており、ユーザーにはこのオブジェクトに対する十分な権限がありません。

関連項目:

Oracle Database SQL言語リファレンス

親トピック: ビューに対する権限

4.12.2 他のスキーマのビューを問い合せるための権限

ビューが配置されているスキーマとは異なるスキーマからユーザーがビューを問い合せるには、ビューの実表に対するSELECT WITH GRANT OPTIONがビュー所有者に付与されている必要があります。

親トピック: ビューに対する権限

4.12.3 ビューの使用による表セキュリティの強化

データベース・ビューでは、ユーザーが参照できるデータを制限することによって表セキュリティを強化できます。

ユーザーがビューを使用するには、ビュー自体に対する適切な権限のみが必要であり、ビューの基礎となるオブジェクトに対する権限は不要です。ただし、ビューの基礎となるオブジェクトに対するアクセス権限が削除されると、ユーザーはアクセスできなくなります。

このように動作するのは、ユーザーがビューを問い合せるときに使用されるセキュリティ・ドメインが、そのビューの定義者のセキュリティ・ドメインであるためです。基礎となるオブジェクトに対する権限がビューの定義者によって取り消されると、そのビューは無効になり、いかなるユーザーも使用できなくなります。したがって、ビューに対するアクセス権が付与されているユーザーでも、ビューの基礎となるオブジェクトに対する定義者権限が取り消された場合は、そのビューを使用できません。

たとえば、ユーザーAがビューを作成するとします。ユーザーAは、ビューの基礎となるオブジェクトに対する定義者権限を持っています。この場合、ユーザーAがそのビューに対するSELECT権限をユーザーBに付与すると、ユーザーBがそのビューの問合せを実行できるようになります。ただし、ユーザーAがそのビューの基礎となるオブジェクトにアクセスできなくなった場合は、ユーザーBも同様にアクセスできなくなります。

ビューの場合は、次のように、表に対してさらに2つのセキュリティ・レベル(列レベルと値ベースのセキュリティ)が追加されます。

- ビューは、実表の中から選択した列へのアクセスを提供できます。たとえば、employees表の中からemployee_id列、last_name列およびmanager_id列のみを表示するようにビューを定義できます。

```
CREATE VIEW employees_manager AS
    SELECT last_name, employee_id, manager_id FROM employees;
```

- ビューでは、表の情報に対して、値ベースのセキュリティを実現できます。ビュー定義でWHERE句を使用すると、実表の中から選択した行のみが表示されます。次の2つの例を考えてみます。

```
CREATE VIEW lowsal AS
    SELECT * FROM employees
    WHERE salary < 10000;
```

lowsalビューでは、employees表のうち給与値が10000未満のすべての行にアクセスできます。lowsalビューで

は、employees表のすべての列にアクセスできることに注目してください。

```
CREATE VIEW own_salary AS
    SELECT last_name, salary
    FROM employees
    WHERE last_name = USER;
```

own_salaryビューでは、このビューの現行ユーザーと一致するlast_nameの行のみにアクセスできます。own_salaryビューはuser疑似列を使用しており、この列の値は常に現行ユーザーを指します。このビューは、列レベルのセキュリティと値ベースのセキュリティの両方を兼ね備えています。

親トピック: ビューに対する権限

4.13 プロシージャ権限

EXECUTE権限は、スタンドアロンまたはパッケージ内でのプロシージャまたは関数の実行をユーザーに許可します。

- プロシージャ権限に対するEXECUTE権限の使用
  EXECUTE権限は、注意して処理する必要がある非常に強力な権限です。
- プロシージャの実行とセキュリティ・ドメイン
  プロシージャに対するEXECUTEオブジェクト権限を使用して、そのプロシージャを実行したり、それを参照するプログラム・ユニットをコンパイルできます。
- プロシージャの作成または置換に必要なシステム権限
  自分のスキーマ内または別のユーザーのスキーマ内でプロシージャを作成または置換するには、特定の権限が必要です。
- プロシージャのコンパイルに必要なシステム権限
  スタンドアロン・プロシージャとパッケージの一部であるプロシージャの両方をコンパイルするには、特定の権限が必要です。
- プロシージャに対する権限がパッケージおよびパッケージ・オブジェクトに与える影響
  強力な権限であるEXECUTE権限は、ユーザーがパッケージ内のPUBLICプロシージャやファンクションを実行することを可能にします。

親トピック: 権限とロール認可の構成

4.13.1 プロシージャ権限に対するEXECUTE権限の使用

EXECUTE権限は、注意して処理する必要がある非常に強力な権限です。

スタンドアロン・プロシージャ、ファクションおよびパッケージを含め、プロシージャに対するオブジェクト権限は、EXECUTE権限のみです。

この権限は、プロシージャの実行、または必要なプロシージャをコールする他のプロシージャのコンパイルが必要なユーザーにのみ付与してください。ユーザーに付与された権限を確認するには、DBA_SYS_PRIVSデータ・ディクショナリ・ビューに問い合せます。

親トピック: プロシージャ権限

4.13.2 プロシージャの実行とセキュリティ・ドメイン

プロシージャに対するEXECUTEオブジェクト権限を使用して、そのプロシージャを実行したり、それを参照するプログラム・ユニットをコンパイルできます。

PL/SQLユニットのコール時には、Oracle Databaseによって実行時権限チェックが実行されます。EXECUTE ANY PROCEDUREシステム権限があるユーザーは、データベース内の任意のプロシージャを実行できます。プロシージャの実行権限は、

ロールを介してユーザーに付与できます。

関連項目:

- 定義者権限および実行者権限について
- Oracle Databaseの実行時権限チェックの詳細は、『Oracle Database PL/SQLパッケージおよびタイプ・リファレンス』を参照してください。

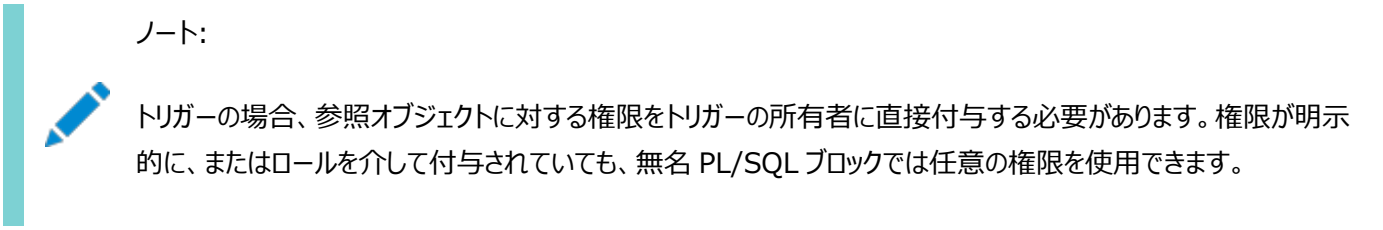
親トピック: プロシージャ権限

4.13.3 プロシージャの作成または置換に必要なシステム権限

自分のスキーマ内または別のユーザーのスキーマ内でプロシージャを作成または置換するには、特定の権限が必要です。

自分のスキーマ内でプロシージャを作成または置換するには、CREATE PROCEDUREシステム権限が必要です。別のユーザーのスキーマ内でプロシージャを作成または置換するには、CREATE ANY PROCEDUREシステム権限が必要です。

プロシージャを所有するユーザーには、プロシージャ本体で参照されるスキーマ・オブジェクトに対する権限も必要です。プロシージャを作成するには、そのプロシージャによって参照されるすべてのオブジェクトに対する必要な権限(システム権限やオブジェクト権限)が明示的に付与されている必要があります。これらの必要な権限は、ロールを介して取得することはできません。これには、作成中のプロシージャ内でコールするプロシージャに対するEXECUTE権限も含まれます。



ノート:

トリガーの場合、参照オブジェクトに対する権限をトリガーの所有者に直接付与する必要があります。権限が明示的に、またはロールを介して付与されていても、無名 PL/SQL ブロックでは任意の権限を使用できます。

親トピック: プロシージャ権限

4.13.4 プロシージャのコンパイルに必要なシステム権限

スタンドアロン・プロシージャとパッケージの一部であるプロシージャの両方をコンパイルするには、特定の権限が必要です。

スタンドアロン・プロシージャをコンパイルするには、COMPILE句を使用してALTER PROCEDURE文を実行する必要があります。パッケージの一部であるプロシージャをコンパイルするには、ALTER PACKAGE文を実行する必要があります。

次の例は、スタンドアロン・プロシージャをコンパイルする方法を示しています。

```
ALTER PROCEDURE psmith.remove_emp COMPILE;
```

スタンドアロン・プロシージャまたはパッケージ・プロシージャが別のユーザーのスキーマ内にある場合、プロシージャを再コンパイルするにはALTER ANY PROCEDURE権限が必要です。自分のスキーマ内にあるプロシージャは、権限なしで再コンパイルできます。

親トピック: プロシージャ権限

4.13.5 プロシージャに対する権限がパッケージおよびパッケージ・オブジェクトに与える影響

強力な権限であるEXECUTE権限は、ユーザーがパッケージ内のPUBLICプロシージャやファンクションを実行することを可能にし

ます。

- プロシージャに対する権限がパッケージおよびパッケージ・オブジェクトに与える影響について
  パッケージに対するEXECUTEオブジェクト権限は、そのパッケージ内のすべてのプロシージャまたはファンクションに適用されます。
- 例: 1つのパッケージ内で使用されるプロシージャ権限
  CREATE PACKAGE BODY文を使用してプロシージャを含むパッケージ本体を作成し、1つのパッケージ内で使用されるプロシージャ権限を管理できます。
- 例: プロシージャ権限およびパッケージ・オブジェクト
  CREATE PACKAGE BODY文でプロシージャ定義を含むパッケージ本体を作成し、プロシージャ権限とパッケージ・オブジェクトを管理できます。

親トピック: プロシージャ権限

4.13.5.1 プロシージャに対する権限がパッケージおよびパッケージ・オブジェクトに与える影響について

パッケージに対するEXECUTEオブジェクト権限は、そのパッケージ内のすべてのプロシージャまたはファンクションに適用されます。

パッケージに対するEXECUTEオブジェクト権限を保持するユーザーは、そのパッケージ内の任意のパブリック・プロシージャまたはファンクションを実行し、任意のパブリック・パッケージ変数の値へのアクセスや変更を実行できます。

パッケージの各構成メンバーに、特定のEXECUTE権限を付与することはできません。したがって、データベース・アプリケーションのプロシージャ、ファンクションおよびパッケージを開発する場合は、セキュリティの確立に関して2つの選択肢を考慮してください。これらの選択肢について、次の例で説明します。

親トピック: プロシージャに対する権限がパッケージおよびパッケージ・オブジェクトに与える影響

4.13.5.2 例: 1つのパッケージ内で使用されるプロシージャ権限

CREATE PACKAGE BODY文を使用してプロシージャを含むパッケージ本体を作成し、1つのパッケージ内で使用されるプロシージャ権限を管理できます。

例4-7では、2つのパッケージの本体に4つのプロシージャを作成しています。

例4-7 1つのパッケージ内で使用されるプロシージャ権限

```
CREATE PACKAGE BODY hire_fire AS
  PROCEDURE hire(...) IS
    BEGIN
      INSERT INTO employees . . .
    END hire;
  PROCEDURE fire(...) IS
    BEGIN
      DELETE FROM employees . . .
    END fire;
END hire_fire;
CREATE PACKAGE BODY raise_bonus AS
  PROCEDURE give_raise(...) IS
    BEGIN
      UPDATE employees SET salary = . . .
    END give_raise;
  PROCEDURE give_bonus(...) IS
    BEGIN
      UPDATE employees SET bonus = . . .
    END give_bonus;
END raise_bonus;
```

次のGRANT EXECUTE文を発行すると、big_bossesロールとlittle_bossesロールが適切なプロシージャを実行でき

るようになります。

```
GRANT EXECUTE ON hire_fire TO big_bosses;
GRANT EXECUTE ON raise_bonus TO little_bosses;
```

親トピック: プロシージャに対する権限がパッケージおよびパッケージ・オブジェクトに与える影響

4.13.5.3 例: プロシージャ権限およびパッケージ・オブジェクト

CREATE PACKAGE BODY文でプロシージャ定義を含むパッケージ本体を作成し、プロシージャ権限とパッケージ・オブジェクトを管理できます。

例4-8は、単一のパッケージ本体にある4つのプロシージャ定義を示しています。2つの追加スタンドアロン・プロシージャと1つのパッケージが特別に作成され、メイン・パッケージ内に定義されているプロシージャへのアクセスを提供します。

例4-8: プロシージャ権限およびパッケージ・オブジェクト

```
CREATE PACKAGE BODY employee_changes AS
  PROCEDURE change_salary(...) IS BEGIN ... END;
  PROCEDURE change_bonus(...) IS BEGIN ... END;
  PROCEDURE insert_employee(...) IS BEGIN ... END;
  PROCEDURE delete_employee(...) IS BEGIN ... END;
END employee_changes;

CREATE PROCEDURE hire
  BEGIN
    employee_changes.insert_employee(...)
  END hire;

CREATE PROCEDURE fire
  BEGIN
    employee_changes.delete_employee(...)
  END fire;

PACKAGE raise_bonus IS
  PROCEDURE give_raise(...) AS
    BEGIN
      employee_changes.change_salary(...)
    END give_raise;

  PROCEDURE give_bonus(...)
    BEGIN
      employee_changes.change_bonus(...)
    END give_bonus;
```

この方法を使用すると、実際に作業を実行するプロシージャ(employee_changesパッケージ内のプロシージャ)が1つのパッケージ内に定義され、宣言されたグローバル変数やカーソルなどを共有できます。トップ・レベルのプロシージャであるhireとfire、および追加のパッケージraise_bonusを宣言することによって、選択的なEXECUTE権限をメイン・パッケージ内のプロシージャに対して付与できます。

```
GRANT EXECUTE ON hire, fire TO big_bosses;
GRANT EXECUTE ON raise_bonus TO little_bosses;
```

EXECUTE権限をパッケージに対して付与することで、すべてのパッケージ・オブジェクトへの均一なアクセスが提供されることに注意してください。

親トピック: プロシージャに対する権限がパッケージおよびパッケージ・オブジェクトに与える影響

4.14 タイプ権限

型、メソッドおよびオブジェクトについて、システム権限とオブジェクト権限を制御できます。

- 名前付きの型に対するシステム権限
  名前付きの型に対するシステム権限を使用すると、ユーザーは自分のスキーマ内での名前付きの型の作成などのアクションを実行できます。
- 名前付きの型のオブジェクト権限
  名前付きの型に適用されるオブジェクト権限は、EXECUTEのみです。
- 名前付きの型のメソッド実行モデル
  名前付きの型のメソッド実行は、他のストアドPL/SQLプロシージャと同じです。
- 型の作成と型を使用した表の作成に必要な権限
  型を作成するには、適切な権限が必要です。
- 例: 型の作成と型を使用した表の作成に必要な権限
  型に対するEXECUTE権限を他のユーザーに付与するには、GRANT OPTION付きのEXECUTE権限が必要です。
- 型アクセスとオブジェクト・アクセスの権限
  DML文に対する列レベルと表レベルの既存の権限は、列オブジェクトと行オブジェクトの両方に適用されます。
- 型の依存性
  プロシージャや表などのストアド・オブジェクトと同様に、他のオブジェクトから参照される型を依存性があると呼びます。

親トピック: 権限とロール認可の構成

4.14.1 名前付きの型に対するシステム権限

名前付きの型に対するシステム権限を使用すると、ユーザーは自分のスキーマ内での名前付きの型の作成などのアクションを実行できます。

表4-4に、名前付きの型(オブジェクト型、VARRAYおよびネストした表)に対するシステム権限のリストを示します。

表4-4 名前付きの型に対するシステム権限

権限	許可される操作
CREATE TYPE	名前付きの型を自分のスキーマ内に作成できます。
CREATE ANY TYPE	名前付きの型を任意のスキーマ内に作成できます。
ALTER ANY TYPE	任意のスキーマにある名前付きの型を変更できます。
DROP ANY TYPE	任意のスキーマにある名前付きの型を削除できます。
EXECUTE ANY TYPE	任意のスキーマにある名前付きの型を使用および参照できます。

RESOURCEロールには、CREATE TYPEシステム権限が含まれています。DBAロールには、これらの権限すべてが含まれています。

親トピック: タイプ権限

4.14.2 名前付きの型のオブジェクト権限

名前付きの型に適用されるオブジェクト権限は、EXECUTEのみです。

名前付きの型に対するEXECUTE権限があるユーザーは、その型を使用して次の操作を実行できます。

- 表の定義
- リレーショナル表への列の定義
- 名前付きの型の変数またはパラメータの宣言

EXECUTE権限によって、ユーザーは、型コンストラクタも含めて、その型のメソッドを起動できます。これは、ストアドPL/SQLプロシージャに対するEXECUTE権限と同じです。

親トピック: タイプ権限

4.14.3 名前付きの型のメソッド実行モデル

名前付きの型のメソッド実行は、他のストアドPL/SQLプロシージャと同じです。

ユーザーには、EXECUTE権限など、名前付きの型を使用するための適切な権限が付与されている必要があります。すべての権限付与と同様に、これらの権限は信頼できるユーザーのみに付与してください。ユーザーに付与された権限を確認するには、DBA_SYS_PRIVSデータ・ディクショナリ・ビューに問い合せます。

関連トピック

- プロシージャ権限

親トピック: タイプ権限

4.14.4 型の作成と型を使用した表の作成に必要な権限

型を作成するには、適切な権限が必要です。

これらの権限は次のとおりです。

- 自分のスキーマにタイプを作成するにはCREATE TYPEシステム権限が必要になり、他のユーザーのスキーマにタイプを作成するにはCREATE ANY TYPEシステム権限が必要になります。これらの権限は、明示的にまたはロールを介して取得できます。
- 型の所有者には、その型の定義内で参照されている他のすべての型にアクセスするためのEXECUTEオブジェクト権限が明示的に付与されているか、EXECUTE ANY TYPEシステム権限が付与されている必要があります。所有者は、ロールを介して必要な権限を取得することはできません。
- 型の所有者が型へのアクセス権を他のユーザーに付与する場合、その所有者には、参照される型に対するEXECUTE権限(GRANT OPTION付きで指定)またはEXECUTE ANY TYPEシステム権限(ADMIN OPTION付きで指定)が必要です。どちらの権限もない型の所有者は、権限不足のため、型へのアクセス権を他のユーザーに付与できません。

型を使用して表を作成するには、表の作成要件と次の要件を満たす必要があります。

- 表の所有者には、その表で参照されているすべての型にアクセスするためのEXECUTEオブジェクト権限が直接付与されているか、EXECUTE ANY TYPEシステム権限が付与されている必要があります。これらの権限がロールを介して付与されている場合、所有者は必要な権限を行使できません。
- 表の所有者が表へのアクセス権を他のユーザーに付与する場合、その所有者には、参照される型に対するEXECUTE

権限(GRANT OPTION付きで指定)またはEXECUTE ANY TYPEシステム権限(ADMIN OPTION付きで指定)が必要です。どちらの権限もない表の所有者は、権限不足のため、表へのアクセス権を付与できません。

関連トピック

- 表権限

親トピック: タイプ権限

4.14.5 例: 型の作成と型を使用した表の作成に必要な権限

型に対するEXECUTE権限を他のユーザーに付与するには、GRANT OPTION付きのEXECUTE権限が必要です。

CONNECTロールとRESOURCEロールを持つ次の3名のユーザーが存在するとします。

- user1
- user2
- user3

次のDDLは、user1のスキーマで実行されます。

```
CREATE TYPE type1 AS OBJECT (
  attr1 NUMBER);
CREATE TYPE type2 AS OBJECT (
  attr2 NUMBER);
GRANT EXECUTE ON type1 TO user2;
GRANT EXECUTE ON type2 TO user2 WITH GRANT OPTION;
```

次のDDLは、user2のスキーマで実行されます。

```
CREATE TABLE tab1 OF user1.type1;
CREATE TYPE type3 AS OBJECT (
  attr3 user1.type2);
CREATE TABLE tab2 (
  col1 user1.type2);
```

次の文は、正常に実行されます。user2には、user1.type2に対するGRANT OPTION付きのEXECUTE権限があるためです。

```
GRANT EXECUTE ON type3 TO user3;
GRANT SELECT ON tab2 TO user3;
```

ただし、次の権限付与は正しく実行されません。user2には、user1.type1に対するGRANT OPTION付きのEXECUTE権限がないためです。

```
GRANT SELECT ON tab1 TO user3;
```

次の文は、user3によって正常に実行されます。

```
CREATE TYPE type4 AS OBJECT (
  attr4 user2.type3);
CREATE TABLE tab3 OF type4;
```

ノート:

CONNECT ロールが現在保持している権限は、CREATE SESSION および SET CONTAINER 権限のみで

す。

親トピック: タイプ権限

4.14.6 型アクセスとオブジェクト・アクセスの権限

DML文に対する列レベルと表レベルの既存の権限は、列オブジェクトと行オブジェクトの両方に適用されます。

表4-5に、オブジェクト表に対する権限をリストします。

表4-5 オブジェクト表に対する権限

権限	許可される操作
SELECT	オブジェクトとその属性に表からアクセスできます。
UPDATE	表の行を構成するオブジェクトの属性を変更できます。
INSERT	表に新規オブジェクトを作成できます。
DELETE	行を削除できます

同様に、列オブジェクトには表に対する権限と列に対する権限が適用されます。インスタンスの取得のみでは、型情報は明らかになりません。ただし、クライアントは、型インスタンスのイメージを解釈する際に名前付きの型の情報にアクセスする必要があります。クライアントが型情報を要求すると、その型に対するEXECUTE権限がチェックされます。

次のスキーマを考えてみます。

```
CREATE TYPE emp_type (
    eno NUMBER, ename CHAR(31), eaddr addr_t);
CREATE TABLE emp OF emp_t;
```

さらに、次の2つの問合せについて考えます。

```
SELECT VALUE(emp) FROM emp;
SELECT eno, ename FROM emp;
```

どちらの問合せの場合も、Oracle Databaseは、emp表に対するユーザーのSELECT権限をチェックします。最初の問合せの場合、ユーザーは、emp_typeの型情報を取得してデータを解釈する必要があります。問合せによってemp_type型がアクセスされると、ユーザーのEXECUTE権限がチェックされます。

ただし、2番目の問合せでは、名前付きの型が含まれていないため、型の権限はチェックされません。

さらに、user3は、前の項のスキーマを使用して次の問合せを実行できます。

```
SELECT tab1.col1.attr2 FROM user2.tab1 tab1;
SELECT attr4.attr3.attr2 FROM tab3;
```

どちらのSELECT文でも、user3には基礎となる型に対する明示的な権限がありませんが、文には成功することに注意してください。これは、型の所有者と表の所有者に、GRANT OPTIONを備えた必要な権限があるためです。

Oracle Databaseは、次のイベントに対する権限をチェックし、アクションに対する権限がクライアントにない場合はエラーを戻します。

- オブジェクトのREF値を使用してオブジェクト・キャッシュ内でオブジェクトを確保すると、Oracle Databaseは、そのオブジェクトが含まれているオブジェクト表に対するSELECT権限をチェックします。
- 既存のオブジェクトを修正したり、オブジェクト・キャッシュからオブジェクトをフラッシュしたりすると、Oracle Databaseは目的のオブジェクト表に対するUPDATE権限をチェックします。
- 新しいオブジェクトをフラッシュすると、Oracle Databaseは目的のオブジェクト表に対するINSERT権限をチェックします。
- オブジェクトを削除すると、Oracle Databaseは目的の表に対するDELETE権限をチェックします。
- 名前付きの型のオブジェクトを確保すると、そのオブジェクトに対するEXECUTE権限がチェックされます。

クライアントの第三世代言語アプリケーションのオブジェクトの属性を変更すると、Oracle Databaseでオブジェクト全体の更新が行われます。そのため、ユーザーにはオブジェクト表に対するUPDATE権限が必要になります。オブジェクト表の特定列に対してのみUPDATE権限を持つことは、アプリケーションがこれらの列に対応した属性のみ変更する場合でも、十分とはいえません。そのため、Oracle Databaseではオブジェクト表の列レベルの権限をサポートしていません。

親トピック: タイプ権限

4.14.7 型の依存性

プロシージャや表などのストアド・オブジェクトと同様に、他のオブジェクトから参照される型を依存性があると呼びます。

表が依存する型については、特殊な問題点がいくつかあります。表には、アクセス用の型定義に依存するデータが含まれているため、その型を変更すると、格納されているすべてのデータにアクセスできなくなります。変更によってこのような結果になるのは、型を使用するために必要な権限が取り消された場合や、型または依存型が削除された場合です。いずれかのアクションが発生すると、表は無効になり、アクセスできなくなります。

必要な権限が再び付与されると、権限がないために無効になった表は自動的に有効になり、アクセスできるようになります。依存型が削除されたことで無効になった表は、アクセス不能のままで、実行できるアクションは表の削除のみです。

型に対する権限の取消しや型の削除は重大な影響があるため、デフォルトでは、SQL文のREVOKEとDROP TYPEは限定されたセマンティクスで実装されます。つまり、どちらの文でも、名前付きの型が表または型に依存している場合は、エラーが戻されて文は取り消されます。ただし、どちらの文も、FORCE句を使用すると常に正常終了します。依存する表がある場合、その表は無効になります。

関連項目:

REVOKEおよびDROP TYPE SQL文の使用の詳細は、*Oracle Database SQL*言語リファレンスを参照してください

親トピック: タイプ権限

4.15 ユーザーへの権限とロールの付与

GRANT文は、プロシージャの実行など、特定のアクションを実行する権限をユーザーに付与します。

- ユーザーおよびロールへのシステム権限とロールの付与
  システム権限とロールをユーザーやロールに付与する前に、これらのタイプの付与に対して権限がどのように機能するかについて注意してください。
- ユーザーおよびロールへのオブジェクト権限の付与
  ユーザーおよびロールにオブジェクト権限を付与できるほか、権限受領者が他のユーザーにその権限を付与することを

許可できます。

親トピック: 権限とロール認可の構成

4.15.1 ユーザーおよびロールへのシステム権限とロールの付与

システム権限とロールをユーザーやロールに付与する前に、これらのタイプの付与に対して権限がどのように機能するかについて注意してください。

- ユーザーおよびロールへのシステム権限とロールの付与のための権限
  システム権限とロールを他のユーザーやロールに付与するには、GRANT SQL文を使用します。
- 例: ユーザーへのシステム権限とロールの付与
  システム権限とロールをユーザーに付与するには、GRANT文を使用できます。
- 例: ディレクトリ・オブジェクトに対するEXECUTE権限の付与
  ディレクトリ・オブジェクトに対するEXECUTE権限を付与するには、GRANT文を使用できます。
- 権限受領ユーザーによる権限付与を可能にするADMINオプションの使用
  WITH ADMIN OPTION句を使用して、権限付与の能力を拡張できます。
- GRANT文を使用した新規ユーザーの作成
  1つのGRANT SQL文で、新規ユーザーを作成してこのユーザーに権限を付与できます。

親トピック: ユーザーへの権限とロールの付与

4.15.1.1 ユーザーおよびロールへのシステム権限とロールの付与のための権限

システム権限とロールを他のユーザーやロールに付与するには、GRANT SQL文を使用します。

次の権限が必要です。

- システム権限を付与するには、ユーザーにADMINオプション付きのシステム権限またはGRANT ANY PRIVILEGEシステム権限が付与されている必要があります。
- ロールを付与するには、ユーザーにADMINオプション付きのロールまたはGRANT ANY ROLEシステム権限が付与されている必要があります。

ノート:

オブジェクト権限は、同じ GRANT 文でシステム権限やロールと同時に付与することはできません。

親トピック: ユーザーおよびロールへのシステム権限とロールの付与

4.15.1.2 例: ユーザーへのシステム権限とロールの付与

システム権限とロールをユーザーに付与するには、GRANT文を使用できます。

例4-9では、システム権限CREATE SESSIONとaccts_payロールをユーザーjwardに付与しています。

例4-9 ユーザーへのシステム権限とロールの付与

```
GRANT CREATE SESSION, accts_pay TO jward;
```

親トピック: ユーザーおよびロールへのシステム権限とロールの付与

4.15.1.3 例: ディレクトリ・オブジェクトに対するEXECUTE権限の付与

ディレクトリ・オブジェクトに対するEXECUTE権限を付与するには、GRANT文を使用できます。

例4-9 では、exec_dirディレクトリ・オブジェクトに対するEXECUTE権限をユーザーjwardに付与しています。

例4-10 ディレクトリ・オブジェクトに対するEXECUTE権限の付与

```
GRANT EXECUTE ON DIRECTORY exec_dir TO jward;
```

親トピック: ユーザーおよびロールへのシステム権限とロールの付与

4.15.1.4 権限受領ユーザーによる権限付与を可能にするADMINオプションの使用

WITH ADMIN OPTION句を使用して、権限付与の能力を拡張できます。

これらの機能は次のとおりです。

- 権限受領者は、データベース内の他のユーザーまたはロールに対して、システム権限またはロールの付与または取消しができます。ただし、自分自身からロールを取り消すことはできません。
- 権限受領者は、ADMINオプション付きのシステム権限やロールを付与できます。
- ロールの権限受領者は、そのロールを変更または削除できます。

例4-11では、new_dbaロールをWITH ADMIN OPTION句付きでユーザーmichaelに付与しています。

例4-11 ADMINオプションの付与

```
GRANT new_dba TO michael WITH ADMIN OPTION;
```

ユーザーmichaelは、new_dbaロール内の権限をすべて暗黙的に使用できることに加え、必要に応じてnew_dbaロールを付与、取消しおよび削除できます。このように、ADMINオプションは非常に強力な機能であるため、このオプションを付けてシステム権限やロールを付与する際は、十分に注意してください。通常、これらの権限はセキュリティ管理者向けに用意されており、システム内の他の管理者やユーザーに付与することはほとんどありません。ユーザーがロールを作成すると、そのロールは自動的にADMINオプション付きでその作成ユーザーに付与されることに注意してください。

親トピック: ユーザーおよびロールへのシステム権限とロールの付与

4.15.1.5 GRANT文を使用した新規ユーザーの作成

1つのGRANT SQL文で、新規ユーザーを作成してこのユーザーに権限を付与できます。

ほとんどの場合、ユーザーにCREATE SESSION権限を付与します。

- GRANT文を使用して新規ユーザーを作成するには、権限およびIDENTIFIED BY句を含めます。

たとえば、新規ユーザーとしてpsmithを作成し、CREATE SESSIONシステム権限をpsmithに付与する方法は、次のとおりです。

```
GRANT CREATE SESSION TO psmith IDENTIFIED BY password;
```

IDENTIFIED BY句を使用してパスワードを指定することで、ユーザー名がデータベースに存在しない場合も、指定したユーザー名とパスワードで新しいユーザーが作成されます。

関連トピック

- ユーザー・アカウントの作成
- パスワードの最低要件

親トピック: ユーザーおよびロールへのシステム権限とロールの付与

4.15.2 ユーザーおよびロールへのオブジェクト権限の付与

ユーザーおよびロールにオブジェクト権限を付与できるほか、権限受領者が他のユーザーにその権限を付与することを許可できます。

- ユーザーおよびロールへのオブジェクト権限の付与について
  GRANT文を使用すると、ロールとユーザーにオブジェクト権限を付与できます。
- WITH GRANT OPTION句が機能するしくみ
  GRANT文でWITH GRANT OPTION句を使用すると、権限受領者はオブジェクト権限を他のユーザーに付与できるようになります。
- オブジェクト所有者にかわるオブジェクト権限の付与
  GRANT ANY OBJECT PRIVILEGEシステム権限を持つユーザーは、オブジェクト所有者のかわりに、すべてのオブジェクト権限の付与と取消しを実行できます。
- 列に対する権限の付与
  表の個々の列に対してINSERT、UPDATEまたはREFERENCES権限を付与できます。
- 行レベルのアクセス制御
  行レベルで、つまりオブジェクト内でアクセスを制御できますが、GRANT文で行うことはできません。

親トピック: ユーザーへの権限とロールの付与

4.15.2.1 ユーザーおよびロールへのオブジェクト権限の付与について

GRANT文を使用すると、ロールとユーザーにオブジェクト権限を付与できます。

オブジェクト権限を付与するには、次のいずれかの条件を満たしている必要があります。

- 指定するオブジェクトを所有している。
- GRANT ANY OBJECT PRIVILEGEシステム権限を付与されている。この権限を使用すると、オブジェクト所有者のかわりに権限の付与と取消しを実行できます。
- オブジェクト権限が付与されるときに、WITH GRANT OPTION句が指定されている。

ノート:

同じ GRANT 文で、オブジェクト権限とともにシステム権限とロールを付与することはできません。

次の例では、emp表のすべての列に対するREAD、INSERTおよびDELETEのオブジェクト権限をユーザーjfeeとtsmithに付与しています。

```
GRANT READ, INSERT, DELETE ON emp TO jfee, tsmith;
```

salaryビューのすべてのオブジェクト権限をユーザーjfeeに付与するには、次の例のようにALLキーワードを使用します。

```
GRANT ALL ON salary TO jfee;
```

ノート:

権限受領者は、最初の権限付与にGRANT OPTIONが含まれていないかぎり、オブジェクトへのアクセス権を再度付与することはできません。したがって、前述の例では、jfeeはGRANT文を使用してオブジェクト権限を他の誰かに付与することができません。

親トピック: ユーザーおよびロールへのオブジェクト権限の付与

4.15.2.2 WITH GRANT OPTION句が機能するしくみ

GRANT文でWITH GRANT OPTION句を使用すると、権限受領者はオブジェクト権限を他のユーザーに付与できるようになります。

スキーマ内にオブジェクトを格納しているユーザーには、関連するすべてのオブジェクト権限がWITH GRANT OPTION句付きで自動的に付与されます。この特別な権限によって、権限受領者の権限は次のように拡張されます。

- 権限受領者は、データベース内の任意のユーザーにGRANT OPTIONの有無を問わずオブジェクト権限を付与でき、そしてデータベース内の任意のロールに付与できます。
- 次の2つの条件が成立する場合、権限受領者は表に対するビューを作成し、そのビューの対応する権限をデータベース内の任意のユーザーまたはロールに対して付与できます。
    - 権限受領者が、表に対するGRANT OPTION付きのオブジェクト権限を受領している。
    - 権限受領者に、CREATE VIEWまたはCREATE ANY VIEWシステム権限がある。

ノート:

オブジェクト権限をロールに付与する場合、WITH GRANT OPTION句は無効です。Oracle Databaseでは、ロールによるオブジェクト権限の伝播を禁止しているため、ロールの権限受領者は、ロールを介して受領したオブジェクト権限を他に伝播することはできません。

親トピック: ユーザーおよびロールへのオブジェクト権限の付与

4.15.2.3 オブジェクト所有者にかわるオブジェクト権限の付与

GRANT ANY OBJECT PRIVILEGEシステム権限を持つユーザーは、オブジェクト所有者のかわりに、すべてのオブジェクト権限の付与と取消しを実行できます。

この権限によって、データベース管理者やアプリケーション管理者は、スキーマに接続せずにスキーマ内のオブジェクトへのアクセス権を付与できます。この権限を持つスキーマ所有者のログイン資格証明はメンテナンスする必要がなく、構成時に必要な接続数が減少します。

このシステム権限はOracle Databaseに用意されているDBAロールに付属しているため、AS SYSDBAで接続するユーザー(ユーザーSYS)に(ADMIN option付きで)付与されます。他のシステム権限と同様に、GRANT ANY OBJECT PRIVILEGEシステム権限を付与できるのは、ADMIN option権限を持っているユーザーのみです。

オブジェクトに対するアクセス権の記録された権限付与者は、オブジェクト所有者とGRANT ANY OBJECT PRIVILEGEシステム権限を行使している個人のどちらかです。GRANT ANY OBJECT PRIVILEGEがある権限付与者にGRANT OPTION付きのオブジェクト権限がない場合、オブジェクト所有者が権限付与者として表示されます。権限付与者がGRANT OPTION付きのオブジェクト権限を持っている場合は、その権限付与者が付与者として記録されます。

ノート:

GRANT 文によって生成された監査レコードには、常に権限付与を実際に実行したユーザーが示されます。

たとえば、次の使用例を考えてみます。ユーザーadamsにGRANT ANY OBJECT PRIVILEGEシステム権限があります。他の付与権限は所持していないとします。このユーザーが次の文を発行します。

```
GRANT SELECT ON HR.EMPLOYEES TO blake WITH GRANT OPTION;
```

DBA_TAB_PRIVSビューを調べると、hrが権限付与者として表示されていることがわかります。

```
SELECT GRANTEE, GRANTOR, PRIVILEGE, GRANTABLE
  FROM DBA_TAB_PRIVS
  WHERE TABLE_NAME = 'EMPLOYEES' and OWNER = 'HR';
GRANTEE  GRANTOR PRIVILEGE    GRANTABLE
-------- ------- -----------  ----------
BLAKE    HR       SELECT      YES
```

ここで、ユーザーblakeにもGRANT ANY OBJECT PRIVILEGEシステム権限があると想定します。このユーザーが次の文を発行します。

```
GRANT SELECT ON HR.EMPLOYEES TO clark;
```

この場合は、DBA_TAB_PRIVSビューを再び問い合せると、blakeが権限付与者として表示されます。

```
GRANTEE  GRANTOR  PRIVILEGE  GRANTABLE
-------- -------- ---------  ----------
BLAKE    HR       SELECT     YES
CLARK    BLAKE    SELECT     NO
```

これは、blakeがすでにHR.EMPLOYEESに対してGRANT OPTION付きのSELECT権限を持っているためです。

関連トピック

- オブジェクト所有者にかわるオブジェクト権限の取消し

親トピック: ユーザーおよびロールへのオブジェクト権限の付与

4.15.2.4 列に対する権限の付与

表の個々の列に対してINSERT、UPDATEまたはREFERENCES権限を付与できます。

ノート:

列固有の INSERT 権限を付与する前に、NOT NULL 制約が定義されている列が表に含まれているかどうかを確認してください。挿入できる列を選んで権限を付与したときに NOT NULL 列が抜けていると、ユーザーは表に行を挿入できません。このような状況を回避するには、各 NOT NULL 列が挿入可能であること、または NULL 以外のデフォルト値があることを確認してください。そうでない場合、権限受領者は表に行を挿入できず、エラーが発生します。

次の文は、accounts表のacct_no列に対するINSERT権限をpsmithに付与しています。

```
GRANT INSERT (acct_no) ON accounts TO psmith;
```

次の例では、emp表のenameおよびjob列に対するオブジェクト権限が、ユーザーjfeeおよびtsmithに付与されます。

```
GRANT INSERT(ename, job) ON emp TO jfee, tsmith;
```

親トピック: ユーザーおよびロールへのオブジェクト権限の付与

4.15.2.5 行レベルのアクセス制御

行レベルで、つまりオブジェクト内でアクセスを制御できますが、GRANT文で行うことはできません。

このタイプのアクセス制御を実行するには、Oracle Virtual Private Database (VPD)またはOracle Label Security (OLS)のいずれかを使用する必要があります。

関連項目:

- Oracle Virtual Private Databaseを使用したデータ・アクセスの制御
- 列レベルOracle Virtual Private Databaseのポリシー
- Oracle Label Security管理者ガイド

親トピック: ユーザーおよびロールへのオブジェクト権限の付与

4.16 ユーザーからの権限とロールの取消し

システムまたはオブジェクトの権限を取り消す場合は、権限の取消しによる連鎖的な影響に注意してください。

- システム権限とロールの取消し
  REVOKE SQL文で、システム権限およびロールを取り消します。
- オブジェクト権限の取消し
  複数のオブジェクト権限、オブジェクト所有者のかわりに付与したオブジェクト権限、列を選択するオブジェクト権限、およびREFERENCESオブジェクト権限を取り消すことができます。
- 権限の取消しによる連鎖的な影響
  DDL操作に関連するオブジェクト権限の取消しでは連鎖的な影響はありませんが、オブジェクト権限の取消しに関する連鎖的な影響はあります。

親トピック: 権限とロール認可の構成

4.16.1 システム権限とロールの取消し

REVOKE SQL文で、システム権限およびロールを取り消します。

ADMINオプション付きでシステム権限またはロールを付与されているユーザーは、他のデータベース・ユーザーまたはロールから権限またはロールを取り消すことができます。取消しを行うユーザーは、権限やロールを最初に付与したユーザーでなくてもかまいません。GRANT ANY ROLEがあるユーザーは、任意のロールを取り消すことができます。

例4-12は、CREATE TABLEシステム権限とaccts_recロールをユーザーpsmithから取り消します。

例4-12 ユーザーからのシステム権限とロールの取消し

```
REVOKE CREATE TABLE, accts_rec FROM psmith;
```

システム権限またはロールのADMINオプションを選択的に取り消すことはできないことに注意してください。権限またはロールを取り消してから、同じ権限またはロールをADMINオプションを指定せずに再度付与する必要があります。

親トピック: ユーザーからの権限とロールの取消し

4.16.2 オブジェクト権限の取消し

複数のオブジェクト権限、オブジェクト所有者のかわりに付与したオブジェクト権限、列を選択するオブジェクト権限、およびREFERENCESオブジェクト権限を取り消すことができます。

- オブジェクト権限の取消しについて
  オブジェクト権限を取り消す場合、ユーザーは一定の要件を満たしている必要があります。
- 複数のオブジェクト権限の取消し
  REVOKE文で、1つのオブジェクトに対する複数の権限を取り消すことができます。
- オブジェクト所有者にかわるオブジェクト権限の取消し
  GRANT ANY OBJECT PRIVILEGEシステム権限を使用して、オブジェクト所有者が権限付与者であるオブジェクト権限を取り消すことができます。
- 列を選択するオブジェクト権限の取消し
  列固有操作に対するGRANTおよびREVOKE操作には異なる権限と制約があります。
- REFERENCESオブジェクト権限の取消し
  REFERENCESオブジェクト権限を取り消すと、外部キー制約に影響を与えます。

親トピック: ユーザーからの権限とロールの取消し

4.16.2.1 オブジェクト権限の取消しについて

オブジェクト権限を取り消す場合、ユーザーは一定の要件を満たしている必要があります。

次のいずれかの条件を満たしている必要があります。

- 以前にユーザーまたはロールにオブジェクト権限を付与している。
- オブジェクト所有者のかわりに権限を付与したり取り消すことができるGRANT ANY OBJECT PRIVILEGEシステム権限を所持している。

取り消すことができるのは、権限付与者として自身で直接認可した権限のみです。GRANT OPTIONを付与した他のユーザーによる権限付与を取り消すことはできません。ただし、連鎖的な影響があります。権限付与者のオブジェクト権限を取り消すと、GRANT OPTIONを使用して伝播されたオブジェクト権限も取り消されます。

親トピック: オブジェクト権限の取消し

4.16.2.2 複数のオブジェクト権限の取消し

REVOKE文で、1つのオブジェクトに対する複数の権限を取り消すことができます。

最初の権限付与者が次の文を発行すると、ユーザーjfeeとpsmithからemp表のSELECT権限とINSERT権限が取り消されます。

```
REVOKE SELECT, INSERT ON emp FROM jfee, psmith;
```

次の文は、最初にhuman_resourceロールに付与したdept表に対するすべてのオブジェクト権限を取り消します。

```
REVOKE ALL ON dept FROM human_resources;
```

ノート:

オブジェクト権限の GRANT OPTION を選択的に取り消すことはできません。オブジェクト権限を取り消してから、

同じ権限を GRANT OPTION を指定せずに再度付与する必要があります。ユーザーが自分自身からオブジェクト権限を取り消すことはできません。

親トピック: オブジェクト権限の取消し

4.16.2.3 オブジェクト所有者にかわるオブジェクト権限の取消し

GRANT ANY OBJECT PRIVILEGEシステム権限を使用して、オブジェクト所有者が権限付与者であるオブジェクト権限を取り消すことができます。

この操作を実行できるのは、オブジェクト所有者によって、または所有者のかわりにGRANT ANY OBJECT PRIVILEGEシステム権限を持つユーザーによって、オブジェクト権限が付与されている場合です。

オブジェクト権限が、オブジェクト所有者とREVOKE文を実行するユーザー(特定のオブジェクト権限とGRANT ANY OBJECT PRIVILEGEシステム権限の両方を持つユーザー)の両方によって付与されている場合は、REVOKE文を発行したユーザーによって付与されたオブジェクト権限のみが取り消されます。この使用例については、オブジェクト所有者にかわるオブジェクト権限の付与の例を使用して説明します。

ここでは、blakeがHR.EMPLOYEESに対するSELECT権限をclarkに付与しているとします。blakeはGRANT ANY OBJECT PRIVILEGEシステム権限を所有していますが、特定のオブジェクト権限も保有しているため、この付与はblakeに起因します。ここでは、ユーザーHRもHR.EMPLOYEESに対するSELECT権限をユーザーclarkに付与しているとします。DBA_TAB_PRIVSビューの問合せでは、HR.EMPLOYEES表について次の権限付与が有効であることが表示されています。

```
GRANTEE  GRANTOR PRIVILEGE    GRANTABLE
-------- ------- -----------  ----------
BLAKE    HR      SELECT        YES
CLARK    BLAKE   SELECT        NO
CLARK    HR      SELECT        NO
```

ユーザーblakeが次のREVOKE文を発行します。

```
REVOKE  SELECT ON HR.EMPLOYEES FROM clark;
```

ユーザーblakeがユーザーclarkに付与したオブジェクト権限のみが削除されます。オブジェクト所有者HRによる権限付与はそのまま残ります。

```
GRANTEE  GRANTOR PRIVILEGE    GRANTABLE
-------- ------- -----------  ----------
BLAKE    HR      SELECT       YES
CLARK    HR      SELECT       NO
```

blakeが再度REVOKE文を発行すると、今度は(HRのかわりに)adamsがGRANT ANY OBEJCT PRIVILEGEシステム権限を使用して付与したオブジェクト権限が削除されます。

関連トピック

- オブジェクト所有者にかわるオブジェクト権限の付与

親トピック: オブジェクト権限の取消し

4.16.2.4 列を選択するオブジェクト権限の取消し

列固有操作に対するGRANTおよびREVOKE操作には異なる権限と制約があります。

表やビューの列を指定してINSERT、UPDATEおよびREFERENCES権限を付与することは可能ですが、同じようなREVOKE文を使用して、列固有の権限を選択的に取り消すことはできません。かわりに、権限付与者は表またはビューの全列に対する

オブジェクト権限を取り消してから、残しておく列固有の権限を選択的に再度付与する必要があります。

たとえば、human_resourcesロールには、dept表のdeptno列およびdname列に対するUPDATE権限が付与されているとします。このUPDATE権限をdeptno列のみから取り消すには、次の2つの文を発行します。

```
REVOKE UPDATE ON dept FROM human_resources;
GRANT UPDATE (dname) ON dept TO human_resources;
```

このREVOKE文によって、ロールhuman_resourcesからdept表の全列に対するUPDATE権限が取り消されます。次にGRANT文によって、human_resourcesロールに対して、dname列のUPDATE権限の付与がやりなおし、リストアまたは再発行されます。

親トピック: オブジェクト権限の取消し

4.16.2.5 REFERENCESオブジェクト権限の取消し

REFERENCESオブジェクト権限を取り消すと、外部キー制約に影響を与えます。

REFERENCESオブジェクト権限の権限受領者がその権限を使用して外部キー制約を作成し、その制約が現在も存在している場合、権限付与者がその権限を取り消すには、REVOKE文にCASCADE CONSTRAINTSオプションを指定します。

たとえば:

```
REVOKE REFERENCES ON dept FROM jward CASCADE CONSTRAINTS;
```

CASCADE CONSTRAINTS句を指定すると、取り消されるREFERENCES権限を使用して現在も定義されている外部キー制約が削除されます。

親トピック: オブジェクト権限の取消し

4.16.3 権限の取消しによる連鎖的な影響

DDL操作に関連するオブジェクト権限の取消しではカスケード効果はありませんが、オブジェクト権限の取消しに関するカスケード効果はあります。

- システム権限の取消しによる連鎖的な影響
  DDL操作に関連するシステム権限を取り消したときには連鎖的な影響は発生しません。
- オブジェクト権限の取消しによる連鎖的な影響
  オブジェクト権限を取り消すと、連鎖的な影響が発生する場合があります。

親トピック: ユーザーからの権限とロールの取消し

4.16.3.1 システム権限の取消しによる連鎖的な影響

DDL操作に関連するシステム権限を取り消したときには連鎖的な影響は発生しません。

これは、権限がADMINオプションとともに付与されたかどうかとは関係ありません。

たとえば、次のような場合を考えてみます。

1. セキュリティ管理者が、ADMIN optionを指定して、ユーザーjfeeにCREATE TABLEシステム権限を付与します。
2. ユーザーjfeeが表を作成します。
3. ユーザーjfeeが、ユーザーtsmithにCREATE TABLEシステム権限を付与します。
4. ユーザーtsmithが表を作成します。

5. セキュリティ管理者が、ユーザーjfeeからCREATE TABLEシステム権限を取り消します。
6. ユーザーjfeeが作成した表はそのまま残ります。ユーザーtsmithの表とCREATE TABLEシステム権限はそのまま残ります。

連鎖的な影響は、DML操作に関連するシステム権限を取り消したときに発生する場合があります。ユーザーのSELECT ANY TABLE権限を取り消すと、そのユーザーのスキーマ内に存在し、この権限に依存しているすべてのプロシージャは、権限が再度認可されないかぎり、正常に実行できなくなります。

親トピック: 権限の取消しによる連鎖的な影響

4.16.3.2 オブジェクト権限の取消しによる連鎖的な影響

オブジェクト権限を取り消すと、連鎖的な影響が発生する場合があります。

次のことに注意してください。

- DMLオブジェクト権限を取り消すと、そのDMLオブジェクト権限に依存するオブジェクト定義にまで影響が及ぶ可能性があります。たとえば、testプロシージャの本体に、emp表のデータを問い合せるSQL文が記述されているとします。testプロシージャの所有者からemp表のSELECT権限を取り消すと、それ以降そのプロシージャを正常に実行できなくなります。
- 表に対するREFERENCES権限をユーザーから取り消すと、そのユーザーが定義した外部キー整合性制約の中で、取り消されたREFERENCES権限を必要とする制約が自動的に削除されます。たとえば、ユーザーjwardにdept表のdeptno列に対するREFERENCES権限が付与されているとします。このユーザーが、emp表のdeptno列に、dept表のdeptno列を参照する外部キーを作成します。dept表のdeptno列に対するREFERENCES権限を取り消すと、同じ操作でemp表のdeptno列に対する外部キー制約も削除されます。
- GRANT OPTIONを使用して伝播されたオブジェクト権限付与は、権限付与者のオブジェクト権限が取り消されると取り消されます。たとえば、GRANT OPTION付きのemp表に対するSELECTオブジェクト権限を付与されているuser1が、user2に対してemp表に対するSELECT権限を付与したとします。その後、user1からSELECT権限を取り消します。このREVOKE文はuser2にも連鎖します。user1とuser2の取り消されたSELECT権限に依存するオブジェクトもすべて、前述のように影響を受ける可能性があります。

ALTERまたはINDEXオブジェクト権限を取り消しても、ALTERおよびINDEX DDLの各オブジェクト権限を必要とするオブジェクト定義には影響を与えません。たとえば、別のユーザーに属する表に索引を作成したユーザーからINDEX権限を取り消しても、その索引はそのまま残ります。

親トピック: 権限の取消しによる連鎖的な影響

4.17 PUBLICロールに対する権限の付与と取消し

ロールPUBLICに対して、権限とロールの付与および取消しを実行できます。

PUBLICにはすべてのデータベース・ユーザーがアクセスできるため、PUBLICに対して付与されたすべての権限またはロールには、すべてのデータベース・ユーザーがアクセスできます。デフォルトでは、PUBLICは付与される権限がありません。

セキュリティ管理者とデータベース・ユーザーは、すべてのデータベース・ユーザーが権限またはロールを必要としている場合のみ、PUBLICに権限またはロールを付与してください。これによって、各データベース・ユーザーには常に、現在のグループ・タスクの正常実行に必要な権限のみが付与されているという一般規則が保証されます。

PUBLICロールから権限を取り消すと、かなりの規模で影響が連鎖する可能性があります。DML操作に関連する任意の権限をPUBLICから取り消すと(たとえばSELECT ANY TABLEまたはUPDATE ON emp)、データベース内のファンクションとパッ

ケージを含むすべてのプロシージャが再認可されるまで再び使用できなくなります。したがって、DMLに関連する権限をPUBLICに付与したり、取り消す場合には注意が必要です。

関連項目:

- オブジェクト依存性の管理の詳細は、『Oracle Database管理者ガイド』を参照してください。
- データの保護に関するガイドライン

親トピック: 権限とロール認可の構成

4.18 オペレーティング・システムまたはネットワークを使用したロールの付与

オペレーティング・システムまたはネットワークを使用してロールを管理すると、大規模エンタープライズでのロールの一元管理が容易になります。

- オペレーティング・システムまたはネットワークを使用したロールの付与について
  Oracle Databaseを実行するオペレーティング・システムを使用して、接続時にロールをユーザーに付与できます。
- オペレーティング・システムのロール識別機能
  OS_ROLES初期化パラメータを使用して、オペレーティング・システムがロールをどのように識別するかを制御できます。
- オペレーティング・システムのロール管理機能
  オペレーティング・システムによって管理されているロールを使用する場合は、データベース・ロールがオペレーティング・システムのユーザーに付与されることに注意してください。
- OS_ROLESがTRUEに設定されている場合のロールの付与と取消し
  OS_ROLES初期化パラメータをTRUEに設定すると、ユーザーに対するロールの付与と取消しをオペレーティング・システムで管理できるようになります。
- OS_ROLESがTRUEに設定されている場合のロールの有効化と無効化
  OS_ROLES初期化パラメータをTRUEに設定すると、オペレーティング・システムによって付与されたロールをSET ROLE文で動的に有効にできます。
- オペレーティング・システムによるロール管理使用時のネットワーク接続
  ロールがオペレーティング・システムによって管理されている場合、デフォルトでユーザーは共有サーバーを通じてデータベースに接続できません。

親トピック: 権限とロール認可の構成

4.18.1 オペレーティング・システムまたはネットワークを使用したロールの付与について

Oracle Databaseを実行するオペレーティング・システムを使用して、接続時にロールをユーザーに付与できます。

この機能を使用することでセキュリティ管理者は、ユーザーのデータベース・ロールをGRANT文やREVOKE文を使用して明示的に付与したり取り消したりする必要がなくなります。

つまり、オペレーティング・システムを使用してロールを管理し、ユーザーのセッション作成時にOracle Databaseにそのロールを渡すことができます。このメカニズムの一部として、ユーザーのデフォルト・ロールや、ADMINオプション付きでユーザーに付与するロールを識別できます。オペレーティング・システムを使用してロールを使用するユーザーを認可する場合は、必ずすべてのロールをデータベース内に作成し、GRANT文を使用してそのロールに権限を割り当てる必要があります。

ロールは、ネットワーク・サービスを介しても付与できます。

ユーザーのデータベース・ロールを識別するためにオペレーティング・システムを使用する方法の利点は、Oracleデータベースの権

限管理をデータベースの外部で実施できることです。オペレーティング・システムに組み込まれたセキュリティ機能によって、ユーザーの権限が制御されます。また、このオプションを使用すると、いくつかのシステム・アクティビティのセキュリティを集中管理できるため、次のような状況で役立ちます。

- MVS版Oracleの管理者が、データベース・ユーザーのロールを識別するためにRACFグループを使用する場合
- UNIX版Oracleの管理者が、データベース・ユーザーのロールを識別するためにUNIXグループを使用する場合
- VMS版Oracleの管理者が、データベース・ユーザーのロールを識別するために権限識別子を使用する場合

ユーザーのデータベース・ロールを識別するためにオペレーティング・システムを使用する方法の主なデメリットは、権限管理がロール・レベルでしか実施できないことです。個々の権限は、オペレーティング・システムを使用して付与することはできません。ただし、GRANT文を使用してデータベース内で付与することは可能です。

この機能を使用する際の第2のデメリットは、オペレーティング・システムがロールを管理している場合に、デフォルトではユーザーが共有サーバーまたはその他のネットワーク接続を介してデータベースに接続できないことです。ただし、このデフォルトはオペレーティング・システムによるロール管理使用時のネットワーク接続の説明に従って変更できます。

マルチテナント環境の場合、データベース管理者のオペレーティング・システム認証を使用できるのはCDBルートのみです。PDB、アプリケーション・ルートおよびアプリケーションPDBには使用できません。

ノート:

この項で説明されている機能は、一部のオペレーティング・システムでしか使用できません。これらの機能が使用できるかどうかを確認するには、使用しているオペレーティング・システム固有の Oracle Database マニュアルを参照してください。

親トピック: オペレーティング・システムまたはネットワークを使用したロールの付与

4.18.2 オペレーティング・システムのロール識別機能

OS_ROLES初期化パラメータを使用して、オペレーティング・システムがロールをどのように識別するかを制御できます。

セッションの作成時に、データベースがオペレーティング・システムを使用して各ユーザーのデータベース・ロールを識別するように、初期化パラメータOS_ROLESをTRUEに設定します。

インスタンスが現在稼働している場合、そのインスタンスを再起動する必要があります。ユーザーがデータベースとのセッションを作成しようとすると、Oracle Databaseはオペレーティング・システムによって識別されるデータベース・ロールを使用して、そのユーザーのセキュリティ・ドメインを初期化します。

ユーザーのデータベース・ロールを識別するためには、各Oracle Databaseユーザーのオペレーティング・システム・アカウントが、どのデータベース・ロールをユーザーが使用できるようになるかを示すオペレーティング・システム識別子(これらはグループ、権限識別子、または他の類似した名前で呼ばれる場合があります)を持っている必要があります。ロールの指定は、どのロールがユーザーのデフォルト・ロールであるか、どのロールがADMINオプションで使用可能かを示すことも可能です。どのオペレーティング・システムを使用している場合も、オペレーティング・システム・レベルでのロールの指定は次の形式になります。

```
ora_ID_ROLE[[_d][_a][_da]]
```

詳細は、次のとおりです。

- IDの定義は、オペレーティング・システムが異なると変わります。たとえばVMSでは、IDはデータベースのインスタンス識別子、VMSではコンピュータ・タイプ、UNIXではシステムIDです。

IDは、ORACLE_SIDと照合する際に大/小文字が区別されます。ROLEでは、大/小文字は区別されません。

- ROLEは、データベース・ロールの名前です。
- dは、このロールがデータベース・ユーザーのデフォルト・ロールであることを示すオプション文字です。
- aは、このロールがADMINオプション付きでユーザーに付与されることを示すオプション文字です。このオプション文字を指定することによって、ユーザーはこのロールを他のロールにのみ付与できるようになります。オペレーティング・システムを使用してロールを管理している場合は、ユーザーにロールを付与できません。

  dまたはaのいずれかを指定する場合は、その文字の直前にアンダースコア(_)を指定してください。

たとえば、オペレーティング・システム・アカウントが、プロファイルで識別される次のロールを持っているとします。

```
ora_PAYROLL_ROLE1
ora_PAYROLL_ROLE2_a
ora_PAYROLL_ROLE3_d
ora_PAYROLL_ROLE4_da
```

対応するユーザーがOracle Databaseのpayrollインスタンスに接続すると、role3とrole4がデフォルト・ロールになり、role2とrole4がADMINオプション付きで付与されます。

親トピック: オペレーティング・システムまたはネットワークを使用したロールの付与

4.18.3 オペレーティング・システムのロール管理機能

オペレーティング・システムによって管理されているロールを使用する場合は、データベース・ロールがオペレーティング・システムのユーザーに付与されることに注意してください。

オペレーティング・システム・ユーザーが接続できるデータベース・ユーザーは、認可されたデータベース・ロールを使用できます。このため、OS_ROLES = TRUEを使用している場合は、権限が付与されているオペレーティング・システム・アカウントにデータベース・アカウントを対応付けるために、すべてのOracle DatabaseユーザーをIDENTIFIED EXTERNALLYとして定義することを考慮してください。

親トピック: オペレーティング・システムまたはネットワークを使用したロールの付与

4.18.4 OS_ROLESがTRUEに設定されている場合のロールの付与と取消し

OS_ROLES初期化パラメータをTRUEに設定すると、ユーザーに対するロールの付与と取消しをオペレーティング・システムで管理できるようになります。

それまでにGRANT文によってユーザーに付与されたロールは適用されません。ただし、それらのロールはデータ・ディクショナリには残っています。オペレーティング・システム・レベルでのユーザーへのロールの付与のみが適用されます。この場合も、ユーザーは権限をロールとユーザーに付与できます。

ノート:

オペレーティング・システムによって ADMIN オプション付きでロールが付与された場合、ユーザーはそのロールを他のロールにのみ付与できます。

親トピック: オペレーティング・システムまたはネットワークを使用したロールの付与

4.18.5 OS_ROLESがTRUEに設定されている場合のロールの有効化と無効化

OS_ROLES初期化パラメータをTRUEを設定すると、オペレーティング・システムによって付与されたロールをSET ROLE文で動的に有効にできます。

ロールがパスワードやオペレーティング・システムによる認可を必要とするように定義されていた場合でも、この文は適用されます。ただし、ユーザーのオペレーティング・システム・アカウントで識別されないロールは、SET ROLE文では指定できません。これは、OS_ROLES = FALSEのときにGRANT文を使用してロールを付与していた場合でも同じです。(このようなロールを指定しても無視されます。)

OS_ROLESがTRUEに設定されている場合、ユーザーは最大148個のロールを使用可能にできます。この数には、ロールに付与されている可能性のある他のロールも含まれることに注意してください。

親トピック: オペレーティング・システムまたはネットワークを使用したロールの付与

4.18.6 オペレーティング・システムによるロール管理使用時のネットワーク接続

ロールがオペレーティング・システムによって管理されている場合、デフォルトでユーザーは共有サーバーを通じてデータベースに接続できません。

リモート・ユーザーは保護されていない接続を介して別のオペレーティング・システム・ユーザーになりすますおそれがあるため、デフォルトでこのような制限が適用されています。

このようなセキュリティに対する危険性の心配がない場合は、初期化パラメータREMOTE_OS_ROLESをTRUEに設定することで、共有サーバーまたはその他のネットワーク接続でオペレーティング・システムのロール管理を使用できます。この変更は、次回インスタンスを起動して、データベースをマウントするときに有効になります。このパラメータのデフォルト設定はFALSEです。

親トピック: オペレーティング・システムまたはネットワークを使用したロールの付与

4.19 SET ROLEおよびデフォルト・ロールの設定による権限の付与と取消しの機能

権限付与およびSET ROLE文は、付与と取消しが適用されるタイミングと方法に影響します。

- 権限の付与と取消しが有効になるとき
  付与と取消しがいつ有効になるかは、付与または取り消す権限によって異なります。
- SET ROLE文が付与と取消しに与える影響
  ユーザーまたはアプリケーションは、ユーザー・セッション中にSET ROLE文を何度でも使用して、そのセッションで使用可能になっているロールを変更できます。
- ユーザーのデフォルト・ロールの指定
  ユーザーがログインすると、Oracle Databaseではそのユーザーに明示的に付与されている権限と、そのユーザーのデフォルト・ロールに含まれる権限が、すべて使用可能になります。
- ユーザーが使用可能にできるロールの最大数
  ロールはいくつでもユーザーに付与できますが、任意の時点でログイン・ユーザーに対して有効にできるロール数は最大148個です。

親トピック: 権限とロール認可の構成

4.19.1 権限の付与と取消しが有効になるとき

付与と取消しがいつ有効になるかは、付与または取り消す権限によって異なります。

権限の付与と取消しは次のように有効になります。

- 任意の対象(ユーザー、ロールおよびPUBLIC)に対するシステム権限とオブジェクト権限の付与および取消しは、即時に有効になります。
- 任意の対象(ユーザー、他のロール、PUBLIC)に対するロールの付与および取消しが有効になるのは、その付与および取消しの実行後、ロールを再度使用可能にするために現行のユーザー・セッションでSET ROLE文を発行したとき、あるいはその付与または取消しを実行した後に新しくユーザー・セッションを作成したときです。

現在使用可能なロールは、SESSION_ROLESデータ・ディクショナリ・ビューを問い合せることによって確認できます。

親トピック: SET ROLEおよびデフォルト・ロールの設定による権限の付与と取消しの機能

4.19.2 SET ROLE文が付与と取消しに与える影響

ユーザーまたはアプリケーションは、ユーザー・セッション中にSET ROLE文を何度でも使用して、そのセッションで使用可能になっているロールを変更できます。

ユーザーには、SET ROLE文で指定したロールが、事前に付与されている必要があります。

次の例では、すでに付与されているロールclerkを使用可能にして、パスワードを指定しています。

```
SET ROLE clerk IDENTIFIED BY password;
```

「パスワードの最低要件」のガイドラインに従って、passwordを安全なパスワードに置き換えます。

次の例は、SET ROLEを使用してすべてのロールを使用禁止にする方法を示しています。

```
SET ROLE NONE;
```

親トピック: SET ROLEおよびデフォルト・ロールの設定による権限の付与と取消しの機能

4.19.3 ユーザーのデフォルト・ロールの指定

ユーザーがログインすると、Oracle Databaseではそのユーザーに明示的に付与されている権限と、そのユーザーのデフォルト・ロールに含まれる権限が、すべて使用可能になります。

1. デフォルト・ロールの設定対象のユーザーにロールがGRANT文で直接付与されているか、CREATE ROLE権限があるユーザーによってロールが作成されていることを確認します。
2. DEFAULT ROLE句を指定してALTER USER文を使用して、ユーザーのデフォルト・ロールを指定します。

たとえば、ユーザーjaneに対してデフォルト・ロールのpayclerkとpettycashを設定する方法は、次のとおりです。

```
ALTER USER jane DEFAULT ROLE payclerk, pettycash;
```

ALTER USER文のDEFAULT ROLE句の制限事項は、『Oracle Database SQL言語リファレンス』を参照してください。

CREATE USER文ではユーザーのデフォルト・ロールを設定できません。ユーザーを最初に作成すると、デフォルトのユーザー・ロール設定はALLであり、ユーザーにその後に付与されるすべてのロールがデフォルト・ロールになります。デフォルトのユーザー・ロールを制限するには、ALTER USER文を使用します。

ノート:

(グローバル・ロールやアプリケーション・ロール以外の)ロールを作成すると、このロールが自分に暗黙的に付与され、自分のデフォルト・ロールのセットが新しいロールを含むように更新されます。ユーザー・セッションで有効にできるロールは148 個のみであることに注意してください。DBA ロールのような集計ロールがユーザーに付与されると、そのロールに付与されるロールはユーザーが持っているロール数に含まれます。たとえば、あるロールには 20 個のロールが付与されており、そのロールをユーザーに付与すると、そのユーザーは 21 個の追加ロールを持っていることになります。したがって、新しいロールをユーザーに付与するときは、ALTER USER 文の DEFAULT ROLE 句を使用してそのユーザーのデフォルト・ロールとして指定されるロールが多すぎないよう確認してください。

親トピック: SET ROLEおよびデフォルト・ロールの設定による権限の付与と取消しの機能

4.19.4 ユーザーが使用可能にできるロールの最大数

ロールはいくつでもユーザーに付与できますが、任意の時点でログイン・ユーザーに対して有効にできるロール数は最大148個です。

したがって、ユーザー・セッションでこのユーザーはすべての権限を使用できるわけではありません。ベスト・プラクティスとして、ユーザーに付与するロールの数を必要最小限のロールに制限します。

関連トピック

- ロールの保護に関するガイドライン

親トピック: SET ROLEおよびデフォルト・ロールの設定による権限の付与と取消しの機能

4.20 ユーザー権限およびロールのデータ・ディクショナリ・ビュー

特別な問合せを使用して、様々なタイプの権限およびロール付与に関する情報を入手できます。

- 権限およびロール付与の情報を確認するデータ・ディクショナリ・ビュー
  Oracle Databaseには、権限およびロール付与に関する情報を表示するデータ・ディクショナリ・ビューが用意されています。
- すべてのシステム権限の付与を表示する問合せ
  DBA_SYS_PRIVSデータ・ディクショナリ・ビューは、ロールとユーザーに対して付与されているすべてのシステム権限を返します。
- すべてのロール付与を表示する問合せ
  DBA_ROLE_PRIVS問合せは、ユーザーと他のロールに対して付与されているロールをすべて返します。
- ユーザーに付与されているオブジェクト権限を表示する問合せ
  DBA_TAB_PRIVSおよびDBA_COL_PRIVSデータ・ディクショナリ・ビューには、ユーザーに付与されているオブジェクト権限が表示されます。
- セッションの現在の権限ドメインを表示する問合せ
  SESSION_ROLESおよびSESSION_PRIVSデータ・ディクショナリ・ビューには、データベース・セッションの現在の権限ドメインが表示されます。
- データベースのロールを表示する問合せ
  DBA_ROLESデータ・ディクショナリ・ビューでは、データベースのすべてのロールと各ロールに対して使用されている認証が表示されます。
- ロールの権限ドメイン情報を表示する問合せ

ROLE_ROLE_PRIVS、ROLE_SYS_PRIVSおよびROLE_TAB_PRIVSの各データ・ディクショナリ・ビューには、ロールの権限ドメインに関する情報が表示されます。

親トピック: 権限とロール認可の構成

4.20.1 権限およびロール付与の情報を確認するデータ・ディクショナリ・ビュー

Oracle Databaseには、権限およびロール付与に関する情報を表示するデータ・ディクショナリ・ビューが用意されています。

次の表に、権限とロールの付与に関する情報にアクセスするために問合せ可能なビューを示します。

表4-6 権限およびロール情報を表示するデータ・ディクショナリ・ビュー

ビュー	説明
ALL_COL_PRIVS	オブジェクト所有者、権限付与者または権限受領者が現行ユーザーまたはPUBLIC である列オブジェクトの権限付与がすべて表示されます。
ALL_COL_PRIVS_MADE	オブジェクト所有者または権限付与者が現行ユーザーである列オブジェクトの権限付与がリストされます
ALL_COL_PRIVS_RECD	権限受領者が現行ユーザーまたは PUBLIC である列オブジェクトの権限付与が表示されます。
ALL_TAB_PRIVS	権限受領者がユーザーまたは PUBLIC であるオブジェクトの権限付与がリストされます。
ALL_TAB_PRIVS_MADE	現行ユーザーが行ったオブジェクトの権限付与、または現行ユーザーが所有するオブジェクトに対する権限付与がすべてリストされます
ALL_TAB_PRIVS_RECD	権限受領者がユーザーまたは PUBLIC であるオブジェクトの権限付与がリストされます。
DBA_COL_PRIVS	データベース内の列オブジェクトの権限付与がすべて表示されます。
DBA_CONTAINER_DATA	マルチテナント環境では、デフォルト(ユーザーレベル)およびオブジェクト固有のCONTAINER_DATA 属性を表示します。CONTAINER_DATA 句で作成されるオブジェクトには、CONTAINER_DATA 属性が含まれます。
DBA_EPG_DAD_AUTHORIZATION	別のユーザー権限の使用が許可されたデータベース・アクセス記述子(DAD)が表示されます
DBA_LOCKDOWN_PROFILES	PDB ロックダウン・プロファイルに関する情報が表示されます

ビュー	説明
DBA_OBJECTS	オブジェクト・リンクまたはメタデータ・リンクがあるオブジェクトがリストされます。これらのオブジェクトを確認するには、OBJECT_NAME および SHARING 列を問い合せます。
DBA_TAB_PRIVS	データベース内のすべてのオブジェクトに対するすべての権限付与がリストされます。
DBA_ROLES	セキュア・アプリケーション・ロールを含めて、データベース内に存在するすべてのロールがリストされます PUBLIC ロールはリストされません
DBA_ROLE_PRIVS	ユーザーとロールに直接付与されているロールがリストされます。
DBA_SYS_PRIVS	ユーザーとロールに付与されているシステム権限がリストされます。
ROLE_ROLE_PRIVS	他のロールに付与されているロールがリストされます。ユーザーがアクセス権限を持っているロールの情報のみが得られます
ROLE_SYS_PRIVS	ロールに付与されているシステム権限がリストされます。ユーザーがアクセス権限を持っているロールの情報のみが得られます
ROLE_TAB_PRIVS	ロールに付与されているオブジェクト権限がリストされます。ユーザーがアクセス権限を持っているロールの情報のみが得られます
SESSION_PRIVS	ユーザーに対して現在使用可能になっている権限がリストされます。
SESSION_ROLES	現在のユーザーに対して使用可能になっているすべてのロールがリストされます。PUBLIC ロールはリストされません
USER_COL_PRIVS	オブジェクト所有者、権限付与者または権限受領者が現行ユーザーである列オブジェクトの権限付与が表示されます。
USER_COL_PRIVS_MADE	オブジェクト所有者が現行ユーザーである列オブジェクトの権限付与が表示されます。
USER_COL_PRIVS_RECD	権限受領者が現行ユーザーである列オブジェクトの権限付与が表示されます。
USER_EPG_DAD_AUTHORIZATION	別のユーザー権限の使用が許可されたデータベース・アクセス記述子(DAD)が表示されます

ビュー	説明
USER_ROLE_PRIVS	現行ユーザーに直接付与されているロールがリストされます。
USER_TAB_PRIVS	権限受領者が現行ユーザーであるすべてのオブジェクトの権限付与がリストされます。
USER_SYS_PRIVS	現行ユーザーに付与されているシステム権限がリストされます。
USER_TAB_PRIVS_MADE	現行ユーザーが所有しているすべてのオブジェクトの権限付与がリストされます。
USER_TAB_PRIVS_RECD	権限受領者が現行ユーザーであるオブジェクトの権限付与がリストされます。
V$PWFILE_USERS	管理権限を付与された現在の PDB のすべてのユーザーをリストします

次の表に、権限とロールの付与に関する情報にアクセスするために問合せ可能なビューを示します。

ここでは、これらのビューの使用例をいくつか示します。各例では、次の文がすでに発行されていることを前提としています。

```
CREATE ROLE security_admin IDENTIFIED BY password;
GRANT CREATE PROFILE, ALTER PROFILE, DROP PROFILE,
    CREATE ROLE, DROP ANY ROLE, GRANT ANY ROLE, AUDIT ANY,
    AUDIT SYSTEM, CREATE USER, BECOME USER, ALTER USER, DROP USER
    TO security_admin WITH ADMIN OPTION;
GRANT READ, DELETE ON SYS.AUD$ TO security_admin;
GRANT security_admin, CREATE SESSION TO swilliams;
GRANT security_admin TO system_administrator;
GRANT CREATE SESSION TO jward;
GRANT READ, DELETE ON emp TO jward;
GRANT INSERT (ename, job) ON emp TO swilliams, jward;
```

関連トピック

- Oracle Databaseリファレンス

親トピック: ユーザー権限およびロールのデータ・ディクショナリ・ビュー

4.20.2 すべてのシステム権限の付与を表示する問合せ

DBA_SYS_PRIVSデータ・ディクショナリ・ビューは、ロールとユーザーに対して付与されているすべてのシステム権限を返します。

たとえば:

```
SELECT GRANTEE, PRIVILEGE, ADM FROM DBA_SYS_PRIVS;
GRANTEE             PRIVILEGE                        ADM
--------------      ---------------------------------- ---
SECURITY_ADMIN      ALTER PROFILE                    YES
SECURITY_ADMIN      ALTER USER                       YES
SECURITY_ADMIN      AUDIT ANY                        YES
SECURITY_ADMIN      AUDIT SYSTEM                     YES
SECURITY_ADMIN      BECOME USER                      YES
SECURITY_ADMIN      CREATE PROFILE                   YES
SECURITY_ADMIN      CREATE ROLE                      YES
SECURITY_ADMIN      CREATE USER                      YES
SECURITY_ADMIN      DROP ANY ROLE                    YES
SECURITY_ADMIN      DROP PROFILE                     YES
```

```
SECURITY_ADMIN     DROP USER                          YES
SECURITY_ADMIN     GRANT ANY ROLE                     YES
SWILLIAMS          CREATE SESSION                     NO
JWARD              CREATE SESSION                     NO
```

関連項目:

DBA_SYS_PRIVSビューの詳細は、『Oracle Databaseリファレンス』を参照してください。

親トピック: ユーザー権限およびロールのデータ・ディクショナリ・ビュー

4.20.3 すべてのロール付与を表示する問合せ

DBA_ROLE_PRIVS問合せは、ユーザーと他のロールに対して付与されているロールをすべて返します。

たとえば:

```
SELECT * FROM DBA_ROLE_PRIVS;
GRANTEE             GRANTED_ROLE                       ADM
------------------- ---------------------------------- ---
SWILLIAMS           SECURITY_ADMIN                     NO
```

関連項目:

DBA_ROLE_PRIVSビューの詳細は、『Oracle Databaseリファレンス』を参照してください。

親トピック: ユーザー権限およびロールのデータ・ディクショナリ・ビュー

4.20.4 ユーザーに付与されているオブジェクト権限を表示する問合せ

DBA_TAB_PRIVSおよびDBA_COL_PRIVSデータ・ディクショナリ・ビューには、ユーザーに付与されているオブジェクト権限が表示されます。

DBA_TAB_PRIVSデータ・ディクショナリ・ビューは、指定のユーザーに対して付与されているオブジェクト権限をすべて(列固有の権限を除く)返します。

たとえば:

```
SELECT TABLE_NAME, PRIVILEGE, GRANTABLE FROM DBA_TAB_PRIVS
    WHERE GRANTEE = 'jward';
TABLE_NAME   PRIVILEGE    GRANTABLE
-----------  ------------ ----------
EMP          SELECT       NO
EMP          DELETE       NO
```

付与されている列固有の権限をすべて表示するには、次の問合せを使用します。

```
SELECT GRANTEE, TABLE_NAME, COLUMN_NAME, PRIVILEGE
    FROM DBA_COL_PRIVS;
GRANTEE      TABLE_NAME     COLUMN_NAME      PRIVILEGE
-----------  ------------   -------------    --------------
SWILLIAMS    EMP            ENAME            INSERT
SWILLIAMS    EMP            JOB              INSERT
JWARD        EMP            NAME             INSERT
JWARD        EMP            JOB              INSERT
```

関連項目:

DBA_TAB_PRIVSビューの詳細は、『Oracle Databaseリファレンス』を参照してください。

親トピック: ユーザー権限およびロールのデータ・ディクショナリ・ビュー

4.20.5 セッションの現在の権限ドメインを表示する問合せ

SESSION_ROLESおよびSESSION_PRIVSデータ・ディクショナリ・ビューには、データベース・セッションの現在の権限ドメインが表示されます。

SESSION_ROLESビューでは、発行者が現在使用できるロールがすべて表示されます。

たとえば:

```
SELECT * FROM SESSION_ROLES;
```

ユーザーswilliamsに対してsecurity_adminロールが使用可能になっている場合に、この問合せを実行すると、Oracle Databaseから次の情報が戻されます。

```
ROLE
------------------------------
SECURITY_ADMIN
```

次の問合せを実行すると、発行者のセキュリティ・ドメインで現在使用可能なシステム権限がすべて表示されます。これには、明示的に付与されている権限と使用可能なロールから付与された権限の両方が含まれています。

```
SELECT * FROM SESSION_PRIVS;
```

ユーザーswilliamsに対してsecurity_adminロールが使用可能になっている場合に、この問合せを実行すると、Oracle Databaseから次の結果が戻されます。

```
PRIVILEGE
----------------------------------------
AUDIT SYSTEM
CREATE SESSION
CREATE USER
BECOME USER
ALTER USER
DROP USER
CREATE ROLE
DROP ANY ROLE
GRANT ANY ROLE
AUDIT ANY
CREATE PROFILE
ALTER PROFILE
DROP PROFILE
```

ユーザーswilliamsに対してsecurity_adminロールが使用禁止になっている場合、最初の問合せでは何も表示されず、2番目の問合せではCREATE SESSION権限の付与に関する行が1行のみ表示されます。

関連項目:

SESSION_ROLESビューの詳細は、『Oracle Databaseリファレンス』を参照してください。

親トピック: ユーザー権限およびロールのデータ・ディクショナリ・ビュー

4.20.6 データベースのロールを表示する問合せ

DBA_ROLESデータ・ディクショナリ・ビューでは、データベースのすべてのロールと各ロールに対して使用されている認証が表示されます。

たとえば:

```
SELECT * FROM DBA_ROLES;
ROLE                  PASSWORD
----------------      --------
CONNECT               NO
RESOURCE              NO
DBA                   NO
SECURITY_ADMIN        YES
```

関連項目:

DBA_ROLESビューの詳細は、『Oracle Databaseリファレンス』を参照してください。

親トピック: ユーザー権限およびロールのデータ・ディクショナリ・ビュー

4.20.7 ロールの権限ドメイン情報を表示する問合せ

ROLE_ROLE_PRIVS、ROLE_SYS_PRIVSおよびROLE_TAB_PRIVSの各データ・ディクショナリ・ビューには、ロールの権限ドメインに関する情報が表示されます。

たとえば:

```
SELECT GRANTED_ROLE, ADMIN_OPTION
   FROM ROLE_ROLE_PRIVS
   WHERE ROLE = 'SYSTEM_ADMIN';
GRANTED_ROLE              ADM
----------------          ----
SECURITY_ADMIN            NO
```

次の問合せを実行すると、security_adminロールに付与されているシステム権限がすべて表示されます。

```
SELECT * FROM ROLE_SYS_PRIVS WHERE ROLE = 'SECURITY_ADMIN';
ROLE                    PRIVILEGE                      ADM
----------------------- -----------------------------  ---
SECURITY_ADMIN           ALTER PROFILE                 YES
SECURITY_ADMIN           ALTER USER                    YES
SECURITY_ADMIN           AUDIT ANY                     YES
SECURITY_ADMIN           AUDIT SYSTEM                  YES
SECURITY_ADMIN           BECOME USER                   YES
SECURITY_ADMIN           CREATE PROFILE                YES
SECURITY_ADMIN           CREATE ROLE                   YES
SECURITY_ADMIN           CREATE USER                   YES
SECURITY_ADMIN           DROP ANY ROLE                 YES
SECURITY_ADMIN           DROP PROFILE                  YES
SECURITY_ADMIN           DROP USER                     YES
SECURITY_ADMIN           GRANT ANY ROLE                YES
```

次の問合せを実行すると、security_adminロールに付与されているオブジェクト権限がすべて表示されます。

```
SELECT TABLE_NAME, PRIVILEGE FROM ROLE_TAB_PRIVS
    WHERE ROLE = 'SECURITY_ADMIN';
TABLE_NAME                    PRIVILEGE
---------------------------   ----------------
AUD$                          DELETE
AUD$                          SELECT
```

関連トピック

- Oracle Databaseリファレンス

親トピック: ユーザー権限およびロールのデータ・ディクショナリ・ビュー

5 権限分析の実行による権限使用の特定

権限分析では、ユーザーが使用中および未使用の権限とロールが動的に分析されます。

- 権限分析とは
  権限分析により、データベース・ロールおよび権限に対する最低限の権限のベスト・プラクティスを実装することで、アプリケーションおよびデータベース操作のセキュリティが向上します。
- 権限分析ポリシーの作成および管理
  SQL*Plus、SQLcl、SQL DeveloperまたはEnterprise Manager Cloud Controlなどのツールを使用して、権限分析ポリシーを作成および管理できます。
- Cloud Controlによるロールの作成および権限の管理
  権限分析レポートで検出された権限を使用して新規ロールを作成し、次にこのロールをユーザーに付与できます。
- チュートリアル: 取得実行の使用によるANY権限使用の分析
  このチュートリアルでは、取得実行を作成してREAD ANY TABLEシステム権限の使用を分析する方法を示します。
- チュートリアル: DBAロールを持つユーザーによる権限の使用の分析
  このチュートリアルでは、DBAロールを持ち、データベースのチューニング操作を実行するユーザーの権限の使用を分析する方法を示します。
- 権限分析ポリシーおよびレポート・データ・ディクショナリ・ビュー
  Oracle Databaseには、分析された権限に関する情報を示すデータ・ディクショナリ・ビューが用意されています。

親トピック: ユーザー認証および認可の管理

5.1 権限分析とは

権限分析により、データベース・ロールおよび権限に対する最低限の権限のベスト・プラクティスを実装することで、アプリケーションおよびデータベース操作のセキュリティが向上します。

- 権限分析について
  Oracle Databaseカーネルの内部で実行する権限分析では、最低限の権限モデルを実装するための使用済および未使用の権限を識別することで、ユーザー、ツールおよびアプリケーション・アカウントの攻撃面を縮小するのに役立ちます。
- 権限分析の利点およびユースケース
  権限の使用の分析は、不必要に付与された権限を検出したり最低限の権限のベスト・プラクティスを実装するために役立ちます。
- 権限分析を実行できるユーザー
  権限分析を使用するには、CAPTURE_ADMINロールが付与されている必要があります。
- 権限分析のタイプ
  様々なタイプの権限分析ポリシーを作成して、特定の目的を実現できます。
- マルチテナント環境による権限分析への影響について
  マルチテナント環境での権限分析ポリシーの作成および使用が可能です。
- 権限分析でのプリコンパイル済データベース・オブジェクトの処理
  権限分析を使用すると、プリコンパイル済データベース・オブジェクトで使用されている権限を取得できます。

親トピック: 権限分析の実行による権限使用の特定

5.1.1 権限分析について

Oracle Databaseカーネルの内部で実行する権限分析では、最低限の権限モデルを実装するための使用済および未使用の権限を識別することで、ユーザー、ツールおよびアプリケーション・アカウントの攻撃面を縮小するのに役立ちます。

権限分析は、指定された期間中にデータベース・ユーザーおよびアプリケーションによって使用されている権限を動的に取得します。

権限分析を使用すると、最低限の権限のガイドラインを迅速かつ効率的に実施するのに役立ちます。最低限の権限モデルでは、ユーザーにジョブの実行に必要な権限およびアクセス権のみが与えられます。ユーザーが様々なタスクを実行することがよくありますが、その場合でも、ユーザーにはすべて同じ強力な権限のセットが付与されます。権限分析なしでは、各ユーザーが持つ必要がある権限を見つけ出すのが困難で、多くの場合、様々なタスクがある場合でも、ユーザーは共通の権限のセットの一部で終わる可能性があります。権限を管理する組織であっても、ユーザーは時間の経過とともに権限を累積し、権限が失われることはほとんどありません。業務分離により、1つのプロセスが様々なユーザーの個別のタスクに分割されます。最低限の権限では、ユーザーが必要なタスクのみを実行できるように分離が強制されます。業務の分離の強制は内部制御に役立ちますが、特権資格証明を盗む悪質なユーザーからのリスクも減少します。

権限分析では、実行時にデータベース・ユーザーおよびアプリケーションによって使用される権限を取得し、問合せ可能なデータ・ディクショナリ・ビューにその結果を書き込みます。アプリケーションに定義者の権限および起動者の権限のプロシージャが含まれている場合、権限分析では、権限取得が作成され有効化される前にプロシージャがコンパイルされていた場合でも、プロシージャのコンパイルと実行に必要な権限が取得されます。ユーザーから権限を取り消すかわりに、ユーザーの権限の使用状況を監査し、Oracle Audit Vault and Database Firewallなどのアプリケーションを使用して適切な管理者にアラートを送信できます。

親トピック: 権限分析とは

5.1.2 権限分析の利点およびユースケース

権限の使用の分析は、不必要に付与された権限の検索と最低限の権限のベスト・プラクティスの実装に役立ちます。

- 最低限の権限のベスト・プラクティス
  データベースにアクセスするアカウントの権限は、アプリケーションまたはユーザーが厳密に必要とする権限に制限する必要があります。
- セキュアなアプリケーションの開発
  管理者によっては、アプリケーションの開発フェーズで、多数の強力なシステム権限およびロールとSYSDBA管理権限をアプリケーション開発者に付与することがあります。

親トピック: 権限分析とは

5.1.2.1 最低限の権限のベスト・プラクティス

データベースにアクセスするアカウントの権限は、アプリケーションまたはユーザーが厳密に必要とする権限に制限する必要があります。

ただし、特にサード・パーティがアプリケーションを開発する場合、必要以上の権限が便宜上アプリケーション接続プール・アカウントに付与される可能性があります。さらに、一部の開発者は、システム権限およびアプリケーションのオブジェクト権限をPUBLICロールに付与します。

たとえば、アプリケーション・データから選択してアプリケーション・プロシージャを実行するには、システム権限SELECT ANY TABLEおよびEXECUTE ANY PROCEDUREをアプリケーション・アカウントappsysに付与します。これで、appsysは、意図しない場合でもアプリケーション以外のデータにアクセスできます。この状況で、ユーザーappsysで権限の使用状況を分析し、

結果に基づき必要に応じて権限を取り消したり付与できます。

また、アプリケーション・アカウントには、データベースでアプリケーションをインストールおよびメンテナンスするために必要な追加の権限が付与されることもよくあります。これらはアプリケーションのメンテナンス期間中にのみ必要なものですが、常に使用可能になっています。適切なプロセスは、アプリケーションのメンテナンスに必要な権限を別のロールに追加し、そのロールをメンテナンス期間中にのみアプリケーションに付与することです。

親トピック: 権限分析の利点およびユースケース

5.1.2.2 セキュアなアプリケーションの開発

管理者によっては、アプリケーションの開発フェーズで、多数の強力なシステム権限およびロールとSYSDBA管理権限をアプリケーション開発者に付与することがあります。

管理者がこのようにするのは、その段階ではアプリケーション開発者が開発中にどの権限を必要としていて、どの権限とロールは不要かは不明である場合があるためです。

アプリケーションが開発されて稼働すると、アプリケーション開発者に必要な権限とそうでない権限が明確になります。フル・リグレッション・テストによってアプリケーションの実行中に権限分析を取得すると、アプリケーションが実行時に使用する必要がある権限をすべてではないにしても、そのほとんどを取得できます。メンテナンス更新をテストするときに権限分析を取得すると、本番システムの更新中に必要な権限を提供できます。この時点で、セキュリティ管理者は不必要な権限の取消しを開始できます。ただし、アプリケーションが現在問題なく稼働しているため、アプリケーション開発者がこの考えに抵抗する場合があります。管理者は、権限分析を使用して、アプリケーションで使用されている各権限を調べることで、権限を取り消した場合に、アプリケーションの稼働を継続できるかを確認できます。

たとえば、app_ownerは、データベースに接続するアプリケーションのアプリケーション・データベース・ユーザーです。ユーザーapp_ownerは、OE、SHおよびPMスキーマの表を問い合せる必要があります。これらのスキーマの各表のSELECTオブジェクト権限を付与するかわりに、セキュリティ管理者は、SELECT ANY TABLE権限をapp_ownerに付与します。しばらくしてから、新しいスキーマHRが作成され、機密データがHR.EMPLOYEES表に挿入されます。ユーザーapp_ownerにはSELECT ANY TABLE権限があるため、この表を問い合せて機密データにアクセスできますが、これはセキュリティ上問題があります。システム権限(特にANY権限)を付与するかわりに、特定の表に対するオブジェクト権限を付与する方がはるかによい方法です。

親トピック: 権限分析の利点およびユースケース

5.1.3 権限分析を実行できるユーザー

権限分析を使用するには、CAPTURE_ADMINロールが付与されている必要があります。

DBMS_PRIVILEGE_CAPTURE PL/SQLパッケージを使用して、権限取得を管理します。権限分析から提供されるデータ・ディクショナリ・ビューを使用して、権限の使用を分析します。

親トピック: 権限分析とは

5.1.4 権限分析のタイプ

様々なタイプの権限分析ポリシーを作成して、特定の目的を実現できます。

- コンテキストベースの権限使用の取得。SYS_CONTEXT関数でのみ、ブール式を指定する必要があります。条件がTRUEと評価されると、使用されている権限が取得されます。この方法は、SYS_CONTEXTでユーザーを指定することによって、データベース・ユーザーにより使用される権限およびロールを取得するために使用できます。
- ロールベースの権限使用の取得。ロールのリストを提供する必要があります。リストのロールがデータベース・セッションで

有効な場合、そのセッションの使用されている権限が取得されます。Oracleデフォルト・ロール、ユーザーが作成したロール、コード・ベース・アクセス制御(CBAC)ロール、およびセキュア・アプリケーション・ロールの権限使用を取得できます。

- ロールおよびコンテキストベースの権限使用の取得。有効なロールのリストと条件のSYS_CONTEXTブール式の両方を指定する必要があります。このようなロールのいずれかがセッションで有効であり、指定されたコンテキスト条件が満たされると、権限分析によって権限の使用の取得が開始されます。
- データベース全体の権限の取得。権限分析ポリシーにタイプを指定しない場合、ユーザーSYSの権限を除いて、データベースの使用されている権限が取得されます。(条件なしで有効であるため、無条件の分析とも呼ばれます。)

次の制約に注意してください:

- 一度に有効にできる権限分析ポリシーは1つのみです。唯一の例外は、ロールやコンテキスト属性ドリブン分析ポリシーなどのデータベース全体以外の権限分析ポリシーと同時にデータベース全体の権限分析ポリシーを有効化できることです。
- SYSユーザーの権限は分析できません。
- 権限分析には権限への付与パスが表示されますが、どの付与パスを維持するかは推奨されません。
- ロール、ユーザーまたはオブジェクトが削除されると、権限分析データ・ディクショナリ・ビューでこれらについての権限取得を反映する値も削除されます。

親トピック: 権限分析とは

5.1.5 マルチテナント環境による権限分析への影響について

マルチテナント環境での権限分析ポリシーの作成および使用が可能です。

マルチテナント環境を使用している場合は、CDBルートか個々のPDBのどちらかで権限分析ポリシーを作成できます。権限分析ポリシーは、それが作成されたコンテナで、CDBルートまたはアプリケーション・ルート内で使用された権限、またはPDB内で使用された権限のどちらかにのみ適用されます。マルチテナント環境全体にグローバルに適用することはできません。CAPTURE_ADMINロールをローカル・ユーザーまたは共通ユーザーにローカルに付与できます。CAPTURE_ADMINロールを共通ユーザーに共通に付与できます。

親トピック: 権限分析とは

5.1.6 権限分析でのプリコンパイル済データベース・オブジェクトの処理

権限分析を使用すると、プリコンパイル済データベース・オブジェクトで使用されている権限を取得できます。

これらのオブジェクトの例として、PL/SQLパッケージ、プロシージャ、ファンクション、ビュー、トリガーおよびJavaクラスとデータがあります。

このような権限は、ストアド・プロシージャがコールされると実行時に使用されないことがあるため、データベース全体の取得について結果を生成するときに、実行時に取得された権限とともに収集されます。権限は、プリコンパイル済データベース・オブジェクトまたは実行時取得で使用されていない場合、使用されていない権限として扱われ、実行時取得名の下に保存されます。権限がプリコンパイル済データベース・オブジェクトに使用されている場合は、取得名ORA$DEPENDENCYの下に保存されます。権限が実行時に取得された場合は、実行時取得名の下に保存されます。プリコンパイル済データベース・オブジェクトと実行時使用の両方について使用されている権限を把握する場合は、ORA$DEPENDENCYと実行時の両方の取得を問い合せる必要があります。使用されていない権限の場合、実行時取得名で問い合せるだけです。

権限分析を使用できるプリコンパイル済オブジェクトの詳細なリストを確認するには、ALL_DEPENDENCIESデータ・ディクショナリ・ビューのTYPE列を問い合せます。

親トピック: 権限分析とは

5.2 権限分析ポリシーの作成および管理

SQL*Plus、SQLcl、SQL DeveloperまたはEnterprise Manager Cloud Controlなどのツールを使用して、権限分析ポリシーを作成および管理できます。

- 権限分析ポリシーの作成および管理について
  DBMS_PRIVILEGE_CAPTURE PL/SQLパッケージまたはOracle Enterprise Manager Cloud Controlを使用して権限を分析できます。
- 権限分析の管理の一般ステップ
  権限分析の一般的な一連のステップに従う必要があります。
- 権限分析ポリシーの作成
  DBMS_PRIVILEGE_CAPTURE.CREATE_CAPTUREプロシージャを使用して権限分析ポリシーを作成できます。
- 権限分析ポリシーの作成の例
  様々な権限分析ポリシーを作成できます。
- 権限分析ポリシーの有効化
  権限分析ポリシーを作成したら、そのポリシーを有効化して権限の使用を取得する必要があります。
- 権限分析ポリシーの無効化
  権限分析レポートを生成する前に、権限分析ポリシーを無効化する必要があります。
- 権限分析レポートの生成
  DBMS_PRIVILEGE_CAPTURE PL/SQLパッケージを使用すると、Enterprise Manager Cloud ControlまたはSQL*Plusのいずれかで、権限分析ポリシー・レポートを生成できます。
- 権限分析ポリシーの削除
  権限分析ポリシーを削除する前に、最初にそれを無効化する必要があります。

親トピック: 権限分析の実行による権限使用の特定

5.2.1 権限分析ポリシーの作成および管理について

DBMS_PRIVILEGE_CAPTURE PL/SQLパッケージまたはOracle Enterprise Manager Cloud Controlを使用して権限を分析できます。

そのためには、CAPTURE_ADMINロールが付与されている必要があります。DBMS_PRIVILEGE_CAPTUREパッケージでは、権限分析ポリシーを作成、有効化、無効化および削除できます。DBA_*ビューで表示できる権限の使用状況を示すレポートも生成します。

関連トピック

- Oracle Database PL/SQLパッケージ・プロシージャおよびタイプ・リファレンス

親トピック: 権限分析ポリシーの作成および管理

5.2.2 権限分析の管理の一般ステップ

権限分析の一般的な一連のステップに従う必要があります。

1. 権限分析ポリシーを定義します。
2. 権限分析ポリシーを有効化します。

   このステップでポリシーが定義した権限使用の記録が始まります。必要に応じて、この取得実行の名前を指定します。権限分析ポリシーを有効にするたびに、それに対して異なる取得実行を作成できます。このようにして、後で行う比較分析のために複数の名前付き取得実行を作成できます。

3. 定義者の権限および起動者の権限のプログラム単位によって使用される権限を取得する必要がある場合は、オプションで、依存性権限を取得するポリシーを有効にします。
4. データの収集に十分な期間が経過したら、権限分析ポリシーの権限使用記録を無効にします。

   このステップでポリシーの権限使用の取得が終了します。

5. 権限分析の結果を生成します。

   このステップでは、権限分析ポリシーおよびレポート・データ・ディクショナリ・ビューに結果を書き込みます。

6. オプションで、権限分析ポリシーおよび取得実行を無効化してから削除します。

   権限分析ポリシーを削除すると、ポリシーによって取得されたデータが削除されます。

関連トピック

- 権限分析ポリシーおよびレポート・データ・ディクショナリ・ビュー

親トピック: 権限分析ポリシーの作成および管理

5.2.3 権限分析ポリシーの作成

権限分析ポリシーを作成するには、DBMS_PRIVILEGE_CAPTURE.CREATE_CAPTUREプロシージャを使用できます。

権限分析ポリシーを作成した後、DBA_PRIV_CAPTURESデータ・ディクショナリ・ビューのリストで確認できます。ポリシーが作成されると、Oracleデータ・ディクショナリおよびSYSスキーマに配置されます。ただし、SYS、およびポリシーを作成したユーザーは、それを削除できます。権限使用の分析を開始できるように、ポリシー作成後に手動で有効にする必要があります。

1. CAPTURE_ADMINロールを持つユーザーとして、データベース・インスタンスにログインします。
2. DBMS_PRIVILEGE_CAPTURE.CREATE_CAPTUREプロシージャに次の構文を使用します。

```
DBMS_PRIVILEGE_CAPTURE.CREATE_CAPTURE(
   name              VARCHAR2,
   description       VARCHAR2 DEFAULT NULL,
   type              NUMBER DEFAULT DBMS_PRIVILEGE_CAPTURE.G_DATABASE,
   roles             ROLE_NAME_LIST DEFAULT ROLE_NAME_LIST(),
   condition         VARCHAR2 DEFAULT NULL);
```

   詳細は、次のとおりです。

   - name: 作成する権限分析ポリシーの名前を指定します。この名前は必ず一意にし、128文字以下にしてください。名前には空白を使用できますが、参照するときは必ず一重引用符でその名前を囲む必要があります。既存のポリシーの名前を確認するには、DBA_PRIV_CAPTURESビューのNAME列を問い合せます。
   - description: 大/小文字混在で最大1024文字で権限分析ポリシーの目的を説明します。オプションです。
   - type: 取得条件のタイプを指定します。typeパラメータを省略する場合、デフォルトはDBMS_PRIVILEGE_CAPTURE.G_DATABASEとなります。オプションです。

次のいずれかのタイプを入力します。

- DBMS_PRIVILEGE_CAPTURE.G_DATABASE: ユーザーSYSの権限を除いて、データベース全体で使用されている権限をすべて取得します。
- DBMS_PRIVILEGE_CAPTURE.G_ROLE: ロールが有効になっているセッションの権限を取得します。typeパラメータのDBMS_PRIVILEGE_CAPTURE.G_ROLEを入力する場合、rolesパラメータも指定する必要があります。ロールが複数の場合、各ロール名をカンマで分けます。
- DBMS_PRIVILEGE_CAPTURE.G_CONTEXT: conditionパラメータで指定された条件がTRUEと評価されているセッションの権限を取得します。typeパラメータのDBMS_PRIVILEGE_CAPTURE.G_CONTEXTを入力する場合、conditionパラメータも指定する必要があります。
- DBMS_PRIVILEGE_CAPTURE.G_ROLE_AND_CONTEXT: ロールが有効になっており、コンテキスト条件がTRUEと評価されているセッションの権限を取得します。typeパラメータのDBMS_PRIVILEGE_CAPTURE.G_ROLE_AND_CONTEXTを入力する場合、rolesとconditionの両方のパラメータも指定する必要があります。

- roles: 使用された権限が分析されるロールを指定します。つまり、指定されたロールのいずれかの権限が使用される場合、権限が分析されます。type引数にDBMS_PRIVILEGE_CAPTURE.G_ROLEまたはDBMS_PRIVILEGE_CAPTURE.G_ROLE_AND_CONTEXTを指定する場合、この引数を指定する必要があります。入力する各ロールは、データベースに存在する必要があります。(DBA_ROLESデータ・ディクショナリ・ビューを問い合せて、既存のロールを確認できます。)複数のロールでは、可変長配列型role_name_listを使用してロール名を入力します。最大10のロールを指定できます。

  たとえば、2つのロールを指定するには、次のようにします。

  ```
  roles => role_name_list('role1', 'role2'),
  ```

- condition: 最大4000文字でブール式を指定します。type引数にDBMS_PRIVILEGE_CAPTURE.G_CONTEXTまたはDBMS_PRIVILEGE_CAPTURE.G_ROLE_AND_CONTEXTを指定する場合、この引数を指定する必要があります。関係演算子(==、>、>=、<、<=、<>、BETWEENおよびIN)を使用したSYS_CONTEXT式のみがこのブール式で許可されます。

  condition式の構文は次のとおりです。

  ```
  predicate::= SYS_CONTEXT(namespace, attribute) relop constant_value |
               SYS_CONTEXT(namespace, attribute)
               BETWEEN
               constant_value
               AND constant_value | SYS_CONTEXT(namespace, attribute)
               IN {constant_value (,constant_value)* }
  relop::= = | < | <= | > | >= | <>
  context_expression::= predicate | (context_expression)
               AND (context_expression) | (context_expression)
               OR (context_expression )
  ```

  たとえば、conditionを使用してIPアドレス192.0.2.1を指定するには、次のようにします。

  ```
  condition => 'SYS_CONTEXT(''USERENV'', ''IP_ADDRESS'')=''192.0.2.1''';
  ```

権限分析ポリシーを作成したら、ポリシーを有効にして権限およびロールの使用を取得する必要があります。

* 必要な数の定数値を追加できます(たとえば、IN {constant_value1}やIN {constant_value1, constant_value2, constant_value3}など)。

関連トピック

- 権限分析ポリシーの有効化

親トピック: 権限分析ポリシーの作成および管理

5.2.4 権限分析ポリシーの作成の例

様々な権限分析ポリシーを作成できます。

- 例: データベース全体の権限の権限分析
  DBMS_PRIVILEGE_CAPTURE.CREATE_CAPTUREは、データベース全体の権限を分析するために使用できます。
- 例: 2つのロールの権限の使用状況の権限分析
  DBMS_PRIVILEGE_CAPTURE.CREATE_CAPTUREプロシージャを使用すると、複数のロールの権限の使用状況を分析できます。
- 例: SQL*Plus使用中の権限の権限分析
  DBMS_PRIVILEGE_CAPTURE.CREATE_CAPTUREプロシージャは、分析用に権限を取得するために使用できます。
- 例: SQL*PlusアクセスのPSMITH権限の権限分析
  DBMS_PRIVILEGE_CAPTURE.CREATE_CAPTUREを使用すると、ユーザーがSQL*Plusを実行する際に、ユーザー・アクセスを分析できます。

親トピック: 権限分析ポリシーの作成および管理

5.2.4.1 例: データベース全体の権限の権限分析

DBMS_PRIVILEGE_CAPTURE.CREATE_CAPTUREは、データベース全体の権限を分析するために使用できます。

例5-1は、データベースのすべての権限の使用を記録するために、DBMS_PRIVILEGE_CAPTUREパッケージを使用して権限分析ポリシーを作成する方法を示しています。

例5-1 データベース全体の権限の権限分析

```
BEGIN
 DBMS_PRIVILEGE_CAPTURE.CREATE_CAPTURE(
  name          => 'db_wide_capture_pol',
  description   => 'Captures database-wide privileges',
  type          => DBMS_PRIVILEGE_CAPTURE.G_DATABASE);
END;
/
```

親トピック: 権限分析ポリシーの作成の例

5.2.4.2 例: 2つのロールの権限の使用状況の権限分析

DBMS_PRIVILEGE_CAPTURE.CREATE_CAPTUREプロシージャを使用すると、複数のロールの権限の使用状況を分析できます。

例5-2は、2つのロールの権限の使用状況の分析方法を示しています。

例5-2 2つのロールの権限の使用状況の権限分析

```
BEGIN
 DBMS_PRIVILEGE_CAPTURE.CREATE_CAPTURE(
```

```
  name          => 'dba_roles_capture_pol',
  description   => 'Captures DBA and LBAC_DBA role use',
  type          => DBMS_PRIVILEGE_CAPTURE.G_ROLE,
  roles         => role_name_list('dba', 'lbac_dba'));
END;
/
```

親トピック: 権限分析ポリシーの作成の例

5.2.4.3 例: SQL*Plus使用中の権限の権限分析

DBMS_PRIVILEGE_CAPTURE.CREATE_CAPTUREプロシージャは、分析用に権限を取得するために使用できます。

例5-3は、SQL*Plusの実行に使用される権限の分析方法を示しています。

例5-3 SQL*Plus使用中の権限の権限分析

```
BEGIN
 DBMS_PRIVILEGE_CAPTURE.CREATE_CAPTURE(
  name             => 'sqlplus_capture_pol',
  description      => 'Captures privilege use during SQL*Plus use',
  type             => DBMS_PRIVILEGE_CAPTURE.G_CONTEXT,
  condition        => 'SYS_CONTEXT(''USERENV'', ''MODULE'')=''sqlplus''');
END;
/
```

親トピック: 権限分析ポリシーの作成の例

5.2.4.4 例: SQL*PlusアクセスセスPSMITH権限の権限分析

DBMS_PRIVILEGE_CAPTURE.CREATE_CAPTUREを使用すると、ユーザーがSQL*Plusを実行する際に、ユーザー・アクセスを分析できます。

例5-4は、SQL*Plusの実行時にセッション・ユーザーPSMITHにより使用される権限の分析方法を示しています。

例5-4 SQL*PlusアクセスセスPSMITH権限の権限分析

```
BEGIN
 DBMS_PRIVILEGE_CAPTURE.CREATE_CAPTURE(
  name          => 'psmith_sqlplus_analysis_pol',
  description   => 'Analyzes PSMITH role priv use for SQL*Plus module',
  type          => DBMS_PRIVILEGE_CAPTURE.G_CONTEXT,
  condition     => 'SYS_CONTEXT(''USERENV'', ''MODULE'')=''sqlplus''
                    AND SYS_CONTEXT(''USERENV'', ''SESSION_USER'')=''PSMITH''');
END;
/
```

親トピック: 権限分析ポリシーの作成の例

5.2.5 権限分析ポリシーの有効化

権限分析ポリシーを作成したら、そのポリシーを有効化して権限の使用を取得する必要があります。

DBMS_PRIVILEGE_CAPTURE.ENABLE_CAPTUREプロシージャは、権限ポリシーを有効化し、その取得実行名を作成します。実行名では、取得が行われる期間を定義します。

1. CAPTURE_ADMINロールを持つユーザーとして、データベース・インスタンスにログインします。
2. 既存の権限分析ポリシーおよびそれらが現在有効かどうかを確認するために、DBA_PRIV_CAPTURESデータ・ディクショナリ・ビューのNAMEおよびENABLED列に問い合せます。
3. DBMS_PRIVILEGE_CAPTURE.ENABLE_CAPTUREプロシージャを実行して、ポリシーを有効にし、必要に応じて

取得実行の名前を作成します。

たとえば、権限分析ポリシーlogon_users_analysisを有効にするには、次のようにします。

```
BEGIN
  DBMS_PRIVILEGE_CAPTURE.ENABLE_CAPTURE (
    name      => 'logon_users_analysis_pol',
    run_name => 'logon_users_04092016');
END;
/
```

run_nameパラメータを指定する必要がない場合は、次のように名前のみを指定してポリシーを有効化できます。

```
EXEC DBMS_PRIVILEGE_CAPTURE.ENABLE_CAPTURE ('logon_users_analysis_pol');
```

親トピック: 権限分析ポリシーの作成および管理

5.2.6 権限分析ポリシーの無効化

権限分析レポートを生成する前に、権限分析ポリシーを無効化する必要があります。

ポリシーを無効化すると、権限は記録されなくなります。権限分析ポリシーの無効化は、権限分析ポリシーの無効化の前後にログオンしたユーザーに対して直接有効になります。権限分析ポリシーを無効にするには、DBMS_PRIVILEGE_CAPTURE.DISABLE_CAPTUREプロシージャを使用できます。

1. CAPTURE_ADMINロールを持つユーザーとして、データベース・インスタンスにログインします。
2. 既存の権限分析ポリシーおよびそれらが現在無効かどうかを確認するために、DBA_PRIV_CAPTURESデータ・ディクショナリ・ビューのNAMEおよびENABLED列に問い合せます。
3. DBMS_PRIVILEGE_CAPTURE.DISBLE_CAPTUREプロシージャを実行して、ポリシーを有効にします。

   たとえば、権限分析ポリシーlogon_users_analysisを無効にするには、次のようにします。

   ```
   EXEC DBMS_PRIVILEGE_CAPTURE.DISABLE_CAPTURE ('logon_users_analysis_pol');
   ```

親トピック: 権限分析ポリシーの作成および管理

5.2.7 権限分析レポートの生成

DBMS_PRIVILEGE_CAPTURE PL/SQLパッケージを使用すると、Enterprise Manager Cloud ControlまたはSQL*Plusのいずれかで、権限分析ポリシー・レポートを生成できます。

- 権限分析レポートの生成について
  権限分析ポリシーが無効になると、権限分析ポリシー用に作成した取得実行に基づいてレポートを生成できます。
- 複数の名前付き取得実行の管理に関する一般プロセス
  権限分析ポリシーを有効にすると、そのポリシーの結果に対して名前付き取得実行を作成できます。
- DBMS_PRIVILEGE_CAPTUREによる権限分析レポートの生成
  DBMS_PRIVILEGE_CAPTURE.GENERATE_RESULTプロシージャは、権限取得結果を示すレポートを生成します。
- Cloud Controlによる権限分析レポートの生成
  Cloud Controlを使用すると、権限分析レポートを生成できます。
- Cloud Controlによる権限分析レポートへのアクセス
  権限分析レポートは、使用されている権限と使用されていない権限の両方に関する情報を提供します。

親トピック: 権限分析ポリシーの作成および管理

5.2.7.1 権限分析レポートの生成について

権限分析ポリシーが無効になると、権限分析ポリシー用に作成した取得実行に基づいてレポートを生成できます。

レポート結果をSQL*Plusに表示するには、権限分析固有のデータ・ディクショナリ・ビューを問い合せます。Enterprise Manager Cloud Controlでは、「権限分析」ページの「アクション」メニューからレポートを表示できます。権限分析プロセス中に権限を使用してレポートを生成する前に取り消す場合、権限付与パスなしで権限が使用された権限として引き続きレポートされます。

関連トピック

- 権限分析ポリシーおよびレポート・データ・ディクショナリ・ビュー

親トピック: 権限分析レポートの生成

5.2.7.2 複数の名前付き取得実行を管理するための一般的なプロセス

権限分析ポリシーを有効にすると、そのポリシーの結果に対して名前付き取得実行を作成できます。

取得実行では、取得が有効になる(開始される)ときから無効になる(停止される)ときまでの期間を定義します。このように、複数の実行を作成した後、権限取得結果の生成時にそれらを比較できます。

複数の名前付き取得実行を管理するための一般的なプロセスは、次のとおりです。

1. ポリシーを作成します。
2. 最初の実行のためにポリシーを有効にします。
3. ユーザー挙動データの収集期間の後、このポリシーおよびその実行を無効にします。
4. 結果を生成してから、この取得実行に関する情報を権限分析データ・ディクショナリ・ビューに問い合せます。

   `run_name`パラメータを`DBMS_PRIVILEGE_CAPTURE.GENERATE_RESULT`プロシージャから省略した場合、このプロシージャは、すべてのレコードをまとめて参照し、それらを分析します。
5. 2番目の実行のためにポリシーを再度有効にします。ポリシーが最初に無効になっていないと、新しい取得実行を作成できません。
6. ユーザー・データを収集した後、ポリシーおよび2番目の実行を無効にします。
7. 結果を生成します。
8. 権限分析データ・ディクショナリ・ビューを問い合せます。両方の取得実行からの結果が、ビューに表示されます。取得実行のどちらかの結果のみを表示する必要がある場合は、結果を再度生成し、権限分析ビューを再度問合せできます。実行名で結果をフィルタすることもできます。

有効にすると、条件を満たす場合に権限分析ポリシーが権限の使用状況の記録を開始します。任意の時点で、データベースの1つの権限分析ポリシーのみ有効化できます。唯一の例外は、タイプ`DBMS_PRIVILEGE_CAPTURE.G_DATABASE`の権限分析ポリシーを異なるタイプの権限分析と同時に有効化できることです。

権限分析ポリシーを削除すると、その関連付けられた取得実行は同様に削除され、権限分析データ・ディクショナリ・ビューに反映されません。

データベースの再起動は、権限分析のステータスを変更しません。たとえば、データベースの停止前に権限分析ポリシーを有効化すると、このポリシーは、データベースの停止後および再起動後に継続して有効です。

関連トピック

- チュートリアル: 取得実行の使用によるANY権限使用の分析

親トピック: 権限分析レポートの生成

5.2.7.3 DBMS_PRIVILEGE_CAPTUREによる権限分析レポートの生成

DBMS_PRIVILEGE_CAPTURE.GENERATE_RESULTプロシージャは、権限取得結果を示すレポートを生成します。

1. CAPTURE_ADMINロールを持つユーザーとして、データベース・インスタンスにログインします。
2. 既存の権限分析ポリシーおよびそれらが現在無効かどうかを確認するために、DBA_PRIV_CAPTURESデータ・ディクショナリ・ビューのNAMEおよびENABLED列に問い合せます。

   権限分析ポリシーは、それに関する権限分析レポートを生成する前に無効化する必要があります。

3. 次の構文を使用してDBMS_PRIVILEGE_CAPTURE.GENERATE_RESULTプロシージャを実行します。

```
DBMS_PRIVILEGE_CAPTURE.GENERATE_RESULT(
  name          VARCHAR2,
  run_name      VARCHAR2 DEFAULT NULL,
  dependency    BOOLEAN DEFAULT NULL);
```

   詳細は、次のとおりです。

   - name: 権限分析ポリシーの名前を指定します。DBA_PRIV_CAPTURESデータ・ディクショナリ・ビューには、既存のポリシーの名前が示されます。
   - run_name: 計算する必要がある権限取得の実行名を指定します。この設定を省略すると、指定した権限取得のすべての実行が計算されます。
   - dependency: Y (はい)またはN (いいえ)を入力して、PL/SQL計算権限使用状況をレポートに含める必要があるかどうかを指定します。

   たとえば、権限分析ポリシーlogon_users_analysisのレポートを生成するには、次のようにします。

```
EXEC DBMS_PRIVILEGE_CAPTURE.GENERATE_RESULT ('logon_users_analysis');
```

4. 権限付与パスを持つDBA_USED_*データ・ディクショナリ・ビューで、使用された権限を問い合せます。

親トピック: 権限分析レポートの生成

5.2.7.4 Cloud Controlによる権限分析レポートの生成

Cloud Controlを使用すると、権限分析レポートを生成できます。

1. CAPTURE_ADMINロールおよびSELECT ANY DICTIONARY権限を付与されているユーザーとして、Cloud Controlにログインします。『Oracle Database 2日でデータベース管理者』で、ログイン方法を説明しています。
2. 「セキュリティ」メニューから、「権限分析」を選択します。
3. 「ポリシー」から、レポートを生成するポリシーを選択します。
4. 「レポートの生成」を選択します。
5. 「権限分析: レポートの生成」ダイアログ・ボックスで、レポートを生成する時間を指定します。

   レポートを今すぐに生成する場合は、「即時」を選択します。レポートを後で生成するには、「後で」を選択してから、レポートを生成する時、分、秒およびタイムゾーンを指定します。

6. 「OK」をクリックします。

   「権限分析」ページに確認メッセージ表示されて、レポートが送信されたことを通知します。このページは、ジョブが完了

するまでリフレッシュできます。レポートを表示するには、ポリシー名を選択してから「レポートの表示」をクリックします。

親トピック: 権限分析レポートの生成

5.2.7.5 Cloud Controlによる権限分析レポートへのアクセス

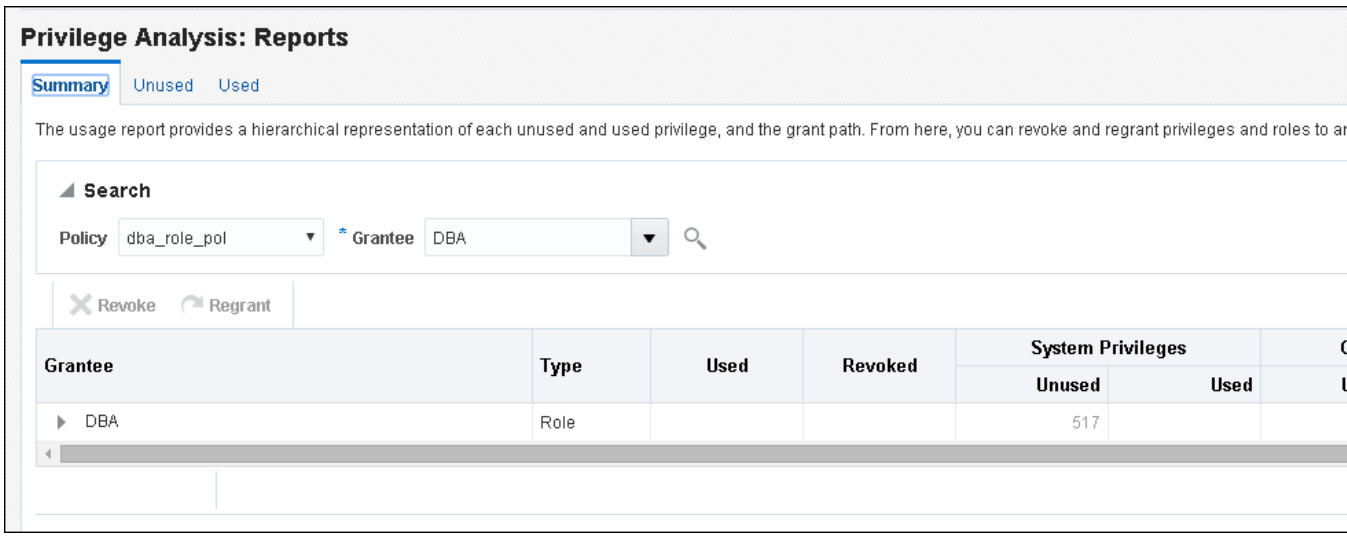
権限分析レポートは、使用されている権限と使用されていない権限の両方に関する情報を提供します。

1. 権限分析レポートの生成。

   詳細は、「Cloud Controlによる権限分析レポートの生成」を参照してください。

2. 「権限分析」ページで、レポートを生成したポリシーを選択します。
3. 「レポートの表示」を選択します。

   「権限分析レポート」ページが表示されます。



4. レポートを表示するには、次のようにします。
   - デフォルトでは選択されているレポートが表示されますが、別のポリシーのレポートを検索するには、「検索」リージョンを使用して、別のレポートを見つけるか現在選択しているポリシーの別の権限受領者を選択します。
   - 未使用の権限を表示するには、「未使用」タブを選択します。使用した権限を表示するには、「使用済」を選択します。両方のサマリーを表示するには、「サマリー」を選択します。

   ここから、必要に応じてユーザーに対して取消しや再付与を行うロールを選択できます。これを行うには、「権限受領者」の下でロールを選択してから、「取消」または「再付与」をクリックします。

親トピック: 権限分析レポートの生成

5.2.8 権限分析ポリシーの削除

権限分析ポリシーを削除するには、そのポリシーを無効にする必要があります。

権限分析ポリシーの削除は、この権限分析に関連付けられているすべての使用されたおよび使用されていない権限レコードも削除します。ポリシーの取得実行を作成した場合、それらはポリシーの削除時に削除されます。

1. CAPTURE_ADMINロールを持つユーザーとして、データベース・インスタンスにログインします。
2. ポリシーおよびポリシーが有効か無効かを確認するために、DBA_PRIV_CAPTURESデータ・ディクショナリ・ビューのNAMEおよびENABLE列に問い合せます。
3. ポリシーが有効な場合、無効にします。

たとえば:

```
EXEC DBMS_PRIVILEGE_CAPTURE.DISABLE_CAPTURE ('logon_users_analysis_pol');
```

4. DBMS_PRIVILEGE_CAPTURE.DROP_CAPTUREプロシージャを実行して、ポリシーを削除します。

   たとえば:

   ```
   EXEC DBMS_PRIVILEGE_CAPTURE.DROP_CAPTURE ('logon_users_analysis_pol');
   ```

   取得実行を含むポリシーを有効にした場合は、その取得実行も削除されます。取得実行を個別に削除するには、DBMS_PRIVILEGE_CAPTURE.DELETE_RUNプロシージャを実行します。ただし、この文を実行する前に、そのポリシーが存在する必要があります。

関連トピック

- 権限分析ポリシーの無効化

親トピック: 権限分析ポリシーの作成および管理

5.3 Cloud Controlによるロールの作成および権限の管理

権限分析レポートで検出された権限を使用して新規ロールを作成し、次にこのロールをユーザーに付与できます。

- Cloud Controlでの権限分析レポートからのロールの作成
  レポート・サマリーを使用して、アプリケーションで必要とされる最低限の権限を検索し、これらの権限をロールにカプセル化できます。
- Cloud Controlによるロールおよび権限の取消しおよび再付与
  Enterprise Manager Cloud Controlを使用して、ロールおよび権限をユーザーに対して取り消したり、再付与したりできます。
- Cloud Controlによる取消しまたは再付与スクリプトの生成
  権限分析レポートの結果に基づいて、ユーザーに対して権限を取り消したり、再付与するスクリプトを生成できます。

親トピック: 権限分析の実行による権限使用の特定

5.3.1 Cloud Controlでの権限分析レポートからのロールの作成

レポート・サマリーを使用して、アプリケーションで必要とされる最低限の権限を検索し、これらの権限をロールにカプセル化できます。

1. CAPTURE_ADMINロールおよびSELECT ANY DICTIONARY権限を付与されているユーザーとして、Cloud Controlにログインします。『Oracle Database 2日でデータベース管理者』で、ログイン方法を説明しています。
2. 「権限分析」ページで、ポリシー名を選択してから、「アクション」メニューの「ロールの作成」をクリックします。
3. 「ロールの作成」ページで、次の詳細情報を入力してから、「OK」をクリックします。
   - 新しいロールの作成元となるポリシーを選択します。
   - 作成する新しいロールの一意の名前を入力します。
   - ロールで何をカプセル化するかに応じて、「使用済」または「未使用」チェック・ボックスを選択します。ロールには、使用中または未使用のシステムおよびオブジェクトの権限およびロールを使用できます。
   - 「直接付与されたシステム権限」、「直接付与されたオブジェクト権限」および「直接付与されたロール」の対応するラジオ・ボタンを選択します。

たとえば、「使用済」チェック・ボックスを選択し、次を選択します。

- すべてのシステム権限: 取得されたシステム権限のうち使用中のものすべてが、作成する新しいロールに含まれます。
- ロールなし: ポリシーで取得されたロールはどれも、新しいロールでは使用されません。
- オブジェクト権限のカスタマイズ: 取得された使用中のオブジェクト権限が一覧表示されるので、ロールに割り当てる権限をこのリストから選択する必要があります。

親トピック: Cloud Controlによるロールの作成および権限の管理

5.3.2 Cloud Controlによるロールおよび権限の取消しおよび再付与

Enterprise Manager Cloud Controlを使用して、ロールおよび権限をユーザーに対して取り消したり、再付与したりできます。

1. Oracle Database Vaultが有効な場合、Oracleシステム権限およびロール管理レルムの所有者として認可されていることを確認します。

   SQL*Plusでは、DV_OWNERロールを付与されているユーザーがDBA_DV_REALM_AUTHデータ・ディクショナリ・ビューを問い合せることで認可を確認できます。ユーザーに認可を付与するには、DBMS_MACADM.ADD_AUTH_TO_REALMプロシージャを使用します。

2. 権限分析レポートの生成。
3. 「権限分析」ページで、レポートを生成したポリシーを選択します。
4. 「レポートの表示」を選択します。
5. 「権限分析: レポート」ページで、サマリー」タブを選択します。
6. 「検索」の下で、「ポリシー」および「権限受領者」メニュー・オプションが設定されていることを確認します。
7. 「権限受領者」領域の下で、権限受領者オプションを展開します。
   たとえば、HR_ADMINロールというロールのロール権限分析レポートの場合は、HR_ADMINロールを展開して、それに関連付けられている権限を表示します。
8. 取り消す各権限を選択してから「取消」をクリックするか、「再付与」を選択して権限をロールに付与します。

関連トピック

- Cloud Controlによる権限分析レポートの生成

親トピック: Cloud Controlによるロールの作成および権限の管理

5.3.3 Cloud Controlによる取消しまたは再付与スクリプトの生成

権限分析レポートの結果に基づいて、ユーザーに対して権限を取り消したり、再付与するスクリプトを生成できます。

- 取消しおよび再付与スクリプトの生成について
  システムやオブジェクトに対する未使用の権限およびロールの一括取消しを実行するためのスクリプトを、権限分析の生成後にダウンロードできます。
- 取消しスクリプトの生成
  Enterprise Manager Cloud Controlを使用すると、ユーザーから権限を取り消すスクリプトを生成できます。
- 再付与スクリプトの生成
  Enterprise Manager Cloud Controlを使用すると、ユーザーから取り消した権限を再付与するスクリプトを生成できます。

親トピック: Cloud Controlによるロールの作成および権限の管理

5.3.3.1 取消しおよび再付与スクリプトの生成について

システムやオブジェクトに対する未使用の権限およびロールの一括取消しを実行するためのスクリプトを、権限分析の生成後にダウンロードできます。

後で、これらの権限をユーザーに再付与することが必要になった場合は、再付与スクリプトを生成できます。再付与スクリプトを生成するには、対応する取消しスクリプトが必要です。

開発環境またはテスト環境で取消スクリプトを実行します。Oracleによって提供されているアカウントおよびロールから権限およびロールを取り消すことはできないということを知っておいてください。

親トピック: Cloud Controlによる取消しまたは再付与スクリプトの生成

5.3.3.2 取消しスクリプトの生成

Enterprise Manager Cloud Controlを使用すると、ユーザーから権限を取り消すスクリプトを生成できます。

1. Oracle Database Vaultが有効な場合、Oracleシステム権限およびロール管理レルムの所有者として認可されていることを確認します。

   SQL*Plusでは、DV_OWNERロールを付与されているユーザーがDBA_DV_REALM_AUTHデータ・ディクショナリ・ビューを問い合せることで認可を確認できます。ユーザーに認可を付与するには、DBMS_MACADM.ADD_AUTH_TO_REALMプロシージャを使用します。

2. Enterprise Managerで、CAPTURE_ADMINロールおよびSELECT ANY DICTIONARY権限が付与されているユーザーとしてターゲット・データベースのホームページにアクセスします。

   詳細は、『Oracle Database 2日でデータベース管理者』を参照してください。

3. 「セキュリティ」メニューから、「権限分析」を選択します。
4. 必要な権限分析レポートが生成されたことを確認します。
5. 「権限分析」ページの「アクション」メニューで「取消スクリプト」を選択します。
6. 「取消スクリプト」ページで、「生成」をクリックします。

   取消しスクリプトの詳細を生成するためのウィザードが表示されます。

7. 「スクリプト詳細」ページで、取消スクリプトを準備する必要のあるポリシーの名前を「ポリシー名」メニューから選択します。
8. 「スクリプト名」フィールドに一意の名前を入力し、「説明」にスクリプトの説明を入力します。

   たとえば、未使用の権限をすべて取り消す場合は、未使用のすべての権限およびロールを表す「すべて」オプションを選択して、「次へ」をクリックします。

   選択内容に基づいて、取消し対象の使用可能な権限、未使用のすべての権限、オブジェクト権限、およびロールがそれぞれのページに表示されます。

9. 「権限受領者(ユーザー/ロール)」で、「すべて」または「カスタマイズ」を選択します。
10. 「未使用のシステム権限」、「未使用のオブジェクト権限」および「未使用のロール」設定で、「すべて」、なしまたは「カスタマイズ」を選択します。
11. 「次へ」をクリックします。

    表示される次のページは、「すべて」、なしまたは「カスタマイズ」のどれを選択したかで異なります。「すべて」を選択した場合、ページには権限のリストが表示されます。なしを選択した場合、ページは省略されます。「カスタマイズ」を選択した場合は、取り消す権限を個別に選択できます。表示される最後のページは、「確認」ページです。

12. 「保存」をクリックします。

「取消スクリプト」ページが表示されます。

13. 「取消スクリプト」ページで、新しく作成したSQLスクリプトを選択してから、「取消スクリプトのダウンロード」をクリックしてこのスクリプトをダウンロードします。これには、各権限またはロールのREVOKE SQL文が含まれています。

スクリプトを表示するには、「取消スクリプトの表示」ボタンをクリックします。

14. 「権限分析」ページに戻るには、「戻る」をクリックします。

関連トピック

- Cloud Controlによる権限分析レポートの生成

親トピック: Cloud Controlによる取消しまたは再付与スクリプトの生成

5.3.3.3 再付与スクリプトの生成

Enterprise Manager Cloud Controlを使用すると、ユーザーから取り消した権限を再付与するスクリプトを生成できます。

1. Oracle Database Vaultが有効な場合、Oracleシステム権限およびロール管理レルムの所有者として認可されていることを確認します。

   SQL*Plusでは、DV_OWNERロールを付与されているユーザーがDBA_DV_REALM_AUTHデータ・ディクショナリ・ビューを問い合せることで認可を確認できます。ユーザーに認可を付与するには、DBMS_MACADM.ADD_AUTH_TO_REALMプロシージャを使用します。

2. Enterprise Managerで、CAPTURE_ADMINロールおよびSELECT ANY DICTIONARY権限が付与されているユーザーとしてターゲット・データベースのホームページにアクセスします。

   詳細は、『Oracle Database 2日でデータベース管理者』を参照してください。

3. 「セキュリティ」メニューから、「権限分析」を選択します。
4. 目的のレポートが生成済であることを確認します。

   詳細は、「Cloud Controlによる権限分析レポートの生成」を参照してください。

5. 「権限分析」ページで、取消スクリプトが基づいているポリシーを選択します。
6. 「アクション」メニューから、「取消スクリプト」を選択します。
7. 「取消スクリプト」ページで、前に作成したポリシーの名前を選択してから、「再付与スクリプトのダウンロード」をクリックしてこのスクリプトをダウンロードします。

   「取消スクリプトの表示」ボタンおよび「再付与スクリプトの表示」ボタンを選択することで、ポリシーに関連付けられているスクリプトを表示できます。

関連トピック

- Cloud Controlによる権限分析レポートの生成

親トピック: Cloud Controlによる取消しまたは再付与スクリプトの生成

5.4 チュートリアル: 取得実行の使用によるANY権限使用の分析

このチュートリアルでは、取得実行を作成してREAD ANY TABLEシステム権限の使用を分析する方法を示します。

- ステップ1: ユーザー・アカウントの作成

2つのユーザーを作成する必要があります。1つはポリシーを作成するユーザーで、もう1つはその権限の使用が分析対象となるユーザーです。

- ステップ2: 権限分析ポリシーの作成および有効化
  ユーザーpa_adminは権限分析ポリシーを作成して有効にする必要があります。
- ステップ3: READ ANY TABLEシステム権限の使用
  ユーザーapp_userとして、READ ANY TABLEシステム権限を使用します。
- ステップ4: 権限分析ポリシーの無効化
  ユーザーapp_userのアクションを取得するレポートを生成するには、ポリシーを無効にする必要があります。
- ステップ5: 権限分析レポートの生成および表示
  権限分析ポリシーを無効化すると、ユーザーpa_adminは、権限分析レポートを生成し表示できます。
- ステップ6: 2番目の取得実行の作成
  これで、いつでもANY_priv_analysis_pol権限分析ポリシーの2番目の取得実行を作成できます。
- ステップ7: このチュートリアルのコンポーネントの削除
  コンポーネントが不要になった場合、このチュートリアルで作成したコンポーネントを削除できます。

親トピック: 権限分析の実行による権限使用の特定

5.4.1 ステップ1: ユーザー・アカウントの作成

2つのユーザーを作成する必要があります。1つはポリシーを作成するユーザーで、もう1つはその権限の使用が分析対象となるユーザーです。

1. CREATE USERシステム権限を持つユーザーとして、データベース・インスタンスにログインします。

   たとえば:

   ```
   sqlplus sec_admin
   Enter password: password
   ```

   マルチテナント環境で、適切なプラガブル・データベース(PDB)に接続する必要があります。

   たとえば:

   ```
   sqlplus sec_admin@hrpdb
   Enter password: password
   ```

   利用可能なPDBを検索するには、DBA_PDBSデータ・ディクショナリ・ビューを問い合せます。現在のPDBを確認するには、show con_nameコマンドを実行します。

2. 次のユーザーを作成します。

   ```
   CREATE USER pa_admin IDENTIFIED BY password;
   CREATE USER app_user IDENTIFIED BY password;
   ```

3. ロールおよびシステム権限を他のユーザーに付与する権限を持ち、Oracleシステム権限およびロール管理レルムの所有者認可を付与されているユーザーとして接続します。(ユーザーSYSはデフォルトでこれらの権限を持っています。)

   たとえば:

   ```
   CONNECT dba_psmith -- Or, CONNECT dba_psmith@hrpdb
   Enter password: password
   ```

   SQL*Plusでは、DV_OWNERロールを付与されているユーザーがDBA_DV_REALM_AUTHデータ・ディクショナリ・ビューを問い合せることで認可を確認できます。ユーザーに認可を付与するには、

DBMS_MACADM.ADD_AUTH_TO_REALMプロシージャを使用します。

4. 次のロールおよび権限をユーザーに付与します。

```
GRANT CREATE SESSION, CAPTURE_ADMIN TO pa_admin;
GRANT CREATE SESSION, READ ANY TABLE TO app_user;
```

ユーザーpa_adminは、ユーザーapp_userが実行するREAD ANY TABLE問合せを分析する権限分析ポリシーを作成します。

親トピック: チュートリアル: 取得実行の使用によるANY権限使用の分析

5.4.2 ステップ2: 権限分析ポリシーの作成および有効化

ユーザーpa_adminは権限分析ポリシーを作成して有効にする必要があります。

1. ユーザーpa_adminとして接続します。

```
CONNECT pa_admin -- Or, CONNECT pa_admin@hrpdb
Enter password: password
```

2. 次の権限分析ポリシーを作成します。

```
BEGIN
 DBMS_PRIVILEGE_CAPTURE.CREATE_CAPTURE(
  name           => 'ANY_priv_analysis_pol',
  description    => 'Analyzes system privilege use',
  type           => DBMS_PRIVILEGE_CAPTURE.G_CONTEXT,
  condition      => 'SYS_CONTEXT(''USERENV'', ''SESSION_USER'')=''APP_USER''');
END;
/
```

この例では、次のようになります。

- typeは、次に説明するconditionパラメータで定義される取得条件のタイプを指定します。このポリシーでは、タイプはコンテキスト・ベースの条件です。
- conditionは、ポリシーを有効にするためにTRUEに評価する必要があるBoolean式を使用して条件を指定します。この場合、セッション・ユーザーがapp_userかどうかを条件でチェックします。

3. ポリシーを有効にし、それに対する取得実行を作成します。

```
BEGIN
  DBMS_PRIVILEGE_CAPTURE.ENABLE_CAPTURE (
   name       => 'ANY_priv_analysis_pol',
   run_name   => 'ANY_priv_pol_run_1');
END;
/
```

この時点で、ポリシーはユーザーapp_userのアクションの記録を開始できます。

親トピック: チュートリアル: 取得実行の使用によるANY権限使用の分析

5.4.3 ステップ3: READ ANY TABLEシステム権限の使用

ユーザーapp_userとして、READ ANY TABLEシステム権限を使用します。

1. ユーザーapp_userとして接続します。

```
CONNECT app_user -- Or, CONNECT app_user@hrpdb
Enter password: password
```

2. HR.EMPLOYEES表を問い合せます。

```
SELECT FIRST_NAME, LAST_NAME, SALARY FROM HR.EMPLOYEES WHERE SALARY > 12000
ORDER BY SALARY DESC;
FIRST_NAME           LAST_NAME                     SALARY
-------------------- ------------------------- ----------
Steven               King                           24000
Neena                Kochhar                        17000
Lex                  De Haan                        17000
John                 Russell                        14000
Karen                Partners                       13500
Michael              Hartstein                      13000
Shelley              Higgins                        12008
Nancy                Greenberg                      12008
```

親トピック: チュートリアル: 取得実行の使用によるANY権限使用の分析

5.4.4 ステップ4: 権限分析ポリシーの無効化

ユーザーapp_userのアクションを取得するレポートを生成するには、ポリシーを無効にする必要があります。

1. ユーザーpa_adminとして接続します。

```
CONNECT pa_admin -- Or, CONNECT pa_admin@hrpdb
Enter password: password
```

2. ANY_priv_analysis_pol権限ポリシーを無効にします。

```
EXEC DBMS_PRIVILEGE_CAPTURE.DISABLE_CAPTURE ('ANY_priv_analysis_pol');
```

親トピック: チュートリアル: 取得実行の使用によるANY権限使用の分析

5.4.5 ステップ5: 権限分析レポートの生成および表示

権限分析ポリシーを無効にすると、ユーザーpa_adminは権限分析レポートの生成および表示を行うことができます。

1. ユーザーpa_adminとして、権限分析の結果を生成します。

```
BEGIN
  DBMS_PRIVILEGE_CAPTURE.GENERATE_RESULT (
    name      => 'ANY_priv_analysis_pol',
    run_name  => 'ANY_priv_pol_run_1');
END;
/
```

生成された結果は、権限分析データ・ディクショナリ・ビューに格納されます。

2. 次のコマンドを入力して、データ・ディクショナリ・ビューの出力の書式を設定します。

```
col username format a10
col sys_priv format a16
col object_owner format a13
col object_name format a23
col run_name format a27
```

3. app_userで使用されるシステム権限と、権限分析中に使用されるオブジェクトを確認します。

```
SELECT SYS_PRIV, OBJECT_OWNER, OBJECT_NAME, RUN_NAME FROM DBA_USED_PRIVS WHERE
USERNAME = 'APP_USER';
```

次のような出力結果が表示されます。最初の行は、app_userがHR.EMPLOYEES表に対するREAD ANY

TABLE権限を使用したことを示します。

```
SYS_PRIV         OBJECT_OWNER  OBJECT_NAME              RUN_NAME
---------------- ------------ ---------------------- -----------------
                 SYSTEM        PRODUCT_PRIVS           ANY_PRIV_POL_RUN_1
                 SYS           DUAL                    ANY_PRIV_POL_RUN_1
                 SYS           DUAL                    ANY_PRIV_POL_RUN_1
CREATE SESSION                                         ANY_PRIV_POL_RUN_1
                 SYS           DBMS_APPLICATION_INFO   ANY_PRIV_POL_RUN_1
READ ANY TABLE   HR            EMPLOYEES               ANY_PRIV_POL_RUN_1
```

この段階では、権限分析結果は、将来追加で取得実行を作成する場合でも、権限分析データ・ディクショナリ・ビューで引き続き利用可能です。

親トピック: チュートリアル: 取得実行の使用によるANY権限使用の分析

5.4.6 ステップ6: 2番目の取得実行の作成

これで、いつでもANY_priv_analysis_pol権限分析ポリシーの2番目の取得実行を作成できます。

1. ユーザーpa_adminとして、ANY_priv_analysis_pol権限分析ポリシーを有効にして取得実行ANY_priv_pol_run_1を使用します。

```
BEGIN
  DBMS_PRIVILEGE_CAPTURE.ENABLE_CAPTURE (
   name       => 'ANY_priv_analysis_pol',
   run_name   => 'ANY_priv_pol_run_2');
END;
/
```

2. ユーザーapp_userとして接続します。

```
CONNECT app_user -- Or, CONNECT app_user@hrpdb
Enter password: password
```

3. HR.JOBS表を問い合せます。

```
SELECT MAX_SALARY FROM HR.JOBS WHERE MAX_SALARY > 20000;
```

4. ユーザーpa_adminとして接続します。

```
CONNECT pa_admin -- Or, CONNECT pa_admin@hrpdb
Enter password: password
```

5. ANY_priv_analysis_pol権限ポリシーを無効にします。

```
EXEC DBMS_PRIVILEGE_CAPTURE.DISABLE_CAPTURE ('ANY_priv_analysis_pol');
```

6. 2番目の権限分析レポートを生成します。

```
BEGIN
  DBMS_PRIVILEGE_CAPTURE.GENERATE_RESULT (
    name      => 'ANY_priv_analysis_pol',
    run_name  => 'ANY_priv_pol_run_2');
END;
/
```

7. app_userで使用されるシステム権限と、権限分析中に使用されるオブジェクトを確認します。

```
SELECT SYS_PRIV, OBJECT_OWNER, OBJECT_NAME, RUN_NAME FROM DBA_USED_PRIVS WHERE
USERNAME = 'APP_USER' ORDER BY RUN_NAME;
```

次のような出力が表示されます。これには、ユーザーpa_adminが作成した両方の取得実行の結果が示されています。

```
SYS_PRIV          OBJECT_OWNER  OBJECT_NAME              RUN_NAME
----------------- ------------- ------------------------ ---------------------
READ ANY TABLE    HR            EMPLOYEES                ANY_PRIV_POL_RUN_1
                  SYS           DUAL                     ANY_PRIV_POL_RUN_1
CREATE SESSION                                           ANY_PRIV_POL_RUN_1
                  SYS           DUAL                     ANY_PRIV_POL_RUN_1
                  SYSTEM        PRODUCT_PRIVS            ANY_PRIV_POL_RUN_1
                  SYS           DBMS_APPLICATION_INFO    ANY_PRIV_POL_RUN_1
                  SYS           DUAL                     ANY_PRIV_POL_RUN_2
                  SYS           DBMS_APPLICATION_INFO    ANY_PRIV_POL_RUN_2
                  SYSTEM        PRODUCT_PRIVS            ANY_PRIV_POL_RUN_2
                  SYS           DUAL                     ANY_PRIV_POL_RUN_2
READ ANY TABLE    HR            JOBS                     ANY_PRIV_POL_RUN_2
```

親トピック: チュートリアル: 取得実行の使用によるANY権限使用の分析

5.4.7 ステップ7: この例で使用したコンポーネントの削除

コンポーネントが不要になった場合、このチュートリアルで作成したコンポーネントを削除できます。

1. ユーザーpa_adminとして、ANY_priv_analysis_pol権限分析ポリシーおよびその関連付けられた取得実行を削除します。

   ```
   EXEC DBMS_PRIVILEGE_CAPTURE.DROP_CAPTURE ('ANY_priv_analysis_pol');
   ```

   このポリシーに関連付けられている取得実行は、DBMS_PRIVILEGE_CAPTURE.DROP_CAPTUREプロシージャを実行すると自動的に削除されます。

   pa_adminユーザーのスキーマに作成されたオブジェクトはSYSスキーマにあるため、次のステップで、このユーザー(このオブジェクトを含む)を削除する場合でも、ANY_priv_analysis_pol権限分析ポリシーを手動で削除する必要があります。

2. ユーザー・アカウントを作成したユーザーとして接続します。

   たとえば:

   ```
   CONNECT sec_admin -- Or, CONNECT sec_admin@hrpdb
   Enter password: password
   ```

3. ユーザーpa_adminおよびapp_userを削除します。

   ```
   DROP USER pa_admin CASCADE;
   DROP USER app_user;
   ```

親トピック: チュートリアル: 取得実行の使用によるANY権限使用の分析

5.5 チュートリアル: DBAロールを持つユーザーによる権限の使用の分析

このチュートリアルでは、DBAロールを持ち、データベースのチューニング操作を実行するユーザーの権限の使用を分析する方法を示します。

- ステップ1: ユーザー・アカウントの作成
  2つのユーザーを作成する必要があります。1つは権限分析ポリシーを作成するユーザーで、もう1つはその権限の使用が分析対象となるユーザーです。

- ステップ2: 権限分析ポリシーの作成および有効化
  ユーザーpa_adminは権限分析ポリシーを作成して有効にする必要があります。
- ステップ3: データベース・チューニング操作の実行
  ユーザーtjonesは、DBAロールを使用して、データベースのチューニング操作を実行します。
- ステップ4: 権限分析ポリシーの無効化
  ユーザーtjonesのアクションを取得するレポートを生成するには、ポリシーを無効にする必要があります。
- ステップ5: 権限分析レポートの生成および表示
  権限分析ポリシーを無効化すると、ユーザーpa_adminは、権限分析レポートを生成し表示できます。
- ステップ6: このチュートリアルのコンポーネントの削除
  コンポーネントが不要になった場合、このチュートリアルで作成したコンポーネントを削除できます。

親トピック: 権限分析の実行による権限使用の特定

5.5.1 ステップ1: ユーザー・アカウントの作成

2つのユーザーを作成する必要があります。1つは権限分析ポリシーを作成するユーザーで、もう1つはその権限の使用が分析対象となるユーザーです。

1. CREATE USERシステム権限を持つユーザーとして、データベース・インスタンスにログインします。

   たとえば:

   ```
   sqlplus sec_admin
   Enter password: password
   ```

   マルチテナント環境で、適切なプラガブル・データベース(PDB)にログインする必要があります。

   たとえば:

   ```
   sqlplus sec_admin@hrpdb
   Enter password: password
   ```

   利用可能なPDBを検索するには、DBA_PDBSデータ・ディクショナリ・ビューを問い合せます。現在のPDBを確認するには、show con_nameコマンドを実行します。

2. 次のユーザーを作成します。

   ```
   CREATE USER pa_admin IDENTIFIED BY password;
   CREATE USER tjones IDENTIFIED BY password;
   ```

3. ロールおよびシステム権限を他のユーザーに付与する権限を持ち、Oracleシステム権限およびロール管理レルムの所有者認可を付与されているユーザーとして接続します。(ユーザーSYSはデフォルトでこれらの権限を持っています。)

   たとえば:

   ```
   CONNECT dba_psmith -- Or, CONNECT dba_psmith@hrpdb
   Enter password: password
   ```

   SQL*Plusでは、DV_OWNERロールを付与されているユーザーがDBA_DV_REALM_AUTHデータ・ディクショナリ・ビューを問い合せることで認可を確認できます。ユーザーに認可を付与するには、DBMS_MACADM.ADD_AUTH_TO_REALMプロシージャを使用します。

4. 次のロールおよび権限をユーザーに付与します。

   ```
   GRANT CREATE SESSION, CAPTURE_ADMIN TO pa_admin;
   GRANT CREATE SESSION, DBA TO tjones;
   ```

ユーザーpa_adminは、ユーザーtjonesが実行するデータベースのチューニング操作を分析する権限分析ポリシーを作成します。

親トピック: チュートリアル: DBAロールを持つユーザーによる権限の使用の分析

5.5.2 ステップ2: 権限分析ポリシーの作成および有効化

ユーザーpa_adminは権限分析ポリシーを作成して有効にする必要があります。

1. ユーザーpa_adminとして接続します。

```
CONNECT pa_admin -- Or, CONNECT pa_admin@hrpdb
Enter password: password
```

2. 次の権限分析ポリシーを作成します。

```
BEGIN
 DBMS_PRIVILEGE_CAPTURE.CREATE_CAPTURE(
  name             => 'dba_tuning_priv_analysis_pol',
  description      => 'Analyzes DBA tuning privilege use',
  type             => DBMS_PRIVILEGE_CAPTURE.G_CONTEXT,
  condition        => 'SYS_CONTEXT(''USERENV'', ''SESSION_USER'')=''TJONES''');
END;
/
```

この例では、次のようになります。

- typeは、次に説明するconditionパラメータで定義される取得条件のタイプを指定します。このポリシーでは、タイプはコンテキスト・ベースの条件です。
- conditionは、ポリシーを有効にするためにTRUEに評価する必要があるBoolean式を使用して条件を指定します。この場合、セッション・ユーザーがtjonesかどうかを条件でチェックします。

3. ポリシーを有効にします。

```
EXEC DBMS_PRIVILEGE_CAPTURE.ENABLE_CAPTURE ('dba_tuning_priv_analysis_pol');
```

この時点で、ポリシーはユーザーtjonesのアクションの記録を開始できます。

親トピック: チュートリアル: DBAロールを持つユーザーによる権限の使用の分析

5.5.3 ステップ3: データベース・チューニング操作の実行

ユーザーtjonesは、DBAロールを使用して、データベースのチューニング操作を実行します。

1. ユーザーtjonesとして接続します。

```
CONNECT tjones -- Or, CONNECT tjones@hrpdb
Enter password: password
```

2. 次のスクリプトを実行して、PLAN_TABLE表を作成します。

```
@$ORACLE_HOME/rdbms/admin/utlxplan.sql
```

このスクリプトの場所は、オペレーティング・システムによって異なる場合があります。このスクリプトでは、PLAN_TABLE表をtjonesスキーマに作成します。

3. 次のEXPLAIN PLAN SQL文をHR.EMPLOYEES表で実行します。

```
EXPLAIN PLAN
```

```
SET STATEMENT_ID = 'Raise in Tokyo'
INTO PLAN_TABLE
FOR UPDATE HR.EMPLOYEES
SET SALARY = SALARY * 1.10
WHERE DEPARTMENT_ID =
 (SELECT DEPARTMENT_ID FROM HR.DEPARTMENTS WHERE LOCATION_ID = 110);
```

次に、ユーザーtjonesでHR.EMPLOYEES表を分析します。

4. 次のいずれかのスクリプトを実行して、CHAINED_ROWS表を作成します。

```
@$ORACLE_HOME/rdbms/admin/utlchain.sql
```

または

```
@$ORACLE_HOME/rdbms/admin/utlchn1.sql
```

5. ANALYZE TABLE文をHR.EMPLOYEES表で実行します。

```
ANALYZE TABLE HR.EMPLOYEES LIST CHAINED ROWS INTO CHAINED_ROWS;
```

親トピック: チュートリアル: DBAロールを持つユーザーによる権限の使用の分析

5.5.4 ステップ4: 権限分析ポリシーの無効化

ユーザーtjonesのアクションを取得するレポートを生成するには、ポリシーを無効にする必要があります。

1. ユーザーpa_adminとして接続します。

```
CONNECT pa_admin -- Or, CONNECT pa_admin@hrpdb
Enter password: password
```

2. dba_tuning_priv_analysis_pol権限ポリシーを無効にします。

```
EXEC DBMS_PRIVILEGE_CAPTURE.DISABLE_CAPTURE ('dba_tuning_priv_analysis_pol');
```

親トピック: チュートリアル: DBAロールを持つユーザーによる権限の使用の分析

5.5.5 ステップ5: 権限分析レポートの生成および表示

権限分析ポリシーを無効にすると、ユーザーpa_adminは権限分析レポートの生成および表示を行うことができます。

1. ユーザーpa_adminとして、権限分析の結果を生成します。

```
EXEC DBMS_PRIVILEGE_CAPTURE.GENERATE_RESULT ('dba_tuning_priv_analysis_pol');
```

生成された結果は、権限分析データ・ディクショナリ・ビューに格納されます。

2. 次のコマンドを入力して、データ・ディクショナリ・ビューの出力の書式を設定します。

```
col username format a8
col sys_priv format a18
col used_role format a20
col path format a150
col obj_priv format a10
col object_owner format a10
col object_name format a10
col object_type format a10
```

3. 権限分析中にtjonesで使用されるシステム権限とロールを確認します。

```
SELECT USERNAME, SYS_PRIV, USED_ROLE, PATH
```

```
 FROM DBA_USED_SYSPRIVS_PATH
 WHERE USERNAME = 'TJONES'
 ORDER BY 1, 2, 3;
```

次のような出力が表示されます。

```
USERNAME SYS_PRIV           USED_ROLE
-------- ------------------ --------------------
PATH
--------------------------------------------------------------------------------
TJONES   ANALYZE ANY        IMP_FULL_DATABASE
GRANT_PATH('TJONES', 'DBA')
TJONES   ANALYZE ANY        IMP_FULL_DATABASE
GRANT_PATH('TJONES', 'DBA', 'IMP_FULL_DATABASE')
TJONES   ANALYZE ANY        IMP_FULL_DATABASE
GRANT_PATH('TJONES', 'DBA', 'DATAPUMP_IMP_FULL_DATABASE', 'IMP_FULL_DATABASE')
...
```

4. 権限分析中にtjonesで使用されるオブジェクト権限とロールを確認します。

```
col username format a9
col used_role format a10
col object_name format a22
col object_type format a12
SELECT USERNAME, OBJ_PRIV, USED_ROLE,
 OBJECT_OWNER, OBJECT_NAME, OBJECT_TYPE
 FROM DBA_USED_OBJPRIVS
 WHERE USERNAME = 'TJONES'
 ORDER BY 1, 2, 3, 4, 5, 6;
```

次のような出力が表示されます。

```
USERNAME  OBJ_PRIV   USED_ROLE  OBJECT_OWN OBJECT_NAME            OBJECT_TYPE
--------- ---------- ---------- ---------- ---------------------- ------------
TJONES    EXECUTE    PUBLIC     SYS        DBMS_APPLICATION_INFO  PACKAGE
TJONES    SELECT     PUBLIC     SYS        DUAL                   TABLE
TJONES    SELECT     PUBLIC     SYS        DUAL                   TABLE
TJONES    SELECT     PUBLIC     SYSTEM     PRODUCT_PRIVS          VIEW
...
```

5. ユーザーtjonesの未使用の権限を確認します。

```
col username format a9
col sys_priv format a35
SELECT USERNAME, SYS_PRIV
 FROM DBA_UNUSED_SYSPRIVS
 WHERE USERNAME = 'TJONES'
 ORDER BY 1, 2;
USERNAME SYS_PRIV
-------- -----------------------------
TJONES   ADMINISTER ANY SQL TUNING SET
TJONES   ADMINISTER DATABASE TRIGGER
TJONES   ADMINISTER RESOURCE MANAGER
TJONES   ADMINISTER SQL TUNING SET
TJONES   ALTER ANY ASSEMBLY
TJONES   ON COMMIT REFRESH
...
```

親トピック: チュートリアル: DBAロールを持つユーザーによる権限の使用の分析

5.5.6 ステップ6: この例で使用したコンポーネントの削除

コンポーネントが不要になった場合、このチュートリアルで作成したコンポーネントを削除できます。

1. ユーザーpa_adminとして、dba_tuning_priv_analysis_pol権限分析ポリシーを削除します。

```
EXEC DBMS_PRIVILEGE_CAPTURE.DROP_CAPTURE ('dba_tuning_priv_analysis_pol');
```

   pa_adminユーザーのスキーマに作成されたオブジェクトはSYSスキーマにあるため、次のステップで、このユーザー(このオブジェクトを含む)を削除する場合でも、dba_tuning_priv_analysis_pol権限分析ポリシーを手動で削除する必要があります。

2. ユーザー・アカウントを作成したユーザーとして接続します。

   たとえば:

```
CONNECT sec_admin -- Or, CONNECT sec_admin@hrpdb
Enter password: password
```

3. ユーザーpa_adminおよびtjonesを削除します。

```
DROP USER pa_admin CASCADE;
DROP USER tjones;
```

親トピック: チュートリアル: DBAロールを持つユーザーによる権限の使用の分析

5.6 権限分析ポリシーおよびレポート・データ・ディクショナリ・ビュー

Oracle Databaseには、分析された権限に関する情報を示すデータ・ディクショナリ・ビューが用意されています。

表5-1に、これらのデータ・ディクショナリ・ビューを示します。

表5-1 権限分析情報を表示するデータ・ディクショナリ・ビュー

ビュー	説明
DBA_PRIV_CAPTURES	既存の権限分析ポリシーの情報をリストします
DBA_USED_PRIVS	レポートされた権限分析ポリシーに使用された権限および取得実行をリストします
DBA_UNUSED_GRANTS	使用されていない権限付与をリストします
DBA_UNUSED_PRIVS	レポートされた権限分析ポリシーに使用されていない権限および取得実行をリストします
DBA_USED_OBJPRIVS	レポートされた権限分析ポリシーに使用されたオブジェクト権限および取得実行をリストします。オブジェクト付与パスを含みません。
DBA_UNUSED_OBJPRIVS	レポートされた権限分析ポリシーに使用されていないオブジェクト権限および取得実行をリストします。オブジェクト権限付与パスを含みません。
DBA_USED_OBJPRIVS_PATH	レポートされた権限分析ポリシーに使用されたオブジェクト権限およ

ビュー	説明
	び取得実行をリストします。オブジェクト権限付与パスを含みます。
DBA_UNUSED_OBJPRIVS_PATH	レポートされた権限分析ポリシーに使用されていないオブジェクト権限および取得実行をリストします。オブジェクト権限付与パスを含みます。
DBA_USED_SYSPRIVS	レポートされた権限分析ポリシーに使用されたシステム権限および取得実行をリストします。システム権限付与パスを含みません。
DBA_UNUSED_SYSPRIVS	レポートされた権限分析ポリシーに使用されていないシステム権限および取得実行をリストします。システム権限付与パスを含みません。
DBA_USED_SYSPRIVS_PATH	レポートされた権限分析ポリシーに使用されたシステム権限および取得実行をリストします。システム権限付与パスを含みます。
DBA_UNUSED_SYSPRIVS_PATH	レポートされた権限分析ポリシーに使用されていないシステム権限および取得実行をリストします。システム権限付与パスを含みます
DBA_USED_PUBPRIVS	レポートされた権限分析ポリシーに使用された PUBLIC ロールの権限および取得実行をすべてリストします
DBA_USED_USERPRIVS	レポートされた権限分析ポリシーに使用されたユーザー権限および取得実行をリストします。ユーザー権限付与パスを含みません。
DBA_UNUSED_USERPRIVS	レポートされた権限分析ポリシーに使用されていないユーザー権限および取得実行をリストします。ユーザー権限付与パスを含みません。
DBA_USED_USERPRIVS_PATH	レポートされた権限分析ポリシーに使用されたユーザー権限および取得実行をリストします。ユーザー権限付与パスを含みます。
DBA_UNUSED_USERPRIVS_PATH	レポートされた権限分析ポリシーに使用されていない権限および取得実行をリストします。ユーザー権限付与パスを含みます。

関連トピック

- Oracle Databaseリファレンス

親トピック: 権限分析の実行による権限使用の特定

6 Microsoft Active Directoryによる集中管理ユーザーの構成

Oracle Databaseでは、中間ディレクトリまたはOracle Enterprise User Securityを使用せずにデータベースでMicrosoft Active Directoryユーザーを直接認証および認可できます。

- Microsoft Active Directoryによる集中管理ユーザーの概要
  集中管理ユーザー(CMU)によりMicrosoft Active Directoryとのシンプルな統合が実現し、ユーザーの集中化された認証と認可が可能になります。
- Oracle DatabaseとMicrosoft Active Directoryの統合の構成
  Microsoft Active Directoryを使用してユーザーを認証および認可するには、OracleデータベースからActive Directoryへの接続を構成する必要があります。
- 集中管理ユーザーの認証の構成
  パスワード認証、Kerberos認証、または公開キー・インフラストラクチャ(PKI)認証を構成できます。
- 集中管理ユーザーの認可の構成
  集中管理ユーザーにより、Active Directoryユーザーがデータベースにアクセスするための認可を管理できます。
- 一元管理ユーザーのトラブルシューティング
  Oracleには、Microsoft Active Directoryユーザーがデータベースにログインしようとしたときに発生する可能性のある一般的なエラーのトラブルシューティングに役立つエラー・メッセージが表示されます。
- Microsoft Active Directoryのアカウント・ポリシーとOracle Databaseの統合
  Oracle DatabaseとMicrosoft Active Directoryの統合の一部として、Active Directoryユーザーがデータベースにログインするときに、Oracle DatabaseによりActive Directoryのアカウント・ポリシーが適用されます。
- Oracle Autonomous Databaseを使用した集中管理ユーザーの構成
  集中管理ユーザー(CMU)をOracle Autonomous Databaseにデプロイできます。
- 一元管理ユーザーのトラブルシューティング
  Oracleには、Microsoft Active Directoryユーザーがデータベースにログインしようとしたときに発生する可能性のある一般的なエラーのトラブルシューティングに役立つエラー・メッセージが表示されます。

親トピック: ユーザー認証および認可の管理

6.1 Microsoft Active Directoryによる集中管理ユーザーの概要

集中管理ユーザー(CMU)によりMicrosoft Active Directoryとのシンプルな統合が実現し、ユーザーの集中化された認証と認可が可能になります。

- Oracle DatabaseとMicrosoft Active Directoryの統合について
  集中管理ユーザーによりMicrosoft Active Directoryとのシンプルな統合が実現し、ユーザーの集中化された認証と認可が可能になります。
- Microsoft Active Directoryによる集中管理ユーザーのしくみ
  この統合は、Microsoft Active Directoryのユーザーおよびグループがデータベース・ユーザーおよびロールに直接マッピングされることで機能します。
- 集中管理ユーザーとMicrosoft Active Directoryによるアーキテクチャ
  Active DirectoryアーキテクチャのCMUでは、Oracle Databaseユーザーおよびロールを Active Directoryで管

理できます。

- サポートされている認証方式
  Oracle DatabaseとMicrosoft Active Directoryの統合では、3つの一般的な認証方式がサポートされています。
- Microsoft Active Directoryによる集中管理ユーザーでサポートされるユーザー
  Active DirectoryのCMUにより、排他的にマップされるユーザー、共有スキーマにマップされるユーザー、管理ユーザーがサポートされます。
- 集中管理ユーザーに対するOracleマルチテナント・オプションの影響
  プラガブル・データベース(PDB)のマルチテナント・データベース・ユーザーは中央のMicrosoft Active Directoryに接続でき、また必要に応じて、個々のPDB内のユーザーは異なるMicrosoft Active Directoryに接続できます。
- データベース・リンクによる集中管理ユーザー
  CMUでは、固定されたユーザー・データベース・リンクと接続ユーザー・データベース・リンクの両方がサポートされますが、現在のユーザーのデータベース・リンクはサポートされません。

親トピック: Microsoft Active Directoryによる集中管理ユーザーの構成

6.1.1 Oracle DatabaseとMicrosoft Active Directoryの統合について

集中管理ユーザーによりMicrosoft Active Directoryとのシンプルな統合が実現し、ユーザーの集中化された認証と認可が可能になります。

Active Directoryサーバー・オペレーティング・システムの最小バージョン要件は、Microsoft Windows Server 2012です。この最小サポート対象バージョンは、Microsoftで以前のリリースがサポートされなくなると更新されます。

この統合により、組織はActive Directoryを使用して、1つのディレクトリとその他の情報技術サービスを使用して、複数のOracleデータベースでユーザーとロールを一元的に管理できます。Active Directoryユーザーは、Active Directoryに格納されている資格証明を使用してOracle Databaseに対する認証を実行できます。Active Directoryユーザーは、Active Directoryグループを使用して、データベース・ユーザー(スキーマ)およびロールに関連付けることもできます。Microsoft Active Directoryのユーザーは、排他または共有Oracle Databaseユーザー(スキーマ)にマップし、ディレクトリのグループ・メンバーシップを介してデータベース・ロールに関連付けることができます。ユーザーのログイン時に、Oracle Databaseは Active Directoryのアカウント・ポリシー(パスワードの有効期限やロックアウトされるまでの指定されたログイン失敗回数など)を受け入れます。

Oracle Database 18cリリース1 (18.1)より前では、データベース・ユーザーの認証および認可をActive Directoryと統合できるように、Oracle Enterprise User Securityを構成し、Oracle Internet Directory(またはOracle Universal Directory)をインストールして構成していました。このアーキテクチャは現在でも使用可能であり、今後はOracleエンタープライズ・ドメインおよび信頼できるデータベース間の現在のユーザー・データベース・リンク、複雑なエンタープライズ・ロールを使用し、データベース・アクセス権限とロールの監査を1箇所で行う必要があるユーザーによって使用が継続されます。

ほとんどの組織には、これらの複雑な要件はありません。かわりに、Active Directoryで集中管理ユーザー(CMU)を使用できます。この統合は、一元的なアイデンティティ管理ソリューションとしてActive Directoryを使用する組織を対象として設計されています。 Oracle Netネーミング・サービスは、従来と同様にディレクトリ・サービスと連携します。

組織は、Active DirectoryのCMUでKerberos、PKIまたはパスワード認証を使用できます。Active DirectoryのCMUの使用には、現在サポートされているOracle Databaseクライアントとの下位互換性があります。これは、パスワード認証にLDAPバインド操作が使用されていないため、パスワード・ベリファイアを格納するには、Active Directoryスキーマを拡張し、Active DirectoryにOracleフィルタを追加する必要があることを意味します。KerberosまたはPKIを使用している組織では、フィルタの追加もActive Directoryスキーマの拡張も行う必要はありません。

Oracle DatabaseとActive Directoryの統合は、次のタイプのユーザーにとって特に役に立ちます。

- 現在Kerberosや公開キー・インフラストラクチャ(PKI)などの厳密認証を使用しているユーザー。これらのユーザーはすでに一元的なアイデンティティ管理システムを使用しています。
- Oracle Enterprise User Security、Oracle Internet Directory、Oracle Unified Directory、Oracle Virtual Directoryを現在使用しており、Active Directoryとの統合が必要なユーザー。

親トピック: Microsoft Active Directoryによる集中管理ユーザーの概要

6.1.2 Microsoft Active Directoryによる集中管理ユーザーのしくみ

この統合は、Microsoft Active Directoryのユーザーおよびグループがデータベース・ユーザーおよびロールに直接マッピングされることで機能します。

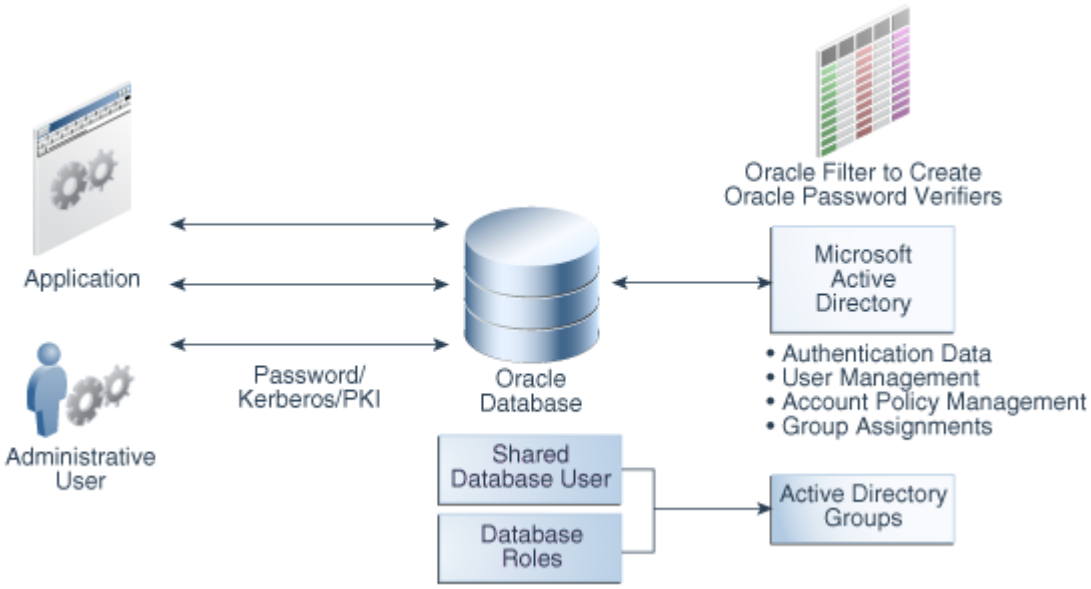
Active Directory統合とOracle Database CMUが機能するためには、Oracleデータベースが、Active Directoryでこのデータベースのために作成されたサービス・アカウントにログインできる必要があります。ユーザーがデータベースにログインするときに、データベースはこのサービス・アカウントを使用してActive Directoryに対しユーザーとグループの情報を問い合せます。このActive Directoryサービス・アカウントには、ユーザーおよびグループ情報の問合せに必要な権限がすべて付与されている必要があり、またActive Directoryでパスワード・ポリシーに関連する更新(ログイン失敗回数、ログイン失敗回数のクリアなど)を書き込むことができる必要があります。ユーザーは、パスワード、Kerberos、またはPKIを使用して認証し、排他スキーマまたは共有スキーマのいずれかに割り当てることができます。共有スキーマへのActive Directoryユーザーのマッピングは、共有スキーマにマップされているActive Directoryグループへのユーザーの関連付けによって決定します。Active Directoryグループは、データベース・グローバル・ロールにもマップできます。Active Directoryセキュリティ管理者は、共有データベース・グローバル・ユーザー(スキーマ)またはデータベース・グローバル・ロール(あるいはその両方)にマップされたグループにユーザーを割り当てることができるため、データベースでActive Directoryユーザーに割り当てられている権限およびロールを更新できます。

親トピック: Microsoft Active Directoryによる集中管理ユーザーの概要

6.1.3 集中管理ユーザーとMicrosoft Active Directoryによるアーキテクチャ

Active DirectoryアーキテクチャのCMUでは、Oracle Databaseユーザーおよびロールを Active Directoryで管理できます。

次の図は、Oracle Database CMU機能を示しています。この図では、ユーザーがアプリケーションから管理者以外のユーザーまたは管理ユーザーとして、パスワード、Kerberosまたは公開キー・インフラストラクチャ(PKI)認証を使用してOracleデータベースに接続します。Active Directoryへのデータベース接続により、これらのユーザーおよびロールをActive Directoryのユーザーおよびグループにマップできます。パスワード認証を使用する予定がある場合は、Active DirectoryにOracleフィルタをインストールする必要があります。Oracle提供のユーティリティを使用して、必要に応じて個々のユーザーに対してOracleパスワード・ベリファイアを生成するOracleフィルタをインストールできます。また、このユーティリティを使用して、Oracleパスワード・ベリファイアを保持するためにActive Directoryスキーマを拡張できます。Oracle Database集中管理ユーザーにより、Active Directory管理者は、Oracle Databaseユーザーおよびロールにマップされ ているActive Directoryユーザーおよびグループの認証、ユーザー管理、アカウント・ポリシー、およびグループの割当てを制御できます。



親トピック: Microsoft Active Directoryによる集中管理ユーザーの概要

6.1.4 サポートされている認証方式

Oracle DatabaseとMicrosoft Active Directoryの統合では、3種類の一般的な認証方式がサポートされています。

これらの認証方法は次のとおりです。

- パスワード認証
- Kerberos認証
- 公開キー・インフラストラクチャ(PKI)認証(証明書ベースの認証)

関連トピック

- 集中管理ユーザーの認証の構成

親トピック: Microsoft Active Directoryによる集中管理ユーザーの概要

6.1.5 Microsoft Active Directoryによる集中管理ユーザーでサポートされているユーザー

Active DirectoryのCMUにより、排他的にマップされるユーザー、共有スキーマにマップされるユーザー、管理ユーザーがサポートされます。

これらのユーザーは次のとおりです。

- 共有スキーマを使用してOracleデータベースにアクセスするディレクトリ・ユーザー。

  このタイプのディレクトリ・ユーザーは、共有スキーマ(データベース・ユーザー)にマップされるディレクトリ・グループの一部となることで、データベース内の共有スキーマに接続できます。共有スキーマを使用すると、Active Directoryで集中的なデータベース・ユーザー管理が可能になります。排他スキーマ(次で説明)を使用するよりも推奨されるベスト・プラクティスです。スキーマに関連付けられているユーザーが1人(たとえば、データベース・バックアップを担当する管理者)のみであっても、関連するすべてのデータベースで変更するのではなく、Active Directoryでのみ変更すればいいので、別のバックアップ管理者の追加や既存の管理者の削除も、容易に管理することができます。

  ユーザーには、Active Directory内のグループにマップされているグローバル・ロールを使用して、各自のタスクに適した

追加権限が付与されます。この設計では、ユーザーは組織内で各自のタスクを変更し、Active Directory内の新しいグループを介して新しいデータベース権限を持つことができます。

Active Directoryユーザーが、同じデータベース上の異なる共有スキーマにマップされるActive Directoryで、誤って(または故意に)複数グループのメンバーになっている可能性があります。そのユーザーに、データベース・スキーマへの排他的マッピングもある可能性があります。ログイン時に、ユーザーに複数のスキーマ・マッピングがありえる場合は、次の優先順位ルールが適用されます。

- ユーザーに排他的マッピングが存在する場合、そのマッピングは他の共有マッピングより優先されます。
- 1人のユーザーに複数の共有スキーマ・マッピングが存在する場合、スキーマID (USER_ID)が最も低い共有ユーザー・マッピングが優先されます。

予期しないスキーマ・マッピングが発生しないように、ユーザーごとに可能なマッピングは1つにすることをお薦めします。

- 排他的にマップされたグローバル・ユーザー(2層および3層アプリケーションの標準Oracle Databaseユーザー)、またはデータベース内で権限が直接付与されているユーザー。

  グローバル・ロールを使用してこれらのユーザーに権限を付与することをお薦めします。このタイプの権限付与では、ユーザーの権限およびロールを集中管理することで認可管理が容易になり、ユーザーの権限およびロールを更新するために各データベースにログインする必要がありません。

- 管理者権限SYSDBA、SYSOPER、SYSBACKUP、SYSDG、SYSKMおよびSYSRACを持つ管理グローバル・ユーザー。

  グローバル・ロールを介してこれらの管理権限を付与することはできません。これらの管理権限をActive Directoryユーザーに付与するには、データベース・ユーザー・アカウントにシステム管理権限がすでに付与されているデータベース・ユーザーに、(排他的または共有スキーマを使用して)ディレクトリ・ユーザーをマップする必要があります。

関連トピック

- 集中管理ユーザーの認可の構成

親トピック: Microsoft Active Directoryによる集中管理ユーザーの概要

6.1.6 集中管理ユーザーに対するOracleマルチテナント・オプションの影響

プラガブル・データベース(PDB)のマルチテナント・データベース・ユーザーは中央のMicrosoft Active Directoryに接続でき、また必要に応じて、個々のPDB内のユーザーは異なるMicrosoft Active Directoryに接続できます。

マルチテナント・データベースのすべてのPDBおよびルート・コンテナでは、共有構成を持つことができます。これにより、CDB全体が単一のActive Directoryサーバーに対してユーザーを認証および認可し、1つのWindowsドメインで複数のActive Directoryサーバー、または信頼できるWindowsドメイン内の複数のActive Directoryサーバーを、共有構成に基づいて認証および認可できます。あるいは、個々のPDBでは、個々の構成に基づいて、同じWindowsドメイン内または異なる(信頼できる、または信頼できない) Windowsドメイン内の異なるActive Directoryサーバーに対してユーザーを認証および認可できます。

親トピック: Microsoft Active Directoryによる集中管理ユーザーの概要

6.1.7 データベース・リンクによる集中管理ユーザー

CMUでは、固定されたユーザー・データベース・リンクと接続ユーザー・データベース・リンクの両方がサポートされますが、現在のユーザーのデータベース・リンクはサポートされません。

CMU-Active Directoryユーザーが固定されたユーザー・データベース・リンクを使用するための特別な要件はありません。パス

ワード、Kerberos、またはPKI認証を使用するCMU-Active Directoryユーザーは、通常のデータベース・ユーザーと同様に固定されたユーザー・データベース・リンクを使用できます。Kerberos認証は、データベース・リンクを使用したOracle Databaseの強力な認証でも機能します。詳細は、My Oracle SupportのNote 1370327.1を参照してください。

CMU-Active Directoryユーザーが接続ユーザー・データベース・リンクを使用するために、パスワード認証のみがサポートされています。ソース・データベースとターゲット・データベースの両方をCMU-Active Directoryで構成して、同じActive Directoryユーザーがパスワード認証を使用して両方のデータベースにログインできるようにする必要があります。

親トピック: Microsoft Active Directoryによる集中管理ユーザーの概要

6.2 Oracle DatabaseとMicrosoft Active Directoryの統合の構成

Microsoft Active Directoryを使用してユーザーを認証および認可するには、OracleデータベースからActive Directoryへの接続を構成する必要があります。

- Oracle DatabaseとMicrosoft Active Directoryの接続の構成について
  この接続を構成する前に、Microsoft Active Directoryのインストールと構成が完了している必要があります。
- Microsoft Active Directoryへの接続
  Oracleデータベースの作成中にMicrosoft Active Directory接続を構成するか、または既存のOracleデータベースとのMicrosoft Active Directoryの接続を構成できます。

親トピック: Microsoft Active Directoryによる集中管理ユーザーの構成

6.2.1 Oracle DatabaseとMicrosoft Active Directory接続の構成について

この接続を構成する前に、Microsoft Active Directoryのインストールと構成が完了している必要があります。

Active DirectoryでOracleサービス・ディレクトリ・ユーザーを作成し、Active DirectoryへのOracle Database接続を構成した後、認証のタイプに基づいて、パスワード、Kerberos、または公開キー・インフラストラクチャ(PKI)認証に対応してデータベースとActive Directoryを構成する必要があります。データベース・ユーザーとグローバル・ロールをActive Directoryユーザーおよびグループにマッピングする前に、Active Directoryユーザーおよびグループが作成されていることを確認する必要があります。CREATE USER、CREATE ROLE、ALTER USER、ALTER ROLEのSQL文をGLOBALLY句とともに使用して、データベース・ユーザーとグローバル・ロールをActive Directoryユーザーおよびグループにマップします。Active Directoryシステム管理者は、要件に対応するためにActive Directoryユーザーからなる新しいActive Directoryグループを設定する必要があります。

Active Directoryシステム管理者は、SASLバインドの有無にかかわらずActive Directory接続を設定します。Oracle Databaseは、自動的にActive Directory接続をまずSASLバインドで試行し、失敗した場合はSASLバインドなしで試行しますが、引き続きTLSで保護されます。これは、Microsoft Active Directory管理者がActive DirectoryでSASL設定をどのように構成しているかにかかわらず、SASLバインドが成功しなかった場合でもOracleデータベースが接続することを意味します。

親トピック: Oracle DatabaseとMicrosoft Active Directoryの統合の構成

6.2.2 Microsoft Active Directoryへの接続

Oracleデータベースの作成中にMicrosoft Active Directory接続を構成するか、または既存のOracleデータベースとのMicrosoft Active Directoryの接続を構成できます。

- ステップ1: Microsoft Active DirectoryでのOracleサービス・ディレクトリ・ユーザー・アカウントの作成および権限の

付与
Oracleサービス・ディレクトリ・ユーザー・アカウントは、Oracle DatabaseとLDAPディレクトリ・サービスとの相互作用に使用されます。

- ステップ2: パスワード認証のためにパスワード・フィルタをインストールしてMicrosoft Active Directoryスキーマを拡張
  Active DirectoryサーバーでOracle `opwdintg.exe`実行可能ファイルを使用すると、パスワード・フィルタをインストールし、Active Directoryスキーマを拡張できます。
- ステップ3: Oracle Databaseソフトウェアのインストール(必要な場合)
  この操作をまだ実行していない場合は、Oracle Universal Installer(OUI)を使用してOracleソフトウェアをインストールします。
- ステップ4: dsi.oraまたはldap.oraファイルの作成
  `dsi.ora`ファイルと`ldap.ora`ファイルには、Active Directory用の集中管理ユーザーの接続を指定します。
- ステップ5: セキュアな接続のためのActive Directory証明書のリクエスト
  `dsi.ora`または`ldap.ora`ファイルの構成が完了したら、セキュアな接続のためのMicrosoft Active DirectoryとOracle Databaseの証明書を準備します。
- ステップ6: セキュアな接続のためのウォレットの作成
  Active Directory証明書をコピーすると、それをOracleウォレットに追加できるようになります。
- ステップ7: Microsoft Active Directory接続の構成
  次に、これまでの設定を使用してデータベースをActive Directoryに接続します。
- ステップ8: Oracleウォレットの確認
  `orapki`ユーティリティでは、このデータベースのウォレットが正常に作成されたことを確認できます。
- ステップ9: 統合のテスト
  統合をテストするには、`ORACLE_HOME`、`ORACLE_BASE`および`ORACLE_SID`環境変数を設定し、LDAPのパラメータ設定を確認する必要があります。

親トピック: Oracle DatabaseとMicrosoft Active Directoryの統合の構成

6.2.2.1 ステップ1: Microsoft Active DirectoryでのOracleサービス・ディレクトリ・ユーザー・アカウントの作成および権限の付与

Oracleサービス・ディレクトリ・ユーザー・アカウントは、Oracle DatabaseとLDAPディレクトリ・サービス間の相互作用に使用されます。

Oracleサービス・ディレクトリ・ユーザー・アカウントは、Oracle DatabaseとLDAPディレクトリ・サービス間の相互作用の他に、Kerberosにも使用できます。
このアカウントは、Active Directoryドメイン・コントローラにバインドしてActive Directoryのユーザー情報およびグループ情報を問い合せる場合や、ログインの成功または失敗を更新する場合や、Kerberosが構成されているときにKerberos認証を更新する場合に、Oracle Databaseで使用されるActive Directoryユーザー・アカウントです。このアカウントに必要な最小権限は、(データベースにログインするActive Directoryユーザーの)`Read properties`権限であり、Active Directoryユーザーによってデータベース・パスワード認証が使用される場合は、`Write lockoutTime` (Active Directoryユーザーのプロパティ)権限、および(Active Directoryユーザーの`orclCommonAttribute`プロパティの)`Control Access`権限です。

1. ユーザー・アカウントの作成およびユーザー・アカウントへの権限の付与のための管理権限がある管理者として、Microsoft Active DirectoryのWindowsドメインコントローラにログインします。
2. Oracleサービス・ディレクトリ・ユーザー・アカウントをActive Directoryユーザーとして作成します。

ディレクトリにサービス・ユーザー・アカウントを作成します。Active Directoryユーザーが使用するWindowsドメインに応じて、サービス・ユーザー・アカウントを作成する場所を選択できます。次のガイドラインに従ってください。

- すべてのActive Directoryユーザーが1つのドメインに配置されている場合、そのドメインでこのアカウントを作成します。このようにすると、パフォーマンスが向上します。
- Active Directoryユーザーが複数のWindowsドメインに存在する場合は、他のすべてのドメインに信頼されているドメインに、このサービス・ユーザー・アカウントを作成します。
  - 選択したドメインは、他のすべてのドメインに信頼されている必要があります。
  - サービス・ユーザーは、これらすべてのWindowsドメインにバインドできる必要があり、これらすべてのWindowsドメイン内のActive Directoryユーザーのプロパティに、付与されている権限でアクセスできる必要があります。
  - 他のすべてのドメインは、信頼できるドメインからのサービス・ユーザーのアクセスを可能にするために、TLS/SSL経由の単純なバインドをサポートしている必要があります。
  - 他のすべてのドメインの管理者は、信頼できるドメインからサービス・ユーザー・アカウントに、必要な最小権限を付与する必要があります。

3. Active Directory内のOracleサービス・ディレクトリ・ユーザー・アカウントに、Oracleデータベースにアクセスする必要があるActive Directoryユーザーのプロパティに対する次の権限を付与します。
   - `Read properties` (Oracleデータベースにログインするactive Directoryユーザーに関して)
   - `Write lockoutTime` (パスワード認証を使用してOracleデータベースにログインするActive Directoryユーザーのプロパティ)
   - `Control Access` (パスワード認証を使用してOracleデータベースにログインするActive Directoryユーザーの`orclCommonAttribute`プロパティに関して)

親トピック: Microsoft Active Directoryへの接続

6.2.2.2 ステップ2: パスワード認証のためにパスワード・フィルタをインストールしてMicrosoft Active Directoryスキーマを拡張

Active DirectoryサーバーでOracle `opwdintg.exe`実行可能ファイルを使用してパスワード・フィルタをインストールし、Active Directoryスキーマを拡張できます。

認証方法がKerberosまたはSSLの場合、このステップを実行する必要はありません。`opwdintg.exe`実行可能ファイルは、Oracleパスワード・フィルタをインストールしてActive Directoryスキーマを拡張し、Active DirectoryでOracle Databaseのパスワード認証を可能にするActive Directoryグループを作成します。このプロシージャは、ユーザー・アカウントのActive Directoryスキーマに`orclCommonAttribute`プロパティを追加します。

ノート:

Oracle データベースへのログインにパスワード認証を使用する必要がある場合、ドメインのすべての Windows ドメイン・コントローラに Oracle パスワード・フィルタをインストールして、このドメインの Active Directory ユーザーに Oracle パスワード・ベリファイアが生成されるようにする必要があります。

また、`orclCommonAttribute` は Active Directory ユーザーの Oracle パスワード・ベリファイアを格納することに注意してください。この属性は、他の Oracle 製品またはエンタープライズ・ユーザー・セキュリティなどの機能によるパスワード認証にも使用されます。セキュリティを考慮して、Oracle サービス・ディレクトリ・ユーザー以外の全員の、

orclCommonAttribute プロパティへのアクセスを拒否する必要があります。

1. 最新バージョンのopwdintg.exe (Oracleパスワード統合)ユーティリティにアクセスします。
   - My Oracle Supportアカウントがある場合: My Oracle Supportで自分のアカウントにログインし、ドキュメントID 2462012.1を検索します。この場所からopwdintg.exeをダウンロードします。これが最新のバージョンです。
   - My Oracle Supportアカウントがない場合: ドキュメントID 2462012.1から最新バージョンのopwdintg.exeをダウンロードできるように、My Oracle Supportアカウントに登録します。
2. 安全なコピー方法(sftpなど)を使用して、opwdintg.exeを各Windowsドメイン・コントローラの一時ディレクトリ(C:¥tempなど)にコピーします。
3. Active Directory管理者として各Windowsドメイン・コントローラに接続します。
   現在、opwdintg.exeユーティリティには英語版のWindows OSが必要です。
4. Windows OSの言語設定が英語であることを確認してください。
5. 各Windowsドメイン・コントローラでopwdintg.exeユーティリティを実行します。
   新しいopwdintg.exeを使用して更新されたパスワード・フィルタを再インストールする場合は、ドメイン・コントローラを再起動する必要があります。
   opwdintg.exeユーティリティを実行するには、次のいずれかの方法を使用します。
   - Windowsエクスプローラを開き、opwdintg.exeユーティリティをダブルクリックします。
   - Windowsのコマンド・プロンプトを開き、次のステップを実行します。
     a. opwdintg.exeユーティリティがあるディレクトリに移動します。たとえば:
     ```
     cd c:¥temp
     ```
     b. 次のコマンドを入力して、コマンドラインからユーティリティを実行します。
     ```
     .¥opwdintg.exe
     ```
6. 次のプロンプトに対して入力します:
   - ADスキーマを拡張しますか。[はい/いいえ]:Yesと入力します。

     Active Directoryスキーマを拡張するには、Windows OSの言語設定を英語にする必要があります。
   - このドメインのスキーマ拡張は永続的です。続行しますか。[はい/いいえ]: Yesと入力します。

     次のことに注意してください。

     a. Active Directoryスキーマを拡張できるのは1回のみです。再度スキーマを拡張しようとすると、エラー・メッセージが表示されますが、このエラーは無視できます。

     b. このステップでは、次の3つのベリファイア・グループを作成します。これらのグループがすでに存在する場合は、エラーが表示されますが、これらのエラーは無視できます。これらのベリファイア・グループは、インストールされているADユーザー・フォルダから移動することも、ユーザー・オブジェクトのこのフォルダ構造の外に移動することもできます。

        - ORA_VFR_MD5は、Oracle Database WebDAVクライアントを使用する場合に必要です。
        - ORA_VFR_11Gにより、Oracle Database 11Gパスワード・ベリファイア機能の使用が有効になります。
        - ORA_VFR_12Cにより、Oracle Database 12Cパスワード・ベリファイアの使用が有効に

なります。

c. Active Directoryスキーマをバックアップしていない場合、いったん拡張するとActive Directoryスキーマ拡張を元に戻すことはできません。

次の2つのプロンプトは、パスワード・フィルタがインストール済かどうかによって異なります。

- Found password filter installed already.削除しますか。[はい/いいえ]: パスワード・フィルタがすでにインストールされている場合、このプロンプトが表示されます。ほとんどの場合、フィルタを削除しないようにNoと入力します。

  Yesと入力してパスワード・フィルタを削除する場合は、opwdintg.exeを再実行して、これらのプロンプトの完了後にパスワード・フィルタを再インストールする必要があります。そうせずにコンピュータを再起動すると、Active Directoryユーザーがパスワードを変更したときにパスワード・ベリファイアが生成されません。

- Oracleパスワード・フィルタをインストールしますか。[はい/いいえ]: パスワード・フィルタがまだインストールされていない場合、このプロンプトが表示されます。「Yes」を入力します。
- この変更にはマシンの再起動が必要です。今すぐ再起動しますか。[はい/いいえ]: Yesと入力します。

親トピック: Microsoft Active Directoryへの接続

6.2.2.3 ステップ3: Oracle Databaseソフトウェアのインストール(必要な場合)

この操作をまだ実行していない場合は、Oracle Universal Installer (OUI)を使用してOracleソフトウェアをインストールします。

データベース全体ではなく、Oracle Databaseソフトウェアをインストールすることのみが必要です。Oracleデータベース・ソフトウェアをインストールすると、Database Configuration Assistant (DBCA)を使用して、データベースの作成時にActive Directoryの集中管理ユーザーを構成できます。データベースの作成後にDBCAを使用して、または手動で、Active Directoryの集中管理ユーザーを構成することもできます。

- 使用しているプラットフォームのOracle Databaseインストレーション・ガイドの手順に従って、Oracleソフトウェアをインストールします。

Oracleデータベース・ソフトウェアをインストールした後、DBCAを使用して、データベースの作成時にActive Directoryの集中管理ユーザーを構成できます。データベースの作成後にDBCAを使用して、または手動で、Active Directoryの集中管理ユーザーを構成することもできます。

親トピック: Microsoft Active Directoryへの接続

6.2.2.4 ステップ4: dsi.oraまたはldap.oraファイルの作成

dsi.oraファイルとldap.oraファイルには、Active Directory用の集中管理ユーザーの接続を指定します。

- dsi.oraファイルとldap.oraファイルの比較
  dsi.oraとldap.oraの使用方法は、他のサービスでldap.oraをどのように使用するかによって異なります。
- dsi.oraファイルの使用について
  dsi.oraファイルを使用して、集中管理ユーザーのActive Directoryサーバーを指定します。
- dsi.oraファイルの作成
  dsi.ora構成ファイルには、集中管理ユーザーのActive Directoryサーバーを検出するための情報を設定します。
- ldap.oraファイルの使用について
  ldap.oraファイルを使用して、集中管理ユーザー用のActive Directoryサーバーを指定できます。
- ldap.oraファイルの作成

このステップでは、ldap.oraがネット・ネーミング・サービスに使用されておらず、集中管理ユーザーのActive Directoryとの接続を設定するために使用できると想定しています。

親トピック: Microsoft Active Directoryへの接続

6.2.2.4.1 dsi.oraファイルとldap.oraファイルの比較

dsi.oraとldap.oraの使用方法は、他のサービスでldap.oraをどのように使用するかによって異なります。

dsi.oraファイルには、Active Directory用の集中管理ユーザーの接続を指定します。ldap.oraファイルには、Active Directoryサーバーへの接続も指定できます。ただし、各PDBに独自のldap.oraを設定することはできず、また、ldap.oraはネット・ネーミング・サービスなど他のサービスですでに使用されている(または将来使用される)可能性があるため、集中管理ユーザーに対してはdsi.oraを使用することをお薦めします。

親トピック: ステップ4: dsi.oraまたはldap.oraファイルの作成

6.2.2.4.2 dsi.oraファイルの使用について

dsi.oraファイルを使用して、集中管理ユーザーのActive Directoryサーバーを指定します。

Active Directoryサーバーを識別するには、dsi.oraファイルを手動で作成する必要があります。dsi.oraファイルには、ldap.oraファイルを配置できるのと同じ場所にある場合に、すべてのプラガブル・データベースのActive Directory接続情報を指定します。PDB固有のウォレット・ロケーションにあるdsi.oraファイルは、そのPDBのみのメインのdsi.oraファイルより優先されます。

ノート:

ネーミング・サービスに ldap.ora を使用している場合は、Active Directory 構成の CMU に対し ldap.ora を変更しないでください。CMU-Active Directory の構成には、dsi.ora のみを使用してください。

dsi.oraの配置

$ORACLE_HOMEの下でなく、$ORACLE_BASEの下の書き込み可能なファイルのディレクトリを使用することをお薦めします。Oracle Database18c以降、オプションで$ORACLE_HOMEディレクトリを読取り専用に設定できます。したがって、dsi.oraファイルを$ORACLE_HOME外のディレクトリに置いて、将来のリリースのdsi.ora構成に対応する必要があります。

dsi.oraの検索順序

dsi.oraファイルを作成すると、Oracle Databaseでは次の順序で検索されます。

1. WALLET_LOCATION設定がsqlnet.oraファイルに含まれている場合、非マルチテナント・データベースまたはマルチテナント・データベースのルート・コンテナについて、Oracleは、sqlnet.oraで指定された場所を検索します。マルチテナント・データベースのPDBの場合、Oracleは、WALLET_LOCATION_specified_in_sqlnet.ora/pdb_guidディレクトリにあるPDBごとのウォレットの場所で検索します。
2. WALLET_LOCATION設定がsqlnet.oraファイルに含まれていない場合、Oracle Databaseはデフォルトのウォレット・ロケーションを検索します。
3. Oracle Databaseがウォレット・ロケーションでdsi.oraを検出できない場合、Oracle Databaseは次の順序で検索します。これらは、Oracle Databaseがldap.oraファイルを検索する場所と同じです。
    a. $LDAP_ADMIN環境変数設定
    b. $ORACLE_HOME/ldap/adminディレクトリ

c. $TNS_ADMIN環境変数設定

d. $ORACLE_HOME/network/adminディレクトリ

dsi.oraを使用する場合

集中管理ユーザーのActive Directoryサーバーの識別には、dsi.oraのみを使用することをお薦めします。dsi.oraとldap.oraの両方がActive Directoryの集中管理ユーザー用に同じデータベース内に構成されていて、両方が同じディレクトリにある場合、dsi.oraはldap.oraファイルより優先されます。異なるディレクトリに配置されている場合、前述した場所の優先順位リストで最初に検出されたディレクトリを使用して、Active Directoryサーバーが検索されます。最初に検出されたdsi.oraまたはldap.oraのディレクトリ・サーバー・タイプがActive Directoryでない場合、集中管理ユーザーは有効になりません。

マルチテナント環境でのdsi.oraの使用

マルチテナント・データベースのPDBごとにdsi.oraファイルを指定できます。PDB固有のdsi.oraは、その1つのPDBの共有dsi.oraまたはldap.oraの共通設定をオーバーライドします。異なるPDBが、CMU用の異なるActive Directoryサーバーに接続できます。個々のPDBのdsi.oraファイルは、そのPDBのウォレットと同じディレクトリにあります。

sqlnet.oraファイルのWALLET_LOCATIONパラメータが設定されている場合、個々のPDBのdsi.oraファイルは、WALLET_LOCATION_specified_in_sqlnet.ora/pdb_guid/ディレクトリのPDBごとのウォレットにあります。

sqlnet.oraファイルのWALLET_LOCATIONパラメータが設定されていない場合、マルチテナント・データベース内の個々のコンテナのデフォルトのウォレットの場所は、$ORACLE_BASE/admin/db_unique_name/pdb_guid/wallet/ディレクトリです。各PDBでデフォルトのウォレット・ロケーションを使用するには、sqlnet.oraにWALLET_LOCATIONを設定しないでください。

db_unique_nameを検索するには、CDBルートに接続して次の問合せを実行します。

```
SELECT DB_UNIQUE_NAME FROM V$DATABASE;
```

CDBルートからpdb_guidを検索するには、次の問合せを実行します。

```
SELECT PDB_NAME,GUID FROM DBA_PDBS;
```

sqlnet.oraのWALLET_LOCATIONパラメータがdsi.oraに与える影響

sqlnet.oraにWALLET_LOCATIONパラメータを設定する場合、または設定しない場合、次の効果があります。

- WALLET_LOCATIONがsqlnet.oraに設定されていない場合、$ORACLE_BASE/admin/db_unique_name/walletディレクトリにあるCDBルート・コンテナのデフォルトのウォレット・ディレクトリにdsi.oraを配置することもできます。ただし、CDBルート・コンテナはCDBデータベース全体ではなくActive Directoryにのみ接続します。
- WALLET_LOCATIONがsqlnet.oraに設定されている場合は、そのウォレットの場所にdsi.oraを配置でき、また、CDBルート・コンテナはCDBデータベース全体ではなくActive Directoryにのみ接続します。

dsi.oraの内容の変更

データベースの起動後にdsi.oraの内容を変更した場合は、データベース・インスタンスを再起動するか次のDDLを再実行して、dsi.ora内の更新された内容を有効にする必要があります。

```
ALTER SYSTEM SET LDAP_DIRECTORY_ACCESS = 'PASSWORD';
```

マルチテナント環境では、CDBルートではなく各PDBで、LDAP_DIRECTORY_ACCESSパラメータを設定する必要があります。

親トピック: ステップ4: dsi.oraまたはldap.oraファイルの作成

6.2.2.4.3 dsi.oraファイルの作成

dsi.ora構成ファイルには、集中管理ユーザーのActive Directoryサーバーを検出するための情報を設定します。

dsi.ora構成ファイルを使用するには:

1. Oracleデータベースが存在するホストにログインします。
2. dsi.oraファイルの検索順序に基づいて、dsi.oraファイルを使用するディレクトリを選択します。(関連トピックを参照してください。)このディレクトリが存在しない場合は、作成します。次に、このディレクトリに移動してdsi.oraファイルを作成します。
3. dsi.oraファイルに次のパラメータを追加します。
    - DSI_DIRECTORY_SERVERS、Active Directoryサーバーのホストおよびポート番号と代替ディレクトリ・サーバーを指定します。ディレクトリ・サーバー名は完全修飾名である必要があります。複数のWindowsドメインを使用する場合は、ここに複数のActive Directoryサーバーを設定することもできます。たとえば:

      ```
      DSI_DIRECTORY_SERVERS = (AD-server.production.examplecorp.com:389:636,
      sparky.production.examplecorp.com:389:636)
      ```

      高可用性およびフェイルオーバー構成のActive Directoryドメイン・サーバーは、CMUを使用して構成できます。次のいずれかの方法で、高可用性およびフェイルオーバーのActive Directoryドメイン・サーバーを構成できます。

        - Active Directoryドメイン・サーバーの前にロードバランサを使用する
        - 各Active Directoryドメイン・サーバーをリスト内のホスト名またはIPアドレス別に一覧表示する
        - 異なるActive Directoryドメイン・サーバーを返すドメイン名を使用する

      ロード・バランサの使用は、特にActive Directoryドメイン・サーバーに対してすでに使用している場合にお薦めの選択肢です。ロード・バランサを使用すると、dsi.oraファイルに変更を加えなくても、ロード・バランサの背後にあるActive Directoryドメイン・サーバーを管理および追加できます。Active Directoryドメイン・サーバーのリストを指定すると、より高速で低コストになりますが、Active Directoryドメイン・サーバーと結び付けるため、変更(新規または削除されたサーバー)をdsi.oraに反映する必要があります。ドメイン名を使用すると、高可用性およびフェイルオーバーが提供されますが、理想的な解決策ではありません。DNSは、毎回同じサーバーではなく、異なるサーバーを返すことが必要になります。CMUは、ドメイン名検索から返された最初のサーバーを試し、それが失敗すると認証は失敗します。ただし、ドメイン名を使用すると、dsi.oraでサーバーのリストを指定しなくても、異なるActive Directoryドメイン・サーバーを使用できるようになります。

    - DSI_DEFAULT_ADMIN_CONTEXT、Active Directoryユーザーおよびグループが配置されている検索ベースを設定します。このパラメータはオプションです。デフォルトでは、Active Directoryのユーザーおよびグループは、Active Directoryのデフォルト・ネーミング・コンテキストで検出されます。このパラメータを設定しないことをお薦めします。このパラメータは、Active Directoryユーザーおよびグループの検索範囲を制限する場合にのみ設定します。たとえば:

      ```
      DSI_DEFAULT_ADMIN_CONTEXT =
      "OU=sales,DC=production,DC=examplecorp,DC=com"
      ```

    - DSI_DIRECTORY_SERVER_TYPE、Active Directoryサーバー・アクセスを決定します。これはActive DirectoryのADに設定する必要があります。この値は大文字で入力してください。

      ```
      DSI_DIRECTORY_SERVER_TYPE = AD
      ```

関連トピック

- dsi.oraファイルの使用について

親トピック: ステップ4: dsi.oraまたはldap.oraファイルの作成

6.2.2.4.4 ldap.oraファイルの使用について

ldap.oraファイルを使用して、集中管理ユーザー用のActive Directoryサーバーを指定できます。

ネット・ネーミング・サービスなど別の目的ですでにldap.oraファイルを使用している場合は、dsi.oraファイルを使用して、ユーザー認証および認可に使用するActive Directoryに接続するように集中管理ユーザーを構成する必要があります。Active Directoryがすでにネット・ネーミング・サービスに使用されている場合でも、集中管理ユーザーのActive Directoryサーバーを識別するために、dsi.oraファイルを作成し、使用する必要があります。データベースが現在、別のサービスにldap.oraを使用していない場合でも、今後ネット・ネーミング・サービスにldap.oraを使用する場合はdsi.oraを使用することをお薦めします。

ldap.oraがネーミング・サービスに使用されている場合は、ldap.oraを変更しないでください。CMU-Active Directoryの構成には、dsi.oraのみを使用してください。

ldap.oraを使用する利点

ldap.oraを使用する利点は、DBCAグラフィカル・インタフェースまたはDBCAサイレント・モードを使用してActive Directoryサーバーへの接続の構成を完了できることです。dsi.oraを使用する場合、Active Directoryへの接続の構成を完了するステップを個別に実行する必要があります。

ldap.oraの配置

通常、ldap.oraファイルは$ORACLE_HOME/network/adminディレクトリに格納されます。WALLET_LOCATIONが$ORACLE_HOME/network/adminに設定されていないかぎり、ldap.oraファイルは通常、sqlnet.oraファイルで指定されているWALLET_LOCATIONと同じディレクトリには指定できません。

ldap.oraの検索順序

ldap.oraファイルを作成すると、Oracle Databaseでは次の順序で検索されます。

1. $LDAP_ADMIN環境変数設定
2. $ORACLE_HOME/ldap/adminディレクトリ
3. $TNS_ADMIN環境変数設定
4. $ORACLE_HOME/network/adminディレクトリ

ldap.oraの内容の変更

データベースの起動後にldap.oraの内容を変更した場合は、データベース・インスタンスを再起動するか次のDDLを再実行して、ldap.ora内の更新された内容を有効にする必要があります。

```
ALTER SYSTEM SET LDAP_DIRECTORY_ACCESS = 'PASSWORD';
```

マルチテナント環境では、CDBルートではなく各PDBで、LDAP_DIRECTORY_ACCESSパラメータを設定します。

親トピック: ステップ4: dsi.oraまたはldap.oraファイルの作成

6.2.2.4.5 ldap.oraファイルの作成

このステップでは、ldap.oraがネット・ネーミング・サービスに使用されておらず、集中管理ユーザーのActive Directoryとの接続を設定するために使用できると想定しています。

1. Oracleデータベースが存在するホストにログインします。
2. ldap.oraファイルの検索順序に基づいて、ldap.oraファイルを使用するディレクトリを選択します。(関連トピックを参照してください。)このディレクトリが存在しない場合は、作成します。次に、このディレクトリに移動してldap.oraファイルを作成します。
3. ldap.oraファイルが存在しない場合、テキスト・エディタを使用して作成します。
   ldap.oraファイルが存在する場合は、このファイルのバックアップを作成してから、ldap.oraを開きます。
4. ldap.oraファイルに次のパラメータを追加します。
   - DIRECTORY_SERVERS、Active Directoryサーバーのホストおよびポート番号と代替ディレクトリ・サーバーを指定します。複数のWindowsドメインを使用する場合は、ここに複数のActive Directoryサーバーを設定することもできます。ディレクトリ・サーバー名は完全修飾名である必要があります。たとえば:

     ```
     DIRECTORY_SERVERS = (AD-server.production.examplecorp.com:389:636,
     sparky.production.examplecorp.com:389:636)
     ```

   - DEFAULT_ADMIN_CONTEXT、Active Directoryユーザーおよびグループが配置されている検索ベースを設定します。このパラメータはオプションです。デフォルトでは、Active Directoryのユーザーおよびグループは、Active Directoryのデフォルト・ネーミング・コンテキストで検出されます。このパラメータを設定しないことをお薦めします。このパラメータは、Active Directoryユーザーおよびグループの検索範囲を制限する場合にのみ設定します。たとえば:

     ```
     DEFAULT_ADMIN_CONTEXT = "OU=sales,DC=production,DC=examplecorp,DC=com"
     ```

   - DIRECTORY_SERVER_TYPE、LDAPサーバー・アクセスを決定します。これはActive DirectoryのADに設定する必要があります。この値は大文字で入力してください。

     ```
     DIRECTORY_SERVER_TYPE = AD
     ```

関連トピック

- ldap.oraファイルの使用について

親トピック: ステップ4: dsi.oraまたはldap.oraファイルの作成

6.2.2.5 ステップ5: セキュアな接続のためのActive Directory証明書のリクエスト

dsi.oraまたはldap.oraファイルの構成が完了したら、セキュアな接続のためのMicrosoft Active DirectoryとOracle Databaseの証明書を準備します。

- Active Directory管理者からActive Directory証明書をリクエストします。

関連トピック

- orapkiユーティリティを使用した証明書失効リスト(CRL)の管理

親トピック: Microsoft Active Directoryへの接続

6.2.2.6 ステップ6: セキュアな接続のためのウォレットの作成

Active Directory証明書をコピーすると、それをOracleウォレットに追加できるようになります。

1. 証明書テキスト・ファイル(AD_CA_Root_cert.txtなど)をActive Directoryサーバーから一時ディレクトリ(ローカル・ホストの/tmpなど)にコピーします。
   sqlnet.oraファイルでウォレット・ロケーションが指定されていない場合、データベースはこの順序で次の場所のウォレットを検索します。ディレクトリの場所を作成する必要があります。

非マルチテナント・データベースの場合、またはマルチテナント・データベースのCDBルート・コンテナの場合:

a. $ORACLE_BASE/admin/db_unique_name/wallet/

b. $ORACLE_HOME/admin/db_unique_name/wallet/

マルチテナント・データベース内のPDBの場合:

c. $ORACLE_BASE/admin/db_unique_name/pdb_guid/wallet/

d. $ORACLE_HOME/admin/db_unique_name/pdb_guid/wallet/

マルチテナント・データベースの個々のコンテナごとに、ウォレット・ファイルを$ORACLE_BASEの下のデフォルトのウォレット・ロケーション、つまり$ORACLE_BASE/admin/db_unique_name/pdb_guid/wallet/ディレクトリに配置することをお薦めします。

db_unique_nameを検索するには、CDBルートに接続して次の問合せを実行します。

```
SELECT DB_UNIQUE_NAME FROM V$DATABASE;
```

CDBルートからpdb_guidを検索するには、次の問合せを実行します。

```
SELECT PDB_NAME,GUID FROM DBA_PDBS;
```

sqlnet.oraを使用してウォレットの場所を指定する場合、指定したウォレットの場所は、非マルチテナント・データベースまたはマルチテナント・データベースのルート・コンテナに対するものです。マルチテナント・データベースの各PDBについて、そのウォレットはWALLET_LOCATION_specified_in_sqlnet.ora/pdb_guidにあります。個々のPDB dsi.oraをWALLET_LOCATION_specified_in_sqlnet.ora/pdb_guidに配置することもできます。

2. 新規ウォレットの作成

   次のコマンドは、指定したパスに自動ログイン・ウォレットを作成します。

```
orapki wallet create -wallet path_of_wallet -auto_login
Enter password: password
```

3. Active Directory (最初のステップで作成)で検索を実行するためのOracleサービス・ディレクトリ・ユーザー・アカウントのユーザー名でウォレットにエントリを作成します。

   たとえば:

```
mkstore -wrl path_of_wallet -createEntry ORACLE.SECURITY.USERNAME oracle
```

4. Oracleサービス・ディレクトリ・ユーザー・アカウントのDNを使用してウォレットにエントリを作成します。

   たとえば:

```
mkstore -wrl path_of_wallet -createEntry ORACLE.SECURITY.DN
cn=oracle,cn=users,dc=production,dc=examplecorp,dc=com
```

   この例では、DNSドメインがproduction.examplecorp.comであることをDNが示しています。Windowsドメイン名は単にproductionです。

5. Oracleサービス・ディレクトリ・ユーザー・アカウントのユーザー・パスワード資格証明を使用してウォレットにエントリを作成します。

   たとえば:

```
mkstore -wrl path_of_wallet -createEntry ORACLE.SECURITY.PASSWORD password
```

6. 証明書をウォレットに追加します。Active Directory管理者から受信したActive Directory証明書を使用します。

   たとえば:

```
orapki wallet add -wallet path_of_wallet -cert /tmp/AD_CA_Root_cert.txt -
trusted_cert
```

sqlnet.ora内でWALLET_LOCATIONが指定されている場合、Active Directory証明書をPDB固有のウォレットの場所(つまり、PDBごとのWALLET_LOCATION_specified_in_sqlnet.ora/pdb_guid)に追加する必要があります。Active Directory証明書をWALLET_LOCATION_specified_in_sqlnet.oraに追加することもできます。ただし、これはルート・コンテナのみに対して有効になり、CDB全体に対しては有効になりません。

7. 資格証明を検証します。
   たとえば:

```
orapki wallet display -wallet path_of_wallet
```

出力は次のようになります。

```
Requested Certificates:
User Certificates:
Oracle Secret Store entries:
ORACLE.SECURITY.DN
ORACLE.SECURITY.PASSWORD
ORACLE.SECURITY.USERNAME
Trusted Certificates:
Subject: CN=ADSVR,DC=production,DC=examplecorp,DC=com
```

ウォレットへの変更はすぐに有効になり、データベースの再起動は必要ありません。

親トピック: Microsoft Active Directoryへの接続

6.2.2.7 ステップ7: Microsoft Active Directory接続の構成

次に、これまでの設定を使用してデータベースをActive Directoryに接続します。

- Microsoft Active Directory接続の構成について
  Microsoft Active Directory接続を構成するには、データベースでパラメータを設定するか、DBCAを使用します。
- Databaseのシステム・パラメータを使用した手動のアクセスの構成
  LDAP固有のOracle Databaseのシステム・パラメータを使用して、Active Directoryサービス接続を手動で構成できます。
- Database Configuration Assistant GUIを使用したアクセスの構成
  Oracle Database Configuration Assistant (DBCA)は、LDAP接続構成を完了し、ウォレットを自動的に作成して、使用するActive Directory証明書を格納します。DBCAは、ldap.oraがCMU-ActiveDirectory用に構成されている場合にのみ動作します。
- Database Configuration Assistantのサイレント・モードを使用したアクセスの構成
  ldap.ora (dsi.oraではない)が正しい場所に作成され、適切に構成されている場合、DBCAサイレント・モードで、Microsoft Active DirectoryとOracle Databaseの統合のために新しいデータベースを作成または既存のデータベースを変更できます。

親トピック: Microsoft Active Directoryへの接続

6.2.2.7.1 Microsoft Active Directory接続の構成について

Microsoft Active Directory接続を構成するには、データベースでパラメータを設定するか、DBCAを使用します。

DBCAでは、集中管理ユーザー用に構成されたldap.oraのみが認識され、推奨されたデフォルトの場所にのみウォレットが作成されます。デフォルトのウォレット・ロケーションを使用するには、sqlnet.oraにWALLET_LOCATIONを設定しないでください。

ノート:

CMU-Active Directory には dsi.ora を使用することをお薦めします。

関連トピック

- Databaseシステム・パラメータを使用した手動アクセスの構成

親トピック: ステップ7: Microsoft Active Directory接続の構成

6.2.2.7.2 Databaseシステム・パラメータを使用した手動アクセスの構成

LDAP固有のOracle Databaseのシステム・パラメータを使用して、Active Directoryサービス接続を手動で構成できます。

1. dsi.oraファイルまたはldap.oraファイルを作成し、ウォレットを作成したことを確認します。
2. ALTER SYSTEMシステム権限を持つユーザーとして、データベース・インスタンスにログインします。

   たとえば、非マルチテナント・データベースでは次のようにします。

   ```
   sqlplus sec_admin
   Enter password: password
   ```

   マルチテナント環境では、適切なPDBにログインします。

   ```
   sqlplus sec_admin@pdb_name
   Enter password: password
   ```

   CDB内の使用可能なPDBを確認するには、CDBルート・コンテナにログインし、DBA_PDBSデータ・ディクショナリ・ビューのPDB_NAME列を問い合せます。現在のコンテナを確認するには、show con_nameコマンドを実行します。

3. LDAPディレクトリ・アクセスのタイプを決定するLDAP_DIRECTORY_ACCESSパラメータを変更します。

   マルチテナント環境を使用している場合は、CDBルートではなく各PDBで、LDAP_DIRECTORY_ACCESSを設定します。CDBルートでこのパラメータを設定すると、ルートにのみ適用され、PDBには適用されません。

   有効な値はPASSWORDおよびNONE (接続を無効化する場合)です。PASSWORDにはActive Directoryサーバー証明書が必要であり、ウォレットを作成する場合は、Oracle用のActive Directoryサービス・ユーザー・アカウントの資格証明を含める必要があります。

   たとえば:

   ```
   ALTER SYSTEM SET LDAP_DIRECTORY_ACCESS = 'PASSWORD';
   ```

   spfileまたはinit.oraファイルでこのパラメータを設定することもできます(init.oraファイルが使用される場合)。その後、データベースを再起動します。

4. LDAP_DIRECTORY_SYSAUTHパラメータをYESに設定して、Active Directoryの管理ユーザーがSYSDBA、SYSOPER、SYSBACKUP、SYSDG、SYSKMまたはSYSRAC管理権限を使用してOracle Databaseにログインできるようにします。
   CDBルートではなく各PDBでLDAP_DIRECTORY_SYSAUTHを設定します。CDBルートでこのパラメータを設定すると、ルートにのみ適用され、PDBには適用されません。
   このパラメータをNOに設定すると、Active Directoryの集中管理ユーザーは、これらの権限でOracleデータベースにログインできません。

   ```
   ALTER SYSTEM SET LDAP_DIRECTORY_SYSAUTH = YES SCOPE=SPFILE ;
   ```

spfileまたはinit.oraファイルでこのパラメータを設定することもできます(init.oraファイルが使用される場合)。その後、データベースを再起動します。

5. データベース・インスタンスを再起動するか、PDBを再度オープンします。
    - 非マルチテナント環境を使用している場合は、データベースを再起動します。

      ```
      SHUTDOWN IMMEDIATE
      STARTUP
      ```

    - マルチテナント環境を使用している場合は、PDBをクローズしてから再度オープンします。

      ```
      ALTER PLUGGABLE DATABASE pdb_name CLOSE IMMEDIATE;
      ALTER PLUGGABLE DATABASE pdb_name OPEN;
      ```

データベースを再起動したかPDBを再度オープンした後は、SYSDBA管理権限でログインし、次のようにLDAPパラメータ設定を確認できます。

```
show parameter ldap
```

親トピック: ステップ7: Microsoft Active Directory接続の構成

6.2.2.7.3 Database Configuration Assistant GUIを使用したアクセスの構成

Oracle Database Configuration Assistant (DBCA)は、LDAP接続構成を完了し、ウォレットを自動的に作成して、使用するActive Directory証明書を格納します。DBCAは、ldap.oraがCMU-ActiveDirectory用に構成されている場合にのみ動作します。

この手順では、Oracleソフトウェアがすでにインストールされており、集中管理ユーザーのActive Directoryサーバーを識別するためにldap.oraファイル(dsi.oraではなく)を使用していることを前提としています。データベース・ソフトウェアをまだインストールしていない場合は、Oracle Universal Installer (OUI)を使用してソフトウェアをインストールできます。その後、DBCAを使用してデータベースを作成すると同時に、Active Directoryの集中管理ユーザーの接続を構成できます。

1. Oracle Databaseソフトウェアがインストールされているホストに、管理権限を持つユーザーとしてログインします。
2. DBCAを起動します。
   デフォルトでは、DBCAユーティリティは$ORACLE_HOME/binディレクトリにあります。
   たとえば:

   ```
   cd $ORACLE_HOME/bin
   ./dbca
   ```

3. 「ネットワーク構成」オプション(またはデータベースの作成時にネットワーク構成オプションに進んだ場合)を選択します。「ネットワーク構成詳細の指定」ウィンドウが表示されます。「ディレクトリ・サービス統合」領域が表示されない場合、ldap.oraファイルが正しく構成されていません。以前に設定したldap.ora構成を確認し、ファイルを修正したら、DBCAを再実行してください。
4. 「ディレクトリ・サービス統合」領域で、次の手順を実行します。
    - 「サービス・ユーザー名」フィールドに、Oracleサービス・ディレクトリ・ユーザー・アカウントの名前を入力します。
    - 「パスワード」フィールドに、Oracleサービス・ディレクトリ・ユーザー・アカウントのパスワードを入力します。
    - 「サービス・ユーザーDN」フィールドに、Oracleサービス・ディレクトリ・ユーザー・アカウントのDNを入力します。DNはActive Directoryサーバーからまたは Active Directoryシステム管理者から直接取得できます。
    - 「アクセス・タイプ」では、リストから認証タイプを選択します(「PASSWORD」など)。(この設定により、LDAP_DIRECTORY_ACCESSパラメータが設定されます。)必要に応じて、「管理権限の認証を許可」チェックボックスを選択し、Active Directoryユーザーが管理権限(SYSDBA、SYSOPER、SYSBACKUPな

ど)を使用してデータベース・スキーマを認証および使用できるようにします。そうしないと、Active Directoryの集中管理ユーザーは、管理権限でデータベースにログインできません。(この設定は`LDAP_DIRECTORY_SYSAUTH`パラメータに対応しています。)

- 「証明書ファイルの場所」フィールドにActive Directory証明書のパスを指定します。マルチテナント環境では、DBCAがデータベース・インスタンス接続用のActive Directory接続を認識および設定します。別のActive Directoryサーバーを PDBに接続する場合は、PDB接続を手動で構成する必要があります。
- 「ウォレット・パスワード」および「パスワードの確認」フィールドに、Oracleサービス・ディレクトリ・ユーザー・アカウントの証明書および資格証明を格納するOracleウォレットのパスワードを入力し、確認のためもう一度入力します。その後、DBCAはサービス・ディレクトリ・ユーザー・アカウントを自動的に検証し、ウォレットを作成してユーザー資格証明を格納し、証明書をインポートします。

5. 「終了」ページになるまで、「次」をクリックします。
6. 「終了」をクリックします。

関連トピック

- ステップ4: dsi.oraまたはldap.oraファイルの作成
- Database Configuration Assistantのサイレント・モードを使用したアクセスの構成

親トピック: ステップ7: Microsoft Active Directory接続の構成

6.2.2.7.4 Database Configuration Assistantのサイレント・モードを使用したアクセスの構成

`ldap.ora` (`dsi.ora`ではない)が正しい場所に作成され、適切に構成されている場合、DBCAサイレント・モードで、Microsoft Active DirectoryとOracle Databaseの統合のために新しいデータベースを作成または既存のデータベースを変更できます。

1. 統合に使用されるOracleデータベースが含まれるホストにログインします。
2. 正しい場所に正しいコンテンツを使用して`ldap.ora`が作成されていることを確認してください。
3. `sqlnet.ora`ファイルに`WALLET_LOCATION`パラメータが指定されていないことを確認します。
4. サイレント・モードでDatabase Configuration Assistant(DBCA)を実行します。
   たとえば、単一インスタンスの非マルチテナント・データベースを作成するには、次のようにします。

```
cd $ORACLE_HOME/bin
./dbca -silent -createDatabase -gdbName inst1.production.examplecorp.com
-templateName General_Purpose.dbc -totalMemory 1000
-registerWithDirService true
-dirServiceUser oracle
-dirServiceUserName cn=oracle,cn=users,dc=production,dc=examplecorp,dc=com
-dirServicePassword service_user_password
-ldapDirectoryAccessType PASSWORD
-useSysAuthForLDAPAccess true
-dirServiceCertificatePath /tmp/AD_CA_Root_cert.txt
-walletPassword wallet_password
-sysPassword sys_password
-systemPassword system_password
```

   CDBまたは非マルチテナント・データベースのルート・コンテナを構成するには:

```
cd $ORACLE_HOME/bin
./dbca -silent -configureDatabase -sourceDB db_name
-registerWithDirService true
-dirServiceUser oracle
-dirServiceUserName cn=oracle,cn=users,dc=production,dc=examplecorp,dc=com
-dirServicePassword service_user_password
-ldapDirectoryAccessType PASSWORD
-useSYSAuthForLDAPAccess true
```

```
-dirServiceCertificatePath /tmp/AD_CA_Root_cert.txt
-walletPassword wallet_password
```

CDBでプラガブル・データベースを構成するには、次のようにします。

```
cd $ORACLE_HOME/bin
./dbca -silent -configurePluggableDatabase -pdbName pdb_name -sourceDB db_name
-registerWithDirService true
-dirServiceUser oracle
-dirServiceUserName cn=oracle,cn=users,dc=production,dc=examplecorp,dc=com
-dirServicePassword service_user_password
-dirServiceCertificatePath /tmp/AD_CA_Root_cert.txt
-walletPassword wallet_password
```

関連トピック

- ldap.oraファイルの使用について

親トピック: ステップ7: Microsoft Active Directory接続の構成

6.2.2.8 ステップ8: Oracleウォレットの確認

orapkiユーティリティでは、このデータベースのウォレットが正常に作成されたことを確認できます。

1. データベースが統合に使用されるホストにログインします。
2. ウォレットを含むディレクトリに移動します。

   WALLET_LOCATIONがsqlnet.oraに設定されていない場合、デフォルトのウォレット・ロケーションは次のとおりです。

   非マルチテナント環境では、walletディレクトリは$ORACLE_BASE/admin/db_unique_name/walletディレクトリにあります。

   マルチテナント環境では、次のいずれかの場所にあります。

   - CDBルートの場合、ウォレットは$ORACLE_BASE/admin/db_unique_name/wallet/ディレクトリにあります。
   - PDBの場合、ウォレットは$ORACLE_BASE/admin/db_unique_name/pdb_guid/wallet/ディレクトリにあります。
3. コマンドラインで、次のコマンドを入力します。

   ls -ltr wallet_location (walletディレクトリにウォレット・ファイルが含まれていることを確認するため)

   たとえば:

```
$ ls -ltr $ORACLE_BASE/admin/db_unique_name/pdb_guid/wallet/
total 12
-rw------- 1 creator_user creator_group 1597 Nov 27 22:47 cwallet.sso
-rw------- 1 creator_user creator_group 1552 Nov 27 22:47 ewallet.p12
-rw-rw-r-- 1 creator_user creator_group 86   Nov 27 22:48 dsi.ora
```

   orapki wallet display -wallet wallet_location (Oracleシークレット・ストアのエントリを検索するため)

   出力には次のエントリが含まれている必要があります。

```
Requested Certificates:
User Certificates:
Oracle Secret Store entries:
ORACLE.SECURITY.DN
ORACLE.SECURITY.PASSWORD
```

```
ORACLE.SECURITY.USERNAME
Trusted Certificates:
Subject: CN=ADSVR,DC=production,DC=examplecorp,DC=com
```

親トピック: Microsoft Active Directoryへの接続

6.2.2.9 ステップ9: 統合のテスト

統合をテストするには、ORACLE_HOME、ORACLE_BASE,およびORACLE_SID環境変数を設定してから、LDAPパラメータの設定を確認する必要があります。

1. データベースが統合に使用されるホストにログインします。
2. ORACLE_HOME、ORACLE_BASEおよびORACLE_SID環境変数を設定します。
   たとえば:

   ```
   export ORACLE_HOME=/app/product/18.1/dbhome_1
   export ORACLE_BASE=/app
   export ORACLE_SID=sales_db
   ```

3. SYSDBA管理権限を持つユーザーとしてデータベース・インスタンスにログインします。
   たとえば:

   ```
   sqlplus sec_admin as sysdba
   Enter password: password
   ```

   マルチテナント環境では、適切なPDBにログインします。たとえば:

   ```
   sqlplus sec_admin@pdb_name as sysdba
   Enter password: password
   ```

   CDB内の使用可能なPDBを確認するには、CDBルート・コンテナにログインし、DBA_PDBSデータ・ディクショナリ・ビューのPDB_NAME列を問い合せます。現在のコンテナを確認するには、show con_nameコマンドを実行します。

4. 次のようにLDAPパラメータの設定を確認します。

   ```
   show parameter ldap
   ```

   出力は次のようになります。

   ```
   NAME                           TYPE        VALUE
   ------------------------------ ---------   -----------------
   ldap_directory_access          string      PASSWORD
   ldap_directory_sysauth         string      YES
   ```

親トピック: Microsoft Active Directoryへの接続

6.3 集中管理ユーザーの認証の構成

パスワード認証、Kerberos認証または公開キー・インフラストラクチャ(PKI)認証を構成できます。

- 集中管理ユーザーに対するパスワード認証の構成
  集中管理ユーザーに対するパスワード認証の構成では、Active Directoryでパスワード・フィルタを使用して、Oracle Databaseのパスワード・ベリファイアをActive Directoryに生成および格納する必要があります。
- 集中管理ユーザー用のKerberos認証の構成
  Kerberos認証を使用する予定の場合は、Microsoft Active Directoryに統合するOracle DatabaseでKerberosを構成する必要があります。

- 集中管理ユーザーのPKI証明書を使用した認証の構成
  集中管理ユーザーの認証にPKI証明書を使用する予定の場合は、Microsoft Active Directoryと統合されるOracleデータベースでTransport Layer Securityを構成する必要があります。

親トピック: Microsoft Active Directoryによる集中管理ユーザーの構成

6.3.1 集中管理ユーザーに対するパスワード認証の構成

集中管理ユーザーに対するパスワード認証の構成では、Active Directoryでパスワード・フィルタを使用して、Oracle Databaseのパスワード・ベリファイアをActive Directoryに生成および格納する必要があります。

- 集中管理ユーザーに対するパスワード認証の構成について
  パスワード認証を構成するには、パスワード・フィルタをデプロイし、ユーザー属性を追加することによってActive Directoryスキーマを拡張し、Active Directoryで異なるバージョンのパスワード・ベリファイアを生成するためのグループを作成する必要があります。
- 集中管理ユーザーのパスワード認証の構成
  Active Directoryサーバーでパスワード認証構成を実行する必要があり、Active Directoryユーザーが管理権限でOracleデータベースにログインする必要がある場合は、Oracleデータベースでもパスワード認証構成を実行する必要があります。
- パスワード認証を使用したOracle Databaseへのログイン
  パスワード認証を使用する場合、集中管理ユーザーにはデータベースへのログイン方法の選択肢があります。

親トピック:集中管理ユーザーの認証の構成

6.3.1.1 集中管理ユーザーに対するパスワード認証の構成について

パスワード認証を構成するには、パスワード・フィルタをデプロイし、ユーザー属性を追加することによってActive Directoryスキーマを拡張し、Active Directoryで異なるバージョンのパスワード・ベリファイアを生成するためのグループを作成する必要があります。

Active Directoryユーザーが管理権限を使用してOracle Databaseにログインするには、Oracle Databaseでパスワード・ファイルも設定する必要があります。

パスワード認証の場合、Oracle DatabaseはActive Directoryで認証するために`ldapbind`コマンドを介してActive Directoryユーザーのパスワードを渡さないため、Oracleフィルタをインストールし、Active Directoryスキーマを拡張する必要があります。Active DirectoryにインストールするOracleフィルタにより、Active Directoryユーザーがパスワードを更新するときにOracle固有のパスワード・ベリファイアが作成されます。Oracleフィルタは、最初のインストール時には必要なOracleパスワード・ベリファイアをすべて生成するわけではありません。ユーザーがActive Directoryパスワードを変更したときに、そのユーザーのOracleパスワード・ベリファイアのみを生成します。

(サイトで必要な場合に)下位互換性を維持するため、Oracleフィルタでは、リリース11g、12c、および18cのOracle Databaseクライアントで機能するパスワード・ベリファイアを生成できます。Oracleパスワード・フィルタは、`ORA_VFR_MD5` (`WebDAV`の場合)、`ORA_VFR_11G` (リリース11gの場合)および`ORA_VFR_12C` (リリース12cおよび18cの場合)という名前のActive Directoryグループを使用して、生成するOracle Databaseパスワード・ベリファイアを決定します。グループ・メンバー・ユーザーに対してOracleパスワード・ベリファイアを生成するには、Active Directoryで空のグループを作成する必要があります。これらは、Active Directoryユーザー用に生成される特定のベリファイアを指定する個別のグループです。たとえば、10人のディレクトリ・ユーザーが、Oracle Databaseリリース18cおよび12cクライアントとのみ通信する、新たに作成されたOracle Databaseリリース18cデータベースにログインする必要がある場合、Active Directoryグループ`ORA_VFR_12C`は10人のActive Directoryユーザーがメンバーになります。この10人のActive DirectoryユーザーがActive Directoryでパ

スワードを変更すると、Oracleフィルタではこれらのユーザーの12cベリファイアのみが生成されます(18cのベリファイアは12cのベリファイアと同じ)。Active Directoryユーザーが Oracleデータベースにログインする必要がもうない場合、Active Directoryユーザー用に生成されたOracleパスワード・ベリファイアをクリアするには、ORA_VFR*グループ*からユーザーを削除し、このユーザーのパスワードを再設定します(またはパスワードの変更が必要)。このユーザーの`orclCommonAttribute`属性を手動でクリアすることもできます。Oracleパスワード・ベリファイアは、ユーザーがORA_VFRグループから削除されると生成されなくなります。

親トピック: 集中管理ユーザーに対するパスワード認証の構成

6.3.1.2 集中管理ユーザーのパスワード認証の構成

Active Directoryサーバーでパスワード認証構成を実行する必要があり、Active Directoryユーザーが管理権限でOracleデータベースにログインする必要がある場合は、Oracleデータベースでもパスワード認証構成を実行する必要があります。

1. Oracle Databaseのパスワード・フィルタをデプロイし、Active Directoryスキーマを拡張します。
   このタスクを実行するためのユーティリティ・ツール`opwdintg.exe`は、`$ORACLE_HOME/bin`にあります。このユーティリティは、Active Directoryにパスワード・フィルタをインストールし、Oracleパスワード・ベリファイアを保持するActive Directoryスキーマを拡張して、Active Directoryパスワード・ベリファイア・グループを作成します。WebDAV、11Gおよび12Cパスワード・ベリファイアを使用してクライアントに接続すると、パスワード・フィルタにより、Microsoft Active Directoryのユーザー・アカウントをOracleデータベースで認証できます。
   a. `opwdintg.exe`実行可能ファイルをデプロイするには、このファイルをActive Directoryサーバーにコピーし、Active Directory管理者に`opwdintg.exe`ユーティリティ・ツールを実行させます。
   b. ユーザー・グループを作成および管理する権限を持つユーザーとして、Microsoft Active Directoryにログインします。
   c. パスワード・ベリファイアのユーザー・グループORA_VFR_MD5、ORA_VFR_11GおよびORA_VFR_12Cを確認します。これらのグループが存在しない場合は、`opwdintg.exe`ユーティリティ・ツールを再実行します。
   d. 次のガイドラインに従い、Oracle Databaseを使用するMicrosoft Active Directoryユーザーをこれらのグループに追加します。
      - クライアントまたはサーバーのいずれかでのみOracle Databaseリリース12c認証が許可される場合、ユーザーをORA_VFR_12Cグループに追加します。(Oracle Databaseリリース18cでは、Oracle Databaseリリース12cと同じベリファイアが使用されます)。
      - クライアントとサーバーの両方でOracle Databaseリリース12cより低い認証のみが許可される場合(つまり、Oracle Databaseリリース11gまたは12.1.0.1クライアントを使用している場合)、ユーザーをORA_VFR_11G グループに追加します。
      - Oracle Database WebDAVクライアントを介してユーザーを認証する必要がある場合、ユーザーはORA_VFR_MD5グループのメンバーである必要があります。

   この構成により、Oracle Databaseパスワード・ベリファイアの生成を詳細に制御できます。必要なユーザーに必要なベリファイアのみが生成されます。たとえば、Microsoft Active Directoryユーザー`pfitch`がORA_VFR_12CおよびORA_VFR_11Gグループに追加されると、12Cと11Gの両方のベリファイアが`pfitch`に対して生成されます。これにより、Oracle Databaseリリース11gクライアントに対し、可能な場合には最もセキュアかつ強力なベリファイアが選択されますが、それ以外の場合は11Gベリファイアが選択されます。

2. データベース・パスワード・ファイルをバージョン12.2に更新します。
   Active Directoryユーザーが管理権限でOracleデータベースにログインする必要がある場合は、データベース・パス

ワード・ファイルをバージョン12.2に更新します。

a. 管理者権限を持つユーザーとして、Microsoft Active Directory接続に使用するデータベースが存在するホストにログインします。

b. $ORACLE_HOME/dbsディレクトリに移動します。

c. ORAPWDユーティリティを実行して、フォーマットを12.2に設定します。

たとえば:

```
orapwd FILE='/app/oracle/product/18.1/db_1/dbs/orapwdb181' FORMAT=12.2
```

この設定により、SYSOPOERやSYSBACKUPなどの様々な管理権限をグローバル・ユーザーに付与できます。

d. ALTER SYSTEM権限を持つユーザーとして、データベース・インスタンスにログインします。

e. spfileまたはinit.oraファイルでLDAP_DIRECTORY_SYSAUTHパラメータがYESに設定されていることを確認します。

f. REMOTE_LOGIN_PASSWORDFILEパラメータをspfileまたはinit.oraファイルでEXCLUSIVEに設定します。

g. データベース・インスタンスを再起動します。

```
SHUTDOWN IMMEDIATE
STARTUP
```

関連トピック

- ステップ2: パスワード認証のためにパスワード・フィルタをインストールしてMicrosoft Active Directoryスキーマを拡張

親トピック: 集中管理ユーザーに対するパスワード認証の構成

6.3.1.3 パスワード認証を使用したOracle Databaseへのログイン

パスワード認証を使用する場合、集中管理ユーザーにはデータベースへのログイン方法の選択肢があります。

Active Directoryユーザーがパスワード認証を使用している場合にActive Directoryに接続するように構成されているデータベースにログインするには、次のログオン・ユーザー名構文を使用できます。

```
sqlplus /nolog
connect "Windows_domain¥Active_Directory_user_name"@tnsname_of_database
Password: password
```

次の接続では、Windowsドメイン名がproductionであると想定しています。

```
connect "production¥pfitch"@inst1
```

Active Directoryユーザーが、データベース・ウォレットで構成されたOracleサービス・ディレクトリ・ユーザー・アカウントと同じActive Directoryドメインにある場合、Active Directoryユーザーは、このユーザー名(samAccountName)を使用してデータベースに直接ログオンできます。

```
sqlplus samAccountName@tnsname_of_database
Enter password: password
```

たとえば:

```
connect pfitch@instl
Enter password: password
```

または、ユーザーはActive DirectoryのWindowsユーザー・ログオン名をDNSドメイン名とともに使用することもできます。

```
connect "Active_Directory_user_name@Windows_DNS_domain_name"@tnsname_of_database
Password: password
```

たとえば:

```
connect "pfitch@production.examplecorp.com"@inst1
```

親トピック: 集中管理ユーザーに対するパスワード認証の構成

6.3.2 集中管理ユーザー用のKerberos認証の構成

Kerberos認証を使用する予定の場合は、Microsoft Active Directoryと統合するOracleデータベースでKerberosを構成する必要があります。

CMU-Active DirectoryはMicrosoft Active Directory Kerberosサーバーのみをサポートします。その他の非Active Directory Kerberosサーバーは、CMU-Active Directoryではサポートされていません。

ノート:

Active Directory ユーザーの Kerberos UPN として外部で識別されるデータベース・ユーザーは作成しません。かわりに、Active Directory ユーザーまたはグループにマップされたグローバル・ユーザーを使用します。

関連トピック

- 共有データベース・グローバル・ユーザーへのディレクトリ・グループのマッピング
- データベース・グローバル・ユーザーへのディレクトリ・ユーザーの排他的マッピング
- Kerberos認証の有効化

親トピック:集中管理ユーザーの認証の構成

6.3.3 集中管理ユーザーのPKI証明書を使用した認証の構成

集中管理ユーザーの認証にPKI証明書を使用する予定の場合は、Microsoft Active Directoryと統合されるOracleデータベースでTransport Layer Securityを構成する必要があります。

CMUでのKerberos認証ではMicrosoft Active Directory-Active Directory Kerberosサーバーを使用する必要がありますが、PKI認証では、Microsoft Active Directory-Active Directoryのサービスのみでなくサード・パーティCAサービスを使用できます。

ノート:

Active Directory ユーザー証明書は、Transport Layer Security 認証を構成する際に使用します。ただし、Active Directory ユーザー証明書の DN として外部で識別されるデータベース・ユーザーは作成しません。かわりに、Active Directory ユーザーまたはグループにマップされたグローバル・ユーザーを使用します。

関連トピック

- 共有データベース・グローバル・ユーザーへのディレクトリ・グループのマッピング
- データベース・グローバル・ユーザーへのディレクトリ・ユーザーの排他的マッピング
- Transport Layer Security認証の構成
- Oracle環境における公開キー・インフラストラクチャ

6.4 集中管理ユーザーの認可の構成

集中管理ユーザーにより、Active DirectoryユーザーがOracleデータベースにアクセスするための認可を管理できます。

Active Directoryを使用して組織でユーザーを追加、変更、または削除するときには、組織のすべてのデータベースからユーザーを追加、変更または削除する必要はありません。

- 集中管理ユーザーの認可の構成について
  Active Directory内のデータベースに対するユーザー認可を管理できます。
- 共有データベース・グローバル・ユーザーへのディレクトリ・グループのマッピング
  データベースのほとんどのユーザーは、ディレクトリ・グループのメンバーシップを介して共有グローバル・データベース・ユーザー(スキーマ)にマップされます。
- グローバル・ロールへのディレクトリ・グループのマッピング
  ディレクトリ・グループにデータベース・グローバル・ロールをマップすると、そのメンバーにはログイン・スキーマを介して付与された権限およびロールを超える権限およびロールが付与されます。
- データベース・グローバル・ユーザーへのディレクトリ・ユーザーの排他的マッピング
  Microsoft Active DirectoryユーザーをOracleデータベース・グローバル・ユーザーに排他的にマップできます。
- ユーザー・マッピング定義の変更または移行
  ALTER USER文を使用して、Active Directoryユーザーからデータベース・グローバル・ユーザーへのマッピングを更新できます。
- 管理ユーザーの構成
  管理ユーザーは、過去と同様に機能しますが、共有スキーマを使用している場合、CMUを使用すると集中的な認証および認可によって制御できます。
- 集中管理ユーザー・ログオン情報の確認
  集中管理ユーザーを構成および認可した後、Oracleデータベース側で一連のSQL問合せを実行して、ユーザー・ログオン情報を検証できます。

親トピック: Microsoft Active Directoryによる集中管理ユーザーの構成

6.4.1 集中管理ユーザーの認可の構成について

Active Directory内のデータベースに対するユーザー認可を管理できます。

ほとんどのOracle Databaseユーザーは、共有データベース・スキーマ(ユーザー)にマップされます。そのため、ディレクトリ・ユーザーが採用されるときや、社内でジョブが変わるとき、あるいは離職するときに各Oracleデータベースで実行する必要がある作業が最小限に抑えられます。ディレクトリ・ユーザーは、Oracleデータベース・グローバル・ユーザー(スキーマ)にマップされたActive Directoryグループに割り当てられます。ユーザーがデータベースにログインすると、データベースがActive Directoryを問い合せ、そのユーザーがメンバーになっているグループを検索します。デプロイメントが共有スキーマを使用している場合、いずれかのグループが共有データベース・スキーマにマップされ、ユーザーはそのデータベース・スキーマに割り当てられます。ユーザーは、データベース・スキーマに付与されたロールと権限を所持します。複数のユーザーが同じ共有データベース・スキーマに割り当てられるため、最小限のロールと権限のセットのみを共有スキーマに付与する必要があります。場合によっては、共有スキーマに権限およびロールを付与しないでください。ユーザーには、データベース・グローバル・ロールを介して適切なロールとスキーマのセットが割り当てられます。グローバル・ロールはActive Directoryグループにマップされます。このように、同じデータベース共有スキーマにマップされている場合でも、異なるユーザーが異なるロールおよび権限を持つことができます。新しく採用されたユーザーは、共有スキーマにマップされたActive Directoryグループに割り当てられたうえで、タスクを完了するために必要な追加のロールと権

限を取得するために、グローバル・ロールにマップされた1つ以上の追加グループに割り当てられます。共有スキーマとグローバル・ロールを組み合せると、データベース操作に対する変更を最小限にして、集中的な認可管理が可能になります。データベースは最初に、適切なActive Directoryグループにマップされている共有スキーマとグローバル・ロールのセットでプロビジョニングされる必要がありますが、ユーザー認可管理はActive Directory内で発生する可能性があります。

Active Directoryユーザーは、データベース・グローバル・ユーザーに排他的にマップすることもできます。これには、Active Directoryユーザーに直接マップされているデータベースで新しいユーザーが必要です。新しいユーザーと離職するユーザーには、メンバーである各データベースの更新が必要です。

SYSOPERやSYSBACKUPなどの管理権限を必要とするActive Directoryユーザーに、グローバル・ロールを介してこれらを付与することはできません。管理権限は、ロールではなくスキーマにのみ付与できます。ただし、管理権限のある場合でも、共有スキーマを使用してユーザー認可管理を容易にできます。SYSOPER権限で共有スキーマを使用すると、データベースに新しいユーザー・スキーマを作成しなくても、SYSOPERを使用してスキーマにマップされたActive Directoryグループに、新しいユーザーを簡単に追加できます。共有スキーマに1人のユーザーしか割り当てられていない場合でも、一元的に管理できます。

グローバル・ロールを使用してユーザーに権限およびロールを付与する場合、セッションで有効にできるロールの最大数は150であることに注意してください。

認可でサポートされているグローバル・ユーザー・マッピングのタイプを次に示します。

- 共有グローバル・ユーザーのマッピング。ディレクトリ・ユーザーは、共有スキーマへのディレクトリ・グループのマッピングを介して、共有データベース・スキーマ(ユーザー)に割り当てられます。グループのメンバーであるディレクトリ・ユーザーはこの共有スキーマを介してデータベースに接続できます。共有スキーマを使用すると、Active Directoryでユーザー認可を集中管理できます。
- 排他的なグローバル・ユーザー・マッピングでは、専用データベース・ユーザーがディレクトリ・ユーザーに排他的にマップされます。共有データベース・スキーマほど一般的ではありませんが、このユーザーは、2層または3層アプリケーションのスキーマ・ユーザーまたはSQL*Plusを使用して、直接データベース・アクセスのために作成されています。 認可の管理を容易にするため、グローバル・ロールを介してこれらのユーザーにデータベース権限を付与することをお薦めします。 ただし、Oracleデータベースでこれらのユーザーに権限を直接付与することもできますが、これはお薦めしません。 これは、2層および3層アプリケーションではグローバル・ユーザーをデータベース・スキーマとして使用でき、グローバル・ユーザーはスキーマ・オブジェクトに対し所有者として完全なデータベース権限を持つためです。

ディレクトリ・ユーザーが複数のグループのメンバーであるのは一般的です。ただし、共有スキーマにマップする必要があるのは、これらのグループのうち1つだけです。

親トピック: 集中管理ユーザーの認可の構成

6.4.2 共有データベース・グローバル・ユーザーへのディレクトリ・グループのマッピング

データベースのほとんどのユーザーは、ディレクトリ・グループのメンバーシップを介して共有グローバル・データベース・ユーザー(スキーマ)にマップされます。

Active Directoryグループは、データベース・グローバル・ユーザーをマップする前に作成する必要があります。ユーザーがデータベースにログインするには、いつでもActive Directoryユーザーをグループに追加できます。データベース側でこれらのマッピングを実行するには、CREATE USERおよびALTER USER権限が必要です。この構成は、パスワード認証、Kerberos認証、および公開キー・インフラストラクチャ(PKI)認証方式を使用するユーザーが使用できます。
アプリケーションに対して同じデータベース・スキーマを共有するユーザーをActive Directoryグループに割り当てることができます。共有Oracle Databaseグローバル・ユーザー(共有スキーマ)がActive Directoryグループにマップされます。これにより、このグループのすべてのActive Directoryユーザーは、その共有グローバル・ユーザー・アカウントを使用してデータベースにログ

インできます。データベース・グローバル・ユーザー・アカウントはグループ・メンバーにより共有されますが、Active Directoryユーザーの認証済アイデンティティ(Windowsドメインとユーザーの`samAccountName`)およびエンタープライズ・アイデンティティ(DN)は、データベース内で追跡および監査されます。

1. `CREATE USER`または`ALTER USER`システム権限が付与されたユーザーとして、データベース・インスタンスにログインします。
2. Active Directoryグループの`DN`を指定する`IDENTIFIED GLOBALLY AS`句を使用して`CREATE USER`または`ALTER USER`文を実行します。
   たとえば、`production.examplecorp.com`ドメインの`sales`組織単位の`widget_sales_group`というディレクトリ・グループを、`WIDGET_SALES`という名前の共有データベース・グローバル・ユーザーにマップするには、次のようにします。

   ```
   CREATE USER widget_sales IDENTIFIED GLOBALLY AS
   'CN=widget_sales_group,OU=sales,DC=production,DC=examplecorp,DC=com';
   ```

   `widget_sales_group`のすべてのメンバーは、データベースにログインするときに`widget_sales`共有スキーマに割り当てられます。

親トピック: 集中管理ユーザーの認可の構成

6.4.3 グローバル・ロールへのディレクトリ・グループのマッピング

ディレクトリ・グループにデータベース・グローバル・ロールをマップすると、そのメンバーにはログイン・スキーマを介して付与された権限およびロールを超える権限およびロールが付与されます。

1. `CREATE ROLE`または`ALTER ROLE`システム権限が付与されたユーザーとして、データベース・インスタンスにログインします。
2. Active Directoryグループの`DN`を指定する`IDENTIFIED GLOBALLY AS`句を使用して`CREATE ROLE`または`ALTER ROLE`文を実行します。
   たとえば、`production.examplecorp.com`ドメインの`sales`組織単位の`widget_sales_group`という名前のディレクトリ・ユーザー・グループを、データベース・グローバル・ロール`WIDGET_SALES_ROLE`にマップするには、次のようにします。

   ```
   CREATE ROLE widget_sales_role IDENTIFIED GLOBALLY AS
   'CN=widget_sales_group,OU=sales,DC=production,DC=examplecorp,DC=com';
   ```

   マルチテナント環境で`C##WIDGET_SALES_ROLE`という共通ロールを作成するには、次のようにします。

   ```
   CREATE ROLE c##widget_sales_role IDENTIFIED GLOBALLY AS
   'CN=widget_sales_group,OU=sales,DC=production,DC=examplecorp,DC=com'
   CONTAINER = ALL;
   ```

   `widget_sales_group`のすべてのメンバーは、データベースにログインするときにデータベース・ロール`widget_sales_role`で認可されます。

親トピック: 集中管理ユーザーの認可の構成

6.4.4 データベース・グローバル・ユーザーへのディレクトリ・ユーザーの排他的マッピング

Microsoft Active DirectoryユーザーをOracle Databaseグローバル・ユーザーに排他的にマッピングできます。

この構成はOracle Database側のみで実行し、Active Directory側では実行しません。これらのマッピングを実行するには、`CREATE USER`および`ALTER USER`権限が必要です。この構成は、パスワード認証、Kerberos認証、および公開キー・イ

ンフラストラクチャ(PKI)認証方式を使用するユーザーが使用できます。

1. CREATE USERまたはALTER USERシステム権限が付与されたユーザーとして、データベース・インスタンスにログインします。
2. Active DirectoryユーザーのDNを指定するIDENTIFIED GLOBALLY AS句を使用してCREATE USERまたはALTER USER文を実行します。
   たとえば、production.examplecorp.comドメインのsales組織単位のPeter Fitch (samAccountNameはpfitch)という名前の既存のActive Directoryユーザーを、PETER_FITCHという名前のデータベース・グローバル・ユーザーにマップするには、次のようにします。

```
CREATE USER peter_fitch IDENTIFIED GLOBALLY AS
'CN=Peter Fitch,OU=sales,DC=production,DC=examplecorp,DC=com';
```

親トピック: 集中管理ユーザーの認可の構成

6.4.5 ユーザー・マッピング定義の変更または移行

ALTER USER文を使用して、Active Directoryユーザーからデータベース・グローバル・ユーザーへのマッピングを更新できます。

更新できるユーザーは、CREATE USER文の句IDENTIFIED BY password、IDENTIFIED EXTERNALLYまたはIDENTIFIED GLOBALLYのいずれかを使用してアカウントが作成されているユーザーです。これは、CUを使用してユーザーを移行するときに便利です。たとえば、Kerberosに対して外部認証されるデータベース・ユーザーは、ユーザー・プリンシパル名(UPN)で識別されます。Kerberos認証でCMUを使用するためにユーザーを移行するには、ALTER USER文を実行してグローバル・ユーザーを宣言し、Active Directory識別名(DN)でそのユーザーを識別する必要があります。

1. ALTER USERシステム権限が付与されたユーザーとして、データベース・インスタンスにログインします。
2. IDENTIFIED GLOBALLY AS句を指定してALTER USER文を実行します。
   たとえば:

```
ALTER USER peter_fitch IDENTIFIED GLOBALLY AS
'CN=Peter Fitch,OU=sales,DC=production,DC=examplecorp,DC=com';
```

親トピック: 集中管理ユーザーの認可の構成

6.4.6 管理ユーザーの構成

管理ユーザーは、過去と同様に機能しますが、共有スキーマを使用している場合、CMUを使用すると集中的な認証および認可によって制御できます。

- 共有アクセス・アカウントを使用したデータベース管理ユーザーの構成
  共有アカウントを使用すると、組織で採用、異動、離職があるときに複数データベースのデータベース管理者の管理が簡略化されます。
- 排他的マッピングを使用したデータベース管理ユーザーの構成
  データベース管理者を、データベース内の排他スキーマにマップすることもできます。

親トピック: 集中管理ユーザーの認可の構成

6.4.6.1 共有アクセス・アカウントを使用したデータベース管理ユーザーの構成

共有アカウントを使用すると、組織で採用、異動、離職があるときに複数データベースのデータベース管理者の管理が簡略化されます。

新しいOracle データベース・アカウントを作成せずに、Active Directoryグループを使用して複数のデータベースで共有アカ

ウントに新しいデータベース管理者を割り当てることができます。

1. 現在のデータベース・インスタンスのパスワード・ファイルが12.2形式であることを確認してください。

```
orapwd file=pwd_file FORMAT=12.2
Enter password for SYS: password
```

2. Active DirectoryでActive Directoryグループを作成します(ad_dba_backup_usersという名前のデータベース管理者バックアップ・ユーザー・グループなど)。
3. Oracle Databaseで、グローバル・ユーザー(共有スキーマなど)を作成し(db_dba_backup_global_userなど)、このユーザーをActive Directory ad_dba_backup_usersグループにマップします。
4. グローバル・ユーザーdb_dba_backup_global_userにSYSBACKUP管理権限を付与します。

この段階では、Active Directoryのad_dba_backup_usersグループに追加されているすべてのActive Directoryユーザーが、SYSBACKUP管理権限を持つ新しいデータベース共有スキーマに割り当てられます。

親トピック: 管理ユーザーの構成

6.4.6.2 排他的マッピングを使用したデータベース管理ユーザーの構成

データベース管理者を、データベース内の排他スキーマにマップすることもできます。

1. 現在のデータベース・インスタンスのパスワード・ファイルが12.2形式であることを確認してください。

```
orapwd file=pwd_file FORMAT=12.2
Enter password for SYS: password
```

2. ユーザーの作成と他のユーザーへの管理権限の付与を実行できるユーザーとして、データベース・インスタンスにログインします。
3. データベース・グローバル・ユーザーを作成します。
   たとえば:

```
CREATE USER peter_fitch IDENTIFIED GLOBALLY AS
'CN=Peter Fitch,OU=sales,DC=production,DC=examplecorp,DC=com';
```

4. 管理権限をこのユーザーに付与します。
   たとえば、SYSKM管理権限をユーザーに付与するには次のようにします。

```
GRANT SYSKM TO peter_fitch;
```

アカウントの保守作業の量と、データベースとActive Directoryの両方におけるマッピングの点から、より集中化された手法は、場合によっては共有データベース・アカウントに1人のActive Directoryユーザーのみが割り当てられていても、これらの管理アカウントにも共有スキーマを使用する方法です。

親トピック: 管理ユーザーの構成

6.4.7 集中管理ユーザー・ログオン情報の確認

集中管理ユーザーを構成および認可した後、Oracleデータベース側で一連のSQL問合せを実行して、ユーザー・ログオン情報を検証できます。

1. ユーザーが構成および認可したActive Directoryから集中管理ユーザーとしてデータベースにログインします。
   たとえば、データベース・インスタンスinst1に、Windowsドメインproductionのエンタープライズ・ユーザーpfitchとしてログインするには、次のようにします。

```
sqlplus /nolog
```

```
connect "production¥pfitch"@inst1
Enter password: password
```

2. マップされたグローバル・ユーザーを検証します。
   マップされたグローバル・ユーザーは、集中管理ユーザー認可を持つデータベース・ユーザー・アカウントです。ユーザーPETER_FITCHはActive Directoryユーザーpfitchの排他的マッピングを持つグローバル・ユーザーとみなされ、ユーザーWIDGET_SALESは、pfitchがメンバーであるActive Directoryグループwidget_sales_groupの共有マッピングを持つグローバル・ユーザーとみなされます。グローバル・ユーザー・アカウントには独自のスキーマがあります。

   ```
   SHOW USER;
   ```

   排他的マッピングであるか共有マッピングであるかに応じて、次のような出力が表示されます。

   ```
   USER is "PETER_FITCH"
   ```

   または

   ```
   USER is "WIDGET_SALES"
   ```

3. 集中管理ユーザーに付与されたロールを確認します。

   ```
   SELECT ROLE FROM SESSION_ROLES ORDER BY ROLE;
   ```

   次のような出力が表示されます。

   ```
   ROLE
   ------------------------------------------------------------------------
   WIDGET_SALES_ROLE
   ...
   ```

4. 次の問合せを実行して、このデータベース・セッションで使用されている現在のスキーマのSYS_CONTEXTネームスペース値、現在のユーザー名、セッション・ユーザー名、認証方式、認証済アイデンティティ、エンタープライズ・アイデンティティ、識別タイプおよびLDAPサーバー・タイプを確認します。

   - このデータベース・セッションで使用されている現在のスキーマを確認します。データベース・スキーマは、含まれているオブジェクトを識別するオブジェクト・コンテナです。現在のスキーマは、このデータベース・セッションのオブジェクト名解決のデフォルト・コンテナです。

     ```
     SELECT SYS_CONTEXT('USERENV', 'CURRENT_SCHEMA') FROM DUAL;
     ```

     排他的マッピングであるか共有マッピングであるかに応じて、次のような出力が表示されます。

     ```
     SYS_CONTEXT('USERENV','CURRENT_SCHEMA')
     ------------------------------------------------------------------------
     PETER_FITCH
     ```

     または

     ```
     SYS_CONTEXT('USERENV','CURRENT_SCHEMA')
     ------------------------------------------------------------------------
     WIDGET_SALES
     ```

   - 現行ユーザーを確認します。この場合、現行ユーザーは現行スキーマと同じです。

     ```
     SELECT SYS_CONTEXT('USERENV', 'CURRENT_USER') FROM DUAL;
     ```

     排他的マッピングであるか共有マッピングであるかに応じて、次のような出力が表示されます。

```
SYS_CONTEXT('USERENV','CURRENT_USER')
--------------------------------------------------------------------------------
PETER_FITCH
```

または

```
SYS_CONTEXT('USERENV','CURRENT_USER')
--------------------------------------------------------------------------------
WIDGET_SALES
```

- セッション・ユーザーを確認します。

```
SELECT SYS_CONTEXT('USERENV', 'SESSION_USER') FROM DUAL;
```

排他的マッピングであるか共有マッピングであるかに応じて、次のような出力が表示されます。

```
SYS_CONTEXT('USERENV','SESSION_USER')
--------------------------------------------------------------------------------
PETER_FITCH
```

または

```
SYS_CONTEXT('USERENV','SESSION_USER')
--------------------------------------------------------------------------------
WIDGET_SALES
```

- 認証方式を確認します。

```
SELECT SYS_CONTEXT('USERENV', 'AUTHENTICATION_METHOD') FROM DUAL;
```

次のような出力が表示されます。

```
SYS_CONTEXT('USERENV','AUTHENTICATION_METHOD')
--------------------------------------------------------------------------------
PASSWORD_GLOBAL
```

- エンタープライズ・ユーザーの認証済アイデンティティを確認します。このユーザーがデータベースにログオンしたときに、Active Directory認証済ユーザー・アイデンティティが取得されて監査されます。

```
SELECT SYS_CONTEXT('USERENV', 'AUTHENTICATED_IDENTITY') FROM DUAL;
```

次のような出力が表示されます。

```
SYS_CONTEXT('USERENV','AUTHENTICATED_IDENTITY')
--------------------------------------------------------------------------------
production¥pfitch
```

- 集中管理ユーザーのエンタープライズ・アイデンティティを確認します。

```
SELECT SYS_CONTEXT('USERENV', 'ENTERPRISE_IDENTITY') FROM DUAL;
```

次のような出力が表示されます。

```
SYS_CONTEXT('USERENV','ENTERPRISE_IDENTITY')
--------------------------------------------------------------------------------
cn=Peter Fitch,ou=sales,dc=production,dc=examplecorp,dc=com
```

- 識別タイプを確認します。

```
SELECT SYS_CONTEXT('USERENV', 'IDENTIFICATION_TYPE') FROM DUAL
```

排他的マッピングであるか共有マッピングであるかに応じて、次のような出力が表示されます。

```
SYS_CONTEXT('USERENV','IDENTIFICATION_TYPE')
------------------------------------------------------------------------
GLOBAL EXCLUSIVE
```

または

```
SYS_CONTEXT('USERENV','IDENTIFICATION_TYPE')
------------------------------------------------------------------------
GLOBAL SHARED
```

- LDAPサーバー・タイプを確認します。

```
SELECT SYS_CONTEXT('USERENV', 'LDAP_SERVER_TYPE') FROM DUAL;
```

次のような出力結果が表示されます。この場合、LDAPサーバー・タイプはActive Directoryです。

```
SYS_CONTEXT('USERENV','LDAP_SERVER_TYPE')
------------------------------------------------------------------------
AD
```

関連トピック

- パスワード認証を使用したOracle Databaseへのログイン

親トピック: 集中管理ユーザーの認可の構成

6.5 集中管理ユーザーのトラブルシューティング

Oracleには、Microsoft Active Directoryユーザーがデータベースにログインしようとしたときに発生する可能性のある一般的なエラーのトラブルシューティングに役立つエラー・メッセージが用意されています。

- ORA-28276接続エラー
  `orclCommonAttribute`属性が正しく設定されていない場合、`ORA-28276: ORACLE`パスワード属性が無効ですというエラーが発生することがあります。
- ORA-01017接続エラー
  Oracle DatabaseおよびMicrosoft Active Directoryでの特殊文字の許容方法の違いにより、「`ORA-01017: ユーザー名/パスワードが無効です。ログオンは拒否されました`」エラーが生成されます。
- ORA-28274接続エラー
  Active Directoryスキーマまたはサービス・ディレクトリに問題が発生し、`ORA-28274: ユーザー・ニックネームに対応するORACLE`パスワード属性が存在しませんというエラーが生成されます。
- ORA-28300接続エラー
  Oracleサービス・ディレクトリに権限の問題が発生し、`ORA-28030: LDAPディレクトリ・サービスのユーザー・エントリの読取り権限がありません`というエラーが生成されます。
- トレース・ファイルによるCMU接続エラーの診断
  トレース設定`gdsi`は、集中管理ユーザー(CMU)接続エラーを追跡します。

親トピック: Microsoft Active Directoryによる集中管理ユーザーの構成

6.5.1 ORA-28276接続エラー

`orclCommonAttribute`属性が正しく設定されていない場合、`ORA-28276: ORACLE`パスワード属性が無効ですというエラーが発生することがあります。

たとえば:

```
SQL> connect "myad¥dev"@orcl_db
Enter password: password
ERROR:
ORA-28276: Invalid ORACLE password attribute.
```

このエラーはorclCommonAttribute属性によってユーザー・パスワードが正しく移入されなかった場合に発生します。たとえば:

```
$ ldapsearch -h <AD_Server> -p 389 -D
"cn=oracleservice,cn=users,dc=myad,dc=example,dc=com" -w **** -U 2 -W
"file:wallet_path"
-P password -b "dc=myad,dc=example,dc=com" -s sub "(sAMAccountName=def*)"
dn orclCommonAttributeCN=def,CN=Users,DC=myad,DC=example,DC=com
orclCommonAttribute=
```

この問題を解決するには:

1. opwdintg.exeを実行し、Active DirectoryのドメインにすべてのWindowsドメイン・コントローラのパスワード・フィルタをインストールします。
2. 各Windowsドメイン・コントローラ・サーバーを再起動します。各Windowsドメイン・コントローラはパスワード・フィルタのインストール後に再起動する必要があります。そうしない場合、パスワード・フィルタがWindowsドメイン・コントローラで機能しません。
3. Active Directoryユーザーを適切なORA_VFRグループに割り当てます。
4. Active Directoryでユーザー・パスワードをリセットします。
5. ldapsearchを実行して、パスワードが生成されていることを確認します。

親トピック: 集中管理ユーザーのトラブルシューティング

6.5.2 ORA-01017接続エラー

Oracle DatabaseおよびMicrosoft Active Directoryでの特殊文字の許容方法の違いにより、「ORA-01017: ユーザー名/パスワードが無効です。ログオンは拒否されました」エラーが生成されます。

集中管理ユーザー(CMU)が作成するユーザー名とパスワードは、Oracle Databaseのユーザー名とパスワードのルールとは異なる作成ルールに従っています。ORA-01017エラーの問題を修正するには、Active Directoryユーザーのユーザー名とパスワードを二重引用符で囲みます。たとえば、ユーザー名がpeter fitch、パスワードがILoveMySalads@_home!であり、Oracleサービス・ユーザーとドメインが同じActive Directoryユーザーの場合、次のログインが機能します。

```
CONNECT "peter fitch"/"ILoveMySalads@_home!"@orcl
```

Active DirectoryユーザーのドメインがOracleサービス・ユーザーとは異なる場合、Windowsドメイン(この場合はEXAMPLE)をユーザー名に含める必要があります。

```
CONNECT "EXAMPLE¥peter fitch"/"ILoveMySalads@_home!"@orcl
CONNECT "EXAMPLE¥peter fitch"@orcl
Enter password: password
```

Enter passwordプロンプトに入力するパスワードが22文字の場合、ILoveMySalads@_home!パスワードは20文字と2つの二重引用符の分の2文字になります。

親トピック: 集中管理ユーザーのトラブルシューティング

6.5.3 ORA-28274接続エラー

Active DirectoryスキーマまたはOracleサービス・ディレクトリに問題が発生し、ORA-28274: ユーザー・ニックネー

ムに対応するORACLEパスワード属性が存在しませんというエラーが生成されます。

Active Directoryスキーマが拡張されていないか、正しく移入されていません。かわりに、Oracleサービス・ディレクトリ・ユーザーにOracleデータベースにログインしようとするユーザーのorclCommonAttribute属性へのアクセスに必要な権限がありません。

この問題を解決するには:

- 解決策1:
    1. opwdintg.exeを実行し、Active DirectoryのドメインにすべてのWindowsドメイン・コントローラのパスワード・フィルタをインストールします。
    2. 各Windowsドメイン・コントローラ・サーバーを再起動します。各Windowsドメイン・コントローラはパスワード・フィルタのインストール後に再起動する必要があります。そうしない場合、パスワード・フィルタがWindowsドメイン・コントローラで機能しません。
    3. Active Directoryユーザーを適切なORA_VFRグループに割り当てます。
    4. Active Directoryでユーザー・パスワードをリセットします。
    5. ldapsearchを実行して、パスワードが生成されていることを確認します。
- 解決策2:
    1. Oracleサービス・ディレクトリ・ユーザー・アカウントに、データベースにアクセスしようとするActive Directoryユーザーのプロパティへのアクセス権である、Read PropertiesおよびWrite lockoutTimeを付与します。
    2. Active DirectoryユーザーのorclCommonAttributeにControl Accessの権限を設定します。

関連トピック

- ステップ1: Microsoft Active DirectoryでのOracleサービス・ディレクトリ・ユーザー・アカウントの作成および権限の付与

親トピック: 集中管理ユーザーのトラブルシューティング

6.5.4 ORA-28300接続エラー

Oracleサービス・ディレクトリに権限の問題が発生し、ORA-28030: LDAPディレクトリ・サービスのユーザー・エントリの読取り権限がありませんというエラーが生成されます。

このエラーはCMUトレースを使用して追跡できます。たとえば:

```
2023-03-27 19:51:55.0 - KZLG_ERR: failed to modify user status Insufficient access
2023-03-27 17:57:27.0 - KZLG_ERR: LDAPERR=50, OER=28300
```

この問題(および権限)を修正するには:

1. Oracleサービス・ディレクトリ・ユーザー・アカウントに、データベースにアクセスしようとするActive Directoryユーザーのプロパティへのアクセス権である、Read PropertiesおよびWrite lockoutTimeを付与します。
2. Active DirectoryユーザーのorclCommonAttributeにControl Accessの権限を設定します。

関連トピック

- ステップ1: Microsoft Active DirectoryでのOracleサービス・ディレクトリ・ユーザー・アカウントの作成および権限の付与
- トレース・ファイルを使用したCMU接続エラーの診断

親トピック: 集中管理ユーザーのトラブルシューティング

6.5.5 トレース・ファイルを使用したCMU接続エラーの診断

トレース設定gdsiは、集中管理ユーザー(CMU)の接続エラーを追跡します。

ALTER SYSTEM権限およびSYSDBA管理者権限を持つユーザーの場合、このトレース・イベントを次のように有効にできます。

```
ALTER SYSTEM SET EVENTS='TRACE[GDSI] DISK LOW';
```

Active Directoryユーザーがログインを試行し、ログインに失敗した場合は、トレース・ファイルを含むディレクトリに移動し、接続エラーについてこれらのファイルをgrepします。

```
grep -i kzlg *.trc
```

その後、詳細情報を含むトレース・ファイルを収集し、確認できます。

トレースを無効にするには、次のコマンドを入力します。

```
ALTER SYSTEM SET EVENTS='TRACE[GDSI] OFF';
```

親トピック: 集中管理ユーザーのトラブルシューティング

6.6 Microsoft Active Directoryのアカウント・ポリシーとのOracle Databaseの統合

Oracle DatabaseとMicrosoft Active Directoryの統合の一部として、Oracle Databaseでは、Active Directoryユーザーが Oracle Databaseにログインするときに Active Directoryアカウント・ポリシーが適用されます。

Active Directoryアカウント・ポリシー設定は、パスワード・ポリシー、アカウント・ロックアウト・ポリシーおよびKerberosポリシーを対象とします。Oracle Databaseでは、Active Directoryの集中管理ユーザーのすべてのアカウント・ポリシーが適用されます。たとえば、Oracleでは、「パスワードの有効期限切れ」、「パスワードの変更が必要」、「ロック・アウトされたアカウント」または「無効なアカウント」などのアカウント・ステータスを持つActive Directoryユーザーはデータベースにログインできなくなります。Kerberos認証を使用している場合、Oracleでは、Kerberosチケットが期限切れになっているActive Directoryユーザーは、データベースにログインできなくなります。パスワード認証を使用している場合、Active Directoryユーザー・アカウントは、不正なパスワードを使用してOracleデータベースにログインしようとして指定した回数連続して失敗すると、指定した期間Active Directoryでロックアウトされます。アカウント・ロックアウト・ポリシーを適用すると、Active Directoryユーザー・アカウントに対するパスワード推測攻撃が効果的に回避されます。

ノート:

Oracle では、Active Directory のデフォルト・ドメイン・ポリシーのみをサポートしており、ファイングレイン・パスワード・ポリシーはサポートしていません。たとえば、デフォルトのドメイン・ポリシーでパスワードの有効期限が設定されているが、ファイングレイン・パスワード・ポリシーの有効期限が短い場合、CMU を Active Directory とともに使用して Oracle データベースにアクセスする Active Directory ユーザーには、デフォルトのドメイン・ポリシーでのパスワードの有効期限のみが適用されます。

親トピック: Microsoft Active Directoryによる集中管理ユーザーの構成

6.7 Oracle Autonomous Databaseを使用した集中管理ユーザーの構成

集中管理ユーザー(CMU)をOracle Autonomous Databaseにデプロイできます。

CMUをOracle Autonomous Databaseにデプロイする手順は、『Oracle Autonomous Database on Shared Exadata Infrastructureの使用』の「Microsoft Active DirectoryとAutonomous Databaseとの併用」を参照してください。

親トピック: Microsoft Active Directoryによる集中管理ユーザーの構成

6.8 集中管理ユーザーのトラブルシューティング

Oracleには、Microsoft Active Directoryユーザーがデータベースにログインしようとしたときに発生する可能性のある一般的なエラーのトラブルシューティングに役立つエラー・メッセージが用意されています。

- ORA-28276接続エラー
  `orclCommonAttribute`属性が正しく設定されていない場合、`ORA-28276: ORACLE`パスワード属性が無効ですというエラーが発生することがあります。
- ORA-01017接続エラー
  Oracle DatabaseおよびMicrosoft Active Directoryでの特殊文字の許容方法の違いにより、「`ORA-01017: ユーザー名/パスワードが無効です。ログオンは拒否されました`」エラーが生成されます。
- ORA-28274接続エラー
  Active DirectoryスキーマまたはOracleサービス・ディレクトリに問題が発生し、`ORA-28274: ユーザー・ニックネームに対応するORACLE`パスワード属性が存在しませんというエラーが生成されます。
- ORA-28300接続エラー
  Oracleサービス・ディレクトリに権限の問題が発生し、`ORA-28030: LDAPディレクトリ・サービスのユーザー・エントリの読取り権限がありません`というエラーが生成されます。
- トレース・ファイルによるCMU接続エラーの診断
  トレース設定`gdsi`は、集中管理ユーザー(CMU)接続エラーを追跡します。

親トピック: Microsoft Active Directoryによる集中管理ユーザーの構成

6.8.1 ORA-28276接続エラー

`orclCommonAttribute`属性が正しく設定されていない場合、`ORA-28276: ORACLE`パスワード属性が無効ですというエラーが発生することがあります。

たとえば:

```
SQL> connect "myad¥dev"@orcl_db
Enter password: password
ERROR:
ORA-28276: Invalid ORACLE password attribute.
```

このエラーは`orclCommonAttribute`属性によってユーザー・パスワードが正しく移入されなかった場合に発生します。たとえば:

```
$ ldapsearch -h <AD_Server> -p 389 -D
"cn=oracleservice,cn=users,dc=myad,dc=example,dc=com" -w **** -U 2 -W
"file:wallet_path"
-P password -b "dc=myad,dc=example,dc=com" -s sub "(sAMAccountName=def*)"
```

```
dn orclCommonAttributeCN=def,CN=Users,DC=myad,DC=example,DC=com
orclCommonAttribute=
```

この問題を解決するには:

1. opwdintg.exeを実行し、Active DirectoryのドメインにすべてのWindowsドメイン・コントローラのパスワード・フィルタをインストールします。
2. 各Windowsドメイン・コントローラ・サーバーを再起動します。各Windowsドメイン・コントローラはパスワード・フィルタのインストール後に再起動する必要があります。そうしない場合、パスワード・フィルタがWindowsドメイン・コントローラで機能しません。
3. Active Directoryユーザーを適切なORA_VFRグループに割り当てます。
4. Active Directoryでユーザー・パスワードをリセットします。
5. ldapsearchを実行して、パスワードが生成されていることを確認します。

親トピック: 集中管理ユーザーのトラブルシューティング

6.8.2 ORA-01017接続エラー

Oracle DatabaseおよびMicrosoft Active Directoryでの特殊文字の許容方法の違いにより、「ORA-01017: ユーザー名/パスワードが無効です。ログオンは拒否されました」エラーが生成されます。

集中管理ユーザー(CMU)が作成するユーザー名とパスワードは、Oracle Databaseのユーザー名とパスワードのルールとは異なる作成ルールに従っています。ORA-01017エラーの問題を修正するには、Active Directoryユーザーのユーザー名とパスワードを二重引用符で囲みます。たとえば、ユーザー名がpeter fitch、パスワードがILoveMySalads@_home!であり、Oracleサービス・ユーザーとドメインが同じActive Directoryユーザーの場合、次のログインが機能します。

```
CONNECT "peter fitch"/"ILoveMySalads@_home!"@orcl
```

Active DirectoryユーザーのドメインがOracleサービス・ユーザーとは異なる場合、Windowsドメイン(この場合はEXAMPLE)をユーザー名に含める必要があります。

```
CONNECT "EXAMPLE¥peter fitch"/"ILoveMySalads@_home!"@orcl
CONNECT "EXAMPLE¥peter fitch"@orcl
Enter password: password
```

Enter passwordプロンプトに入力するパスワードが22文字の場合、ILoveMySalads@_home!パスワードは20文字と2つの二重引用符の分の2文字になります。

親トピック: 集中管理ユーザーのトラブルシューティング

6.8.3 ORA-28274接続エラー

Active DirectoryスキーマまたはOracleサービス・ディレクトリに問題が発生し、ORA-28274: ユーザー・ニックネームに対応するORACLEパスワード属性が存在しませんというエラーが生成されます。

Active Directoryスキーマが拡張されていないか、正しく移入されていません。かわりに、Oracleサービス・ディレクトリ・ユーザーにOracleデータベースにログインしようとするユーザーのorclCommonAttribute属性へのアクセスに必要な権限がありません。

この問題を解決するには:

- 解決策1:
    1. opwdintg.exeを実行し、Active DirectoryのドメインにすべてのWindowsドメイン・コントローラのパス

ワード・フィルタをインストールします。
2. 各Windowsドメイン・コントローラ・サーバーを再起動します。各Windowsドメイン・コントローラはパスワード・フィルタのインストール後に再起動する必要があります。そうしない場合、パスワード・フィルタがWindowsドメイン・コントローラで機能しません。
3. Active Directoryユーザーを適切なORA_VFRグループに割り当てます。
4. Active Directoryでユーザー・パスワードをリセットします。
5. ldapsearchを実行して、パスワードが生成されていることを確認します。

- 解決策2:
  1. Oracleサービス・ディレクトリ・ユーザー・アカウントに、データベースにアクセスしようとするActive Directoryユーザーのプロパティへのアクセス権である、Read PropertiesおよびWrite lockoutTimeを付与します。
  2. Active DirectoryユーザーのorclCommonAttributeにControl Accessの権限を設定します。

関連トピック

- ステップ1: Microsoft Active DirectoryでのOracleサービス・ディレクトリ・ユーザー・アカウントの作成および権限の付与

親トピック: 集中管理ユーザーのトラブルシューティング

6.8.4 ORA-28300接続エラー

Oracleサービス・ディレクトリに権限の問題が発生し、ORA-28030: LDAPディレクトリ・サービスのユーザー・エントリの読取り権限がありませんというエラーが生成されます。

このエラーはCMUトレースを使用して追跡できます。たとえば:

```
2023-03-27 19:51:55.0 - KZLG_ERR: failed to modify user status Insufficient access
2023-03-27 17:57:27.0 - KZLG_ERR: LDAPERR=50, OER=28300
```

この問題(および権限)を修正するには:

1. Oracleサービス・ディレクトリ・ユーザー・アカウントに、データベースにアクセスしようとするActive Directoryユーザーのプロパティへのアクセス権である、Read PropertiesおよびWrite lockoutTimeを付与します。
2. Active DirectoryユーザーのorclCommonAttributeにControl Accessの権限を設定します。

関連トピック

- ステップ1: Microsoft Active DirectoryでのOracleサービス・ディレクトリ・ユーザー・アカウントの作成および権限の付与
- トレース・ファイルを使用したCMU接続エラーの診断

親トピック: 集中管理ユーザーのトラブルシューティング

6.8.5 トレース・ファイルを使用したCMU接続エラーの診断

トレース設定gdsiは、集中管理ユーザー(CMU)の接続エラーを追跡します。

ALTER SYSTEM権限およびSYSDBA管理者権限を持つユーザーの場合、このトレース・イベントを次のように有効にできます。

```
ALTER SYSTEM SET EVENTS='TRACE[GDSI] DISK LOW';
```

Active Directoryユーザーがログインを試行し、ログインに失敗した場合は、トレース・ファイルを含むディレクトリに移動し、接

続エラーについてこれらのファイルをgrepします。

```
grep -i kzlg *.trc
```

その後、詳細情報を含むトレース・ファイルを収集し、確認できます。

トレースを無効にするには、次のコマンドを入力します。

```
ALTER SYSTEM SET EVENTS='TRACE[GDSI] OFF';
```

親トピック: 集中管理ユーザーのトラブルシューティング

7 Oracle DBaaSデータベースに対するIAMユーザーの認証と認可

Identity and Access Management (IAM)ユーザーは、Oracle Database as a service (Oracle DBaaS)インスタンスに接続するように構成できます。

- Oracle DBaaSに対するIAMユーザーの認証と認可の概要
  Oracle DBaaSインスタンスに対するIAMユーザーの認証および認可を開始する前に、全体的なプロセスについて理解する必要があります。
- IAM用のOracle DBaaSの構成
  IAMと連携するようにOracle DBaaSを構成するには、Oracle DBaaSデータベースの管理者が、最初にIAM統合を有効にし、次にOracle DBaaSのIAMユーザーおよびロールを認可する必要があります。
- Oracle DBaaS用のIAMの構成
  Oracle DBaaSインスタンスと連携するようにIAMを構成するには、IAM管理者がIAMポリシーを作成し、ユーザーにIAMのデータベース・パスワードを作成させる必要があります。
- インスタンス・プリンシパルまたはリソース・プリンシパルを使用したデータベースへのアクセス
  Oracle Cloud Infrastructure (OCI)アプリケーションまたは関数は、それ固有のインスタンス・プリンシパルまたはリソース・プリンシパルを使用してデータベース・インスタンスに接続できます。
- データベース・クライアント接続の構成
  IAMクライアント接続の構成によって、Oracle DBaaSインスタンスへのIAMユーザーの認証が制御されます。
- Oracle DBaaSとIAMの統合でのデータベース・リンク
  IAM資格証明を使用してOracle DBaaSデータベースにアクセスする際のデータベース・リンクの使用がサポートされています。
- IAM接続のトラブルシューティング
  「ORA-01017: ユーザー名/パスワードが無効です。ログオンは拒否されました」エラーは、Identity and Access Management (IAM)とのOracle DBaaS統合全体のいくつかの異なる問題によって発生する可能性があります。

親トピック: ユーザー認証および認可の管理

7.1 Oracle DBaaSに対するIAMユーザーの認証と認可の概要

Oracle DBaaSインスタンスに対するIAMユーザーの認証および認可を開始する前に、全体的なプロセスについて理解する必要があります。

- Oracle DBaaSに対するIAMユーザーの認証と認可について
  Oracle DBaaSインスタンスのユーザーは、Oracle Cloud Infrastructure (OCI) Identity and Access Management (IAM)で集中管理できます。
- Oracle DBaaSとIAMの統合のアーキテクチャ
  Oracle DBaaSインスタンスとIAMの統合のアーキテクチャは、DBaaSインスタンスに対する認証または接続に、IAMユーザーがOracle Cloud Infrastructure (OCI) IAMデータベース・パスワード・ベリファイアを使用しているか、OCI IAMトークンを使用しているかによって異なります。
- Oracle DBaaSとマップするIAMユーザーおよびグループ
  IAMユーザーは、データベース・スキーマのIAMユーザーへの排他的マッピングと、ユーザーがメンバーとして所属する

IAMグループへのデータベース共有スキーマのマッピングのいずれかで、スキーマにマップされる必要があります。

親トピック: Oracle DBaaSデータベースに対するIAMユーザーの認証と認可

7.1.1 Oracle DBaaSに対するIAMユーザーの認証と認可について

Oracle DBaaSインスタンスのユーザーは、Oracle Cloud Infrastructure (OCI) Identity and Access Management (IAM)で集中管理できます。

この統合は、次のOracle Database環境で実行できます。

- Oracle Autonomous Database on Dedicated Exadata Infrastructure
- Oracle Autonomous Database Serverless
- Oracle Base Database Service
- Oracle Exadata Database Service on Dedicated Infrastructure

IAMを構成する手順では、これらの環境を網羅するために「Oracle DBaaS」という用語を使用します。

ノート:

Oracle Database は、アイデンティティ・ドメインを含む Oracle Cloud Infrastructure (OCI) IAM と、アイデンティティ・ドメインを含まないレガシーIAM に対する Oracle DBaaS 統合をサポートしています。アイデンティティ・ドメインを含む IAM は、2021 年 11 月 8 日以降に作成された新しい OCI テナンシで導入されました。Autonomous Database Serverless は、デフォルトおよびデフォルト以外のアイデンティティ・ドメイン内のユーザーとグループをサポートしています。他の DBaaS プラットフォームは、デフォルトのアイデンティティ・ドメイン内のユーザーおよびグループのみをサポートします。

Oracle Database管理者は、OCI IAM管理者と連携して、Oracle DBaaSインスタンスに接続する必要があるOCI IAMユーザーの認証および認可を管理します。IAMユーザーが接続できるOracle DBaaSインスタンスのタイプは、Oracle Autonomous Database Serverless、Oracle Autonomous Database on Dedicated Exadata InfrastructureおよびOracle Base Database Serviceです。

このタイプの接続により、IAMユーザーはOracle DBaaSにアクセスできます。これらのユーザーは通常、ユーザー名とパスワードを使用してログインします(SQL*Plusなどを使用)。ユーザーは、DBaaSインスタンスにアクセスするときに、トークンを使用するIAMシングル・サインオン(SSO)資格証明を使用してログインすることもできます。IAMパスワード認証を使用するか、IAM SSOトークン認証を使用するかの選択は、ユースケースおよびユーザー・プリファレンスによって異なります。

既存のサポートされているデータベース・クライアントを使用するレガシー・アプリケーションは、IAMユーザー名とパスワードの使用にシームレスに移行できます。また、IAMデータベースの段階的なパスワード・ロールオーバー機能を使用して、IAMに2つ目のデータベース・パスワードを設定し、停止時間なしでアプリケーション・パスワードを更新することもできます。

IAMトークンをサポートするように更新されたツールおよびアプリケーションにより、IAMでユーザーを直接認証し、データベース・アクセス・トークンをDBaaSインスタンスに渡すことができます。SQL*Plusなどの既存のデータベース・ツールでは、IAMデータベース・パスワードを使用し、既存のパスワード・ログイン・プロトコルを使用してデータベースで直接認証でき、データベース・クライアントでは、IAMユーザー名とIAMデータベース・パスワードを使用してOCI IAMからデータベース・トークン(`db-token`)をリクエストし、`db-token`をIAMユーザー・アクセスのためにデータベースに送信できます。データベース・クライアントは、IAMユーザー名およびIAMデータベース・パスワードと引き換えに、`db-token`をリクエストできます。その他のすべてのIAM資格証明(`API-key`、インスタンス・プリンシパル、リソース・プリンシパル、セキュリティ・トークン、委任トークン)では、OCI CLIなどのアプリケーションまたはヘルパー・クライアントから`db-token`をリクエストする必要があります。データベース・アクセス・トークン(`db-token`)は

範囲が指定された所有証明(POP)トークンで、公開キーが付属しています。db-tokenがデータベースに送信される前に、データベース・クライアントは、トークンの公開キーに関連付けられている秘密キーを使用してdb-tokenに署名します。これにより、トークンの送信者がトークンの正しい所有者であることが「証明」されます。db-tokenを使用できる範囲を削減するために、db-tokenのリクエストの一部としてオプションで範囲を含めることができます。db-tokenのデフォルト範囲はテナンシ全体ですが、コンパートメントおよび個々のデータベースを範囲として定義することもできます。詳細は、OCI CLIコマンド・リファレンスのgetの説明を参照してください。

IAMユーザーおよびOCIアプリケーションは、次のいずれかの方法を使用して、IAMからデータベース・トークンをリクエストできます。

- 既存の有効なセキュリティ(セッション)トークンの使用
- IAMで認識されるAPIキーの使用
- OCIクラウド・シェル内での委任トークンの使用
- OCIコンピュート・インスタンス上のアプリケーションに対するOCIインスタンス・プリンシパルの使用
- リソース・プリンシパルを持つアプリケーションに対するOCIリソース・プリンシパルの使用
- IAMユーザー名およびIAMデータベース・パスワードの使用(データベース・クライアントのみがリクエスト可能)

IAMユーザーがOracle DBaaSインスタンスに接続できるようにするための一般的なプロセスは次のとおりです。

1. IAM管理者は、IAMユーザー・アカウントおよびグループを作成および管理し、タスクに基づいてIAMユーザーを適切なIAMグループに追加します。
2. Oracle DBaaSインスタンスでは、データベース管理者がOracle DBaaSとIAMエンドポイント間の接続を有効にします。

   データベースがAutonomous Database on Dedicated Exadata Infrastructureの場合は、新しいPDBのIAM接続が自動的に有効になります。詳細は、Oracle DBaaSのドキュメントを参照してください。

3. Oracle DBaaSサーバーで、データベース管理者が、次のタイプのマッピングを実行してIAMユーザーの認可を有効にします。
   - 共有Oracle Databaseグローバル・ユーザー・アカウントへのIAMグループのマッピング
   - Oracle Databaseグローバル・ロールへのIAMグループのマッピング
   - Oracle Databaseグローバル・ユーザーへのIAMユーザーの排他的マッピング

   IAMユーザーは、1つのスキーマにマップする必要があります(排他的にマップするか、共有スキーマにマップします)。ユーザーはオプションで、1つ以上のグローバル・ロールにマップされるIAMグループのメンバーにすることもできます。

4. 次のユースケースは、集中管理されたIAM認証および認可を使用してOracle DBaaSに接続する、いくつかの一般的なシナリオです。
   - SQL*Plusの使用による、IAMユーザー名およびIAMデータベース・パスワードを使用したOracle DBaaSへの接続。
   - SQL*Plusの使用による、IAM SSOトークンを使用した接続。
   - SQLclの使用による、IAMパスワードまたはIAMトークンを使用したOracle DBaaSへの接続。
   - Oracle Cloud Infrastructure (OCI) Cloud Shell内のSQL*Plusの使用による、IAMパスワードまたはIAM SSOトークンを使用したOracle DBaaSへの接続。IAMを使用した認証と認可は、ローカル・データベース・ユーザー・アカウント(非グローバル)に対する認証とは対照的に、時間がかかります。

関連トピック

- Oracle DBaaSの外部認証の有効化
- Oracle Autonomous Database Serverlessの使用

- Oracle Autonomous Database Serverlessの使用
- Oracle Exadata Database Service on Dedicated InfrastructureへのIdentity and Access Management (IAM)ユーザーの接続

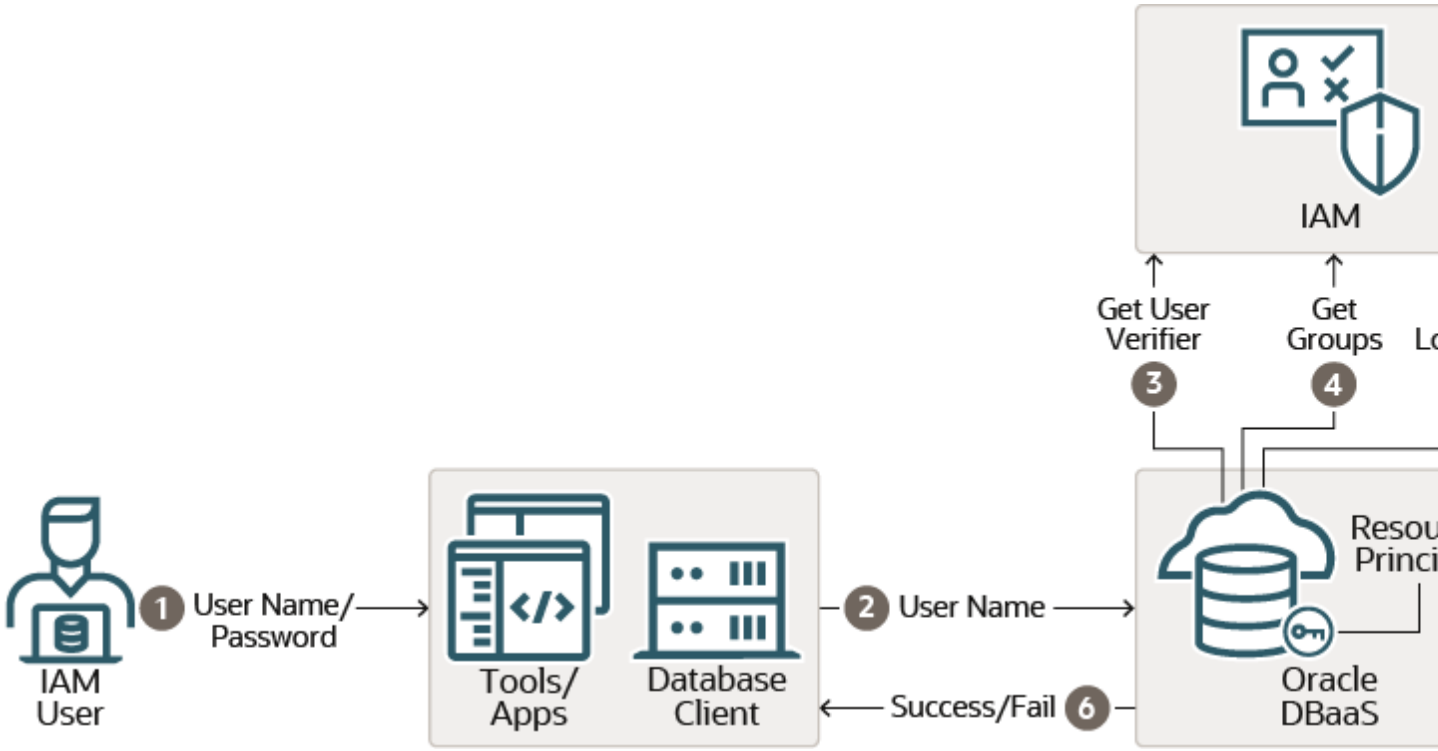
親トピック: Oracle DBaaSに対するIAMユーザーの認証と認可の概要

7.1.2 Oracle DBaaSとIAMの統合のアーキテクチャ

Oracle DBaaSインスタンスとIAMの統合のアーキテクチャは、DBaaSインスタンスに対する認証または接続に、IAMユーザーがOracle Cloud Infrastructure (OCI) IAMデータベース・パスワード・ベリファイアを使用しているか、OCI IAMトークンを使用しているかによって異なります。

次の図は、Oracle Cloud Infrastructure (OCI) IAMデータベース・パスワード・ベリファイアを使用してOracle DBaaSで認証する仕組みを示しています。

図7-1 OCI IAMデータベース・パスワード・ベリファイアを使用してOracle DBaaSへの認証を行うIAMユーザー



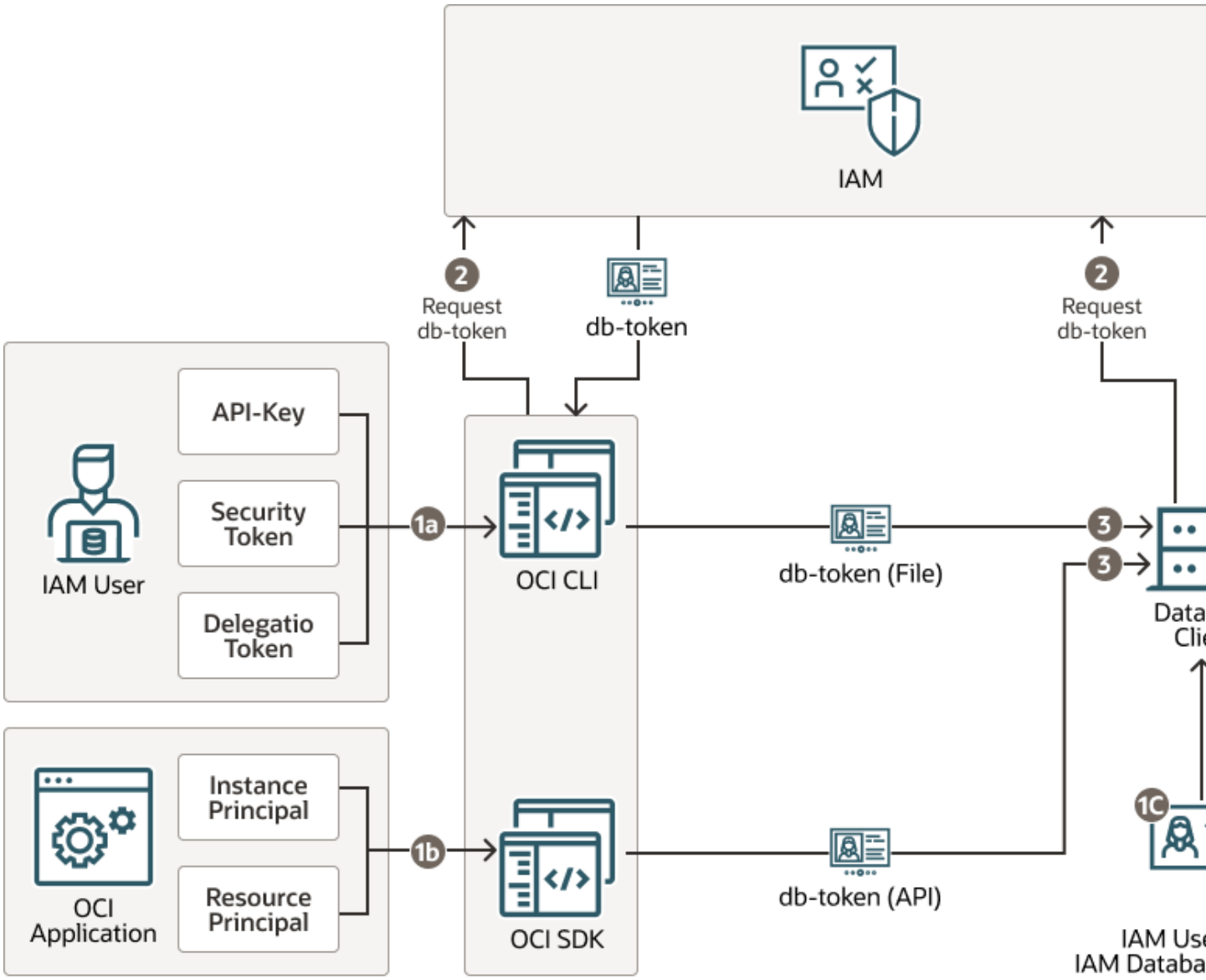
1. IAMユーザーが、Oracle Databaseクライアントに関連付けられているツールまたはアプリケーション・クライアントにログインします。このユーザーは、IAMユーザー名およびIAMデータベース・パスワードを使用してログインし、これによって認証プロセスが開始されます。ユーザーは、Oracle Database release12.1.0.2以降の任意のデータベース・クライアントを使用できます。以前のバージョンのデータベース・クライアントは、12Cデータベース検証をサポートしていません。
2. IAMユーザーの接続リクエストが、データベース・クライアントを介して送信されます。
3. IAMユーザー名がOracle DBaaSインスタンスに送信されると、データベースが、ユーザーのOracle Cloud Infrastructure (OCI) IAMデータベース・パスワード・ベリファイアをIAMからリクエストします。(IAMユーザー・プロファイルに、IAMデータベース・パスワード・ベリファイアが格納されます。)このベリファイアはパスワードをハッシュしたバージョンであり、クリアテキストではありません。IAMからのパスワード・ベリファイアが、データベース・クライアントによって生成されたパスワード・ベリファイアと一致する場合、ユーザーは認証されます。Oracle DBaaSインスタンスは、リソース・プリンシパルを使用してIAMと通信します。リソース・プリンシパルは、IAMによって認識され、データベースがIAMと安全に通信するために使用するOracle DBaaSのアイデンティティです。
4. 認証が成功すると、Oracle DBaaSインスタンスがIAMユーザー・グループを取得します。そのIAMユーザーがOracle

Databaseスキーマにマップされ、かつOCIアカウントからロック・アウトされていない場合、IAMユーザーはデータベースに正常にアクセスできます。ユーザーには、ユーザーがメンバーであるグループにマップされているグローバル・ロールも付与されます。

5. Oracle Cloud Infrastructure (OCI)ログイン・カウンタが、OCIコンソール・パスワードとOCIデータベース・パスワードの両方のログインを追跡します。IAMデータベース・パスワードを使用したデータベースのログインが成功すると、このカウンタがリセットされます。
6. 前述のステップの結果に基づいて、IAMユーザー・データベースのアクセス試行が成功または失敗します。

次の図は、IAMユーザーまたはOracle Cloud Infrastructure (OCI)アプリケーションがOCI IAMトークンを使用してOracle DBaaSインスタンスにアクセスしたときに実行されるアクションの開始を示しています。

図7-2 OCI IAMトークンを使用してOracle DBaaSへの認証を行うIAMユーザーまたはOCIアプリケーション、パート1



1. データベースへのアクセスには、次のいずれかが必要です。
   - 1a: IAMユーザーの場合は、ユーザーがローカル・システムにAPI-keyを格納しているか、最近のOCIへのサインインからのセキュリティ・トークンを持っている必要があります。OCI CLIでは、API-key、セキュリティ・トークン、委任トークン、インスタンス・プリンシパルを使用できます。最新の有効なセキュリティ・トークンがない場合、ユーザーにOCI IAMでの認証を求めるプロンプトが表示される可能性があります。(使用可能なユーザー資格証明の詳細は、ユーザー資格証明を参照してください。)OCIクラウド・シェル環境では、委任トークンを使

用できます。

- 1b: OCIアプリケーションの場合は、インスタンス・プリンシパルまたはリソース・プリンシパルを持つようにアプリケーションが構成されている必要があります。OCI SDKでは、すべてのキー・タイプ(API-key、セキュリティ・トークン、委任トークン、インスタンス・プリンシパルおよびリソース・プリンシパル)を使用できます。
- 1c: IAMユーザー名とIAMデータベース・パスワードを使用してIAMからdb-tokenをリクエストするようにデータベース・クライアントを構成できます。データベース・クライアントのみがこのタイプのトークンを使用してデータベースにアクセスできます。データベース・クライアントは、他の資格証明を使用してdb-tokenをリクエストできません。

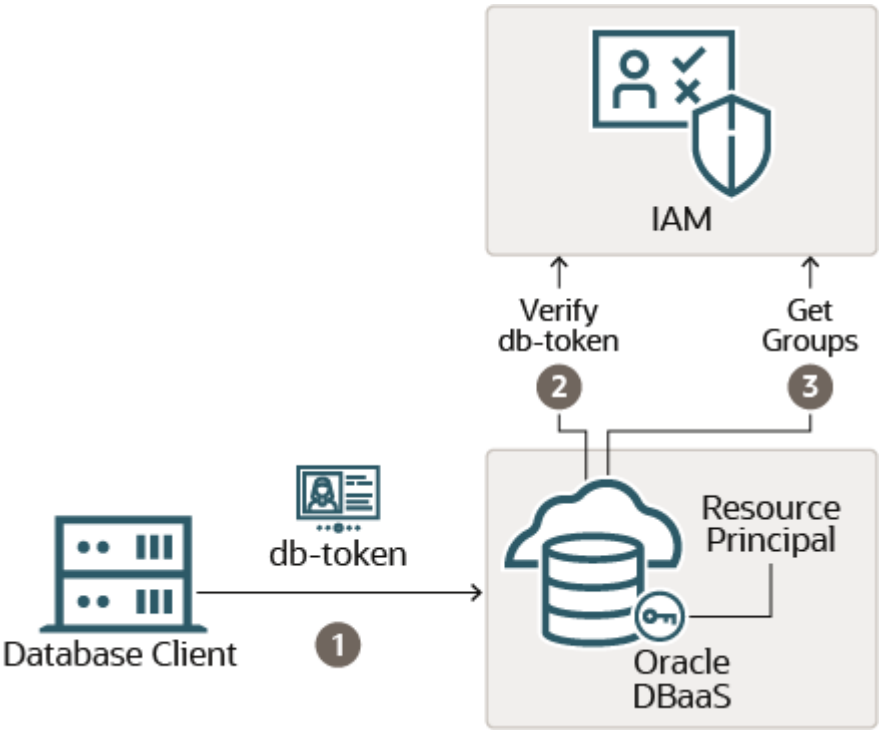
2. アプリケーション、OCI CLIまたはデータベース・クライアントは、いずれかのプリンシパル資格証明を使用して、db-tokenをリクエストするIAMへのコールを実行します。Oracle DBaaSへのアクセスに使用できるのは、db-tokenのみです。db-tokenのリクエストは、Oracle Cloud Infrastructure (OCI)パブリックSDKで、OCI IAMに接続するように記述されたアプリケーションによって実行できます。(ソフトウェア開発キットとコマンドライン・インタフェースを参照してください。)OCIパブリックSDKを使用してOCI IAMと直接接続するようにアプリケーションを変更できない場合は、OCIコマンドライン・インタフェース(OCI CLI)などのヘルパー・ツールを使用して、ユーザーのdb-tokenを取得できます。データベース・クライアントは、IAMユーザー名およびIAMデータベース・パスワードを使用してdb-tokenをリクエストするように構成することもできます。
3. IAMと連携するように更新されたアプリケーションまたはツールが、クライアントAPIを介してdb-tokenを属性としてデータベース・クライアントに直接渡すことができます。db-tokenを直接取得するようにアプリケーションを更新できない場合は、OCI CLIなどのヘルパー・ツールによって、ローカル・ディレクトリのデフォルトまたは指定した場所にdb-tokenを配置できます。接続文字列またはsqlnet.oraファイルのTOKEN_AUTH=OCI_TOKENの設定により、データベース・クライアントは、デフォルトまたは指定されたファイルの場所からdb-tokenを取得できます。ユーザーは、oci iam db-token getコマンドを実行し、ユーザー・アカウント資格証明を格納するプロファイルを指定することで、OCI CLIでトークンをリクエストできます。例:

```
oci iam db-token get --profile PeterFitch
```

db-tokenおよび対応する秘密キーのディレクトリの場所には、OCI CLIがファイルをその場所に書き込み、データベース・クライアントがこれらのファイルを取得するのに十分な権限のみ(プロセス・ユーザーによる読取りおよび書込みのみなど)が必要です。トークンとキーによってデータベースへのアクセスが許可されるため、これらはファイル・システム内で保護される必要があります。

次の図は、OCI IAMトークン認証プロセスの続きを示しています。

図7-3 OCI IAMトークンを使用してOracle DBaaSへの認証を行うIAMユーザーまたはOCIアプリケーション、パート2



1. db-tokenが署名されて、Oracle DBaaSに送信されます。データベース・クライアント/サーバー・リンクおよびDN一致でTLSを有効にする必要があります。(Autonomous Databaseウォレット・ファイルを使用してAutonomous Databaseインスタンスに接続する場合、TLSおよびDNSの一致はすでに設定されています。)DN一致は、JDBCドライバでデフォルトでオンになっていますが、OCI-Cデータベース・クライアント(およびインスタント・クライアント)用に構成される必要があります。データベース・クライアントがIAMユーザー名およびIAMデータベース・パスワードを使用して取得するdb-tokenには秘密キーが含まれず、データベース・クライアントによって署名されません。
2. 有効なコピーがまだローカルで使用できない場合、Oracle DBaaSインスタンスが、IAM公開キーをリクエストします。このキーは、db-tokenがIAMによって送信されたことを検証するために使用されます。Oracle DBaaSインスタンスは、リソース・プリンシパルを使用してIAMと通信します。
3. この認可ステップが正常に完了すると、Oracle DBaaSインスタンスが、IAMからIAMユーザーのグループをリクエストします。このアクションにより、ユーザーがグローバル・スキーマにマップされて、さらにそのユーザーが、メンバーとして所属するグローバル・ロールにマップされます。IAMユーザーがこれらのステップを正常に完了すると、ユーザーはOracle DBaaSインスタンスにアクセスできます。

IAM SSOトークンベースの認証では、最新のOracle Database 19c (19.16)クライアントをダウンロードする必要があります。

関連トピック

- Oracle Autonomous Database Serverlessの使用

親トピック: Oracle DBaaSに対するIAMユーザーの認証および認可の概要

7.1.3 Oracle DBaaSとマップするIAMユーザーおよびグループ

IAMユーザーは、IAMユーザーへのデータベース・スキーマの排他的マッピングと、ユーザーがメンバーとして所属するIAMグループへのデータベース共有スキーマのマッピングのいずれかで、スキーマにマップされる必要があります。

ログインおよび認可ステップを正常に完了するには、IAMユーザーをデータベース・スキーマにマップする必要があります。IAMユーザーが独自のスキーマ・オブジェクトを保持する必要がある場合は、IAMユーザーをデータベース・スキーマに直接マップできます(排他的マッピング)。より一般的には、IAMユーザーは、データベース・スキーマにマップされるIAMグループのメンバーとなります(共有スキーマ・マッピング)。共有スキーマ・マッピングを使用すると、複数のIAMユーザーが同じスキーマを共有できるため、新

規ユーザーが組織に加入するたびに新しいデータベース・スキーマを作成する必要はありません。この運用効率により、データベース管理者は、新規ユーザーの構成、権限とロールの更新、およびアカウントの削除を行わずに、データベース・アプリケーションのメンテナンス、パフォーマンスおよびチューニングのタスクに集中できます。

データベース・グループのデータベース管理者は、IAMグループのメンバーにできます(たとえば、販売アプリケーションの販売アプリケーション開発者は、`sales_app_dev_group`というIAMグループのメンバーです)。このシナリオでは、すべての関連データベースが共有スキーマを`sales_app_dev_group`グループにマップできます。データベース・グローバル・ロールはスキーマに付与できません。IAMグループにのみマップできます。複数のIAMユーザーが同じ共有スキーマにマップされている場合、グローバル・ロールでIAMユーザー権限を区別できます。

IAMユーザーは、そのIAMユーザーがOracle DBaaSインスタンスにアクセスできるように、データベース・スキーマまたは共有スキーマに排他的にマップされる必要があることに注意してください。

親トピック: Oracle DBaaSに対するIAMユーザーの認証および認可の概要

7.2 IAM用のOracle DBaaSの構成

IAMと連携するようにOracle DBaaSを構成するには、Oracle DBaaSデータベースの管理者が、最初にIAM統合を有効にし、次にOracle DBaaSのIAMユーザーおよびロールを認可する必要があります。

- Oracle DBaaSに対する外部認証の有効化
  Oracle DBaaSでIAM接続を有効化する方法は、使用しているOracle DBaaSのプラットフォームによって異なります。
- IAMユーザーおよびOracle Cloud Infrastructureアプリケーションの認可の構成
  Oracle DBaaSデータベースの管理者は、IAMユーザーおよびOracle Cloud Infrastructure (OCI)アプリケーションをOracle Databaseのグローバル・スキーマおよびグローバル・ロールにマップできます。
- IAMプロキシ認証の構成
  プロキシ認証により、IAMユーザーはアプリケーションのメンテナンスなどのタスクのためにデータベース・スキーマにプロキシできます。

親トピック: Oracle DBaaS Databaseに対するIAMユーザーの認証および認可

7.2.1 Oracle DBaaSに対する外部認証の有効化

Oracle DBaaSでIAM接続を有効化する方法は、使用しているOracle DBaaSのプラットフォームによって異なります。

- Oracle Autonomous Database on Dedicated Exadata Infrastructure: IAM接続は、このプラットフォームで機能するように自動的に構成されます。専用Exadataインフラストラクチャでの Oracle Autonomous Databaseの使用を参照してください
- Oracle Autonomous Database Serverless: このプラットフォームを使用するには、IAM接続を有効にする必要があります。「Oracle Autonomous Database Serverlessの使用」を参照してください。
- Oracle Base Database Service: ベース・データベース・サービスでのアイデンティティおよびアクセス管理認証の使用を参照してください。
- Oracle Exadata Database Service on Dedicated Infrastructure: Oracle Exadata Database Service on Dedicated InfrastructureへのIdentity and Access Management (IAM)ユーザーの接続を参照してください。

Oracle Autonomous Database Serverless以外のデータベース

1. 前提条件およびその他の必要な情報については、Oracle DBaaSプラットフォームのドキュメントを参照してください。
2. Oracle Autonomous Database以外のインスタンスの場合は、IDENTITY_PROVIDER_CONFIGパラメータを設定します。

```
ALTER SYSTEM SET IDENTITY_PROVIDER_TYPE=OCI_IAM SCOPE=BOTH;
```

IDENTITY_PROVIDER_CONFIGが別の値に設定されている場合は、次の文を実行します。

```
ALTER SYSTEM RESET IDENTITY_PROVIDER_CONFIG SCOPE=BOTH;
```

Microsoft Azureなどの別のアイデンティティ・プロバイダが使用されていたため、IDENTITY_PROVIDER_CONFIGパラメータが別の値に設定されている可能性があります。

親トピック: IAM用のOracle DBaaSの構成

7.2.2 IAMユーザーおよびOracle Cloud Infrastructureアプリケーションの認可の構成

Oracle DBaaSデータベースの管理者は、IAMユーザーおよびOracle Cloud Infrastructure (OCI)アプリケーションをOracle Databaseのグローバル・スキーマおよびグローバル・ロールにマップできます。

- IAMユーザーおよびOracle Cloud Infrastructureアプリケーションの認可の構成について
  Oracle DBaaSのデータベース・ユーザー(スキーマ)へのIAMユーザーおよびOracle Cloud Infrastructure (OCI)アプリケーションのマッピングを作成します。
- 共有Oracle Databaseグローバル・ユーザーへのIAMグループのマッピング
  IAMグループおよびIAM動的グループにマップされたOracle Databaseグローバル・ユーザーにより、IAMユーザーおよびOCIアプリケーションがログインするときに、スキーマおよびそのスキーマに付与された権限とロールがIAMユーザーおよびOCIアプリケーションに付与されます。
- Oracle Databaseグローバル・ロールへのIAMグループのマッピング
  IAMグループおよび動的グループにマップされたOracle Databaseグローバル・ロールは、メンバー・ユーザーおよびアプリケーションに、ログイン・スキーマを介して付与されている権限およびロールに加えて、さらに権限およびロールを付与します。
- Oracle Databaseグローバル・ユーザーへのIAMユーザーの排他的マッピング
  IAMユーザーをOracle Databaseグローバル・ユーザーに排他的にマッピングできます。
- IAMユーザー・マッピング定義の変更または移行
  ALTER USER文を使用して、IAMユーザーからデータベース・グローバル・ユーザーへのマッピングを更新できます。
- インスタンス・プリンシパルおよびリソース・プリンシパルのマッピング
  インスタンス・プリンシパルおよびリソース・プリンシパルは、アプリケーションでデータベース・トークンを取得して、Oracle DBaaSインスタンスへの接続を確立するために使用できます。
- IAMユーザーのログオン情報の確認
  Oracle DBaaSインスタンスのIAMユーザーを構成および認可した後、Oracleデータベース側で一連のSQL問合せを実行して、ユーザー・ログオン情報を確認できます。

親トピック: IAM用のOracle DBaaSの構成

7.2.2.1 IAMユーザーおよびOracle Cloud Infrastructureアプリケーションの認可の構成について

Oracle DBaaSのデータベース・ユーザー(スキーマ)へのIAMユーザーおよびOracle Cloud Infrastructure (OCI)アプリケーションのマッピングを作成します。

IAMデータベース・パスワード認証とIAMトークンベースの認証の使用には、認可に違いがあります。IAMデータベース・パスワード・ベリファイアによる認可は、IAMユーザーおよびグループへのデータベース・スキーマおよびグローバル・ロールのマッピングのみに基づきます。IAMトークンベースの認証では、IAMポリシーは、IAMユーザーがテナンシ・データベースにアクセスするための追加の認可です。IAMユーザーは、IAMポリシーを介して認可され、さらにデータベース・グローバル・スキーマへのマッピング(排他的または共有)を介して認可されている必要があります。

トークンとパスワード・ベリファイアの両方のデータベース・アクセスのために、Oracle DBaaSインスタンスへのIAMユーザーおよびOCIアプリケーションのマッピングを作成します。IAMユーザー・アカウント自体はIAMで管理されます。ユーザー・アカウントとユーザー・グループは、デフォルト・ドメインまたはカスタムのデフォルト以外のドメイン内に存在できます。

IAMユーザーがトークンを使用してOracle DBaaSインスタンスにアクセスすると、データベースで、ユーザーによるデータベースへのアクセスが許可されていることを確認するために、IAMポリシーに対して認可チェックが実行されます。IAMポリシーによって、IAMユーザーによるデータベースへのアクセスが許可されている場合、データベースが、ユーザー・グループのIAMを問い合せます。パスワード・ベリファイア認証を使用する場合、IAMユーザーが認証を正常に完了すると、データベースが、ユーザー・グループのIAMを問い合せます。データベースは、IAMエンドポイントを問い合せて、ユーザーがメンバーとして所属するグループを検索します。デプロイメントが共有スキーマを使用している場合、いずれかのIAMグループが共有データベース・スキーマにマップされ、IAMユーザーはそのデータベース・スキーマに割り当てられます。IAMユーザーは、データベース・スキーマに付与されたロールと権限を所持します。複数のIAMユーザーを同じ共有データベース・スキーマに割り当てることができるため、最小限のロールと権限のセットのみを共有スキーマに付与する必要があります。場合によっては、共有スキーマに権限およびロールを付与しないでください。ユーザーには、データベース・グローバル・ロールを介して適切なロールとスキーマのセットが割り当てられます。グローバル・ロールはIAMグループにマップされます。このように、同じデータベース共有スキーマにマップされている場合でも、異なるユーザーが異なるロールおよび権限を持つことができます。新しく採用されたユーザーは、共有スキーマにマップされたIAMグループに割り当てられたうえで、タスクを完了するために必要な追加のロールと権限を取得するために、グローバル・ロールにマップされた1つ以上の追加グループに割り当てられます。共有スキーマとグローバル・ロールを組み合せると、データベース操作に対する変更を最小限にして、集中的な認可管理が可能になります。データベースは最初に、適切なIAMグループにマップされている共有スキーマとグローバル・ロールのセットでプロビジョニングされる必要がありますが、ユーザー認可管理はIAM内で発生する可能性があります。

IAMユーザーが、データベース・スキーマへの排他的マッピングを介して、または共有データベース・スキーマにマップされている1つのIAMグループのメンバーとして、1つのスキーマにのみマップされるようにします。IAMユーザーに複数のスキーマがマップされている場合、データベースでは、共有スキーマへのグループ・マッピングよりも、排他的マッピングが優先されます。1人のユーザーに対して複数のグループがマップされている場合は、最も古いマッピングが選択されます。

グローバル・ロールを使用してユーザーに権限およびロールを付与する場合、セッションで有効にできるロールの最大数は150であることに注意してください。

IAMユーザーおよびグループを削除して、同じ名前を使用して再作成した場合、同じ名前を使用するデータベースからIAMへのマッピングは引き続き機能します。ただし、IAMユーザーを再作成するには、IAMユーザーが、IAMデータベース・パスワードの作成、API公開キーの再アップロード、OCI構成ファイルの更新の1つ以上を実行して、IAMを使用したデータベースの認証と認可についてIAMポリシーを再度調べる必要があります。IAMポリシーで、`database-connections`および`autonomous-database-family`リソース・タイプを使用または管理できるグループが指定されている場合、IAM認証および認可を許可するには、そのグループにユーザーを追加する必要があります。

トークンを使用してデータベースにアクセスするには、ユーザーがIAMポリシーおよびデータベース・マッピングによって認可される必要があります。IAMデータベース・パスワード・ベリファイアを使用してデータベースにアクセスするには、データベース・マッピングを介した認可が必要です。IAMユーザーのデータベース・スキーマ・マッピングが存在しない場合、有効なトークンまたはパスワードがある場合でも、IAMユーザーはデータベースにアクセスできません。

IAMユーザーは、付与されているロールに基づいて様々なタスクを実行するための認可を取得します。次の使用方法が可能で

す。

- 共有Oracle Databaseグローバル・ユーザーにマップされたIAMグループ: 共有データベース・グローバル・ユーザー・アカウントでは、共有スキーマへのIAMグループのマッピングを介して、IAMユーザーが共有データベース・スキーマ(ユーザー)に割り当てられます。グループのメンバーであるIAMユーザーはこの共有スキーマを介してデータベースに接続できます。共有スキーマを使用すると、IAMでユーザー認可を集中管理できます。
- Oracle Databaseグローバル・ロールにマップされたIAMグループ: 共有Oracle Databaseグローバル・ロールに付与された権限は、IAMグループに追加されたユーザーが使用できるようになります。
- Oracle Databaseグローバル・ユーザーに排他的にマップされたローカルのIAMユーザー: 排他的なグローバル・ユーザー・マッピングでは、専用データベース・ユーザーがローカルのIAMユーザーに排他的にマップされます。共有データベース・スキーマほど一般的ではありませんが、このユーザーは、ユーザーが独自のスキーマ・オブジェクトを必要とする場合のために作成されます。認可の管理を容易にするため、グローバル・ロールを介してこれらのユーザーにデータベース権限を付与することをお薦めします。これらのユーザーは、排他スキーマに対する直接権限およびロール付与を持つこともできます。

  アイデンティティ・ドメインを使用するIAMでは、デフォルト・ドメインおよびカスタム非デフォルト・ドメインでユーザーおよびグループがサポートされます。デフォルト・ドメインでユーザーとグループを指定する場合、ドメイン接頭辞は必要ありません。デフォルト以外のドメインでユーザーとグループを指定する場合は、ドメインに接頭辞を付ける必要があります。

親トピック: IAMユーザーおよびOracle Cloud Infrastructureアプリケーションの認可の構成

7.2.2.2 共有Oracle Databaseグローバル・ユーザーへのIAMグループのマッピング

IAMグループおよびIAM動的グループにマップされたOracle Databaseグローバル・ユーザーにより、IAMユーザーおよびOCIアプリケーションがログインするときに、スキーマおよびそのスキーマに付与された権限とロールがIAMユーザーおよびOCIアプリケーションに付与されます。

1. CREATE USERまたはALTER USERシステム権限を持つユーザーとして、Oracle DBaaSインスタンスにログインします。
2. IAMグループ名(動的グループでも可)を指定するIDENTIFIED GLOBALLY AS句を使用してCREATE USERまたはALTER USER文を実行します。
   たとえば、shared_sales_schemaという名前の新しいデータベース・グローバル・ユーザー・アカウントを作成し、それをWidgetSalesGroupという名前の既存のIAMグループにマップするには、次のようにします。

   ```
   CREATE USER shared_sales_schema IDENTIFIED GLOBALLY AS
   'IAM_GROUP_NAME=WidgetSalesGroup';
   ```

   次の例は、デフォルト以外のドメインでこれを実現する方法を示しています。

   ```
   CREATE USER shared_sales_schema IDENTIFIED GLOBALLY AS
   'IAM_GROUP_NAME=sales_domain/WidgetSalesGroup';
   ```

親トピック: IAMユーザーおよびOracle Cloud Infrastructureアプリケーションの認可の構成

7.2.2.3 Oracle Databaseグローバル・ロールへのIAMグループのマッピング

IAMグループおよび動的グループにマップされたOracle Databaseグローバル・ロールは、メンバー・ユーザーおよびアプリケーションに、ログイン・スキーマを介して付与されている権限およびロールに加え、さらに権限およびロールを付与します。

グローバル・ロールはデータベース・スキーマ(ユーザー)に付与できません。グループにマップして、データベースへのアクセス時にIAMユーザーに割り当てることのみ可能です。

1. CREATE ROLEまたはALTER ROLEシステム権限を付与されたユーザーとして、Oracle DBaaSインスタンスにログインします
2. IAMグループ名(動的グループでも可)を指定するIDENTIFIED GLOBALLY AS句を使用してCREATE ROLEまたはALTER ROLE文を実行します。
   たとえば、widget_mgr_roleという名前の新しいデータベース・グローバル・ロールを作成し、それをデフォルト・ドメインを使用してWidgetManagerGroupという名前の既存のIAMグループにマップするには、次のようにします。

```
CREATE ROLE widget_mgr_role IDENTIFIED GLOBALLY AS
'IAM_GROUP_NAME=WidgetManagerGroup';
```

   次の例は、デフォルト以外のドメインsales_domainを指定してロールを作成する方法を示しています:

```
CREATE ROLE widget_sales_role IDENTIFIED GLOBALLY AS
'IAM_GROUP_NAME=sales_domain/WidgetManagerGroup';
```

   sales_domainドメイン内のWidgetManagerGroupのすべてのメンバーは、データベースにログインするときにデータベース・グローバル・ロールwidget_sales_roleで認可されます。

親トピック: IAMユーザーおよびOracle Cloud Infrastructureアプリケーションの認可の構成

7.2.2.4 Oracle Databaseグローバル・ユーザーへのIAMユーザーの排他的マッピング

IAMユーザーをOracle Databaseグローバル・ユーザーに排他的にマップできます。

1. CREATE USERまたはALTER USERシステム権限を付与されたユーザーとして、Oracle DBaaSインスタンスにログインします。
2. IAMデータベース・ユーザー名を指定するIDENTIFIED GLOBALLY AS句を使用してCREATE USERまたはALTER USER文を実行します。
   デフォルトでは、IAMデータベース・ユーザー名はドメイン名を含むIAMユーザー名と同じです。データベースへの認証を容易にする一意のIAMデータベース・ユーザー名も作成できます。OCI IAMユーザー・プロファイルでは、データベースへの認証を容易にするために、一意のIAMデータベース・ユーザー名を作成できます。これは、IAMプロファイルでIAMデータベース・パスワードを作成および管理するときに設定できます。IAMデータベース・ユーザー名を追加または変更すると、IAMユーザーのスキーマへのマッピングが無効になるため、データベース・スキーマを新しいIAMデータベース・ユーザー名に再マップする必要があります。
   たとえば、peter_fitchという名前の新しいデータベース・グローバル・ユーザーを作成し、デフォルト・ドメインを使用して、IAMデータベース・ユーザー名peterfitchで指定した既存のIAMユーザーにこのユーザーをマップするには:

```
CREATE USER peter_fitch IDENTIFIED GLOBALLY AS
'IAM_PRINCIPAL_NAME=peterfitch';
```

   次の例は、デフォルト以外のドメインsales_domainを指定してユーザーを作成する方法を示しています:

```
CREATE USER peter_fitch2 IDENTIFIED GLOBALLY AS
'IAM_PRINCIPAL_NAME=sales_domain/peterfitch';
```

親トピック: IAMユーザーおよびOracle Cloud Infrastructureアプリケーションの認可の構成

7.2.2.5 IAMユーザー・マッピング定義の変更または移行

ALTER USER文を使用して、IAMユーザーからデータベース・グローバル・ユーザーへのマッピングを更新できます。

更新できるデータベース・スキーマは、IAMユーザーにマップされたデータベース・スキーマと、CREATE USER文の句IDENTIFIED BY password、IDENTIFIED EXTERNALLYまたはIDENTIFIED GLOBALLYのいずれかを使用して

アカウントが作成されているデータベース・スキーマです。これは、既存のスキーマを、IAMの使用に移行する場合に役立ちます。IAMユーザーまたはIAMグループを削除して、以前のIAMユーザーまたはグループとまったく同じ名前を使用して再作成すると、そのIAMユーザーまたはIAMグループ名を使用するデータベースからの既存のマッピングは引き続き機能します。

1. ALTER USERシステム権限が付与されたユーザーとして、Oracle DBaasインスタンスにログインします。
2. IDENTIFIED GLOBALLY AS句を指定してALTER USER文を実行します。
   たとえば、既存のスキーマshared_sales_schemaを別のIAMグループに変更する場合は、次のようにします。

   ```
   ALTER USER shared_sales_schema IDENTIFIED GLOBALLY AS
   'IAM_GROUP_NAME=BiggerWidgetSalesGroup';
   ```

   次の例は、デフォルト以外のドメインsales_domainを指定してスキーマを変更する方法を示しています。

   ```
   ALTER USER shared_sales_schema IDENTIFIED GLOBALLY AS
   'IAM_GROUP_NAME=sales_domain/BiggerWidgetSalesGroup';
   ```

親トピック: IAMユーザーおよびOracle Cloud Infrastructureアプリケーションの認可の構成

7.2.2.6 インスタンス・プリンシパルおよびリソース・プリンシパルのマッピング

インスタンス・プリンシパルおよびリソース・プリンシパルは、アプリケーションでデータベース・トークンを取得して、Oracle DBaaSインスタンスへの接続を確立するために使用できます。

インスタンス・プリンシパルおよびリソース・プリンシパルを使用する場合は、動的グループのみをマップできます。インスタンス・プリンシパルおよびリソース・プリンシパルを排他的にマップすることはできません。これらをマップする場合は、共有マッピングを使用し、インスタンスまたはリソース・インスタンスをIAM動的グループに配置する必要があります。

関連トピック

- 動的グループの管理
- インスタンスからのサービスのコールに関する項
- 実行中のファンクションからの他のOracle Cloud Infrastructureリソースへのアクセスに関する項

親トピック: IAMユーザーおよびOracle Cloud Infrastructureアプリケーションの認可の構成

7.2.2.7 IAMユーザーのログオン情報の確認

Oracle DBaaSインスタンスのIAMユーザーを構成および認可した後、Oracleデータベース側で一連のSQL問合せを実行して、ユーザー・ログオン情報を確認できます。

1. 構成して認可したIAMユーザーとしてOracle DBaaSインスタンスにログインします。
   たとえば、IAMでデフォルト・ドメインを使用しているデータベース・グローバル・ユーザーpeterfitchとしてデータベース・インスタンスinst1にログインするには、次のようにします。

   ```
   sqlplus /nolog
   CONNECT "peterfitch"@inst1
   Enter password: password
   ```

   この例では、ユーザーpeterfitchがデフォルト以外のドメインsales_domainにある場合にログインする方法を示します。

   ```
   sqlplus /nolog
   CONNECT "sales_domain/peterfitch"@inst1
   Enter password: password
   ```

2. マップされたグローバル・ユーザーを検証します。

マップされたグローバル・ユーザーは、IAMユーザー認可を持つデータベース・ユーザー・アカウントです。ユーザーPETER_FITCH_SCHEMAはIAMユーザーpeterfitchの排他的マッピングを持つグローバル・ユーザーとみなされ、ユーザーWIDGET_SALESは、peterfitchがメンバーであるIAMグループwidget_sales_groupの共有マッピングを持つグローバル・ユーザーとみなされます。

```
SHOW USER;
```

排他的マッピングであるか共有マッピングであるかに応じて、次のような出力が表示されます。

```
USER is "PETER_FITCH_SCHEMA"
```

または

```
USER is "WIDGET_SALES"
```

3. 集中管理ユーザーに付与されたロールを確認します。

```
SELECT ROLE FROM SESSION_ROLES ORDER BY ROLE;
```

次のような出力が表示されます。

```
ROLE
--------------------------------------------------------------------
WIDGET_SALES_ROLE
...
```

4. 次の問合せを実行して、このデータベース・セッションで使用されている現在のスキーマのSYS_CONTEXTネームスペース値、現在のユーザー名、セッション・ユーザー名、認証方式、認証済アイデンティティ、エンタープライズ・アイデンティティ、識別タイプおよびサーバー・タイプを確認します。
    - このデータベース・セッションで使用されている現在のスキーマを確認します。データベース・スキーマは、含まれているオブジェクトを識別するオブジェクト・コンテナです。現在のスキーマは、このデータベース・セッションのオブジェクト名解決のデフォルト・コンテナです。

      ```
      SELECT SYS_CONTEXT('USERENV', 'CURRENT_SCHEMA') FROM DUAL;
      ```

      排他的マッピングであるか共有マッピングであるかに応じて、次のような出力が表示されます。

      ```
      SYS_CONTEXT('USERENV','CURRENT_SCHEMA')
      --------------------------------------------------------------------
      PETER_FITCH_SCHEMA
      ```

      または

      ```
      SYS_CONTEXT('USERENV','CURRENT_SCHEMA')
      --------------------------------------------------------------------
      WIDGET_SALES
      ```

    - 現行ユーザーを確認します。この場合、現行ユーザーは現行スキーマと同じです。

      ```
      SELECT SYS_CONTEXT('USERENV', 'CURRENT_USER') FROM DUAL;
      ```

      排他的マッピングであるか共有マッピングであるかに応じて、次のような出力が表示されます。

      ```
      SYS_CONTEXT('USERENV','CURRENT_USER')
      --------------------------------------------------------------------
      PETER_FITCH_SCHEMA
      ```

      または

```
SYS_CONTEXT('USERENV','CURRENT_USER')
--------------------------------------------------------------------------------
WIDGET_SALES
```

- セッション・ユーザーを確認します。

```
SELECT SYS_CONTEXT('USERENV', 'SESSION_USER') FROM DUAL;
```

排他的マッピングであるか共有マッピングであるかに応じて、次のような出力が表示されます。

```
SYS_CONTEXT('USERENV','SESSION_USER')
--------------------------------------------------------------------------------
PETER_FITCH_SCHEMA
```

または

```
SYS_CONTEXT('USERENV','SESSION_USER')
--------------------------------------------------------------------------------
WIDGET_SALES
```

- 認証方式を確認します。

```
SELECT SYS_CONTEXT('USERENV', 'AUTHENTICATION_METHOD') FROM DUAL;
```

次のような出力が表示されます。

```
SYS_CONTEXT('USERENV','AUTHENTICATION_METHOD')
--------------------------------------------------------------------------------
PASSWORD_GLOBAL
```

ユーザーがトークンを使用して認証している場合、出力はTOKEN_GLOBALです。

- エンタープライズ・ユーザーの認証済アイデンティティを確認します。このユーザーがデータベースにログオンしたときに、IAM認証済ユーザー・アイデンティティが取得されて監査されます。

```
SELECT SYS_CONTEXT('USERENV', 'AUTHENTICATED_IDENTITY') FROM DUAL;
```

次のような出力が表示されます。

```
SYS_CONTEXT('USERENV','AUTHENTICATED_IDENTITY')
--------------------------------------------------------------------------------
sales_domain/peterfitch
```

- ユーザーのニックネームがエンタープライズ・ユーザーに対して設定されている場合は、このニックネームを確認します。

```
SELECT SYS_CONTEXT('USERENV', 'USER_NICKNAME') FROM DUAL;
```

次のような出力が表示されます。

```
SYS_CONTEXT('USERENV','USER_NICKNAME')
--------------------------------------------------------------------------------
pfitch
```

- 集中管理ユーザーのエンタープライズ・アイデンティティを確認します。

```
SELECT SYS_CONTEXT('USERENV', 'ENTERPRISE_IDENTITY') FROM DUAL;
```

エンタープライズ・アイデンティティには、IAMユーザーまたはOCIアプリケーションのOCIアイデンティティ(OCID)が表示されます。次のような出力が表示されます。

```
SYS_CONTEXT('USERENV','ENTERPRISE_IDENTITY')
```

```
--------------------------------------------------------------------------------
ocid1.user.region1..aaaaaaaaj7ot4g2sagkjtw3enbg4ied3x554zwyywurgrm2232j4c
rm5zha
```

- 識別タイプを確認します。

```
SELECT SYS_CONTEXT('USERENV', 'IDENTIFICATION_TYPE') FROM DUAL
```

排他的マッピングであるか共有マッピングであるかに応じて、次のような出力が表示されます。

```
SYS_CONTEXT('USERENV','IDENTIFICATION_TYPE')
--------------------------------------------------------------------------------
GLOBAL EXCLUSIVE
```

または

```
SYS_CONTEXT('USERENV','IDENTIFICATION_TYPE')
--------------------------------------------------------------------------------
GLOBAL SHARED
```

- サーバー・タイプを確認します。

```
SELECT SYS_CONTEXT('USERENV', 'LDAP_SERVER_TYPE') FROM DUAL;
```

次のような出力結果が表示されます。この場合、LDAPサーバー・タイプはIAMです。

```
SYS_CONTEXT('USERENV','LDAP_SERVER_TYPE')
--------------------------------------------------------------------------------
OCI_IAM
```

親トピック: IAMユーザーおよびOracle Cloud Infrastructureアプリケーションの認可の構成

7.2.3 IAMプロキシ認証の構成

プロキシ認証により、IAMユーザーはアプリケーションのメンテナンスなどのタスクのためにデータベース・スキーマにプロキシできます。

- IAMプロキシ認証の構成について
  IAMユーザーは、プロキシ認証を使用してOracle DBaaSに接続できます。
- IAMユーザーのプロキシ認証の構成
  IAMユーザーのプロキシ認証を構成するには、IAMユーザーがグローバル・スキーマへのマッピング(排他的マッピングまたは共有マッピング)をすでに持っている必要があります。IAMユーザーがプロキシする別のデータベース・スキーマも使用可能である必要があります。
- IAMユーザー・プロキシ認証の検証
  パスワードとトークン両方の認証方法について、IAMユーザー・プロキシ構成を検証できます。

親トピック: IAM用のOracle DBaaSの構成

7.2.3.1 IAMプロキシ認証の構成について

IAMユーザーは、プロキシ認証を使用してOracle DBaaSに接続できます。

プロキシ認証は、通常、実際のユーザーを認証し、アプリケーションを管理するためにスキーマ権限およびロールを含むデータベース・スキーマの使用をユーザーに認可するために使用されます。アプリケーション・スキーマ・パスワードの共有などの代替方法は、安全でないものとみなされ、どの実際のユーザーがアクションを実行したかを監査できません。

たとえば、アプリケーション・データベース管理者である名前付きIAMユーザーが資格証明を使用して認証し、データベース・スキーマ・ユーザー(hrappなど)にプロキシできる環境でのユースケースが考えられます。この認証により、IAM管理者は、アプリ

ケーションのメンテナンスを実行するためにhrapp権限およびロールをユーザーhrappとして使用できますが、認証にはIAM資格証明を使用します。アプリケーション・データベース管理者は、データベースにサインインし、アプリケーション・スキーマにプロキシしてこのスキーマを管理できます。

パスワード認証とトークン認証両方の方法について、プロキシ認証を構成できます。

親トピック: IAMプロキシ認証の構成

7.2.3.2 IAMユーザーのプロキシ認証の構成

IAMユーザーのプロキシ認証を構成するには、IAMユーザーがグローバル・スキーマへのマッピング(排他的マッピングまたは共有マッピング)をすでに持っている必要があります。IAMユーザーがプロキシする別のデータベース・スキーマも使用可能である必要があります。

このタイプのユーザーがいることを確認したら、IAMユーザーにデータベース・ユーザーへのプロキシを許可するようにデータベース・ユーザーを変更します。

1. ALTER USERシステム権限を持つユーザーとして、Autonomous Databaseインスタンスにログインします。
2. ローカル・データベース・ユーザー・アカウントにプロキシする権限をIAMユーザーに付与します。
   IAMユーザーはコマンドで参照できないため、データベース・グローバル・ユーザー(IAMユーザーにマップ)とターゲット・データベース・ユーザーの間にプロキシを作成する必要があります。
   次の例では、hrappはプロキシ先のデータベース・スキーマで、peterfitch_schemaはユーザーpeterfitchに排他的にマップされるデータベース・グローバル・ユーザーです。

   ```
   ALTER USER hrapp GRANT CONNECT THROUGH peterfitch_schema;
   ```

この段階で、IAMユーザーはプロキシを使用してデータベース・インスタンスにログインできます。たとえば、パスワード・ベリファイアを使用して接続するには、次のようにします。

```
CONNECT peterfitch[hrapp]@connect_string
Enter password: password
```

トークンを使用して接続するには、次のようにします。

```
CONNECT [hrapp]/@connect_string
```

親トピック: IAMプロキシ認証の構成

7.2.3.3 IAMユーザー・プロキシ認証の検証

パスワードとトークン両方の認証方法について、IAMユーザー・プロキシ構成を検証できます。

1. CREATE USERおよびALTER USERシステム権限を持つユーザーとして、Autonomous Databaseインスタンスにログインします。
2. IAMユーザーとして接続し、SHOW USERおよびSELECT SYS_CONTEXTコマンドを実行します。
   たとえば、データベース・ユーザーhrappにプロキシするときに、IAMユーザーpeterfitchのプロキシ認証を確認するとします。IAMユーザーを使用してデータベースをプロキシした後、次の問合せを実行します。データベースの認証およびアクセス方法に応じて、これらの問合せに対して異なる値が取得されます。
   - パスワード認証の場合、IAMユーザーがデフォルト・ドメインにあると仮定すると、次のようになります。

     ```
     CONNECT peterfitch[hrapp]/password¥!@connect_string
     SHOW USER;
     --The output should be "USER is HRAPP"
     SELECT SYS_CONTEXT('USERENV','AUTHENTICATION_METHOD') FROM DUAL;
     --The output should be "PASSWORD_GLOBAL_PROXY"
     ```

```
SELECT SYS_CONTEXT('USERENV','PROXY_USER') FROM DUAL;
--The output should be "PETERFITCH_SCHEMA"
SELECT SYS_CONTEXT('USERENV','CURRENT_USER') FROM DUAL;
--The output should be "HRAPP"
```

- トークン認証の場合、デフォルト以外のドメインにあるユーザーに対し、sales_domain:

```
CONNECT [hrapp]/@connect_string
SHOW USER;
--The output should be USER is "HRAPP "
SELECT SYS_CONTEXT('USERENV','AUTHENTICATION_METHOD') FROM DUAL;
--The output should be "TOKEN_GLOBAL_PROXY"
SELECT SYS_CONTEXT('USERENV','PROXY_USER') FROM DUAL;
--The output should be "PETERFITCH_SCHEMA"
SELECT SYS_CONTEXT('USERENV','CURRENT_USER') FROM DUAL;
--The output should be "HRAPP"
```

親トピック: IAMプロキシ認証の構成

7.3 Oracle DBaaS用のIAMの構成

Oracle DBaaSインスタンスと連携するようにIAMを構成するには、IAM管理者がIAMポリシーを作成し、ユーザーにIAMのデータベース・パスワードを作成させる必要があります。

- トークンで認証されるユーザーを認可するためのIAMポリシーの作成
  Oracle DBaaSインスタンスと連携するようにIAMを構成するには、IAM管理者がIAMポリシーを作成し(IAMトークンを使用している場合)、IAMグループを作成してグループ・メンバーシップを管理する必要があります。
- IAMデータベース・パスワードの作成
  Oracle DBaaSパスワード検証プロセスには、Oracle Cloud Infrastructure (OCI)コンソール・パスワードとは異なる、IAMユーザーが設定したIAMデータベース・パスワードが必要です。

親トピック: Oracle DBaaS Databaseに対するIAMユーザーの認証および認可

7.3.1 トークンを使用して認証を行うユーザーを認可するためのIAMポリシーの作成

Oracle DBaaSインスタンスと連携するようにIAMを構成するには、IAM管理者がIAMポリシーを作成し(IAMトークンを使用している場合)、IAMグループを作成してグループ・メンバーシップを管理する必要があります。

IAM管理者は、データベース管理者と連携して、データベースの適切なIAMグループを作成する必要があります。個々のIAMユーザーは、パスワード・ベリファイアを使用している場合、プロファイルでIAMデータベース・パスワードを作成することが必要になります。
パスワード・ベリファイアを使用して認証するユーザーに対しては、ポリシーを作成する必要はありません。

- allow groupコマンドを使用して、ポリシーを作成します。たとえば、次のようになります。

```
allow group DBUsers to use database-connections in tenancy
```

- コンパートメントtesting_compartmentでのみDBaaSインスタンスにアクセスするようにDBUsersグループのメンバーを制限するポリシーを作成するには、次のようにします

```
allow group DBUsers to use autonomous-database-family in compartment
testing_compartment
```

- コンパートメント内の単一データベースにグループのアクセスを制限するポリシーを作成するには、次のようにします。

```
allow group DBUsers to use autonomous-database-family in compartment
testing_compartment where target.database.id =
'ocid1.autonomousdatabase.oc1.iad.aaaabbbbcccc'
```

次のことに注意してください。

- database-connectionsリソース・タイプは、autonomous-database-familyリソース・タイプに含まれています。ユースケースに応じて、どちらのリソースも使用できます。
- データベースへのアクセスを有効にする最小動詞はuseです。manage動詞を使用して、データベースへのアクセスを有効にすることもできます。

ポリシー文の構文の詳細は、Oracle Cloud Infrastructureドキュメンテーションを参照してください。

親トピック: Oracle DBaaS用のIAMの構成

7.3.2 IAMデータベース・パスワードの作成

Oracle DBaaSパスワード検証プロセスには、Oracle Cloud Infrastructure (OCI)コンソール・パスワードとは異なる、IAMユーザーが設定したIAMデータベース・パスワードが必要です。

OCI IAMデータベース・パスワードで使用できる文字のセットは、OCIコンソール・パスワードで使用できる文字のセットと似ていますが、OCI IAMデータベース・パスワードでは二重引用符文字は使用できない点が異なります。IAMデータベース・パスワードの作成の詳細は、ユーザー資格証明の管理に関する項を参照してください。

1. OCIコンソールにログインして、ユーザー・ページに移動します。
2. 使用しているIAMバージョンに応じて、「自分のプロファイル」または「ユーザー設定」(ナビゲーション・ツールバーの右上)にアクセスします。
3. プロファイルまたは設定の左側の「リソース」で、「データベース・パスワード」リンクをクリックします。
4. 「データベース・パスワードの作成」ボタンをクリックします。
5. リストされている複雑度のルールを確実に適用して、説明とパスワードを追加します。
6. 「データベース・パスワードの作成」をクリックして、パスワードを保存します。
   パスワードが作成されると、その説明と作成日が「データベース・パスワード」の下に表示されます。

親トピック: Oracle DBaaS用のIAMの構成

7.4 インスタンス・プリンシパルまたはリソース・プリンシパルを使用したデータベースへのアクセス

Oracle Cloud Infrastructure (OCI)アプリケーションまたは関数は、それ固有のインスタンス・プリンシパルまたはリソース・プリンシパルを使用してデータベース・インスタンスに接続できます。

動的グループへのマッピングを使用して、インスタンス・プリンシパルおよびリソース・プリンシパルをデータベース・グローバル・スキーマまたは共有スキーマに排他的にマップできます。インスタンス・プリンシパルとリソース・プリンシパルをデータベース・グローバル・スキーマに排他的にマップする場合は、プリンシパルのOCIDを使用する必要があります。例:

```
CREATE USER widget IDENTIFIED GLOBALLY
AS 'IAM_PRINCIPAL_OCID=ocid1.instance.region1.sea.1234567890abcdef';
```

共有スキーマを使用する場合は、インスタンス・プリンシパルおよびリソース・プリンシパルを動的グループに追加し、その動的グループを共有スキーマにマップする必要があります。

関連トピック

- 動的グループの管理に関する項
- インスタンスからのサービスのコールに関する項
- 実行中のファンクションからの他のOracle Cloud Infrastructureリソースへのアクセスに関する項
- インスタンス・プリンシパルを使用したOracle Cloud Infrastructure APIへのアクセス
- Oracle Autonomous Database Serverlessの使用

親トピック: Oracle DBaaS Databaseに対するIAMユーザーの認証および認可

7.5 データベース・クライアント接続の構成

IAMクライアント接続の構成によって、Oracle DBaaSインスタンスへのIAMユーザーの認証が制御されます。

- IAMを使用したAutonomous Databaseインスタンスへの接続について
  IAMユーザーは、IAMデータベース・パスワード・ベリファイアまたはIAMトークンのいずれかを使用して、Autonomous Databaseインスタンスに接続できます。
- IAM接続でサポートされるクライアント・ドライバ
  Oracle DBaaSは、IAM接続用の複数のタイプのクライアント・ドライバをサポートしています。
- IAMデータベース・パスワード・ベリファイアを使用するクライアント接続
  IAMユーザーに必要な認可を構成すると、このユーザーは、追加の構成なしで、SQL*PlusやSQLclなどの既存のクライアント・アプリケーションを使用してログインできます。
- IAMユーザー名およびデータベース・パスワードでリクエストされたトークンを使用するクライアント接続
  IAMユーザー名およびデータベース・パスワードでリクエストされたトークンを使用するクライアント接続を作成できます。
- クライアント・アプリケーションまたはツールでリクエストされたトークンを使用するクライアント接続
  Oracle DBaaSへのIAMトークン・アクセスの場合、クライアント・アプリケーションまたはツールによって、IAMユーザーのために、IAMからデータベース・トークンがリクエストされます。
- クライアント・ウォレットを使用しないTLS接続
  クライアント・ウォレットを使用しないTransport Layer Security (TLS)接続の使用は、IAM接続でサポートされています。
- 一般的なデータベース・クライアント構成
  IAMユーザーは、ラップトップでSQLclなどのクライアント・ツールを使用して、Oracle DBaaSインスタンスに接続できます。

親トピック: Oracle DBaaS Databaseに対するIAMユーザーの認証および認可

7.5.1 IAMを使用したAutonomous Databaseインスタンスへの接続について

IAMユーザーは、IAMデータベース・パスワード・ベリファイアまたはIAMトークンを使用して、Autonomous Databaseインスタンスに接続できます。

IAMデータベース・パスワード・ベリファイアの使用は、Oracle Databaseパスワード認証プロセスに似ています。ただし、パスワード・ベリファイア(パスワードの暗号化ハッシュ)はOracle Databaseに格納されるのではなく、Oracle Cloud Infrastructure (OCI) IAMユーザー・プロファイルの一部として格納されます。

データベースにIAMトークンを使用する2番目の接続方法は、より新しいものです。トークン・ベース・アクセスの使用は、Autonomous Databaseなどのクラウド・リソースに適しています。トークンは、IAMエンドポイントで強制できる強度に基づいています。これはマルチファクタ認証にでき、パスワードのみを使用するより強力です。トークンを使用するもう1つの利点は、パスワード・ベリファイア(機密とみなされる)がメモリーに格納されることがなく、メモリーから取得できないことです。データベース・アクセスにトークンを使用する場合は、TCPS (TLS)接続が必要です。

ノート:

IAM トークンを渡すときにネイティブ・ネットワーク暗号化を設定することはできません。Transport Layer Security (TLS)のみがサポートされ、ネイティブ・ネットワーク暗号化や TLS によるネイティブ・ネットワーク暗号化はサポートされていません。

親トピック: データベース・クライアント接続の構成

7.5.2 IAM接続でサポートされるクライアント・ドライバ

Oracle DBaaSは、IAM接続用の複数のタイプのクライアント・ドライバをサポートしています。

IAMデータベースのパスワード・ベリファイアは、サポートされているデータベース・クライアントと連携します。IAMトークンを使用するには、最新のOracle Database Client 19c (19.16以上)が必要です。以前の一部のクライアント(19cおよび21c)では、トークン・アクセス用の制限された機能セットが用意されています。Oracle Database Client 21cは、IAMトークン・アクセス機能を十分にはサポートしていません。Oracle Database Client 23cは、IAMトークン・アクセス機能をサポートしています。

親トピック: データベース・クライアント接続の構成

7.5.3 IAMデータベース・パスワード・ベリファイアを使用するクライアント接続

IAMユーザーに必要な認可を構成すると、このユーザーは、追加の構成なしで、SQL*PlusやSQLclなどの既存のクライアント・アプリケーションを使用してログインできます。

IAMユーザーは、現在サポートされている任意のデータベース・クライアントを使用して、IAMユーザー名とIAMデータベース・パスワード(Oracle Cloud Infrastructure (OCI)コンソール・パスワードではありません)を入力します。唯一の制約は、データベース・クライアントのバージョンがOracle Databaseリリース12.1.0.2以降で、Oracle Database 12cのパスワードを使用することです。データベース・クライアントで、12Cのパスワード・ベリファイアを使用できる必要があります。11Gのベリファイア暗号化の使用は、IAMではサポートされていません。IAMユーザーがOCI DBaaSインスタンスに接続するために、特別なクライアントやツールの構成は必要ありません。

親トピック: データベース・クライアント接続の構成

7.5.4 IAMユーザー名およびデータベース・パスワードでリクエストされたトークンを使用するクライアント接続

IAMユーザー名およびデータベース・パスワードでリクエストされたトークンを使用するクライアント接続を作成できます。

- IAMユーザー名およびデータベース・パスワードでリクエストされたトークンを使用するクライアント接続について
  IAMユーザーは、IAMユーザー名およびIAMデータベース・パスワードを使用して取得されたIAMトークンを使用して、Oracle DBaaSインスタンスに接続できます。
- IAMユーザー名およびデータベース・パスワードでリクエストされたトークンを使用するクライアント接続に設定するパラメータ
  これらのパラメータを設定するには、`sqlnet.ora`ファイルまたは`tnsnames.ora`ファイルを変更します。
- IAMユーザー名およびデータベース・パスワードを使用してトークンを取得するためのデータベース・クライアントの構成
  指定したIAMユーザー名およびIAMデータベース・パスワードを使用してIAMデータベース・トークンを取得するようにデータベース・クライアントを構成できます。

- IAMトークンを取得するための安全性の高い外部パスワード・ストア・ウォレットの構成
  IAMユーザー名および安全性の高い外部パスワード・ストア(SEPS)でIAMデータベース・トークンをリクエストするようにできます。

親トピック: データベース・クライアント接続の構成

7.5.4.1 IAMユーザー名およびデータベース・パスワードでリクエストされたトークンを使用するクライアント接続について

IAMユーザーは、IAMユーザー名およびIAMデータベース・パスワードを使用して取得されたIAMトークンを使用して、Oracle DBaaSインスタンスに接続できます。

どちらの場合も、トークンは、SQL*PlusまたはSEPSを使用して、データベース・パスワードで取得されます。

以前のリリースでは、IAMのユーザー名とパスワードのみを使用して、IAMからパスワード・ベリファイアを取得できました。パスワード・ベリファイアは機密とみなされるため、これらの資格証明によるトークンの取得は、パスワード・ベリファイアの取得よりも安全です。トークンを使用することは、ベリファイアを渡したり使用する必要がないことを意味します。アプリケーションは、データベース・クライアントAPIを介してIAMのユーザー名およびパスワードで取得されたトークンを渡すことはできません。このタイプのトークンを取得できるのは、データベース・クライアントのみです。データベース・クライアントは、IAMユーザー名およびIAMデータベース・パスワードを使用してのみデータベース・トークンを取得できます。

IAMユーザー名およびIAMデータベース・パスワードをツールに直接入力するか、SEPSウォレットを使用してこれらの資格証明を安全に格納できます。

親トピック: IAMユーザー名およびデータベース・パスワードでリクエストされたトークンを使用するクライアント接続

7.5.4.2 IAMユーザー名およびデータベース・パスワードでリクエストされたトークンを使用するクライアント接続に設定するパラメータ

これらのパラメータを設定するには、`sqlnet.ora`ファイルまたは`tnsnames.ora`ファイルを変更します。

IAMユーザー名およびデータベース・パスワードのトークン・リクエストのトークン固有パラメータ

- PASSWORD_AUTHパラメータ

  認証方法を設定します。この構成では、`OCI_TOKEN`の設定を使用する必要があります。パスワード・ベリファイアは機密とみなされるため、ユーザーおよびパスワード資格証明を使用したトークンの取得は、パスワード・ベリファイアを使用するよりも安全です。このパラメータは、IAMユーザー名およびデータベース・パスワードでIAMベアラー・トークンを取得するために必要です。

  構文:

  ```
  PASSWORD_AUTH=authentication_method
  ```

  例:

  ```
  PASSWORD_AUTH=OCI_TOKEN
  ```

- OCI_IAM_URLパラメータ

  データベース・トークンを取得するためにデータベース・クライアントが接続する必要があるIAM URLを指定します。このパラメータは、IAMユーザー名およびデータベース・パスワードを使用してIAMベアラー・トークンを取得するために必要です。この設定はリージョンに固有です。リージョンの適切なURLは、Identity and Access Managementデータ・プレーンAPIを参照してください。次に、リージョンURLに

/v1/actions/generateScopedAccessBearerTokenを追加します。

構文:

```
OCI_IAM_URL=authentication_regional_endpoint.com/v1/actions/generateScopedAccessBearerToken
```

例:

次の例では、Phoenix URL (https://auth.us-phoenix-1.oraclecloud.com)を使用しています。

```
https://auth.us-phoenix-1.oraclecloud.com/v1/actions/generateScopedAccessBearerToken
```

- OCI_TENANCYパラメータ

  ユーザーのテナンシのOCIDを指定します。この設定は、OCIコンソールの右上にあるユーザーのアイコンの下にあります。このパラメータは、IAMユーザー名およびデータベース・パスワードを使用してIAMベアラー・トークンを取得するために必要です。

  構文:

  ```
  OCI_TENANCY=tenancy_OCI..OCID
  ```

  例:

  region1の後に2つのピリオドがあることに注意してください。

  ```
  OCI_TENANCY=ocid1.tenancy.region1..12345
  ```

- OCI_COMPARTMENTパラメータ

  データベース・トークン・リクエストの範囲を定義します。region_nameの後に2つのピリオドがあることに注意してください。トークンは、指定したコンパートメント内のデータベースにのみ使用できます。この値を省略すると、テナンシ全体がリクエストの範囲になります。OCI_DATABASEが設定されている場合を除き、このパラメータはオプションです。

  構文:

  ```
  OCI_COMPARTMENT=compartment_OCID
  ```

  例:

  region1の後に2つの期間があることに注意してください。

  ```
  OCI_COMPARTMENT=ocid1.compartment.region1..12345
  ```

- OCI_DATABASEパラメータ

  アクセスするデータベースのOCIDを指定します。このパラメータは、トークンをデータベースのみに制限します。このパラメータは省略可能です。

  構文:

  ```
  OCI_DATABASE=database_OCID
  ```

  例:

  ```
  OCI_DATABASE=ocid1.autonomousdatabase.oc1.iad.12345
  ```

IAMユーザー名およびデータベース・パスワードのトークン・リクエストのDN固有パラメータ

- SSL_SERVER_CERT_DNパラメータ

  このクライアントのデータベース・サーバーの識別名(DN)を指定します。(このパラメータはベアラー・トークンに固有ではありません。)

  構文:

  ```
  SSL_SERVER_CERT_DN=DN
  ```

  例:

  ```
  SSL_SERVER_CERT_DN="C=US,O=ExampleCorporation,CN=sslserver2"
  ```

- SSL_SERVER_DN_MATCHパラメータ

  DN一致を介してサーバー側の検証を強制します。このパラメータをTRUEに設定します。

  構文:

  ```
  SSL_SERVER_DN_MATCH=TRUE|FALSE
  ```

  例:

  ```
  SSL_SERVER_DN_MATCH=TRUE
  ```

sqlnet.oraの例

```
PASSWORD_AUTH=OCI_TOKEN
OCI_IAM_URL=https://auth.region1.example.com/v1/actions/generateScopedAccessBearerTok
en
OCI_TENANCY=ocid1.tenancy..12345
OCI_COMPARTMENT=ocid1.compartment.region1..12345
OCI_DATABASE=ocid1.autonomousdatabase.oc1.iad.12345
SSL_SERVER_CERT_DN="C=US,O=ExampleCorporation,CN=sslserver2",
SSL_SERVER_DN_MATCH=TRUE
```

tnsnames.oraの例

```
db_connection=
  (DESCRIPTION=
    (ADDRESS=(PROTOCOL=tcps)(HOST=sales1-svr)(PORT=5678))
    (SECURITY=
      (PASSWORD_AUTH=OCI_TOKEN)

(OCI_IAM_URL=https://auth.region1.example.com/v1/actions/generateScopedAccessBearerTo
ken)
      (OCI_TENANCY=ocid1.tenancy..12345)
      (OCI_COMPARTMENT=ocid1.compartment.region1..12345)
      (OCI_DATABASE=ocid1.autonomousdatabase.oc1.iad.12345)
      (SSL_SERVER_CERT_DN="C=US,O=ExampleCorporation,CN=sslserver2")
      (SSL_SERVER_DN_MATCH=TRUE))
    (CONNECT_DATA=(SERVICE_NAME=sales.us.example.com)))
```

詳細は、次のとおりです。

- (PROTOCOL=tcps)は、プロトコルをTCPSに設定します。プロトコルとしてTCPSを使用する必要があります。そうしないと、接続に失敗します。データベース・クライアントからサーバーにトークンを渡す場合は、TCPSを有効にする必要があります。
- SECURITYでは、認証およびDNパラメータを設定します。

親トピック: IAMユーザー名およびデータベース・パスワードによってリクエストされたトークンを使用するクライアント接続

7.5.4.3 IAMユーザー名およびデータベース・パスワードを使用してトークンを取得するためのデータベース・クライアントの構成

指定したIAMユーザー名およびIAMデータベース・パスワードを使用してIAMデータベース・トークンを取得するようにデータベース・クライアントを構成できます。

1. Oracle DBaaSクライアントにログインします。
2. IAMユーザー名およびデータベース・パスワードでリクエストされるトークンを取得するように適切なパラメータを設定します。
3. `sqlnet.ora`ファイルで、`WALLET_LOCATION`パラメータをクライアントの場所に設定します。ルート証明書はこのディレクトリに格納されます。

   例:

```
WALLET_LOCATION =
  (SOURCE=
    (METHOD=FILE)
      (METHOD_DATA=
        DIRECTORY=/ora_db/wallet)
```

関連トピック

- IAMユーザー名およびデータベース・パスワードでリクエストされたトークンを使用するクライアント接続に設定するパラメータ

親トピック: IAMユーザー名およびデータベース・パスワードによってリクエストされたトークンを使用するクライアント接続

7.5.4.4 IAMトークンを取得するための安全性の高い外部パスワード・ストア・ウォレットの構成

IAMユーザー名および安全性の高い外部パスワード・ストア(SEPS)でIAMデータベース・トークンをリクエストするように設定できます。

1. Oracle DBaaSクライアントにログインします。
2. 安全性の高い外部パスワード・ストアを使用するようにこのクライアントを構成します。
3. IAMユーザー名およびデータベース・パスワードによってリクエストされたトークンを取得するように、適切なパラメータを設定します。

関連トピック

- 安全性の高い外部パスワード・ストアの使用を目的とするクライアントの構成
- IAMユーザー名およびデータベース・パスワードによってリクエストされたトークンを使用するクライアント接続のために設定するパラメータ

親トピック: IAMユーザー名およびデータベース・パスワードによってリクエストされたトークンを使用するクライアント接続

7.5.5 クライアント・アプリケーションまたはツールによってリクエストされたトークンを使用するクライアント接続

Oracle DBaaSへのIAMトークン・アクセスの場合、クライアント・アプリケーションまたはツールによって、IAMユーザーのために、IAMからデータベース・トークンがリクエストされます。

クライアント・アプリケーションは、データベース・クライアントAPIを介してデータベース・トークンを直接データベース・クライアントに

渡します。

IAMトークンをリクエストするようにアプリケーションまたはツールが更新されていない場合、IAMユーザーはOracle Cloud Infrastructure (OCI)コマンドライン・インタフェース(CLI)を使用して、データベース・トークンをリクエストおよび格納できます。次の資格証明を使用して、データベース・アクセス・トークン(`db-token`)をリクエストできます。

- セキュリティ・トークン(IAM認証あり)、委任トークン(OCIクラウド・シェル内)および`API-keys`。これらは、認証を有効にするためにIAMユーザーを表す資格証明です
- インスタンス・プリンシパル・トークン。これは、認証後にインスタンスが、サービス・リソースに対してアクションを実行するための認可されたアクター(またはプリンシパル)となることができるようにします
- リソース・プリンシパル・トークン。これは、アプリケーションが他のOracle Cloud Infrastructureサービスに対してそれ自体を認証できるようにする資格証明です

IAMユーザーがスラッシュ/ログインを使用してクライアントにログインし、`OCI_IAM`パラメータが構成されている場合(`sqlnet.ora`、`tnsnames.ora`または接続文字列の一部として)、データベース・クライアントはファイルからデータベース・トークンを取得します。IAMユーザーがユーザー名とパスワードを送信すると、接続で、IAMデータベース・パスワード・ベリファイアを使用するクライアント接続について記述されているIAMデータベース・ベリファイア・アクセスが使用されます。このガイドの手順では、データベース・トークンのヘルパーとしてOCI CLIを使用する方法を示します。アプリケーションまたはツールがIAMと連携するように更新されている場合は、アプリケーションまたはツールの手順に従います。一般的なユースケースには、SQLPlusオンプレミス、SQLclオンプレミス、Cloud ShellのSQL*Plus、SEPウォレットを使用するアプリケーションなどがあります。

関連トピック

- IAMデータベース・パスワード・ベリファイアを使用するクライアント接続

親トピック: データベース・クライアント接続の構成

7.5.6 クライアント・ウォレットを使用しないTLS接続

クライアント・ウォレットを使用しないTransport Layer Security (TLS)接続の使用は、IAM接続でサポートされています。

このタイプの接続を構成する前に、Oracle DBaaS環境が要件を満たしていることを確認してください。

関連トピック

- クライアント・ウォレットを使用しないTransport Layer Security接続

親トピック: データベース・クライアント接続の構成

7.5.7 一般的なデータベース・クライアント構成

IAMユーザーは、ラップトップでSQLclなどのクライアント・ツールを使用して、Oracle DBaaSインスタンスに接続できます。

- IAMデータベース・パスワードを使用するSQL*Plusのクライアント接続の構成
  IAMデータベース・パスワードを使用するようにSQL*Plusを構成できます。
- IAMトークンを使用するSQL*Plusのクライアント接続の構成
  IAMトークンを使用するSQL*Plusのクライアント接続を構成できます。

親トピック: データベース・クライアント接続の構成

7.5.7.1 IAMデータベース・パスワードを使用するSQL*Plusのクライアント接続の構成

IAMデータベース・パスワードを使用するようにSQL*Plusを構成できます。

- IAMユーザーとして、次の構文を使用してAutonomous Databaseインスタンスにログインします。

```
CONNECT user_name@db_connect_string
Enter password: password
```

ここで、user_nameはIAMユーザー名です。結合されたdomain_name/user_nameには128バイトの制限があります。

次の例は、IAMユーザーpeter_fitchがAutonomous Databaseインスタンスにログインする方法を示しています。

```
sqlplus /nolog
connect peter_fitch@db_connect_string
Enter password: password
```

いくつかの特殊文字によって、user_nameおよびpasswordの前後に二重引用符が必要になります。例:

```
"peter_fitch@example.com"@db_connect_string
"IAM database password"
```

親トピック: 一般的なデータベース・クライアント構成

7.5.7.2 IAMトークンを使用するSQL*Plusのクライアント接続の構成

IAMトークンを使用するSQL*Plusのクライアント接続を構成できます。

1. IAMユーザー・アカウントがあることを確認します。
2. IAM管理者およびOracle Database管理者に問い合せて、コンパートメントまたはテナンシ内のデータベースにアクセスできるポリシーがあり、データベースのグローバル・スキーマにマップされていることを確認します。
3. アプリケーションまたはツールで直接のIAM統合がサポートされていない場合は、OCI CLIをダウンロードしてインストールし、構成します。(OCIコマンドライン・インタフェースのクイックスタートに関する項を参照。)OCI CLI構成の一部としてAPIキーを設定し、デフォルト値を選択します。
    a. IAMユーザーのAPIキー・アクセスを設定します。
    b. db-tokenを取得します。たとえば、次のようにします。
        - Oracle Cloud Infrastructure (OCI)コマンドライン・インタフェースで、API-keyを使用してdb-tokenを取得するには、次のようにします。

          ```
          oci iam db-token get
          ```

        - セキュリティ(またはセッション)トークンを使用してdb-tokenを取得するには、次のようにします。

          ```
          oci iam db-token get --auth security_token
          ```

          セキュリティ・トークンの有効期限が切れると、ユーザーがOCIに再度ログインするためのウィンドウが表示されます。これにより、ユーザーのセキュリティ・トークンが生成されます。OCI CLIでは、このリフレッシュされたトークンを使用してdb-tokenを取得します。

        - 委任トークンを使用したdb-tokenの取得: クラウド・シェルにログインすると、委任トークンが自動的に生成され、/etcディレクトリに配置されます。このトークンを取得するには、クラウド・シェルで次のコマンドを実行します。

          ```
          oci iam db-token get
          ```

        - OCIコマンドライン・インタフェースを使用してインスタンス・トークンを取得するには、次のようにします。

          ```
          oci iam db-token get --auth instance_principal
          ```

c. データベース・クライアントは、IAMユーザー名およびIAMデータベース・パスワードを使用してデータベース・トークンを取得するように構成することもできます。

詳細は、IAMユーザー名およびデータベース・パスワードでリクエストされたトークンを使用するクライアント接続を参照してください。

詳細は、必要なキーとOCIDを参照してください。

4. Oracle Databaseクライアント・リリース19cおよび21cの最新リリース更新を使用していることを確認します。
この構成は、Oracle Databaseクライアント・リリース19cまたは21cでのみ機能します。
5. 既存のプロセスに従ってAutonomous Databaseからウォレットをダウンロードし、SQL*Plusで使用するためにウォレットを構成する手順に従います。
   a. sqlnet.oraでSSL_SERVER_DN_MATCH=ONを検索して、DN一致が有効になっていることを確認します。
   b. sqlnet.oraファイルにTOKEN_AUTH=OCI_TOKENを追加して、IAMトークンを使用するようにデータベース・クライアントを構成します。データベース・トークン・ファイルのデフォルトの場所を使用するため、トークンの場所を含める必要はありません。

tnsnames.ora接続文字列内のTOKEN_AUTHおよびTOKEN_LOCATION値は、その接続のsqlnet.ora設定より優先されます。たとえば、接続文字列について、トークンがデフォルトの場所(Linuxの場合は~/.oci/db-token)にあるとします。

```
(description=
  (retry_count=20)(retry_delay=3)
  (address=(protocol=tcps)(port=1522)
  (host=example.us-phoenix-1.oraclecloud.com))

(connect_data=(service_name=aaabbbccc_exampledb_high.example.oraclecloud.com))
  (security=(ssl_server_cert_dn="CN=example.uscom-east-1.oraclecloud.com,
     OU=Oracle BMCS US, O=Example Corporation,
     L=Redwood City, ST=California, C=US")
  (TOKEN_AUTH=OCI_TOKEN)))
```

接続文字列をTOKEN_AUTHパラメータで更新した後、IAMユーザーは次のコマンドを実行してSQL*Plusを起動し、Autonomous Databaseインスタンスにログインできます。接続記述子自体を含めるか、tnsnames.oraファイルからの記述子の名前を使用できます。

```
connect /@exampledb_high
```

または:

```
connect /@(description=
  (retry_count=20)(retry_delay=3)
  (address=(protocol=tcps)(port=1522)
  (host=example.us-phoenix-1.oraclecloud.com))
  (connect_data=(service_name=aaabbbccc_exampledb_high.example.oraclecloud.com))
  (security=(ssl_server_cert_dn="CN=example.uscom-east-1.oraclecloud.com,
     OU=Oracle BMCS US, O=Example Corporation,
     L=Redwood City, ST=California, C=US")
  (TOKEN_AUTH=OCI_TOKEN)))
```

TOKEN_AUTHはsqlnet.oraファイルまたは接続文字列のいずれかによってすでに設定されているため、データベース・クライアントはdb-tokenを取得するようにすでに構成されています。データベース・クライアントは、db-tokenを取得し、秘密キーを使用して署名してからトークンをAutonomous Databaseに送信します。スラッシュ/のかわりにIAMユーザー名とIAMデータベース・パスワードが指定されている場合、データベース・クライアントは、db-tokenではなくパスワードを使用して接続します。

親トピック: 一般的なデータベース・クライアント構成

7.6 Oracle DBaaSとIAMの統合でのデータベース・リンク

IAM資格証明を使用してOracle DBaaSデータベースにアクセスする際のデータベース・リンクの使用がサポートされています。

IAMへのOracle DBaaS接続のデータベース・リンクを構成する方法は、Oracle DBaaSプラットフォームによって異なります。Oracle DBaaSプラットフォームに対応する次のトピックを確認し、詳細は関連リンクをクリックしてください。

- Oracle Autonomous Database Serverless: 固定データベース・リンクにデータベース・ユーザーが使用される、固定ユーザー・データベース・リンクを使用できます。データベース・リンクを作成するためのデータベース・ユーザーは、データベース・リンクでパスワード認証のみを使用できます。IAMユーザーは、パスワードまたはトークン・アクセスを使用してソース・データベースに対する認証を行えます。IAMユーザーを固定データベース・リンクとして構成することや、接続ユーザーまたは現行のユーザーのデータベース・リンクを使用することはできません。「Oracle Autonomous Database Serverlessの使用」を参照してください
- Oracle Autonomous Database on Dedicated Exadata InfrastructureおよびAutonomous Database以外のDBaaSプラットフォーム: 接続ユーザーおよび固定ユーザー・データベース・リンクを使用できますが、現行のユーザーのデータベース・リンクは使用できません。接続ユーザー・データベース・リンクの場合、IAMユーザーはソース・リンク・データベースとターゲット・リンク・データベースの両方にプロビジョニングされている必要があります。データベース・パスワード・ベリファイアまたはIAMデータベース・トークンを使用して、接続ユーザー・データベース・リンクに接続および使用できます。固定ユーザー・データベース・リンクの場合、ユーザーはターゲット・データベース・ユーザーを使用してパスワード認証により、ターゲット・データベースに接続できます。また、IAMユーザーは、IAMのユーザー名およびパスワードまたはIAMトークンを使用して、最初のPDBに接続できます。専用Exadataインフラストラクチャでの Oracle Autonomous Databaseの使用を参照してください

親トピック: Oracle DBaaS Databaseに対するIAMユーザーの認証および認可

7.7 IAM接続のトラブルシューティング

「ORA-01017: ユーザー名/パスワードが無効です。ログオンは拒否されました」エラーは、Identity and Access Management (IAM)とOracle DBaaSの統合全体の様々な問題が原因である可能性があります。

- ORA-01017エラーについてクライアント側で確認する領域
  クライアント側のORA-01017エラーは、IAM資格証明の問題、クライアント構成またはIAMプロファイルの問題が原因で発生する可能性があります。
- データベース・クライアントのトレース・ファイル
  クライアント側でIAM接続をトラブルシューティングするために、2つのレベルのトレース・ファイルを生成できます。
- Oracle Cloud Infrastructure IAMおよびOracle DatabaseでのORA-01017エラーのチェック
  Oracle DatabaseインスタンスのORA-01017エラーは、IAMと連携するようにデータベースを設定した方法が原因で発生する可能性があります。
- 不適切に構成されたIAMユーザーが原因のORA-01017エラー
  不適切に構成されたIAMユーザーから、いくつかのORA-01017エラーが発生する可能性があります。
- トークンを使用してデータベースにアクセスしようとしたときに発生するORA-12599およびORA-03114エラー
  ORA-12599: 「TNS: 暗号チェックサムの不一致が発生しました」エラーおよびORA-03114: 「Oracleに接続されていません。」エラーは、接続しようとしているデータベースがネイティブ・ネットワーク暗号化によって保護されていることを示しています。
- IAM管理者がORA-01017エラーに対処するために実行できるアクション
  ORA-01017エラーに対処するいくつかのアクションは、IAM管理者のみが実行できます。

7.7.1 ORA-01017エラーについてクライアント側で確認する領域

クライアント側のORA-01017エラーは、IAM資格証明の問題、クライアント構成またはIAMプロファイルの問題が原因で発生する可能性があります。

IAMトークンのトラブルシューティング

- トークンに使用されるOracle Cloud Infrastructure (OCI) CLIのバージョンを確認します。OCI CLIは、OCIバージョン3.4以上である必要があります。これには、IAMから新しいdb-tokenを取得するコマンドが含まれています。OCIのバージョンを確認するには、次のコマンドを実行します。

  ```
  oci --version
  ```

- Oracle Databaseクライアント・バージョンを確認します。最新バージョンを確認するには、Oracle Databaseのドキュメントを確認します。現在、次のドライバのみがサポートされています。
  - JDBC: バージョン19.13.0.0.1以降のバージョンの19c JDBCクライアント、JDBC: バージョン21.5以降のバージョンの21c
  - Instant Client/SQL*Plus (Linuxのみ): バージョン19.13 (-2の注釈付き)以降のバージョンの19c
  - Instant Client/OCI/SQL*Plus (Linuxのみ): バージョン21.5以降のバージョンの21c (すべての機能がInstant Client/OCIバージョン21cでサポートされているわけではありません。可能な場合は最新の19cまたはバージョン23cクライアントを使用してください。)
  - SQLcl: バージョン21.4以降
  - ODP.net: バージョン19.13以降のバージョンの19c
  - ODP.net: バージョン21.4以降のバージョンの21c
  - Oracle Databaseリリース23c: すべてのクライアント

  これらのドライバの最新バージョンは、IAMトークンを使用してデータベースにアクセスするときに必要です。サポートされているすべてのデータベース・クライアントは、IAMデータベースのパスワードの使用時に動作します。

- tnsnames.oraファイルに指定されたトークンの場所を確認します。データベース・クライアントとOCI CLIは、データベース・トークンと秘密キーを格納および取得するのに、同じデフォルトの場所(~/.oci/db-token)を使用します。異なる場所を指定できますが、OCI CLIとデータベース・クライアントの両方が同じディレクトリを使用するように構成される必要があります。正しいTOKEN_LOCATION値がtnsnames.oraまたはsqlnet.oraファイルの接続文字列で指定されていることを確認します。sqlnet.oraのTOKEN_LOCATIONの値よりもtnsnames.oraが優先され、それよりも接続文字列が優先されます。
- トークンが期限切れかどうかを確認します。IAMデータベース・トークンは1時間のみ有効です。データベース・トークンの期限が切れたら、API-keyを使用している場合は、次のOCI CLIコマンドを再実行して、別のトークンをリクエストします。

  ```
  oci iam db-token get
  ```

- tnsnames.oraのTOKEN_AUTHパラメータ値を確認します。パラメータTOKEN_AUTH=OCI_TOKENが接続文字列、tnsnames.oraまたはsqlnet.oraのいずれかで設定されていることを確認します。TOKEN_AUTHの値については、接続文字列、tnsnames.ora、sqlnet.oraの順に優先されます。
- デフォルトのユーザー指定のトークンの場所で、トークンまたは秘密キーが欠落しているかどうかを確認します。OCI CLIコマンドoci cli db-token getを実行した後で、TOKEN_LOCATIONで指定されたディレクトリにトークンと秘密キーの両方があることを確認します。db-tokenおよび秘密キーの場所は、次のコマンドを実行して確認できます。

```
[oracle@localhost ~]$ oci iam db-token get
```

次のような出力が表示されます。

```
Private key written at /home/oracle/.oci/db-token/oci_db_key.pem
db-token written at: /home/oracle/.oci/db-token/token
db-token is valid until 2022-01-05 15:36:51
```

場所がTOKEN_LOCATIONの設定と一致しない場合は、OCI CLIコマンドを更新するか、TOKEN_LOCATIONパラメータを更新します。

- 自分のOCI IAMプロファイルを確認します。
  - 公開APIキーがOCIユーザー・アカウントに存在することを確認します。OCI CLIでは、デフォルトで、クライアントでAPI-keyを使用して、IAMからdb-tokenをリクエストします。公開APIキーがOCIユーザー・アカウントに存在しない場合、IAMはデータベース・トークンを戻しません。
  - IAMアカウントがロックされていないことを確認します。ロックされている場合は、IAM管理者にロック解除を依頼します。
  - IAMデータベース・パスワードを使用している場合は、IAMプロファイルでIAMデータベース・パスワードを設定していることを確認します。
- APIキーを使用していない場合は、セキュリティ・トークンを使用していることを明示的に指定します。次のコマンドを使用します。

```
oci iam db-token get --auth security_token
```

セキュリティ・トークンが存在しないか期限切れの場合、このコマンドは、IAM (またはフェデレーテッドIdP)へのサインインのためにブラウザを開こうとします。環境にブラウザがない場合、このコマンドは失敗します。

IAMデータベース・パスワードとIAMトークンの両方のトラブルシューティング

- Oracle Instant Clientのクライアント・トレースのみを確認します。SQL*PlusとInstant Clientを併用すると、クライアント・トレースで、いくつかの情報が示されます。Oracle Databaseクライアント・トレース・ファイルは、2つの異なるトレース・レベルを使用して生成できます。

関連トピック

- データベース・クライアントのトレース・ファイル

親トピック: IAM接続のトラブルシューティング

7.7.2 データベース・クライアントのトレース・ファイル

クライアント側でIAM接続をトラブルシューティングするために、2つのレベルのトレース・ファイルを生成できます。

生成できるトレース・ファイルの2つのレベルは次のとおりです。

- 低レベル・トレースでは、エラーの発生時にトレースが出力されます。
  - TCPSがIAM接続用に設定されていない場合は、プロトコルがTCPSである必要があるというメッセージが出力されます。
  - SSL_SERVER_DN_MATCHがTRUEに設定されていない場合は、値がFALSEであるというメッセージが出力されます。
  - 無効なTOKEN_LOCATIONが指定されている場合は、トークンの場所が存在しないというメッセージが出力されます。

        - 指定されたTOKEN_LOCATIONまたはデフォルトのトークンの場所にdb-tokenおよび秘密キーが存在しない場合は、メッセージが出力されます。
        - アプリケーションがdb-tokenまたは秘密キーのみを渡した場合は、欠落している属性のメッセージが出力されます。
        - db-tokenが期限切れの場合は、メッセージが出力されます。
    - 高レベル・トレースでは、前述のようにエラーの発生時にトレースが出力されます。さらに、次のように、成功時にトレースが出力されます。
        - SSL_SERVER_DN_MATCHが存在する場所(tnsnames.oraまたはsqlnet.ora)が出力されます。また、TRUEに設定されている場合は、TRUEの値が出力されます。
        - db-tokenと秘密キーの両方がアプリケーションによって設定されている場合は、メッセージが出力されます。
        - TOKEN_AUTHに、正しい値OCI_TOKENがある場合は、値が出力されます。
        - db-tokenが期限切れでない場合は、メッセージが出力されます。

IAM接続のクライアント・トレースを制御する場合は、次のいずれかの方法を使用できます。

- クライアント側のsqlnet.oraファイルに次の設定を追加します。
    - EVENT_25701=14 (低レベル・トレースの場合)
    - EVENT_25701=15 (高レベル・トレースの場合)
- 環境変数EVENT_25701を設定します。
    - EVENT_25701=14 (低レベル・トレースの場合)
    - EVENT_25701=15 (高レベル・トレースの場合)

クライアント・トレース・ファイルは、次の場所に作成されます。

- Linux: $ORACLE_HOME/log/diag/clients
- Windows: %ORACLE_HOME%¥log¥diag¥clients

クライアント側のsqlnet.oraのADR_BASEパラメータを使用して、トレース・メッセージが格納されるディレクトリを指定できます。ディレクトリ・パスが有効で、書込み権限があることを確認します。diag_adr_enabledパラメータがfalseに設定されていないことを確認します。

ADR_BASEの設定例を次に示します。

```
ADR_BASE=/oracle/iam/trace
```

親トピック: IAM接続のトラブルシューティング

7.7.3 Oracle Cloud Infrastructure IAMおよびOracle DatabaseでのORA-01017エラーの確認

Oracle DatabaseインスタンスのORA-01017エラーは、IAMと連携するようにデータベースを設定した方法が原因で発生する可能性があります。

- IAM構成が有効になっているかどうかを確認します。IAM統合用にOCIサーバーが構成されおり、1つ以上のデータベース・スキーマ(データベース・ユーザー)がIAMユーザーまたはグループにマップされている必要があります。これは、IAMトークンとIAMデータベース・パスワードの両方のユースケースに適用されます。構成が有効になっているかどうかを

確認するには、SQL*Plusで次のコマンドを実行します。

```
SELECT NAME, VALUE
FROM V$PARAMETER
WHERE NAME='identity_provider_type';
```

あるいは、次のコマンドを使用します。

```
SHOW PARAMETER IDENTITY_PROVIDER_TYPE
```

戻り値がOCI_IAMではない場合は、外部認証を有効にします。

- IAMにマップされているスキーマを確認します。どのIAMユーザーおよびIAMグループがマッピングで使用されるかに注意してください。この情報を確認するには、SQL*Plusで次の問合せを実行します。

```
SELECT USERNAME, EXTERNAL_NAME, CREATED
FROM DBA_USERS
WHERE AUTHENTICATION_TYPE='GLOBAL';
```

出力で、IAM_USERまたはIAM_GROUPで始まるEXTERNAL_NAMEが少なくとも1つあることを確認します。IAMユーザーまたはグループの名前を書き留めます。グローバル・スキーマがない場合は、新しいスキーマを作成するか、既存のスキーマを変更し、そのユーザーがメンバーであるIAMユーザーまたはIAMグループにマップする必要があります。

- Oracle Databaseインスタンスを再起動する必要があるかどうかをチェックします。場合によっては、IAM構成が導入される前に存在していたデータベース・インスタンスを再起動する必要があります。ただし、それを行う前に、他のすべてのトラブルシューティング・ガイドラインに従ってから、データベースを再起動してください

関連トピック

- IAMユーザーおよびOracle Cloud Infrastructureアプリケーションの認可の構成

親トピック: IAM接続のトラブルシューティング

7.7.4 不適切に構成されたIAMユーザーが原因のORA-01017エラー

不適切に構成されたIAMユーザーから、いくつかのORA-01017エラーが発生する可能性があります。

- IAMユーザーがOracle DBaaSインスタンスにログインできることを確認します。IAMユーザーに、フェデレーテッド・ユーザーとしてではなく、IAMユーザーのログインを試行するように依頼します。このユーザーがアカウントからロックアウトされていないことを確認します。(これが発生した場合は、ユーザーがIAM管理者に連絡する必要があります。)ユーザーのIAMアカウントがロックされている場合、このユーザーはOracle DBaaSインスタンスにログインできません。

  また、IAMユーザー名と、ユーザーがメンバーとして所属するIAMグループも確認する必要があります。これらのユーザー名またはグループ名のいずれかが、Oracle DBaaSサーバーから検出された、マップされたIAMユーザーおよびグループ名と一致する必要があります。マッピングがない場合、ユーザーのデータベースへのアクセスが拒否されます。その場合、IAM管理者が、ユーザーがアクセスする必要があるDBaaSインスタンスにマップされているIAMグループに、ユーザーを追加する必要があります。

- API公開キーがIAMユーザー・プロファイルに登録されていることを確認します。IAMを使用したOracle DBaaSインスタンス構成でトークンが使用され、API-keyを使用してデータベース・トークンを取得する場合、API公開キーがユーザーのIAMユーザー・プロファイルに登録される必要があります。
- IAMユーザー・プロファイルでIAMデータベース・パスワードが設定されていることを確認します。IAMを使用したOracle DBaaSインスタンス構成でデータベース・パスワード認証が使用されている場合は、IAMデータベース・パスワードがユーザーのIAMユーザー・プロファイルに設定されていることを確認します。また、Database Passwordsが、IAM

ユーザー・プロファイルのUser Capabilityセクションの許可される設定であることを確認します。

関連トピック

- IAMユーザーおよびOracle Cloud Infrastructureアプリケーションの認可の構成

親トピック: IAM接続のトラブルシューティング

7.7.5 トークンを使用してデータベースにアクセスしようとしたときにORA-12599およびORA-03114エラーが発生する

ORA-12599: 「TNS: 暗号チェックサムの不一致が発生しました」エラーおよびORA-03114: 「Oracleに接続されていません。」エラーは、接続しようとしているデータベースがネイティブ・ネットワーク暗号化によって保護されていることを示します。

トークンを使用してOracleデータベースにアクセスする場合は、ネットワーク・ネイティブ暗号化ではなくTransport Layer Security (TLS)接続を確立する必要があります。これらのエラーを修正するには、TLSがデータベースに対して適切に構成されていることを確認してください。ローカル・データベースのユーザー名とパスワードを使用して構成をテストし、次のSYSCONTEXT USERENVパラメータを確認する必要があります。

- NETWORK_PROTOCOL
- TLS_VERSION

関連トピック

- Transport Layer Security認証の構成

親トピック: IAM接続のトラブルシューティング

7.7.6 IAM管理者がORA-01017エラーに対処するために実行できるアクション

ORA-01017エラーに対処するいくつかのアクションは、IAM管理者のみが実行できます。

- IAMユーザーがAPIキーを再作成する必要があるかどうかを確認します。IAMユーザーが削除されてから、まったく同じユーザー名で再作成された場合、Oracle Cloud Infrastructure (OCI) IAMでは、これは別のユーザーOCIDを持つ別のユーザーとみなされます。この場合、IAMユーザーはユーザー・アカウントおよびAPI-keyを再作成する必要があります。このアクションは、データベースのIAMユーザーおよびIAMグループ・マッピングには影響しません。
- 必要に応じて、IAMユーザー・アカウントのロックを解除します。ユーザーが非アクティブであるか、ロックされている場合、データベース・アクセスが許可されるようにするには、IAM管理者がユーザー・アカウントのロックを解除する必要があります。
- IAMポリシーを確認します。ユーザーがIAMデータベース・トークンを使用してデータベースにアクセスできるようにするには、IAMポリシーが必要です。リソースはdatabase-connectionsと呼ばれ、autonomous-database-familyのメンバーでもあります。Oracle DBaaSインスタンスでIAMデータベース・パスワードを使用している場合、IAMポリシーを作成する必要はありません。IAMポリシーを構成する場合は、ポリシーにuseまたはmanageのタグが必要であることに注意してください。例:
    - テナンシでautonomous-database-familyを使用するようにallow all-usersを設定します。これにより、すべてのIAMテナンシ・ユーザーは、IAMデータベース・トークンを使用して、テナンシ内のすべてのOracle DBaaSインスタンスにアクセスできます。
    - production_compartmentコンパートメントでdatabase-connectionsを使用するようにallow group DBUsersを設定します。これにより、DBUsers IAMグループのメンバーであるIAMユーザーは、

IAMトークンを使用して、production_compartmentコンパートメントのデータベースにアクセスできます。

- IAMユーザーおよびグループのマッピングを確認します。IAMユーザーは、データベース内のスキーマ(つまり、データベース・ユーザー)からの排他的マッピングを持っているか、データベース内のスキーマにマップされているIAMグループのメンバーです。次のSQL*Plus問合せを実行し、その出力を確認して、マップされたIAMユーザーおよびグループを探します。ユーザーにデータベース・スキーマへの1つのマッピングがあることを確認します。

```
SELECT USERNAME, EXTERNAL_NAME,
FROM DBA_USERS
WHERE AUTHENTICATION_TYPE='GLOBAL';
```

関連トピック

- トークンを使用して認証を行うユーザーを認可するためのIAMポリシーの作成

親トピック: IAM接続のトラブルシューティング

8 Oracle DatabaseのMicrosoft Azure Active Directoryユーザーの認証および認可

Oracle Databaseは、Microsoft Azure ADユーザーがシングル・サインオンを使用して接続するように構成できます。

- Microsoft Azure ADとのOracle Database統合の概要
  Microsoft Azure ADを構成してOracleデータベースにアクセスするには、全体的なプロセスについて理解する必要があります。
- Microsoft Azure AD統合のためのOracle Databaseの構成
  Microsoft Azure ADとOracle Databaseインスタンスの統合では、データベースをAzure ADに登録する必要があります。
- Oracle Databaseスキーマおよびロールのマッピング
  Azure ADユーザーは、1つのデータベース・スキーマにマップされ、オプションで1つ以上のデータベース・ロールにマップされます。
- Oracle DatabaseへのAzure ADクライアント接続の構成
  Azure AD登録済データベースに接続するようにクライアント接続を構成できます
- Microsoft Azure ADプロキシ認証の構成
  プロキシ認証により、Azure ADユーザーは、アプリケーションのメンテナンスなどのタスクのためにデータベース・スキーマにプロキシできます。
- Microsoft Azure AD接続のトラブルシューティング
  トレース・ファイルを使用して、Microsoft Azure AD接続の問題を診断できます。`ORA-12599`および`ORA-03114`エラーを簡単に修正することもできます。

親トピック: ユーザー認証および認可の管理

8.1 Oracle DatabaseとMicrosoft Azure ADの統合の概要

Oracleデータベースにアクセスするための Microsoft Azure ADの構成を開始する前に、全体的なプロセスについて理解する必要があります。

- Oracle DatabaseとMicrosoft Azure ADの統合について
  Oracle DatabaseおよびMicrosoft Azure ADは、ユーザーおよびアプリケーションがAzure AD資格証明を使用してデータベースに接続できるように構成できます。
- Oracle DatabaseとMicrosoft Azure ADの統合のアーキテクチャ
  Microsoft Azure Active Directoryアクセス・トークンは、OAuth 2.0標準とその拡張に従います。
- Oracle Databaseスキーマおよびロールへの Azure ADユーザーのマッピング
  Microsoft AzureユーザーをOracle Databaseインスタンスに対して認証するには、その前にMicrosoft Azureユーザーをデータベーススキーマにマップし、(ロールを介して)必要な権限を付与しておく必要があります。
- Azure ADを使用したOracle Databaseへの接続のユースケース
  Oracle Databaseでは、データベースへの接続のユースケースをいくつかサポートしています。
- Oracle DatabaseでMicrosoft Azure ADアイデンティティを認証する一般的なプロセス
  Azure AD OAuth2アクセス・トークンを使用してAzure ADユーザーがデータベースに接続できるように、Oracle Database管理者およびMicrosoft Azure AD管理者がロールを再生します。

親トピック: Oracle DatabaseのMicrosoft Azure Active Directoryユーザーの認証および認可

8.1.1 Oracle DatabaseとMicrosoft Azure ADの統合の概要

Oracle DatabaseおよびMicrosoft Azure ADは、ユーザーおよびアプリケーションがAzure AD資格証明を使用してデータベースに接続できるように構成できます。

Azure ADのユーザーおよびアプリケーションは、Azure ADシングル・サインオン(SSO)資格証明を使用してログインし、データベースにアクセスできます。これは、ユーザーまたはアプリケーションが最初にAzure ADからリクエストするAzure AD OAuth2アクセス・トークンを使用して行われます。このOAuth2アクセス・トークンには、ユーザーIDおよびデータベース・アクセス情報が含まれ、データベースに送信されます。マルチファクタ認証およびパスワードレス認証の構成の詳細は、Microsoft社の記事Azure Active Directoryのパスワードレス認証オプションを参照してください。

この統合は、次のOracle Database環境で実行できます。

- オンプレミスOracle Databaseリリース19.18以降
- 共有Exadataインフラストラクチャ上のOracle Autonomous Database
- Oracle Autonomous Database on Dedicated Exadata Infrastructure
- Oracle Base Database Service
- Oracle Exadata Cloud Service (Oracle ExaCS)

Azure ADを構成する手順では、これらの環境を網羅するために「Oracle Database」という用語を使用します。

このタイプの統合により、Azure ADユーザーはOracle Databaseインスタンスにアクセスできます。Azure ADのユーザーおよびアプリケーションは、Azure ADシングル・サインオン(SSO)資格証明を使用してログインし、Azure ADのOAuth2アクセス・トークンを取得してデータベースに送信できます。

Azure AD管理者は、Oracle Databaseを作成してAzure ADに登録します。Azure AD内では、これはアプリ登録と呼ばれ、アプリケーション登録の略です。これは、Azure ADを使用しているソフトウェアについてAzure ADが理解しておく必要があるデジタル情報です。Azure AD管理者は、Azure ADでデータベース・アプリを登録するためのアプリケーション(アプリ)ロールも作成します。アプリケーション・ロールは、Azureユーザー、グループおよびアプリケーションをデータベース・スキーマおよびロールに接続します。Azure AD管理者は、Azure ADユーザー、グループ、またはアプリケーションをアプリケーションのロールに割り当てます。これらのアプリケーション・ロールは、データベース・グローバル・スキーマ、グローバル・ロール、またはスキーマとロールの両方にマップされます。アプリケーション・ロールに割り当てられたAzure ADユーザー、グループ、またはアプリケーションは、データベース・グローバル・スキーマ、グローバル・ロール、あるいはスキーマとロールの両方にマップされます。Oracleグローバル・スキーマは、Azure ADユーザーに排他的にマップすることもできます。Azure ADゲスト・ユーザー(組織ユーザー以外)またはAzure ADサービス・プリンシパル(アプリケーション)のみ、Azure ADアプリケーション・ロールを介してデータベース・グローバル・スキーマにマップできます。Oracleグローバル・ロールはAzureアプリケーション・ロールからのみマップでき、Azureユーザーからはマップできません。

Azure ADトークンをサポートするように更新されたツールおよびアプリケーションにより、Azure ADでユーザーを直接認証し、データベース・アクセス・トークンをOracle Databaseインスタンスに渡すことができます。SQL*Plusなどの既存のデータベース・ツールを構成して、ファイルの場所からAzure ADトークンを使用できます。このような場合、Azure ADトークンは、Microsoft PowerShellやAzure CLIなどのツールを使用して取得し、ファイルの場所に配置できます。Azure AD OAuth2データベース・アクセス・トークンは、有効期限付きで発行されます。Oracle Databaseクライアント・ドライバは、トークンをデータベースに渡す前に、トークンの形式が有効であることと、有効期限が切れていないことを確認します。トークンのスコープはデータベースです。つまり、トークンにはそのトークンが使用されるデータベースに関する情報が含まれています。データベースAzure ADアプリ登録でAzure ADプリンシパルが割り当てられたアプリケーション・ロールは、アクセス・トークンの一部として含められます。Azure ADトークンのディレクトリの場所には、ユーザーがトークン・ファイルをその場所に書き込み、データベース・クライアントがこれらのファイルを取得するために十分な権限のみ(ユーザーによる読取りおよび書込みのみなど)を付与する必要があります。データベー

スへのアクセスはトークンによって許可されるため、トークンはファイル・システム内で保護される必要があります。

Azure ADユーザーは、いくつかの方法を使用してAzure ADからトークンをリクエストし、Azureログイン・ウィンドウを開いてAzure AD資格証明を入力できます。

Oracle Databaseは、次のAzure ADプリンシパルを表すトークンを受け入れます。

- Azure ADユーザー。Azure ADテナンシの登録ユーザーです
- ゲスト・ユーザー。Azure ADテナンシでゲスト・ユーザーとして登録されているユーザーです
- サービス。クライアント資格証明フロー(接続プール・ユースケース)を使用し、それ自体としてデータベースに接続する登録済アプリケーションです

Oracle Databaseでは、次のAzure AD認証フローがサポートされています。

- 認証コード。ブラウザを使用して、クライアント環境のAzure ADへの認証に使用する、(アプリケーションではなく)人間のユーザーに最も一般的に使用されます
- クライアント資格証明。(エンド・ユーザーとしてではなく)それ自体として接続するデータベース・アプリケーションで使用されます
- On-Behalf-Of (OBO)。アプリケーションがログイン・ユーザーの代理としてアクセス・トークンを要求してデータベースに送信します
- リソース所有者パスワード資格証明(ROPC)。本番環境での使用は推奨されませんが、ポップアップ・ブラウザでのユーザー認証の組込みが困難なテスト環境で使用できます。ROPCでは、トークン・リクエスト・コールの一部としてAzure ADユーザー名とパスワード資格証明が必要です。

親トピック: Oracle DatabaseとMicrosoft Azure ADの統合の概要

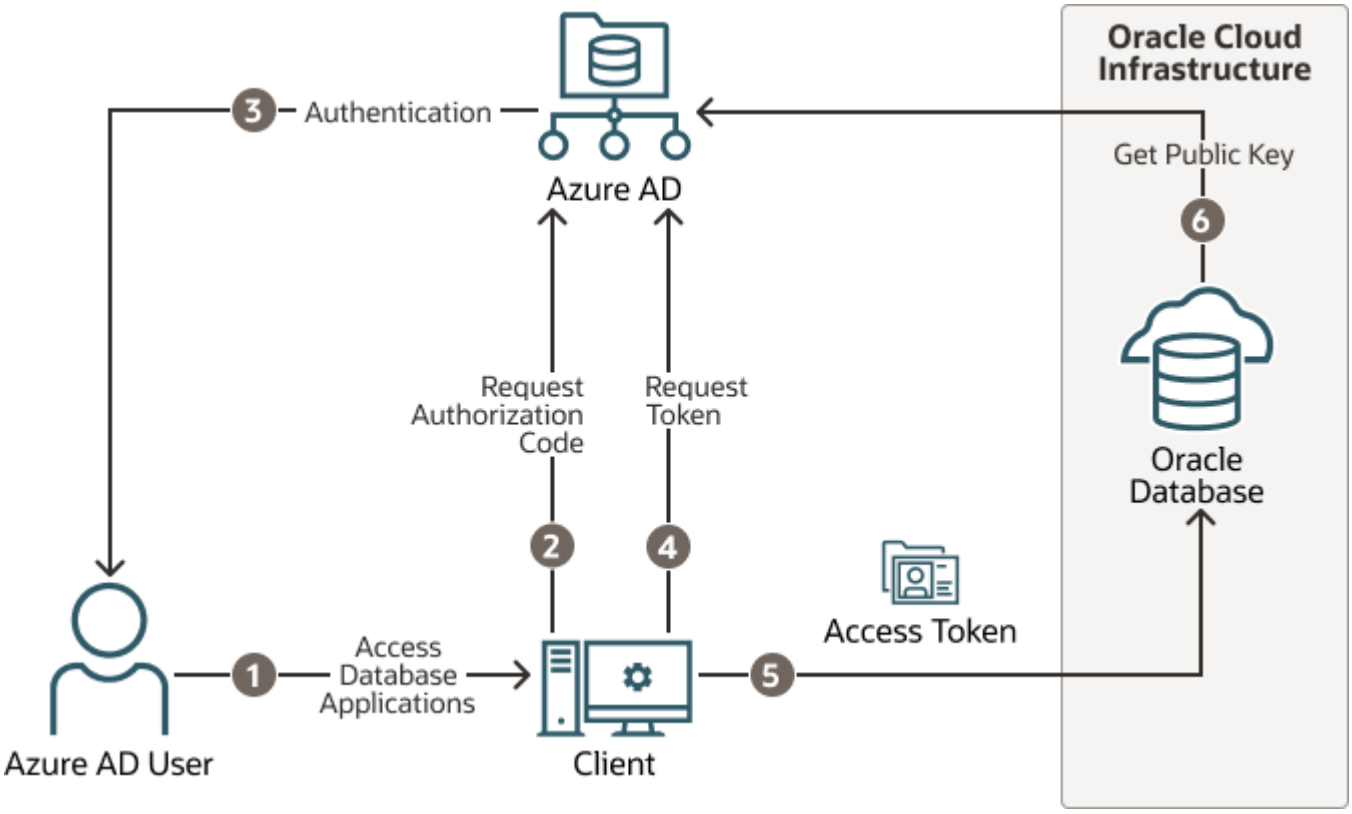
8.1.2 Oracle DatabaseとMicrosoft Azure ADの統合のアーキテクチャ

Microsoft Azure Active Directoryアクセス・トークンは、OAuth 2.0標準とその拡張に従います。

Azure ADアクセス・トークンは、データベース・クライアント(SQLPlusまたはSQLclなど)からデータベースにアクセスする前に必要です。Oracleクライアント(OCI、JDBC、ODPなど)がファイルの場所からAzure ADトークンを選択するよう構成されたり、トークンはデータベース・クライアントAPIを介してクライアントに渡されることがあります。Azureユーザーはスクリプト(例についてはMicrosoftを参照)を使用してトークンを取得し、それをデータベース・クライアントが取得できるファイルの場所に置くことができます。アプリケーションは、Azure SDKを使用してアクセス・トークンを取得し、データベース・クライアントAPIを介してそのトークンを渡すことができます。アプリケーションが直接トークンを取得できない場合は、Microsoft PowerShellやAzureコマンドライン・インタフェースなどのコマンドライン・ツールを使用して、Azure ADトークンを取得できます。

次の図は、0Auth2トークンを使用したOAuth 2.0標準の一般的なフロー図です。サポートされている各フローの詳細は、Microsoft Azure ADドキュメントの『Authentication flow support in MSAL』を参照してください。

図8-1 対話型認可コード・フローを使用したAzure ADユーザーによるデータベースへのアクセス



認可コード・フローはOAuth2標準で、標準の一部として詳細に説明されています。フローには2つのステップがあります。最初のステップでは、ユーザーを認証し、認可コードを取得します。2番目のステップでは、認可コードを使用してデータベース・アクセス・トークンを取得します。

1. Azure ADユーザーは、リソース、Oracle Databaseインスタンスへのアクセスをリクエストします。
2. データベース・クライアントまたはアプリケーションは、Azure ADからの認可コードをリクエストします。
3. Azure ADはAzure ADユーザーを認証し、認可コードを返します。
4. ヘルパー・ツールまたはアプリケーションはAzure ADの認可コードを使用して、OAuth2トークンと交換します。
5. データベース・クライアントは、OAuth2アクセス・トークンをOracleデータベースに送信します。トークンには、データベースのためのAzure ADアプリケーション登録でユーザーが割り当てられたデータベース・アプリケーション・ロールが含まれます。
6. Oracle DatabaseインスタンスはAzure AD公開キーを使用して、アクセス・トークンがAzure ADによって作成されたことを確認します。

データベース・クライアントとデータベース・サーバーの両方を、AzureポータルのAzure Active Directoryセクションで「app registrations」機能を使用して登録する必要があります。データベース・クライアントは、Azure ADアプリケーション登録で登録する必要があります。データベース・クライアントには、データベースのアクセス・トークン取得を許可する権限も付与する必要があります。

親トピック: Oracle DatabaseとMicrosoft Azure ADの統合の概要

8.1.3 Oracle Databaseスキーマおよびロールの Azure ADユーザーのマッピング

Microsoft AzureユーザーをOracle Databaseインスタンスに対して認証するには、その前にMicrosoft AzureユーザーをOracle Databaseスキーマにマップし、(ロールを介して)必要な権限を付与しておく必要があります。

Microsoft Azureでは、Azure AD管理者は、ユーザー、グループおよびアプリケーションをデータベース・アプリケーション・ロールに割り当てることができます。

Azure ADスキーマをデータベース・スキーマに排他的にマップするには、Azure ADユーザーが組織に参加するとき、またはデー

タベースに対して認可されるときにデータベース管理者がデータベース・スキーマを作成する必要があります。また、データベース管理者は、Azure ADユーザーが割り当てられているタスクに合わせて、データベース・スキーマに付与されている権限およびロールを変更する必要があります。Azure ADユーザーが組織を離れた場合、データベース管理者は、使用されていないアカウントがデータベースに残らないようにデータベース・スキーマを削除する必要があります。データベース・アプリケーション・ロールを使用すると、Azure AD管理者は、グローバル・スキーマとグローバル・ロールにマップされているアプリケーション・ロールにユーザーを割り当てることによってアクセスおよびロールを制御できます。このようにして、データベースへのユーザー・アクセスはAzure AD管理者によって管理されるため、データベース管理者がすべてのユーザーのスキーマを作成、管理および削除する必要はありません。

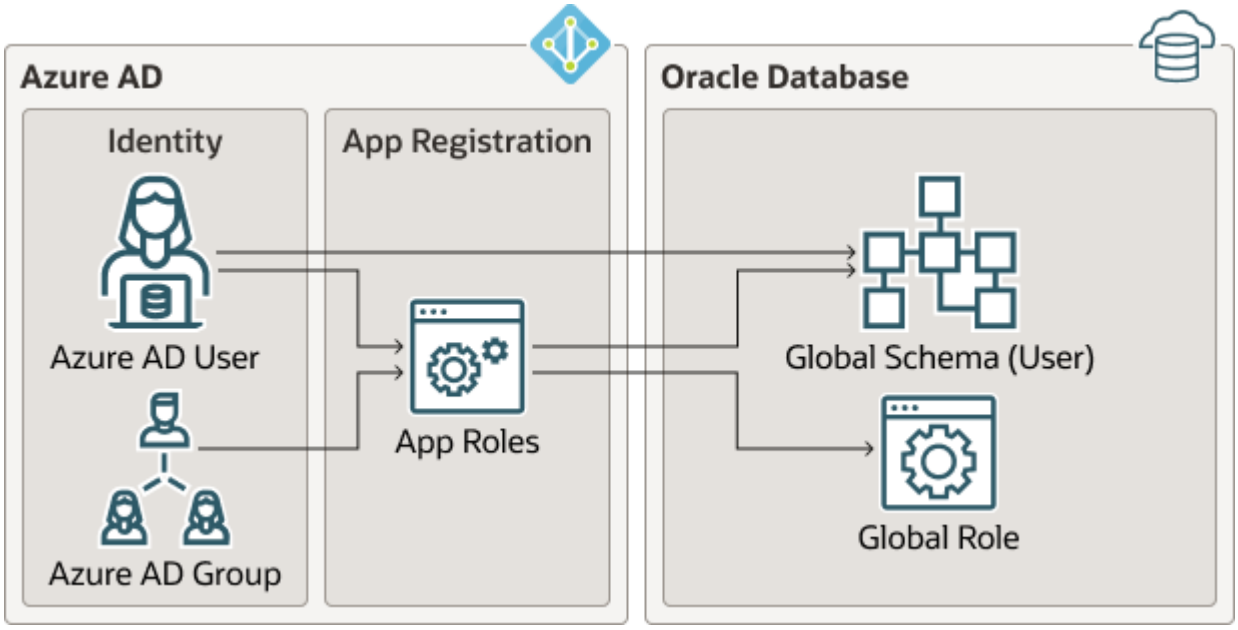
Azure ADユーザーは、データベース・スキーマ(ユーザー)に排他的にまたはアプリケーション・ロールを介してマップできます。

- Azure ADユーザーとOracle Databaseスキーマの間の排他的マッピングの作成。このタイプのマッピングでは、Azure ADユーザーに対してデータベース・スキーマを作成する必要があります。Azure ADユーザーが必要とするデータベース権限およびロールは、データベース・スキーマに付与する必要があります。データベース・スキーマは、Azure ADユーザーがデータベースに対して認可された場合に作成するだけでなく、付与された権限およびロールをAzure ADのロールおよびタスクの変更に応じて変更する必要があります。最後に、データベース・スキーマはAzure ADユーザーが組織を離れるときに削除する必要があります。
- Azure ADアプリケーション・ロールとOracle Databaseスキーマの間の共有マッピングの作成。このタイプのマッピングは排他的マッピングよりも一般的で、アプリケーション・ロールに直接割り当てられるAzure ADユーザーか、アプリケーション・ロールに割り当てられるAzure ADグループのメンバー用です。アプリケーション・ロールは、Oracle Databaseスキーマにマップされます(共有スキーマ・マッピング)。共有スキーマ・マッピングを使用すると、複数のAzure ADユーザーが同じOracle Databaseスキーマを共有できるため、新規ユーザーが組織に加入するたびに新しいデータベース・スキーマを作成する必要はありません。この運用効率により、データベース管理者は、新規ユーザーの構成、権限とロールの更新、およびアカウントの削除を行わずに、データベース・アプリケーションのメンテナンス、パフォーマンスおよびチューニングのタスクに集中できます。

マップされたグローバル・スキーマに直接付与されるデータベース・ロールおよび権限に加えて、マップされたグローバル・ロールを介して追加のロールおよび権限を付与できます。同じ共有グローバル・スキーマにマップされている異なるAzure ADユーザーで、異なる権限およびロールが必要な場合があります。Azureアプリケーション・ロールは、Oracle Databaseグローバル・ロールにマップできます。アプリケーション・ロールに割り当てられているAzure ADユーザーまたはアプリケーション・ロールに割り当てられているAzure ADグループのメンバーであるAzure ADユーザーには、データベースへのアクセス時にOracle Databaseグローバル・ロールが付与されます。

次の図は、使用可能な様々なタイプの割当ておよびマッピングを示しています。

図8-2 Azure ADとOracle Databaseの間の割当てとマッピング



これらのマッピングは、次のようになります。

- Azure ADユーザーは、Oracle Databaseグローバル・スキーマ(ユーザー)に直接マップできます。
- Azure ADユーザー、Azure ADグループまたはアプリケーションはアプリケーション・ロールに割り当てられ、さらにOracle Databaseグローバル・スキーマ(ユーザー)またはグローバル・ロールのいずれかにマップされます。

親トピック: Oracle DatabaseとMicrosoft Azure ADの統合の概要

8.1.4 Azure ADを使用したOracle Databaseへの接続のユースケース

Oracle Databaseでは、データベースへの接続にいくつかのユースケースをサポートしています。

- OAuth2認可コード・フロー: 人間のユーザーに対して最も一般的なフローです。クライアントは、Azure ADユーザーにAzure ADで認可コードを取得するよう指示します。このコードは、データベース・アクセス・トークンの取得に使用されます。Microsoft Azureの記事『Microsoft identity platform and OAuth 2.0 authorization code flow』を参照してください。
- リソース所有者のパスワード資格証明(ROPC): このフローは、本番サーバーではお薦めしません。ポップアップ認証ウィンドウでは動作しないテスト・ソフトウェアで使用できます。これは、非グラフィックのユーザー・インタフェース環境でユーザーの認証にポップアップ・ウィンドウを使用できない場合に使用されます。
- クライアント資格証明: このフローは、アプリケーションのデータベースへの接続に使用されます。アプリケーションはAzure ADアプリケーションの登録によって登録する必要があり、クライアントIDとクライアント・パスワードが必要です。これらのクライアント資格証明は、アプリケーションがデータベースに接続するときのAzure ADからのデータベース・アクセス・トークン取得に使用する必要があります。アプリケーションは、ファイル・システムまたはデータベース・クライアントAPIを介してトークンを渡すことができます。
- On-Behalf-Of (OBO)トークン: :Azureアプリケーションは、ログイン・ユーザーのOBOトークンをリクエストします。OBOトークンは、Azure ADユーザー・アイデンティティおよびデータベースに割り当てられたアプリケーション・ロールを持つデータベースのアクセス・トークンでもあります。これにより、Azure ADユーザーはアプリケーションではなくユーザーとしてデータベースにログインできます。Azure ADユーザーのOBOトークンをリクエストし、APIを介してデータベース・クライアントに渡すことができるのはアプリケーションのみです。

親トピック: Oracle DatabaseとMicrosoft Azure ADの統合の概要

8.1.5 Oracle DatabaseでMicrosoft Azure ADアイデンティティを認証する一般的なプロセス

Azure AD OAuth2アクセス・トークンを使用してAzure ADユーザーがデータベースに接続できるように、Oracle Database管理者およびMicrosoft Azure AD管理者がロールを再生します。

一般プロセスは次のとおりです。

1. Oracle Database管理者は、Oracle Database環境がMicrosoft Azure AD統合の要件を満たしていることを確認します。「Microsoft Azure AD統合のためのOracle Databaseの要件」を参照してください。
2. Azure AD管理者はデータベースのAzure ADアプリケーション登録を作成し、Oracle Database管理者はデータベース・アクセス用のAzure ADトークンを使用できるようにデータベースを有効にします。

   アプリケーション登録プロセスの一環として、Azure AD管理者は、Azureユーザー、グループおよびアプリケーションとOracle Databaseスキーマおよびロールの間のマッピングに使用するAzureアプリケーション・ロールを作成します。

3. Oracle Database管理者は、グローバル・スキーマを作成して、Azure ADユーザー(排他的スキーマ・マッピング)またはAzureアプリケーション・ロール(共有スキーマ・マッピング)のいずれかにマップします。Azure ADユーザーまたはアプリケーションは、1つのスキーマにマップする必要があります。
4. オプションで、Oracle管理者はグローバルOracle Databaseロールを作成し、Azureアプリケーション・ロールにマップします。
5. Oracle Databaseインスタンスに接続するAzure ADエンド・ユーザーは、クライアント・アプリケーションをAzure ADクライアントとして登録します(Oracleデータベースを登録する方法と同様です)。

   アプリケーション・クライアントがパブリックでないかぎり、Azure ADクライアントにはクライアント識別およびクライアント・シークレットがあります。アプリケーション・クライアントがパブリックの場合は、アプリケーション・クライアント識別のみが必要です。

6. Azure ADエンド・ユーザー(データベース管理者でもかまいません)は、PowerShellやAzureコマンドライン・インタフェースなどのユーティリティを使用して接続し、OAuth2データベース・アクセス・トークンを取得して、ローカル・ファイル・ディレクトリに格納します。アプリケーションは、Azure ADから直接Azure AD OAuth2アクセス・トークンをリクエストし、データベース・クライアントAPIを介して渡すこともできます。Azure AD OAuth2トークンの受渡しの詳細は、次のOracle Databaseクライアント・ドキュメントを参照してください。
   - JDBCシン・クライアント: 『Oracle Database JDBC開発者ガイド』
   - Oracle Call Interface (OCI): 『Oracle Call Interfaceプログラマーズ・ガイド』
   - Oracle Data Provider for .NET (ODP): 『Oracle Data Provider for .NET開発者ガイドfor Microsoft Windows』のOracle Databaseへの接続に関する項
7. Oracle Databaseインスタンスに接続すると、Azure ADエンド・ユーザーは必要に応じてデータベース操作を実行します。

親トピック: Oracle DatabaseとMicrosoft Azure ADの統合の概要

8.2 Microsoft Azure AD統合のためのOracle Databaseの構成

Oracle DatabaseインスタンスとのMicrosoft Azure AD統合では、データベースをAzure ADに登録する必要があります。

- Microsoft Azure AD統合のためのOracle Databaseの要件
  Microsoft Azure ADを使用してOracle Databaseインスタンスを構成するには、その前に環境が特別な要件を満たしていることを確認する必要があります。

- Oracle DatabaseインスタンスのMicrosoft Azure ADテナンシへの登録
  Azure AD管理者権限を持つユーザーは、Microsoft Azure ADを使用して、Oracle Databaseインスタンスを Microsoft Azure ADテナンシに登録します。
- Microsoft Azure AD v2アクセス・トークンの有効化
  Microsoft Azure AD v2アクセス・トークンを有効にするには、Azureポータルのupn属性を使用するように構成する必要があります。
- Microsoft Azure ADでのアプリケーション・ロールの管理
  Azure ADでは、Azure ADユーザーおよびグループに割り当てられ、Oracle Databaseのグローバル・スキーマおよびロールにもマップされるアプリケーション・ロールを作成および管理できます。
- Oracle DatabaseでのAzure AD外部認証の有効化
  Oracle DatabaseでのMicrosoft Azure AD外部認証を有効にする必要があります。
- Oracle DatabaseのAzure AD外部認証の無効化
  Oracle DatabaseインスタンスのAzure AD外部認証を無効にするには、ALTER SYSTEM文を使用する必要があります。

親トピック: Oracle DatabaseのMicrosoft Azure Active Directoryユーザーの認証および認可

8.2.1 Microsoft Azure AD統合のためのOracle Databaseの要件

Microsoft Azure ADを使用してOracle Databaseインスタンスを構成するには、その前に環境が特別な要件を満たしていることを確認する必要があります。

オンプレミスの非クラウドOracleデータベースの場合は、このドキュメントのステップに従います。Oracleデータベースが次のいずれかのDBaaSプラットフォーム上にある場合は、追加の要件についてプラットフォームのドキュメントを参照してください。

- Oracle Autonomous Database Serverlessの使用
- Oracle Autonomous Database on Dedicated Exadata Infrastructureeの使用
- Base Database ServiceでのAzure Active Directory認証の使用
- Oracle DatabaseのMicrosoft Azure Active Directoryユーザーの認証および認可

次のことに注意してください。

- Oracle Databaseサーバーは、Azure AD公開キーをリクエストできる必要があります。エンタープライズ・ネットワーク接続の設定によっては、プロキシ設定の構成が必要になる場合があります。
- Azure ADトークンをリクエストする必要があるユーザーやアプリケーションは、Azure ADへのネットワーク接続も可能である必要があります。接続のプロキシ設定を構成する必要がある場合があります。
- トークンを安全に転送するため、Oracle DatabaseクライアントとOracle Databaseサーバーの間にTransport Layer Security (TLS)を構成する必要があります。このTLS接続は、一方向または相互のいずれかです。
- 自己署名または既知の認証局によって署名されるTLSサーバー証明書を作成できます。既知の認証局(CA)によって署名された証明書を使用する利点は、データベース・クライアントが、ルート証明書を使用してローカル・ウォレットを作成および保持するのではなく、システムのデフォルト証明書ストアを使用してOracle Databaseサーバー証明書を検証できることです。これはLinuxおよびWindowsクライアントにのみ適用されることに注意してください。

関連トピック

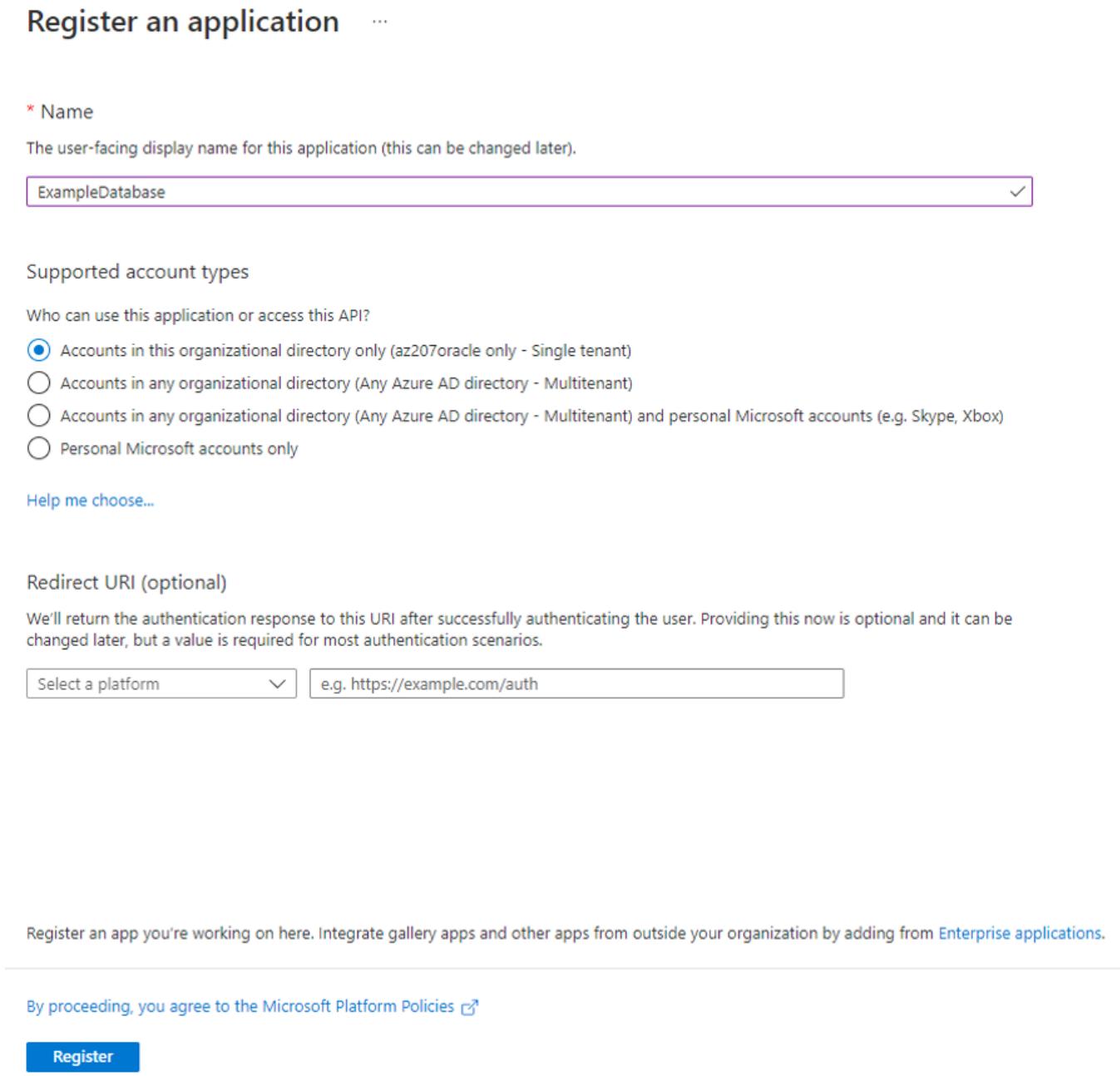
- クライアント・ウォレットを使用しないTransport Layer Security接続

親トピック: Microsoft Azure AD統合のためのOracle Databaseの構成

8.2.2 Oracle DatabaseインスタンスのMicrosoft Azure ADテナンシへの登録

Azure AD管理者権限を持つユーザーは、Microsoft Azure ADを使用して、Oracle Databaseインスタンスを Microsoft Azure ADテナンシに登録します。

1. アプリケーションを登録するためのMicrosoft Azure AD権限を持つ管理者としてAzureポータルにログインします。
2. 「Azure Active directory admin center」ページの左側のナビゲーション・バーで、「Azure Active Directory」を選択します。
3. 「MS - App registrations」ページの左側のナビゲーション・バーで、「App registrations」を選択します。
4. 「New registration」を選択します。
   「Register an application」ウィンドウが表示されます。



5. 「Register an application」ページで、次のOracle Databaseインスタンス登録情報を入力します。
   - 「Name」フィールドに、Oracle Databaseインスタンス接続の名前(`Example Database`など)を入力します。
   - 「Supported account types」で、ユースケースに合ったアカウント・タイプを選択します。
     - Accounts in this organizational directory only (tenant_name only - Single

tenant)

- Accounts in any organizational directory (Any Azure AD directory - Multitenant)
- Accounts in any organizational directory (Any Azure AD directory - Multitenant) and personal Microsoft accounts (e.g. Skype, Xbox)
- Personal Microsoft accounts only

6. 「Redirect URI」(オプション)設定は省略します。Azure ADにはデータベース・サーバー用のリダイレクトURIが不要なため、これを作成する必要はありません。
7. 「Register」をクリックします。
   「Register」をクリックすると、Azure ADにアプリの登録の「Overview」ペインが表示され、「Essentials」の下にアプリケーション(クライアント) IDが表示されます。この値は、Microsoft IDプラットフォーム内のアプリケーションの一意の識別子です。アプリケーションという用語はOracle Databaseインスタンスを指します。
8. データベース・アプリケーション登録のスコープを登録します。
   スコープとは、データベースにアクセスする権限です。各データベースには、データベース・スコープを使用する権限のリクエストによって、クライアントがデータベースとの信頼を確立できるようにするスコープが必要です。これにより、データベース・クライアントはデータベースのアクセス・トークンを取得できます。
   - 左側のナビゲーション・バーで、「Expose an API」を選択します。
   - 「Set the App ID URI」の下の「Application ID URI」フィールドに、次の書式を使用してデータベース接続のアプリケーションID URIを入力し、「Save」をクリックします。

     ```
     your_tenancy_url/application_(client)_id
     ```

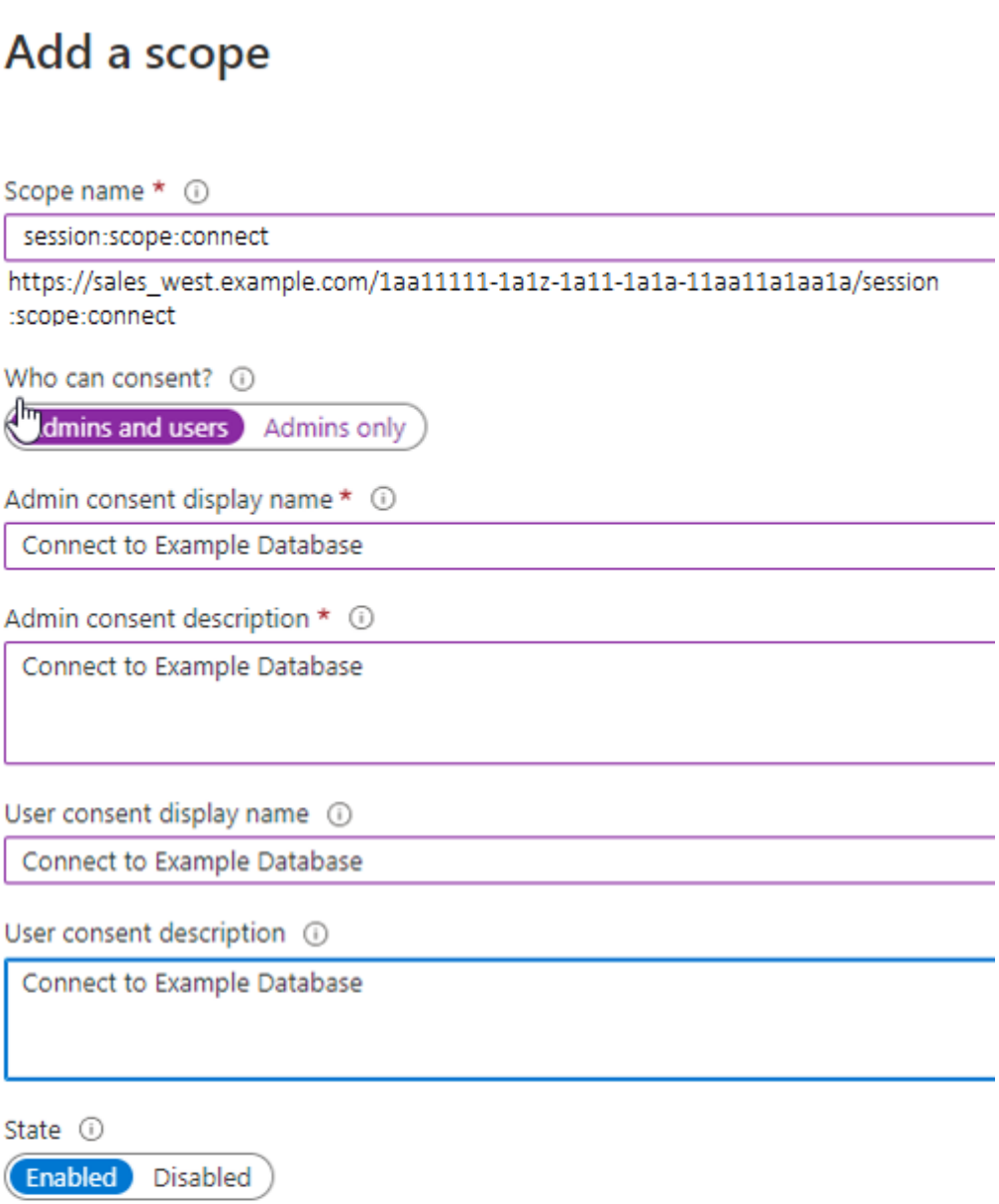
     詳細は、次のとおりです。

     - `your_tenancy_url`には、Azure ADテナンシの接頭辞および完全修飾ドメイン名として`https`を含める必要があります。
     - `application_(client)_id`は、Oracle Databaseインスタンスを Azure ADに登録したときに生成されたIDです。これは、アプリの登録の「Overview」ペインに表示されます。

     例:

     ```
     https://sales_west.example.com/1aa11111-1a1z-1a11-1a1a-11aa11a1aa1a
     ```

   - 「Add a scope」を選択し、次の設定を入力します。



- 「Scope name」には、スコープの名前を指定します。次の名前を入力します。

  ```
  session:scope:connect
  ```

  この名前は任意のテキストにできます。ただし、スコープ名の入力は必須です。このスコープ名は、後でデータベース・クライアント・アプリケーションにデータベースへのアクセスを承認するときに使用すること

が必要になります。

- 「Who can consent」には、必要な権限を指定します。「Admins and users」を選択するか、より制限を厳しくする場合は「Admins only」を選択します。
- 「Admin consent」表示名には、管理者のみが表示できるスコープの目的(`Connect to Oracle`など)を入力します。
- 「Admin consent display name」には、管理者のみが表示できるスコープの目的(`Connect to Example Database`など)を入力します。
- 「User consent display name」は、スコープの目的の簡単な説明です(たとえば、`Connect to Example Database`)。これは、ユーザーが「Who can consent」で「Admins and users」を指定した場合に表示されます。
- 「User consent description」は、スコープの目的の詳細な説明です(たとえば、`Connect to Example Database`)。これは、ユーザーが「Who can consent」で「Admins and users」を指定した場合に表示されます。
- 「State」は、接続を有効または無効にします。「Enabled」を選択します。

これらのステップを完了すると、1つ以上のAzureアプリケーション・ロールを追加し、Oracleスキーマおよびロールのマッピングを実行する準備が整います。

関連トピック

- Quickstart: Register an application with the Microsoft identity platform

親トピック: Microsoft Azure AD統合のためのOracle Databaseの構成

8.2.3 Microsoft Azure AD v2アクセス・トークンの有効化

Microsoft Azure AD v2アクセス・トークンを有効にするには、Azureポータルのupn属性を使用するように構成する必要があります。

Azure AD v2アクセス・トークンは、Autonomous Database Serverlessでのみサポートされており、組織アカウント(Azure AD)と個人用Microsoftアカウント(MSA)の両方の認証を含み、v1トークンよりも幅広いアクセス・シナリオをサポートしています。このトークンは、アプリケーション登録(プレビュー)エクスペリエンスを使用してAzureポータルに登録されているアプリケーションで使用できます。

1. 使用しているAzure ADアクセス・トークンのバージョンを確認します。
2. Microsoft Azureポータルにログインします。
3. 「Azure Active Directory」を検索して選択します。
4. 「Manage」で、「App registrations」を選択します。
5. シナリオおよび目的の結果に基づいてオプションの要求を構成するアプリケーションを選択します。
6. 「Manage」で、「Token configuration」を選択します。
7. 「Add optional claim」をクリックし、「upn」を選択します。

関連トピック

- Azure ADアクセス・トークンのバージョンの確認

親トピック: Microsoft Azure AD統合のためのOracle Databaseの構成

8.2.4 Microsoft Azure ADでのアプリケーション・ロールの管理

Azure ADでは、Azure ADユーザーおよびグループに割り当てられ、Oracle Databaseのグローバル・スキーマおよびロールにもマップされるアプリケーション・ロールを作成および管理できます。

- Microsoft Azure ADアプリケーション・ロールの作成
  データベースに接続する必要のあるAzure ADユーザー、グループおよびアプリケーションは、データベース・アプリケーション・ロールに割り当てられます。
- Microsoft Azure ADアプリケーション・ロールへのユーザーおよびグループの割当て
  Microsoft Azure ADユーザーがOracleデータベースにアクセスするためには、その前に、Oracle Databaseスキーマ・ユーザーまたはロールにマップされるアプリケーション・ロールにMicrosoft Azure ADユーザーを割り当てておく必要があります。
- アプリケーション・ロールへのアプリケーションの割当て
  クライアント資格証明フローを使用してデータベースに接続する必要のあるアプリケーションは、アプリケーション・ロールに割り当てる必要があります。

親トピック: Microsoft Azure AD統合のためのOracle Databaseの構成

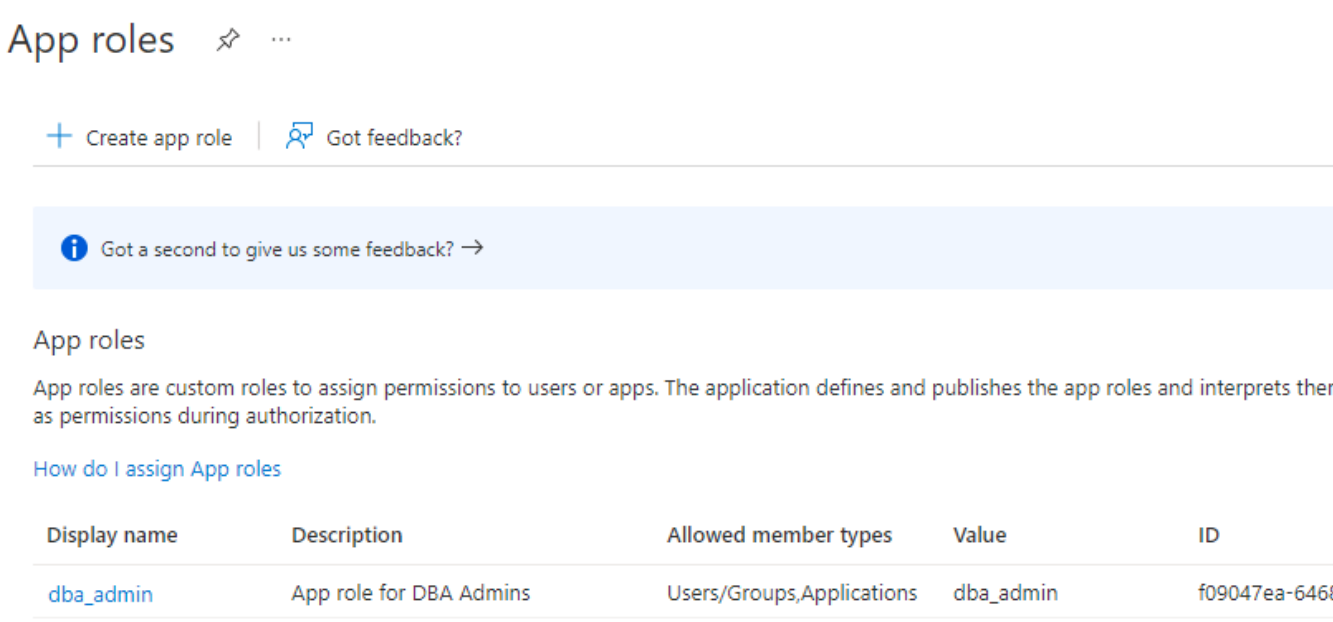
8.2.4.1 Microsoft Azure ADアプリケーション・ロールの作成

データベースに接続する必要のあるAzure ADユーザー、グループおよびアプリケーションは、データベース・アプリケーション・ロールに割り当てられます。

アプリケーション・ロールの作成方法の詳細は、Microsoft Azureの記事『Create and assign a custom role in Azure Active Directory』を参照してください。次のステップでは、Oracleデータベースで使用するアプリケーション・ロールを作成する方法について説明します。

1. アプリケーション・ロールを作成する権限を持つ管理者としてAzure ADにログインします。
2. 作成したOracle Databaseアプリ登録にアクセスします。
   a. 「Directory + subscription」フィルタを使用して、Oracle Databaseアプリ登録を含むAzure Active Directoryテナントを見つけます。
   b. 「Azure Active Directory」を選択します。
   c. 「Manage」で、「App registrations」を選択し、以前に登録したOracle Databaseインスタンスを選択します。
3. 「Manage」で、「App roles」を選択します。
4. 「App roles」ページで、「Create app role」を選択します。
5. 「Create app role」ページで、次の情報を入力します。
   a. 「Display name」は、ロールの表示名です(`HR App Schema`など)。この名前には空白を含めることができます。
   b. 「Value」は、ロールの実際の名前です(`HR_APP`など)。この設定は、スキーマまたはロールへのデータベース・マッピングで参照される文字列と正確に一致させます。この名前にはスペースを含めません。
   c. 「Description」では、このロールの目的を説明します。
   d. 「Do you want to enable this app role?」では、ロールをアクティブ化できます。
6. 「適用」をクリックします。

   アプリのロールが「App roles」ペインに表示されます。



親トピック: Microsoft Azure ADでのアプリケーション・ロールの管理

8.2.4.2 Microsoft Azure ADアプリケーション・ロールへのユーザーおよびグループの割当て

Microsoft Azure ADユーザーがOracleデータベースにアクセスするためには、その前に、Oracle Databaseスキーマ・ユーザーまたはロールにマップされるアプリケーション・ロールにMicrosoft Azure ADユーザーを割り当てておく必要があります。

ユーザーおよびグループをアプリケーション・ロールに割り当てる手順の詳細は、Microsoft Azureの記事『Add app roles to your application and receive them in the token』を参照してください。次のステップでは、Oracleデータベース用にこれを実行する方法について説明します。

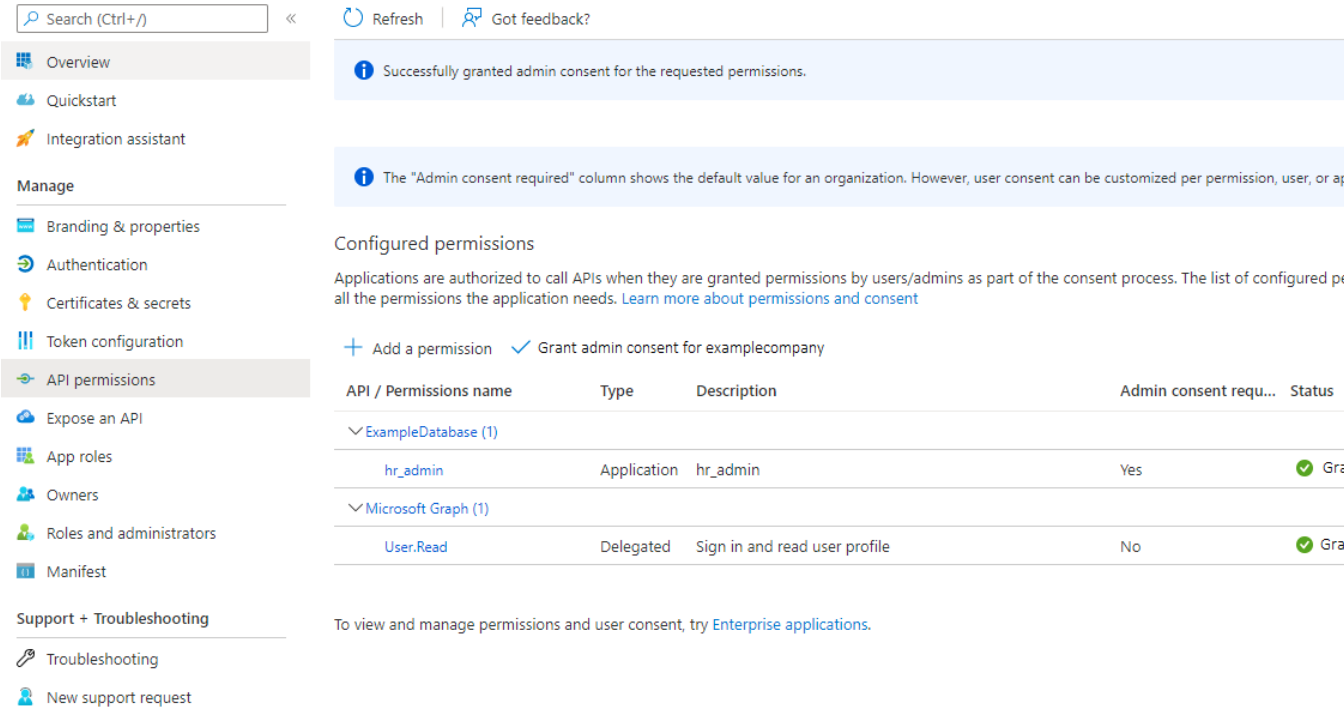
1. Azure ADユーザーおよびグループをアプリケーション・ロールに割り当てる権限を持つ管理者としてAzure ADにログインします。
2. エンタープライズ・アプリケーションで、作成したOracle Databaseアプリケーション登録の名前を検索します。これは、アプリケーション登録の作成時に自動的に作成されます。
    a. 「Directory + subscription」フィルタを使用して、Oracle接続を含むAzure Active Directoryテナントを見つけます。
    b. 「Azure Active Directory」を選択します。
    c. 「Manage」で、「Enterprise applications」を選択し、以前に登録したOracle Databaseアプリケーション登録名を選択します。
3. 「Getting Started」で「Assign users and groups」を選択します。
4. 「Add user/group」を選択します。
5. 「Add assignment」ウィンドウで、「Users and groups」を選択して、ユーザーおよびセキュリティ・グループのリストを表示します。
6. このリストから、アプリケーション・ロールに割り当てるユーザーおよびグループを選択し、「Select」をクリックします。
7. 「Add assignment」ウィンドウで、「Select a role」を選択して、作成したアプリケーション・ロールのリストを表示します。
8. アプリケーション・ロールを選択し、「Select」を選択します。
9. 「Assign」をクリックします。

親トピック: Microsoft Azure ADでのアプリケーション・ロールの管理

8.2.4.3 アプリケーション・ロールへのアプリケーションの割当て

クライアント資格証明フローを使用してデータベースに接続する必要のあるアプリケーションは、アプリケーション・ロールに割り当てる必要があります。

1. Azure ADユーザーおよびグループをアプリケーション・ロールに割り当てる権限を持つ管理者としてAzure ADにログインします。
2. アプリケーションのアプリ登録にアクセスします。
3. 「Manage」で、「API permissions」を選択します。
4. 「Configured permissions」領域で、「+ Add a permission」を選択します。
5. 「Request API permission」ペインで、「My APIs」タブを選択します。
6. このアプリケーションにアクセスを許可する対象のOracle Databaseアプリケーションを選択します。次に、「Application permissions」オプションを選択します。
7. アプリケーションに割り当てるデータベース・アプリケーション・ロールを選択し、画面下部の「Add Permission」ボックスをクリックしてアプリケーション・ロールを割り当て、ダイアログ・ボックスを閉じます。割り当てたアプリケーション・ロールが「Configured permissions」の下に表示されることを確認します。



8. 「Grant admin consent for tenancy」を選択してテナンシ・ユーザーに承認を与え、確認ダイアログ・ボックスで「Yes」を選択します。

関連トピック

- Configure the admin consent workflow

親トピック: Microsoft Azure ADでのアプリケーション・ロールの管理

8.2.5 Oracle Databaseに対するAzure AD外部認証の有効化

Oracle DatabaseでのMicrosoft Azure AD外部認証を有効にする必要があります。

使用しているプラットフォームのAzure AD認証の詳細は、次のドキュメントのリンクを参照してください。

1. `ALTER SYSTEM`システム権限が付与されたユーザーとして、Oracle Databaseインスタンスにログインします。
2. `IDENTITY_PROVIDER_TYPE`パラメータを次のように設定します。

```
ALTER SYSTEM SET IDENTITY_PROVIDER_TYPE=AZURE_AD SCOPE=BOTH;
```

3. IDENTITY_PROVIDER_TYPEパラメータが正しく設定されていることを確認します。

```
SELECT NAME, VALUE FROM V$PARAMETER WHERE NAME='identity_provider_type';
```

次の出力が表示されます。

```
NAME                    VALUE
----------------------  -------
identity_provider_type  AZURE_AD
```

4. 次の構文を使用して、IDENTITY_PROVIDER_CONFIGパラメータを設定します。

```
ALTER SYSTEM SET IDENTITY_PROVIDER_CONFIG =
{
   application_id_uri : string , // from registered app, to be mapped in jwt
"aud" claim;
                                 // Domain qualified to support cross tenancy
resource access
   tenant_id : string,           // from tenant config
   app_id: string                // from registered resource app
}SCOPE=BOTH;
```

例:

```
ALTER SYSTEM SET IDENTITY_PROVIDER_CONFIG =
{
  "application_id_uri" : "https://www.example.com/11aa1a11-aaaa-1111-1111-
1111aa11111",
  "tenant_id" : "111a1111-a11a-111a-1a1a-1111111111a",
  "app_id" : "11aa1a11-aaaa-1111-1111-1111aa11111"
}SCOPE=BOTH;
```

Azure AD外部認証のためのOracle Databaseの有効化の詳細は、オンプレミス(非クラウド)のOracleデータベースに関してこのドキュメントで説明する情報に加え、次のプラットフォーム固有のドキュメントを参照してください。

- Oracle Autonomous Database Serverlessの使用
- Oracle Autonomous Database on Dedicated Exadata Infrastructure

親トピック: Microsoft Azure AD統合のためのOracle Databaseの構成

8.2.6 Oracle Databaseに対するAzure AD外部認証の無効化

Oracle DatabaseインスタンスのAzure AD外部認証を無効にするには、ALTER SYSTEM文を使用する必要があります。

Oracle Databaseに加えて、この手順はOracle Autonomous Database on Dedicated Exadata InfrastructureおよびOracle Exadata Cloud Service (Oracle ExaCS)にも使用できます。これらの製品でAzure AD外部認証を無効にする場合は、製品ドキュメントを参照してください。

Oracle Autonomous Database on Shared Exadata InfrastructureからAzure ADを無効にするには、Oracle Autonomous Database Serverlessの使用を参照してください。次の手順は、他のすべてのプラットフォームに適用されます。

1. ALTER SYSTEMシステム権限が付与されたユーザーとして、Oracle Databaseインスタンスにログインします。
2. アイデンティティ・プロバイダのパラメータを次のように設定します。

```
ALTER SYSTEM RESET IDENTITY_PROVIDER_CONFIG SCOPE=BOTH;
ALTER SYSTEM RESET IDENTITY_PROVIDER_TYPE SCOPE=BOTH;
```

親トピック: Microsoft Azure AD統合のためのOracle Databaseの構成

8.3 Oracle Databaseスキーマおよびロールのマッピング

Azure ADユーザーは、1つのデータベース・スキーマにマップされ、オプションで1つ以上のデータベース・ロールにマップされます。

- Oracle DatabaseスキーマのMicrosoft Azure ADユーザーへの排他的マッピング
  Oracle DatabaseスキーマをMicrosoft Azure ADユーザーに排他的にマップできます。
- 共有Oracleスキーマのアプリケーション・ロールへのマッピング
  このマッピングでは、Oracleスキーマがアプリケーション・ロールにマップされます。したがって、そのアプリケーション・ロールを持つすべてのユーザーが同じ共有スキーマを取得します。
- アプリケーション・ロールへのOracle Databaseグローバル・ロールのマッピング
  Azureアプリケーション・ロールにマップされるOracle Databaseグローバル・ロールは、Azureユーザーおよびアプリケーションに、ログイン・スキーマを介して付与されている権限およびロールに加え、さらに権限およびロールを付与します。

親トピック: Oracle DatabaseのMicrosoft Azure Active Directoryユーザーの認証および認可

8.3.1 Oracle DatabaseスキーマのMicrosoft Azure ADユーザーへの排他的マッピング

Oracle DatabaseスキーマをMicrosoft Azure ADユーザーに排他的にマップできます。

1. CREATE USERまたはALTER USERシステム権限を付与されたユーザーとして、Oracle Databaseインスタンスにログインします。
2. Azure ADユーザー名を指定するIDENTIFIED GLOBALLY AS句を使用してCREATE USERまたはALTER USER文を実行します。
   たとえば、peter_fitchという名前の新しいデータベース・スキーマ・ユーザーを作成し、このユーザーをpeter.fitch@example.comという名前の既存のAzure ADユーザーにマップするには、次のようにします。

```
CREATE USER peter_fitch IDENTIFIED GLOBALLY AS
'AZURE_USER=peter.fitch@example.com';
```

3. CREATE SESSION権限をユーザーに付与します。

```
GRANT CREATE SESSION TO peter_fitch;
```

親トピック: Oracle Databaseスキーマおよびロールのマッピング

8.3.2 共有Oracleスキーマのアプリケーション・ロールへのマッピング

このマッピングでは、Oracleスキーマがアプリケーション・ロールにマップされます。したがって、そのアプリケーション・ロールを持つすべてのユーザーが同じ共有スキーマを取得します。

1. CREATE USERまたはALTER USERシステム権限を持つユーザーとして、Oracle Databaseインスタンスにログインします。
2. Azureアプリケーション・ロール名を指定するIDENTIFIED GLOBALLY AS句を使用してCREATE USERまたはALTER USER文を実行します。
   たとえば、dba_azureという名前の新しいデータベース・グローバル・ユーザー・アカウント(スキーマ)を作成し、それをAZURE_DBAという名前の既存のAzure ADアプリケーション・ロールにマップするには、次のようにします。

```
CREATE USER dba_azure IDENTIFIED GLOBALLY AS 'AZURE_ROLE=AZURE_DBA';
```

親トピック: Oracle Databaseスキーマおよびロールのマッピング

8.3.3 アプリケーション・ロールへのOracle Databaseグローバル・ロールのマッピング

AzureアプリにマップされたOracle Databaseグローバル・ロールは、Azureユーザーおよびアプリケーションに、ログイン・スキーマを介して付与されている権限およびロールに加え、さらに権限およびロールを付与します。

1. `CREATE ROLE`または`ALTER ROLE`システム権限が付与されたユーザーとして、Oracle Databaseインスタンスにログインします
2. Azure ADアプリケーション・ロールの名前を指定する`IDENTIFIED GLOBALLY AS`句を使用して`CREATE ROLE`または`ALTER ROLE`文を実行します。
   たとえば、`widget_sales_role`という名前の新しいデータベース・グローバル・ロールを作成し、それを`WidgetManagerGroup`という名前の既存のAzure ADアプリケーション・ロールにマップするには、次のようにします。

```
CREATE ROLE widget_sales_role IDENTIFIED GLOBALLY AS
'AZURE_ROLE=WidgetManagerGroup';
```

親トピック: Oracle Databaseスキーマおよびロールのマッピング

8.4 Oracle DatabaseへのAzure ADクライアント接続の構成

Azure AD登録済データベースに接続するようにクライアント接続を構成できます

- Azure ADへのクライアント接続の構成について
  Azure ADトークンを使用してOracle Databaseインスタンスに接続するようにクライアントを構成するには、様々な方法があります。
- Azure AD接続でサポートされるクライアント・ドライバ
  Oracle Databaseは、Azure AD接続用の複数のタイプのクライアント・ドライバをサポートしています。
- PowerShellでのSQL*PlusクライアントからOracle Databaseへの接続の操作フロー
  Azureユーザー、Azure ADおよびOracleデータベースの間の接続では、これらのコンポーネントを通じた`OAuth2`トークンの受渡しが利用されます。
- Azure ADアプリ登録によるクライアントの登録
  このタイプの登録は、Azure ADアプリ登録によるOracle Databaseの登録と同様に行われます。
- Azure AD OAuth2トークンの取得の例
  これらの例は、Azure AD `OAuth2`トークンを取得する様々な方法を示しています。
- Azure ADアクセス・トークン用のSQL*Plusの構成
  特定の場所からAzure ADデータベース・アクセス・トークンを取得し、/スラッシュ・ログインが使用されるときにそのトークンを使用するように、SQL*Plusを構成する必要があります。
- データベースがインターネットに接続するためのネットワーク・プロキシの作成
  このネットワーク・プロキシを使用すると、OracleデータベースはAzure ADエンドポイントに到達できるようになります。
- クライアントによるAzureトークンの直接取得の有効化
  クライアント自身によるAzureトークンの直接取得を有効にするパラメータを設定できます。

親トピック: Oracle DatabaseのMicrosoft Azure Active Directoryユーザーの認証および認可

8.4.1 Azure ADへのクライアント接続の構成について

Azure ADトークンを使用してOracle Databaseインスタンスに接続するようにクライアントを構成するには、様々な方法があります。

環境に最も適したクライアント接続方法を選択してください。このガイドでは、Azure AD OAuth2アクセス・トークンを取得する様々な方法を使用してSQL*Plusを接続する例を紹介します。すべてのOracle Databaseリリース19cクライアントで、ファイルとして渡されるトークンを受け入れることができます。JDBC Thin、Instant ClientおよびODP.netドライバは、データベース・クライアントAPIを介してアプリケーションからのトークンも受け入れます。SQL*PlusなどのOracle Databaseツールではトークンを直接取得できないため、PowerShellやAzure CLIなどのツールを使用してAzure AD OAuth2アクセス・トークンを取得する必要があります。Azure ADトークンを取得するには、Azure ADアプリ登録プロセスによってクライアントを登録する必要があります。このクライアントの登録は、アプリ登録を使用したOracle DatabaseサーバーのAzure ADへの登録と同様に行われます。データベースとクライアントの両方をAzure ADに登録する必要があります。

データベースを登録する必要があるのは、クライアントがデータベースのアクセス・トークンを取得する権限を取得できるようにするためです。クライアントを登録する必要があるのは、Azure ADが信頼できるクライアントがアクセス・トークンをリクエストしていることを認識できるようにするためです。

クライアントをAzure ADに接続する方法の詳細は、次のMicrosoft Azureの記事を参照してください。

- 『Quickstart: Configure a client application to access a web API』
- 『Choose the right Azure command-line tool』
- 『Get Azure AD tokens by using the Microsoft Authentication Library』
- 『Install the Azure CLI on Linux』

親トピック: Oracle DatabaseへのAzure ADクライアント接続の構成

8.4.2 Azure AD接続でサポートされるクライアント・ドライバ

Oracle Databaseは、Azure AD接続用の複数のタイプのクライアント・ドライバをサポートしています。

- JDBC Thin: Oracle Database 19.16 (2022年7月)、Oracle Database 21.8 (2022年10月)
- OCI (Cドライバ): Oracle Database 19.16 (2022年7月)
- OCIベースのOracle Instant Client
- Oracle Data Provider (コア): Oracle Database 19.16、Oracle Database 21.7
- Oracle Data Provider (管理対象外): OCIベース
- Oracle Data Provider (管理対象): Oracle Database 19.16、Oracle Database 21.7
- OCIベースのその他のドライバは、OCI互換性を採用しています

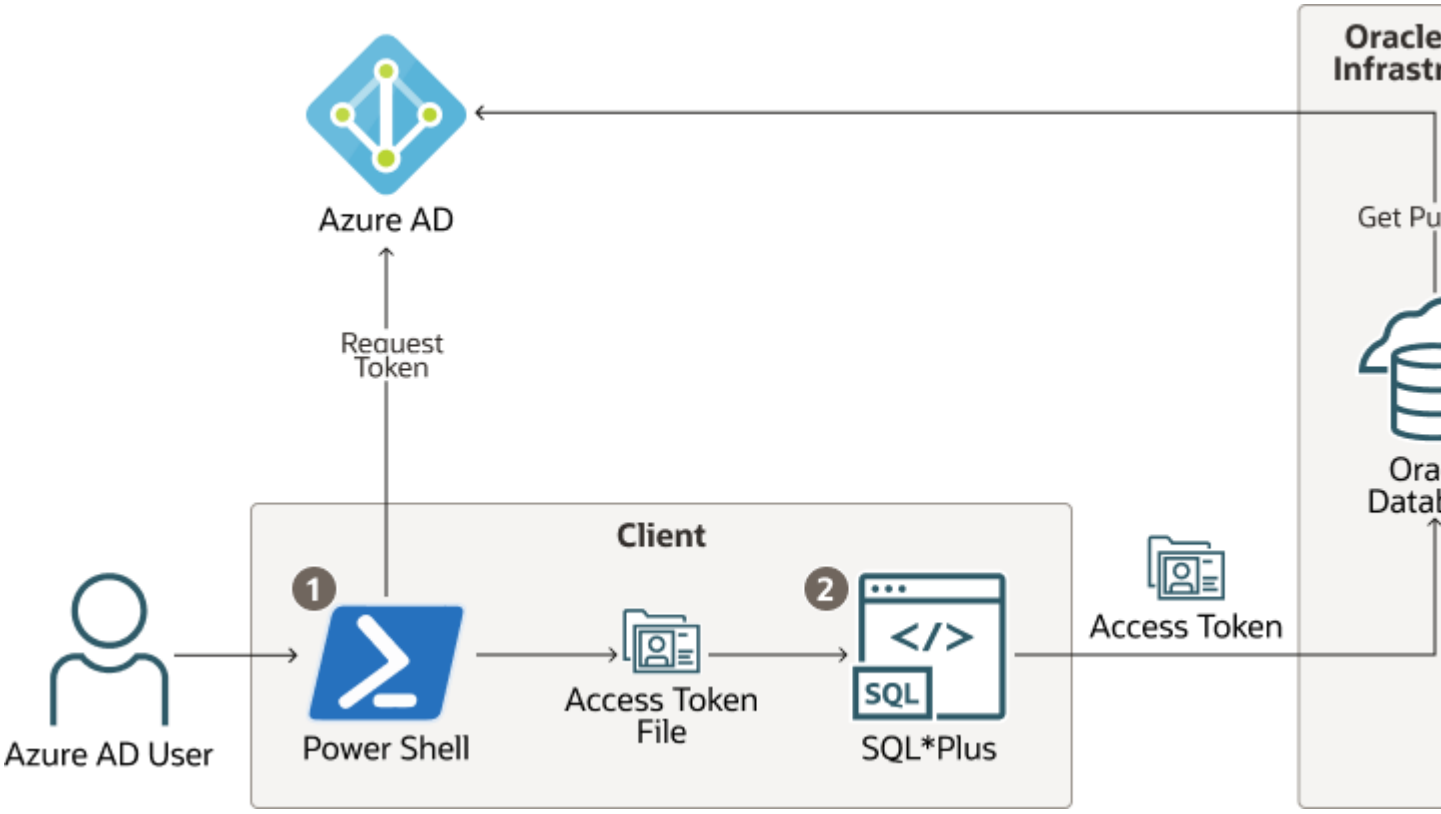
親トピック: Oracle DatabaseへのAzure ADクライアント接続の構成

8.4.3 PowerShellでのSQL*PlusクライアントからOracle Databaseへの接続の操作フロー

Azureユーザー、Azure ADおよびOracleデータベースの間の接続では、これらのコンポーネントを通じたOAuth2トークンの受渡しが利用されます。

この例では、パブリック・クライアントでのリソース所有者パスワード資格証明(ROPC)フローの使用を示します。ROPCの詳細は、Microsoft Azureの記事『Microsoft identity platform and OAuth 2.0 Resource Owner Password Credentials』を参照してください。

図8-3 パブリック・クライアントを使用したROPC操作フロー



1. AzureユーザーがPowerShellでデータベースのAzure ADアクセス・トークンをリクエストし、返されたトークンが、特定のファイルの場所で`token`というファイルに書き込まれます。
2. Azureユーザーは、/スラッシュ・ログインを使用してデータベースに接続します。`sqlnet.ora`または`tnsnames.ora`のいずれかの接続文字列からInstant Clientに、Azure AD `OAuth2`トークンが必要であり、指定されたファイルの場所からこのトークンを取得するよう指示されます。アクセス・トークンがデータベースに送信されます。
3. データベースは、(Azure AD公開キーを使用して)アクセス・トークンがAzure ADから取得されたことを確認し、トークンに追加のクレームがないかチェックします。
4. データベースがスキーマ・マッピング(排他的または共有)を検出し、セッションを作成します。データベースは、Azureユーザーにアプリケーション・ロールによっても割り当てられている、すべてのグローバル・ロールも付与します。

親トピック: Oracle DatabaseへのAzure ADクライアント接続の構成

8.4.4 Azure ADアプリ登録によるクライアントの登録

このタイプの登録は、Azure ADアプリ登録によるOracle Databaseの登録と同様に行われます。

- 機密およびパブリック・クライアント登録
  ユースケースに応じて、データベース・クライアントを機密またはパブリックとしてAzureに登録できます。
- Azure ADへのデータベース・クライアント・アプリケーションの登録
  クライアント・アプリ登録の作成は、Microsoft Azure ADテナンシを使用してOracle Databaseインスタンスを作成する場合と同様に行われます。

親トピック: Oracle DatabaseへのAzure ADクライアント接続の構成

8.4.4.1 機密およびパブリック・クライアント登録

ユースケースに応じて、データベース・クライアントを機密またはパブリックとしてAzureに登録できます。

認証フローおよびアプリケーション・シナリオの詳細は、Microsoft Azureの記事『Authentication flows and

application scenarios』を参照してください。

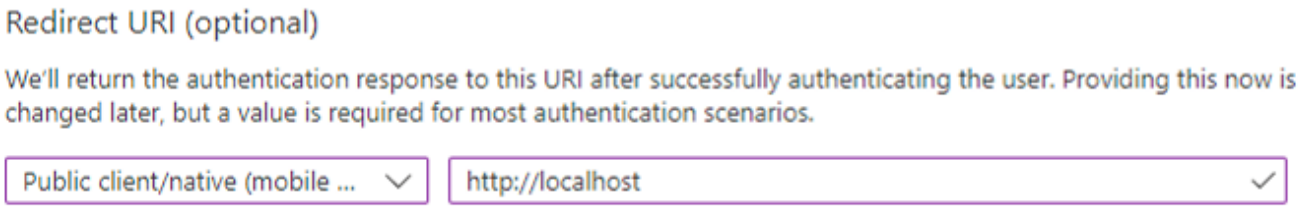
機密クライアント・アプリケーションを登録する場合、クライアントにはクライアントIDに加えてシークレットが必要です。機密クライアント・アプリケーションは、Azure ADリクエストを行うときにクライアントIDとシークレットの両方を使用します。ただし、企業においてSQL*PlusおよびSQLclのすべてのユーザーが独自のシークレットを持つ個別のアプリ登録を作成することは実用的ではありません。また、組織内でシークレットを共有し始めると、シークレットはもはやシークレットではなくなります。パブリック・クライアント・アプリケーションのみを作成するほうがはるかに有用です。パブリック・クライアント・アプリケーションはシークレットを持たず、クライアントIDのみを持ちます。すべてのデータベース・ツール・ユーザーは、Azure ADに接続するときにパブリック・クライアントIDを使用して、アクセス・トークンを取得できます。その場合もAzure ADユーザーは、自分のユーザー資格証明を使用してAzure ADに対して認証される必要があります。

親トピック: Azure ADアプリケーション登録によるクライアントの登録

8.4.4.2 Azure ADへのデータベース・クライアント・アプリケーションの登録

クライアント・アプリ登録の作成は、Microsoft Azure ADテナンシを使用してOracle Databaseインスタンスを作成する場合と同様に行われます。

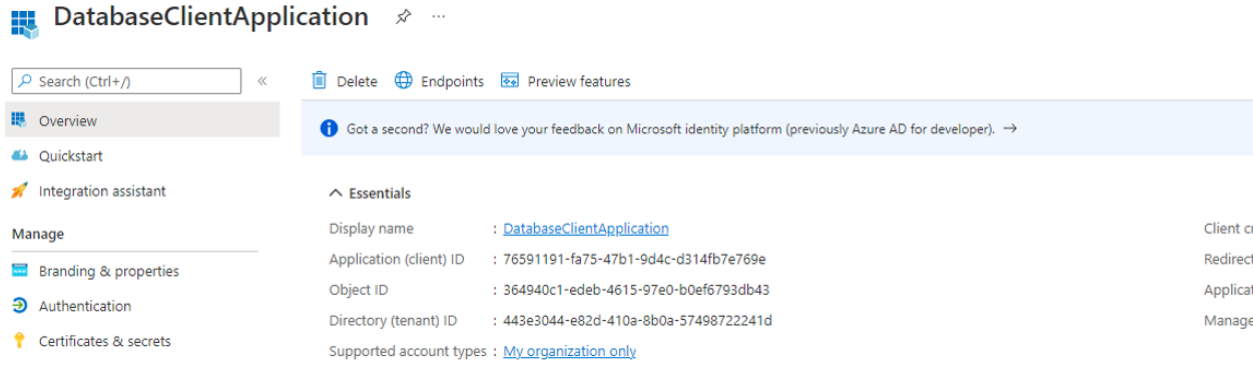
1. アプリケーションを登録するためのMicrosoft Azure AD権限を持つ管理者としてAzureポータルにログインします。
2. 「Azure Active directory admin center」ページの左側のナビゲーション・バーで、「Azure Active Directory」を選択します。
3. 「MS - App registrations」ページの左側のナビゲーション・バーで、「App registrations」を選択します。
4. 「New registration」を選択します。
5. 「Register an application」ページで、次のOracle Databaseインスタンス登録情報を入力します。
    - 「Name」フィールドに、クライアント・アプリケーションの名前(`DatabaseClientApplication`など)を入力します。
    - 「Supported account types」で、ユースケースに合ったアカウント・タイプを選択します。
        - Accounts in this organizational directory only (tenant_name only - Single tenant)
        - Accounts in any organizational directory (Any Azure AD directory - Multitenant)
        - Accounts in any organizational directory (Any Azure AD directory - Multitenant) and personal Microsoft accounts (e.g. Skype, Xbox)
        - Personal Microsoft accounts only
6. 「Redirect URI」(オプション)で、クライアント・アプリケーションのリダイレクトURIを構成します。



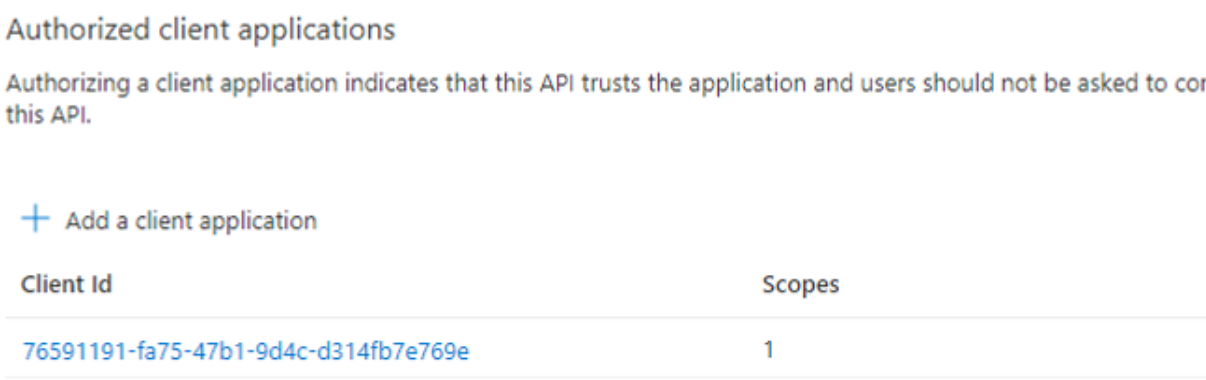
    - 「Public client/native (mobile & desktop)」、「Web」または「Single-page application (SPA)」を選択します。SQL*Plusを使用してOracle Databaseインスタンスにアクセスする必要があるデータベース管理者など、複数のユーザーがこのクライアント・アプリケーションを使用する場合は、「Public client」を選択します。
    - 別のアドレスを使用する場合を除き、リダイレクトURIとして`http://localhost`を追加します。このリダイ

レクトURIは、認可フローで必要です。

7. 「Register」をクリックします。
   これで、データベース・クライアントはAzure ADに登録されました。次は、Oracle Databaseインスタンスの認可クライアント・アプリケーションのリストに新しいクライアントを追加する必要があります。
8. このクライアント・アプリケーションのリストに新しいクライアントを追加するには、次の手順を実行します。
   - 新しいクライアントのアプリケーション(クライアント) IDをメモします。このIDはアプリケーションの「Overview」ページにあります。



   - 「App registrations」ページで、メニューからデータベース・サーバーを選択して、そのアプリの登録ページを開きます。
   - 左側で、「Expose an API」を選択します。
   - 「Authorized client applications」が表示されるまで、メイン・ページを下にスクロールします。
   - 「+」を選択して、クライアント・アプリケーションを追加します。
   - 新しいクライアントのアプリケーション(クライアント) IDを「Client Id」フィールドにコピーします。



   - 「Add application」をクリックします。

関連トピック

- Quickstart: Register an application with the Microsoft identity platform

親トピック: Azure ADアプリケーション登録によるクライアントの登録

8.4.5 Azure AD OAuth2トークンの取得の例

次の例では、Azure AD OAuth2トークンを取得するための様々な方法を示します。

- 例: PowerShellで、リソース所有者のパスワード資格証明を使用してトークンを取得する
  この例では、PowerShellで、リソース所有者のパスワード資格証明(ROPC)を使用してAzure ADアクセス・トークンを取得する方法を示します。
- 例: Microsoft Authentication LibraryでPythonを使用して、認可フローを使用する

Microsoft Authentication Library (MSAL)を使用したこの例はPythonで作成されているため、PowerShellやLinuxなどの様々なプラットフォームで実行できます。

- 例: リソース所有者パスワード資格証明フローでCurlを使用する
  この例では、パブリックAzure ADクライアントでリソース所有者パスワード資格証明(ROPC)フローを使用して、Azure AD APIに対してcurlコマンドを使用する方法を示します。
- 例: Azure CLIを使用した認可フロー
  この例では、Azure CLIを使用してアクセス・トークンを取得し、取得したトークンをファイルに書き込む方法を示します。

親トピック: Oracle DatabaseへのAzure ADクライアント接続の構成

8.4.5.1 例: PowerShellで、リソース所有者のパスワード資格証明を使用してトークンを取得する

この例では、PowerShellで、リソース所有者のパスワード資格証明(ROPC)を使用してAzure ADアクセス・トークンを取得する方法を示します。

OAuth2アクセス・トークンを取得するには、PowerShellからRESTコールを実行します。この構成には、Oracle Databaseインスタンスを Azure ADに登録したときに生成された、またはユーザーが指定した複数の値が必要です。

1. 必要に応じて、Azure Active Directory PowerShellモジュールをインストールします。
   Microsoftの記事『Install the Azure Az PowerShell module』の手順に従って、Azure PowerShellをダウンロードしてインストールします。インストールの実行には約20分以上かかります。Azure PowerShellのデバッグ・オプションを設定すると、インストールの進捗状況を確認できます。
2. Azure PowerShellのインストールが完了したら、PowerShellにログインし、次の変数を示されている順番で設定します。
   a. `$TenantDomain = "user_tenancy_domain_name"`
      この値はテナンシ・ドメイン名です。例:
      ```
      $TenantDomain = "example.com"
      ```
   b. `$AppClientId ="application_client_id"`
      この値は、データベース・サーバーではなく、データベース・クライアントのアプリケーション・クライアントIDを設定します。これが、アプリの登録の「Overview」ペインの「Application (client) ID」値です。例:
      ```
      $AppClientId ="111a1a1a-aa1a-1a1a-11aa-1a11111111aa"
      ```
   c. `$Username = "user_name"`
      この値は、Oracle Databaseインスタンスにアクセスするユーザーの名前です。例:
      ```
      $Username = "peter.fitch@example.com"
      ```
   d. PowerShellスクリプトでのユーザー・パスワードの入力方法は、企業または個人のセキュリティ標準によって異なります。パスワードを安全に取得する独自の方法を使用するか、この例のようにコマンド履歴およびコマンドライン・ウィンドウでパスワードを非表示にしてください。パスワード変数は、使用後に削除する必要があります。これらを次に示す順序で入力します。
      i. `$securePassword = Read-Host " Enter Password" -AsSecureString`
      ii.
      ```
      $Password =
      [System.Runtime.InteropServices.Marshal]::PtrToStringAuto([Syste
      m.Runtime.InteropServices.Marshal]::SecureStringToBSTR($securePa
      ssword))
      ```
   e. `$Scope = "database_app_id_uri/scope"`
      この値は、データベースのアプリケーションID URIおよびデータベースのスコープ(権限)を/スラッシュで区切って設定します。これらの値は、データベース・アプリ登録の「Expose an API」ページで確認できます。次の例で

は、https://example.com/111aa1aa-1111-1111-a1a1-1a11a111111aがアプリケーションID URIで、session:scope:connectがスコープです。

```
$Scope = "https://example.com/111aa1aa-1111-1111-a1a1-
1a11a111111a/session:scope:connect"
```

f. $requestBody = @{client_id=$AppClientId;grant_type="password";username=$Username;password=$Password;scope=$Scope;}
これは、この後行われるRESTコールのリクエスト本文です。

g. $OAuthResponse = Invoke-RestMethod -Method Post -Uri https://login.microsoftonline.com/$TenantDomain/oauth2/v2.0/token -Body $requestBody
これは、ユーザーのOAuth2アクセス・トークンを取得します。

h. 不要な場合は、変数からパスワードを削除できます。

i. $securePassword = $null (安全なパスワード文字列の場合)

ii. $Password = $null (クリアテキストのパスワード文字列の場合)

i. $AccessToken = $OAuthResponse.access_token | Out-File -FilePath .¥token -Encoding ASCII (ASCIIエンコーディングを使用して、OAuth2トークンを現在のファイルの場所に書き込む)

3. Azure AD OAuth2アクセス・トークンはJSON Web Token (JWT)形式のトークンであるため、必要に応じて、トークンのコンテンツをWebサイトにコピーして貼り付けることで、エンコードされたコンテンツのクリアテキストを表示できます。

https://jwt.io/

次のことに注意してください。

a. デフォルトのPowerShell UTF16ファイル・エンコーディングはトークンに使用できません。かわりにASCIIエンコーディングを使用してください。

b. トークンは、移動時に行われるファイルのエンコーディングの変更によっては、クロスプラットフォーム(WindowsからLinuxやLinuxからWindowsなど)で動作しない場合があります。

これで、OAuth2アクセス・トークンが取得され、ファイルとして格納されました。次のステップでは、SQL*Plusクライアントがストア・アクセス・トークンを使用し、それをデータベースに送信できるようにします。

親トピック: Azure AD OAuth2トークンの取得の例

8.4.5.2 例: Microsoft Authentication LibraryでPythonを使用して、認可フローを使用する

Microsoft Authentication Library (MSAL)を使用したこの例はPythonで作成されているため、PowerShellやLinuxなどの様々なプラットフォームで実行できます。

ユーザーに対してマルチファクタ認証が有効になっている場合は、ユーザーが2番目の認証を追加するためのOAuth2認可フローが必要です。認可フローではAzure ADへのラウンド・トリップが2回必要であるため、MSALを使用して処理することをお薦めします。MSALでPythonスクリプトを使用する方法は、Microsoftの記事『Get Azure AD tokens by using the Microsoft Authentication Library』を参照してください。これらの手順はDatabricksサービス用ですが、Databricksスコープではなく、データベース・アプリケーションIDのURIとスコープにスコープを変更します。

1. クライアント・アプリ登録を設定するためのステップはすでに完了しているため、このステップは省略します。ただし、クライアント・アプリ登録のリダイレクトURI (http://localhost)を追加していることを確認してください。
2. 直接、MSAL Pythonライブラリを使用してAzure ADトークンを取得する手順に進みます。

ディレクトリ(テナント) ID、パブリック・アプリケーション・クライアントのクライアントIDおよびデータベース・アプリケーションIDのURIおよびスコープが必要になります。scopesのコード・セクションには、この変数を変更しないようにという指示があります。このPythonコードはDatabricksスコープ用に作成されているため、このスコープ変数を、実際に使用するデータベースのスコープにあわせて変更することが必要になります。例:

```
scopes = ['https://example.com/1111aa1a-a1aa-1a11-11aa-
1a1a11aa1111/session:connect']
```

3. トークンをファイルの場所に書き込むようにコードを変更します。
   次のサンプル・コードを使用し、末尾のprint文に追加します。元のstdoutをバックアップしてリストアするための行が追加されていることに注目してください。

```
stdout_backup = sys.stdout
with open('token', 'w') as token_file:
    sys.stdout = token_file
    print(acquire_tokens_result['access_token'])
sys.stdout = stdout_backup
```

親トピック: Azure AD OAuth2トークンの取得の例

8.4.5.3 例: リソース所有者パスワード資格証明フローでCurlを使用する

この例では、パブリックAzure ADクライアントでリソース所有者パスワード資格証明(ROPC)フローを使用してAzure AD APIに対してcurlコマンドを使用する方法を示します。

このコマンドにはクリアテキスト・パスワードが含まれるため、エンド・ユーザー向けではなく、アプリケーション向けのコマンドです。これは保護される必要があります。

- 次のcurlコマンドを入力します。

```
curl -X POST -H 'Content-Type: application/x-www-form-urlencoded'
https://login.microsoftonline.com/az207oracleoutlook.onmicrosoft.com/oauth2/v2.
0/token
-d 'client_id=571c3f0a-aa3c-4f0a-93ed-4f75748955ea' -d
'scope=https://example.com/383fe7ee-1433-4844-a2d5-
5b80d811256d/session:scope:connect'
-d 'username=peter.fitch@example.com' -d 'password=password' -d
'grant_type=password'
```

レスポンスは、トークン・タイプ、スコープ、有効期限および実際のトークンを含むJSONファイルです。アクセス・トークンのみがファイルに書き込まれて格納されるように、このファイルを解析する必要があります。

親トピック: Azure AD OAuth2トークンの取得の例

8.4.5.4 例: Azure CLIを使用した認可フロー

この例では、Azure CLIを使用してアクセス・トークンを取得し、取得したトークンをファイルに書き込む方法を示します。

Azure CLIのインストール方法は、Microsoft Azureの記事『Install the Azure CLI on Linux』を参照してください。

1. Azureテナンシにログインします。

```
$ az login
```

2. アクセス・トークンを取得し、次の構文を使用してトークン変数に割り当てます。

```
token=$(az account get-access-token --resource=database_app_id_uri --query
accessToken --output tsv)
```

例:

```
token=$(az account get-access-token --resource=https://example.com/1111aa1a-a1aa-1a11-11aa-1a1a11aa1111 --query accessToken --output tsv)
```

Azure CLIクライアント・アプリケーションIDにデータベース・リソースにアクセスする権限がないというエラーが表示される場合は、エラー・メッセージからAzure CLIクライアント・アプリケーションIDをコピーして、データベース・リソースに対して認可されるクライアント・アプリケーションのリストに追加します。(Azure ADのデータベース・アプリ登録に移動し、「Expose an API」、「Add a client application」の順にクリックします)。

3. トークンをファイルに書き込みます。

```
$ echo "$token" >> token
```

親トピック: Azure AD OAuth2トークンの取得の例

8.4.6 Azure ADアクセス・トークン用のSQL*Plusの構成

特定の場所からAzure ADデータベース・アクセス・トークンを取得し、/スラッシュ・ログインが使用されるときにそのトークンを使用するように、SQL*Plusを構成する必要があります。

最新のSQL*PlusおよびInstant Clientのみが、Azure AD OAuth2トークンを使用できます。Azure ADトークンにはデフォルトの場所がないため、この場所を指定する必要があります。

1. Azure ADユーザー・アカウントを持っていることを確認します。
2. 次のいずれかについて、Azure AD管理者またはOracle Database管理者に確認します。
   - Azure ADトークンの取得に使用できるアプリケーション・クライアントID。アプリケーションを登録するAzure AD権限を持っている場合は、独自のクライアント・アプリ登録を作成します。これは、Oracle DatabaseインスタンスをAzure ADテナンシに登録する場合と同様です。
   - 自分がデータベース内のグローバル・スキーマにマップされていること。
3. Oracle Databaseクライアント・リリース19cの最新リリース更新を使用していることを確認します。
   この構成は、Oracle Databaseクライアント・リリース19cでのみ機能します。
4. 既存のプロセスに従ってOracle Databaseインスタンスからウォレットをダウンロードし、SQL*Plusで使用するためにウォレットを構成する手順に従います。
5. クライアントで、sqlnet.oraファイルの次のパラメータを設定します。
   - パラメータSSL_SERVER_DN_MATCH = ONをチェックして、DN一致が有効であることを確認します。
   - TOKEN_AUTHパラメータを設定して、クライアントがAzure ADトークンを使用できるようにします。トークンの場所を指すTOKEN_LOCATIONパラメータを含めます。例:

     ```
     TOKEN_AUTH=OAUTH
     TOKEN_LOCATION="token_location"
     ```

     デフォルトの場所はないことに注意してください。トークンの名前がtokenの場合は、指定する必要があるのはファイル・ディレクトリ(/test/oracle/aad-tokenなど)のみです。トークン名がtokenと異なる場合(たとえば、azure.token)、この名前をパスに含める必要があります(たとえば、/test/oracle/aad-token/azure.token)。

TOKEN_AUTHおよびTOKEN_LOCATIONパラメータは、tnsnames.oraおよびsqlnet.oraで指定できます。tnsnames.ora接続文字列内のTOKEN_AUTHおよびTOKEN_LOCATION値は、その接続のsqlnet.ora設定より優先されます。例:

```
(description=
```

```
  (retry_count=20)(retry_delay=3)
  (address=(protocol=tcps)(port=1522)
  (host=example.us-phoenix-1.oraclecloud.com))
  (connect_data=(service_name=aaabbbccc_exampledb_high.example.oraclecloud.com))
  (security=(ssl_server_cert_dn="CN=example.uscom-east-1.oraclecloud.com,
     OU=Oracle BMCS US, O=Example Corporation,
     L=Redwood City, ST=California, C=US")
  (TOKEN_AUTH=OAUTH)(TOKEN_LOCATION="/test/oracle/aad-token"))
```

TOKEN_AUTHおよびTOKEN_LOCATIONパラメータによって接続文字列を更新したら、AzureユーザーはSQL*Plusを起動する次のコマンドを実行して、Oracle Databaseインスタンスにログインできます。接続記述子自体を含めるか、tnsnames.oraファイルからの記述子の名前を使用できます。

```
connect /@exampledb_high
```

または、接続文字列を使用できます。例:

```
connect /@(description=
  (retry_count=20)(retry_delay=3)
  (address=(protocol=tcps)(port=1522)
  (host=example.us-phoenix-1.oraclecloud.com))
  (connect_data=(service_name=aaabbbccc_exampledb_high.example.oraclecloud.com))
  (security=(ssl_server_cert_dn="CN=example.uscom-east-1.oraclecloud.com,
     OU=Oracle BMCS US, O=Example Corporation,
     L=Redwood City, ST=California, C=US")
(TOKEN_AUTH=OAUTH)(TOKEN_LOCATION="/test/oracle/aad-token")
```

TOKEN_AUTHはsqlnet.oraファイルまたは接続文字列のいずれかによってすでに設定されているため、データベース・クライアントはAzure OAuth2トークンを取得するようにすでに構成されています。データベース・クライアントはOAuth2トークンを取得し、そのトークンをOracle Databaseインスタンスに送信します。

関連トピック

- Oracle DatabaseインスタンスのMicrosoft Azure ADテナンシへの登録

親トピック: Oracle DatabaseへのAzure ADクライアント接続の構成

8.4.7 データベースがインターネットに接続するためのネットワーク・プロキシの作成

このネットワーク・プロキシを使用すると、OracleデータベースはAzure ADエンドポイントに到達できるようになります。

- デフォルトのOracle Database環境用のネットワーク・プロキシの作成
  ネットワーク・プロキシを作成するには、環境変数を設定してからリスナーを再起動する必要があります。
- Oracle Real Application Clusters環境用のネットワーク・プロキシの作成
  ネットワーク・プロキシを作成するには、環境変数を設定してからデータベースを再起動する必要があります。

親トピック: Oracle DatabaseへのAzure ADクライアント接続の構成

8.4.7.1 デフォルトのOracle Database環境用のネットワーク・プロキシの作成

ネットワーク・プロキシを作成するには、環境変数を設定してからリスナーを再起動する必要があります。

データベースを再起動する必要はありません。

1. Oracleデータベースがインストールされているサーバーで、http_proxy環境変数を設定します。
   例:

   ```
   export http_proxy=http://www-proxy-example.com:80/
   ```

2. リスナーを再起動します。

```
lsnrctl stop
lsnrctl start
```

親トピック: データベースがインターネットに接続するためのネットワーク・プロキシの作成

8.4.7.2 Oracle Real Application Clusters環境用のネットワーク・プロキシの作成

ネットワーク・プロキシを作成するには、環境変数を設定してからデータベースを再起動する必要があります。

1. Oracleデータベースがインストールされているサーバーで、http_proxy環境変数を設定します。
   ネットワーク・プロキシを設定するには、次の構文を使用します。入力するプロキシ・コマンドでは、プロキシ名の前にhttp://が必要であり、プロキシの末尾にポート番号が必要です。

   ```
   http_proxy=http://....:80/
   ```

   例:

   ```
   srvctl setenv database -db db_name -env "http_proxy=http://www-
   proxy.example.com:80/"
   ```

2. データベースを停止します。

   ```
   $srvctl stop database -db db_name
   ```

3. 環境変数の値を表示して、値が正しく設定されていることを確認します。

   ```
   $ srvctl getenv database -db db_name
   ```

   次のような出力が表示されます。

   ```
   db_name:
   http_proxy=http://www-proxy.example.com:80/
   https_proxy=http://www-proxy.example.com:80/
   ```

4. データベースを再起動します。

   ```
   $ srvctl start database -db db_name
   ```

親トピック: データベースがインターネットに接続するためのネットワーク・プロキシの作成

8.4.8 クライアントによるAzureトークンの直接取得の有効化

クライアント自身がAzureトークンを直接取得できるようにパラメータを設定できます。

この機能は、JDBCシン・クライアント、ODP.NET Coreクラスまたは ODP.NET管理対象ドライバ・クラスを使用する環境で使用できます。これにより、クライアントは、次の方法を使用して認証リクエストをユーザーに表示できます。

- ユーザーがWebアプリケーションを使用している場合、ユーザーの認証を求める認証リクエストがダイアログ・ボックスに表示されます。
- ユーザーがコマンドライン・シェルで作業している場合、認証リクエストはプロンプトとして表示されます。

これらの認証リクエスト・タイプのいずれかでこの機能を有効にするには、クライアントのsqlnet.oraファイルまたは接続文字列で次のパラメータを設定する必要があります。接続文字列はsqlnet.oraよりも優先されます。

表8-1 トークンを直接取得するパラメータ

パラメータ	説明
TOKEN_AUTH	トークン認証を設定します。次のいずれかの値を入力します。 ● AZURE_DEVICE_CODE は、Azure AD アクセス・トークンをリクエストするためのデバイス・コード・フローに従うようにデータベース・ドライバに通知します。これは、環境がブラウザを開けない場合(コマンドラインのみの環境)に、人間のユーザーにも使用されます。デバイス・コードと Azure AD ログイン URL がツールの標準出力に書き込まれ、ユーザーが携帯電話またはラップトップの Azure AD にログインし、デバイス・コードを入力します。ユーザーは別のチャネルを介して認証され、認証が成功した場合は引き続きデータベースにアクセスできます。 ● AZURE_INTERACTIVE は、データベースのアクセス・トークンを取得するために Azure OAuth2 対話型(OAuth2 認可)フローを進む必要があることをドライバに伝えます。これにより、外部スクリプトを使用せずに、Azure AD から直接トークンを取得するようにデータベース・クライアントが構成されます。これは、SQLcl などのツールにログインし、各自の環境でブラウザ・ウィンドウを開いて Azure AD への認証を行うことができる人間のユーザー向けです。 ● AZURE_MANAGED_IDENTITY を使用すると、ドライバは、ホスト・システムに割り当てられている識別情報として認証できます。ホスト・システムは、仮想マシンなどの Azure AD によって管理されるリソースである必要があります。 ● AZURE_SERVICE_PRINCIPAL を使用すると、ドライバは登録済アプリケーションのシークレットまたは証明書を使用して認証できます。
AZURE_CLIENT_ID	アプリケーションが登録されたときに Azure AD によってアプリケーションに割り当てられた、一意のアプリケーション(クライアント) ID。このアプリケーションは、ユーザーのためにデータベースのアクセス・トークン取得をリクエストするデータベース・クライアントです。
AZURE_DB_APP_ID_URI	アプリケーション ID URI は、Azure AD 内のアプリケーションを一意に識別する URI です。この値は、データベースの Azure AD アプリケーション登録の概要画面から取得します。
AZURE_TENANT_ID	データベースの Azure テナンシ ID を指定します。

親トピック: Oracle DatabaseへのAzure ADクライアント接続の構成

8.5 Microsoft Azure ADプロキシ認証の構成

プロキシ認証により、Azure ADユーザーは、アプリケーションのメンテナンスなどのタスクのためにデータベース・スキーマにプロキシできます。

- Microsoft Azure ADプロキシ認証の構成について
  Azureユーザーは、プロキシ認証を使用してOracle Autonomous Databaseに接続できます。
- Azure ADユーザーのためのプロキシ認証の構成
  Azure ADユーザーのプロキシ認証を構成するには、このユーザーがグローバル・スキーマへのマッピング(排他的マッピングまたは共有マッピング)をすでに持っている必要があります。Azure ADユーザーがプロキシする別のデータベース・スキーマも使用可能になっている必要があります。
- Azure ADユーザー・プロキシ認証の検証

Azure ADユーザー・プロキシ構成でトークン認証を検証できます。

親トピック: Oracle DatabaseのMicrosoft Azure Active Directoryユーザーの認証および認可

8.5.1 Microsoft Azure ADプロキシ認証の構成について

Azureユーザーはプロキシ認証を使用して、Oracle Autonomous Databaseに接続できます。

プロキシ認証は、通常、実際のユーザーを認証し、アプリケーションを管理するためにスキーマ権限およびロールを含むデータベース・スキーマの使用をユーザーに認可するために使用されます。アプリケーション・スキーマ・パスワードの共有などの代替方法は、安全でないものとみなされ、どの実際のユーザーがアクションを実行したかを監査できません。

たとえば、アプリケーション・データベース管理者である名前付きAzure ADユーザーが資格証明を使用して認証し、データベース・スキーマ・ユーザー(hrappなど)にプロキシできる環境でのユースケースが考えられます。この認証により、Azure AD管理者は、アプリケーションのメンテナンスを実行するためにhrapp権限およびロールをユーザーhrappとして使用できますが、認証にはAzure AD資格証明を使用します。アプリケーション・データベース管理者は、データベースにサインインし、アプリケーション・スキーマにプロキシしてこのスキーマを管理できます。

親トピック: Microsoft Azure ADプロキシ認証の構成

8.5.2 プロキシ認証のためのAzure ADユーザーの構成

Azure ADユーザーのプロキシ認証を構成するには、このユーザーがすでにグローバル・スキーマへのマッピング(排他的マッピングまたは共有マッピング)を持っている必要があります。Azure ADユーザーがプロキシする別のデータベース・スキーマも使用可能になっている必要があります。

このタイプのユーザーがいることを確認したら、Azure ADユーザーにデータベース・ユーザーへのプロキシを許可するようにデータベース・ユーザーを変更します。

1. ALTER USERシステム権限を持つユーザーとして、Autonomous Databaseインスタンスにログインします。
2. ローカル・データベース・ユーザー・アカウントにプロキシする権限をAzure ADユーザーに付与します。
   コマンドではAzure ADユーザーを参照できないため、プロキシはデータベース・グローバル・ユーザー(Azure ADユーザーにマップ)とターゲット・データベース・ユーザーの間で作成する必要があります。
   次の例では、hrappはプロキシ先のデータベース・スキーマで、peterfitch_schemaはユーザーpeterfitchに排他的にマップされるデータベース・グローバル・ユーザーです。

```
ALTER USER hrapp GRANT CONNECT THROUGH peterfitch_schema;
```

この段階で、Azure ADユーザーはプロキシを使用してデータベース・インスタンスにログインできます。例:

```
CONNECT [hrapp]/@connect_string
```

親トピック: Microsoft Azure ADプロキシ認証の構成

8.5.3 Azure ADユーザー・プロキシ認証の検証

Azure ADユーザー・プロキシ構成でトークン認証を検証できます。

1. CREATE USERおよびALTER USERシステム権限が付与されたユーザーとして、Oracle Autonomous Databaseインスタンスにログインします。
2. Azure ADユーザーとして接続し、SHOW USERおよびSELECT SYS_CONTEXTコマンドを実行します。
   たとえば、データベース・ユーザーhrappにプロキシするときに、Azure ADユーザーpeterfitchのプロキシ認証を確

認するとします。

```
CONNECT [hrapp]/@connect_string
SHOW USER;
--The output should be USER is "HRAPP "
SELECT SYS_CONTEXT('USERENV','AUTHENTICATION_METHOD') FROM DUAL;
--The output should be "TOKEN_GLOBAL"
SELECT SYS_CONTEXT('USERENV','PROXY_USER') FROM DUAL;
--The output should be "PETERFITCH_SCHEMA"
SELECT SYS_CONTEXT('USERENV','CURRENT_USER') FROM DUAL;
--The output should be "HRAPP"
```

親トピック: Microsoft Azure ADプロキシ認証の構成

8.6 Microsoft Azure AD接続のトラブルシューティング

トレース・ファイルを使用して、Microsoft Azure AD接続の問題を診断できます。ORA-12599およびORA-03114エラーを簡単に修正することもできます。

- Azure ADとのOracle Databaseクライアント接続をトラブルシューティングするためのトレース・ファイル
  トレース・ファイルを使用して、Microsoft Azure ADを使用したOracle Database統合をトラブルシューティングできます。
- トークンを使用してデータベースにアクセスしようとしたときに発生するORA-12599およびORA-03114エラー
  ORA-12599: 「TNS: 暗号チェックサムの不一致が発生しました」エラーおよびORA-03114: 「Oracleに接続されていません。」エラーは、接続しようとしているデータベースがネイティブ・ネットワーク暗号化によって保護されていることを示しています。
- Azure ADアクセス・トークンのバージョンの確認
  JSON WebトークンのWebサイトを使用すると、サイトで使用されているMicrosoft Azure ADアクセス・トークンのバージョンを確認できます。

親トピック: Oracle DatabaseのMicrosoft Azure Active Directoryユーザーの認証および認可

8.6.1 Oracle DatabaseクライアントとAzure ADの接続をトラブルシューティングするためのトレース・ファイル

トレース・ファイルを使用して、Oracle DatabaseとMicrosoft Azure ADの統合に関してトラブルシューティングできます。

- 接続のトラブルシューティングに使用されるトレース・ファイルについて
  クライアント側でMicrosoft Azure AD接続をトラブルシューティングするために、2つのレベルのトレース・ファイルを生成できます。
- トークン認証のクライアント・トレースの設定
  クライアント側のsqlnet.oraファイルにEVENT設定を追加して、クライアント・トレースを制御できます。

親トピック: Microsoft Azure AD接続のトラブルシューティング

8.6.1.1 接続のトラブルシューティングに使用するトレース・ファイルについて

クライアント側でMicrosoft Azure AD接続をトラブルシューティングするために、2つのレベルのトレース・ファイルを生成できます。

生成できるトレース・ファイルの2つのレベルは次のとおりです。

- 低レベル・トレースでは、エラーの発生時にトレースが出力されます。

        - TCPSがAzure AD接続用に設定されていない場合は、プロトコルがTCPSである必要があるというメッセージが出力されます。
        - SSL_SERVER_DN_MATCHがTRUEに設定されていない場合は、値がFALSEであるというメッセージが出力されます。
        - TOKEN_LOCATIONが指定されていない場合は、トークンの場所が存在しないというメッセージが出力されます。
        - 指定されたTOKEN_LOCATIONにトークンが存在しない場合は、メッセージが出力されます。
        - OCI_ATTR_TOKEN_ISBEARERをtrueに設定せずにアプリケーションがトークンを渡した場合、欠落している属性についてのメッセージが出力されます。
        - アプリケーションがOCI_ATTR_TOKEN_ISBEARERをTRUEに設定し、トークンを渡さなかった場合、欠落している属性についてのメッセージが出力されます。
        - トークンが期限切れの場合は、メッセージが出力されます。
- 高レベル・トレースでは、前述のようにエラーの発生時にトレースが出力されます。さらに、次のように、成功時にトレースが出力されます。
    - SSL_SERVER_DN_MATCHが存在する場所(tnsnames.oraまたはsqlnet.ora)が出力されます。また、TRUEに設定されている場合は、TRUEの値が出力されます。
    - アプリケーションでトークンとOCI_ATTR_TOKEN_ISBEARER=trueの両方が設定されている場合は、メッセージが出力されます。
    - TOKEN_AUTHに、正しい値OAUTHが設定されている場合は、その値が出力されます。
    - トークンが期限切れでない場合は、メッセージが出力されます。

親トピック: Oracle DatabaseクライアントとAzure ADの接続をトラブルシューティングするためのトレース・ファイル

8.6.1.2 トークン認証のクライアント・トレースの設定

クライアント側のsqlnet.oraファイルにEVENT設定を追加して、クライアント・トレースを制御できます。

これらのEVENT設定は、IAMおよびAzure ADの両方でOracle Databaseとの接続に使用できます。

- 次のいずれかの方法を使用します。
    - クライアント側のsqlnet.oraファイルに次の設定を追加します。
        - EVENT_25701=14 (低レベル・トレースの場合)
        - EVENT_25701=15 (高レベル・トレースの場合)
    - 環境変数EVENT_25701を設定します。
        - EVENT_25701=14 (低レベル・トレースの場合)
        - EVENT_25701=15 (高レベル・トレースの場合)

    クライアント・トレース・ファイルは、次の場所に作成されます。
    - Linux: $ORACLE_HOME/log/diag/clients
    - Windows: %ORACLE_HOME%¥log¥diag¥clients

    クライアント側のsqlnet.oraのADR_BASEパラメータを使用して、トレース・メッセージが格納されるディレクトリを指定できます。ディレクトリ・パスが有効で、書込み権限があることを確認します。DIAG_ADR_ENABLEDパラメータがFALSEに設定されていないことを確認します。

    ADR_BASEの設定例を次に示します。

    ```
    ADR_BASE=/oracle/oauth2/trace
    ```

親トピック: Oracle DatabaseクライアントとAzure ADの接続をトラブルシューティングするためのトレース・ファイル

8.6.2 トークンを使用してデータベースにアクセスしようとしたときにORA-12599およびORA-03114エラーが発生する

ORA-12599: 「TNS: 暗号チェックサムの不一致が発生しました」エラーおよびORA-03114: 「Oracleに接続されていません。」エラーは、接続しようとしているデータベースがネイティブ・ネットワーク暗号化によって保護されていることを示します。

トークンを使用してOracleデータベースにアクセスする場合は、ネットワーク・ネイティブ暗号化ではなくTransport Layer Security (TLS)接続を確立する必要があります。これらのエラーを修正するには、TLSがデータベースに対して適切に構成されていることを確認してください。ローカル・データベースのユーザー名とパスワードを使用して構成をテストし、次のSYSCONTEXT USERENVパラメータを確認する必要があります。

- NETWORK_PROTOCOL
- TLS_VERSION

関連トピック

- Transport Layer Security認証の構成

親トピック: Microsoft Azure AD接続のトラブルシューティング

8.6.3 Azure ADアクセス・トークンのバージョンの確認

JSON WebトークンのWebサイトを使用すると、サイトで使用されているMicrosoft Azure ADアクセス・トークンのバージョンを確認できます。

デフォルトではAzure AD Microsoft Azure AD v1アクセス・トークンですが、サイトでv2の使用が選択されている場合があります。Oracle Databaseではv1トークンがサポートされており、Autonomous Database Serverlessではv2トークンもサポートされています。v2アクセス・トークンを使用する必要がある場合は、Oracleデータベースに対してそれらの使用を有効にできます。使用しているAzure ADアクセス・トークンのバージョンを特定するには、Azure AD管理者に確認するか、次のようにJSON WebトークンのWebサイトからバージョンを確認します。

1. JSON WebトークンのWebサイトにアクセスします。

   ```
   https://jwt.io/
   ```

2. トークン文字列をコピーして「エンコード」フィールドに貼り付けます。
3. デコードフィールドを確認します。ここにはトークン文字列に関する情報が表示されています。
   そのフィールドの近くか下部に、次のバージョンのいずれかを示すverというクレームが表示されます。
   - "ver": "1.0"
   - "ver": "2.0"

関連トピック

- Microsoft Azure AD v2アクセス・トークンの有効化

親トピック: Microsoft Azure AD接続のトラブルシューティング

9 定義者権限および実行者権限のセキュリティの管理

ユーザー定義プロシージャの実行中に権限へのアクセス制御で実行者権限および定義者権限を使用すると、セキュリティ上のメリットが得られます。

- 定義者権限および実行者権限について
  定義者権限および実行者権限は、ユーザー作成プロシージャまたはプログラム・ユニットの実行中に必要な権限へのアクセスを制御するときに使用されます。
- プロシージャに対する権限が定義者権限に与える影響
  定義者と呼ばれるプロシージャの所有者は、プロシージャが参照するオブジェクトに対する必要なオブジェクト権限を所有している必要があります。
- プロシージャに対する権限が実行者権限に与える影響
  実行者権限プロシージャは、すべての実行者権限で実行されます。
- 実行者権限プロシージャを作成する場合
  特定の状況で実行者権限プロシージャを作成することをお薦めします。
- プロシージャ・コールおよびビュー・アクセスの実行者権限の制御
  INHERIT PRIVILEGES権限およびINHERIT ANY PRIVILEGES権限は、実行者権限プロシージャの実行時に使用される権限を規制します。
- ビューの定義者権限および実行者権限
  CREATE VIEW SQL文でBEQEATH句を使用して、ユーザー作成ビューで定義者権限と実行者権限を制御できます。
- 定義者権限および実行者権限のコード・ベース・アクセス制御の使用
  データベース・ロールをPL/SQLファンクション、プロシージャまたはパッケージに付与するときに使用するコード・ベース・アクセス制御は、定義者権限および実行者権限のプロシージャと一緒に使用すると効果的です。
- データベース・リンクの定義者権限の制御
  アプリケーションでデータベース・リンクと定義者権限プロシージャが使用されている場合は、定義者権限プロシージャの権限付与を制御できます。

親トピック: ユーザー認証および認可の管理

9.1 定義者権限および実行者権限について

定義者権限および実行者権限は、ユーザー作成プロシージャまたはプログラム・ユニットの実行中に必要な権限へのアクセスを制御するときに使用されます。

定義者権限プロシージャでは、プロシージャが所有者の権限で実行されます。権限は、作成されたスキーマにバインドされます。実行者の権限プロシージャは現行ユーザー(プロシージャを実行するユーザー)の権限で実行されます。

たとえば、ユーザーbixbyが表cust_recordsを変更するために設計されるプロシージャを作成し、このプロシージャのEXECUTE権限をユーザーrlaytonに付与するとします。bixbyが定義者権限でプロシージャを作成した場合、プロシージャはbixbyのスキーマの表cust_recordsを検索します。プロシージャが実行者権限で作成された場合、rlaytonが実行すると、プロシージャはrlaytonのスキーマの表cust_recordsを検索します。

デフォルトでは、すべてのプロシージャは、定義者権限とみなされます。作成時または変更時に、AUTHID CURRENT_USER句を使用してプロシージャを実行者権限プロシージャに指定するか、AUTHID DEFINER句を使用して定義者権限プロシージャに変更できます。

権限分析ポリシーを作成して、定義者権限および実行者権限プロシージャの権限の使用を取得できます。

関連トピック

- 権限分析の実行による権限使用の特定
- Oracle Database PL/SQL言語リファレンス

親トピック: 定義者権限および実行者権限のセキュリティの管理

9.2 プロシージャに対する権限が定義者権限に与える影響

定義者と呼ばれるプロシージャの所有者は、プロシージャが参照するオブジェクトに対する必要なオブジェクト権限を所有している必要があります。

プロシージャ所有者が別のユーザーにそのプロシージャを使用する権限を付与すると、(プロシージャで参照されるオブジェクトに対する)プロシージャ所有者の権限が、権限受領者のプロシージャ実行に適用されます。プロシージャの定義者の権限は、ロールを介してではなく、プロシージャ所有者に直接付与する必要があります。これらは、定義者権限と呼ばれます。

所有者以外のプロシージャのユーザーは、実行者と呼ばれます。実行者権限プロシージャの場合は、参照オブジェクトに対する追加の権限が必要ですが、 定義者権限プロシージャの場合は不要です。

定義者権限プロシージャのユーザーに必要なのは、そのプロシージャを実行する権限のみで、そのプロシージャでアクセスする基礎となるオブジェクトに対する権限は不要です。これは、定義者権限プロシージャは、その実行者に関係なく、プロシージャを所有するユーザーのセキュリティ・ドメインの下で動作するためです。プロシージャの所有者は、参照オブジェクトに対する必要なオブジェクト権限をすべて所有している必要があります。定義者権限プロシージャのユーザーに付与する権限は、できるかぎり控えめに付与してください。これによって、データベース・アクセスを厳密に制御できます。

定義者権限プロシージャを使用すると、プライベート・データベース・オブジェクトへのアクセスを制御し、データベースのセキュリティ・レベルを強化できます。定義者権限プロシージャを記述し、ユーザーにEXECUTE権限のみを付与することによって、そのプロシージャを介さない場合には、このユーザーが参照オブジェクトにアクセスできないように規定できます。

実行時には、定義者権限プロシージャの所有者の権限によってそのプロシージャの参照オブジェクトへのアクセスが許可されているかどうかが、プロシージャの実行前にチェックされます。参照オブジェクトに対して必要な権限が、定義者権限プロシージャの所有者から取り消されていると、所有者を含むユーザーは、プロシージャを実行できません。

定義者権限プロシージャを使用する場合の例は次のとおりです。表へのアクセスが制限されていないプロシージャを持つAPIを作成する必要があるとします。ただし、一般ユーザーが表のデータを直接選択し、INSERT文、UPDATE文およびDELETE文を使用して変更しないようにする必要があります。これを実行するには、個別の権限の弱いスキーマで、APIを構成する表およびプロシージャを作成します。デフォルトでは各プロシージャは定義者権限ユニットであるため、作成時にAUTHID DEFINERを指定する必要がありません。次に、EXECUTE権限をこのAPIを使用する必要があるユーザーに付与しますが、データ・アクセスを許可する権限を付与しないでください。この解決策は、API動作の完全な制御およびユーザーが基礎オブジェクトにアクセスする方法を提供します。

独自のスキーマで、定義者権限プロシージャおよびこれらのプロシージャにアクセスするビューを作成することをお薦めします。このスキーマに非常に弱い権限を付与するか、権限を付与しません。これによって、他のユーザーがこれらのプロシージャまたはビューを実行する場合、このスキーマの不要な高い権限にアクセスしません。

ノート:

トリガーの処理は、定義者権限プロシージャと同じパターンに従います。ユーザーは、実行権限がある SQL 文を実

行します。この SQL 文の実行結果として、トリガーが起動されます。トリガーされたアクション内の文は、そのトリガーを所有するユーザーのセキュリティ・ドメインで一時的に実行されます。トリガーの概要は、『Oracle Database 概要』を参照してください。

関連トピック

- PL/SQLブロックでのロールの機能

親トピック: 定義者権限および実行者権限のセキュリティの管理

9.3 プロシージャに対する権限が実行者権限に与える影響

実行者権限プロシージャは、すべての実行者権限で実行されます。

実行者の使用可能な任意のロールを介してその実行者に付与された権限は、定義者権限プロシージャによって実行者権限プロシージャが直接または間接的にコールされないかぎり有効です。実行者権限プロシージャのユーザーには、そのプロシージャが実行者のスキーマ内で解決される外部参照を介してアクセスする、オブジェクトに対する権限(直接またはロールを介して付与されたもの)が必要です。実行者が実行者権限プロシージャを実行する場合、このユーザーは、実行者のすべての権限を一時的に保持します。

実行者には、DML文または動的SQL文に埋め込まれているプログラム参照にアクセスする権限が実行時に必要です。これは、この種のプログラム参照は実質的に実行時に再コンパイルされるためです。

PL/SQLファンクションの直接コールなど、他のすべての外部参照の場合、所有者権限はコンパイル時にチェックされ、実行時にはチェックされません。したがって、実行者権限プロシージャのユーザーには、DML文や動的SQL文の外側にある外部参照に対する権限は不要です。したがって、実行者権限プロシージャの開発者による権限の付与が必要なのは、プロシージャ自体に対する権限付与のみで、その実行者権限プロシージャによって直接参照されるすべてのオブジェクトに対する権限付与は必要ありません。

複数のプログラム・ユニットからなり、そのうちのいくつかは定義者権限、その他は実行者権限とするソフトウェア・バンドルを作成して、プログラム・エントリ・ポイントを制限できます(制御されたステップイン)。エントリ・ポイント・プロシージャの実行権限があるユーザーは、内部プログラム・ユニットも間接的に実行できますが、内部プログラムを直接コールすることはできません。問合せ処理を厳密に制御するには、PL/SQLパッケージ仕様を明示的なカーソルを使用して作成できます。

関連トピック

- プロシージャ・コールおよびビュー・アクセスの実行者権限の制御

親トピック: 定義者権限および実行者権限のセキュリティの管理

9.4 実行者権限プロシージャを作成する場合

特定の状況で実行者権限プロシージャを作成することをお薦めします。

これらの状況は次のとおりです。

- 権限の高いスキーマのPL/SQLプロシージャを作成する場合。権限の弱いユーザーがプロシージャを起動する場合、それらのユーザーが実行を許可された部分のみ実行できます。つまり、実行者権限プロシージャは、実行するユーザーの権限で実行されます。
- PL/SQLプロシージャがSQLを含まず、PL/SQLプロシージャを他のユーザーが使用できる場合。DBMS_OUTPUT PL/SQLパッケージは、SQLを含まないすべてのユーザーが使用できるPL/SQLサブプログラムの例です。この状況で実

行者権限プロシージャを使用する必要がある理由は、ユニットが実行時にSQL文を発行しないので、実行時システムが権限をチェックする必要がないためです。AUTHID CURRENT_USERを指定すると、実行者権限プロシージャがコール・スタックを使用する場合にCURRENT_USERおよびCURRENT_SCHEMAの値と現在有効なロールが変更されないため、プロシージャの起動がより効率的になります。

関連項目:

- Oracle Virtual Private Databaseのポリシーの構成
- ANY権限とPUBLICロールについて
- Oracle Databaseで名前の解決を処理する方法と、実行者権限および定義者権限を使用して実行時の権限チェックを処理する方法の詳細は、『Oracle Database PL/SQLパッケージおよびタイプ・リファレンス』を参照してください。
- 実行者権限と定義者権限のユニットの違いの詳細は、『Oracle Database PL/SQLパッケージおよびタイプ・リファレンス』を参照してください。
- CREATE PACKAGE文で明示的なカーソルを定義する方法の詳細は、『Oracle Database PL/SQLパッケージおよびタイプ・リファレンス』を参照してください。

親トピック: 定義者権限および実行者権限のセキュリティの管理

9.5 プロシージャ・コールおよびビュー・アクセスの実行者権限の制御

INHERIT PRIVILEGES権限およびINHERIT ANY PRIVILEGES権限は、実行者権限プロシージャの実行時に使用される権限を規制します。

- スキーマの権限が実行者権限プロシージャの使用に与える影響
  権限の低いユーザーが権限の高いユーザーが所有するプロシージャを実行するような場合は、実行者権限プロシージャが便利です。
- INHERIT [ANY] PRIVILEGES権限による権限アクセスの制御方法
  INHERIT PRIVILEGESおよびINHERIT ANY PRIVILEGES権限を使用して、実行者権限プロシージャを保護します。
- 他のユーザーへのINHERIT PRIVILEGES権限の付与
  デフォルトで、すべてのユーザーにINHERIT PRIVILEGES ON USER newuser TO PUBLICが付与されます。
- 例: 実行するユーザーのINHERIT PRIVILEGESの付与
  GRANT文で、実行するユーザーのINHERIT PRIVILEGES権限をプロシージャ所有者に付与できます。
- 例: INHERIT PRIVILEGESの取消し
  REVOKE文で、ユーザーのINHERIT PRIVILEGES権限を取り消すことができます。
- 他のユーザーへのINHERIT ANY PRIVILEGES権限の付与
  デフォルトでは、ユーザーSYSは、INHERIT ANY PRIVILEGESシステム権限を持ち、この権限を他のデータベース・ユーザーまたはロールに付与できます。
- 例: 信頼できるプロシージャ所有者へのINHERIT ANY PRIVILEGESの付与
  GRANT文で、INHERIT ANY PRIVILEGES権限を信頼できるプロシージャ所有者に付与できます。
- INHERIT PRIVILEGESおよびINHERIT ANY PRIVILEGESの管理
  デフォルトでは、PUBLICは、新しいユーザー・アカウントおよびアップグレードされたユーザー・アカウントのINHERIT PRIVILEGE権限を持ち、SYSユーザーは、INHERIT ANY PRIVILEGES権限を持ちます。

親トピック: 定義者権限および実行者権限のセキュリティの管理

9.5.1 スキーマの権限が実行者権限プロシージャの使用に与える影響

権限の低いユーザーが権限の高いユーザーが所有するプロシージャを実行するような場合は、実行者権限プロシージャが便利です。

ユーザーが実行者権限プロシージャ(またはAUTHID CURRENT_USER句で作成されたPL/SQLプログラム・ユニット)を実行する場合、プロシージャの実行中に実行するユーザーのすべての権限を一時的に継承します。

その期間中に、プロシージャ所有者は、プロシージャを通じて、この実行するユーザーの権限にアクセスできます。次の使用例を考えてみます。

1. ユーザーebrownは、check_syntax実行者権限プロシージャを作成し、ユーザーjwardにそのEXECUTE権限を付与します。
2. 準プログラマのユーザーebrownは、仕事に必要な最低限のセットの権限のみです。check_syntaxプロシージャは、ebrownのスキーマにあります。
3. マネージャのユーザーjwardは、ユーザーebrownよりさらに強力なセットの権限を持ちます。
4. ユーザーjwardがcheck_syntax実行者権限プロシージャを実行する場合、プロシージャは実行中にユーザーjwardより高い権限を継承します。
5. ユーザーebrownがcheck_syntaxプロシージャを所有するため、jwardがcheck_syntaxプロシージャを実行するたびに、ユーザーjwardの権限にアクセスできます。

jwardがプロシージャを実行するたびに権限の弱いebrownのプロシージャがjwardの高い権限にアクセスできるこのタイプの状況の危険性は、プロシージャ所有者が実行するユーザーの高い権限を悪用できるリスクがあることです。たとえば、ユーザーebrownは、check_syntaxプロシージャをリライトしてebrownを上げるかebrownの不良なパフォーマンス評価レコードを削除して、jwardの高い権限を使用する可能性があります。また、ebrownは、元から定義者権限プロシージャとしてプロシージャを作成し、EXECUTE権限をjwardに付与して、後でjwardに通知することなく不正な可能性のある実行者権限プロシージャに変更できた可能性があります。アプリケーション・ユーザーなどのランダムなユーザーが実行者権限プロシージャを使用するデータベースにアクセスできる場合、これらのタイプのリスクが増加します。

ユーザーjwardがebrownの実行者権限プロシージャを実行する場合、信頼要素が含まれます。ebrownがjwardの権限にアクセスする場合に悪質な方法でcheck_syntaxプロシージャを使用しないことを確認する必要があります。INHERIT PRIVILEGESおよびINHERIT ANY PRIVILEGES権限は、ユーザーjwardに対してユーザーebrownのプロシージャがjwardの権限にアクセスできるかどうかの制御をサポートできます。ユーザーは、実行する実行者権限プロシージャのユーザーへのINHERIT PRIVILEGES権限を付与または取り消すことができます。SYSユーザーは、INHERIT ANY PRIVILEGES権限を管理します。

親トピック: プロシージャ・コールおよびビュー・アクセスの実行者権限の制御

9.5.2 INHERIT [ANY] PRIVILEGES権限による権限アクセスの制御方法

INHERIT PRIVILEGESおよびINHERIT ANY PRIVILEGES権限を使用して、実行者権限プロシージャを保護します。

INHERIT PRIVILEGESおよびINHERIT ANY PRIVILEGES権限は、ユーザーが実行者権限プロシージャを実行するか、実行者権限プロシージャを参照するBEQUEATH CURRENT_USERビューに問い合せる場合に使用される権限を規制します。

ユーザーが実行者権限プロシージャを実行する場合、Oracle Databaseは、プロシージャ所有者が実行するユーザーの

INHERIT PRIVILEGES権限を持っているか、所有者にINHERIT ANY PRIVILEGES権限が付与されているかを確認します。権限チェックに失敗した場合、Oracle Databaseは、 ORA-06598: INHERIT PRIVILEGES権限が不十分ですエラーを戻しません。

これらの2つの権限の利点は、実行者権限プロシージャを実行するか、BEQUEATH CURRENT_USERビューに問い合せる場合、実行するユーザーに権限にアクセスできるユーザーの制御を提供することです。

親トピック: プロシージャ・コールおよびビュー・アクセスの実行者権限の制御

9.5.3 他のユーザーへのINHERIT PRIVILEGES権限の付与

デフォルトで、すべてのユーザーにINHERIT PRIVILEGES ON USER newuser TO PUBLICが付与されます。

付与が行われるのは、ユーザー・アカウントの作成時または以前に作成されたアカウントが現在のリリースにアップグレードされたときです。

実行するユーザーは、他のユーザーのINHERIT PRIVILEGE権限を取り消して、信頼するユーザーにのみ付与できます。

INHERIT PRIVILEGES権限の付与の構文は次のとおりです。

```
GRANT INHERIT PRIVILEGES ON USER invoking_user TO procedure_owner;
```

詳細は、次のとおりです。

- invoking_userは、実行者権限プロシージャを実行するユーザーです。このユーザーは、データベース・ユーザー・アカウントである必要があります。
- procedure_ownerは、実行者権限プロシージャを所有するユーザーです。この値は、データベース・ユーザー・アカウントである必要があります。INHERIT PRIVILEGES権限をプロシージャの所有者に付与するかわりに、プロシージャに付与されるロールに権限を付与できます。

次のユーザーまたはロールは、実行者権限プロシージャを実行するユーザーによって付与されるINHERIT PRIVILEGES権限を持つ必要があります。

- 実行者権限プロシージャを所有するユーザーまたはロール
- BEQUEATH CURRENT_USERビューを所有するユーザーまたはロール

親トピック: プロシージャ・コールおよびビュー・アクセスの実行者権限の制御

9.5.4 例: 実行するユーザーのINHERIT PRIVILEGESの付与

GRANT文で、実行するユーザーのINHERIT PRIVILEGES権限をプロシージャ所有者に付与できます。

例9-1は、実行するユーザーjwardがユーザーebrownにINHERIT PRIVILEGES権限を付与する方法を示しています。

例9-1 プロシージャ所有者への実行するユーザーのINHERIT PRIVILEGESの付与

```
GRANT INHERIT PRIVILEGES ON USER jward TO ebrown;
```

この文により、jwardが実行するとき、ebrownが書き込むまたは今後書き込む実行者権限プロシージャがjwardの権限にアクセスできます。

親トピック: プロシージャ・コールおよびビュー・アクセスの実行者権限の制御

9.5.5 例: INHERIT PRIVILEGESの取消し

REVOKE文で、ユーザーのINHERIT PRIVILEGES権限を取り消すことができます。

例9-2は、ユーザーjwardがebrownの権限の使用を取り消す方法を示しています。

例9-2 INHERIT PRIVILEGESの取消し

```
REVOKE INHERIT PRIVILEGES ON USER jward FROM ebrown;
```

親トピック: プロシージャ・コールおよびビュー・アクセスの実行者権限の制御

9.5.6 他のユーザーへのINHERIT ANY PRIVILEGES権限の付与

デフォルトでは、ユーザーSYSは、INHERIT ANY PRIVILEGESシステム権限を持ち、この権限を他のデータベース・ユーザーまたはロールに付与できます。

すべてのANY権限と同様に、信頼できるユーザーまたはロールにのみこの権限を付与します。ユーザーまたはロールにINHERIT ANY PRIVILEGES権限が付与されると、このユーザーの実行者権限プロシージャは、実行するユーザーの権限にアクセスできます。DBA_SYS_PRIVSデータ・ディクショナリ・ビューを問い合せて、INHERIT ANY PRIVILEGES権限を付与されたユーザーを確認できます。

親トピック: プロシージャ・コールおよびビュー・アクセスの実行者権限の制御

9.5.7 例: 信頼できるプロシージャ所有者へのINHERIT ANY PRIVILEGESの付与

GRANT文で、INHERIT ANY PRIVILEGES権限を信頼できるプロシージャ所有者に付与できます。

例9-3は、ユーザーebrownへのINHERIT ANY PRIVILEGES権限の付与方法を示しています。

例9-3 信頼できるプロシージャ所有者へのINHERIT ANY PRIVILEGESの付与

```
GRANT INHERIT ANY PRIVILEGES TO ebrown;
```

強力なユーザーのINHERIT ANY PRIVILEGES権限の取消しに注意してください。たとえば、ユーザーSYSTEMが一連の実行者権限プロシージャを作成したとします。SYSTEMのINHERIT ANY PRIVILEGESを取り消す場合、INHERIT PRIVILEGE権限を特に付与しないかぎり、他のユーザーはプロシージャを実行できません。

親トピック: プロシージャ・コールおよびビュー・アクセスの実行者権限の制御

9.5.8 INHERIT PRIVILEGESおよびINHERIT ANY PRIVILEGESの管理

デフォルトでは、PUBLICは、新しいユーザー・アカウントおよびアップグレードされたユーザー・アカウントのINHERIT PRIVILEGE権限を持ち、SYSユーザーは、INHERIT ANY PRIVILEGES権限を持ちます。

デフォルトでは、様々なOracleで定義されているユーザーの権限の悪用に対して保護できるよう設計されている一連のINHERIT PRIVILEGESの付与を構成します。

顧客が定義するユーザーのINHERIT PRIVILEGES ON USER user_name TO PUBLICのデフォルトの付与を取り消して、その特定のユーザーに応じて詳細な付与のINHERIT PRIVILEGESを付与できます。INHERIT ANY PRIVILEGES権限を付与されたユーザーを確認するには、DBA_SYS_PRIVSデータ・ディクショナリ・ビューを問い合せます。

1. PUBLICからINHERIT PRIVILEGES権限を取り消します。

   たとえば:

```
REVOKE INHERIT PRIVILEGES ON invoking_user FROM PUBLIC;
```

失敗したINHERIT PRIVILEGESチェックの実行時エラーのため、この時点で実行者権限プロシージャを実行するユーザーは実行できないことに注意してください。

2. INHERIT PRIVILEGES権限を信頼できるユーザーまたはロールに選択的に付与します。
3. 同様に、INHERIT ANY PRIVILEGES権限を信頼できるユーザーまたはロールにのみ選択的に付与します。

監査ポリシーを作成してこれらの2つの権限の付与および取消しを監査できますが、失敗したINHERIT PRIVILEGES権限チェックによって発生する実行時エラーは監査できません。

関連項目:

- SQLインジェクション攻撃の詳細は、『Oracle Database PL/SQLパッケージおよびタイプ・リファレンス』を参照してください。
- GRANT文およびデフォルトの権限の詳細は、『Oracle Database PL/SQLパッケージおよびタイプ・リファレンス』を参照してください。

親トピック: プロシージャ・コールおよびビュー・アクセスの実行者権限の制御

9.6 ビューの定義者権限および実行者権限

CREATE VIEW SQL文でBEQEATH句を使用して、ユーザー作成ビューで定義者権限と実行者権限を制御できます。

- ビューの定義者権限および実行者権限の制御について
  ユーザー定義ビューを構成して、ビューで参照される実行者権限関数に対応できます。
- CREATE VIEW文のBEQUEATH句の使用
  BEQUEATHは、実行ユーザーの権限を使用してどのように実行者権限関数を実行するかを制御します。
- 実行するユーザーのユーザー名またはユーザーIDの確認
  実行者権限または定義者権限を使用するかどうかに基づいて、PL/SQLファンクションを使用して、実行するユーザーを確認できます。
- BEQUEATH DEFINERおよびBEQUEATH_CURRENT_USERビューの確認
  ビューがBEQUEATH DEFINERまたはBEQUEATH CURRENT_USERビューであるかどうかを確認できます。

親トピック: 定義者権限および実行者権限のセキュリティの管理

9.6.1 ビューの定義者権限および実行者権限の制御について

ユーザー定義ビューを構成して、ビューで参照される実行者権限関数に対応できます。

ユーザーがIDまたは権限依存のSQL関数または実行者権限のPL/SQLまたはJava関数を起動すると、現在のスキーマ、現在のユーザーおよび操作の実行内の現在有効なロールがビューの所有者に設定することなく問い合せたユーザーの環境から継承できます。

この構成は、ビュー自体から実行者権限のオブジェクトに変更しません。ビュー内での名前解決は、引き続きビューの所有者のスキーマを使用して処理され、ビューの権限チェックは、ビューの所有者の権限で実行されます。ただし、実行時に、ビューで参照される関数は、ビュー所有者ではなく実行するユーザーの権限で実行されます。

この機能の利点は、関数をビューで参照する場合に一貫した結果を戻すために実行するユーザーに正確な情報を戻す必要がある SYS_CONTEXTや USERENVなどの関数を有効にすることです。

親トピック: ビューの定義者権限および実行者権限

9.6.2 CREATE VIEW文のBEQUEATH句の使用

BEQUEATHは、実行ユーザーの権限を使用してどのように実行者権限関数を実行するかを制御します。

ビューを参照するSQLを発行するユーザーの権限を使用して実行者権限関数を実行するには、CREATE VIEW文で、BEQUEATH句をCURRENT_USERに設定します。

ビューに対するSQL問合せまたはDML文を発行する場合、ビュー所有者は、実行するユーザーのINHERIT PRIVILEGES権限を付与するか、INHERIT ANY PRIVILEGES権限を持つ必要があります。そうしないと、SELECT問合せまたはDML文がBEQUEATH CURRENT_USERビューを含む場合、実行時システムは、エラー「ORA-06598: INHERIT PRIVILEGES権限が不十分です」を表示します。

- BEQUEATH CURRENT_USER句を使用して、実行者権限を使用して実行するビューの関数を設定します。

たとえば:

```
CREATE VIEW MY_OBJECTS_VIEW BEQUEATH CURRENT_USER AS
 SELECT GET_OBJS_FUNCTION;
```

ビューの所有者の権限を使用してビュー内の関数を実行する場合、BEQUEATH句を省略するか、DEFINERに設定します。

たとえば:

```
CREATE VIEW my_objects_view BEQUEATH DEFINER AS
 SELECT OBJECT_NAME FROM USER_OBJECTS;
```

関連項目:

- INHERIT PRIVILEGE権限の使用方法の詳細は、プロシージャ・コールおよびビュー・アクセスの実行者権限の制御を参照してください
- INHERIT PRIVILEGESおよびINHERIT ANY PRIVILEGES権限の付与の詳細は、『Oracle Database SQL言語リファレンス』を参照してください。
- Oracle Database Real Application SecurityアプリケーションでのBEQUEATH CURRENT_USERビューの使用方法の詳細は、『Oracle Database Real Application Security管理者および開発者ガイド』を参照してください。

親トピック: ビューの定義者権限および実行者権限

9.6.3 実行するユーザーのユーザー名またはユーザーIDの確認

実行者権限または定義者権限を使用するかどうかに基づいて、PL/SQLファンクションを使用して、実行するユーザーを確認できます。

- 実行者権限または定義者権限のどちらを使用するかに基づいて、ORA_INVOKING_USER関数またはORA_INVOKING_USERID関数を使用して、実行するユーザーを確認します。
  - ORA_INVOKING_USER: この関数を使用して、現在の文またはビューを実行しているユーザーの名前を戻します。この関数は、介在しているビューをBEQUEATH句で指定されたとみなします。実行するユーザーがOracle Database Real Application Securityで定義されているユーザーの場合、この関数はXS$NULLを戻します。

- ORA_INVOKING_USERID: この関数を使用して、現在の文またはビューを実行しているユーザーの識別子(ID)を戻します。この関数は、介在しているビューをBEQUEATH句で指定されたとみなします。実行するユーザーがOracle Database Real Application Securityで定義されているユーザーである場合、この関数は、すべてのReal Application Securityセッションに共通でデータベース・ユーザーのIDとは異なるIDを戻します。

  たとえば:

```
CONNECT HR
Enter password: password
SELECT ORA_INVOKING_USER FROM DUAL;
ORA_INVOKING_USER
--------------------
HR
```

関連項目:

Oracle Database Real Application Securityアプリケーションに使用される類似の関数の詳細は、『Oracle Database Real Application Security管理者および開発者ガイド』を参照してください。

親トピック: ビューの定義者権限および実行者権限

9.6.4 BEQUEATH DEFINERおよびBEQUEATH_CURRENT_USERビューの確認

ビューがBEQUEATH DEFINERまたはBEQUEATH CURRENT_USERビューであるかどうかを確認できます。

- ビューがBEQUEATH DEFINERまたはBEQUEATH CURRENT_USERビューであるかどうかを確認するには、そのビューの*_VIEWSまたは*_VIEWS_AE静的データ・ディクショナリ・ビューのBEQUEATH列を問い合せます。

関連項目:

- 静的データ・ディクショナリ・ビュー*_VIEWSの詳細は、『Oracle Databaseリファレンス』を参照してください。
- 静的データ・ディクショナリ・ビュー*_VIEWS_AEの詳細は、『Oracle Databaseリファレンス』を参照してください。

たとえば:

```
SELECT BEQUEATH FROM USER_VIEWS WHERE VIEW_NAME = 'MY_OBJECTS';
BEQUEATH
------------
CURRENT_USER
```

親トピック: ビューの定義者権限および実行者権限

9.7 定義者権限および実行者権限のコード・ベース・アクセス制御の使用

データベース・ロールをPL/SQLファンクション、プロシージャまたはパッケージに付与するときに使用するコード・ベース・アクセス制御は、定義者権限および実行者権限のプロシージャと一緒に使用すると効果的です。

- アプリケーションのコード・ベース・アクセス制御の使用について
  コード・ベース・アクセス制御(CBAC)を使用すると、定義者権限のプログラム・ユニットの管理を向上できます。

- コード・ベースのアクセス制御ロールをプログラム・ユニットに付与できる者
  次のすべての条件を満たした場合、コード・ベースのアクセス制御ロールをプログラム・ユニットに付与できます。
- コード・ベース・アクセス制御による実行者権限のプログラム・ユニットの処理方法
  コード・ベース・アクセス制御では、実行ユーザーのコンテキストで、そのコンテキストに関連付けられたロールを使用してプログラム・ユニットを実行できます。
- コード・ベース・アクセス制御による定義者権限のプログラム・ユニットの処理方法
  コード・ベース・アクセス制御を使用して定義者権限を保護できます。
- CBAC付与のためのユーザーへのデータベース・ロールの付与
  GRANT文のDELEGATEオプションで、CBAC付与を行うユーザーによるロールへの権限付与を制限できます。
- プログラム・ユニットに対するデータベース・ロールの付与と取消し
  GRANTおよびREVOKE文で、プログラム・ユニットに対するデータベース・ロールの付与または取消しを行うことができます。
- チュートリアル: コード・ベース・アクセス制御による機密データへのアクセス制御
  このチュートリアルでは、コード・ベース・アクセス制御を使用して、HRスキーマの機密データへのアクセスを制御する方法を示します。

親トピック: 定義者権限および実行者権限のセキュリティの管理

9.7.1 アプリケーションのコード・ベース・アクセス制御の使用について

コード・ベース・アクセス制御(CBAC)を使用すると、実行者権限のプログラム・ユニットの管理を向上できます。

アプリケーションは、昇格した権限を必要とする一方で、コール側の環境でプログラム・ユニットを頻繁に実行する必要があります。従来PL/SQLプログラムは、定義者権限を使用してプログラムの権限を一時的に昇格します。

ただし、定義者権限ベースのプログラム・ユニットは、実行者のコンテキストではなくプログラム・ユニットの定義者または所有者のコンテキストで実行されます。また、定義者権限ベースのプログラムを使用すると、多くの場合にプログラム・ユニットが必要以上の権限を取得します。

コード・ベース・アクセス制御(CBAC)は、PL/SQLファンクション、プロシージャまたはパッケージへのデータベース・ロールのアタッチを可能にして、解決策を提供します。これらのデータベース・ロールは実行時に有効で、呼び出すユーザーの環境の必要な権限でプログラム・ユニットを実行できます。

CBACロールの使用を取得する権限分析ポリシーを作成できます。

関連トピック

- 権限分析の実行による権限使用の特定

親トピック: 定義者権限および実行者権限のコード・ベース・アクセス制御の使用

9.7.2 コード・ベースのアクセス制御ロールをプログラム・ユニットに付与できる者

次のすべての条件を満たした場合、コード・ベースのアクセス制御ロールをプログラム・ユニットに付与できます。

これらの条件は次のとおりです。

- 権限付与者は、ユーザーSYS、またはプログラム・ユニットの所有者です。
- 権限付与者がプログラム・ユニットを所有する場合、権限付与者はGRANT ANY ROLEシステム権限を持つか、プログラム・ユニットに付与するロールに対してADMINまたはDELEGATEオプションを持つ必要があります。
- 付与対象のロールは、所有者に対して直接付与されるロールです。

- 付与対象のロールは、標準データベース・ロールです。

これらの3つの条件が満たされない場合、1番目の条件が満たされないときは、エラーORA-28702: プログラム・ユニットの文字列が権限付与者によって所有されていませんが生成され、2番目と3番目の条件が満たされないときは、エラーORA-1924: ロールの文字列は付与されていないか、存在していませんが生成されます。

関連トピック

- CBAC付与のためのユーザーへのデータベース・ロールの付与
- プログラム・ユニットに対するデータベース・ロールの付与と取消し

親トピック: 定義者権限および実行者権限のコード・ベース・アクセス制御の使用

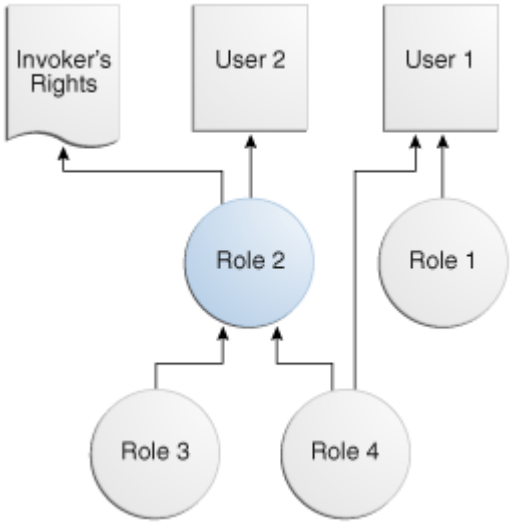
9.7.3 コード・ベース・アクセス制御による実行者権限のプログラム・ユニットの処理方法

コード・ベース・アクセス制御では、実行ユーザーのコンテキストで、そのコンテキストに関連付けられたロールを使用してプログラム・ユニットを実行できます。

2つのアプリケーション・ユーザー1および2が存在するシナリオを検討します。アプリケーション・ユーザー2は、実行者権限のプログラム・ユニットを作成し、データベース・ロール2を実行者権限ユニットに付与して、実行者権限ユニットの実行権限をアプリケーション・ユーザー1に付与します。

次の図は、アプリケーション・ユーザー1および2に付与されるデータベース・ロール1および2と、実行者権限のプログラム・ユニットを示しています。

図9-1 アプリケーション・ユーザーに付与されるロールおよび実行者権限のプログラム・ユニット



付与は次のとおりです。

- アプリケーション・ユーザー1に直接データベース・ロール1および4が付与されます。
- アプリケーション・ユーザー2に直接アプリケーション・ロール3および4を含むデータベース・ロール2が付与されます。
- 実行者権限のプログラム・ユニットにデータベース・ロール2が付与されます。

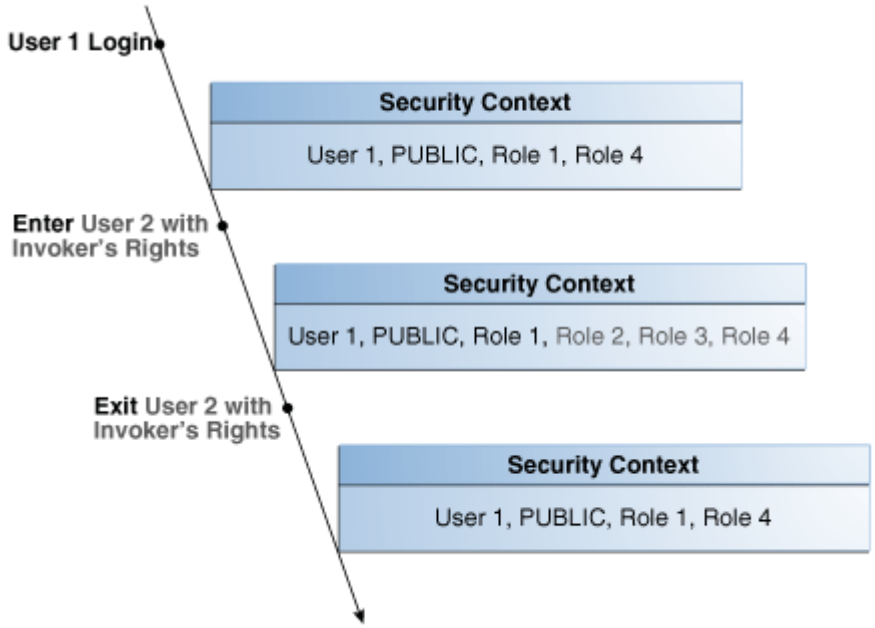
アプリケーション・ユーザー1がログインして実行者権限のプログラム・ユニットを実行する場合、実行者権限ユニットは、ユーザー1の結合されたデータベース・ロールおよび実行者権限ユニットに付加されたデータベース・ロールで実行されます。

次の図は、実行者権限のユニットが実行されるセキュリティ・コンテキストを示しています。アプリケーション・ユーザー1が最初にログオンする場合、アプリケーション・ユーザー1は、データベースPUBLICロール(デフォルト)およびそれに付与されたデータベース・ロール1および4を持ちます。次に、アプリケーション・ユーザー1は、アプリケーション・ユーザー2によって作成された実行者権限の

プログラム・ユニットを実行します。

実行者権限のユニットは、アプリケーション・ユーザー1のコンテキストで実行され、それに付加される追加のデータベース・ロール2を持ちます。データベース・ロール2の一部であるため、データベース・ロール3および4が含まれます。実行者権限ユニットを終了した後、アプリケーション・ユーザー1のみ、それに付与されたアプリケーション・ロール、PUBLIC、ロール1およびロール4を持ちます。

図9-2 実行者権限のプログラム・ユニットIRが実行されるセキュリティ・コンテキスト



親トピック: 定義者権限および実行者権限のコード・ベース・アクセス制御の使用

9.7.4 コード・ベース・アクセス制御による定義者権限のプログラム・ユニットの処理方法

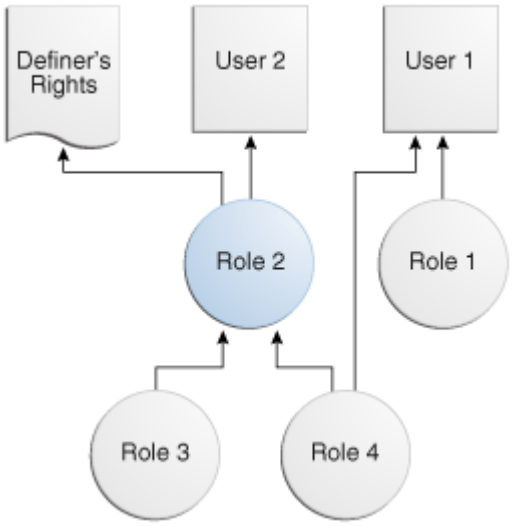
コード・ベース・アクセス制御を使用して定義者権限を保護できます。

コード・ベース・アクセス制御は、定義するユーザーの権限で動作するプログラム・ユニットを有効にすることで定義者権限のプログラム・ユニットと連携し、このユーザーに関連付けられるデータベース・ロールを組み合せた権限で機能します。

アプリケーション・ユーザー2が定義者権限のプログラム・ユニットを作成し、定義者権限のプログラム・ユニットにロール2を付与して、定義者権限のプログラム・ユニットのEXECUTE権限をアプリケーション・ユーザー1に付与するシナリオを検討します。

次の図は、アプリケーション・ユーザー1および2に付与されるデータベース・ロールと、定義者権限のプログラム・ユニットを示しています。

図9-3 アプリケーション・ユーザーに付与されるロールおよび定義者権限のプログラム・ユニット



付与は次のとおりです。

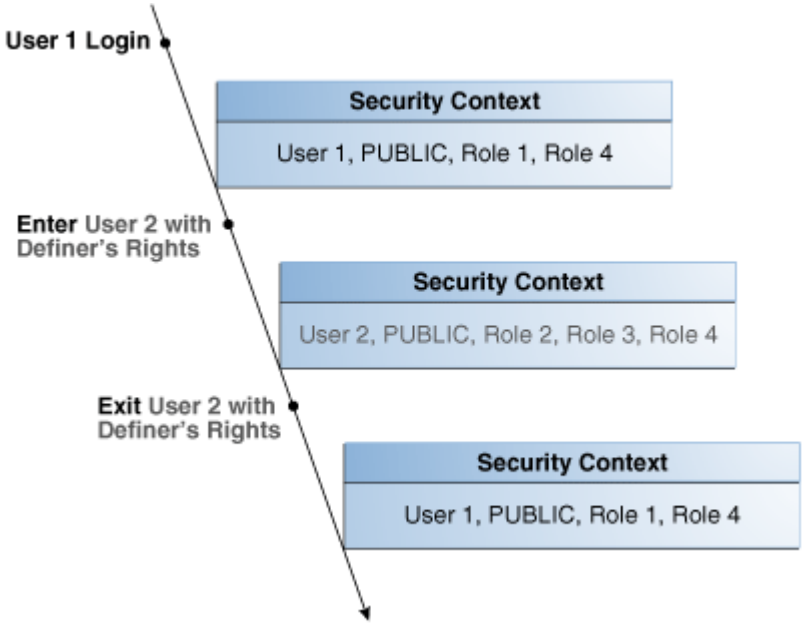
- アプリケーション・ユーザー1に直接データベース・ロール1および4が付与されます。
- アプリケーション・ユーザー2に直接データベース・ロール3および4を含むデータベース・ロール2が付与されます。
- 定義者権限のプログラム・ユニットにデータベース・ロール2が付与されます。

アプリケーション・ユーザー1がログインして定義者権限のプログラム・ユニットを実行する場合、定義者権限ユニットは、アプリケーション・ユーザー2の結合されたデータベース・ロールおよび定義者権限ユニットに付加されたデータベース・ロール(ロール2、3および4)で実行されます。

次の図は、定義者権限のプログラム・ユニットが実行されるセキュリティ・コンテキストを示しています。アプリケーション・ユーザー1が最初にログオンする場合、アプリケーション・ユーザー1は、データベースPUBLICロール(デフォルト)およびそれに付与されたデータベース・ロール1および4を持ちます。次に、アプリケーション・ユーザー1は、アプリケーション・ユーザー2によって作成された定義者権限のプログラム・ユニットを実行します。

定義者権限のプログラム・ユニットは、アプリケーション・ユーザー2のコンテキストで実行され、それに付加される追加のデータベース・ロール2を持ちます。データベース・ロール2の一部であるため、データベース・ロール3および4が含まれます。定義者権限ユニットを終了した後、アプリケーション・ユーザー1のみ、それに付与されたデータベース・ロール(PUBLIC、ロール1およびロール4)を持ちます。

図9-4 定義者権限のプログラム・ユニットDRが実行されるセキュリティ・コンテキスト



親トピック: 定義者権限および実行者権限のコード・ベース・アクセス制御の使用

9.7.5 CBAC付与のためのユーザーへのデータベース・ロールの付与

GRANT文のDELEGATEオプションで、CBAC付与を行うユーザーによるロールへの権限付与を制限できます。

データベース・ロールをCBAC付与を行うユーザーに付与する場合、GRANT文にDELEGATEオプションを含めて、権限受領者にロールに対する追加権限が付与されないようにすることができます。

DELEGATEオプションを使用すると、ロールはプログラム・ユニットに付与されますが、他のプリンシパルまたはロール自体の管理にロールを付与することはできません。他のプリンシパルへのロールの付与を可能にするADMINオプションを付与に使用することもできます。ADMINとDELEGATEオプションは共存できます。オプションごとに別のGRANT文で付与する必要はありますが、両方をユーザーに付与することができます。これらのオプション付きでユーザーにロールが付与されているかどうかを確認するには、ユーザーのUSER_ROLE_PRIVSまたはDBA_ROLE_PRIVSのDELEGATE_OPTION列またはADMIN_OPTION列を問い合せます。

DELEGATEおよびADMINオプションを使用するための構文は次のとおりです。

```
GRANT role_list to user_list WITH DELEGATE OPTION;
GRANT role_list to user_list WITH ADMIN OPTION;
```

たとえば:

```
GRANT cb_role1 to usr1 WITH DELEGATE OPTION;
GRANT cb_role1 to usr1 WITH ADMIN OPTION;
GRANT cb_role1, cb_role2 to usr1, usr2 with DELEGATE OPTION;
GRANT cb_role1, cb_role2 to usr1, usr2 with ADMIN OPTION;
```

マルチテナント環境では、ADMINオプションの場合と同様に共通ロールの共通ユーザーへの付与などの共通付与にDELEGATEオプションを使用できます。

たとえば:

```
GRANT c##cb_role1 to c##usr1 WITH DELEGATE OPTION CONTAINER = ALL;
```

CBAC付与自体は、PDBでローカルにのみ行えることに注意してください。

関連項目:

ADMINオプションの詳細は、『Oracle Database SQL言語リファレンス』を参照してください。

親トピック: 定義者権限および実行者権限のコード・ベース・アクセス制御の使用

9.7.6 プログラム・ユニットに対するデータベース・ロールの付与と取消し

GRANTおよびREVOKE文で、プログラム・ユニットに対するデータベース・ロールの付与または取消しを行うことができます。

次の構文を使用して、PL/SQLファンクション、プロシージャまたはパッケージのデータベース・ロールを付与または取り消します。

```
GRANT role_list TO code_list
REVOKE {role_list | ALL} FROM code_list
```

詳細は、次のとおりです。

```
role_list ::=  code-based_role_name[, role_list]
code_list ::=  {
```

```
      {FUNCTION  [schema.]function_name}
   |  {PROCEDURE [schema.]procedure_name}
   |  {PACKAGE   [schema.]package_name}
                }[, code_list]
```

たとえば:

```
GRANT cb_role1 TO FUNCTION func1, PACKAGE pack1;
GRANT cb_role2, cb_role3 TO FUNCTION HR.func2, PACKAGE SYS.pack2;
REVOKE cb_role1 FROM FUNCTION func1, PACKAGE pack1;
REVOKE ALL FROM FUNCTION HR.func2, PACKAGE SYS.pack2;
```

関連トピック

- コード・ベースのアクセス制御ロールをプログラム・ユニットに付与できる者
- CBAC付与のためのユーザーへのデータベース・ロールの付与

親トピック: 定義者権限および実行者権限のコード・ベース・アクセス制御の使用

9.7.7 チュートリアル: コード・ベース・アクセス制御による機密データへのアクセス制御

このチュートリアルでは、コード・ベース・アクセス制御を使用して、HRスキーマの機密データへのアクセスを制御する方法を示します。

- このチュートリアルについて
  このチュートリアルでは、自分の部門のために特定の従業員情報に対するアクセス権が必要なユーザーを作成します。
- ステップ1: ユーザーを作成してHRにCREATE ROLE権限を付与
  開始するには、"Finance"ユーザー・アカウントを作成し、このHRユーザーにCREATE ROLE権限を付与する必要があります。
- ステップ2: print_employees実行者権限プロシージャを作成
  print_employees実行者権限プロシージャは、現在のユーザーの部門内の従業員情報を表示します。
- ステップ3: hr_clerkロールを作成して権限を付与
  次に、hr_clerkロールを作成し、print_employeesプロシージャに対するEXECUTE権限を付与する必要があります。
- ステップ4: コード・ベース・アクセス制御HR.print_employeesプロシージャのテスト
  コード・ベース・アクセス制御HR.print_employeesプロシージャをテストする準備が整いました。
- ステップ5: view_emp_roleロールを作成して権限を付与
  ここでは、ユーザーHRでview_emp_roleロールを作成し、このロールに権限を付与する必要があります。
- ステップ6: HR.print_employeesプロシージャの再テスト
  適切な権限がある場合、ユーザー"Finance"は、HR.print_employeesプロシージャを再試行できます。
- ステップ7: このチュートリアルのコンポーネントの削除
  このチュートリアルのコンポーネントが不要になった場合、それらを削除できます。

親トピック: 定義者権限および実行者権限のコード・ベース・アクセス制御の使用

9.7.7.1 このチュートリアルについて

このチュートリアルでは、自分の部門のために特定の従業員情報に対するアクセス権が必要なユーザーを作成します。

ただし、HR.EMPLOYEES表には従業員給与などの機密情報が含まれており、ユーザーにはアクセスできないようにする必要があります。アクセス制御は、コード・ベース・アクセス制御を使用して実装します。従業員データは、実行者権限プロシージャを通じてユーザーに表示されます。SELECT権限をユーザーに直接付与するかわりに、データベース・ロールを通じてSELECT権限を

実行者権限プロシージャに付与します。このプロシージャでは、給与のような機密情報を非表示にします。このプロシージャは実行者権限プロシージャであるため、プロシージャ内で呼出し元のコンテキストがわかります。この場合、呼出し元のコンテキストは財務部門です。ユーザーの名前は"Finance"であるため、ユーザーは財務部門に勤務する従業員のデータのみにアクセス可能です。

親トピック: チュートリアル: コード・ベース・アクセス制御による機密データへのアクセス制御

9.7.7.2 ステップ1: ユーザーを作成してHRにCREATE ROLE権限を付与

開始するには、"Finance"ユーザー・アカウントを作成し、このHRユーザーにCREATE ROLE権限を付与する必要があります。

1. ユーザー・アカウントおよびロールを作成する権限を持つ管理者としてデータベース・インスタンスにログインします。

   たとえば:

   ```
   sqlplus sec_admin
   Enter password: password
   ```

2. "Finance"ユーザー・アカウントを作成します。

   ```
   GRANT CONNECT TO "Finance" IDENTIFIED BY password;
   ```

   "Finance"が、大文字と小文字はこのまま、二重引用符で囲んで入力されていることを確認します。「パスワードの最低要件」のガイドラインに従って、passwordを安全なパスワードに置き換えます。

3. CREATE ROLE権限をユーザーHRに付与します。

   ```
   GRANT CREATE ROLE TO HR;
   ```

親トピック: チュートリアル: コード・ベース・アクセス制御による機密データへのアクセス制御

9.7.7.3 ステップ2: print_employees実行者権限プロシージャを作成

print_employees実行者権限プロシージャは、現在のユーザーの部門内の従業員情報を表示します。

プロシージャ内で呼出し元がだれであるかを知る必要があるため、このプロシージャを実行者権限プロシージャとして作成する必要があります。

1. ユーザーHRとして接続します。

   ```
   CONNECT HR
   Enter password: password
   ```

2. 次のようにprint_employeesプロシージャを作成します。

   ```
   create or replace procedure print_employees
   authid current_user
   as
   begin
     dbms_output.put_line(rpad('ID', 10) ||
                          rpad('First Name', 15)  ||
                          rpad('Last Name', 15)   ||
                          rpad('Email', 15)       ||
                          rpad('Phone Number', 20));
     for rec in (select e.employee_id, e.first_name, e.last_name,
                        e.email, e.phone_number
                   from hr.employees e, hr.departments d
                  where e.department_id = d.department_id
                    and d.department_name =
                        sys_context('userenv', 'current_user'))
     loop
   ```

```
    dbms_output.put_line(rpad(rec.employee_ID, 10)  ||
                         rpad(rec.first_name, 15)   ||
                         rpad(rec.last_name, 15)    ||
                         rpad(rec.email, 15)        ||
                         rpad(rec.phone_number, 20));
  end loop;
end;
/
```

この例では、次のようになります。

- dbms_output.put_lineは、表ヘッダーを印刷します。
- for rec in (select ...は、呼出し元の部門の従業員情報を検索します。このチュートリアルでは、ユーザー"Finance"の財務部門となります。"Marketing"(これもHR.EMPLOYEES表のDEPARTMENT_NAME列にリストされています)という名前のユーザーを作成していた場合には、プロシージャはマーケティング部門従業員の情報を取得できました。
- loopおよびdbms_output.put_lineは、出力に財務部門の従業員データを移入します。

親トピック: チュートリアル: コード・ベース・アクセス制御による機密データへのアクセス制御

9.7.7.4 ステップ3: hr_clerkロールを作成して権限を付与

次に、hr_clerkロールを作成し、print_employeesプロシージャに対するEXECUTE権限を付与する必要があります。

このロールを作成したら、これを"Finance"に付与する必要があります。

1. hr_clerkロールを作成します。

```
CREATE ROLE hr_clerk;
```

2. print_employeesプロシージャのEXECUTE権限をhr_clerkロールに付与します。

```
GRANT EXECUTE ON print_employees TO hr_clerk;
```

3. hr_clerkロールを"Finance"に付与します。

```
GRANT hr_clerk TO "Finance";
```

親トピック: チュートリアル: コード・ベース・アクセス制御による機密データへのアクセス制御

9.7.7.5 ステップ4: コード・ベース・アクセス制御HR.print_employeesプロシージャのテスト

コード・ベース・アクセス制御HR.print_employeesプロシージャをテストする準備が整いました。

コード・ベース・アクセス制御のHR.print_employeesプロシージャをテストするには、ユーザー"Finance"がHR.EMPLOYEES表を問い合せて、HR.print_employeesプロシージャの実行を試みる必要があります。

1. データベース・インスタンスにユーザー"Finance"として接続します。

```
CONNECT "Finance"
Enter password: password
```

2. HR.EMPLOYEES表の直接問合せを試行します。

```
SELECT EMPLOYEE_ID, FIRST_NAME, LAST_NAME, SALARY FROM HR.EMPLOYEES;
```

ユーザーFinanceにはHR.EMPLOYEESへのSELECT権限がないため、問合せは失敗します。

```
ERROR at line 1:
```

```
ORA-00942: table or view does not exist
```

3. HR.print_employeesプロシージャを実行します。

```
EXEC HR.print_employees;
```

ユーザー"Finance"が適切な権限を持っていないため、問合せは失敗します。

```
ERROR at line 1:
ORA-00942: table or view does not exist
ORA-06512: at "HR.PRINT_EMPLOYEES", line 13ORA-06512: at line 1
```

親トピック: チュートリアル: コード・ベース・アクセス制御による機密データへのアクセス制御

9.7.7.6 ステップ5: view_emp_roleロールを作成して権限を付与

ここでは、ユーザーHRでview_emp_roleロールを作成し、このロールに権限を付与する必要があります。

ユーザーHRがSELECT権限HR.EMPLOYEESとHR.DEPARTMENTSをview_emp_roleロールに付与し、HR.EMPLOYEESとHR.DEPARTMENTSのSELECTをview_emp_roleロールに付与します。

1. ユーザーHRとして接続します。

```
CONNECT HR
Enter password: password
```

2. view_emp_roleロールを作成します。

```
CREATE ROLE view_emp_role;
```

3. view_emp_roleロールにHR.EMPLOYEESとHR.DEPARTMENTSのSELECT権限を付与します。

```
GRANT SELECT ON HR.EMPLOYEES TO view_emp_role;
GRANT SELECT ON HR.DEPARTMENTS TO view_emp_role;
```

4. view_emp_roleロールをHR.print_employees実行者権限プロシージャに付与します。

```
GRANT view_emp_role TO PROCEDURE HR.print_employees;
```

親トピック: チュートリアル: コード・ベース・アクセス制御による機密データへのアクセス制御

9.7.7.7 ステップ6: HR.print_employeesプロシージャの再テスト

適切な権限がある場合、ユーザー"Finance"は、HR.print_employeesプロシージャを再試行できます。

1. ユーザー"Finance"として接続します。

```
CONNECT "Finance"
Enter password: password
```

2. サーバー出力をディスプレイに設定します。

```
SET SERVEROUTPUT ON;
```

3. HR.EMPLOYEES表の直接問合せを試行します。

```
SELECT EMPLOYEE_ID, FIRST_NAME, LAST_NAME, SALARY FROM HR.EMPLOYEES;
```

問合せが失敗します。

```
ERROR at line 1:
ORA-00942: table or view does not exist
```

4. HR.print_employeesプロシージャを実行して、従業員情報を表示します。

```
EXEC HR.print_employees;
```

呼出しが成功します。

```
ID         First Name     Last Name      Email          Phone Number
108        Nancy          Greenberg      NGREENBE       515.124.4569
109        Daniel         Faviet         DFAVIET        515.124.4169
110        John           Chen           JCHEN          515.124.4269
111        Ismael         Sciarra        ISCIARRA       515.124.4369
112        Jose Manuel    Urman          JMURMAN        515.124.4469
113        Luis           Popp           LPOPP          515.124.4567
PL/SQL procedure successfully completed.
```

親トピック: チュートリアル: コード・ベース・アクセス制御による機密データへのアクセス制御

9.7.7.8 ステップ7: このチュートリアルのコンポーネントの削除

このチュートリアルのコンポーネントが不要になった場合、それらを削除できます。

1. 管理者権限を持つユーザーとして接続します。

   たとえば:

```
CONNECT sec_admin
Enter password: password
```

2. ユーザー"Finance"を削除します。

```
DROP USER "Finance";
```

3. hr_clerkロールを削除します。

```
DROP ROLE hr_clerk;
```

4. ユーザーHRとして接続します。

```
CONNECT HR
Enter password: password
```

5. view_emp_roleロールとHR.print_employeesプロシージャを削除します。

```
DROP ROLE view_emp_role;
DROP PROCEDURE print_employees;
```

6. 管理者権限ユーザーとして接続します。

```
CONNECT sec_admin
Enter password: password
```

7. HRからCREATE ROLE権限を取り消します。

```
REVOKE CREATE ROLE FROM HR;
```

親トピック: チュートリアル: コード・ベース・アクセス制御による機密データへのアクセス制御

9.8 データベース・リンクの定義者権限の制御

アプリケーションでデータベース・リンクと定義者権限プロシージャが使用されている場合は、定義者権限プロシージャの権限付与を制御できます。

- データベース・リンクの定義者権限の制御について
  定義者権限プロシージャがデータベース・リンクに接続する場合、データベース・リンク上の操作でプロシージャ所有者の資格証明を使用する必要があります。
- 他のユーザーへのINHERIT REMOTE PRIVILEGES権限の付与
  INHERIT REMOTE PRIVILEGES権限は、現行ユーザーがデータベースの接続ユーザーを介して明示的な権限を持つことを可能にします。
- 例: 接続ユーザーのINHERIT REMOTE PRIVILEGESの付与
  接続ユーザーのINHERIT REMOTE PRIVILEGES権限を現行ユーザーに付与できます。
- 他のユーザーへのINHERIT ANY REMOTE PRIVILEGES権限の付与
  INHERIT ANY REMOTE PRIVILEGES権限により、権限受領ユーザーはconnected_userデータベース・リンクを任意のユーザーとしてオープンできます。
- INHERIT [ANY] REMOTE PRIVILEGES権限の取消し
  INHERIT REMOTE PRIVILEGES権限とINHERIT ANY REMOTE PRIVILEGES権限とでは取消方法が異なります。
- 例: INHERIT REMOTE PRIVILEGES権限の取消し
  REVOKE SQL文で、INHERIT REMOTE PRIVILEGES権限を取り消すことができます。
- 例: PUBLICからのINHERIT REMOTE PRIVILEGES権限の取消し
  REVOKE SQL文で、PUBLICおよび個々のプロシージャ所有者のINHERIT REMOTE PRIVILEGESを取り消すことができます。
- チュートリアル: 定義者権限プロシージャでのデータベース・リンクの使用
  このチュートリアルでは、データベース・リンクを使用する定義者権限プロシージャでのINHERIT REMOTE PRIVILEGES権限の使用方法を示します。

親トピック: 定義者権限および実行者権限のセキュリティの管理

9.8.1 データベース・リンクの定義者権限の制御について

定義者権限プロシージャがデータベース・リンクに接続する場合、データベース・リンク上の操作でプロシージャ所有者の資格証明を使用する必要があります。

INHERIT REMOTE PRIVILEGESおよびINHERIT ANY REMOTE PRIVILEGES権限は、接続ユーザー・データベース・リンクが定義者権限プロシージャで使用される場合に適用されます。これらの権限により、定義者権限プロシージャでの接続ユーザー・データベース・リンクの操作で、ログインしているユーザーの資格証明の使用が可能になります。

定義者権限プロシージャを実行するユーザーが定義者権限ブロック内で接続ユーザー・データベース・リンクを使用できるように、INHERIT REMOTE PRIVILEGESおよびINHERIT ANY REMOTE PRIVILEGES権限を付与できます。定義者権限プロシージャは、プロシージャ所有者の権限で実行されます。ただし、接続ユーザー・データベース・リンクの操作には、ログインしているユーザーの資格証明が必要です。したがって、定義者権限内でデータベース・リンクの操作を可能にするには、INHERIT REMOTE PRIVILEGESおよびINHERIT ANY REMOTE PRIVILEGES権限を付与する必要があります。

アップグレード中にINHERIT REMOTE PRIVILEGESおよびINHERIT ANY REMOTE PRIVILEGES権限はデフォルトで既存のユーザーに付与されないことに注意してください。

INHERIT REMOTE PRIVILEGESおよびINHERIT ANY REMOTE PRIVILEGES権限は、ユーザーが定義者権限プロシージャでユーザー・データベース・リンクに接続を試みる状況にのみ適用されます。また、これらの権限は、プライベート、パブリックを問わず、作成されたデータベース・リンクに適用されます。デフォルトでは、データベース・リンクはプライベート・リンクとして作成されます。さらに、デフォルトではINHERIT REMOTE PRIVILEGESはPUBLICに付与されません。

これらの権限を付与する方法は次のとおりです。

- `GRANT INHERIT REMOTE PRIVILEGES ON USER dbuser_1 TO dbuser_2`: このシナリオでは、`dbuser_1`が明示的に`INHERIT REMOTE PRIVILEGE`権限を`dbuser_2`に付与し、ユーザー`dbuser_2`が所有する定義者権限プロシージャを使用します。
- `GRANT INHERIT REMOTE PRIVILEGES ON USER dbuser_1 TO PUBLIC`。このシナリオでは、`dbuser_1`が`INHERIT REMOTE PRIVILEGE`権限をpublicに付与します。この付与により、`dbuser_1`は他のユーザーが所有する定義者権限プロシージャを使用できます。
- `GRANT INHERIT ANY REMOTE PRIVILEGES TO dbuser_2`: このシナリオでは、いずれのユーザーも`dbuser_2`が所有する定義者権限プロシージャを使用できます。

`INHERIT REMOTE PRIVILEGE`権限のないユーザーが定義者権限を実行しようとすると、`ORA-25433: ユーザーには INHERIT REMOTE PRIVILEGESがありません`エラーが表示されます。

親トピック: データベース・リンクの定義者権限の制御

9.8.2 他のユーザーへのINHERIT REMOTE PRIVILEGES権限の付与

`INHERIT REMOTE PRIVILEGES`権限は、現行ユーザーがデータベースの接続ユーザーを介して明示的な権限を持つことを可能にします。

`INHERIT REMOTE PRIVILEGES`権限を付与する構文は次のとおりです。

```
GRANT INHERIT REMOTE PRIVILEGES ON USER connected_user TO current_user:
```

詳細は、次のとおりです。

- `connected_user`は、定義者権限プロシージャを実行するユーザーです。
- `current_user`は、定義者権限プロシージャを所有するユーザーです。この値は、データベース・ユーザー・アカウントである必要があります。`INHERIT REMOTE PRIVILEGES`権限をプロシージャの所有者に付与するかわりに、プロシージャに付与されるロールに権限を付与できます。

定義者権限プロシージャを所有するユーザーまたはロールは、定義者権限プロシージャを実行するユーザーによって付与される`INHERIT REMOTE PRIVILEGES`権限を持つ必要があります。

いずれのユーザーも、自分が実行する定義者権限プロシージャを所有するユーザーに対して、自分の持つ`INHERIT REMOTE PRIVILEGES`権限の付与または取消しを行うことができます。

親トピック: データベース・リンクの定義者権限の制御

9.8.3 例: 接続ユーザーのINHERIT REMOTE PRIVILEGESの付与

接続ユーザーの`INHERIT REMOTE PRIVILEGES`権限を現行ユーザーに付与することができます。

この例では、接続ユーザー`jward`は、現行ユーザー`ebrown`のリモート権限を持つ必要があります。これにより、`jward`は、`ebrown`が作成した定義者権限プロシージャを実行できます。

例9-4に、管理者(またはユーザー`jward`)がユーザー`jward`の`INHERIT REMOTE PRIVILEGES`をユーザー`ebrown`に付与する方法を示します。この権限付与により、`ebrown`が記述する(または今後記述する予定の)定義者権限プロシージャが、そのプロシージャの実行時に`ebrown`の権限にアクセスできるようになります。

例9-4 現行ユーザーへの接続ユーザーのINHERIT REMOTE PRIVILEGESの付与

```
GRANT INHERIT REMOTE PRIVILEGES on user jward to ebrown;
```

親トピック: データベース・リンクの定義者権限の制御

9.8.4 他のユーザーへのINHERIT ANY REMOTE PRIVILEGES権限の付与

INHERIT ANY REMOTE PRIVILEGES権限により、権限受領ユーザーはconnected_userデータベース・リンクを任意のユーザーとしてオープンできます。

すべてのANY権限と同様に、INHERIT ANY REMOTE PRIVILEGESは、信頼できるユーザーのみに付与する必要がある強力な権限です。デフォルトでは、ユーザーSYSがINHERIT ANY REMOTE PRIVILEGESシステム権限WITH GRANT OPTIONを持っています。INHERIT ANY REMOTE PRIVILEGES権限を付与されたユーザーを確認するには、DBA_SYS_PRIVSデータ・ディクショナリ・ビューを問い合せます。

マルチテナント環境でのセキュリティ向上のため、PDBロックダウン・プロファイルでINHERIT ANY REMOTE PRIVILEGES権限を保護することをお薦めします。PDBロックダウン・プロファイルは、ローカル・プラガブル・データベース(PDB)ユーザーが持っているINHERIT REMOTE PRIVILEGEの種類に関係なく、PDBユーザーが接続ユーザー・データベース・リンクを共通ユーザーとしてオープンすることを防ぎます。PDBがPDBロックダウン・プロファイルによって保護されている場合、GRANT INHERIT REMOTE PRIVILEGESおよびGRANT INHERIT ANY REMOTE権限などの付与は成功しますが、これらの付与の影響は、PDBロックダウンが継続しているかぎり適用されません。

INHERIT ANY REMOTE PRIVILEGES権限を付与する構文は次のとおりです。

```
GRANT INHERIT ANY REMOTE PRIVILEGES TO current_user;
```

ここで、current_userは、定義者権限プロシージャを所有するユーザーです。

関連トピック

- PDBロックダウン・プロファイルを使用したPDBでの操作の制限

親トピック: データベース・リンクの定義者権限の制御

9.8.5 INHERIT [ANY] REMOTE PRIVILEGES権限の取消し

INHERIT REMOTE PRIVILEGES権限とINHERIT ANY REMOTE PRIVILEGES権限とでは取消方法が異なります。

INHERIT REMOTE PRIVILEGES権限は、ユーザーが別のユーザーから取り消すことができます。INHERIT ANY REMOTE PRIVILEGES権限は管理権限を持つユーザーが取り消す必要があります。

取消しの構文は次のとおりです

```
REVOKE INHERIT REMOTE PRIVILEGES ON USER connected_user FROM current_user;
```

詳細は、次のとおりです。

- connected_userは、定義者権限プロシージャを実行するユーザーです。
- current_userは、定義者権限プロシージャを所有するユーザーです。

INHERIT REMOTE PRIVILEGESまたはINHERIT ANY REMOTE PRIVILEGES権限をユーザーから取り消す場合、次のように標準の取消し構文を使用します。

```
REVOKE INHERIT REMOTE PRIVILEGES FROM connected_user;
REVOKE INHERIT ANY REMOTE PRIVILEGES FROM current_user;
```

関連項目:

REVOKE SQL文の詳細は、『Oracle Database SQL言語リファレンス』を参照してください。

親トピック: データベース・リンクの定義者権限の制御

9.8.6 例: INHERIT REMOTE PRIVILEGES権限の取消し

REVOKE SQL文で、INHERIT REMOTE PRIVILEGES権限を取り消すことができます。

INHERIT REMOTE PRIVILEGES権限を取り消すと、ユーザーjwardがjward所有の定義者権限プロシージャを実行する場合、jwardがebrownに対して、jwardの資格証明を使用して接続ユーザー・データベース・リンクをオープンする権限を明示的に拒否しているため、定義者権限プロシージャ内での接続ユーザー・データベース・リンクの操作はすべて失敗します。

例9-5に、接続ユーザーjwardのINHERIT REMOTE PRIVILEGESプロシージャをプロシージャ所有者ebrownから取り消す方法を示します。

例9-5 INHERIT REMOTE PRIVILEGES権限の取消し

```
REVOKE INHERIT REMOTE PRIVILEGES ON USER jward FROM ebrown;
```

親トピック: データベース・リンクの定義者権限の制御

9.8.7 例: PUBLICからのINHERIT REMOTE PRIVILEGES権限の取消し

REVOKE SQL文で、PUBLICおよび個々のプロシージャ所有者のINHERIT REMOTE PRIVILEGESを取り消すことができます。

例9-6に、PUBLICから権限を取り消す方法を示します。

例9-6 PUBLICからのINHERIT REMOTE PRIVILEGES権限の取消し

```
REVOKE INHERIT REMOTE PRIVILEGES FROM PUBLIC;
```

親トピック: データベース・リンクの定義者権限の制御

9.8.8 チュートリアル: 定義者権限プロシージャでのデータベース・リンクの使用

このチュートリアルでは、データベース・リンクを使用する定義者権限プロシージャでのINHERIT REMOTE PRIVILEGES権限の使用方法を示します。

- このチュートリアルについて
  このチュートリアルでは、INHERIT REMOTE PRIVILEGES権限の付与と取消しをテストします。
- ステップ1: ユーザー・アカウントの作成
  データベース・リンクを含む定義者権限プロシージャを作成するユーザーと、そのプロシージャを実行するユーザーを作成します。
- ステップ2: ユーザーIDを格納する表の作成(ユーザーdbuser2として)
  この表のユーザーIDは、データベース・リンクで使用するIDです。
- ステップ3: データベース・リンクおよび定義者権限プロシージャの作成(ユーザーdbuser1として)
  ユーザーdbuser1は、データベース・リンクに加えて、そのデータベース・リンクを参照する定義者権限プロシージャを作成できます。
- ステップ4: 定義者権限プロシージャのテスト

定義者権限プロシージャをテストする前に、ユーザーdbuser2がdbuser1にINHERIT REMOTE PRIVILEGESを付与する必要があります。

- ステップ5: このチュートリアルのコンポーネントの削除
  このチュートリアルのコンポーネントが不要になった場合、それらを削除できます。

親トピック: データベース・リンクの定義者権限の制御

9.8.8.1 このチュートリアルについて

このチュートリアルでは、INHERIT REMOTE PRIVILEGES権限の付与と取消しをテストします。

これを行うには、2人のユーザーを作成する必要があります。1人はデータベース・リンクを参照する定義者権限プロシージャを作成するユーザー、もう1人はこの定義者権限プロシージャを実行するユーザーです。いずれのユーザーも自分のスキーマで同一のルックアップ表を作成します。定義者権限プロシージャでは、定義者権限ユーザーに属するルックアップ表を2人目のユーザーが問い合せることを可能にする必要があります。

親トピック: チュートリアル: 定義者権限プロシージャでのデータベース・リンクの使用

9.8.8.2 ステップ1: ユーザー・アカウントの作成

データベース・リンクを含む定義者権限プロシージャを作成するユーザーと、そのプロシージャを実行するユーザーを作成します。

1. ユーザーを作成して権限付与を実行する権限を持つユーザーとして接続します。

   たとえば:

   ```
   sqlplus sec_admin
   Enter password: password
   ```

2. 次のようにユーザー・アカウントを作成します。

   ```
   GRANT CONNECT, RESOURCE, UNLIMITED TABLESPACE TO dbuser1 IDENTIFIED BY
   password;
   GRANT CONNECT, RESOURCE, UNLIMITED TABLESPACE TO dbuser2 IDENTIFIED BY
   password;
   ```

   「パスワードの最低要件」のガイドラインに従って、passwordを安全なパスワードに置き換えます。

親トピック: チュートリアル: 定義者権限プロシージャでのデータベース・リンクの使用

9.8.8.3 ステップ2: ユーザーIDを格納する表の作成(ユーザーdbuser2として)

この表のユーザーIDは、データベース・リンクで使用するIDです。

1. ユーザーdbuser2として、インスタンスinst1に接続します。

   ```
   connect dbuser2@inst1
   Enter password: password
   ```

2. 次のように表を作成します。

   ```
   CREATE TABLE dbusertab(ID NUMBER(2));
   ```

3. この表にID値10を移入します。

   ```
   INSERT INTO dbusertab VALUES(10);
   ```

親トピック: チュートリアル: 定義者権限プロシージャでのデータベース・リンクの使用

9.8.8.4 ステップ3: データベース・リンクおよび定義者権限プロシージャの作成(ユーザーdbuser1として)

ユーザーdbuser1は、データベース・リンクに加えて、そのデータベース・リンクを参照する定義者権限プロシージャを作成できます。

1. ユーザーdbuser1として、インスタンスinst1に接続します。

```
connect dbuser1@inst1
Enter password: password
```

2. 定義者権限プロシージャで使用するデータベース・リンクを作成します。

```
CREATE DATABASE LINK dblink USING 'inst1';
```

3. dbusertab表を作成し、その表にID 20を移入します。

```
CREATE TABLE DBUSERTAB(ID NUMBER(2));
INSERT INTO dbusertab VALUES(20);
```

4. データベース・リンクへの参照を含む定義者権限プロシージャを作成します

```
CREATE OR REPLACE PROCEDURE test_remote_db_link
AS
v_id varchar(50);
BEGIN
    SELECT ID INTO v_id FROM dbusertab@dblink;
    DBMS_OUTPUT.PUT_LINE('v_id : ' || v_id);
END ;
/
```

5. 定義者権限プロシージャをテストします。

```
SET SERVEROUTPUT ON
EXEC test_remote_db_link;
```

出力は次のようになり、ユーザーdbuser1が自分のバージョンの表dbusertabに対してプロシージャを実行したことが示されます。

```
v_id : 20
```

6. ユーザーdbuser2にtest_remote_db_linkプロシージャに対するEXECUTE権限を付与します。

```
GRANT EXECUTE ON test_remote_db_link TO dbuser2;
```

親トピック: チュートリアル: 定義者権限プロシージャでのデータベース・リンクの使用

9.8.8.5 ステップ4: 定義者権限プロシージャのテスト

定義者権限プロシージャをテストする前に、ユーザーdbuser2がdbuser1にINHERIT REMOTE PRIVILEGESを付与する必要があります。

1. ユーザーdbuser2として、インスタンスinst1に接続します。

```
connect dbuser2@inst1
Enter password: password
```

2. ユーザーdbuser2のINHERIT REMOTE PRIVILEGE権限をdbuser1に付与します。

```
GRANT INHERIT REMOTE PRIVILEGES ON user dbuser2 TO dbuser1;
```

3. 新規セッションを開始するまで付与は有効にならないため、再ログインします。

```
connect dbuser2@inst1
Enter password: password
```

4. test_remote_db_link定義者権限プロシージャを実行します。

```
SET SERVEROUTPUT ON
EXEC dbuser1.test_remote_db_link;
```

出力は次のようになり、ユーザーdbuser1がデータベース・リンクを使用してdbuser2のスキーマに接続し、dbuser2のスキーマのdbusertab表の値にアクセスできることが示されます。

```
v_id : 10
```

5. ユーザーdbuser2のINHERIT REMOTE PRIVILEGE権限をdbuser1から取り消します。

```
REVOKE INHERIT REMOTE PRIVILEGES ON USER dbuser2 FROM dbuser1;
```

6. test_remote_db_link定義者権限プロシージャの実行を再試行します。

```
EXEC dbuser1.test_remote_db_link;
```

ORA-25433: ユーザーDBUSER1には接続ユーザーDBUSER2のINHERIT REMOTE PRIVILEGESがありませんエラーが表示されます。

親トピック: チュートリアル: 定義者権限プロシージャでのデータベース・リンクの使用

9.8.8.6 ステップ5: このチュートリアルのコンポーネントの削除

このチュートリアルのコンポーネントが不要になった場合、それらを削除できます。

1. ユーザー・アカウントおよびデータベース・リンクを削除する権限を持つユーザーとして接続します

たとえば:

```
connect sec_admin
Enter password: password
```

2. ユーザー・アカウントを削除します。

```
DROP USER dbuser1 CASCADE;
DROP USER dbuser2 CASCADE;
```

3. dblinkデータベース・リンクを削除します。

```
DROP PUBLIC DATABASE LINK dblink;
```

親トピック: チュートリアル: 定義者権限プロシージャでのデータベース・リンクの使用

10 PL/SQLパッケージおよびタイプでのファイングレイン・アクセスの管理

Oracle Databaseにはファイングレイン・アクセスのためのパッケージとタイプが用意されており、これにより外部ネットワーク・サービスやウォレットへのアクセスを制御できます。

- PL/SQLパッケージおよびタイプでのファイングレイン・アクセスの管理について
  PL/SQLパッケージのセットと1つの型を使用して、外部ネットワーク・サービスおよびウォレットに対するユーザー・アクセスを構成できます。
- 外部ネットワーク・サービスに対するファイングレイン・アクセス・コントロールについて
  Oracle Application Securityアクセス制御リスト(ACL)で、外部ネットワーク・サービスに対するファイングレイン・アクセス・コントロールを実装できます。
- Oracleウォレットへのアクセス制御について
  リモートWebサーバーによって保護されているWebページにアクセスするときに、ユーザーはOracleウォレットに格納されているクライアント資格証明とパスワードを使用して認証を受けることができます。
- 外部ネットワーク・サービスを使用するパッケージに依存しているアップグレードされたアプリケーション
  アップグレードされたアプリケーションでORA-24247ネットワーク・アクセス・エラーが発生することがあります。
- 外部ネットワーク・サービスのアクセス制御の構成
  DBMS_NETWORK_ACLパッケージは、外部ネットワーク・サービスのアクセス制御を構成します。
- Oracleウォレットへのアクセス制御の構成
  Oracleウォレットに対するファイングレイン・アクセス・コントロールにより、パスワードや証明書を必要とするネットワーク・サービスへのアクセスをユーザーに提供できます。
- 外部ネットワーク・サービスのアクセス制御の構成の例
  1つのロールとネットワーク接続など、様々な状況に適したアクセス制御を構成できます。
- ネットワーク・ホスト・コンピュータのグループの指定
  ネットワーク・ホスト・コンピュータのグループを指定するには、ワイルドカードを使用できます。
- 複数のアクセス制御リスト割当てでのホスト・コンピュータの優先順位
  ドメインに割り当てられているアクセス制御リストの優先順位は、サブ・ドメインに割り当てられているアクセス制御リストの優先順位よりも低くなります。
- ポート範囲指定によるアクセス制御リスト割当てでのホストの優先順位
  アクセス制御リスト内のホストの優先順位は、ポート範囲の使用の影響を受けます。
- ネットワーク・ホストへのユーザー・アクセスに影響を与える権限割当てのチェック
  管理者とユーザーは、どちらもネットワーク接続およびドメイン権限をチェックできます。
- Javaデバッグ・ワイヤ・プロトコル操作のネットワーク・アクセスの構成
  Java PL/SQLプロシージャをデバッグするには、jdwp ACL権限を付与されていることが必要です。
- ユーザー・アクセス用に構成されたアクセス制御リストのデータ・ディクショナリ・ビュー
  Oracle Databaseでは、既存のアクセス制御リストに関する情報の検索に使用できるデータ・ディクショナリ・ビューが提供されています。

親トピック: ユーザー認証および認可の管理

10.1 PL/SQLパッケージおよびタイプでのファイングレイン・アクセスの管理について

PL/SQLパッケージのセットと1つの型を使用して、外部ネットワーク・サービスおよびウォレットに対するユーザー・アクセスを構成できます。

パッケージはUTL_TCP、UTL_SMTP、UTL_MAIL、UTL_HTTP、UTL_INADDR、DBMS_LDAPのPL/SQLパッケージ、型はHttpUriTypeです。

次の使用方法が可能です。

- データベースから外部ネットワーク・サービスへのアクセスが必要なユーザーやロールに、ファイングレイン・アクセス・コントロールを構成します。これによって、特定のユーザー・グループは、付与されている権限に基づいて1つ以上のホスト・コンピュータに接続できます。通常、この機能は、特定のホスト・アドレスで実行されるアプリケーションへのアクセスを制御するために使用します。
- パスワードまたはクライアント証明書認証を必要とするHTTPリクエストを処理するために、Oracleウォレットにファイングレイン・アクセス・コントロールを構成します。この機能により、Oracleウォレットに格納されているパスワードとクライアント証明書を使用するユーザーに、保護されている外部HTTPリソースにUTL_HTTPパッケージを介してアクセスする権限を付与できます。たとえば、アプリケーションで資格証明をハードコードするかわりに、ウォレットに格納されている資格証明を使用するようにアプリケーションを構成できます。

親トピック: PL/SQLパッケージおよびタイプでのファイングレイン・アクセスの管理

10.2 外部ネットワーク・サービスに対するファイングレイン・アクセス・コントロールについて

Oracle Application Securityアクセス制御リスト(ACL)で、外部ネットワーク・サービスに対するファイングレイン・アクセス・コントロールを実装できます。

このマニュアルでは、DBMS_NETWORK_ACL_ADMIN PL/SQLパッケージを使用してデータベース・ユーザーおよびロールのアクセス制御を構成する方法について説明します。

この機能を使用することで、データベース・ユーザーがPL/SQLネットワーク・ユーティリティ・パッケージUTL_TCP、UTL_SMTP、UTL_MAIL、UTL_HTTP、UTL_INADDR、PL/SQLパッケージDBMS_LDAPとDBMS_DEBUG_JDWPおよびHttpUriType型を使用して接続できる外部ネットワーク・ホストが制限されるため、ネットワーク接続のセキュリティが強化されます。この機能を使用しない場合、デフォルトでは、PL/SQLユーティリティ・パッケージは、PUBLICユーザーに付与されるEXECUTE権限付きで作成されるため、データベースへのアクセス権を獲得した侵入者が故意にネットワークを攻撃する可能性があります。これらのPL/SQLネットワーク・ユーティリティ・パッケージおよびDBMS_NETWORK_ACL_ADMINとDBMS_NETWORK_ACL_UTILITYパッケージは、IPバージョン4(IPv4)アドレスとIPバージョン6(IPv6)アドレスの両方をサポートしています。このマニュアルでは、両方のバージョンに対してアクセス制御を管理する方法について説明します。Oracle DatabaseでのIPv4およびIPv6表記法の詳細は、『Oracle Database Net Services管理者ガイド』を参照してください。

関連項目:

電子メール・アラート用に外部ネットワーク・サービスへのアクセス制御を構成する例は、例: ファイングレイン監査ポリシーへの電子メール・アラートの追加を参照してください

親トピック: PL/SQLパッケージおよびタイプでのファイングレイン・アクセスの管理

10.3 Oracleウォレットへのアクセス制御について

リモートWebサーバーによって保護されているWebページにアクセスするときに、ユーザーはOracleウォレットに格納されているクライアント資格証明とパスワードを使用して認証を受けることができます。

Oracleウォレットでは、ユーザーのパスワードとクライアント証明書を安全に格納できます。

ウォレットに対するアクセス制御を構成するには、次のコンポーネントが必要です。

- Oracleウォレット。Oracleウォレットは、Oracle Databaseの`mkstore`ユーティリティまたはOracle Wallet Managerを使用して作成できます。HTTPリクエストでは、外部パスワード・ストアまたはウォレット内のクライアント証明書を使用してユーザーが認証されます。
- ユーザーにウォレットの使用権限を付与するアクセス制御リスト。アクセス制御リストを構成するには、`DBMS_NETWORK_ACL_ADMIN` PL/SQLパッケージを使用します。

Oracleウォレットを使用すると、保護されているWebページへのアクセスに必要なパスワードとクライアント証明書を安全に格納できます。

関連トピック

- Oracleウォレットへのアクセス制御の構成

親トピック: PL/SQLパッケージおよびタイプでのファイングレイン・アクセスの管理

10.4 外部ネットワーク・サービスを使用するパッケージに依存しているアップグレードされたアプリケーション

アップグレードされたアプリケーションで`ORA-24247`ネットワーク・アクセス・エラーが発生することがあります。

Oracle Database 11gリリース1 (11.1)より前のリリースからアップグレードしているときに、アプリケーションがPL/SQLネットワーク・ユーティリティ・パッケージ(`UTL_TCP`、`UTL_SMTP`、`UTL_MAIL`、`UTL_HTTP`、`UTL_INADDR`および`DBMS_LDAP`)または`HttpUriType`型に依存している場合は、そのアプリケーションの実行を試みたときに、`ORA-24247`エラーが発生する可能性があります。

エラー・メッセージは次のとおりです。

```
ORA-24247: network access denied by access control list (ACL)
```

その場合は、この章に記載されている手順を使用してアプリケーションのネットワーク・アクセスを再構成してください。

関連項目:

PL/SQLネットワーク・ユーティリティ・パッケージに依存しているアプリケーションの互換性に関する問題については、『Oracle Databaseアップグレード・ガイド』を参照してください。

親トピック: PL/SQLパッケージおよびタイプでのファイングレイン・アクセスの管理

10.5 外部ネットワーク・サービスのアクセス制御の構成

DBMS_NETWORK_ACLパッケージは、外部ネットワーク・サービスのアクセス制御を構成します。

- 外部ネットワーク・サービスのアクセス制御の構成の構文
  DBMS_NETWORK_ACL_ADMIN.APPEND_HOST_ACEプロシージャを使用して、アクセス制御権限をユーザーに付与できます。
- リスナーによる外部ネットワーク・サービスのアクセス制御の認識の有効化
  外部ネットワーク・サービスのアクセス制御を認識するようにリスナーが構成されていない場合、TNS-01166: Listener rejected registration or update of service ACLエラーが発生する可能性があります。
- 例: 外部ネットワーク・サービスのアクセス制御の構成
  DBMS_NETWORK_ACL_ADMIN.APPEND_HOST_ACEプロシージャで、外部ネットワーク・サービスのアクセス制御を構成できます。
- 外部ネットワーク・サービスのアクセス制御権限の取消し
  外部ネットワーク・サービスのアクセス制御権限を削除できます。
- 例: 外部ネットワーク・サービス権限の取消し
  DBMS_NETWORK_ACL_ADMIN.REMOVE_HOST_ACEプロシージャを使用して、外部ネットワーク・サービス権限を取り消すことができます。

親トピック: PL/SQLパッケージおよびタイプでのファイングレイン・アクセスの管理

10.5.1 外部ネットワーク・サービスのアクセス制御の構成の構文

DBMS_NETWORK_ACL_ADMIN.APPEND_HOST_ACEプロシージャを使用して、アクセス制御権限をユーザーに付与できます。

このプロシージャは、指定された権限を持つアクセス制御エントリ(ACE)を指定されたホストのACLに追加し、存在しない場合はACLを作成します。結果の構成は、作成したユーザーのスキーマでなくSYSスキーマにあります。

構文は次のとおりです。

```
BEGIN
 DBMS_NETWORK_ACL_ADMIN.APPEND_HOST_ACE (
  host         => 'host_name',
  lower_port   => null|port_number,
  upper_port   => null|port_number,
  ace          => ace_definition);
END;
```

詳細は、次のとおりです。

- host: ホストの名前を入力します。ホスト名またはホストのIPアドレスのいずれかです。ワイルドカードを使用して、ドメインまたはIPサブネットを指定できます。(ドメイン名でワイルドカードを使用する場合の優先順位は、複数のアクセス制御リスト割当てでのホスト・コンピュータの優先順位を参照してください。)ホストまたはドメイン名は大/小文字が区別されません。次に例を示します。

  ```
  host     => 'www.example.com',
  host     => '*example.com',
  ```

- lower_port:(オプション)TCP接続に対するポート範囲の下限を入力します。connect権限にのみ、この設定を使用します。resolve権限には省略します。デフォルトはnullで、ポート制限がない(つまり、すべてのポートにACL

が適用される)ことを意味します。ポート番号の範囲は1から65535です。

たとえば:

```
lower_port => 80,
```

- upper_port:(オプション)TCP接続に対するポート範囲の上限を入力します。connect権限にのみ、この設定を使用します。resolve権限には省略します。デフォルトはnullで、ポート制限がない(つまり、すべてのポートにACLが適用される)ことを意味します。ポート番号の範囲は1から65535です。

  たとえば:

  ```
  upper_port => 3999);
  ```

  lower_portに値を入力してupper_portをnullのままに(または省略)すると、upper_portの設定は、lower_portと同じ設定とみなされます。たとえば、lower_portを80に設定し、upper_portを省略すると、このupper_portの設定は80とみなされます。

  アクセス制御リスト割当てにポート範囲が指定されている場合、アクセス制御リストのresolve権限は機能しません。

- ace: 次の形式のXS$ACE_TYPE定数を使用して、ACEを定義します。

  ```
  ace    => xs$ace_type(privilege_list => xs$name_list('privilege'),
                        principal_name => 'user_or_role',
                        principal_type => xs$ace_type_user));
  ```

  詳細は、次のとおりです。

  - privilege_list: 大/小文字を区別しない1つ以上の次の権限を入力します。一重引用符で各権限を囲み、カンマ('http', 'http_proxy'など)で区切ります。

    アクセス制御を強化するために、ユーザーがUTL_HTTP、HttpUriType、UTL_SMTPまたはUTL_MAILのみを使用する場合、connect権限のかわりにhttp、http_proxyまたはsmtp権限のみ付与します。

    - http: UTL_HTTPパッケージおよびHttpUriType型を通じてホストへのHTTPリクエストを作成します

    - http_proxy: UTL_HTTPパッケージおよびHttpUriType型を通じてプロキシを介したHTTPリクエストを作成します。ユーザーがプロキシを介してHTTPリクエストを作成する場合、http権限と組み合せてhttp_proxyを含む必要があります。

    - smtp: UTL_SMTPおよびUTL_MAILパッケージを通じて、SMTPをホストに送信します

    - resolve: UTL_INADDRパッケージを通じて、ネットワーク・ホスト名またはIPアドレスを解決します

    - connect: UTL_TCP、UTL_SMTP、UTL_MAIL、UTL_HTTPおよびDBMS_LDAPパッケージまたはHttpUriType型を通じて、ホストのネットワーク・サービスに接続するユーザー権限を付与します

    - jdwp: JavaまたはPL/SQLストアド・プロシージャのJavaデバッグ・ワイヤ・プロトコル・デバッグ操作に使用されます。詳細は、Javaデバッグ・ワイヤ・プロトコル操作のネットワーク・アクセスの構成を参照してください。

  - principal_name: データベース・ユーザー名またはロールを入力します。二重引用符(たとえば、'"ACCT_MGR"'など)で入力しないかぎり、この値は大/小文字を区別しません。

  - principal_type: データベース・ユーザーまたはロールのXS_ACL.PTYPE_DBを入力します。principal_type値のデフォルトがOracle Database Real Application Securityアプリケーション・ユーザーの指定に使用されるPTYPE_XSであるため、PTYPE_DBを指定する必要があります。

関連項目:

aceパラメータ設定に含むことができる追加のXS$ACE_TYPEパラメータgranted、inverted、start_dateおよびend_dateの詳細は、『Oracle Database Real Application Security管理者および開発者ガイド』を参照してください。

親トピック: 外部ネットワーク・サービスのアクセス制御の構成

10.5.2 リスナーによる外部ネットワーク・サービスのアクセス制御の認識の有効化

外部ネットワーク・サービスのアクセス制御を認識するようにリスナーが構成されていない場合、TNS-01166: Listener rejected registration or update of service ACLエラーが発生する可能性があります。

1. listener.oraファイルに次の行を追加します。

```
LOCAL_REGISTRATION_ADDRESS_LISTENER = ON
```

2. リスナーを再起動します。

```
./lsnrctl stop
./lsnrctl start
```

親トピック: 外部ネットワーク・サービスのアクセス制御の構成

10.5.3 例: 外部ネットワーク・サービスのアクセス制御の構成

DBMS_NETWORK_ACL_ADMIN.APPEND_HOST_ACEプロシージャで、外部ネットワーク・サービスのアクセス制御を構成できます。

例10-1は、ホストwww.example.comに作成されるACLのacct_mgrデータベース・ロールへのhttpおよびsmtp権限の付与方法を示しています。

例10-1 データベース・ロールの外部ネットワーク・サービスへの権限の付与

```
BEGIN
 DBMS_NETWORK_ACL_ADMIN.APPEND_HOST_ACE(
  host       => 'www.example.com',
  ace        =>  xs$ace_type(privilege_list => xs$name_list('http', 'smtp'),
                             principal_name => 'acct_mgr',
                             principal_type => xs_acl.ptype_db));
END;
/
```

親トピック: 外部ネットワーク・サービスのアクセス制御の構成

10.5.4 外部ネットワーク・サービスのアクセス制御権限の取消し

外部ネットワーク・サービスのアクセス制御権限を削除できます。

- 外部ネットワーク・サービスのアクセス制御権限を取り消すには、DBMS_NETWORK_ACL_ADMIN.REMOVE_HOST_ACEプロシージャを実行します。

関連項目:

DBMS_NETWORK_ACL_ADMIN.REMOVE_HOST_ACEプロシージャの詳細は、『Oracle Database PL/SQLパッケージ

およびタイプ・リファレンス』を参照してください。

親トピック: 外部ネットワーク・サービスのアクセス制御の構成

10.5.5 例: 外部ネットワーク・サービス権限の取消し

DBMS_NETWORK_ACL_ADMIN.REMOVE_HOST_ACEプロシージャを使用して、外部ネットワーク・サービス権限を取り消すことができます。

例10-2は、外部ネットワーク権限の取消し方法を示しています。

例10-2 外部ネットワーク・サービス権限の取消し

```
BEGIN
 DBMS_NETWORK_ACL_ADMIN.REMOVE_HOST_ACE (
  host       => 'www.example.com',
  lower_port => 80,
  upper_port => upper_port => 3999,
  ace        => xs$ace_type(privilege_list => xs$name_list('http', 'smtp'),
                            principal_name => 'acct_mgr',
                            principal_type => xs_acl.ptype_db),
  remove_empty_acl => TRUE);
END;
/
```

ここで、remove_empty_aclのTRUE設定は、ACEの削除時に空になる場合にACLを削除します。

親トピック: 外部ネットワーク・サービスのアクセス制御の構成

10.6 Oracleウォレットへのアクセス制御の構成

Oracleウォレットに対するファイングレイン・アクセス・コントロールにより、パスワードや証明書を必要とするネットワーク・サービスへのアクセスをユーザーに提供できます。

- Oracleウォレットへのアクセス制御の構成について
  アクセス制御を構成して、パスワードおよびクライアント証明書にアクセス権を付与できます。
- ステップ1: Oracleウォレットの作成
  Oracleウォレットでは標準およびPKCS11の両タイプのウォレットを使用できるほか、自動ログイン・ウォレットにすることもできます。
- ステップ2: Oracleウォレットのアクセス制御権限の構成
  ウォレットを作成した後、ウォレットのアクセス制御権限を構成できます。
- ステップ3: パスワードとクライアント証明書を使用するHTTPリクエストの作成
  UTL_HTTPパッケージでウォレット情報を格納したHTTPリクエスト・オブジェクトを作成し、クライアント証明書またはパスワードを使用して認証できます。
- Oracleウォレットのアクセス制御権限の取消し
  Oracleウォレットのアクセス制御権限を取り消すことができます。
- ORA-29024エラーのトラブルシューティング
  ORA-29024「証明書の検証に失敗しました」というエラーは、機能、コンポーネントまたは製品、あるいは失敗した操作でOracleウォレットが必要となっている場合に発生します。

親トピック: PL/SQLパッケージおよびタイプでのファイングレイン・アクセスの管理

10.6.1 Oracleウォレットへのアクセス制御の構成について

アクセス制御を構成して、パスワードおよびクライアント証明書にアクセス権を付与できます。

これらのパスワードとクライアント証明書はOracleウォレットに格納されます。アクセス制御を構成すると、PL/SQLネットワーク・ユーティリティ・パッケージを使用するときに外部ネットワーク・サービスに対してユーザーが自身を証明できるようになります。

これにより、ユーザーは、パスワードまたは証明書IDが必要なネットワーク・サービスへのアクセスを取得できます。

親トピック: Oracleウォレットへのアクセス制御の構成

10.6.2 ステップ1: Oracleウォレットの作成

Oracleウォレットでは標準およびPKCS11の両タイプのウォレットを使用できるほか、自動ログイン・ウォレットにすることもできます。

1. ウォレットを作成するには、`mkstore`コマンドライン・ユーティリティまたはOracle Wallet Managerユーザー・インタフェースを使用します。

   ウォレットにパスワードを格納するには、`mkstore`ユーティリティを使用する必要があります。

2. ウォレットがファイルにエクスポートされていることを確認します。
3. ウォレットを作成したディレクトリをノートにとっておきます。このディレクトリ・パスは、このセクションの手順の終了後に必要になります。

関連トピック

- 例: 非共有ウォレットのパスワードを使用するACLアクセスの構成
- 例: 共有データベース・セッションに使用するウォレットのACLアクセスの構成

親トピック: Oracleウォレットへのアクセス制御の構成

10.6.3 ステップ2: Oracleウォレットのアクセス制御権限の構成

ウォレットを作成した後、ウォレットのアクセス制御権限を構成できます。

- `DBMS_NETWORK_ACL_ADMIN.APPEND_WALLET_ACE`プロシージャを使用して、ウォレット・アクセス制御権限を構成します。

`DBMS_NETWORK_ACL_ADMIN.APPEND_WALLET_ACE`プロシージャの構文は次のとおりです。

```
BEGIN
 DBMS_NETWORK_ACL_ADMIN.APPEND_WALLET_ACE (
  wallet_path => 'directory_path_to_wallet',
  ace         => xs$ace_type(privilege_list => xs$name_list('privilege'),
                             principal_name => 'user_or_role',
                             principal_type => xs$ace_type_user));
END;
```

詳細は、次のとおりです。

- `wallet_path`: ステップ1: Oracleウォレットの作成で作成したウォレットを含むディレクトリへのパスを入力します。ウォレットのパスを指定する際は、絶対パスを使用し、ディレクトリ・パスの前に`file:`を入れる必要があります。`$ORACLE_HOME`などの環境変数を使用したり、`file:`の後およびパス名の前にスペースを挿入することはできません。たとえば:

  ```
  wallet_path    => 'file:/oracle/wallets/hr_wallet',
  ```

- ace: XS$ACE_TYPE定数を使用して、ACLを定義します。たとえば:

```
ace         =>  xs$ace_type(privilege_list => xs$name_list(privilege),
                            principal_name => 'hr_clerk',
                            principal_type => xs_acl.ptype_db);
```

ここでは、xs$ace_typeを使用してウォレット権限を入力する場合、privilegeが次のいずれかである必要があります(これらの権限名のアンダースコアの使用に注意してください)。

- use_client_certificates
- use_passwords

これらのパラメータの詳細は、外部ネットワーク・サービスのアクセス制御の構成の構文のaceパラメータの説明を参照してください。ウォレットには、use_client_certificatesまたはuse_passwords権限を指定する必要があります。

関連項目:

aceパラメータ設定に含むことができる追加のXS$ACE_TYPEパラメータgranted、inverted、start_dateおよびend_dateの詳細は、『Oracle Database Real Application Security管理者および開発者ガイド』を参照してください。

親トピック: Oracleウォレットへのアクセス制御の構成

10.6.4 ステップ3: パスワードとクライアント証明書を使用するHTTPリクエストの作成

UTL_HTTPパッケージでウォレット情報を格納したHTTPリクエスト・オブジェクトを作成し、クライアント証明書またはパスワードを使用して認証できます。

- パスワードとクライアント証明書を使用するHTTPSリクエストの作成
  UTL_HTTPパッケージは、SQLとPL/SQLからHypertext Transfer Protocol (HTTP)のコールアウトを行います。
- 他のアプリケーションとセッションを共有している場合に、リクエスト・コンテキストを使用してウォレットを保留
  他のアプリケーションとデータベース・セッションを共有している場合は、リクエスト・コンテキストを使用してウォレットを保留にする必要があります。
- 認証にクライアント証明書のみを使用
  ユーザーに対してACLウォレットで適切な権限が付与されているかぎり、ユーザーの認証にクライアント証明書のみを使用できます。
- 認証にパスワードを使用
  要求したURLが保護されていてユーザー名とパスワードの認証を要求される場合は、ウォレットからユーザー名とパスワードを設定して認証します。

親トピック: Oracleウォレットへのアクセス制御の構成

10.6.4.1 パスワードとクライアント証明書を使用するHTTPリクエストの作成

UTL_HTTPパッケージは、SQLとPL/SQLからHypertext Transfer Protocol (HTTP)のコールアウトを行います。

- UTL_HTTP PL/SQLパッケージを使用して、HTTPリクエストとそのレスポンスで個別に使用されるリクエスト・コンテキスト・オブジェクトを作成します。

たとえば:

```
DECLARE
 req_context UTL_HTTP.REQUEST_CONTEXT_KEY;
```

```
 req         UTL_HTTP.REQ;
BEGIN
 req_context := UTL_HTTP.CREATE_REQUEST_CONTEXT (
              wallet_path          => 'file:path_to_directory_containing_wallet',
              wallet_password      => 'wallet_password'|NULL);
 req := UTL_HTTP.BEGIN_REQUEST(
              url                  => 'URL_to_application',
              request_context      => 'request_context'|NULL);
 ...
END;
```

詳細は、次のとおりです。

- `req_context`: `UTL_HTTP.CREATE_REQUEST_CONTEXT_KEY`データ型を使用して、リクエスト・コンテキスト・オブジェクトを作成します。このオブジェクトには、Oracle Databaseがリクエスト・コンテキストの識別に使用する、ランダムに生成された数値キーが格納されています。`UTL_HTTP.CREATE_REQUEST_CONTEXT`ファンクションは、リクエスト・コンテキスト自体を作成します。
- `req`: `UTL_HTTP.REQ`データ型を使用して、HTTPリクエストを開始するためのオブジェクトを作成します。このオブジェクトは後で、パスワードで保護されたWebページにアクセスするために、ウォレットからユーザー名とパスワードを設定する際に参照します。
- `wallet_path`: ウォレットが格納されているディレクトリのパスを入力します。このパスは、前のセクションのステップ2: Oracleウォレットのアクセス制御権限の構成でアクセス制御リストを作成したときに指定したパスと同じであることを確認してください。ディレクトリ・パスの前に`file:`を含める必要があります。`$ORACLE_HOME`などの環境変数を使用しないでください。

  たとえば:

  ```
  wallet_path          => 'file:/oracle/wallets/hr_wallet',
  ```

- `wallet_password`: ウォレットを開くためのパスワードを入力します。デフォルトは`NULL`で、自動ログイン・ウォレットに使用します。たとえば:

  ```
  wallet_password      => 'wallet_password');
  ```

- `url`: ウォレットを使用するアプリケーションのURLを入力します。

  たとえば:

  ```
  url                  => 'www.hr_access.example.com',
  ```

- `request_context`: このセクションの前の方で作成したリクエスト・コンテキスト・オブジェクトの名前を入力します。このオブジェクトにより、ウォレットは同一データベース・セッション内の他のアプリケーションとは共有されなくなります。

  たとえば:

  ```
  request_context      => req_context);
  ```

関連項目:

`UTL_HTTP`パッケージの詳細は、『Oracle Database PL/SQLパッケージおよびタイプ・リファレンス』を参照してください。

親トピック: ステップ3: パスワードとクライアント証明書を使用するHTTPリクエストの作成

10.6.4.2 他のアプリケーションとセッションを共有している場合に、リクエスト・コンテキストを使用してウォレットを保留

他のアプリケーションとデータベース・セッションを共有している場合は、リクエスト・コンテキストを使用してウォレットを保留にする必要があります。

アプリケーションがデータベース・セッションを排他的に使用している場合は、UTL_HTTP.SET_WALLETプロシージャを使用してデータベース・セッションでウォレットを保留できます。

- UTL_HTTP.SET_WALLETプロシージャを使用して、ウォレットを保有するようにリクエストを構成します。

たとえば:

```
DECLARE
 req         UTL_HTTP.REQ;
BEGIN
 UTL_HTTP.SET_WALLET(
              path          => 'file:path_to_directory_containing_wallet',
              password      => 'wallet_password'|NULL);
 req := UTL_HTTP.BEGIN_REQUEST(
              url           => 'URL_to_application');
 ...
END;
```

要求された保護されているURLが、認証にユーザー名とパスワードを必要とする場合は、SET_AUTHENTICATION_FROM_WALLETプロシージャを使用してウォレットからユーザー名とパスワードを設定して認証します。

親トピック: ステップ3: パスワードとクライアント証明書を使用するHTTPリクエストの作成

10.6.4.3 認証にクライアント証明書のみを使用

ユーザーに対してACLウォレットで適切な権限が付与されているかぎり、ユーザーの認証にクライアント証明書のみを使用できます。

要求された保護されているURLが、認証にクライアント証明書のみを必要とする場合、BEGIN_REQUESTファンクションはウォレットに割り当てられているACLでユーザーにuse_client_certificates権限が付与されているものとみなして、ウォレットから必要なクライアント証明書を送信します。

認証はリモートWebサーバーで成功し、ユーザーはGET_RESPONSEファンクションを使用してHTTP応答を取得できます。

親トピック: ステップ3: パスワードとクライアント証明書を使用するHTTPリクエストの作成

10.6.4.4 認証にパスワードを使用

要求したURLが保護されていてユーザー名とパスワードの認証を要求される場合は、ウォレットからユーザー名とパスワードを設定して認証します。

たとえば:

```
DECLARE
 req_context UTL_HTTP.REQUEST_CONTEXT_KEY;
 req         UTL_HTTP.REQ;
BEGIN
...
 UTL_HTTP.SET_AUTHENTICATION_FROM_WALLET(
  r               => HTTP_REQUEST,
  alias           => 'alias_to_retrieve_credentials_stored_in_wallet',
  scheme          => 'AWS|Basic',
  for_proxy       => TRUE|FALSE);
```

```
END;
```

詳細は、次のとおりです。

- r: 前のセクションで作成したUTL_HTTP.BEGIN_REQUESTプロシージャで定義したHTTPリクエストを入力します。たとえば:

  ```
  r               => req,
  ```

- alias: Oracleウォレットに格納されているユーザー名とパスワード資格証明を識別および取得するための別名を入力します。たとえば、このユーザー名とパスワード資格証明を識別するための別名がhr_accessであるとします。

  ```
  alias           => 'hr_access',
  ```

- scheme: 次のいずれかを入力します。

  - AWS: Amazon Simple Storage Service(S3)スキームを指定します。このスキームは、Amazon.comのWebサイトへのアクセスを構成する場合にのみ使用します。(この設定の詳細は、Amazonにお問い合せください。)
  - Basic: HTTP basic認証を指定します。デフォルトはBasicです。

  たとえば:

  ```
  scheme          => 'Basic',
  ```

- for_proxy: HTTP認証情報がWebサーバーではなくHTTPプロキシ・サーバーへのアクセス用かどうかを指定します。デフォルトはFALSEです。

  たとえば:

  ```
  for_proxy       => TRUE);
  ```

ウォレットのユーザー名とパスワードを使用するには、ウォレットに割り当てられているACLでユーザーにuse_passwords権限が付与されている必要があります。

親トピック: ステップ3: パスワードとクライアント証明書を使用するHTTPリクエストの作成

10.6.5 Oracleウォレットのアクセス制御権限の取消し

Oracleウォレットのアクセス制御権限を取り消すことができます。

- ウォレットのアクセス制御リスト(ACL)のアクセス制御エントリ(ACE)から権限を取り消すには、DBMS_NETWORK_ACL_ADMIN.REMOVE_WALLET_ACEプロシージャを実行します。

たとえば:

```
BEGIN
 DBMS_NETWORK_ACL_ADMIN.REMOVE_WALLET_ACE (
  wallet_path   => 'file:/oracle/wallets/hr_wallet',
  ace           =>  xs$ace_type(privilege_list => xs$name_list(privilege),
                                principal_name => 'hr_clerk',
                                principal_type => xs_acl.ptype_db),
  remove_empty_acl  => TRUE);
END;
/
```

この例では、remove_empty_aclのTRUE設定により、ウォレットACEの削除時に空になる場合にACLが削除されます。

親トピック: Oracleウォレットへのアクセス制御の構成

10.6.6 ORA-29024エラーのトラブルシューティング

ORA-29024「証明書の検証に失敗しました」というエラーは、機能、コンポーネントまたは製品、あるいは失敗した操作でOracleウォレットが必要となっている場合に発生します。

次の方法をこの順序で使用すると、このエラーをトラブルシューティングできます。

1. 関連するOracleドキュメントで、失敗している構成に関連する手順を確認してください。

   たとえば、UTL_HTTPの使用中にこのエラーが発生した場合は、セキュアなWebサイトにウォレットなしでアクセスしており、この操作には作成したウォレットが必要であるということです。UTL_HTTP PL/SQLパッケージについては、『Oracle Database PL/SQLパッケージ・プロシージャおよびタイプ・リファレンス』を参照してください。

   別の例としては、TLS接続を介したデータベース・サーバーへのリモート接続の実行中にこのエラーが発生する可能性があります。これは、この接続でOracleウォレットが必要となっていることを示しています。この問題をトラブルシューティングするには、Oracleウォレットと証明書について正確に理解している必要があります。「Transport Layer Security認証の構成」を参照してください。

2. ドキュメントに従ってウォレットを構成した後もこのエラーが発生する場合は、次の解決策を試してください。
   - 次のように、orapkiユーティリティを使用してウォレットを開きます。

     ```
     orapki wallet display -wallet wallet_file_directory
     ```

     このコマンドに失敗した場合は、そのウォレットが破損しているということです。新しいウォレットを作成し、シナリオを再チェックします。
   - 現在の構成にユーザー証明書および信頼できる証明書を含むウォレットが必要な場合は、ユーザー証明書および信頼できる証明書の両方が有効であり期限切れで失効していないかどうかを確認します。
   - UTL_HTTP構成でウォレットの使用中にこのエラーが発生した場合は、セキュアなWebサイトのすべての証明書がそのウォレットにあり証明連鎖が完成しているかどうかを確認します。
   - プロキシ・サーバーが含まれている場合は、ターゲットWebサイトがプロキシallowlistにあることを確認します。

UTL_HTTPSコールのためにセキュアなサイトの完成した証明連鎖を取得する方法については、次のMy Oracle Supportノートを参照してください。

- ノート169768.1 Configuring Wallet Manager to enable HTTPS connections via UTL_HTTP.REQUEST
- ノート230917.1 Troubleshooting the UTL_HTTP Package

親トピック: Oracleウォレットへのアクセス制御の構成

10.7 外部ネットワーク・サービスのアクセス制御の構成の例

1つのロールとネットワーク接続など、様々な状況に適したアクセス制御を構成できます。

- 例: 1つのロールおよびネットワーク接続のアクセス制御の構成
  DBMS_NETWORK_ACL_ADMIN.APPEND_HOST_ACEプロシージャで、1つのロールおよびネットワーク接続のアクセス制御を構成できます。
- 例: ユーザーとロールに対するアクセス制御の構成

DBMS_NETWORK_ACL_ADMIN.APPEND_HOST_ACEで、1つのユーザーおよびロールの権限を拒否または付与するアクセス制御を構成できます。

- 例: DBA_HOST_ACESビューを使用した付与権限の表示
  DBA_HOST_ACEデータ・ディクショナリ・ビューにはユーザーに付与されている権限が表示されます。
- 例: 非共有ウォレットのパスワードを使用するACLアクセスの構成
  DBMS_NETWORK_ACL_ADMINおよびUTL_HTTP PL/SQLパッケージで、非共有ウォレットのパスワードを使用してACLアクセスを構成できます。
- 例: 共有データベース・セッションに使用するウォレットのACLアクセスの構成
  DBMS_NETWORK_ACL_ADMINおよびUTL_HTTP PL/SQLパッケージで、共有データベース・セッションに使用するウォレットのACLアクセスを構成できます。

親トピック: PL/SQLパッケージおよびタイプでのファイングレイン・アクセスの管理

10.7.1 例: 1つのロールおよびネットワーク接続のアクセス制御の構成

DBMS_NETWORK_ACL_ADMIN.APPEND_HOST_ACEプロシージャで、1つのロールおよびネットワーク接続のアクセス制御を構成できます。

例10-3は、1つのロール(acct_mgr)のアクセス制御を構成し、このロールにwww.us.example.comホストにアクセスするhttp権限を付与する方法を示しています。権限は、2013年1月1日に期限切れになります。

例10-3 1つのロールおよびネットワーク接続のアクセス制御の構成

```
BEGIN
 DBMS_NETWORK_ACL_ADMIN.APPEND_HOST_ACE(
  host       => 'www.us.example.com',
  lower_port => 80,
  ace        =>  xs$ace_type(privilege_list => xs$name_list('http'),
                             principal_name => 'acct_mgr',
                             principal_type => xs_acl.ptype_db,
                             end_date => TIMESTAMP '2013-01-01 00:00:00.00 -08:00');
END;
/
```

親トピック: 外部ネットワーク・サービスのアクセス制御の構成の例

10.7.2 例: ユーザーとロールに対するアクセス制御の構成

DBMS_NETWORK_ACL_ADMIN.APPEND_HOST_ACEで、1つのユーザーおよびロールの権限を拒否または付与するアクセス制御を構成できます。

その後、DBA_HOST_ACESデータ・ビュー・ディクショナリを問い合せて、権限付与に関する情報を確認できます。

例10-4では、データベース・ロール(acct_mgr)に付与しますが、ロールを持つ場合でも特定のユーザー(psmith)を拒否します。ACEが指定された順序で評価されるため、順序は重要です。この場合、拒否ACE(granted => false)を最初に追加する必要があり、追加しないとユーザーを拒否できません。

例10-4 ユーザーおよびロールに対する付与および拒否を使用したアクセス制御リストの構成

```
BEGIN
 DBMS_NETWORK_ACL_ADMIN.APPEND_HOST_ACE(
  host       => 'www.us.example.com',
  lower_port => 80,
  upper_port => 80,
  ace        =>  xs$ace_type(privilege_list => xs$name_list('http'),
                             principal_name => 'psmith',
```

```
                              principal_type => xs_acl.ptype_db,
                              granted        => false));
 DBMS_NETWORK_ACL_ADMIN.APPEND_HOST_ACE(
  host       => 'www.us.example.com',
  lower_port => 80,
  upper_port => 80,
  ace        =>  xs$ace_type(privilege_list => xs$name_list('http'),
                              principal_name => 'acct_mgr',
                              principal_type => xs_acl.ptype_db,
                              granted        => true));
END;
```

親トピック: 外部ネットワーク・サービスのアクセス制御の構成の例

10.7.3 例: DBA_HOST_ACESビューを使用した付与権限の表示

DBA_HOST_ACEデータ・ディクショナリ・ビューにはユーザーに付与されている権限が表示されます。

例10-5は、前述のアクセス制御リストで付与された権限をDBA_HOST_ACESデータ・ディクショナリ・ビューに表示する方法を示しています。

例10-5 DBA_HOST_ACESビューを使用した付与権限の表示

```
SELECT PRINCIPAL, PRIVILEGE, GRANT_TYPE FROM DBA_HOST_ACE WHERE PRIVILEGE = 'HTTP';
PRINCIPAL    PRIVILEGE  GRANT_TYPE
------------ ---------- --------------------
PSMITH       HTTP       FALSE
ACCT_MGR     HTTP       TRUE
```

親トピック: 外部ネットワーク・サービスのアクセス制御の構成の例

10.7.4 例: 非共有ウォレットのパスワードを使用するACLアクセスの構成

DBMS_NETWORK_ACL_ADMINおよびUTL_HTTP PL/SQLパッケージで、非共有ウォレットのパスワードを使用してACLアクセスを構成できます。

例10-6では、人事部門の2つのロール、hr_clerkとhr_managerによるウォレットへのアクセスを構成しています。これらのロールは、use_passwords権限を使用して、ウォレットに格納されているパスワードにアクセスします。この例では、ウォレットは同一データベース・セッション内の他のアプリケーションとは共有されません。

例10-6 非共有ウォレットのパスワードを使用するACLアクセスの構成

```
/* 1. At a command prompt, create the wallet. The following example uses the
      user name hr_access as the alias to identify the user name and password
      stored in the wallet. You must use this alias name when you call the
      SET_AUTHENTICATION_FROM_WALLET procedure later on. */
$ mkstore -wrl $ORACLE_HOME/wallets/hr_wallet -create
Enter password: password
Enter password again: password
$ mkstore -wrl $ORACLE_HOME/wallets/hr_wallet -createCredential hr_access hr_usr
Your secret/Password is missing in the command line
Enter your secret/Password: password
Re-enter your secret/Password: password
Enter wallet password: password
/* 2. In SQL*Plus, create an access control list to grant privileges for the
      wallet. The following example grants the use_passwords privilege to the
      hr_clerk role.*/
BEGIN
 DBMS_NETWORK_ACL_ADMIN.APPEND_WALLET_ACE (
  wallet_path => 'file:/oracle/wallets/hr_wallet',
  ace         => xs$ace_type(privilege_list => xs$name_list('use_passwords'),
```

```
                              principal_name => 'hr_clerk',
                              principal_type => xs_acl.ptype_db));
END;
/
/* 3. Create a request context and request object, and then set the authentication
      for the wallet. */
DECLARE
  req_context  UTL_HTTP.REQUEST_CONTEXT_KEY;
  req          UTL_HTTP.REQ;
BEGIN
 req_context := UTL_HTTP.CREATE_REQUEST_CONTEXT(
     wallet_path          => 'file:/oracle/wallets/hr_wallet',
     wallet_password      => NULL,
     enable_cookies       => TRUE,
     max_cookies          => 300,
     max_cookies_per_site => 20);
  req := UTL_HTTP.BEGIN_REQUEST(
     url                  => 'www.hr_access.example.com',
     request_context      => req_context);
  UTL_HTTP.SET_AUTHENTICATION_FROM_WALLET(
     r                    => req,
     alias                => 'hr_access'),
     scheme               => 'Basic',
     for_proxy            => FALSE);
END;
/
```

親トピック: 外部ネットワーク・サービスのアクセス制御の構成の例

10.7.5 例: 共有データベース・セッションに使用するウォレットのACLアクセスの構成

DBMS_NETWORK_ACL_ADMINおよびUTL_HTTP PL/SQLパッケージで、共有データベース・セッションに使用するウォレットのACLアクセスを構成できます。

例10-7は、ウォレットを共有データベース・セッションに使用するように構成し、現在のデータベース・セッション内のすべてのアプリケーションがこのウォレットへのアクセス権を持ちます。

例10-7 共有データベース・セッションに使用するウォレットのACLアクセスの構成

```
/* Follow these steps:
   1. Use Oracle Wallet Manager to create the wallet and add the client
      certificate.
   2. In SQL*Plus, configure access control to grant privileges for the wallet.
      The following example grants the use_client_certificates privilege
      to the hr_clerk and hr_mgr roles. */
BEGIN
 DBMS_NETWORK_ACL_ADMIN.APPEND_WALLET_ACE (
  wallet_path => 'file:/oracle/wallets/hr_wallet',
  ace         => xs$ace_type(privilege_list => xs$name_list('use-
client_certificates'),
                              principal_name => 'hr_clerk',
                              principal_type => xs_acl.ptype_db));
 DBMS_NETWORK_ACL_ADMIN.APPEND_WALLET_ACE (
  wallet_path => 'file:/oracle/wallets/hr_wallet',
  ace         => xs$ace_type(privilege_list =>
xs$name_list('use_client_certificates'),
                              principal_name => 'hr_mgr',
                              principal_type => xs_acl.ptype_db));
END;
/
COMMIT;
/* 3. Create a request object to handle the HTTP authentication for the wallet.*/
DECLARE
  req  UTL_HTTP.req;
```

```
BEGIN
  UTL_HTTP.SET_WALLET(
    path            => 'file:/oracle/wallets/hr_wallet',
    password        => NULL);
 req := UTL_HTTP.BEGIN_REQUEST(
    url             => 'www.hr_access.example.com',
    method          => 'POST',
    http_version    => NULL,
    request_context => NULL);
END;
/
```

親トピック: 外部ネットワーク・サービスのアクセス制御の構成の例

10.8 ネットワーク・ホスト・コンピュータのグループの指定

ネットワーク・ホスト・コンピュータのグループを指定するには、ワイルドカードを使用できます。

- アクセス制御リストをネットワーク・ホスト・コンピュータのグループに割り当てるには、アスタリスク(*)ワイルドカード文字を使用します。

たとえば、ドメインに属しているホスト・コンピュータには`*.example.com`を入力し、IPサブネットに属しているIPv4アドレスには`192.0.2.*`を入力します。アスタリスク・ワイルドカードは、最初(ドメインのピリオド(.)の前)または最後(IPサブネットのピリオド(.)の後)に配置する必要があります。たとえば、`*.example.com`は有効ですが、`*example.com`や`*.example.*`は無効です。ワイルドカード文字の使用は、同じホスト・コンピュータに割り当てられている複数のアクセス制御リストの優先順位に影響を与えることに注意してください。IPv6アドレスには、ワイルドカード文字を使用できません。

クラスレス・ドメイン間ルーティング(CIDR)表記法では、インターネット上でのIPパケット・ルーティングにおけるIPv4アドレスとIPv6アドレスの分類が定義されています。`DBMS_NETWORK_ACL_ADMIN`パッケージは、IPv4アドレスとIPv6アドレスの両方に対してCIDR表記法をサポートしています。このパッケージでは、IPv4射影IPv6アドレスまたはサブネットは、IPv4ネイティブ・アドレスまたはサブネットと同等であるとみなします。たとえば、`::ffff:192.0.2.1`は`192.0.2.1`と同等で、`::ffff:192.0.2.1/120`は`192.0.2.*`と同等です。

親トピック: PL/SQLパッケージおよびタイプでのファイングレイン・アクセスの管理

10.9 複数のアクセス制御リスト割当てでのホスト・コンピュータの優先順位

ドメインに割り当てられているアクセス制御リストの優先順位は、サブ・ドメインに割り当てられているアクセス制御リストの優先順位よりも低くなります。

ホスト・コンピュータとそのドメインに割り当てられている複数のアクセス制御リストでは、ホスト・コンピュータに割り当てられているアクセス制御リストが、ドメインに割り当てられているアクセス制御リストよりも優先されます。

ドメインに割り当てられているアクセス制御リストの優先順位は、サブ・ドメインに割り当てられているアクセス制御リストの優先順位よりも低くなります。たとえば、ホスト`server.us.example.com`に割り当てられているアクセス制御リストが、そのドメインに割り当てられている他のアクセス制御リストに先行して最初に選択されます。追加のアクセス制御リストがサブ・ドメインに割り当てられていた場合、優先順位は次のようになります。

1. `server.us.example.com`
2. `*.us.example.com`
3. `*.example.com`
4. `*.com`
5. `*`

同様に、IPアドレス(IPv4とIPv6の両方)とIPアドレスが所属するサブネットに割り当てられている複数のアクセス制御リストでは、IPアドレスに割り当てられているアクセス制御リストが、サブネットに割り当てられているアクセス制御リストよりも優先されます。サブネットに割り当てられているアクセス制御リストの優先順位は、サブネットに含まれる小さいサブネットに割り当てられているアクセス制御リストの優先順位よりも低くなります。

たとえば、IPアドレス`192.0.2.3`に割り当てられているアクセス制御リストが、そのIPアドレスが所属するサブネットに割り当てられている他のアクセス制御リストに先行して最初に選択されます。追加のアクセス制御リストがサブネットに割り当てられていた場合、優先順位は次のようになります。

1. `192.0.2.3` (または `::ffff:192.0.2.3`)
2. `192.0.2.3/31` (または `::ffff:192.0.2.3/127`)
3. `192.0.2.3/30` (または `::ffff:192.0.2.3/126`)
4. `192.0.2.3/29` (または `::ffff:192.0.2.3/125`)
5. ...
6. `192.0.2.3/24` (または `::ffff:192.0.2.3/120`または`192.0.2.*`)
7. ...
8. `192.0.2.3/16` (または `::ffff:192.0.2.3/112`または`192.0.*`)
9. ...
10. `192.0.2.3/8` (または `::ffff:192.0.2.3/104`または`192.*`)
11. ...
12. `::ffff:192.0.2.3/95`
13. `::ffff:192.0.2.3/94`
14. ...
15. `*`

親トピック: PL/SQLパッケージおよびタイプでのファイングレイン・アクセスの管理

10.10 ポート範囲指定によるアクセス制御リスト割当てでのホストの優先順位

アクセス制御リスト内のホストの優先順位は、ポート範囲の使用の影響を受けます。

ポート範囲の指定があるアクセス制御リストがホスト・コンピュータ、ドメインまたはIPサブネットに割り当てられている場合、そのアクセス制御リストは、同じホスト、ドメインまたはIPサブネットに割り当てられているポート範囲の指定がないアクセス制御リストよりも優先されます。

たとえば、`server.us.example.com`のポート80から99のいずれかのポートへのTCP接続があるとします。ポート範囲外の`server.us.example.com`に割り当てられている他のアクセス制御リストに先行して、`server.us.example.com`でポート80から99に割り当てられているアクセス制御リストが最初に選択されます。

親トピック: PL/SQLパッケージおよびタイプでのファイングレイン・アクセスの管理

10.11 ネットワーク・ホストへのユーザー・アクセスに影響を与える権限割当てのチェック

管理者とユーザーは、どちらもネットワーク接続およびドメイン権限をチェックできます。

- ネットワーク・ホストへのユーザー・アクセスに影響を与える権限割当てについて
  OracleにはDBA向けのデータ・ディクショナリ・ビューが用意されており、権限の割当てに関する情報を確認できます。
- ユーザーのネットワーク接続およびドメインに対する権限のチェック方法
  データベース管理者は、DBA_HOST_ACESデータ・ディクショナリ・ビューを問い合せて、特定のユーザーまたはロールに付与された権限を確認できます。
- 例: 管理者によるユーザー・ネットワーク・アクセス制御権限のチェック
  DBA_HOST_ACESデータ・ディクショナリ・ビューでは、ユーザーのネットワーク・アクセス制御権限をチェックできます。
- ユーザーによる各自のネットワーク接続およびドメインに対する権限のチェック方法
  ユーザーはUSER_HOST_ACESデータ・ディクショナリ・ビューを問い合せて、自分のネットワークおよびドメインの権限を調べることができます。
- 例: ユーザーによるネットワーク・アクセス制御権限のチェック
  USER_HOST_ACESデータ・ディクショナリ・ビューには、ホスト・コンピュータのネットワーク・アクセス制御権限が表示されます。

親トピック: PL/SQLパッケージおよびタイプでのファイングレイン・アクセスの管理

10.11.1 ネットワーク・ホストへのユーザー・アクセスに影響を与える権限割当てについて

OracleにはDBA向けのデータ・ディクショナリ・ビューが用意されており、権限の割当てに関する情報を確認できます。

データベース管理者は、DBA_HOST_ACESデータ・ディクショナリ・ビューを使用して、アクセス制御リストでデータベース・ユーザーまたはロールに対して付与または拒否したネットワーク権限を問い合せ、それらの権限が特定の期間のみ有効かどうかを問い合せることができます

現時点でユーザーに権限が付与されているかどうか、ユーザーに割り当てられているロール、アクセス制御エントリの順序などを判断するために、ビューで提供される情報を使用したデータの組合せが必要になる場合があります。

データベース管理者権限のないユーザーには、アクセス制御リストにアクセスしたり、DBMS_NETWORK_ACL_ADMINの各ファンクションを起動する権限はありません。ただし、データベース管理者権限のないユーザーは、USER_HOST_ACESデータ・ディクショナリ・ビューを問い合せて各自の権限をチェックすることはできます。

データベース管理者とユーザーは、次のDBMS_NETWORK_ACL_UTILITYファンクションを使用して、2つのホスト、ドメインまたはサブネットが同じかどうか、あるいはホスト、ドメインまたはサブネットが他のホスト、ドメインまたはサブネットと同じ、または他のホスト、ドメインまたはサブネットに含まれるかどうか、を判断します。

- EQUALS_HOST: 2つのホスト、ドメインまたはサブネットが同じかどうかを示す値を戻します。
- CONTAINS_HOST: ホスト、ドメインまたはサブネットが他のホスト、ドメインまたはサブネットと同じかどうか、または他のホスト、ドメインまたはサブネットに含まれるかどうかを示す値を示します。また、ACL割当てのために、含まれているドメインまたはサブネットの相対的な優先順位も示します。

IPv6アドレスを使用しない場合、データベース管理者およびユーザーは、次のDBMS_NETWORK_ACL_UTILITYファンクションを使用して、ホストが所属しているドメインまたはIPv4サブネットのリストを生成し、ホストの割当てに従って優先順位別にアクセス制御リストをソートできます。

- DOMAINS: アクセス制御リストが、指定のネットワーク・ホスト、サブドメインまたはIPサブネットに対する権限に影響を与える可能性があるドメインまたはIPサブネットのリストを戻します。
- DOMAIN_LEVEL: 指定のホストのドメイン・レベルを戻します。

親トピック: ネットワーク・ホストへのユーザー・アクセスに影響を与える権限割当てのチェック

10.11.2 ユーザーのネットワーク接続およびドメインに対する権限のチェック方法

データベース管理者は、DBA_HOST_ACESデータ・ディクショナリ・ビューを問い合せて、特定のユーザーまたはロールに付与された権限を確認できます。

DBA_HOST_ACESビューには、ネットワーク接続またはドメインへのアクセスを決定するアクセス制御リストが表示され、各アクセス制御リストについて、ユーザーのアクセス権限が付与されている(GRANTED)か、拒否されている(DENIED)か、あるいは適用外(NULL)かを確認できます。このビューの問合せを実行できるのは、データベース管理者のみです。

親トピック: ネットワーク・ホストへのユーザー・アクセスに影響を与える権限割当てのチェック

10.11.3 例: 管理者によるユーザー・ネットワーク・アクセス制御権限のチェック

DBA_HOST_ACESデータ・ディクショナリ・ビューでは、ユーザーのネットワーク・アクセス制御権限をチェックできます。

例10-8は、ユーザーprestonによるwww.us.example.comへの接続について、データベース管理者がユーザーの権限をチェックする方法を示しています。

この例では、ユーザーprestonにはwww.us.example.comで見つかったすべてのネットワーク・ホスト接続の権限が付与されました。ところが、prestonはポート80でのホスト接続へのアクセス権を付与されたが、ポート3000-3999でのホスト接続へのアクセス権を拒否されたと仮定します。この場合は、ポート80でのホスト接続のためにアクセス制御を構成し、ポート3000-3999でのホスト接続のために別個のアクセス制御構成を行う必要があります。

例10-8 管理者によるユーザー・ネットワーク・アクセス制御権限のチェック

```
SELECT HOST, LOWER_PORT, UPPER_PORT,
       ACE_ORDER, PRINCIPAL, PRINCIPAL_TYPE,
       GRANT_TYPE, INVERTED_PRINCIPAL, PRIVILEGE,
       START_DATE, END_DATE
  FROM (SELECT ACES.*,
DBMS_NETWORK_ACL_UTILITY.CONTAINS_HOST('www.us.example.com', HOST) PRECEDENCE
          FROM DBA_HOST_ACES ACES)
 WHERE PRECEDENCE IS NOT NULL
 ORDER BY PRECEDENCE DESC,
          LOWER_PORT NULLS LAST,
          UPPER_PORT NULLS LAST,
          ACE_ORDER;
HOST                 LOWER_PORT UPPER_PORT ACE_ORDER PRINCIPAL PRINCIPAL_TYPE
GRANT_TYPE INVERTED_PRINCIPAL PRIVILEGE START_DATE END_DATE
------------------ ---------- ---------- --------- --------- ---------------- -------
--- ------------------ --------- ---------- --------
www.us.example.com          80         80         1 PRESTON DATABASE USER      GRANT
NO                 HTTP
www.us.example.com          80         80         2 SEBASTIAN DATABASE USER    GRANT
NO                 HTTP
*.us.example.com                                  1 ACCT_MGR DATABASE USER     GRANT
NO                 CONNECT
*                                                 1 HR_DBA DATABASE USER       GRANT
NO                 CONNECT
*                                                 1 HR_DBA DATABASE USER       GRANT
NO                 RESOLVE
```

親トピック: ネットワーク・ホストへのユーザー・アクセスに影響を与える権限割当てのチェック

10.11.4 ユーザーによる各自のネットワーク接続およびドメインに対する権限のチェック方法

ユーザーはUSER_HOST_ACESデータ・ディクショナリ・ビューを問い合せて、自分のネットワークおよびドメインの権限を調べることができます。

USER_HOST_ACESビューはPUBLICであるため、すべてのユーザーが問い合せることができます。

このビューでは、アクセス制御リストはユーザーに表示されません。このビューでユーザーの権限ステータスが判定され(GRANTEDまたはDENIED)、NULLの場合、ユーザーはアクセス制御リストが自分に適用されないときは知る必要がないため、除外されます。言い換えるとOracle Databaseが提示するのは、Oracle Databaseによってアクセス権が明示的に付与または拒否されるネットワーク・ホスト上のユーザーのみです。そのため出力には、データベース管理者固有のDBA_HOST_ACESビューからの出力に見られる*.example.comおよび*は表示されません。

親トピック: ネットワーク・ホストへのユーザー・アクセスに影響を与える権限割当てのチェック

10.11.5 例: ユーザーによるネットワーク・アクセス制御権限のチェック

USER_HOST_ACESデータ・ディクショナリ・ビューには、ホスト・コンピュータのネットワーク・アクセス制御権限が表示されます。

例10-9は、ユーザーprestonが、www.us.example.comへの接続について、自分の権限をチェックする方法を示しています。

例10-9 ユーザーによるネットワーク・アクセス制御権限のチェック

```
SELECT HOST, LOWER_PORT, UPPER_PORT, PRIVILEGE, STATUS
  FROM (SELECT ACES.*,
DBMS_NETWORK_ACL_UTILITY.CONTAINS_HOST('www.us.example.com', HOST) PRECEDENCE
          FROM USER_HOST_ACES ACES)
 WHERE PRECEDENCE IS NOT NULL
 ORDER BY PRECEDENCE DESC,
          LOWER_PORT NULLS LAST,
          UPPER_PORT NULLS LAST;

HOST                LOWER_PORT UPPER_PORT PRIVILEGE STATUS
------------------- ---------- ---------- --------- -------
www.us.example.com          80         80  HTTP      GRANTED
```

親トピック: ネットワーク・ホストへのユーザー・アクセスに影響を与える権限割当てのチェック

10.12 Javaデバッグ・ワイヤ・プロトコル操作のネットワーク・アクセスの構成

Java PL/SQLプロシージャをデバッグするには、jdwp ACL権限を付与されていることが必要です。

SQL Developer、JDeveloperまたはOracle Developer Tools For Visual Studio (ODT)などのJavaデバッグ・ワイヤ・プロトコル(JDWP)ベースのデバッガを通じて、データベースのJavaおよびPL/SQLプロシージャをデバッグする場合、jdwp ACL権限を付与して、特定のホストのデバッガにデータベース・セッションを接続する必要があります。

DEBUG CONNECT SESSIONシステム権限と組み合せたjdwp権限が必要です。

jdwp ACL権限が付与されていない場合、リモート・ホストからJavaおよびPL/SQLストアド・プロシージャをデバッグしようとするときに、次のエラーが表示される可能性があります。

```
ORA-24247: network access denied by access control list (ACL)
```

```
ORA-06512: at "SYS.DBMS_DEBUG_JDWP", line line_number
```

- JDWP操作のネットワーク・アクセスを構成するには、DBMS_NETWORK_ACL_ADMIN.APPEND_HOST_ACEプロシージャを使用します。

次の例は、JDWP操作のネットワーク・アクセスの構成方法を示しています。

```
BEGIN
  DBMS_NETWORK_ACL_ADMIN.APPEND_HOST_ACE(
    host         => 'host',
    lower_port   => null|port_number,
    upper_port   => null|port_number,
    ace => xs$ace_type(privilege_list => xs$name_list('jdwp'),
                       principal_name => 'username',
                       principal_type => xs_acl.ptype_db));
END;
/
```

詳細は、次のとおりです。

- hostは、ホスト名、ドメイン名、IPアドレスまたはサブネットのいずれかです。
- port_numberでは、ポートの範囲を指定できます。ポートを使用する場合、lower_portおよびupper_port値を省略します。
- 引用符で囲まれていないかぎり(たとえば、principal_name => '"PSMITH"'など)、usernameは大/小文字を区別しません。

関連項目:

- JDWPを使用したサーバー・アプリケーションのデバッグの詳細は、『Oracle Database Java開発者ガイド』を参照してください。
- SQL Developerのリモート・デバッグの詳細は、Oracle SQL開発者ユーザーズ・ガイドを参照してください。

親トピック: PL/SQLパッケージおよびタイプでのファイングレイン・アクセスの管理

10.13 ユーザー・アクセス用に構成されたアクセス制御リストのデータ・ディクショナリ・ビュー

Oracle Databaseでは、既存のアクセス制御リストに関する情報の検索に使用できるデータ・ディクショナリ・ビューが提供されています。

表10-1に、これらのビューを示します。

表10-1 アクセス制御リストに関する情報を表示するデータ・ディクショナリ・ビュー

ビュー	説明
DBA_HOST_ACES	ネットワーク・ホストに定義されたネットワーク権限が表示されます。このビューのSELECT 権限は、SELECT_CATALOG_ROLE ロールのみに付与されます。
DBA_WALLET_ACES	ウォレット・パス、ACE 順序、開始時間と終了時間、付与タイプ、権限およびプリン

ビュー	説明
	シパルの情報がリストされます
DBA_WALLET_ACLS	ウォレットに対するアクセス制御リスト割当てが表示されます。このビューの SELECT 権限は、SELECT_CATALOG_ROLE ロールのみに付与されます。
DBA_HOST_ACLS	ネットワーク・ホストに対するアクセス制御リスト割当てが表示されます。このビューの SELECT 権限は、SELECT_CATALOG_ROLE ロールのみに付与されます。
USER_HOST_ACES	現行ユーザーがネットワーク・ホストにアクセスするためのネットワーク権限のステータスが表示されます。このビューの SELECT 権限は PUBLIC に付与されます。
USER_WALLET_ACES	ウォレットのコンテンツにアクセスするため、現在のユーザーのウォレット権限のステータスが表示されます。このビューの SELECT 権限は PUBLIC に付与されます。

関連トピック

- Oracle Databaseリファレンス

親トピック: PL/SQLパッケージおよびタイプでのファイングレイン・アクセスの管理

11 Enterprise Managerによるマルチテナント環境のセキュリティの管理

Oracle Enterprise Managerを使用して、マルチテナント環境の共通およびローカルのユーザーとロールを管理できます。

この項では、次の項目について説明します。

- Enterprise Managerによるマルチテナント環境のセキュリティの管理について
  Oracle Enterprise Manager Cloud Controlではマルチテナント環境のセキュリティ管理がサポートされます。
- Enterprise Managerによるマルチテナント環境へのログイン
  マルチテナント環境では、CDBまたはPDBにログインしたり、PDBから別のPDBまたはルートに切り替えることができます。
- Enterprise Managerの共通ユーザーおよびローカル・ユーザーの管理
  マルチテナント環境では、Oracle Enterprise Managerを使用すると、共通ユーザーとローカル・ユーザーを作成、編集および削除できます。
- Enterprise Managerの共通およびローカル・ロールおよび権限の管理
  マルチテナント環境では、Oracle Enterprise Managerを使用して、共通ロールとローカル・ロールの作成、編集、削除および取消しを行うことができます。

親トピック: ユーザー認証および認可の管理

11.1 Enterprise Managerによるマルチテナント環境のセキュリティの管理について

Oracle Enterprise Manager Cloud Controlではマルチテナント環境のセキュリティ管理がサポートされます。

マルチテナント環境では、Oracle Enterprise Manager Cloud Controlを使用して、ルートと関連のプラガブル・データベース(PDB)の両方の共通ユーザーと共通ロールの作成、管理および監視ができます。

Enterprise Managerによって、ルートと指定されたPDBの間を容易に切り替えることができます。

親トピック: Enterprise Managerによるマルチテナント環境のセキュリティの管理

11.2 Enterprise Managerによるマルチテナント環境へのログイン

マルチテナント環境では、CDBまたはPDBにログインしたり、PDBから別のPDBまたはルートに切り替えることができます。

この項では、次の項目について説明します。

- CDBまたはPDBへのログイン
  様々な種類のEnterprise Managerデータベース・ログイン・ページが、ログイン時にリクエストした機能に基づいて自動的に表示されます。
- 別のPDBへの、またはルートへの切替え
  Oracle Enterprise Managerで、PDBから別のPDBまたはルートに切り替えることができます。

親トピック: Enterprise Managerによるマルチテナント環境のセキュリティの管理

11.2.1 CDBまたはPDBへのログイン

様々な種類のEnterprise Managerデータベース・ログイン・ページが、ログイン時にリクエストした機能に基づいて自動的に表示されます。

CDB管理者(CDBターゲットに対してCONNECT権限を持つEnterprise Managerユーザー)としてマルチテナント環境にログインしてCDBの範囲の機能を使用するには:

1. ユーザーSYSTEMまたはSYSMANとしてOracle Enterprise Manager Cloud Controlにログインします。

   URLは次のとおりです。

   ```
   https://host:port/em
   ```

2. 「データベース」ページに移動します。
3. アクセスするデータベースを選択します。

   データベースのホームページが表示されます。

4. 実行するアクションのメニュー項目を選択します。たとえば、ユーザーを認証するには「管理」、「セキュリティ」、「ユーザー」の順に選択します。

   「データベース・ログイン」ページが表示されます。次に示す例はCDBの「データベース・ログイン」ページです(表示されているデータベース名がCDB$ROOTのため)。この名前から、このページは口語的にマルチテナント環境のrootのデータベース・ログイン・ページと呼ばれます。「データベース」フィールドは現在のデータベースを示します。PDBを選択した場合、PDBの名前がこのフィールドに表示されます。

5. 適切な資格証明を使用してログインします。

   共通ユーザーのみがルートにログインでき、共通ユーザーの名前はC##またはc##で始まることに注意してください。PDBには、共通ユーザーとローカル・ユーザーの両方がそれぞれの権限に応じてログインできます。

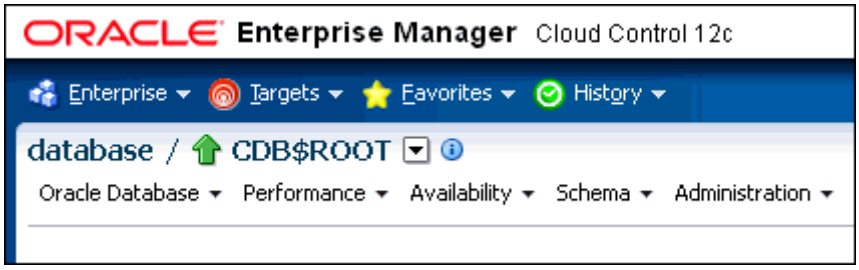
親トピック: Enterprise Managerによるマルチテナント環境へのログイン

11.2.2 別のPDBへの、またはルートへの切替え

Oracle Enterprise Managerで、PDBから別のPDBまたはルートに切り替えることができます。

1. ページの左上に「データベース」リンクがあります。

   「データベース」リンクに、現在のコンテナ名が表示されます。次に示す例では、現在のデータベースは、口語的にルートと呼ばれるCDB自体(CDB$ROOT)です。



2. コンテナの右にあるメニュー・アイコンを選択し、このメニューから、アクセスするデータベースを選択します。

   メニュー項目が表示されない場合は、データベースのホームページなど、メニュー項目が表示されるページに移動します。

3. 実行するアクティビティ(ユーザーの作成など)を決定する際は、適切な権限でログインします。

   あらかじめ適切な権限で認証を行わずにアクティビティを実行しようとすると、適切な権限でログインするよう要求されます。

親トピック: Enterprise Managerによるマルチテナント環境へのログイン

11.3 Enterprise Managerの共通ユーザーおよびローカル・ユーザーの管理

マルチテナント環境では、Oracle Enterprise Managerを使用すると、共通ユーザーとローカル・ユーザーを作成、編集および削除できます。

この項では、次の項目について説明します。

- Enterprise Managerの共通ユーザー・アカウントの作成
  共通ユーザーは、ルートに存在し、CDBのPDBにアクセスできるユーザーです。
- Enterprise Managerの共通ユーザー・アカウントの編集
  共通ユーザー・アカウントは、ルートから編集できます。
- Enterprise Managerの共通ユーザー・アカウントの削除
  共通ユーザーは、CDBルートから削除できます。
- Enterprise Managerのローカル・ユーザー・アカウントの作成
  ローカル・ユーザーは、特定のPDBにのみ存在し、マルチテナント環境の他のPDBにアクセスできないユーザーです。
- Enterprise Managerのローカル・ユーザー・アカウントの編集
  ローカル・ユーザーは、そのローカル・ユーザーが存在するPDBから編集できます。
- Enterprise Managerのローカル・ユーザー・アカウントの削除
  ローカル・ユーザーは、そのローカル・ユーザーが存在するPDBから削除できます。

親トピック: Enterprise Managerによるマルチテナント環境のセキュリティの管理

11.3.1 Enterprise Managerの共通ユーザー・アカウントの作成

共通ユーザーは、ルートに存在し、CDBのPDBにアクセスできるユーザーです。

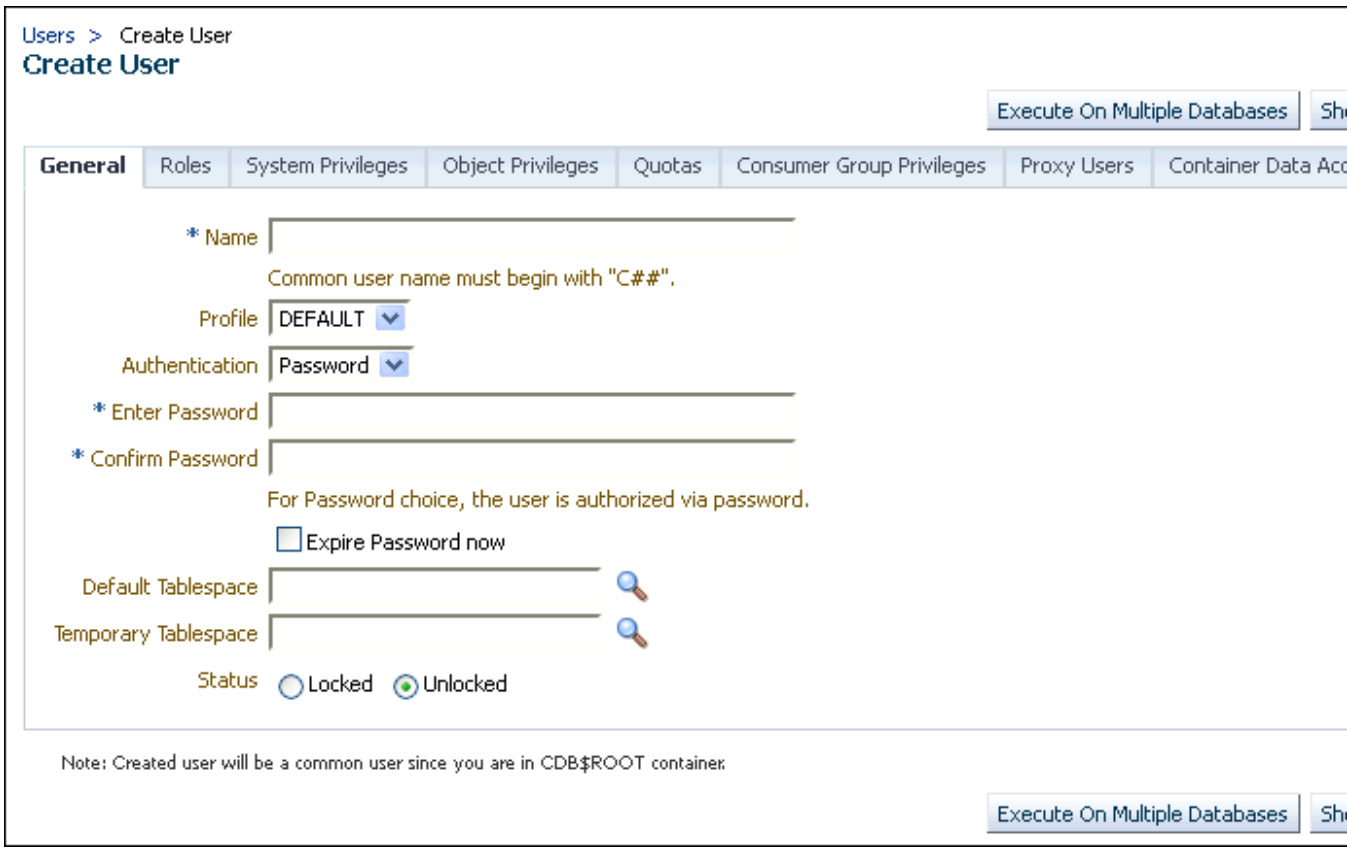
1. Enterprise Managerデータベースのホームページで、共通`CREATE USER`および`SET CONTAINER`権限を持つ共通ユーザーとしてルートにログインします。

2. 「管理」メニューから、「セキュリティ」を選択し、「ユーザー」を選択します。

   要求された場合は、ログイン情報を入力します。その後、「ユーザー」ページが表示されます。

3. 「作成」をクリックします。

   「ユーザーの作成」ページが表示されます。



4. 共通ユーザーを作成するオプションを選択して、このユーザーに権限を付与します。

   ユーザー名の前にC##またはc##を付けてください。

5. 「OK」または「適用」をクリックします。

   共通ユーザーは、ルートで作成され、関連付けられているPDBの「ユーザー」ページに表示されます。

関連トピック

- Enterprise Managerによるマルチテナント環境へのログイン

親トピック: Enterprise Managerの共通ユーザーおよびローカル・ユーザーの管理

11.3.2 Enterprise Managerの共通ユーザー・アカウントの編集

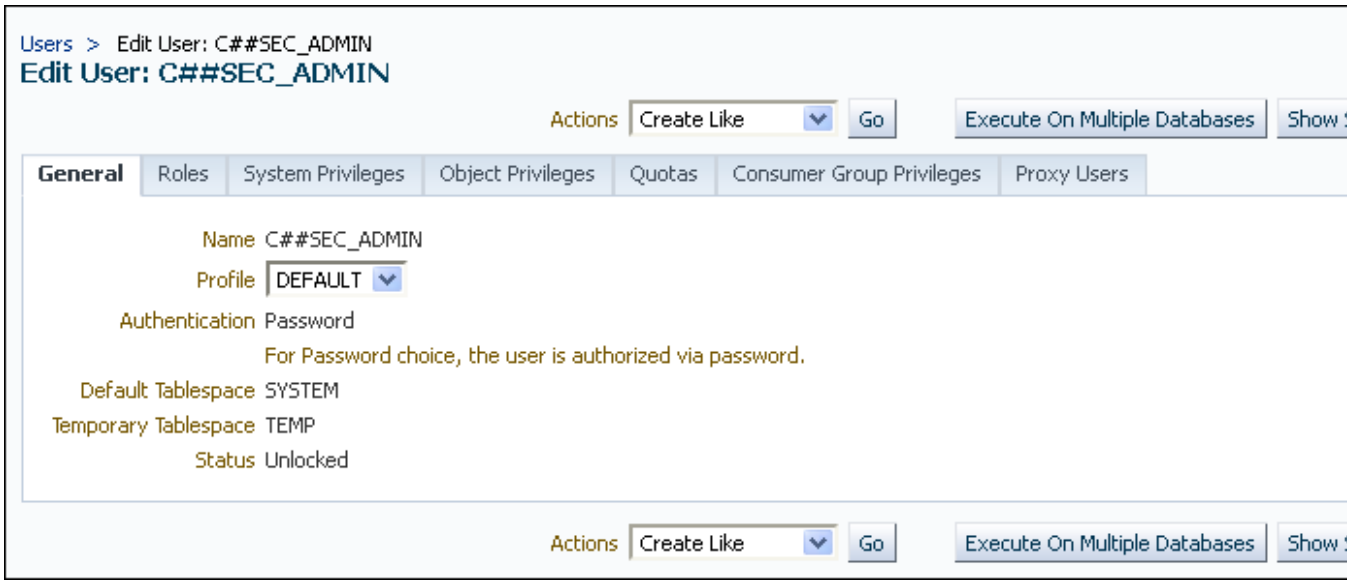
共通ユーザー・アカウントは、ルートから編集できます。

1. Enterprise Managerデータベースのホームページで、共通CREATE USERおよびSET CONTAINER権限を持つ共通ユーザーとしてルートにログインします。
   - ルートにログインする場合は、共通CREATE USERおよびSET CONTAINER権限を持つ共通ユーザーである必要があります。
   - PDBにログインする場合は、そのPDBのCREATE USER権限を持っていることを確認してください。
2. 「管理」メニューから、「セキュリティ」を選択し、「ユーザー」を選択します。

要求された場合は、ログイン情報を入力します。その後、「ユーザー」ページが表示されます。ルートでは、共通ユーザーのみが表示されます。PDBでは、共通ユーザーとローカル・ユーザーの両方がリストされます。

3. 編集する共通ユーザーを選択して、「編集」をクリックします。

   ユーザーの編集ページが表示されます。ルートの共通ユーザーについては、その共通ユーザーのすべての設定を変更できます。PDBの共通ユーザーについては、ユーザー・パスワード、デフォルト表領域、一時表領域は変更できません。設定の変更は、現在のPDBにのみ適用されます。次の画面に、PDBでの共通ユーザーの「ユーザーの編集」ページを示します。



4. 必要に応じて、共通ユーザーを変更します。
5. 「適用」をクリックします。

関連トピック

- Enterprise Managerによるマルチテナント環境へのログイン
- 共通またはローカル・ユーザー・アカウントを変更する方法

親トピック: Enterprise Managerの共通ユーザーおよびローカル・ユーザーの管理

11.3.3 Enterprise Managerの共通ユーザー・アカウントの削除

共通ユーザーは、CDBルートから削除できます。

1. Enterprise Managerデータベースのホームページで、共通`CREATE USER`および`SET CONTAINER`権限を持つ共通ユーザーとしてルートにログインします。
   PDBから共通ユーザーを削除することはできません。
2. 「管理」メニューから、「セキュリティ」を選択し、「ユーザー」を選択します。

   要求された場合は、ログイン情報を入力します。その後、共通ユーザーのみをリストする「ユーザー」が表示されます。

3. 削除する共通ユーザーを選択して、「削除」をクリックします。
4. 共通ユーザーの削除を確定します。

関連トピック

- Enterprise Managerによるマルチテナント環境へのログイン

親トピック: Enterprise Managerの共通ユーザーおよびローカル・ユーザーの管理

11.3.4 Enterprise Managerのローカル・ユーザー・アカウントの作成

ローカル・ユーザーは、特定のPDBにのみ存在し、マルチテナント環境の他のPDBにアクセスできないユーザーです。

1. Enterprise Managerデータベースのホームページで、ローカルCREATE USER権限を持つローカル・ユーザーまたは共通ユーザーとしてルートにログインします。
2. 「管理」メニューから、「セキュリティ」を選択し、「ユーザー」を選択します。

   要求された場合は、ログイン情報を入力します。その後、現在のPDBのローカル・ユーザーのみを示す「ユーザー」ページが表示されます。

3. 「作成」をクリックします。

   「ユーザーの作成」ページが表示されます。

4. ローカル・ユーザーを作成するオプションを選択して、このユーザーに権限を付与します。

   ユーザー名の前にC##またはc##を付けないでください。

5. 「OK」をクリックします。

   現在のPDBでローカル・ユーザーが作成されます。

関連トピック

- Enterprise Managerによるマルチテナント環境へのログイン
- ローカル・ユーザー・アカウントの作成について

親トピック: Enterprise Managerの共通ユーザーおよびローカル・ユーザーの管理

11.3.5 Enterprise Managerのローカル・ユーザー・アカウントの編集

ローカル・ユーザーは、そのローカル・ユーザーが存在するPDBから編集できます。

1. Enterprise Managerデータベースのホームページで、ローカルCREATE USER権限を持つローカル・ユーザーまたは共通ユーザーとしてPDBにログインします。
2. 「管理」メニューから、「セキュリティ」を選択し、「ユーザー」を選択します。

   要求された場合は、ログイン情報を入力します。その後、現在のPDBのローカル・ユーザーおよび共通ユーザーのみを示す「ユーザー」ページが表示されます。

3. 編集するローカル・ユーザーを選択して、「編集」をクリックします。

   ユーザーの編集ページが表示されます。

4. 必要に応じて、ローカル・ユーザーを変更します。
5. 「適用」をクリックします。

関連トピック

- Enterprise Managerによるマルチテナント環境へのログイン
- 共通またはローカル・ユーザー・アカウントを変更する方法

親トピック: Enterprise Managerの共通ユーザーおよびローカル・ユーザーの管理

11.3.6 Enterprise Managerのローカル・ユーザー・アカウントの削除

ローカル・ユーザーは、そのローカル・ユーザーが存在するPDBから削除できます。

1. Enterprise Managerデータベースのホームページで、ローカルCREATE USER権限を持つローカル・ユーザーまたは共通ユーザーとしてPDBにログインします。
2. 「管理」メニューから、「セキュリティ」を選択し、「ユーザー」を選択します。

   要求された場合は、ログイン情報を入力します。その後、現在のPDBのローカル・ユーザーおよび共通ユーザーのみを示す「ユーザー」ページが表示されます。(PDBから共通ユーザーを削除することはできません。)

3. 削除するローカル・ユーザーを選択して、「削除」をクリックします。

   ユーザーを削除してよいかどうか、確認を求められます。

4. ローカル・ユーザーの削除を確定します。

関連トピック

- Enterprise Managerによるマルチテナント環境へのログイン

親トピック: Enterprise Managerの共通ユーザーおよびローカル・ユーザーの管理

11.4 Enterprise Managerの共通およびローカル・ロールおよび権限の管理

マルチテナント環境では、Oracle Enterprise Managerを使用して、共通ロールとローカル・ロールの作成、編集、削除および取消しを行うことができます。

この項では、次の項目について説明します。

- Enterprise Managerの共通ロールの作成
  共通ロールを使用して、共通権限を共通ユーザーに割り当てることができます。
- Enterprise Managerの共通ロールの編集
  共通ロールは、ルートから編集できます。
- Enterprise Managerの共通ロールの削除
  共通ロールは、ルートから削除できます。
- Enterprise Managerの共通権限付与の取消し
  共通権限付与は、ルートから取り消すことができます。
- Enterprise Managerのローカル・ロールの作成
  共通ロールを使用して、後でローカル・ユーザーに権限のローカル・セットを割り当てることができます。
- Enterprise Managerのローカル・ロールの編集
  ローカル・ロールは、そのローカル・ロールが存在するPDBで編集できます。
- Enterprise Managerのローカル・ロールの削除
  ローカル・ロールは、そのローカル・ロールが存在するPDBから削除できます。
- Enterprise Managerのローカル権限付与の取消し
  ローカル権限は、その権限が使用されるPDBで取り消すことができます。

親トピック: Enterprise Managerによるマルチテナント環境のセキュリティの管理

11.4.1 Enterprise Managerの共通ロールの作成

共通ロールを使用して、共通権限を共通ユーザーに割り当てることができます。

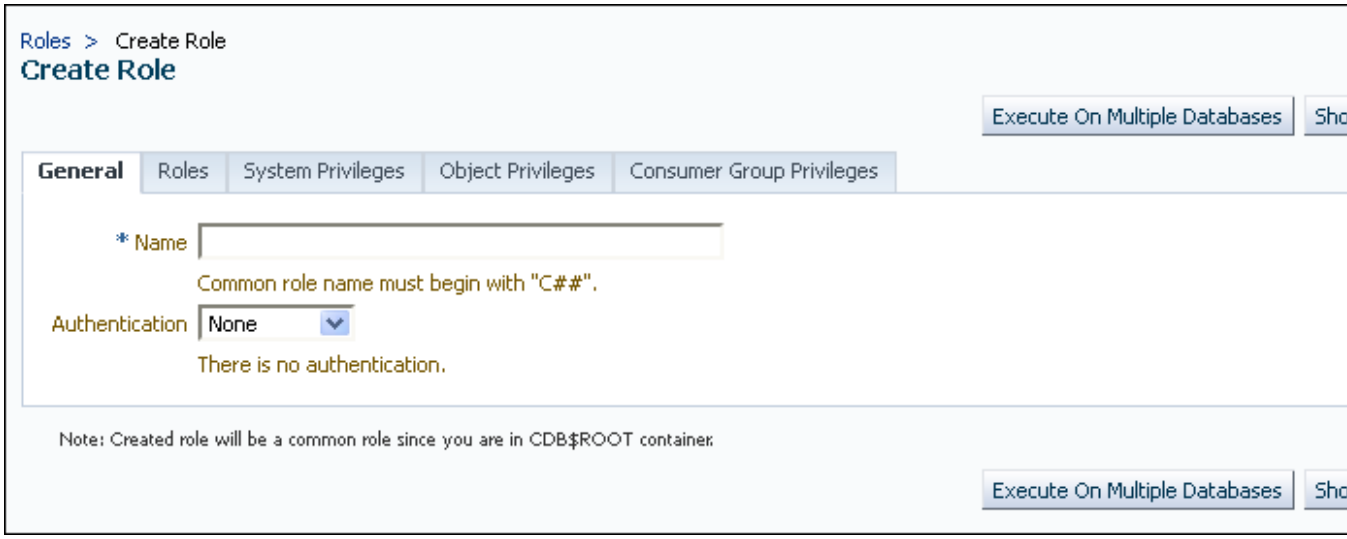
これらのロールは、マルチテナント環境のすべてのコンテナで有効です。

1. Enterprise Managerデータベースのホームページで、共通`CREATE ROLE`および`SET CONTAINER`権限を持つ共通ユーザーとしてルートにログインします。
2. 「管理」メニューから、「セキュリティ」を選択し、「ロール」を選択します。

   要求された場合は、ログイン情報を入力します。その後、「ロールの作成」ページが表示されます。

3. 「作成」をクリックします。

   「ロールの作成」ページが表示されます。



4. 共通ロールを作成するオプションを選択して、このロールに権限を付与します。

   ロール名の前に`C##`または`c##`を付けてください。

5. 「OK」をクリックします。

   共通ロールがルートに作成されます。

関連トピック

- Enterprise Managerによるマルチテナント環境へのログイン
- 共通ロールの作成の規則
- PDBへのアクセス権限の付与または取消し

親トピック: Enterprise Managerの共通およびローカル・ロールおよび権限の管理

11.4.2 Enterprise Managerの共通ロールの編集

共通ロールは、ルートから編集できます。

1. Enterprise Managerデータベースのホームページで、ルートまたはPDBにログインします。ルートにログインする場合は、共通`CREATE ROLE`および`SET CONTAINER`権限を持つ共通ユーザーである必要があります。PDBにログインする場合は、そのPDBの`CREATE ROLE`権限を持っていることを確認してください。
2. 「管理」メニューから、「セキュリティ」を選択し、「ロール」を選択します。

要求された場合は、ログイン情報を入力します。その後、「ロール」ページが表示されます。ルートでは、共通ロールのみが表示されます。PDBでは、共通ロールとローカル・ロールの両方が表示されます。

3. 編集する共通ロールを選択して、「編集」をクリックします。

   ロールの編集ページが表示されます。ルートの共通ユーザーについては、その共通ユーザーのすべての設定を変更できます。

   PDBの共通ロールについては、ロールの認証のみ変更可能で、このユーザーに別のロール、システム権限、オブジェクト権限およびコンシューマ・グループ権限を付与します。これらの設定は現在のPDBにのみ適用されます。

4. 必要に応じて、共通ユーザーを変更します。
5. 「適用」をクリックします。

関連トピック

- Enterprise Managerによるマルチテナント環境へのログイン

親トピック: Enterprise Managerの共通およびローカル・ロールおよび権限の管理

11.4.3 Enterprise Managerの共通ロールの削除

共通ロールは、ルートから削除できます。

1. Enterprise Managerデータベースのホームページで、共通CREATE ROLEおよびSET CONTAINER権限を持つ共通ユーザーとしてルートにログインします。
   PDBから共通ロールを削除することはできません。
2. 「管理」メニューから、「セキュリティ」を選択し、「ロール」を選択します。

   要求された場合は、ログイン情報を入力します。その後、共通ロールのみを示す「ロール」ページが表示されます。

3. 削除する共通ロールを選択して、「削除」をクリックします。
4. 共通ロールの削除を確定します。

関連トピック

- Enterprise Managerによるマルチテナント環境へのログイン

親トピック: Enterprise Managerの共通およびローカル・ロールおよび権限の管理

11.4.4 Enterprise Managerの共通権限付与の取消し

共通権限付与は、ルートから取り消すことができます。

1. Enterprise Managerデータベースのホームページで、共通CREATE USER、CREATE ROLEおよびSET CONTAINER権限を持つ共通ユーザーとしてルートにログインします。
2. 「管理」メニューから、「セキュリティ」を選択し、「ユーザー」を選択します。

   「ユーザー」ページに、共通ユーザーが表示されます。

3. 権限を取り消すユーザーを選択して、「編集」をクリックします。

   ユーザーの編集ページが表示されます。

4. 「ロール」または該当する「権限」タブを選択します。

   このユーザーに割り当てられているロールと権限のリストが表示されます。

5. 「リストの編集」を選択し、必要に応じてロールまたは権限を削除します。
6. 「OK」ボタンをクリックします。

関連トピック

- Enterprise Managerによるマルチテナント環境へのログイン
- PDBへのアクセス権限の付与または取消し

親トピック: Enterprise Managerの共通およびローカル・ロールおよび権限の管理

11.4.5 Enterprise Managerのローカル・ロールの作成

共通ロールを使用して、後でローカル・ユーザーに権限のローカル・セットを割り当てることができます。

これらのロールは、定義対象のPDBコンテナ全体で有効になります。

1. Enterprise Managerデータベースのホームページで、ローカルCREATE ROLE権限を持つユーザーとしてPDBにログインします。
2. 「管理」メニューから、「セキュリティ」を選択し、「ロール」を選択します。

   ロール・ページが表示されます。
3. 「作成」をクリックします。

   要求された場合は、ログイン情報を入力します。その後、「ロールの作成」ページが表示されます。
4. ローカル・ロールを作成するオプションを選択して、このロールに権限を付与します。

   ロール名の前にC##またはc##を付けないでください。
5. 「OK」をクリックします。

   現在のPDBでローカル・ロールが作成されます。

関連トピック

- Enterprise Managerによるマルチテナント環境へのログイン
- PDBへのアクセス権限の付与または取消し

親トピック: Enterprise Managerの共通およびローカル・ロールおよび権限の管理

11.4.6 Enterprise Managerのローカル・ロールの編集

ローカル・ロールは、そのローカル・ロールが存在するPDBで編集できます。

1. Enterprise Managerデータベースのホームページで、ローカルCREATE ROLE権限を持つユーザーとしてPDBにログインします。
2. 「管理」メニューから、「セキュリティ」を選択し、「ロール」を選択します。

   要求された場合は、ログイン情報を入力します。その後、現在のPDBのローカル・ロールおよび共通ロールのみを示す「ロール」ページが表示されます。
3. 編集するローカル・ロールを選択して、「編集」をクリックします。

   ユーザーの編集ページが表示されます。
4. 必要に応じて、ローカル・ユーザーを変更します。
5. 「適用」をクリックします。

関連トピック

- Enterprise Managerによるマルチテナント環境へのログイン

親トピック: Enterprise Managerの共通およびローカル・ロールおよび権限の管理

11.4.7 Enterprise Managerのローカル・ロールの削除

ローカル・ロールは、そのローカル・ロールが存在するPDBから削除できます。

1. Enterprise Managerデータベースのホームページで、ローカルCREATE ROLE権限を持つユーザーとしてPDBにログインします。
2. 「管理」メニューから、「セキュリティ」を選択し、「ロール」を選択します。

   要求された場合は、ログイン情報を入力します。その後、現在のPDBのローカル・ロールおよび共通ロールのみを示す「ロール」ページが表示されます。(PDBから共通ロールを削除することはできません。)

3. 削除するローカル・ロールを選択して、「削除」をクリックします。

   ロールを削除してよいかどうか、確認を求められます。

4. ローカル・ロールの削除を確定します。

関連トピック

- Enterprise Managerによるマルチテナント環境へのログイン

親トピック: Enterprise Managerの共通およびローカル・ロールおよび権限の管理

11.4.8 Enterprise Managerのローカル権限付与の取消し

ローカル権限は、その権限が使用されるPDBで取り消すことができます。

1. Enterprise Managerデータベースのホームページで、CREATE USERおよびCREATE ROLE権限を持つ共通ユーザーまたはローカル・ユーザーとしてPDBにログインします。
2. 「管理」メニューから、「セキュリティ」を選択し、「ユーザー」を選択します。

   要求された場合は、ログイン情報を入力します。その後、「ユーザー」ページが表示されます。PDBでは、共通ユーザーとローカル・ユーザーの両方がリストされます。

3. 権限を取り消すユーザーを選択して、「編集」をクリックします。

   ユーザーの編集ページが表示されます。

4. 「ロール」または該当する「権限」タブを選択します。

   このユーザーに割り当てられているロールと権限のリストが表示されます。

5. 「リストの編集」を選択し、必要に応じて権限を削除します。
6. 「OK」ボタンをクリックします。

関連トピック

- Enterprise Managerによるマルチテナント環境へのログイン
- PDBへのアクセス権限の付与または取消し

親トピック: Enterprise Managerの共通およびローカル・ロールおよび権限の管理

第II部 アプリケーション開発のセキュリティ

第II部では、アプリケーション開発のセキュリティの管理方法について説明します。

- アプリケーション開発者のセキュリティの管理
  アプリケーション開発者のセキュリティ・ポリシーでは、パスワード管理や外部プロシージャおよびアプリケーション権限の保護といった領域を網羅する必要があります。

12 アプリケーション開発者のセキュリティの管理

アプリケーション開発者のセキュリティ・ポリシーでは、パスワード管理や外部プロシージャおよびアプリケーション権限の保護といった領域を網羅する必要があります。

- アプリケーション・セキュリティ・ポリシーについて
  アプリケーション・セキュリティ・ポリシーは、アプリケーション・セキュリティの要件およびデータベース・オブジェクトへのユーザー・アクセスを規制するルールのリストです。
- アプリケーション・ベースのセキュリティの使用に関する考慮事項
  アプリケーション・セキュリティの実装では、アプリケーション・ユーザーとデータベース・ユーザーについて、アプリケーション内とデータベース内のどちらでセキュリティを強制するのかを検討する必要があります。
- アプリケーション設計におけるパスワードの保護
  Oracleでは、パスワード・サービスの安全な呼出し(スクリプトを使用するなど)のための戦略や、これらの戦略をその他の機密データに適用するための戦略を用意しています。
- 外部プロシージャの保護
  外部プロシージャはデータベースとは別に.dllまたは.soファイルに保存され、資格証明認証で保護できます。
- LOBロケータの署名を使用したLOBの保護
  LOBロケータの署名を再生成することで、ラージ・オブジェクト(LOB)を保護できます。
- アプリケーション権限の管理
  ほとんどのデータベース・アプリケーションでは、異なるスキーマ・オブジェクトごとに異なる権限が関与します。
- アプリケーション権限の管理にロールを使用する利点
  複数のアプリケーション権限を1つのロールにグループ化すると、権限の管理に役立ちます。
- アプリケーションへのアクセスを制御するセキュア・アプリケーション・ロールの作成
  セキュア・アプリケーション・ロールは関連するPL/SQLパッケージまたはプロシージャでのみ使用可能にできます。
- 権限とユーザーのデータベース・ロールとの関連付け
  ユーザーの権限が、現在のデータベース・ロールに関連した権限のみであることを確認します。
- スキーマを使用したデータベース・オブジェクトの保護
  スキーマとは、データベース・オブジェクトを含めることができるセキュリティ・ドメインです。ユーザーおよびロールに付与された権限によって、これらのデータベース・オブジェクトへのアクセスが制御されます。
- アプリケーションでのオブジェクト権限
  アプリケーションの設計時には、ユーザーのタイプとユーザーに必要なレベル・アクセスについて検討する必要があります。
- データベース通信のセキュリティを強化するためのパラメータ
  プロトコル・エラーによる不正パケットの処理や認証エラーの上限の構成など、パラメータを使用してセキュリティを管理できます。

親トピック: アプリケーション開発のセキュリティ

12.1 アプリケーション・セキュリティ・ポリシーについて

アプリケーション・セキュリティ・ポリシーは、アプリケーション・セキュリティの要件およびデータベース・オブジェクトへのユーザー・アクセスを規制するルールのリストです。

安全なデータベース・アプリケーションを作成する最初のステップは、アプリケーション・セキュリティ・ポリシーの作成です。セキュリティ・ポリシーは、データベース・アプリケーションごとに立案する必要があります。たとえば、各データベース・アプリケーションには、アプリケーションの実行時に異なるレベルのセキュリティを提供する1つ以上のデータベース・ロールが必要です。データベース・ロール

は、他のロールに付与するか、または特定のユーザーに直接付与できます。

(SQL*PlusやSQL Developerなどのツールを使用して)SQL文を制限なしで処理できるアプリケーションの場合でも、機密扱いのスキーマ・オブジェクトや重要なスキーマ・オブジェクトへの不当なアクセスを防ぐために、セキュリティ・ポリシーが必要です。特に、使用するアプリケーションでパスワードが安全に処理されることを確認する必要があります。

親トピック: アプリケーション開発者のセキュリティの管理

12.2 アプリケーション・ベースのセキュリティの使用に関する考慮事項

アプリケーション・セキュリティの実装では、アプリケーション・ユーザーとデータベース・ユーザーについて、アプリケーション内とデータベース内のどちらでセキュリティを強制するのかを検討する必要があります。

- アプリケーション・ユーザーはデータベース・ユーザーでもあるか
  できるかぎり、アプリケーション・ユーザーがデータベース・ユーザーであるアプリケーションを作成し、データベースに備わるセキュリティ・メカニズムを使用できるようにします。
- アプリケーション内またはデータベース内でのセキュリティ規定
  アプリケーションでできるだけ多くのデータベース・セキュリティ・メカニズムが使用されるようにすることをお薦めします。

親トピック: アプリケーション開発者のセキュリティの管理

12.2.1 アプリケーション・ユーザーはデータベース・ユーザーでもあるか

できるかぎり、アプリケーション・ユーザーがデータベース・ユーザーであるアプリケーションを作成し、データベースに備わるセキュリティ・メカニズムを使用できるようにします。

多くの市販のパッケージ・アプリケーションでは、アプリケーション・ユーザーは、データベース・ユーザーではありません。これらのアプリケーションでは、複数のユーザーがユーザー自身をアプリケーションに対して認証し、次に、アプリケーションが単一の高い権限を持つユーザーとしてデータベースに接続します。これを、One Big Application Userモデルと呼びます。

この方法で作成されたアプリケーションは、通常、データベースに本来備わっているセキュリティ機能の多くを使用できません。これは、ユーザーの識別情報がデータベースで認識されないためです。ただし、クライアント識別子を使用して、いくつかのタイプの追跡を実行できます。たとえば、Oracle Call InterfaceのメソッドOCIAttrSetのOCI_ATTR_CLIENT_IDENTIFIER属性を使用して、クライアント接続の監査と監視を有効にできます。クライアント識別子を使用すると、個々のWebユーザーの監査証跡データの収集、Webユーザーによるデータ・アクセスを制限するルールの適用、および各Webユーザーが使用しているアプリケーションの監視および追跡を行うことができます。

表12-1は、One Big Application Userモデルが様々なOracle Databaseセキュリティ機能にどのように影響するかを説明しています。

表12-1 One Big Application Userモデルの影響を受ける機能

Oracle Database機能	One Big Application Userモデルの制限
監査	セキュリティの基本原則の 1 つは、監査によるアカウンタビリティです。One Big Application User がデータベース内のすべての操作を実行する場合、データベース監査では、個々のユーザーが実行したアクションに対する責任をそのユーザー自身に持たせることができません。アプリケーションでは、独自の監査メカニズムを実装して、ユーザーのアクションを個別に捕捉する必要があります。

Oracle Database機能	One Big Application Userモデルの制限
Oracle 厳密認証	データベースに対するクライアントの認証が、個々のユーザーではなくアプリケーションの場合、認証の厳密な形式(たとえば、SSL やトークンなどでのクライアント認証)は使用できません。
ロール	ロールは、データベース・ユーザーに割り当てられます。エンタープライズ・ロールは、データベース内で作成されていない場合でも、エンタープライズ・ユーザーに割り当てられ、データベースに認識されます。アプリケーション・ユーザーがデータベース・ユーザーではない場合、ロールの有用性は低くなります。この場合、アプリケーションでは独自のメカニズムを作成して、様々なアプリケーション・ユーザーがアプリケーション内のデータへアクセスするのに必要とする権限を識別する必要があります。
エンタープライズ・ユーザー管理	エンタープライズ・ユーザー管理機能は、Oracle Internet Directory などの LDAP ベースのディレクトリにユーザー情報を安全に格納して認可を管理することで、Oracle データベースが Oracle Identity Management インフラストラクチャを使用できるようにします。エンタープライズ・ユーザーは、データベース内に作成する必要はありませんが、データベースに認識される必要があります。One Big Application User モデルでは、Oracle Identity Management を利用できません。

親トピック: アプリケーション・ベースのセキュリティの使用に関する考慮事項

12.2.2 アプリケーション内またはデータベース内でのセキュリティ規定

アプリケーションでできるだけ多くのデータベース・セキュリティ・メカニズムが使用されるようにすることをお薦めします。

ユーザーがデータベース・ユーザーでもあるアプリケーションは、アプリケーションの中にセキュリティを組み込むか、きめ細かい権限、仮想プライベート・データベース(アプリケーション・コンテキストによるファイングレイン・アクセス・コントロール)、ロール、ストアド・プロシージャ、および監査(ファイングレイン監査を含む)などのデータベースに本来備わっているセキュリティ・メカニズムを使用できます。

セキュリティがアプリケーション内ではなくデータベース自体で規定されている場合は、セキュリティを回避することはできません。アプリケーション・ベースのセキュリティの主なデメリットは、ユーザーがアプリケーションを回避してデータにアクセスすると、セキュリティが回避されることです。たとえば、データベースへのSQL*Plusアクセス権を持つユーザーは、人事管理アプリケーションを介さずに問合せを実行できます。したがって、このユーザーは、アプリケーション内のセキュリティ対策のすべてをバイパスします。

One Big Application Userモデルを使用するアプリケーションでは、データベースのセキュリティ・メカニズムを使用せずに、アプリケーション内にセキュリティ規定を作成する必要があります。ユーザーを認識するのは、データベースではなくアプリケーションであるため、アプリケーション自体が各ユーザーに対してセキュリティ対策を規定する必要があります。

このアプローチでは、データにアクセスする各アプリケーションでのセキュリティの再実装が必要になります。組織では複数のアプリケーションに同じセキュリティ・ポリシーを実装する必要があるため、セキュリティが高コストになり、新規アプリケーションごとに高コストな再実装が必要になります。

関連トピック

- セキュリティに関する潜在的な問題となる非定型ツールの使用

親トピック: アプリケーション・ベースのセキュリティの使用に関する考慮事項

12.3 アプリケーション設計におけるパスワードの保護

Oracleでは、パスワード・サービスの安全な呼出し(スクリプトを使用するなど)のための戦略や、これらの戦略をその他の機密データに適用するための戦略を用意しています。

- アプリケーションでのパスワードの保護に関する一般的なガイドライン
  アプリケーションでのパスワードの保護に関するガイドラインでは、プラットフォーム固有のセキュリティ脅威などの領域を扱います。
- 外部パスワード・ストアを使用したパスワードの保護
  データベースに接続するためのパスワード資格証明は、クライアント側のOracleウォレットを使用して格納できます。
- ORAPWDユーティリティを使用したパスワードの保護
  SYSDBAまたはSYSOPERユーザーはパスワード・ファイルを使用して、ネットワーク経由でアプリケーションに接続できます。
- 例: パスワードを読み取るためのJavaコード
  パスワードの読取りに使用できるJavaパッケージを作成できます。

親トピック: アプリケーション開発者のセキュリティの管理

12.3.1 アプリケーションでのパスワードの保護に関する一般的なガイドライン

アプリケーションでのパスワードの保護に関するガイドラインでは、プラットフォーム固有のセキュリティ脅威などの領域を扱います。

- プラットフォーム固有のセキュリティへの脅威
  次の潜在的なセキュリティへの脅威は、わかりにくい可能性があるので注意してください。
- パスワード入力を処理するアプリケーションの設計のガイドライン
  Oracleには、パスワード入力を扱うアプリケーションの設計に関するガイドラインが用意されています。
- パスワードの形式と動作の構成のガイドライン
  Oracle Databaseには、パスワードの形式と動作の構成に関するガイドラインが用意されています。
- SQLスクリプトにおけるパスワードの処理のガイドライン
  Oracleには、SQLスクリプトにおけるパスワードの取扱いに関するガイドラインが用意されています。

親トピック: アプリケーション設計におけるパスワードの保護

12.3.1.1 プラットフォーム固有のセキュリティへの脅威

次の潜在的なセキュリティへの脅威は、わかりにくい可能性があるので注意してください。

これらのセキュリティへの脅威は次のとおりです。

- UNIXおよびLinuxプラットフォームでは、同じホスト・コンピュータ上のすべてのオペレーティング・システム・ユーザーがコマンド・パラメータを表示できます。結果として、コマンドラインに入力したパスワードは他のユーザーに公開される可能性があります。ただし、UNIXおよびLinux以外のプラットフォームがこの脅威を回避できるとは考えないでください。
- HP Tru64およびIBM AIXなど一部のUNIXプラットフォームでは、すべてのオペレーティング・システム・ユーザーがあらゆるプロセスの環境変数を表示できます。ただし、UNIXおよびLinux以外のプラットフォームがこの脅威を回避できるとは考えないでください。
- Microsoft Windowsでは、コマンド・リコール機能(上矢印)に、コマンド起動間のユーザー入力が記憶されます。たとえば、SQL*PlusのCONNECT SYSTEM/password表記法を使用し、終了してから上矢印を押してCONNECTコ

マンドを繰り返した場合、コマンド・リコール機能により接続文字列が明らかになり、パスワードが表示されます。また、Microsoft Windows以外のプラットフォームがこの脅威を回避できるとは考えないでください。

親トピック: アプリケーションでのパスワードの保護に関する一般的なガイドライン

12.3.1.2 パスワード入力を処理するアプリケーションの設計のガイドライン

Oracleには、パスワード入力を扱うアプリケーションの設計に関するガイドラインが用意されています。

- パスワードの入力を対話形式で求めるアプリケーションを設計します。コマンドライン・ユーティリティの場合、コマンド・プロンプトでパスワードを公開することをユーザーに強制しないでください。

  アプリケーションの設計に使用するプログラミング言語のAPIについて、ユーザーからのパスワードを処理する最適な方法を確認します。この機能を処理するJavaコードの例は、「例: パスワードを読み取るためのJavaコード」を参照してください。

- コード・インジェクション攻撃からデータベースを保護します。コード・インジェクション攻撃の標的は主に、データベースにSQLを送信するクライアント・アプリケーション・ツール(SQL*Plus、Oracle Call Interface (OCI)、JDBCアプリケーションなど)です。これらのツールを使用して作成されたデータベース・ドライバも含まれます。SQLインジェクション攻撃では、PL/SQLアプリケーションで想定されていない方法でSQL文が動作します。インジェクション攻撃は、データベースに文が送信される前に行われます。たとえば、侵入者はWHERE句をTRUEに設定してパスワード認証を回避できます。

  SQLインジェクション攻撃の問題に対処するには、バインド変数引数を使用するか妥当性チェックを作成します。バインド変数を使用できない場合は、DBMS_ASSERT PL/SQLパッケージを使用して入力値のプロパティを検証することを検討してください。DBMS_ASSERTパッケージの詳細は、『Oracle Database PL/SQLパッケージおよびタイプ・リファレンス』を参照してください。また、PUBLICなどのロールに対する権限付与も検討してください。

  インジェクションはデータベースに文が送信される前に行われる可能性があるため、クライアント・アプリケーションのユーザーがSQLインジェクションをPL/SQLに結び付けることができるとは限らないことに注意してください。

  SQLインジェクションの防止の詳細は、『Oracle Database PL/SQL言語リファレンス』を参照してください。

- 可能な場合は、認証を遅延するアプリケーションを設計します。たとえば:
  - ログインに証明書を使用します。
  - オペレーティング・システムで提供される機能を使用してユーザーを認証します。たとえば、Microsoft Windows上で動作するアプリケーションはドメイン認証を使用できます。
- パスワードをマスクまたは暗号化します。パスワードを格納する必要がある場合、パスワードをマスクまたは暗号化します。たとえば、ログ・ファイルでパスワードをマスクし、リカバリ・ファイルでパスワードを暗号化できます。
- 各接続を認証します。たとえば、スキーマAがデータベース1に存在する場合、データベース2のスキーマAが同一のユーザーであるとは考えないでください。同様に、ローカル・オペレーティング・システム・ユーザーpsmithは、必ずしもリモート・ユーザーpsmithと同じユーザーではありません。
- ファイルまたはリポジトリにクリアテキスト・パスワードを格納しないでください。パスワードをファイルに格納すると、侵入者がパスワードにアクセスする危険性が高まります。
- 1つのマスター・パスワードを使用します。たとえば:
  - 1人のデータベース・ユーザーに、他のデータベース・ユーザーとなるためのプロキシ認証を付与できます。この場合、必要となるデータベース・パスワードは1つのみです。詳細は、「プロキシ・ユーザー・アカウントと、そのアカウ

ントを介して接続するユーザーの認可」を参照してください。

- マスター・パスワードでオープンできるパスワード・ウォレットを作成できます。ウォレットにはその他のパスワードが含められます。Wallet Managerの詳細は、『Oracle Databaseエンタープライズ・ユーザー・セキュリティ管理者ガイド』を参照してください。

親トピック: アプリケーションでのパスワードの保護に関する一般的なガイドライン

12.3.1.3 パスワードの形式と動作の構成のガイドライン

Oracle Databaseには、パスワードの形式と動作の構成に関するガイドラインが用意されています。

- パスワードの存続期間を制限します。パスワード存続期間を設定して、その期間が過ぎるとパスワードが期限切れになり、変更しないとユーザーがアカウントにログインできなくなるようにすることができます。パスワードの存続期間を制御するために使用できるパラメータは、パスワード・エイジングおよび期限切れの制御についてを参照してください。
- ユーザーが古いパスワードを再利用する機能を制限します。詳細は、ユーザーによる以前のパスワードの再利用の制御を参照してください。
- 強力かつ安全なパスワードを作成するようユーザーに要求します。強力なパスワードの作成に関するガイドラインは、パスワードの保護に関するガイドラインを参照してください。パスワードの複雑度検証についてでは、パスワードの要件をカスタマイズする方法について説明しています。
- パスワードで大/小文字を区別できるようにします。詳細は、パスワードでの大/小文字の区別の管理を参照してください。

関連トピック

- パスワードの最低要件
- パスワードの複雑度検証について

親トピック: アプリケーションでのパスワードの保護に関する一般的なガイドライン

12.3.1.4 SQLスクリプトにおけるパスワードの処理のガイドライン

Oracleには、SQLスクリプトにおけるパスワードの取扱いに関するガイドラインが用意されています。

- プログラムまたはスクリプトのコマンドラインにパスワードを指定してSQL*Plusを起動しないでください。パスワードが必須であるにもかかわらず省略した場合、SQL*Plusではパスワードの入力を求めるプロンプトが表示され、パスワードが表示されないようにエコー機能が自動的に無効となります。

  次の各例は、パスワードがコマンドライン上で公開されないため安全です。また、Oracle Databaseではこれらのパスワードがネットワークを介して自動的に暗号化されます。

```
$ sqlplus system
Enter password: password
SQL> CONNECT SYSTEM
Enter password: password
```

  次の例では、パスワードが他のオペレーティング・システム・ユーザーに公開されます。

```
sqlplus system/password
```

  次の例は、2つのセキュリティ上のリスクをもたらします。1つ目の例では、覗き込んでいる可能性のある他のユーザーにパスワードを公開してしまいます。2つ目の例では、Microsoft Windowsなどの一部のプラットフォームにおいて、パスワードがコマンドラインのリコール攻撃を受けやすくなります。

```
$ sqlplus /nolog
SQL> CONNECT SYSTEM/password
```

- たとえばパスワードまたは秘密キーを必要とするSQLスクリプトの場合、アカウントを作成したりあるアカウントでログインするには、置換変数&1、&2などの位置パラメータを使用しないでください。かわりに、ユーザーに値の入力を求めるプロンプトを表示するスクリプトを設計します。また、スクリプトから、またはスプール・モードを使用している場合に出力を表示するエコー機能を無効にする必要があります。エコー機能を無効にするには、次の設定を使用します。

```
SET ECHO OFF
```

スクリプトにより値の目的を明白にする必要があります。たとえば、値によりアカウントまたは証明書などの新しい値が設定されるかどうか、または既存のアカウントへのログインなど、値が認証されるかどうかを明白にする必要があります。

次の例は、セキュリティ上のリスクをもたらす方法でユーザーがスクリプトを起動することが回避されるため安全です。パスワードはエコーされず、スプール・ファイルに記録されません。

```
SET VERIFY OFF
 ACCEPT user CHAR PROMPT ‘Enter user to connect to: ‘
 ACCEPT password CHAR PROMPT ‘Enter the password for that user: ' HIDE
 CONNECT &user/&password
```

この例では、次のようになります。

- SET VERIFY OFFは、パスワードの表示を防止します。(SET VERIFYは、置換の前後にスクリプトの各行をリストします。)SET VERIFY OFFコマンドをHIDEコマンドと結合する方法は、パスワードおよびその他の機密入力データを非表示にする便利なテクニックです。
- ACCEPT password CHAR PROMPTは、ACCEPT passwordプロンプトにHIDEオプションを含めます。これによって入力パスワードのエコーが回避されます。

次の例では位置パラメータを使用しており、ユーザーがコマンドライン上でパスワードを渡すことでスクリプトを起動できるため、セキュリティ上のリスクをもたらします。ユーザーがパスワードを入力せず、入力を求めるプロンプトが表示される場合、ユーザーが入力した内容がすべて画面およびスプール・ファイル(スプールが有効な場合)にエコーされるため危険です。

```
CONNECT &1/&2
```

- バッチ・スクリプトのログイン時間を制御します。パスワードを必要とするバッチ・スクリプトでは、実行されることになっている時間内のみにバッチ・スクリプトがログインできるようにアカウントを構成します。たとえば、毎日午後8時から1時間実行するバッチ・スクリプトを想定します。この時間のみスクリプトがログインできるようにアカウントを設定します。侵入者がアクセスしようとした場合、安全性の低いアカウントが利用される可能性は低くなります。
- パスワードの入力を求めるDML文またはDDL SQL文を使用する場合は注意してください。この場合、機密情報がネットワークを介してクリアテキストで渡されます。Oracle厳密認証を使用して、この問題に対処できます。

次のパスワード変更例は、パスワードが公開されないため安全です。

```
password psmith
Changing password for psmith
New password: password
Retype new password: password
```

この例は、パスワードがコマンドラインおよびネットワーク上の両方に公開されるため、セキュリティ上のリスクをもたらします。

```
ALTER USER psmith IDENTIFIED BY password
```

親トピック: アプリケーションでのパスワードの保護に関する一般的なガイドライン

12.3.2 外部パスワード・ストアを使用したパスワードの保護

データベースに接続するためのパスワード資格証明は、クライアント側のOracleウォレットを使用して格納できます。

Oracleウォレットは、ユーザーのログインに必要な認証および署名用証明書を格納する安全性の高いソフトウェア・コンテナです。

関連項目:

- 安全性の高い外部パスワード・ストアの詳細は、パスワード資格証明用の安全性の高い外部パスワード・ストアの管理を参照してください
- Oracle Wallet Managerを使用したOracleウォレットの構成の詳細は、『Oracle Databaseエンタープライズ・ユーザー・セキュリティ管理者ガイド』を参照してください。

親トピック: アプリケーション設計におけるパスワードの保護

12.3.3 ORAPWDユーティリティを使用したパスワードの保護

SYSDBAまたはSYSOPERユーザーはパスワード・ファイルを使用して、ネットワーク経由でアプリケーションに接続できます。

- パスワード・ファイルを作成するには、ORAPWDユーティリティを使用します。

関連項目:

パスワード・ファイルの作成および管理方法の詳細は、『Oracle Database管理者ガイド』を参照してください。

親トピック: アプリケーション設計におけるパスワードの保護

12.3.4 例: パスワードを読み取るためのJavaコード

パスワードの読取りに使用できるJavaパッケージを作成できます。

例12-1に、パスワードの読取りに使用できるJavaパッケージの作成方法を示します。

例12-1 パスワードを読み取るためのJavaコード

```
// Change the following line to a name for your version of this package
package passwords.sysman.emSDK.util.signing;
import java.io.IOException;
import java.io.PrintStream;
import java.io.PushbackInputStream;
import java.util.Arrays;

/**
 * The static readPassword method in this class issues a password prompt
 * on the console output and returns the char array password
 * entered by the user on the console input.
 */
public final class ReadPassword {
  //---------------------------------
  /**
   * Test driver for readPassword method.
   * @param args the command line args
   */
  public static void main(String[] args) {
```

```
    char[] pass = ReadPassword.readPassword("Enter password: ");
    System.out.println("The password just entered is ¥""
      + new String(pass) + "¥"");
    System.out.println("The password length is " + pass.length);
  }
   * Issues a password prompt on the console output and returns
   * the char array password entered by the user on the console input.
   * The password is not displayed on the console (chars are not echoed).
   * As soon as the returned char array is not needed,
   * it should be erased for security reasons (Arrays.fill(charArr, ' '));
   * A password should never be stored as a java String.
   *
   * Note that Java 6 has a Console class with a readPassword method,
   * but there is no equivalent in Java 5 or Java 1.4.
   * The readPassword method here is based on Sun's suggestions at
   * http://java.sun.com/developer/technicalArticles/Security/pwordmask.
   *
   * @param prompt the password prompt to issue
   * @return new char array containing the password
   * @throws RuntimeException if some error occurs
   */
  public static final char[] readPassword(String prompt)
  throws RuntimeException {
    try {
      StreamMasker masker = new StreamMasker(System.out, prompt);
      Thread threadMasking = new Thread(masker);
      int firstByte = -1;
      PushbackInputStream inStream = null;
      try {
        threadMasking.start();
        inStream = new PushbackInputStream(System.in);
        firstByte = inStream.read();
      } finally {
        masker.stopMasking();
      }
      try {
        threadMasking.join();
      } catch (InterruptedException e) {
        throw new RuntimeException("Interrupt occurred when reading password");
      }
      if (firstByte == -1) {
        throw new RuntimeException("Console input ended unexpectedly");
      }
      if (System.out.checkError()) {
        throw new RuntimeException("Console password prompt output error");
      }
      inStream.unread(firstByte);
      return readLineSecure(inStream);
    }
    catch (IOException e) {
      throw new RuntimeException("I/O error occurred when reading password");
    }
  }
  //----------------------------------
  /**
   * Reads one line from an input stream into a char array in a secure way
   * suitable for reading a password.
   * The char array will never contain a '¥n' or '¥r'.
   *
   * @param inStream the pushback input stream
   * @return line as a char array, not including end-of-line-chars;
   *  never null, but may be zero length array
   * @throws RuntimeException if some error occurs
   */
  private static final char[] readLineSecure(PushbackInputStream inStream)
  throws RuntimeException {
    if (inStream == null) {
```

```
      throw new RuntimeException("readLineSecure inStream is null");
    }
    try {
      char[] buffer = null;
      try {
        buffer = new char[128];
        int offset = 0;
        // EOL is '¥n' (unix), '¥r¥n' (windows), '¥r' (mac)
        loop:
        while (true) {
          int c = inStream.read();
          switch (c) {
          case -1:
          case '¥n':
            break loop;
          case '¥r':
            int c2 = inStream.read();
            if ((c2 != '¥n') && (c2 != -1))
              inStream.unread(c2);
            break loop;
          default:
            buffer = checkBuffer(buffer, offset);
            buffer[offset++] = (char) c;
            break;
          }
        }
        char[] result = new char[offset];
        System.arraycopy(buffer, 0, result, 0, offset);
        return result;
      }
      finally {
        if (buffer != null)
          Arrays.fill(buffer, ' ');
      }
    }
    catch (IOException e) {
      throw new RuntimeException("I/O error occurred when reading password");
    }
  }
  //----------------------------------
  /**
   * This is a helper method for readLineSecure.
   *
   * @param buffer the current char buffer
   * @param offset the current position in the buffer
   * @return the current buffer if it is not yet full;
   *  otherwise return a larger buffer initialized with a copy
   *  of the current buffer and then erase the current buffer
   * @throws RuntimeException if some error occurs
   */
  private static final char[] checkBuffer(char[] buffer, int offset)
  throws RuntimeException
  {
    if (buffer == null)
      throw new RuntimeException("checkBuffer buffer is null");
    if (offset < 0)
      throw new RuntimeException("checkBuffer offset is negative");
    if (offset < buffer.length)
      return buffer;
    else {
      try {
        char[] bufferNew = new char[offset + 128];
        System.arraycopy(buffer, 0, bufferNew, 0, buffer.length);
        return bufferNew;
      } finally {
        Arrays.fill(buffer, ' ');
      }
```

```
  }
}
//---------------------------------
/**
 * This private class prints a one line prompt
 * and erases reply chars echoed to the console.
 */
private static final class StreamMasker
extends Thread {
  private static final String BLANKS = StreamMasker.repeatChars(' ', 10);
  private String m_promptOverwrite;
  private String m_setCursorToStart;
  private PrintStream m_out;
  private volatile boolean m_doMasking;
  //---------------------------------
  /**
   * Constructor.
   * @throws RuntimeException if some error occurs
   */
  public StreamMasker(PrintStream outPrint, String prompt)
  throws RuntimeException {
    if (outPrint == null)
      throw new RuntimeException("StreamMasker outPrint is null");
    if (prompt == null)
      throw new RuntimeException("StreamMasker prompt is null");
    if (prompt.indexOf('¥r') != -1)
      throw new RuntimeException("StreamMasker prompt contains a CR");
    if (prompt.indexOf('¥n') != -1)
      throw new RuntimeException("StreamMasker prompt contains a NL");
    m_out = outPrint;
    m_setCursorToStart = StreamMasker.repeatChars('¥010',
      prompt.length() + BLANKS.length());
    m_promptOverwrite = m_setCursorToStart + prompt + BLANKS
      + m_setCursorToStart + prompt;
  }
  //---------------------------------
  /**
   * Begin masking until asked to stop.
   * @throws RuntimeException if some error occurs
   */
  public void run()
  throws RuntimeException {
    int priorityOriginal = Thread.currentThread().getPriority();
    Thread.currentThread().setPriority(Thread.MAX_PRIORITY);
    try {
      m_doMasking = true;
      while (m_doMasking) {
        m_out.print(m_promptOverwrite);
        if (m_out.checkError())
          throw new RuntimeException("Console output error writing prompt");
        try {
          Thread.currentThread().sleep(1);
        } catch (InterruptedException ie) {
          Thread.currentThread().interrupt();
          return;
        }
      }
      m_out.print(m_setCursorToStart);
    } finally {
      Thread.currentThread().setPriority(priorityOriginal);
    }
  }
  //---------------------------------
  /**
   * Instructs the thread to stop masking.
   */
  public void stopMasking() {
```

```
      m_doMasking = false;
    }
    //---------------------------------
    /**
     * Returns a repeated char string.
     *
     * @param c the char to repeat
     * @param length the number of times to repeat the char
     * @throws RuntimeException if some error occurs
     */
    private static String repeatChars(char c, int length)
    throws RuntimeException {
      if (length < 0)
        throw new RuntimeException("repeatChars length is negative");
      StringBuffer sb = new StringBuffer(length);
      for (int i = 0; i < length; i++)
        sb.append(c);
      return sb.toString();
    }
  }
}
```

親トピック: アプリケーション設計におけるパスワードの保護

12.4 外部プロシージャの保護

外部プロシージャはデータベースとは別に.dllまたは.soファイルに保存され、資格証明認証で保護できます。

- 外部プロシージャの保護について
  安全上の理由のため、Oracle外部プロシージャは、データベースと物理的に異なるプロセスで実行されます。
- 資格証明の認証に対するextprocの構成に関する一般プロセス
  セキュリティを強化するために、extprocプロセスを構成して、資格証明を介して認証できます。
- extprocプロセス認証および偽装の予期される動作
  extprocプロセスには、認証および偽装に対する一連の動作が含まれています。
- 外部プロシージャの認証の構成
  extprocプロセスの資格証明を構成するには、DBMS_CREDENTIAL PL/SQLパッケージを使用します。
- レガシー・アプリケーションの外部プロシージャ
  セキュリティを最大にするために、ENFORCE_CREDENTIAL環境変数をTRUEに設定します。

親トピック: アプリケーション開発者のセキュリティの管理

12.4.1 外部プロシージャの保護について

安全上の理由のため、Oracle外部プロシージャは、データベースと物理的に異なるプロセスで実行されます。

ほとんどの場合、このプロセスを構成して、Oracleソフトウェア・アカウント以外のユーザーとして実行します。アプリケーションがこの外部プロシージャを呼び出す場合(.dllまたは.soファイルのライブラリにアクセスする必要がある場合など)、Oracle Databaseがextprocと呼ばれるオペレーティング・システム・プロセスを作成します。デフォルトでは、extprocプロセスは、サーバー・プロセスで直接通信します。つまり、資格証明を使用しない場合、Oracle Databaseは、デフォルトのOracle Databaseサーバー構成でextprocプロセスを作成し、oracleソフトウェア・アカウントとしてextprocを実行します。また、Oracle Databaseリスナーを介して通信できます。

関連項目:

外部プロシージャの保護に関するガイドライン

親トピック: 外部プロシージャの保護

12.4.2 資格証明の認証に対するextprocの構成に関する一般プロセス

セキュリティを強化するために、extprocプロセスを構成して、資格証明を介して認証できます。

一般プロセスは次のとおりです。

1. 資格証明を作成します。

   資格証明は、暗号化されたコンテナにあります。パブリック・シノニムとプライベート・シノニムの両方でこの資格証明を参照できます。「外部プロシージャの認証の構成」では、この資格証明を作成し、データベースを構成して使用する方法について説明します。

2. 専用サーバーまたは共有サーバー・プロセスで実行しているデータベースへの最初の接続を行います。

3. アプリケーションは、外部プロシージャへのコールを行います。

   これが最初のコールの場合、Oracle Databaseはextprocプロセスを作成します。extprocの資格証明を使用する場合、Oracleリスナーを使用してextprocプロセスを起動できません。

4. extprocプロセスは、偽装(つまり、指定された資格証明のかわりに実行)して、必要な.dll、.so、.slまたは.aファイルをロードし、SQLおよびC間のデータを送信します。

親トピック: 外部プロシージャの保護

12.4.3 extprocプロセス認証および偽装の予期される動作

extprocプロセスには、認証および偽装に対する一連の動作が含まれています。

表12-2では、可能性のある認証および偽装シナリオに基づくextprocプロセスの予期される動作について説明します。

この表では、資格証明が明示的に指定されず、ENFORCE_CREDENTIAL環境変数がTRUEに設定されている場合、GLOBAL_EXTPROC_CREDENTIALは、デフォルトの資格証明の予約された資格証明名です。そのため、ENFORCE_CREDENTIALがTRUEに設定されている場合、この名前で資格証明を作成することをお薦めします。

表12-2 extprocプロセス認証および偽装設定の予期される動作

ENFORCE_CREDENTIAL環境変数設定	資格証明を使用したPL/SQLライブラリか	GLOBAL_EXTPROC_CREDENTIAL資格証明の有無	予期される動作
FALSE	いいえ	いいえ	Oracle リスナーまたは Oracle サーバー・プロセスの所有者のオペレーティング・システム権限で認証される 12c より前のリリースの認証を使用します。
FALSE	いいえ	はい	Oracle Database インスタンス全体の指定された

ENFORCE_CREDENTIAL環境変数設定	資格証明を使用したPL/SQLライブラリか	GLOBAL_EXTPROC_CREDENTIAL資格証明の有無	予期される動作
			GLOBAL_EXTPROC_CREDENTIAL を使用して認証および偽装します。 GLOBAL_EXTPROC_CREDENTIAL 資格証明のみを使用している場合、このグローバル資格証明の EXECUTE 権限が暗黙的にすべてのユーザーに自動的に付与されます。
FALSE	はい	いいえ	PL/SQL ライブラリで定義された資格証明を使用して認証および偽装します
FALSE	はい	はい	認証および偽装します。 PL/SQL ライブラリおよび GLOBAL_EXTPROC_CREDENTIAL 設定の両方で資格証明が定義された場合、PL/SQL ライブラリの資格証明が優先されます。
TRUE	いいえ	いいえ	エラーORA-28575: 外部プロシージャ・エージェントへの RPC 接続をオープンできませんを戻します
TRUE	いいえ	はい	Oracle システム全体の指定された GLOBAL_EXTPROC_CREDENTIAL を使用して認証および偽装します(脚注 1)
TRUE	はい	いいえ	PL/SQL ライブラリで定義された資格証明を使用して認証および偽装します
TRUE	はい	はい	認証および偽装します(脚注 2)

親トピック: 外部プロシージャの保護

12.4.4 外部プロシージャの認証の構成

extprocプロセスの資格証明を構成するには、DBMS_CREDENTIAL PL/SQLパッケージを使用します。

1. CREATE CREDENTIALまたはCREATE ANY CREDENTIAL権限が付与されたユーザーを使用してSQL*Plusにログインします。

```
sqlplus psmith
Enter password: password
Connected.
```

マルチテナント環境で、適切なプラガブル・データベース(PDB)に接続する必要があります。たとえば:

```
sqlplus psmith@hpdb
Enter password: password
Connected.
```

また、CREATE LIBRARYまたはCREATE ANY LIBRARY権限および外部コールを含むライブラリのEXECUTEオブジェクト権限があることも確認します。

2. DBMS_CREDENTIAL PL/SQLパッケージを使用して、新しい資格証明を作成します。

たとえば:

```
BEGIN
  DBMS_CREDENTIAL.CREATE_CREDENTIAL (
    credential_name   => 'smith_credential',
    user_name         => 'tjones',
    password          => 'password')
END;
/
```

この例では、次のようになります。

- credential_name: 資格証明の名前を入力します。オプションで、スキーマの名前(たとえば、psmith.smith_credentialなど)で接頭辞を付けます。ENFORCE_CREDENTIAL環境変数がTRUEに設定されている場合、credential_name GLOBAL_EXTPROC_CREDENTIALを使用して資格証明を作成する必要があります。
- user_name: ユーザーとして実行するために使用する有効なオペレーティング・システム・ユーザー名を入力します。
- password: user_nameユーザーに対するパスワードを入力します。

3. 資格証明とPL/SQLライブラリを関連付けます。

たとえば:

```
CREATE OR REPLACE LIBRARY ps_lib
 AS 'smith_lib.so' IN DLL_LOC
 CREDENTIAL smith_credential;
```

この例では、DLL_LOCは、$ORACLE_HOME/binディレクトリを指すディレクトリ・オブジェクトです。DLLへの絶対パスの使用はお薦めしません。

PL/SQLライブラリがextprocプロセスを介して外部プロシージャ・コールによってロードされる場合、extprocでは、定義されたsmith_credential資格証明のかわりに認証および偽装できます。

4. 外部プロシージャのコール方法や渡す引数をPL/SQLに対して通知するPL/SQLプロシージャまたは関数を作成して、

外部プロシージャを登録します。

たとえば、Cで記述された外部プロシージャを登録する関数を作成するには、CREATE FUNCTION文のAS LANGUAGE C、LIBRARYおよびNAME句のみ次のように使用します。

```
CREATE OR REPLACE FUNCTION getInt (x VARCHAR2, y BINARY_INTEGER)
RETURN BINARY_INTEGER
AS LANGUAGE C
LIBRARY ps_lib
NAME "get_int_vals"
PARAMETERS (x STRING, y int);
```

関連項目:

- 外部プロシージャの保護に関するガイドライン
- DBMS_CREDENTIALパッケージの詳細は、『Oracle Database PL/SQLパッケージおよびタイプ・リファレンス』を参照してください。
- extprocエージェントの詳細は、『Oracle Call Interfaceプログラマーズ・ガイド』を参照してください。
- extproc.oraファイルの詳細は、『Oracle Database Net Services管理者ガイド』を参照してください。

親トピック: 外部プロシージャの保護

12.4.5 レガシー・アプリケーションの外部プロシージャ

セキュリティを最大にするために、ENFORCE_CREDENTIAL環境変数をTRUEに設定します。

ただし、下位互換性に対応する必要がある場合、ENFORCE_CREDENTIALをFALSEに設定します。FALSEによって、extprocプロセスは、指定された資格証明のかわりにユーザー定義コールアウト関数を認証、偽装および実行できます。

- 資格証明がPL/SQLライブラリで定義されます。
- 資格証明は定義されていませんが、 GLOBAL_EXTPROC_CREDENTIAL資格証明が存在します。

これらの資格証明定義が設定されていない場合、ENFORCE_CREDENTIALパラメータをFALSEに設定すると、Oracleリスナーまたは Oracleサーバー・プロセスの所有者のオペレーティング・システム権限で認証されるextprocプロセスが設定されます。

extprocプロセス上で実行されるレガシー・アプリケーションでは、レガシー・アプリケーション・コードを変更して、すべての別名ライブラリと資格証明を関連付けることをお薦めします。これを実行できない場合、Oracle Databaseは GLOBAL_EXTPROC_CREDENTIAL資格証明を使用して、認証の処理方法を決定します。GLOBAL_EXTPROC_CREDENTIAL資格証明が定義されていない場合、extprocプロセスは、Oracleリスナーまたは Oracleサーバー・プロセスの所有者のオペレーティング・システム権限で認証されます。

親トピック: 外部プロシージャの保護

12.5 LOBロケータの署名を使用したLOBの保護

LOBロケータの署名を再生成することで、ラージ・オブジェクト(LOB)を保護できます。

- LOBロケータの署名を使用したLOBの保護について
  LOBロケータ(ラージ・オブジェクト(LOB)値の実際の場所へのポインタ)には署名を割り当てることができます。この署

名はLOBの保護に使用できます。

- LOBロケータの署名キーの暗号化の管理
  ALTER DATABASE DICTIONARY SQL文を使用して、LOBロケータの署名キーを暗号化できます。

親トピック: アプリケーション開発者のセキュリティの管理

12.5.1 LOBロケータの署名を使用したLOBの保護について

LOBロケータ(ラージ・オブジェクト(LOB)値の実際の場所へのポインタ)には署名を割り当てることができます。この署名はLOBの保護に使用できます。

LOBを作成するときに、Oracle DatabaseによってLOBロケータに自動的に署名が割り当てられます。Oracle Databaseは、クライアントからロケータを受信したときに署名が一致することを検証し、ロケータが改ざんされていないことを確認します。署名ベース・セキュリティは、永続LOBロケータと一時LOBロケータの両方に使用できます。これは、索引構成表(IOT)ロケータから取得される分散CLOB、BLOBおよびNBLOBにも使用されます。

Oracle Real Applications Clusters (Oracle RAC)環境では、すべてのインスタンスが同じ署名キーを共有し、そのキーはデータベース内に永続化されます。マルチテナント環境では、各プラガブル・データベース(PDB)が独自の署名キーを持ちます。LOBロケータが改ざんされている場合、署名検証によってLOBが拒否され、「ORA-64219:無効なLOBロケータが見つかりました」エラーが発生します。

スタンドアロン・データベースまたはPDBからクライアントに送信されるLOBロケータのLOB署名の生成に使用されたLOB署名キーを暗号化、キー更新および削除できます。署名キーを暗号化する予定の場合、キーが存在するデータベース(またはPDB)にオープンTDEキーストアが必要です。

LOB署名機能を有効にするには、LOB_SIGNATURE_ENABLE初期化パラメータをTRUEに設定する必要があります。デフォルトでは、LOB_SIGNATURE_ENABLEは、Oracle Databaseリリース19cではFALSEに設定されています。

親トピック: LOBロケータの署名を使用したLOBの保護

12.5.2 LOBロケータの署名キーの暗号化の管理

ALTER DATABASE DICTIONARY SQL文を使用して、LOBロケータの署名キーを暗号化できます。

1. ALTER DATABASE DICTIONARY権限を持つユーザーとして、データベース・インスタンスにログインします。
2. 必要に応じて、LOB_SIGNATURE_ENABLE初期化パラメータをTRUEに設定して、LOB署名キー機能を有効にします。

   ```
   ALTER SYSTEM SET LOB_SIGNATURE_ENABLE = TRUE;
   ```

   または、データベースの再起動の前にinit.ora初期化ファイルのLOB_SIGNATURE_ENABLEパラメータを設定できます。これにより、すべてのPDBのLOB署名キー機能が有効になります。
3. 署名キーを暗号化する予定の場合、データベースまたはPDBにオープンTDEキーストアがあることを確認します。
   TDEキーストアを作成するには、SYSKM管理権限が必要です。
   たとえば、ソフトウェアTDEキーストアを作成して開くには、次のようにします。

   ```
   ADMINISTER KEY MANAGEMENT CREATE KEYSTORE '/etc/ORACLE/WALLETS/orcl' IDENTIFIED
   BY password;
   ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN IDENTIFIED BY password;
   ```

4. ALTER DATABASE DICTIONARY文を実行して、LOB署名キー構成を設定します。
   - 不明瞭化せずにLOBロケータの署名キーを暗号化するには、次の文を実行します。

```
ALTER DATABASE DICTIONARY ENCRYPT CREDENTIALS;
```

- クライアントに送信されるLOBロケータのLOBロケータの署名キーを再生成するには、次の文を使用します。データベースが制限モードの場合、Oracle Databaseでは、再生成された署名キーを暗号化するために新しいLOB署名キーが再生成されます。データベースが非制限モードの場合、新しい署名キーは再生成されませんが、そのかわりにOracle Databaseでは新しい暗号化キーを使用して既存のLOB署名キーを暗号化します。データベース管理者またはPDB管理者は、できればデータベースの停止時に定期的にこの文を制限モードで実行することをお薦めします。

  ```
  ALTER DATABASE DICTIONARY REKEY CREDENTIALS;
  ```

- 暗号化されたLOBロケータの署名キーを削除して、暗号化されたフォームのかわりに、不明瞭化された形式の新しいLOBの署名キーを再生成するには、次の文を実行します。

  ```
  ALTER DATABASE DICTIONARY DELETE CREDENTIALS;
  ```

関連トピック

- 透過的データ暗号化の構成

親トピック: LOBロケータの署名を使用したLOBの保護

12.6 アプリケーション権限の管理

ほとんどのデータベース・アプリケーションでは、異なるスキーマ・オブジェクトごとに異なる権限が関与します。

各アプリケーションに必要な権限の追跡は、複雑な場合があります。また、アプリケーションを実行するユーザーの認可には、多くのGRANT操作が関与する場合があります。

- アプリケーションの権限管理を簡素化するために、アプリケーションごとに作成した1つのロールに、1人のユーザーがそのアプリケーションを実行するために必要なすべての権限を付与します。

  実際には、1つのアプリケーションに複数のロールがある可能性があり、各ロールには、アプリケーションの実行中に使用できる機能の多少を決める権限の特定サブセットが付与されます。

たとえば、すべての管理アシスタントが休暇アプリケーションを使用して、部門のメンバーが取得した休暇を記録するとします。このアプリケーションを効率的に管理するには、次の操作手順が必要です。

1. VACATIONロールを作成します。
2. 休暇アプリケーションに必要なすべての権限をVACATIONロールに付与します。
3. VACATIONロールをすべての管理アシスタントに付与します。より効率的な方法は、管理アシスタントが持つ権限を定義したロールを作成し、VACATIONロールをそのロールに付与します。

関連項目:

- ロールの作成、使用可能化と使用禁止化、および権限の付与と取消しの詳細は、「権限とロール認可の構成」を参照してください。
- ROLE_TAB_PRIVS、ROLE_SYS_PRIVSおよびDBA_ROLE_PRIVSデータ・ディクショナリ・ビューのセキュリティ使用の詳細は、ユーザー権限およびロールのデータ・ディクショナリ・ビューを参照してください

親トピック: アプリケーション開発者のセキュリティの管理

12.7 アプリケーション権限の管理にロールを使用する利点

複数のアプリケーション権限を1つのロールにグループ化すると、権限の管理に役立ちます。

次の管理オプションを考えてみます。

- アプリケーションを実行するユーザーに、多数の個別の権限ではなく、ロールを付与できます。したがって、従業員が業務を変更するときは、多くの権限ではなく、1つのロールのみを付与または取り消す必要があります。
- アプリケーションに対応付けられている権限の変更は、そのアプリケーションのすべてのユーザーが保持する権限ではなく、ロールに付与されている権限のみを修正することで実行できます。
- 特定のアプリケーションの実行に必要な権限は、ROLE_TAB_PRIVSとROLE_SYS_PRIVSの各データ・ディクショナリ・ビューを問い合せることで判断できます。
- どのユーザーに、どのアプリケーションの権限があるかは、DBA_ROLE_PRIVSデータ・ディクショナリ・ビューを問い合せることで判断できます。

親トピック: アプリケーション開発者のセキュリティの管理

12.8 アプリケーションへのアクセスを制御するセキュア・アプリケーション・ロールの作成

セキュア・アプリケーション・ロールは関連するPL/SQLパッケージまたはプロシージャでのみ使用可能にできます。

- ステップ1: セキュア・アプリケーション・ロールの作成
  IDENTIFIED USING句を持つCREATE ROLE文で、セキュア・アプリケーション・ロールを作成します。
- ステップ2: アプリケーションに対するアクセス・ポリシーを定義するPL/SQLパッケージの作成
  アプリケーションに対するアクセス・ポリシーを定義するPL/SQLパッケージを作成できます。

親トピック: アプリケーション開発者のセキュリティの管理

12.8.1 ステップ1: セキュア・アプリケーション・ロールの作成

IDENTIFIED USING句を持つCREATE ROLE文で、セキュア・アプリケーション・ロールを作成します。

この文を実行するには、CREATE ROLEシステム権限が必要です。

たとえば、sec_mgr.hr_adminパッケージに対応付けられるhr_adminというセキュア・アプリケーション・ロールを作成するには、次の手順を実行します。

1. 次のようにセキュア・アプリケーション・ロールを作成します。

   ```
   CREATE ROLE hr_admin IDENTIFIED USING sec_mgr.hr_admin_role_check;
   ```

   この例から次のようなことがわかります。

   - 作成されるhr_adminは、セキュア・アプリケーション・ロールです。
   - ロールを使用可能にできるのは、PL/SQLプロシージャsec_mgr.hr_admin_role_check内で定義されたモジュールのみです。この時点で、このプロシージャは終了する必要がありません。「ステップ2: アプリケーションに対するアクセス・ポリシーを定義するPL/SQLパッケージの作成」で、パッケージまたはプロシージャを作成する方法を説明しています。

2. セキュア・アプリケーション・ロールに対して、このロールに通常対応付ける権限を付与します。

   たとえば、HR.EMPLOYEES表に対するSELECT、INSERT、UPDATEおよびDELETE権限をhr_adminロールに付与するには、次の文を入力します。

   ```
   GRANT SELECT, INSERT, UPDATE, DELETE ON HR.EMPLOYEES TO hr_admin;
   ```

   このロールをユーザーに直接付与しないでください。ユーザーがセキュリティ・ポリシーを通過した場合は、PL/SQLプロシージャまたはパッケージによってこのロールが自動的に付与されます。

親トピック: アプリケーションへのアクセスを制御するセキュア・アプリケーション・ロールの作成

12.8.2 ステップ2: アプリケーションに対するアクセス・ポリシーを定義するPL/SQLパッケージの作成

アプリケーションに対するアクセス・ポリシーを定義するPL/SQLパッケージを作成できます。

- アプリケーションに対するアクセス・ポリシーを定義するPL/SQLパッケージの作成について
  セキュア・アプリケーション・ロールを有効または無効にするには、PL/SQLパッケージ内にロールのセキュリティ・ポリシーを作成する必要があります。
- アプリケーションに対するアクセス・ポリシーを定義するPL/SQLパッケージまたはプロシージャの作成
  作成するPL/SQLパッケージまたはプロシージャでは、アクセス・ポリシーを定義するために実行者権限を使用する必要があります。
- セキュア・アプリケーション・ロールのテスト
  セキュア・アプリケーション・ロールを付与されたユーザーとして、そのロールで付与される権限を要するアクションを実行します。

親トピック: アプリケーションへのアクセスを制御するセキュア・アプリケーション・ロールの作成

12.8.2.1 アプリケーションに対するアクセス・ポリシーを定義するPL/SQLパッケージの作成について

セキュア・アプリケーション・ロールを有効または無効にするには、PL/SQLパッケージ内にロールのセキュリティ・ポリシーを作成する必要があります。

個別のプロシージャを作成してこれを実行することもできますが、パッケージを使用すると、一連のプロシージャをグループ化できます。これにより、一緒に使用するポリシーのグループを作成して、アプリケーションを保護するための強固なセキュリティ戦略を提示できます。セキュリティ・ポリシーに失敗したユーザー(潜在的な侵入者)については、監査チェックをパッケージに追加して、その失敗を記録できます。通常、このパッケージは、セキュリティ管理者のスキーマに作成します。

このパッケージまたはプロシージャは、次の内容を実行する必要があります。

- 実行者権限を使用してロールを有効にする必要があります。実行者権限を使用してパッケージを作成するには、AUTHIDプロパティをCURRENT_USERに設定する必要があります。定義者権限を使用してパッケージを作成することはできません。

  実行者権限と定義者権限の詳細は、『Oracle Database PL/SQL言語リファレンス』を参照してください。

- ユーザーを検証するためのセキュリティ・チェックを1つ以上組み込む必要があります。ユーザーを検証する方法の1つは、SYS_CONTEXT SQLファンクションを使用することです。SYS_CONTEXTの詳細は、Oracle Database SQL言語リファレンスを参照してください。ユーザーに関するセッション情報を検索するために、アプリケーション・コンテキストでSYS_CONTEXTを使用できます。詳細は、「アプリケーション・コンテキストを使用したユーザー情報の取得」を参照してください。

- ユーザーがセキュリティ・チェックを通過するときにSET ROLE SQL文またはDBMS_SESSION.SET_ROLEプロシージャを発行する必要があります。パッケージは実行者権限を使用して作成するため、SET ROLE SQL文またはDBMS_SESSION.SET_ROLEプロシージャを発行することによりロールを設定する必要があります。(ただし、このタイプのロール有効化でSET ROLE ALL文を使用することはできません。)PL/SQL埋込みSQL構文はSET ROLE文をサポートしませんが、動的SQL(たとえばEXECUTE IMMEDIATEにより)を使用することによりSET ROLEを起動できます。

  EXECUTE IMMEDIATEの詳細は、『Oracle Database PL/SQL言語リファレンス』を参照してください。

このパッケージまたはプロシージャの作成方法が原因で、セキュア・アプリケーション・ロールを使用可能または使用禁止にする際にログイン・トリガーを使用できません。かわりに、ユーザーがセキュリティ・アプリケーション・ロールで付与された権限を使用する前のユーザー・ログイン時に、アプリケーションからパッケージを直接起動します。

親トピック: ステップ2: アプリケーションに対するアクセス・ポリシーを定義するPL/SQLパッケージの作成

12.8.2.2 アプリケーションに対するアクセス・ポリシーを定義するPL/SQLパッケージまたはプロシージャの作成

作成するPL/SQLパッケージまたはプロシージャでは、アクセス・ポリシーを定義するために実行者権限を使用する必要があります。

たとえば、hr_adminロールを使用する全員を、オンサイト(特定の端末を使用)で午前8時から午後5時まで勤務する従業員に限定するとします。システム管理者またはセキュリティ管理者として、アプリケーションに対してアクセス・ポリシーを定義するプロシージャを作成できます。

1. 次のようにプロシージャを作成します。

```
CREATE OR REPLACE PROCEDURE hr_admin_role_check
 AUTHID CURRENT_USER
 AS
 BEGIN
  IF (SYS_CONTEXT ('userenv','ip_address')
    IN ('192.0.2.10' , '192.0.2.11')
     AND
    TO_CHAR (SYSDATE, 'HH24') BETWEEN 8 AND 17)
  THEN
    EXECUTE IMMEDIATE 'SET ROLE hr_admin';
  END IF;
 END;
/
```

   この例では、次のようになります。

   - AUTHID CURRENT_USERは、実行者権限を使用できるように、AUTHIDプロパティをCURRENT_USERに設定します。
   - IF (SYS_CONTEXT ('userenv','ip_address')は、ユーザー・セッション情報を取得するSYS_CONTEXT SQLファンクションを使用してユーザーを検証します。
   - BETWEEN ... TO_CHARは、アクセス権を付与または拒否するテストを作成します。テストはオンサイト(つまり、特定の端末を使用中)で午前8:00から午後5:00までの時間に作業するユーザーについて、アクセスを制限します。ユーザーがこのチェックを通過すると、hr_adminロールが付与されます。
   - THEN... EXECUTEは、ユーザーがこのテストを通過した場合は、EXECUTE IMMEDIATEコマンドを使用してSET ROLE文を発行し、そのユーザーにロールを付与します。

2. ロールが割り当てられたユーザーに対して、hr_admin_role_checkプロシージャのEXECUTE権限を付与します。

   たとえば:

```
GRANT EXECUTE ON hr_admin_role_check TO psmith;
```

親トピック: ステップ2: アプリケーションに対するアクセス・ポリシーを定義するPL/SQLパッケージの作成

12.8.2.3 セキュア・アプリケーション・ロールのテスト

セキュア・アプリケーション・ロールを付与されたユーザーとして、そのロールで付与される権限を要するアクションを実行します。

セキュア・アプリケーション・ロールを付与されたユーザーとしてログインすると、そのロールが有効になります。

1. そのユーザーとしてデータベース・セッションにログインします。

   たとえば:

```
CONNECT PSMITH@hrpdb
Enter password: password
```

2. セキュア・アプリケーション・ロールで付与される権限を要求するアクションを実行します。

   たとえば、そのロールがsec_admin.hr_admin_role_checkプロシージャに対してEXECUTE権限を付与するとします。

```
EXECUTE sec_admin.hr_admin_role_check;
```

親トピック: ステップ2: アプリケーションに対するアクセス・ポリシーを定義するPL/SQLパッケージの作成

12.9 権限とユーザーのデータベース・ロールとの関連付け

ユーザーの権限が、現在のデータベース・ロールに関連した権限のみであることを確認します。

- ユーザーの権限が現在のデータベース・ロールのみである理由
  1人のユーザーが、多数のアプリケーションおよび多数の対応付けられたロールを使用できます。
- ロールを自動的に使用可能または使用禁止にするSET ROLE文の使用
  各アプリケーションの開始時にSET ROLE文を使用して、各アプリケーションに対応付けられているロールを自動的に使用可能にし、他のすべてのロールを使用禁止にします。

親トピック: アプリケーション開発者のセキュリティの管理

12.9.1 ユーザーの権限が現在のデータベース・ロールのみである理由

1人のユーザーが、多数のアプリケーションおよび多数の対応付けられたロールを使用できます。

ただし、このユーザーの権限は、現在のデータベース・ロールに関連した権限のみであることを確認する必要があります。

次の使用例を考えてみます。

- (「受注」というアプリケーションの)ORDERロールには、INVENTORY表に対するUPDATE権限が含まれています。
- (「在庫」というアプリケーションの)INVENTORYロールには、INVENTORY表に対するSELECT権限が含まれています。
- 何人かの受注入力担当には、ORDERロールとINVENTORYロールの両方が付与されています。

このシナリオでは、両方のロールを付与された受注入力担当がINVENTORYアプリケーションの実行時にORDERロールの権限を使用してINVENTORY表を更新できます。問題は、INVENTORY表の更新は、INVENTORYアプリケーションにとって承認さ

れた操作ではないということです。ORDERアプリケーションにとって承認された操作です。この問題を防ぐには、SET ROLE文を次のセクションの説明のとおりに使用します。

親トピック: 権限とユーザーのデータベース・ロールとの関連付け

12.9.2 ロールを自動的に使用可能または使用禁止にするSET ROLE文の使用

各アプリケーションの開始時にSET ROLE文を使用して、各アプリケーションに対応付けられているロールを自動的に使用可能にし、他のすべてのロールを使用禁止にします。

この方法によって、各アプリケーションでは、必要な場合にのみ、ユーザーの特定の権限を動的に使用可能にします。SET ROLE文は、権限の管理を簡素化します。ユーザーがどのような情報にアクセスできるかと、その情報にいつアクセスできるかを制御します。また、このSET ROLE文によって、ユーザーは、明確に定義された権限ドメイン内で操作を続行できます。あるユーザーがロールからのみ権限を取得している場合、そのユーザーはこれらの権限を組み合せて不正な操作を実行することはできません。

関連トピック

- SET ROLEおよびデフォルト・ロールの設定による権限の付与と取消しの機能
- 権限の付与と取消しが有効になるとき

親トピック: 権限とユーザーのデータベース・ロールとの関連付け

12.10 スキーマを使用したデータベース・オブジェクトの保護

スキーマとは、データベース・オブジェクトを含めることができるセキュリティ・ドメインです。ユーザーおよびロールに付与された権限によって、これらのデータベース・オブジェクトへのアクセスが制御されます。

- 一意スキーマでのデータベース・オブジェクトの保護
  ほとんどのスキーマはユーザー名と考えることができます。つまり、ユーザーがデータベースに接続してそのデータベース・オブジェクトへのアクセスを可能にするアカウントです。
- 共有スキーマでのデータベース・オブジェクトの保護
  多くのアプリケーションでは、ユーザーがアクセスする必要があるのはアプリケーション・スキーマのみであるため、データベースで自分自身のアカウントつまりスキーマを必要としません。

親トピック: アプリケーション開発者のセキュリティの管理

12.10.1 一意スキーマでのデータベース・オブジェクトの保護

ほとんどのスキーマはユーザー名と考えることができます。つまり、ユーザーがデータベースに接続してそのデータベース・オブジェクトへのアクセスを可能にするアカウントです。

ただし、一意スキーマではデータベースへの接続は許可されませんが、関連する一連のオブジェクトを格納するために使用されます。この種のスキーマは通常ユーザーとして作成されますが、CREATE SESSIONシステム権限は(明示的にもロールを介しても)付与されません。

- オブジェクトを保護するには、CREATE SCHEMAを使用して、1つのトランザクション内に複数の表とビューを作成する場合は、一意スキーマにCREATE SESSIONおよびRESOURCE権限を一時的に付与します。

たとえば、所定のスキーマが特定のアプリケーションのスキーマ・オブジェクトを所有する場合があります。アプリケーション・ユーザーに権限がある場合、そのユーザーは、一般的なデータベース・ユーザー名を使用してデータベースに接続し、アプリケーションとそれに対応するオブジェクトを使用できます。ただし、ユーザーはアプリケーションに設定されたスキーマを使用して、データベースに

接続することはできません。この構成は、対応付けられたオブジェクトへのスキーマを介したアクセスを防ぎ、スキーマ・オブジェクトの保護を強化します。この場合、アプリケーションではALTER SESSION SET CURRENT_SCHEMA文を発行して、ユーザーを適切なアプリケーション・スキーマに接続できます。

親トピック: スキーマを使用したデータベース・オブジェクトの保護

12.10.2 共有スキーマでのデータベース・オブジェクトの保護

多くのアプリケーションでは、ユーザーがアクセスする必要があるのはアプリケーション・スキーマのみであるため、データベースで自分自身のアカウントつまりスキーマを必要としません。

たとえば、ユーザーJohn、FiruzehおよびJaneはすべて給与アプリケーションのユーザーで、financeデータベースのpayrollスキーマにアクセスする必要があるとします。この場合、データベースに自分自身のオブジェクトを作成する必要があるユーザーはいません。これらのユーザーに必要なのは、payrollオブジェクトへのアクセスのみです。この問題に対処するために、Oracle Databaseではエンタープライズ・ユーザー(スキーマに依存しないユーザー)が提供されています。

エンタープライズ・ユーザー、つまりディレクトリ・サービスで管理されるユーザーは、共有データベース・スキーマを使用するため、データベース・ユーザーとして作成する必要はありません。管理コストを削減するために、管理者はディレクトリに1つのエンタープライズ・ユーザーを1回作成し、他の多数のユーザーもアクセスできる共有スキーマで、そのユーザーを指し示すことができます。

関連項目:

エンタープライズ・ユーザーの管理の詳細は、『Oracle Databaseエンタープライズ・ユーザー・セキュリティ管理者ガイド』を参照してください。

親トピック: スキーマを使用したデータベース・オブジェクトの保護

12.11 アプリケーションでのオブジェクト権限

アプリケーションの設計時には、ユーザーのタイプとユーザーに必要なレベル・アクセスについて検討する必要があります。

- アプリケーション開発者に必要なオブジェクト権限に関する知識
  オブジェクト権限によって、エンド・ユーザーは、表、ビュー、順序、プロシージャ、ファンクション、パッケージなどのオブジェクトに対してアクションを実行できます。
- オブジェクト権限によって許可されるSQL文
  アプリケーションの実装時およびテスト時には、必要な各ロールを作成する必要があります。

親トピック: アプリケーション開発者のセキュリティの管理

12.11.1 アプリケーション開発者に必要なオブジェクト権限に関する知識

オブジェクト権限によって、エンド・ユーザーは、表、ビュー、順序、プロシージャ、ファンクション、パッケージなどのオブジェクトに対してアクションを実行できます。

表12-3に、オブジェクトの各タイプで使用できるオブジェクト権限の概要を示します。

表12-3 権限とスキーマ・オブジェクトとの関連

オブジェクト権限	表への適用	ビューへの適用	順序への適用	プロシージャへの適用(1)

オブジェクト権限	表への適用	ビューへの適用	順序への適用	プロシージャへの適用(1)
ALTER	はい	いいえ	はい	いいえ
DELETE	はい	はい	いいえ	いいえ
EXECUTE	いいえ	いいえ	いいえ	はい
INDEX	はい脚注 2	いいえ	いいえ	いいえ
INSERT	はい	はい	いいえ	いいえ
REFERENCES	はいはい(2)	いいえ	いいえ	いいえ
SELECT	はい	はい脚注 3	はい	いいえ
UPDATE	はい	はい	いいえ	いいえ

脚注 1 スタンドアロンのストアド・プロシージャ、ファンクションおよびパブリック・パッケージ構成

脚注2

ロールに付与できない権限

脚注3

スナップショットに対しても付与可能

関連トピック

- オブジェクト・アクションの監査

親トピック: アプリケーションでのオブジェクト権限

12.11.2 オブジェクト権限によって許可されるSQL文

アプリケーションの実装時およびテスト時には、必要な各ロールを作成する必要があります。

各ロールの使用例をテストし、データベースへのアクセス権がアプリケーション・ユーザーに正しく付与されることを確認します。テスト終了後は、アプリケーションの管理者と共同で各ユーザーに適切なロールが割り当てられていることを確認します。

表12-4に、表12-3で示したオブジェクト権限によって許可されるSQL文を示します。

表12-4 データベース・オブジェクト権限によって許可されるSQL文

オブジェクト権限	許可されるSQL文
ALTER	ALTER オブジェクト(表または順序)

オブジェクト権限	許可されるSQL文
	CREATE TRIGGER ON オブジェクト(表のみ)
DELETE	DELETE FROM オブジェクト(表、ビューまたはシノニム)
EXECUTE	EXECUTE オブジェクト(プロシージャまたはファンクション) パブリック・パッケージ変数への参照
INDEX	CREATE INDEX ON オブジェクト(表、ビューまたはシノニム)
INSERT	INSERT INTO オブジェクト(表、ビューまたはシノニム)
REFERENCES	オブジェクト(表のみ)に対する FOREIGN KEY 整合性制約を定義する CREATE または ALTER TABLE 文
SELECT	SELECT...FROM オブジェクト(表、ビュー、シノニムまたはスナップショット) 順序を使用する SQL 文

関連トピック

- 権限とロールについて
- オブジェクト・アクションの監査

親トピック: アプリケーションでのオブジェクト権限

12.12 データベース通信のセキュリティを強化するためのパラメータ

プロトコル・エラーによる不正パケットの処理や認証エラーの上限の構成など、パラメータを使用してセキュリティを管理できます。

- プロトコル・エラーによってデータベースで受信した不正パケット
  SEC_PROTOCOL_ERROR_TRACE_ACTION初期化パラメータで、プロトコル・エラーが発生したときにトレース・ファイルをどのように管理するかを制御します。
- 不正パケット受信後のサーバー実行の制御
  SEC_PROTOCOL_ERROR_FURTHER_ACTION初期化パラメータで、サーバーで不正パケットを受信した後のサーバー実行を制御します。
- 認証の最大試行回数の構成
  SEC_MAX_FAILED_LOGIN_ATTEMPTS初期化パラメータに設定された認証試行回数を超過すると、確立できなかった接続がデータベースで削除されます。
- データベース・バージョン・バナーの表示構成
  SEC_RETURN_SERVER_RELEASE_BANNER初期化パラメータを使用して、認証中の詳細な製品情報の表示を抑制できます。
- 不正なアクセスおよびユーザー・アクションの監査に関するバナーの構成

SEC_USER_UNAUTHORIZED_ACCESS_BANNERおよびSEC_USER_AUDIT_ACTION_BANNER初期化パラメータで、不正アクセスやユーザーの監査に関するバナーの表示を制御します。

親トピック: アプリケーション開発者のセキュリティの管理

12.12.1 プロトコル・エラーによってデータベースで受信した不正パケット

SEC_PROTOCOL_ERROR_TRACE_ACTION初期化パラメータで、プロトコル・エラーが発生したときにトレース・ファイルをどのように管理するかを制御します。

サーバーが不正なパケット、順序に誤りがあるパケット、プライベートまたは未使用のリモート・プロシージャ・コールを受信した場合、Oracle Call Interface(OCI)やTwo-Task Common(TTC)などのネットワーキング通信ユーティリティでは、スタック・トレースおよびヒープ・ダンプを格納する大規模なディスク・ファイルを生成できます。

通常、このディスク・ファイルは、非常に大規模になる可能性があります。侵入者は、サーバーに不正なパケットを繰り返し送信し、ディスクあふれやサービス拒否(DOS)攻撃を発生させることで、システムを使用できないようにする可能性があります。認証されていないクライアントが、この種の攻撃を仕掛ける可能性もあります。

これらの攻撃は、SEC_PROTOCOL_ERROR_TRACE_ACTION初期化パラメータを次の値のいずれかに設定することで防止できます。

- None: サーバーが不正なパケットを無視し、トレース・ファイルまたはログ・メッセージを生成しないように構成します。サーバーの可用性が不正なパケットの受信を認識することよりも圧倒的に重要な場合は、この設定を使用します。

  たとえば:

  ```
  SEC_PROTOCOL_ERROR_TRACE_ACTION = None
  ```

- Trace(デフォルト設定): トレース・ファイルを作成します。これは、ネットワーク・クライアントが不具合の結果として不正なパケットを送信している場合など、デバッグを目的とする場合に便利です。

  たとえば:

  ```
  SEC_PROTOCOL_ERROR_TRACE_ACTION = Trace
  ```

- Log: サーバー・トレース・ファイルに1行の短いメッセージを書き込みます。この選択肢では、一定レベルの監査とシステムの可用性とのバランスがとれます。

  たとえば:

  ```
  SEC_PROTOCOL_ERROR_TRACE_ACTION = Log
  ```

- Alert: データベース管理者または監視コンソールにアラート・メッセージを送信します。

  たとえば:

  ```
  SEC_PROTOCOL_ERROR_TRACE_ACTION = Alert
  ```

親トピック: データベース通信のセキュリティを強化するためのパラメータ

12.12.2 不正パケット受信後のサーバー実行の制御

SEC_PROTOCOL_ERROR_FURTHER_ACTION初期化パラメータで、サーバーで不正パケットを受信した後のサーバー実行を制御します。

Oracle Databaseは、クライアントまたはサーバー・プロトコルからエラーを検出した後も実行を継続する必要があります。ただ

し、これによって、ディスクあふれまたはサービス拒否攻撃を発生させる可能性のある不正なパケットを、サーバーがさらに受信する可能性があります。

- サーバーが悪意のあるクライアントから不正なパケットを受信しているときに、サーバー・プロセスの実行を詳細に制御するには、SEC_PROTOCOL_ERROR_FURTHER_ACTION初期化パラメータを次の値のいずれかに設定します。
  - Continue: サーバーの実行を続行します。ただし、サーバーがさらに攻撃を受ける可能性があることに注意してください。

    たとえば:

    ```
    SEC_PROTOCOL_ERROR_FURTHER_ACTION = Continue
    ```

  - (Delay,m): クライアントをm秒間遅延させます(サーバーが次のリクエストを同じクライアント接続から受け付けるまで)。この設定により悪意のあるクライアントはサーバー・リソースを過剰使用できなくなります。正当なクライアントのパフォーマンスも低下しますが、引き続き機能します。この設定を入力する場合は、カッコで囲みます。

    たとえば:

    ```
    SEC_PROTOCOL_ERROR_FURTHER_ACTION = (Delay,3)
    ```

    ALTER SYSTEMまたはALTER SESSION SQL文を使用してSEC_PROTOCOL_ERROR_FURTHER_ACTIONを設定している場合は、Delay設定を一重引用符または二重引用符で囲む必要があります。

    ```
    ALTER SYSTEM SEC_PROTOCOL_ERROR_FURTHER_ACTION = '(Delay,3)';
    ```

  - (Drop,n): クライアント接続は、n個の不正パケットを受信した後に強制的に終了します。この設定を使用すると、トランザクションの損失など、クライアントを犠牲にしてサーバー自体を保護できます。ただし、クライアントは再度接続して、同じ操作を再試行できます。この設定はカッコで囲みます。
    SEC_PROTOCOL_ERROR_FURTHER_ACTIONのデフォルト値は、(Drop,3)です。

    たとえば:

    ```
    SEC_PROTOCOL_ERROR_FURTHER_ACTION = (Drop,10)
    ```

    Delay設定と同様に、ALTER SYSTEMまたはALTER SESSIONを使用してこの設定を変更している場合は、Drop設定を一重引用符または二重引用符で囲む必要があります。

親トピック: データベース通信のセキュリティを強化するためのパラメータ

12.12.3 認証の最大試行回数の構成

SEC_MAX_FAILED_LOGIN_ATTEMPTS初期化パラメータに設定された認証試行回数を超過すると、確立できなかった接続がデータベースで削除されます。

接続作成の一環として、リスナーはサーバー・プロセスを開始し、そのプロセスをクライアントに付加します。この物理的な接続を使用して、クライアントは接続を認証できます。サーバー・プロセスが開始された後、このサーバー・プロセスに対してクライアントが認証がされます。侵入者は、サーバー・プロセスを起動し、様々なユーザー名とパスワードを使用して認証リクエストを無制限に発行し、データベースへのアクセスを試みます。

アプリケーション接続に対するログイン失敗回数を制限するには、SEC_MAX_FAILED_LOGIN_ATTEMPTS初期化パラメータを設定して、接続に対する認証試行回数を制限します。認証の試行が指定した回数失敗すると、データベース・プロセスは

接続を切断し、サーバー・プロセスが終了します。デフォルトでは、SEC_MAX_FAILED_LOGIN_ATTEMPTSは3に設定されます。

SEC_MAX_FAILED_LOGIN_ATTEMPTS初期化パラメータは、潜在的な侵入者によるアプリケーションへの攻撃を防ぐために設計されているだけでなく、パスワードを忘れた正当なユーザーも対象となることに留意してください。sqlnet.ora INBOUND_CONNECT_TIMEOUTパラメータとFAILED_LOGIN_ATTEMPTSプロファイル・パラメータもログイン失敗を制限しますが、この2つのパラメータは正当なユーザー・アカウントにのみ適用されるという点で異なります。

たとえば、最大試行回数を5に制限するには、initsid.ora初期化パラメータ・ファイルで、次のようにSEC_MAX_FAILED_LOGIN_ATTEMPTSを設定します。

```
SEC_MAX_FAILED_LOGIN_ATTEMPTS = 5
```

親トピック: データベース通信のセキュリティを強化するためのパラメータ

12.12.4 データベース・バージョン・バナーの表示構成

SEC_RETURN_SERVER_RELEASE_BANNER初期化パラメータを使用して、認証中の詳細な製品情報の表示を抑制できます。

クライアント接続(Oracle Call Interfaceクライアントを含む)が認証されてから、詳細な製品バージョン情報にアクセスできるようにする必要があります。侵入者は、データベース・バージョンを使用して、データベース・ソフトウェアに存在するセキュリティの脆弱性に関する情報を検出する可能性があります。

- 認証されていないクライアントに対してデータベース・バージョン・バナーの表示を制限するには、initsid.ora初期化パラメータ・ファイルでSEC_RETURN_SERVER_RELEASE_BANNER初期化パラメータをTRUEまたはFALSEのいずれかに設定します。

  デフォルトでは、SEC_RETURN_SERVER_RELEASE_BANNERはFALSEに設定されます。

たとえば、TRUEに設定すると、Oracle Databaseに正確なデータベース・バージョンが表示されます。たとえば、リリース19.1.0.0の場合は次のようになります。

```
Oracle Database 19c Enterprise Edition Release 19.1.0.0 - Production
```

リリース番号にポイント・リリース表記法(Oracle Databaseリリース19.1.0.1など)が使用されている場合、バナーには次のように表示されます。

```
Oracle Database 19c Enterprise Edition Release 19.1.0.1 - Production
```

ただし、同じリリースでこのパラメータをNOに設定すると、このバナーは制限されて、リリース19.1で始まる次の固定テキスト(19.1.0.1のかわりに19.1.0.0.0)が表示されます。

```
Oracle Database 19c Release 19.1.0.0.0 - Production
```

親トピック: データベース通信のセキュリティを強化するためのパラメータ

12.12.5 不正なアクセスおよびユーザー・アクションの監査に関するバナーの構成

SEC_USER_UNAUTHORIZED_ACCESS_BANNERおよびSEC_USER_AUDIT_ACTION_BANNER初期化パラメータで、不正アクセスやユーザーの監査に関するバナーの表示を制御します。

不正なアクセスおよびユーザー・アクション監査をユーザーに警告するには、バナーを作成して構成する必要があります。この通知は、クライアント・アプリケーションがデータベースにログインすると使用可能になります。

- これらのバナーを構成して表示するには、データベース・サーバー側で次のsqlnet.oraパラメータを設定して、バナー情報が含まれるテキスト・ファイルを指し示します。
  - SEC_USER_UNAUTHORIZED_ACCESS_BANNER。たとえば:

    ```
    SEC_USER_UNAUTHORIZED_ACCESS_BANNER =
    /opt/Oracle/12c/dbs/unauthaccess.txt
    ```

  - SEC_USER_AUDIT_ACTION_BANNER。たとえば:

    ```
    SEC_USER_AUDIT_ACTION_BANNER = /opt/Oracle/12c/dbs/auditactions.txt
    ```

デフォルトでは、これらのパラメータは設定されません。さらに、バナー・テキストに使用される文字数には512バイトの制限があることを注意してください。

これらのパラメータを設定したら、これらのバナーを取得してエンドユーザーに表示するように、Oracle Call Interfaceアプリケーションで適切なOCI APIを使用する必要があります。

親トピック: データベース通信のセキュリティを強化するためのパラメータ

第III部 データへのアクセス制御

第III部では、データへのアクセスの制御方法について説明します。

- アプリケーション・コンテキストを使用したユーザー情報の取得
  アプリケーション・コンテキストにはユーザーIDが格納されており、これに基づいてデータベースのデータにユーザーがアクセスできるかを否かを決定できます。
- Oracle Virtual Private Databaseを使用したデータ・アクセスの制御
  Oracle Virtual Private Database (VPD)を使用すると、データにアクセスするユーザーをフィルタ処理できます。
- 透過的機密データ保護の使用
  透過的機密データ保護により、機密データを保持するデータベースのすべての表の列を確認できます。
- データ・ディクショナリでの機密性の高い資格証明データの暗号化
  機密性の高い資格証明情報(データ・ディクショナリに格納されているパスワードなど)を暗号化できます。
- 手動によるデータ暗号化
  DBMS_CRYPTO PL/SQLパッケージを使用して、データを手動で暗号化できます。

13 アプリケーション・コンテキストを使用したユーザー情報の取得

アプリケーション・コンテキストにはユーザーIDが格納されており、これに基づいてデータベースのデータにユーザーがアクセスできるかを否かを決定できます。

- アプリケーション・コンテキストについて
  アプリケーション・コンテキストには、ユーザーがデータに対して保持するアクセスを制御する場合に多くのメリットがあります。
- アプリケーション・コンテキストの種類
  アプリケーション・コンテキストには、3種類の一般的なカテゴリがあります。
- データベース・セッション・ベースのアプリケーション・コンテキストの使用
  データベース・セッション・ベースのアプリケーション・コンテキストを使用すると、ユーザーのセッション・ベースの情報を取得できます。
- グローバル・アプリケーション・コンテキスト
  グローバル・アプリケーション・コンテキストを使用して、Oracle Real Application Clusters環境などのデータベース・セッション間でアプリケーション値にアクセスできます。
- クライアント・セッション・ベースのアプリケーション・コンテキストの使用
  クライアント・セッション・ベースのアプリケーション・コンテキストは、ユーザー・グローバル領域(UGA)に格納されます。
- アプリケーション・コンテキストのデータ・ディクショナリ・ビュー
  Oracle Databaseには、アプリケーション・コンテキストに関する情報を提供するデータ・ディクショナリ・ビューが用意されています。

親トピック: データへのアクセス制御

13.1 アプリケーション・コンテキストについて

アプリケーション・コンテキストには、ユーザーがデータに対して保持するアクセスを制御する場合に多くのメリットがあります。

- アプリケーション・コンテキストとは
  アプリケーション・コンテキストは、Oracle Databaseがメモリーに格納する名前と値のペアです。
- アプリケーション・コンテキストの構成要素
  アプリケーション・コンテキストには2つのコンポーネントがあり、名前と値のペアを構成します。
- アプリケーション・コンテキストの値の格納場所
  Oracle Databaseではアプリケーション・コンテキスト値が保護データ・キャッシュに格納されます。
- アプリケーション・コンテキストを使用する利点
  ほとんどのアプリケーションには、アプリケーション・コンテキストに使用可能なある種の情報が含まれています。
- エディションがアプリケーション・コンテキストの値に与える影響
  Oracle Databaseでは、アプリケーション・コンテキスト・パッケージの影響を受けるすべてのエディションでアプリケーション・コンテキストが設定されます。
- マルチテナント環境でのアプリケーション・コンテキスト
  マルチテナント環境のどこでアプリケーションを作成するかによって、アプリケーション・コンテキストを作成する場所が決まります。

親トピック: アプリケーション・コンテキストを使用したユーザー情報の取得

13.1.1 アプリケーション・コンテキストとは

アプリケーション・コンテキストとは、Oracle Databaseがメモリーに格納する名前と値のペアです。

コンテキストには、**ネームスペース**と呼ばれるラベル(たとえば、従業員IDを取得するアプリケーション・コンテキストには`empno_ctx`)があります。このコンテキストにより、Oracle Databaseは認証中にデータベース・ユーザーと非データベース・ユーザーに関する情報を入手できます。

コンテキスト内は名前と値のペア(結合配列)です。名前は、値を保持するメモリー内の場所を指定します。アプリケーションはアプリケーション・コンテキストを使用して、ユーザーに関するセッション情報(ユーザーIDまたは他のユーザー固有の情報など)またはクライアントIDにアクセスして、その情報をデータベースに安全に引き渡すことができます。

この情報を使用して、ユーザーがアプリケーションを通じてデータにアクセスできるようにしたりアクセスできないようにできます。アプリケーション・コンテキストを使用して、データベース・ユーザーと非データベース・ユーザーの両方を認証できます。

関連トピック

- アプリケーション・コンテキスト値の監査

親トピック: アプリケーション・コンテキストについて

13.1.2 アプリケーション・コンテキストの構成要素

アプリケーション・コンテキストには2つのコンポーネントがあり、名前と値のペアを構成します。

これらのコンポーネントは次のとおりです。

- 名前。値に関連付けられている属性セットの名前を示します。たとえば、`empno_ctx`アプリケーション・コンテキストは従業員IDを`HR.EMPLOYEES`表から取得する場合、名前を`employee_id`にすることができます。
- 値。属性により設定される値を示します。たとえば、`empno_ctx`アプリケーション・コンテキストでは、`HR.EMPLOYEES`表から従業員IDを取得する場合、このIDの値を設定する、`emp_id`と呼ばれる値を作成できます。

アプリケーション・コンテキストは、データベース・セッション中にアクセスされる情報を保持するグローバル変数と考えてください。セキュア・アプリケーション・コンテキストの値を設定するには、`DBMS_SESSION.SET_CONTEXT`プロシージャを使用するPL/SQLパッケージ・プロシージャを作成する必要があります。実際、コンテキストが`INITIALIZED EXTERNALLY`または`INITIALIZED GLOBALLY`とマークされていない場合は、事実上、この方法がアプリケーション・コンテキスト値を設定できる唯一の方法となります。アプリケーション・コンテキスト属性には、アプリケーション・コンテキストの作成時ではなく、実行時に値を割り当てることができます。ユーザーではなくトラステッド・プロシージャによって値が割り当てられるため、これはセキュア・アプリケーション・コンテキストと呼ばれます。クライアント・セッション・ベースのアプリケーション・コンテキストの場合、Oracle Call Interface(OCI)コールを使用してアプリケーション・コンテキストを設定することもできます。

親トピック: アプリケーション・コンテキストについて

13.1.3 アプリケーション・コンテキストの値の格納場所

Oracle Databaseではアプリケーション・コンテキスト値が保護データ・キャッシュに格納されます。

このキャッシュはユーザー・グローバル領域(UGA)またはシステム("共有"と呼ばれる場合もある)グローバル領域(SGA)にあります。格納されたアプリケーション・コンテキストの値はセッション中に取得されます。アプリケーション・コンテキストには、このデータ・キャッシュ内の値が格納されるため、アプリケーションのパフォーマンスが向上します。アプリケーション・コンテキストは単独で使用でき、Oracle Virtual Private Databaseポリシーまたはその他のファイングレイン・アクセス制御ポリシーと併用することもできます。

関連トピック

- Oracle Virtual Private Databaseでのアプリケーション・コンテキストの使用

親トピック: アプリケーション・コンテキストについて

13.1.4 アプリケーション・コンテキストを使用する利点

ほとんどのアプリケーションには、アプリケーション・コンテキストに使用可能なある種の情報が含まれています。

たとえば、ORDER_NUMBER列とCUSTOMER_NUMBER列を備えた表を使用する受注管理アプリケーションでは、それらの列の値をセキュリティ属性として使用して、顧客によるアクセスをその顧客のIDに基づいて顧客自身の注文のみに制限できます。

アプリケーション・コンテキストは、次の目的で使用されます。

- ファイングレイン・アクセス・コントロールの規定(Oracle Virtual Private Databaseポリシーなどで)
- 複数層環境でのユーザー識別情報の保持
- アプリケーションに対する強力なセキュリティの規定(アプリケーション・コンテキストはユーザーではなくトラステッド・プロシージャによって制御されるため)
- アプリケーションがファイングレイン監査やPL/SQL条件文またはループで使用する際に必要な属性に対して、保護データ・キャッシュとしての機能を果すことによるパフォーマンスの向上

  このキャッシュによって、これらの属性が必要になるたびにデータベースに問い合せるというオーバーヘッドが軽減されます。アプリケーションはセッション・データを表から繰り返し取得する必要がなく、このデータがアプリケーション・コンテキストによってキャッシュに格納されるため、アプリケーションのパフォーマンスが大幅に向上します。
- アプリケーションで定義、変更およびアクセスできる名前と値のペアの保持領域として機能

親トピック: アプリケーション・コンテキストについて

13.1.5 エディションがアプリケーション・コンテキストの値に与える影響

Oracle Databaseでは、アプリケーション・コンテキスト・パッケージの影響を受けるすべてのエディションでアプリケーション・コンテキストが設定されます。

アプリケーション・コンテキストで設定される値は、アプリケーション・コンテキストが影響するすべてのエディションで表示されます。データベース内のすべてのエディション、およびそれが使用可能であるかどうかを検索するには、ALL_EDITIONSデータ・ディクショナリ・ビューを問い合せます。

関連項目:

エディションの詳細は、『Oracle Database開発ガイド』を参照してください。

親トピック: アプリケーション・コンテキストについて

13.1.6 マルチテナント環境でのアプリケーション・コンテキスト

マルチテナント環境のどこでアプリケーションを作成するかによって、アプリケーション・コンテキストを作成する場所が決まります。

アプリケーション・ルートまたはCDBルートにアプリケーションがインストールされると、そのアプリケーションはアプリケーション・コンテナまたはシステム・コンテナおよび関連付けられたアプリケーションPDBからアクセスできるようになります。そのルートで共通アプリ

ケーション・コンテキストを作成する必要があります。

アプリケーション・コンテナで使用する共通アプリケーション・コンテキストを作成する場合は、次の点に注意してください。

- マルチテナント環境でアプリケーション・コンテキストを作成するには、CREATE CONTEXT SQL文でCONTAINER句を設定します。たとえば、アプリケーション・ルートで共通アプリケーション・コンテキストを作成するには、CONTAINERをALLに設定してCREATE CONTEXTを実行する必要があります。PDBでアプリケーション・コンテキストを作成するには、CONTAINERをCURRENTに設定します。
- ローカル・アプリケーション・コンテキストと共通アプリケーション・コンテキストに同じ名前を使用することはできません。既存のアプリケーション・コンテキストの名前は、次の問合せを実行して検索できます。

```
SELECT OBJECT_NAME FROM DBA_OBJECTS WHERE OBJECT_TYPE ='CONTEXT';
```

- 共通アプリケーション・コンテキストを管理するために作成するPL/SQLパッケージは、共通PL/SQLパッケージであることが必要です。つまり、それがアプリケーション・ルートまたはCDBルートに存在する必要があります。特定のPDBのアプリケーション・コンテキストを作成する場合、関連付けられたPL/SQLパッケージをそのPDBに格納する必要があります。
- 共通アプリケーション・コンテキストのアプリケーション・コンテナまたはシステム・コンテナから共通セッション・アプリケーション・コンテキストの下に設定した名前と値のペアは、共通ユーザーが異なるコンテナにアクセスする場合、他のアプリケーション・コンテナまたはシステム・コンテナからアクセスできません。
- アプリケーション・コンテナまたはシステム・コンテナから共通グローバル・アプリケーション・コンテキストの下に設定した名前と値のペアは、同じユーザー・セッションの同じコンテナ内の場合にかぎりアクセスできます。
- アプリケーションは、アプリケーション・ルート、CDBルートまたはPDBのいずれに存在していても、アプリケーション・コンテキストの値を取得できます。
- CDBまたはアプリケーション・コンテナへのPDBの接続操作中、共通アプリケーション・コンテキストの名前がPDBのローカル・アプリケーション・コンテキストの名前と競合する場合は、PDBを制限モードでオープンする必要があります。データベース管理者は、PDBを通常モードでオープンする前に、競合を修正する必要があります。
- 切断操作中、共通アプリケーション・コンテキストはその共通セマンティクスを維持することで、後にそのPDBが同じ名前の共通アプリケーション・コンテキストがある別のCDBに接続する場合、引き続き共通オブジェクトのように動作します。PDBが同じ共通アプリケーション・コンテキストが存在しないアプリケーション・コンテナまたはシステム・コンテナに接続する場合、そのPDBはローカル・オブジェクトのように動作します。

アプリケーション・コンテキストがローカル・アプリケーション・コンテキストであるかアプリケーションの共通アプリケーション・コンテキストであるかを確認するには、DBA_CONTEXTまたはALL_CONTEXTデータ・ディクショナリ・ビューのSCOPE列を問い合せます。

親トピック: アプリケーション・コンテキストについて

13.2 アプリケーション・コンテキストの種類

アプリケーション・コンテキストには、3種類の一般的なカテゴリがあります。

カテゴリは次のとおりです。

- データベース・セッション・ベースのアプリケーション・コンテキスト。このタイプのアプリケーション・コンテキストは、データベース・ユーザー・セッション(つまりUGA)のキャッシュに格納されているデータを取得します。データベース・セッション・ベースのアプリケーション・コンテキストは、3つのカテゴリに分類されます。
    - ローカルで初期化。ユーザーのセッションに対してアプリケーション・コンテキストをローカルで初期化します。

- 外部で初期化。Oracle Call Interface(OCI)アプリケーション、ジョブ・キュー・プロセスまたは接続ユーザーのデータベース・リンクからアプリケーション・コンテキストを初期化します。
- グローバルに初期化。LDAPディレクトリなどの集中格納場所から属性と値を使用します。

このタイプのアプリケーション・コンテキストについては、「データベース・セッション・ベースのアプリケーション・コンテキストの使用」を参照してください。

- グローバル・アプリケーション・コンテキスト。このタイプは、システム・グローバル領域(SGA)に格納されるデータを取得するため、3層アーキテクチャの中間層アプリケーションなど、セッションを使用しないアプリケーションに対して使用できます。グローバル・アプリケーション・コンテキストは、セッション・コンテキストをセッション間で、たとえば接続プールの実装を通じて共有する場合に便利です。

  このタイプについては、「グローバル・アプリケーション・コンテキスト」を参照してください。

- クライアント・セッション・ベースのアプリケーション・コンテキスト。このタイプのアプリケーション・コンテキストは、クライアント側のOracle Call Interface関数を使用してユーザー・セッション・データを設定し、次に、必要なセキュリティ・チェックを実行してユーザー・アクセスを制限します。

  このタイプについては、「クライアント・セッション・ベースのアプリケーション・コンテキストの使用」を参照してください。

表13-1に、異なる種類のアプリケーション・コンテキストの要約を示します。

表13-1 アプリケーション・コンテキストの種類

アプリケーション・コンテキストの種類	UGAへの格納	SGAへの格納	接続ユーザー・データベース・リンクのサポート	ユーザーのアプリケーション・コンテキストの集中保管のサポート	セッションを使用しない複数層アプリケーションのサポート
ローカルで初期化されるデータベース・セッション・ベースのアプリケーション・コンテキスト	はい	いいえ	いいえ	いいえ	いいえ
外部で初期化されるデータベース・セッション・ベースのアプリケーション・コンテキスト	はい	いいえ	はい	いいえ	いいえ
グローバルに初期化されるデータベース・セッション・ベースのアプリケーション・コンテキスト	はい	いいえ	いいえ	はい	いいえ
グローバル・アプリケーション・コンテキスト	いいえ	はい	いいえ	いいえ	はい
クライアント・セッション・ベースのアプリケーション・コンテ	はい	いいえ	はい	いいえ	はい

アプリケーション・コンテキストの種類	UGAへの格納	SGAへの格納	接続ユーザー・データベース・リンクのサポート	ユーザーのアプリケーション・コンテキストの集中保管のサポート	セッションを使用しない複数層アプリケーションのサポート
キスト					

親トピック: アプリケーション・コンテキストを使用したユーザー情報の取得

13.3 データベース・セッション・ベースのアプリケーション・コンテキストの使用

データベース・セッション・ベースのアプリケーション・コンテキストを使用すると、ユーザーのセッション・ベースの情報を取得できます。

- データベース・セッション・ベースのアプリケーション・コンテキストについて
  データベース・セッション・ベースのアプリケーション・コンテキストで、データベース・ユーザーのセッション情報を取得します。
- データベース・セッション・ベースのアプリケーション・コンテキストのコンポーネント
  データベース・セッション・ベースのアプリケーション・コンテキストは、コンテキストのデータを取得および設定して、ユーザーがログインするときにこのコンテキストを設定します。
- データベース・セッション・ベースのアプリケーション・コンテキストの作成
  データベース・セッション・ベースのアプリケーション・コンテキストは、ユーザーのセッション情報を格納する名前付きのオブジェクトです。
- データベース・セッション・ベースのアプリケーション・コンテキストを設定するためのパッケージの作成
  PL/SQLパッケージを使用し、セッション情報を取得してアプリケーション・コンテキストの名前と値の属性を設定できます。
- データベース・セッションのアプリケーション・コンテキスト・パッケージを実行するログイン・トリガー
  データベース・インスタンスにログインした後、ユーザーはデータベース・セッションのアプリケーション・コンテキスト・パッケージを実行する必要があります。
- 例: 単純なログイン・トリガーの作成
  CREATE TRIGGER文で簡単なログイン・トリガーを作成できます。
- 例: 本番環境用のログイン・トリガーの作成
  CREATE TRIGGER文で、本番環境のログイン・トリガーを作成できます。
- 例: 開発環境用のログイン・トリガーの作成
  CREATE TRIGGER文で、開発環境のログイン・トリガーを作成できます。
- 例: データベース・セッション・ベースのアプリケーション・コンテキストの作成と使用
  このチュートリアルでは、データベースへのログインを試みるユーザーのIDをチェックするアプリケーション・コンテキストの作成方法を示します。
- データベース・セッション・ベースのアプリケーション・コンテキストの外部での初期化
  データベース・セッション・ベースのアプリケーション・コンテキストを外部で初期化すると、アプリケーション・コンテキストがユーザー・グローバル領域(UGA)に格納されるため、パフォーマンスが向上します。
- データベース・セッション・ベースのアプリケーション・コンテキストのグローバルな初期化
  データベース・セッション・ベースのアプリケーションが集中格納場所に格納されている場合、そのアプリケーションはLDAPディレクトリ全域から使用できます。
- 外部化されたデータベース・セッション・ベースのアプリケーション・コンテキスト
  多くのアプリケーションでは、ファイングレイン・アクセス・コントロールに使用される属性をデータベース・メタデータ表に格納します。

親トピック: アプリケーション・コンテキストを使用したユーザー情報の取得

13.3.1 データベース・セッション・ベースのアプリケーション・コンテキストについて

データベース・セッション・ベースのアプリケーション・コンテキストで、データベース・ユーザーのセッション情報を取得します。

このタイプのアプリケーション・コンテキストは、Oracle Database内でPL/SQLプロシージャを使用し、管理の対象となるデータを取得、設定および保護します。

データベース・セッション・ベースのアプリケーション・コンテキストは、Oracle Databaseの中で完全管理されます。Oracle Databaseが値を設定し、ユーザーがセッションを終了すると、キャッシュに保管されたアプリケーション・コンテキスト値を自動的にクリアします。ユーザー接続が異常終了すると(たとえば停電)、PMONバックグラウンド・プロセスはアプリケーション・コンテキスト・データをクリーン・アップします。アプリケーション・コンテキストをキャッシュから消去する必要はありません。

アプリケーション・コンテキストをOracle Databaseで管理する利点は、アプリケーション・コンテキストを集中管理できる点です。このデータベースにアクセスするアプリケーションは、このアプリケーション・コンテキストを使用して、そのアプリケーションに対するユーザーのアクセスを許可または防止する必要があります。これによって、パフォーマンス向上とセキュリティ強化の両方の利点が得られます。

ノート:

ユーザーがアプリケーション・ユーザー(つまり、データベースに存在しないユーザー)の場合は、グローバル・アプリケーション・コンテキストの使用を考慮してください。

関連トピック

- グローバル・アプリケーション・コンテキスト

親トピック: データベース・セッション・ベースのアプリケーション・コンテキストの使用

13.3.2 データベース・セッション・ベースのアプリケーション・コンテキストのコンポーネント

データベース・セッション・ベースのアプリケーション・コンテキストは、コンテキストのデータを取得および設定して、ユーザーがログインするときにこのコンテキストを設定します。

データベース・セッション・ベースのアプリケーション・コンテキストを作成して使用するには、アプリケーション・コンテキスト、データの取得とコンテキストの設定を実行するプロシージャ、およびユーザーのログイン時にコンテキストを設定する手段の3つのコンポーネントを使用する必要があります。

- アプリケーション・コンテキスト。アプリケーション・コンテキストを作成するには、CREATE CONTEXT SQL文を使用します。この文は、アプリケーション・コンテキスト(ネームスペース)名を設定し、セッション・データを取得してアプリケーション・コンテキストを設定するように設計されたPL/SQLプロシージャに対して、その名前を関連付けます。
- データの取得とコンテキストの設定を実行するPL/SQLプロシージャ。このプロシージャで実行する必要があるタスクの概要については、データベース・セッション・ベースのアプリケーション・コンテキストを管理するパッケージについてを参照してください。理想的には、必要に応じて他のプロシージャ(タスクのエラー・チェックなど)を挿入できるように、このプロシージャは1つのパッケージ内に作成します。
- ユーザーのログイン時にアプリケーション・コンテキストを設定する手段。アプリケーション・コンテキストを使用するアプリケーションにログインするユーザーは、そのアプリケーション・コンテキストを設定するPL/SQLパッケージを実行する必要が

あります。そのためには、ユーザーがログインするたびに起動するログイン・トリガーを使用するか、この機能をアプリケーションに埋め込むことができます。

ローカルで初期化されるデータベース・セッション・ベースのアプリケーション・コンテキストの作成方法と使用方法は、例: データベース・セッション・ベースのアプリケーション・コンテキストの作成と使用を参照してください。

さらに、セッション・ベースのアプリケーション・コンテキストは、外部で、またはグローバルに初期化できます。いずれの場合も、コンテキスト情報はユーザー・セッションに格納されます。

- 外部での初期化。この初期化は、OCIインタフェース、ジョブ・キュー・プロセスまたは接続ユーザーのデータベース・リンクから発生します。詳細は、データベース・セッション・ベースのアプリケーション・コンテキストの外部での初期化を参照してください。
- グローバルな初期化。この初期化では、LDAPディレクトリなどの集中格納場所から属性と値を使用します。詳細は、データベース・セッション・ベースのアプリケーション・コンテキストのグローバルな初期化を参照してください。

親トピック: データベース・セッション・ベースのアプリケーション・コンテキストの使用

13.3.3 データベース・セッション・ベースのアプリケーション・コンテキストの作成

データベース・セッション・ベースのアプリケーション・コンテキストは、ユーザーのセッション情報を格納する名前付きのオブジェクトです。

- データベース・セッション・ベースのアプリケーション・コンテキストの作成について
  ユーザー作成のネームスペースを使用して、データベース・ユーザー・セッション(UGA)には、セッション・ベースのアプリケーション・コンテキストが格納されます。
- データベース・セッション・ベースのアプリケーション・コンテキストの作成
  CREATE CONTEXT SQL文を使用して、データベース・セッション・ベースのアプリケーション・コンテキストを作成できます。
- 複数のアプリケーションのデータベース・セッション・ベースのアプリケーション・コンテキスト
  各アプリケーションには、独自の属性を持つアプリケーション・コンテキストを作成できます。

親トピック: データベース・セッション・ベースのアプリケーション・コンテキストの使用

13.3.3.1 データベース・セッション・ベースのアプリケーション・コンテキストの作成について

ユーザー作成のネームスペースを使用して、データベース・ユーザー・セッション(UGA)には、セッション・ベースのアプリケーション・コンテキストが格納されます。

各アプリケーション・コンテキストには一意の属性が必要で、1つのネームスペースに属している必要があります。つまり、コンテキスト名は、スキーマ内のみでなくデータベース内でも一意である必要があります。

アプリケーション・コンテキストを作成するには、CREATE ANY CONTEXTシステム権限が必要です。アプリケーション・コンテキストを削除する場合は、DROP CONTEXT文を使用するためにDROP ANY CONTEXT権限が必要になります。

アプリケーション・コンテキストの所有権については、CREATE ANY CONTEXT権限とDROP ANY CONTEXT権限を付与されたユーザーがそのアプリケーション・コンテキストを作成および削除できた場合でも、SYSスキーマが所有しています。Oracle Databaseは、作成したスキーマ・アカウントにコンテキストを関連付けますが、このユーザーを削除した場合、コンテキストはそれでもSYSスキーマの中に存在します。ユーザーSYSであれば、アプリケーション・コンテキストを削除できます。

既存のアプリケーション・コンテキストの名前は、次の問合せを実行して検索できます。

```
SELECT OBJECT_NAME FROM DBA_OBJECTS WHERE OBJECT_TYPE ='CONTEXT';
```

親トピック: データベース・セッション・ベースのアプリケーション・コンテキストの作成

13.3.3.2 データベース・セッション・ベースのアプリケーション・コンテキストの作成

CREATE CONTEXT SQL文を使用して、データベース・セッション・ベースのアプリケーション・コンテキストを作成できます。

データベース・セッション・ベースのアプリケーション・コンテキストを作成する場合、そのアプリケーション・コンテキストのネームスペースを作成し、ユーザーのセッション情報を保持する名前と値のペアを管理するPL/SQLパッケージとそのネームスペースを関連付ける必要があります。PL/SQLパッケージは、コンテキストの作成時には不要ですが、実行時には存在している必要があります。

- データベース・セッション・ベースのアプリケーション・コンテキストを作成するには、CREATE CONTEXT SQL文を使用します。

  たとえば:

  ```
  CREATE CONTEXT empno_ctx USING set_empno_ctx_pkg CONTAINER = CURRENT;
  ```

この例では、次のようになります。

- empno_ctxはコンテキスト・ネームスペースです。
- set_empno_ctx_pkgは、そのempno_ctxネームスペースの属性を設定するパッケージです(コンテキストの作成時には不要です)。このアプリケーション・コンテキストで使用可能なパッケージの作成方法の例は、ステップ3: セッション・データを取得してアプリケーション・コンテキストを設定するパッケージの作成を参照してください。
- CONTAINERは、現在のPDBでアプリケーション・コンテキストを作成します。アプリケーション・ルートまたはCDBルートでアプリケーション・コンテキストを作成するには、CONTAINERをALLに設定する必要があります。

コンテキストを作成するときは、そのコンテキストの名前/値の属性をCREATE CONTEXT文に設定しないでください。かわりに、これらは、アプリケーション・コンテキストに関連付けるPL/SQLパッケージに設定してください。これは、不正なユーザーによって、適切な属性の検証なしにコンテキスト属性が変更されないようにするためです。このパッケージがアプリケーション・コンテキストと同じコンテナにあることを確認してください。たとえば、アプリケーション・コンテキストをPDBに作成した場合、PL/SQLパッケージはそのPDBに存在する必要があります。

ノート:

CLIENTCONTEXT というコンテキストは作成できません。これは、クライアント・セッション・ベースのアプリケーション・コンテキストで使用される予約語です。このタイプのアプリケーション・コンテキストの詳細は、クライアント・セッション・ベースのアプリケーション・コンテキストの使用を参照してください。

親トピック: データベース・セッション・ベースのアプリケーション・コンテキストの作成

13.3.3.3 複数のアプリケーションのデータベース・セッション・ベースのアプリケーション・コンテキスト

各アプリケーションには、独自の属性を持つアプリケーション・コンテキストを作成できます。

たとえば、General Ledger(一般会計)、Order Entry(受注管理)およびHuman Resources(人事管理)という3つのアプリケーションがあるとします。

これらの各アプリケーションには、異なる属性を指定できます。

- 受注管理アプリケーション・コンテキストには、CUSTOMER_NUMBER属性を指定できます。
- 一般会計アプリケーション・コンテキストには、SET_OF_BOOKS属性とTITLE属性を指定できます。

- 人事管理アプリケーション・コンテキストには、ORGANIZATION_ID、POSITION、COUNTRYの各属性を指定できます。

属性がアクセスするデータは、アプリケーション外の表に保管されます。たとえば、受注管理アプリケーションではOE.CUSTOMERSという名前の表が使用されており、この中のCUSTOMER_NUMBER列にはCUSTOMER_NUMBER属性のデータが入ります。いずれの場合にも、アプリケーション・コンテキストを厳密なセキュリティ・ニーズに適用できます。

親トピック: データベース・セッション・ベースのアプリケーション・コンテキストの作成

13.3.4 データベース・セッション・ベースのアプリケーション・コンテキストを設定するためのパッケージの作成

PL/SQLパッケージを使用し、セッション情報を取得してアプリケーション・コンテキストの名前と値の属性を設定できます。

- データベース・セッション・ベースのアプリケーション・コンテキストを管理するパッケージについて
  これは、アプリケーション・コンテキストによって表されるセッション・データを管理するプロシージャを定義します。
- SYS_CONTEXTファンクションを使用したセッション情報の取得
  SYS_CONTEXTファンクションを使用して、アプリケーション・コンテキストのセッション情報を取得できます。
- SYS_CONTEXT設定の確認
  DUAL表に格納されるSYS_CONTEXT設定を確認できます。
- SYS_CONTEXTでの動的SQL
  指定した問合せを実行する間にポリシーの変更が予想されるセッション中は、その問合せに動的SQLを使用する必要があります。
- パラレル問合せでのSYS_CONTEXT
  パラレル問合せに埋め込まれているSQL関数内でSYS_CONTEXTを使用すると、この関数にアプリケーション・コンテキストが挿入されます。
- データベース・リンクでのSYS_CONTEXT
  SYS_CONTEXTファンクションはデータベース・リンクと一緒に使用できます。
- セッション情報を設定するためのDBMS_SESSION.SET_CONTEXT
  SYS_CONTEXTでユーザーのセッション・データを取得した後、ユーザー・セッションからアプリケーション・コンテキスト値を設定できます。
- 例: アプリケーション・コンテキストの値を作成する単純なプロシージャ
  プロシージャでDBMS_SESSION.SET_CONTEXT文を使用して、アプリケーション・コンテキストの値を設定できます。

親トピック: データベース・セッション・ベースのアプリケーション・コンテキストの使用

13.3.4.1 データベース・セッション・ベースのアプリケーション・コンテキストを管理するパッケージについて

これは、アプリケーション・コンテキストにより示されたセッション・データを管理するプロシージャを定義します。

このパッケージは、通常、セキュリティ管理者のスキーマに作成されます。パッケージでは、次のタスクを実行する必要があります。

- セッション情報の取得。ユーザー・セッション情報の取得には、SYS_CONTEXT SQL関数を使用できます。このSYS_CONTEXT関数は、コンテキストのネームスペースに関連付けられているパラメータの値を戻します。この関数は、SQL文とPL/SQL文の両方で使用できます。通常は、組み込まれているUSERENVネームスペースを使用してユーザーのセッション情報を取得します。SYS_SESSION_ROLESネームスペースを使用して、セッションに対して指定したロールが現在有効化されているかどうかを示すこともできます。

- CREATE CONTEXTで作成したアプリケーション・コンテキストの名前/値の属性の設定。アプリケーション・コンテキストの名前/値の属性の設定には、DBMS_SESSION.SET_CONTEXTプロシージャを使用できます。この名前/値の属性には、ユーザーID、IPアドレス、認証モード、アプリケーション名などの情報を格納できます。設定した属性の値は、再設定するまで、またはユーザーがセッションを終了するまでそのまま残ります。次のことに注意してください。
  - ネームスペース内のパラメータの値がすでに設定されている場合は、SET_CONTEXTによってこの値が上書きされます。
  - コンテキストの値が変更された場合、その内容はただちに反映され、SYS_CONTEXTファンクションによって値を取得する後続のコールでは、最新の値が戻されます。
- ユーザーによる実行。パッケージを作成した後、ログインする場合ユーザーはそのパッケージを実行する必要があります。ユーザーがログインするときパッケージを自動的に実行するためのログイン・トリガーを作成するか、この機能をアプリケーションに埋め込むことができます。アプリケーション・コンテキスト・セッション値はユーザーがセッションを終了すると自動的にクリアされるので、セッション・データを手動で削除する必要はありません。

プロシージャはトラステッド・プロシージャであることに注意してください。ユーザーが自分独自のアプリケーション・コンテキスト属性値を設定できないように設計されています。ユーザーはプロシージャを実行しますが、プロシージャがアプリケーション・コンテキスト値を設定します、ユーザーではありません。

関連項目:

- SYS_CONTEXTファンクションの詳細は、『Oracle Database SQL言語リファレンス』を参照してください。
- データベース・セッション・ベースのアプリケーション・コンテキストの作成方法は、例: データベース・セッション・ベースのアプリケーション・コンテキストの作成と使用を参照してください

親トピック: データベース・セッション・ベースのアプリケーション・コンテキストを設定するためのパッケージの作成

13.3.4.2 SYS_CONTEXTファンクションを使用したセッション情報の取得

SYS_CONTEXTファンクションを使用して、アプリケーション・コンテキストのセッション情報を取得できます。

SYS_CONTEXT関数には、ログインしたユーザーのカレント・セッションを表すデフォルト・ネームスペースUSERENVがあります。SYS_CONTEXTを使用すると、ユーザー・ホスト・コンピュータID、ホストIPアドレス、オペレーティング・システム・ユーザー名など、ユーザーに関する様々な種類のセッション・ベースの情報を取得できます。セッション・データを設定ではなく、取得するにはUSERENVのみを使用することに注意してください。事前定義の属性は、『Oracle Database SQL言語リファレンス』のPL/SQLファンクションの説明でリストされています。

- セッション情報を取得するには、ネームスペースとパラメータを設定し、オプションでSYS_CONTEXTファンクションの長さの値を設定します。

  たとえば:

  ```
  SYS_CONTEXT ('USERENV','HOST')
  ```

PL/SQLファンクションのSYS_CONTEXTの構文は、次のとおりです。

```
SYS_CONTEXT ('namespace','parameter'[,length])
```

詳細は、次のとおりです。

- namespaceはアプリケーション・コンテキストの名前です。文字列、または文字列として評価される式を指定できます。

SYS_CONTEXTファンクションは、現時点でコンテキストのネームスペースに関連付けられているパラメータの値を戻します。ネームスペース内のパラメータの値がすでに設定されている場合は、SET_CONTEXTによってこの値が上書きされます。

- parameterは、namespaceアプリケーション・コンテキスト内のパラメータです。この値には、文字列または式を指定できます。
- lengthは戻り型のデフォルト最大サイズ(256バイト)ですが、最大で4000バイトの値を指定して長さを変更できます。NUMBERデータ型の値を入力するか、NUMBERに明示的に変換できる値を入力します。SYS_CONTEXT戻り型のデータ型はVARCHAR2です。この設定はオプションです。

ノート:

USERENV アプリケーション・コンテキスト・ネームスペースは、Oracle Database の以前のリリースで提供されていた USERENV 関数にかわる機能です。

親トピック: データベース・セッション・ベースのアプリケーション・コンテキストを設定するためのパッケージの作成

13.3.4.3 SYS_CONTEXT設定の確認

DUAL表に格納されるSYS_CONTEXT設定を確認できます。

DUAL表はデータ・ディクショナリ内の小さい表で、既知の結果を保証するためにOracle Databaseおよびユーザーが記述したプログラムから参照できます。この表には、DUMMYという列と、値Xが格納されている行があります。

- SYS_CONTEXT設定を確認するには、DUAL表に対してSELECT SQL文を発行します。

たとえば、ログインしたホスト・コンピュータを確認する場合、EMP_USERSの下のSHOBEEN_PCホスト・コンピュータにログインしたと想定しています。

```
SELECT SYS_CONTEXT ('USERENV', 'HOST') FROM DUAL;
SYS_CONTEXT(USERENV,HOST)
-------------------------
EMP_USERS¥SHOBEEEN_PC
```

親トピック: データベース・セッション・ベースのアプリケーション・コンテキストを設定するためのパッケージの作成

13.3.4.4 SYS_CONTEXTでの動的SQL

指定した問合せを実行する間にポリシーの変更が予想されるセッション中は、その問合せに動的SQLを使用する必要があります。

静的SQLと動的SQLでは文の解析方法が異なるため、必ず動的SQLを使用してください。

- 静的SQL文はコンパイル時に解析されます。パフォーマンス上の理由から実行時には再解析されません。
- 動的SQL文は、実行されるたびに解析されます。

SQL文のコンパイル時にはポリシーAを規定し、その後、ポリシーBに変更して文を実行する場合を考えてみます。静的SQLでは、ポリシーAが規定されたままです。文はコンパイル時に解析されますが、実行時には再解析されません。動的SQLでは、文は実行時に解析されるため、ポリシーBへの切替えが有効となります。

たとえば、次のポリシーを考えてみます。

```
EMPLOYEE_NAME = SYS_CONTEXT ('USERENV', 'SESSION_USER')
```

ポリシーEMPLOYEE_NAMEは、データベース・ユーザー名と一致しています。Oracle Virtual Private DatabaseではSQL述語の形で表され、述語はポリシーとみなされます。述語が変更された場合、正しい結果を生成するために文を再度解析する必要があります。

関連トピック

- ファイングレイン・アクセス・コントロールのポリシー関数に対する自動再解析

親トピック: データベース・セッション・ベースのアプリケーション・コンテキストを設定するためのパッケージの作成

13.3.4.5 パラレル問合せでのSYS_CONTEXT

パラレル問合せに埋め込まれているSQL関数内でSYS_CONTEXTを使用すると、この関数にアプリケーション・コンテキストが挿入されます。

ユーザーIDを5に設定する、SQL文内のユーザー定義関数を考えてみます。

```
CREATE FUNCTION set_id
 RETURN NUMBER IS
BEGIN
 IF SYS_CONTEXT ('hr', 'id') = 5
   THEN RETURN 1; ELSE RETURN 2;
 END IF;
END;
```

次の文を考えてみます。

```
SELECT * FROM emp WHERE set_id( ) = 1;
```

この文をパラレル問合せとして実行すると、アプリケーション・コンテキスト情報を含むユーザー・セッションはパラレル実行サーバー(問合せ子プロセス)に伝播されます。

親トピック: データベース・セッション・ベースのアプリケーション・コンテキストを設定するためのパッケージの作成

13.3.4.6 データベース・リンクでのSYS_CONTEXT

SYS_CONTEXTファンクションはデータベース・リンクと一緒に使用できます。

ユーザー・セッション内のSQL文にデータベース・リンクが含まれている場合は、そのデータベース・リンクのホスト・コンピュータでSYS_CONTEXT関数が実行され、そのホスト・コンピュータにあるコンテキスト情報が取得されます。

リモートのPL/SQLプロシージャ・コールがデータベース・リンクで実行されている場合は、そのプロシージャ内部のすべてのSYS_CONTEXT関数が、そのリンクの宛先データベースで実行されます。

この場合、データベース・リンクの宛先サイトで使用できるのは、外部で初期化されたアプリケーション・コンテキストのみです。セキュリティ上の理由から、データベース・リンクの開始サイトから宛先サイトに伝播されるのは、外部で初期化されたアプリケーション・コンテキストのみです。

親トピック: データベース・セッション・ベースのアプリケーション・コンテキストを設定するためのパッケージの作成

13.3.4.7 セッション情報を設定するためのDBMS_SESSION.SET_CONTEXT

SYS_CONTEXTでユーザーのセッション・データを取得した後、ユーザー・セッションからアプリケーション・コンテキスト値を設定できます。

コンテキスト値を設定するには、DBMS_SESSION.SET_CONTEXTプロシージャを使用します。DBMS_SESSION PL/SQLパッケージに対するEXECUTE権限を所有している必要があります。

DBMS_SESSION.SET_CONTEXTの構文は、次のとおりです。

```
DBMS_SESSION.SET_CONTEXT (
   namespace VARCHAR2,
   attribute VARCHAR2,
   value     VARCHAR2,
   username  VARCHAR2,
   client_id VARCHAR2);
```

詳細は、次のとおりです。

- namespaceは、設定するアプリケーション・コンテキストのネームスペース(30バイトに制限)です。たとえば、custno_ctxという名前のネームスペースを使用していた場合は、次のように指定します。

  ```
  namespace => 'custno_ctx',
  ```

- attributeは、設定するアプリケーション・コンテキストの属性(30バイトに制限)です。たとえばcustno_ctxネームスペースのctx_attrib属性を作成する場合は、次のようになります。

  ```
  attribute => 'ctx_attrib',
  ```

- valueは、設定するアプリケーション・コンテキストの値(4000バイトに制限)です。通常、これはSYS_CONTEXT関数で取得されて変数に格納された値です。たとえば:

  ```
  value => ctx_value,
  ```

- usernameは、アプリケーション・コンテキストのデータベース・ユーザー名属性です。デフォルトはNULLで、すべてのユーザーにセッションへのアクセスを許可します。データベース・セッション・ベースのアプリケーション・コンテキストでは、この設定は省略され、デフォルトのNULLが使用されます。この設定はオプションです。

  usernameおよびclient_idパラメータは、グローバルにアクセスされるアプリケーション・コンテキストに対して使用されます。詳細は、DBMS_SESSION.SET_CONTEXTのusernameおよびclient_idパラメータを参照してください。

- client_idは、アプリケーション・コンテキストのアプリケーション固有のclient_id属性(最大64バイト)です。デフォルトはNULLで、クライアントIDが指定されていないことを意味します。データベース・セッション・ベースのアプリケーション・コンテキストでは、この設定は省略され、デフォルトのNULLが使用されます。

関連項目:

- ユーザー・セッション情報を取得するパッケージを作成し、その情報に基づいてアプリケーション・コンテキストを設定する方法については、例: データベース・セッション・ベースのアプリケーション・コンテキストの作成と使用を参照してください
- DBMS_SESSION.SET_CONTEXTプロシージャの詳細は、『Oracle Database PL/SQLパッケージおよびタイプ・リファレンス』を参照してください。
- DBMS_SESSIONパッケージの詳細は、『Oracle Database PL/SQLパッケージおよびタイプ・リファレンス』を参照してください。

親トピック: データベース・セッション・ベースのアプリケーション・コンテキストを設定するためのパッケージの作成

13.3.4.8 例: アプリケーション・コンテキストの値を作成する単純なプロシージャ

プロシージャでDBMS_SESSION.SET_CONTEXT文を使用して、アプリケーション・コンテキストの値を設定できます。

例13-1に、empno_ctxアプリケーション・コンテキストの属性を作成する単純なプロシージャの作成方法を示します。

例13-1 アプリケーション・コンテキストの値を作成する単純なプロシージャ

```
CREATE OR REPLACE PROCEDURE set_empno_ctx_proc(
  emp_value IN VARCHAR2)
 IS
 BEGIN
  DBMS_SESSION.SET_CONTEXT('empno_ctx', 'empno_attrib', emp_value);
 END;
/
```

この例では、次のようになります。

- `emp_value IN VARCHAR2`は`emp_value`を入力パラメータとして使用します。このパラメータは、アプリケーション・コンテキスト属性`empno_attrib`に関連付けられている値を指定します。制限は4000バイトです。
- `DBMS_SESSION.SET_CONTEXT('empno_ctx', 'empno_attrib', emp_value)`は、`DBMS_SESSION.SET_CONTEXT`プロシージャを次のように使用して、アプリケーション・コンテキストの値を設定します。
  - `'empno_ctx'`は、アプリケーション・コンテキストのネームスペースを示します。ネームスペース名は一重引用符で囲みます。
  - `'empno_attrib'`は、アプリケーション・コンテキストのネームスペースに関連付ける属性を作成します。
  - `emp_value`は、`empno_attrib`属性の値を指定します。ここでは、`emp_value`パラメータを参照しています。

これで、`set_empno_ctx_proc`プロシージャを実行して、アプリケーション・コンテキストを設定できます。

```
EXECUTE set_empno_ctx_proc ('42783');
```

(実際には、アプリケーション・コンテキストの値はプロシージャ自体に設定することになるため、そのプロシージャがトラステッド・プロシージャとなります。この例は、データが設定される様子を示すことのみを目的に記載しています。)

アプリケーション・コンテキストの設定をチェックするには、次のSELECT文を実行します。

```
SELECT SYS_CONTEXT ('empno_ctx', 'empno_attrib') empno_attrib FROM DUAL;
EMPNO_ATTRIB
--------------
42783
```

`SESSION_CONTEXT`データ・ディクショナリ・ビューを問い合せると、データベース・インスタンスのカレント・セッションにあるすべてのアプリケーション・コンテキスト設定を検索することもできます。たとえば:

```
SELECT * FROM SESSION_CONTEXT;
NAMESPACE                ATTRIBUTE           VALUE
-------------------------------------------------
EMPNO_CTX                EMP_ID              42783
```

親トピック: データベース・セッション・ベースのアプリケーション・コンテキストを設定するためのパッケージの作成

13.3.5 データベース・セッションのアプリケーション・コンテキスト・パッケージを実行するログオン・トリガー

データベース・インスタンスにログインした後、ユーザーはデータベース・セッションのアプリケーション・コンテキスト・パッケージを実行する必要があります。

これを自動的に処理するようにログイン・トリガーを作成できます。パッケージを実行するためのEXECUTE権限をユーザーに付与

する必要はありません。

次のことに注意してください。

- ログイン・トリガーによって呼び出されたPL/SQLパッケージ・プロシージャに未処理例外があるか(たとえばセキュリティ・チェックに失敗したために)なんらかの例外を発生すると、ログイン・トリガーは失敗します。ログイン・トリガーに失敗すると、ログインは失敗し、つまりユーザーはデータベースにログインする権限が拒否されます。
- ログイン・トリガーがパフォーマンスに影響を与える可能性があります。また、ログイン・トリガーは、最初にサンプル・スキーマ・ユーザーでテストしてから、データベース用に作成してください。これによって、エラーがあった場合は簡単に修正できます。
- 会計帳簿の変更や職階の変更が頻繁にある場合は注意が必要です。この場合、新しい属性値はすぐに選択できない可能性があるため、カーソルの再解析を強制実行して、新しい属性値を選択する必要があります。

ノート:

ユーザー・コンテキスト(EMPNO、GROUP、MANAGER などの情報)は、ユーザーがデータにアクセスする前に設定されるため、ログイン・トリガーを使用できます。

親トピック: データベース・セッション・ベースのアプリケーション・コンテキストの使用

13.3.6 例: 単純なログイン・トリガーの作成

CREATE TRIGGER文で簡単なログイン・トリガーを作成できます。

例13-2に、PL/SQLプロシージャを実行する単純なログイン・トリガーを示します。

例13-2 単純なログイン・トリガーの作成

```
CREATE OR REPLACE TRIGGER set_empno_ctx_trig AFTER LOGON ON DATABASE
 BEGIN
 sec_mgr.set_empno_ctx_proc;
 END;
```

親トピック: データベース・セッション・ベースのアプリケーション・コンテキストの使用

13.3.7 例: 本番環境用のログイン・トリガーの作成

CREATE TRIGGER文で、本番環境のログイン・トリガーを作成できます。

例13-3は、WHEN OTHERS例外を使用するログイン・トリガーの作成方法を示しています。一方、PL/SQLロジックにエラーが発生して未処理例外を引き起こす場合は、データベースへのすべての接続がブロックされます。

この例は、セキュリティ管理者のスキーマの表にエラーを書き込むWHEN OTHERS例外を示しています。本番環境では、こちらのほうが、出力をユーザー・セッションへ送信することでセキュリティ攻撃を受けやすくなるよりも安全です。

例13-3 本番環境用のログイン・トリガーの作成

```
CREATE OR REPLACE TRIGGER set_empno_ctx_trig AFTER LOGON ON DATABASE
 BEGIN
 sec_mgr.set_empno_ctx_proc;
 EXCEPTION
  WHEN OTHERS THEN
        v_code := SQLCODE;
        v_errm := SUBSTR(SQLERRM, 1 , 64);
```

```
       -- Invoke another procedure,
       -- declared with PRAGMA AUTONOMOUS_TRANSACTION,
       -- to insert information about errors.
  INSERT INTO sec_mgr.errors VALUES (v_code, v_errm, SYSTIMESTAMP);
 END;
/
```

親トピック: データベース・セッション・ベースのアプリケーション・コンテキストの使用

13.3.8 例: 開発環境用のログイン・トリガーの作成

CREATE TRIGGER文で、開発環境のログイン・トリガーを作成できます。

例13-4に、同じログイン・トリガーを開発環境用に作成する方法を示します。この場合、エラーをデバッグのためにユーザー・セッションに出力できます。

例13-4 開発環境用のログイン・トリガーの作成

```
CREATE TRIGGER set_empno_ctx_trig
 AFTER LOGON ON DATABASE
 BEGIN
 sysadmin_ctx.set_empno_ctx_pkg.set_empno;
 EXCEPTION
  WHEN OTHERS THEN
   RAISE_APPLICATION_ERROR(
    -20000, 'Trigger sysadmin_ctx.set_empno_ctx_trig violation. Login denied.');
 END;
/
```

親トピック: データベース・セッション・ベースのアプリケーション・コンテキストの使用

13.3.9 例: データベース・セッション・ベースのアプリケーション・コンテキストの作成と使用

このチュートリアルでは、データベースへのログインを試みるユーザーのIDをチェックするアプリケーション・コンテキストの作成方法を示します。

- ステップ1: ユーザー・アカウントの作成とユーザーSCOTTがアクティブであることの確認
  このチュートリアルを開始するには、必要なデータベース・アカウントを作成し、SCOTTユーザー・アカウントがアクティブであることを確認する必要があります。
- ステップ2: データベース・セッション・ベースのアプリケーション・コンテキストの作成
  sysadmin_ctxユーザーとして、データベース・セッション・ベースのアプリケーション・コンテキストを作成します。
- ステップ3: セッション・データを取得してアプリケーション・コンテキストを設定するパッケージの作成
  次に、セッション・データを取得してアプリケーション・コンテキストを設定するPL/SQLパッケージを作成する必要があります。
- ステップ4: パッケージに対するログイン・トリガーの作成
  ログイン・トリガーは、ユーザーがログインすると実行されます。
- ステップ5: アプリケーション・コンテキストのテスト
  すべてのコンポーネントの準備ができると、アプリケーション・コンテキストをテストできます。
- ステップ6: このチュートリアルのコンポーネントの削除
  このチュートリアルのコンポーネントが不要になった場合、それらを削除できます。

親トピック: データベース・セッション・ベースのアプリケーション・コンテキストの使用

13.3.9.1 ステップ1: ユーザー・アカウントの作成とユーザーSCOTTがアクティブであることの確認

このチュートリアルを開始するには、必要なデータベース・アカウントを作成し、SCOTTユーザー・アカウントがアクティブであることを確認する必要があります。

1. ユーザーSYSとしてログインし、SYSDBA管理権限を使用して接続します。

```
sqlplus sys as sysdba
Enter password: password
```

2. マルチテナント環境で、適切なPDBに接続します。

   たとえば:

```
CONNECT SYS@hrpdb AS SYSDBA
Enter password: password
```

   使用可能なPDBを検索するには、show pdbsコマンドを実行します。現在のPDBを確認するには、show con_nameコマンドを実行します。

3. sysadmin_ctxローカル・ユーザー・アカウントを作成します。このアカウントは、データベース・セッション・ベースのアプリケーション・コンテキストを管理します。

```
CREATE USER sysadmin_ctx IDENTIFIED BY password;
GRANT CREATE SESSION, CREATE ANY CONTEXT, CREATE PROCEDURE, CREATE TRIGGER,
ADMINISTER DATABASE TRIGGER TO sysadmin_ctx;
GRANT READ ON HR.EMPLOYEES TO sysadmin_ctx;
GRANT EXECUTE ON DBMS_SESSION TO sysadmin_ctx;
```

   「パスワードの最低要件」のガイドラインに従って、passwordを安全なパスワードに置き換えます。

4. Lisa Ozerに対して、次のユーザー・アカウントを作成します。HR.EMPLOYEES表には、lozerというLisa Ozerの電子メール・アカウントがあります。

```
GRANT CREATE SESSION TO LOZER IDENTIFIED BY password;
```

   passwordを安全なパスワードに置き換えます。

5. このチュートリアルでは、サンプル・ユーザーSCOTTも使用するため、DBA_USERSデータ・ディクショナリ・ビューを問い合せて、SCOTTのアカウント・ステータスがOPENになっていることを確認します。

```
SELECT USERNAME, ACCOUNT_STATUS FROM DBA_USERS WHERE USERNAME = 'SCOTT';
```

   DBA_USERSビューに、ユーザーSCOTTがロックされて期限切れになっていると表示された場合は、次の文を入力して、SCOTTアカウントのロックを解除し、新しいパスワードを作成します。

```
ALTER USER SCOTT ACCOUNT UNLOCK IDENTIFIED BY password;
```

   安全なパスワードを入力します。セキュリティを向上させるため、以前のリリースのOracle Databaseと同じパスワードをSCOTTアカウントに指定しないでください。パスワードを作成するための最低要件は、パスワードの最低要件を参照してください。

親トピック: 例: データベース・セッション・ベースのアプリケーション・コンテキストの作成と使用

13.3.9.2 ステップ2: データベース・セッション・ベースのアプリケーション・コンテキストの作成

sysadmin_ctxユーザーとして、データベース・セッション・ベースのアプリケーション・コンテキストを作成します。

1. sysadmin_ctxでSQL*Plusにログインします。

```
CONNECT sysadmin_ctx -- Or, CONNECT sysadmin_ctx@hrpdb
Enter password: password
```

2. 次の文を使用してアプリケーション・コンテキストを作成します。

```
CREATE CONTEXT empno_ctx USING set_empno_ctx_pkg;
```

ユーザーsysadmin_ctxがこのアプリケーション・コンテキストを作成した場合でも、SYSスキーマがこのコンテキストを所有することに注意してください。

親トピック: 例: データベース・セッション・ベースのアプリケーション・コンテキストの作成と使用

13.3.9.3 ステップ3: セッション・データを取得してアプリケーション・コンテキストを設定するパッケージの作成

次に、セッション・データを取得してアプリケーション・コンテキストを設定するPL/SQLパッケージを作成する必要があります。

- パッケージを作成するには、CREATE OR REPLACE PACKAGE文を使用します。

例13-5に、セッション・データを取得してアプリケーション・コンテキストを設定するために必要なパッケージの作成方法を示します。パッケージを作成する前に、ユーザーsysadmin_ctxでログインしていることを確認してください。(最初の行のCREATE OR REPLACEの前にカーソルを置くことで、このテキストをコピーして貼り付けることができます。)

例13-5 セッション・データを取得してデータベース・セッション・コンテキストを設定するためのパッケージ

```
CREATE OR REPLACE PACKAGE set_empno_ctx_pkg IS
   PROCEDURE set_empno;
 END;
 /
 CREATE OR REPLACE PACKAGE BODY set_empno_ctx_pkg IS
   PROCEDURE set_empno
   IS
    emp_id HR.EMPLOYEES.EMPLOYEE_ID%TYPE;
   BEGIN
    SELECT EMPLOYEE_ID INTO emp_id FROM HR.EMPLOYEES
       WHERE email = SYS_CONTEXT('USERENV', 'SESSION_USER');
    DBMS_SESSION.SET_CONTEXT('empno_ctx', 'employee_id', emp_id);
   EXCEPTION
    WHEN NO_DATA_FOUND THEN NULL;
  END;
 END;
/
```

このパッケージでは、次の処理を実行するset_empnoというプロシージャが作成されます。

- emp_id HR.EMPLOYEES.EMPLOYEE_ID%TYPEは、ログインするユーザーの従業員IDを格納する変数emp_idを宣言します。HR.EMPLOYEESのEMPLOYEE_ID列と同じデータ型を使用します。
- SELECT EMPLOYEE_ID INTO emp_id FROM HR.EMPLOYEESは、SELECT文を実行し、HR.EMPLOYEES表のemployee_id列データに格納されている従業員IDをemp_id変数にコピーします。
- WHERE email = SYS_CONTEXT('USERENV', 'SESSION_USER')は、WHERE句を使用して、セッション・ユーザーの電子メール・アカウントと一致するすべての従業員IDを検索します。SYS_CONTEXT関数は、事前定義のUSERENVコンテキストを使用してユーザー・セッションID (これはemail列データと同じです)を取得します。たとえば、Lisa OzerのユーザーIDと電子メール・アドレスはどちらも同じでlozerです。
- DBMS_SESSION.SET_CONTEXT('empno_ctx', 'employee_id', emp_id)は、DBMS_SESSION.SET_CONTEXTプロシージャを使用して、アプリケーション・コンテキストを設定します。

- 'empno_ctx': アプリケーション・コンテキストempno_ctxをコールします。empno_ctxは一重引用符で囲みます。
- 'employee_id': 属性にemployee_idという名前を指定して、empno_ctxアプリケーション・コンテキストの名前と値のペアの属性値を作成します。employee_idは一重引用符で囲みます。
- emp_id: employee_id属性の値をemp_id変数に格納された値に設定します。

要約すると、set_empno_ctx_pkg.set_empnoプロシージャは、「ユーザーのセッションIDを取得して、このユーザーIDをHR.EMPLOYEES表にリストされたユーザーの従業員IDおよび電子メール・アドレスと照合する」プロシージャです。

- EXCEPTION ... WHEN_NO_DATA_FOUNDは、WHEN NO_DATA_FOUNDシステム例外を追加して、SELECT文によって発生した可能性のあるno data foundエラーを捕捉します。この例外がないと、パッケージとログイン・トリガーは正常に機能し、必要に応じてアプリケーション・コンテキストが設定されますが、HR.EMPLOYEES表にリストされたユーザーを除き、システム管理者以外のユーザーはデータベースにログインできなくなります。他のユーザーは、有効なデータベース・ユーザーであればデータベースにログインできます。アプリケーション・コンテキスト情報が設定された後は、特定のアプリケーションへのユーザー・アクセスを制御する手段として、このセッション情報を使用できます。

親トピック: 例: データベース・セッション・ベースのアプリケーション・コンテキストの作成と使用

13.3.9.4 ステップ4: パッケージに対するログイン・トリガーの作成

ログオン・トリガーは、ユーザーがログインすると実行されます。

- sysadmin_ctxユーザーとして、set_empno_ctx_pkg.set_empnoパッケージ・プロシージャのログオン・トリガーを作成します。

```
CREATE TRIGGER set_empno_ctx_trig AFTER LOGON ON DATABASE
 BEGIN
 sysadmin_ctx.set_empno_ctx_pkg.set_empno;
 END;
/
```

親トピック: 例: データベース・セッション・ベースのアプリケーション・コンテキストの作成と使用

13.3.9.5 ステップ5: アプリケーション・コンテキストのテスト

すべてのコンポーネントの準備ができると、アプリケーション・コンテキストをテストできます。

1. ユーザーlozerでログインします。

   ```
   CONNECT lozer -- Or, CONNECT lozer@hrpdb
   Enter password: password
   ```

   ユーザーlozerがログインすると、このユーザーの従業員IDがempno_ctxアプリケーション・コンテキストによって収集されます。これは次のようにして確認できます。

   ```
   SELECT SYS_CONTEXT('empno_ctx', 'employee_id') emp_id FROM DUAL;
   ```

   次の出力が表示されます。

   ```
   EMP_ID
   ----------------------------------------------------------
   168
   ```

2. ユーザーSCOTTでログインします。

```
CONNECT SCOTT -- Or, CONNECT SCOTT@hrpdb
Enter password: password
```

ユーザーSCOTTはHR.EMPLOYEES表に従業員としてリストされていないため、empno_ctxアプリケーション・コンテキストではこのユーザーの従業員IDを収集できません。

```
SELECT SYS_CONTEXT('empno_ctx', 'employee_id') emp_id FROM DUAL;
```

次の出力が表示されます。

```
EMP_ID
------------------------------------------------------------
```

これ以降、アプリケーションではユーザー・セッション情報を使用してユーザーがデータベース内で許可されるアクセス数を判別できます。そのためには、Oracle Virtual Private Databaseを使用できます。

関連トピック

- Oracle Virtual Private Databaseを使用したデータ・アクセスの制御

親トピック: 例: データベース・セッション・ベースのアプリケーション・コンテキストの作成と使用

13.3.9.6 ステップ6: このチュートリアルのコンポーネントの削除

このチュートリアルのコンポーネントが不要になった場合、それらを削除できます。

1. SYSDBA管理権限を持つSYSとして接続します。

```
CONNECT SYS AS SYSDBA -- Or, CONNECT SYS@hrpdb AS SYSDBA
Enter password: password
```

2. ユーザーsysadmin_ctxとlozerを削除します。

```
DROP USER sysadmin_ctx CASCADE;
DROP USER lozer;
```

3. アプリケーション・コンテキストを削除します。

```
DROP CONTEXT empno_ctx;
```

このアプリケーション・コンテキストはsysadmin_ctxが作成しましたが、SYSスキーマの所有となっていることに注意してください。

4. 他のユーザーがSCOTTを使用しない場合、このアカウントはロックして期限切れにできます。

```
ALTER USER SCOTT PASSWORD EXPIRE ACCOUNT LOCK;
```

親トピック: 例: データベース・セッション・ベースのアプリケーション・コンテキストの作成と使用

13.3.10 データベース・セッション・ベースのアプリケーション・コンテキストの外部での初期化

データベース・セッション・ベースのアプリケーション・コンテキストを外部で初期化すると、アプリケーション・コンテキストがユーザー・グローバル領域(UGA)に格納されるため、パフォーマンスが向上します。

- データベース・セッション・ベースのアプリケーション・コンテキストの外部による初期化について
  セッション・ベースのアプリケーション・コンテキストを外部で初期化するには、特別な種類のネームスペースを使用する必要があります。

- ユーザーからのデフォルト値
  Oracle Databaseでは、アプリケーションのユーザーからデフォルト値を取得して使用できます。
- 他の外部リソースからの値
  アプリケーション・コンテキストによって、外部リソースを介して属性と値の初期化を受け入れることができます。
- 例: 外部化されたデータベース・セッション・ベースのアプリケーション・コンテキストの作成
  CREATE CONTEXT SQL文で、外部化されたデータベース・セッション・ベースのアプリケーション・コンテキストを作成できます。
- 中間層サーバーからのアプリケーション・コンテキスト値の初期化
  中間層サーバーは、データベース・ユーザーのかわりにアプリケーション・コンテキスト値を初期化できます。

親トピック: データベース・セッション・ベースのアプリケーション・コンテキストの使用

13.3.10.1 データベース・セッション・ベースのアプリケーション・コンテキストの外部による初期化について

セッション・ベースのアプリケーション・コンテキストを外部で初期化するには、特別な種類のネームスペースを使用する必要があります。

このネームスペースは外部リソースからの属性値の初期化を受け入れ、ローカル・ユーザー・セッションに格納する必要があります。

アプリケーション・コンテキストは外部で初期化することによりUGAに格納されて、属性をセッションから別のセッションへ自動伝播できるようになるため、パフォーマンスが向上します。接続ユーザー・データベース・リンクをサポートするのは、OCIベースの外部ソースから初期化されたアプリケーション・コンテキストのみです。

親トピック: データベース・セッション・ベースのアプリケーション・コンテキストの外部での初期化

13.3.10.2 ユーザーからのデフォルト値

Oracle Databaseでは、アプリケーションのユーザーからデフォルト値を取得して使用できます。

ユーザーからデフォルトを取得することが必要な場合があります。最初、これらのデフォルト値はヒントまたはプリファレンスとして機能し、検証後にトラステッド・コンテキストになります。同様に、クライアントでは、一部のデフォルト値を初期化してから、ログイン・イベント・トリガーまたはアプリケーションを使用して値の検証を行うと便利です。

ジョブ・キューに関しては、ジョブ発行ルーチンで、ジョブの発行時に設定されたコンテキストを記録し、バッチ・ジョブの実行時にそのコンテキストをリストアします。コンテキストの整合性を保持するために、ジョブ・キューは、コンテキストを設定する特定のPL/SQLパッケージを回避できません。一方、外部で初期化されたアプリケーション・コンテキストは、コンテキスト値の初期化をジョブ・キュー・プロセスから受け入れます。

リモート・セッションへのコンテキストの自動伝播によって、セキュリティ上の問題が発生する場合があります。開発者または管理者は、ユーザーのログイン時にログイン・トリガーを使用してコンテキストを再設定することで、特定のPL/SQLプロシージャ以外のリソースからデフォルト値を取得するコンテキストを効果的に処理できます。

親トピック: データベース・セッション・ベースのアプリケーション・コンテキストの外部での初期化

13.3.10.3 他の外部リソースからの値

アプリケーション・コンテキストによって、外部リソースを介して属性と値の初期化を受け入れることができます。

このような外部リソースには、Oracle Call Interface (OCI)インタフェース、ジョブ・キュー・プロセスまたはデータベース・リンクなどが含まれます。

外部で初期化されたアプリケーション・コンテキストでは、次の機能が提供されます。

- リモート・セッションの場合は、外部で初期化されたアプリケーション・コンテキスト・ネームスペースにあるコンテキスト値を

自動的に伝播します。

- ジョブ・キューの場合は、外部で初期化されたアプリケーション・コンテキスト・ネームスペースにあるコンテキスト値をリストアします。
- OCIインタフェースの場合は、外部で初期化されたアプリケーション・コンテキスト・ネームスペースにあるコンテキスト値を初期化するメカニズムを提供します。

Oracle Call Interfaceを使用しているクライアント・プログラムであればこのタイプのネームスペースの初期化は可能ですが、ログイン・イベント・トリガーを使用して値を検証できます。属性の値を解釈して信頼するかどうかはアプリケーションによって異なります。

親トピック: データベース・セッション・ベースのアプリケーション・コンテキストの外部での初期化

13.3.10.4 例: 外部化されたデータベース・セッション・ベースのアプリケーション・コンテキストの作成

CREATE CONTEXT SQL文で、外部化されたデータベース・セッション・ベースのアプリケーション・コンテキストを作成できます。

例13-6に、外部ソースから値を取得するデータベース・セッション・ベースのアプリケーション・コンテキストを作成する方法を示します。

例13-6 外部化されたデータベース・セッション・ベースのアプリケーション・コンテキストの作成

```
CREATE CONTEXT ext_ctx USING ext_ctx_pkg INITIALIZED EXTERNALLY;
```

親トピック: データベース・セッション・ベースのアプリケーション・コンテキストの外部での初期化

13.3.10.5 中間層サーバーからのアプリケーション・コンテキスト値の初期化

中間層サーバーは、データベース・ユーザーのかわりにアプリケーション・コンテキスト値を初期化できます。

このプロセスでは、コンテキスト属性が初期化時にリモート・セッションに対して伝播され、ネームスペースが外部で初期化されている場合、リモート・データベースはこれらの値を受け入れます。

たとえば、OCIまたはJDBC/OCI経由の軽量ユーザー・セッションを作成する3層アプリケーションの場合は、USERENVのPROXY_USER属性にアクセスできます。この属性を使用すると、ユーザー・セッションが中間層アプリケーションによって作成されたかどうかを判断できます。ユーザーがデータへのアクセスを許可されるのは、そのユーザーがプロキシ化されている接続に対してのみです。ユーザーがデータベースに直接接続している場合、そのユーザーはどのデータにもアクセスできません。

Oracle Virtual Private Database内のUSERENVネームスペースからPROXY_USER属性を使用すると、ユーザーによるデータへのアクセスを特定の中間層アプリケーションを経由した場合のみに制限できます。他の方法では、セキュア・アプリケーション・ロールを作成し、ユーザーは特定のプロキシ経由でのみデータベースにアクセスするというポリシーを規定できます。

関連項目:

- プロキシ認証の詳細およびUSERENV属性CLIENT_IDENTIFIERを使用して複数層にわたってユーザー識別情報を保持する方法は、複数層環境でのユーザー識別情報の保持を参照してください
- セキュア・アプリケーション・ロールを使用して特定のプロキシを経由するようにポリシーを実施する方法は、プロキシ認証に対する中間層サーバーの使用を参照してください
- Oracle Call Interfaceプログラマーズ・ガイド

親トピック: データベース・セッション・ベースのアプリケーション・コンテキストの外部での初期化

13.3.11 データベース・セッション・ベースのアプリケーション・コンテキストのグローバルな初期化

データベース・セッション・ベースのアプリケーションが集中格納場所に格納されている場合、そのアプリケーションはLDAPディレクトリ全域から使用できます。

- データベース・セッション・ベースのアプリケーション・コンテキストのグローバルな初期化について
  ユーザーのデータベース・セッション・ベースのアプリケーション・コンテキストは、集中格納場所を使用して格納できます。
- LDAPでのデータベース・セッション・ベースのアプリケーション・コンテキストの使用
  グローバルに初期化されたアプリケーション・コンテキストは、拡張可能で効率的な標準ディレクトリ・アクセス・プロトコルであるLDAPを使用します。
- グローバルに初期化されたデータベース・セッション・ベースのアプリケーション・コンテキストの動作
  グローバルに初期化されたセキュア・アプリケーションを使用するには、エンタープライズ・ユーザー・セキュリティを最初に構成する必要があります。
- データベース・セッション・ベースのアプリケーション・コンテキストのグローバルな初期化
  ユーザーの初期アプリケーション・コンテキスト(部門名や職位など)は、LDAPディレクトリに設定および格納できます。

親トピック: データベース・セッション・ベースのアプリケーション・コンテキストの使用

13.3.11.1 データベース・セッション・ベースのアプリケーション・コンテキストのグローバルな初期化について

ユーザーのデータベース・セッション・ベースのアプリケーション・コンテキストは、集中格納場所を使用して格納できます。

集中格納場所を使用することでアプリケーションでは、初期化中にユーザーの識別情報に基づいてユーザー・コンテキストを設定できます。

特に、この機能では、Oracle Label Securityのラベルと権限がサポートされます。アプリケーション・コンテキストをグローバルに初期化すると、多数のユーザーとデータベースのコンテキストを簡単に管理できます。

たとえば、多くの組織では、LDAPベースのディレクトリでユーザー情報を集中管理する必要があります。エンタープライズ・ユーザー・セキュリティは、Oracle Internet Directoryでのユーザーと認可の集中管理をサポートします。ただし、Oracle Virtual Private Databaseの規定に使用するために、アプリケーションではLightweight Directory Access Protocol(LDAP)から追加の属性(ユーザーの職位、組織、物理的な位置など)を取得することが必要な場合があります。これらのタイプの属性は、アプリケーション・コンテキストをグローバルに初期化することで取得できます。

親トピック: データベース・セッション・ベースのアプリケーション・コンテキストのグローバルな初期化

13.3.11.2 LDAPでのデータベース・セッション・ベースのアプリケーション・コンテキストの使用

グローバルに初期化されたアプリケーション・コンテキストは、拡張可能で効率的な標準ディレクトリ・アクセス・プロトコルであるLDAPを使用します。

LDAPディレクトリには、このアプリケーションが割り当てられているユーザーのリストが格納されます。Oracle Databaseでは、ディレクトリ・サービス(通常はOracle Internet Directory)を使用して、エンタープライズ・ユーザーを認証および認可します。

ノート:

Microsoft Active Directory や Sun Microsystems の SunONE などのサード・パーティ製ディレクトリも

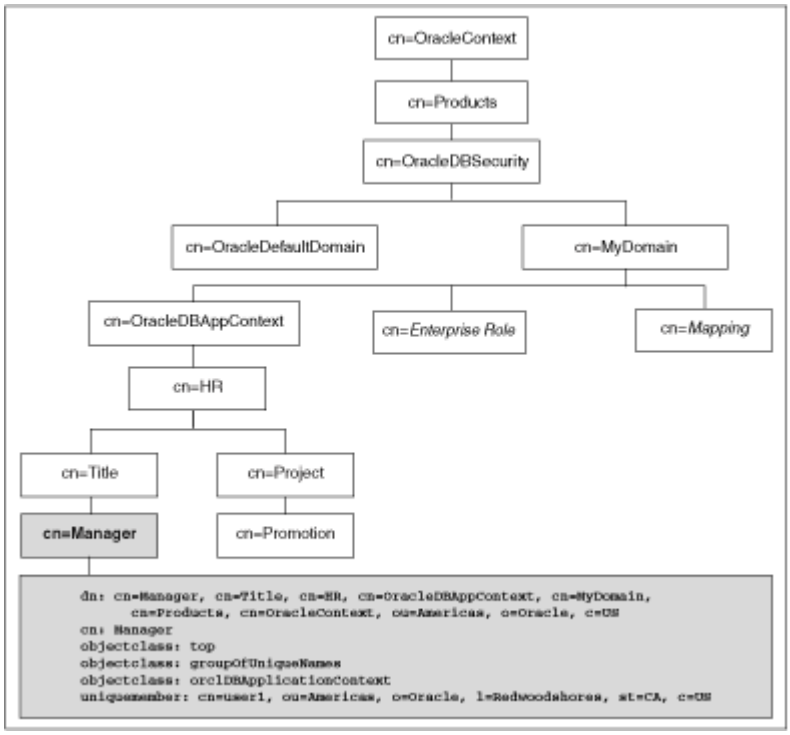
ディレクトリ・サービスとして使用できます。

orclDBApplicationContext LDAPオブジェクト(groupOfUniqueNamesのサブクラス)は、アプリケーション・コンテキスト値をディレクトリに格納します。アプリケーション・コンテキスト・オブジェクトの場所は、Human Resourcesの例に基づく図13-1で説明しています。

LDAPオブジェクトinetOrgPersonにより、一部の属性では複数のエントリが存在可能です。ただし、これらのエントリがデータベースにロードされ、SYS_LDAP_USER_DEFAULTコンテキスト・ネームスペースでアクセスされると、これらのエントリのうちの最初のエントリのみが返されます。たとえば、ユーザーに対するinetOrgPersonオブジェクトでは、telephoneNumberに複数のエントリが可能です(そのため、ユーザーは複数の電話番号を保存できます)。SYS_LDAP_USER_DEFAULTコンテキスト・ネームスペースを使用すると、最初の電話番号だけが取得されます。提供された属性と値のリストが要求に関して十分でない場合は、DBMS_LDAP PL/SQLパッケージを使用してディレクトリから追加の値をフェッチできます。

LDAP側では、アプリケーション・コンテキスト値のリストをデータベースに返すorclDBApplicationContext値を取得するための内部C関数が必要になります。この例では、HRはネームスペース、TitleとProjectは属性、ManagerとPromotionは値です。

図13-1 LDAPディレクトリの情報ツリーにおけるアプリケーション・コンテキストの位置



親トピック: データベース・セッション・ベースのアプリケーション・コンテキストのグローバルな初期化

13.3.11.3 グローバルに初期化されたデータベース・セッション・ベースのアプリケーション・コンテキストの動作

グローバルに初期化されたセキュア・アプリケーションを使用するには、エンタープライズ・ユーザー・セキュリティを最初に構成する必要があります。

次に、データベースとディレクトリのユーザーに対してアプリケーション・コンテキスト値を構成します。

グローバル・ユーザー(エンタープライズ・ユーザー)がデータベースに接続すると、エンタープライズ・ユーザー・セキュリティ機能によって、データベースに接続しているユーザーの識別情報が検証されます。認証後、グローバル・ユーザー・ロールとアプリケーション・コンテキストがディレクトリから取得されます。ユーザーがデータベースにログインするときは、グローバル・ロールと初期アプリケーショ

ン・コンテキストがすでに設定されています。

関連項目:

エンタープライズ・ユーザー・セキュリティの構成の詳細は、『Oracle Databaseエンタープライズ・ユーザー・セキュリティ管理者ガイド』を参照してください。

親トピック: データベース・セッション・ベースのアプリケーション・コンテキストのグローバルな初期化

13.3.11.4 データベース・セッション・ベースのアプリケーション・コンテキストのグローバルな初期化

ユーザーの初期アプリケーション・コンテキスト(部門名や職位など)は、LDAPディレクトリに設定および格納できます。

これらの値はユーザーのログイン中に取得されるため、コンテキストは適切に設定されます。さらに、ユーザーに関するすべての情報が取得され、アプリケーション・コンテキスト・ネームスペースSYS_USER_DEFAULTSに格納されます。

1. データベースにアプリケーション・コンテキストを作成します。

```
CREATE CONTEXT hr USING hrapps.hr_manage_pkg INITIALIZED GLOBALLY;
```

2. 新規のエントリを作成し、LDAPディレクトリに追加します。

   LDAPディレクトリに追加されたエントリの例を次に示します。これらのエントリは、アプリケーション(ネームスペース)HRに属性値Managerを持つ属性名Titleを作成し、ユーザー名user1およびuser2を割り当てます。次の例では、cn=exampleはドメイン名を示します。

```
dn:
cn=OracleDBAppContext,cn=example,cn=OracleDBSecurity,cn=Products,cn=OracleConte
xt,ou=Americas,o=oracle,c=US
changetype: add
cn: OracleDBAppContext
objectclass: top
objectclass: orclContainer
dn:
cn=hr,cn=OracleDBAppContext,cn=example,cn=OracleDBSecurity,cn=Products,cn=Oracl
eContext,ou=Americas,o=oracle,c=US
changetype: add
cn: hr
objectclass: top
objectclass: orclContainer
dn: cn=Title,cn=hr,
cn=OracleDBAppContext,cn=example,cn=OracleDBSecurity,cn=Products,cn=OracleConte
xt,ou=Americas,o=oracle,c=US
changetype: add
cn: Title
objectclass: top
objectclass: orclContainer
dn: cn=Manager,cn=Title,cn=hr,
cn=OracleDBAppContext,cn=example,cn=OracleDBSecurity,cn=Products,cn=OracleConte
xt,ou=Americas,o=oracle,c=US
cn: Manager
objectclass: top
objectclass: groupofuniquenames
objectclass: orclDBApplicationContext
uniquemember: CN=user1,OU=Americas,O=Oracle,L=Redwoodshores,ST=CA,C=US
uniquemember: CN=user2,OU=Americas,O=Oracle,L=Redwoodshores,ST=CA,C=US
```

3. このユーザーに関するLDAP inetOrgPersonオブジェクト・エントリが存在する場合、接続では、複数の属性がinetOrgPersonから取得され、これらがネームスペースSYS_LDAP_USER_DEFAULTに割り当てられます。コンテ

キストには、inetOrgPersonオブジェクト・クラスの一部であるNULL以外の値のみが移入されます。他の属性は作成されません。

次に、inetOrgPersonエントリの例を示します。

```
dn: cn=user1,ou=Americas,O=oracle,L=redwoodshores,ST=CA,C=US
changetype: add
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
cn: user1
sn: One
givenName: User
initials: UO
title: manager, product development
uid: uone
mail: uone@us.example.com
telephoneNumber: +1 650 555 0105
employeeNumber: 00001
employeeType: full time
```

4. データベースに接続します。

user1がドメインexampleに属するデータベースに接続する場合、user1のTitleはManagerに設定されます。user1に関するすべての情報が、LDAPディレクトリから取得されます。値は、次の構文を使用して取得できます。

```
SYS_CONTEXT('namespace','attribute name')
```

たとえば:

```
DECLARE
 tmpstr1 VARCHAR2(30);
 tmpstr2 VARCHAR2(30);
BEGIN
 tmpstr1 = SYS_CONTEXT('HR','TITLE);
 tmpstr2 = SYS_CONTEXT('SYS_LDAP_USER_DEFAULT','telephoneNumber');
 DBMS_OUTPUT.PUT_LINE('Title is ' || tmpstr1);
 DBMS_OUTPUT.PUT_LINE('Telephone Number is ' || tmpstr2);
END;
```

次に、この例の出力を示します。

```
Title is Manager
Telephone Number is +1 650 555 0105
```

親トピック: データベース・セッション・ベースのアプリケーション・コンテキストのグローバルな初期化

13.3.12 外部化されたデータベース・セッション・ベースのアプリケーション・コンテキスト

多くのアプリケーションでは、ファイングレイン・アクセス・コントロールに使用される属性をデータベース・メタデータ表に格納します。

たとえば、employees表に、コスト・センター、職責、署名認証などのファイングレイン・アクセス・コントロールに有効な情報を含めることができます。ただし、多くの組織ではユーザー情報をOracle Internet DirectoryなどのLDAPベースのディレクトリに集中化して、ユーザーを管理し、アクセス制御しています。アプリケーション・コンテキスト属性は、Oracle Internet Directoryに格納して1人以上のエンタープライズ・ユーザーに割り当てることができます。また、これらの属性は、エンタープライズ・ユーザーがログインすると自動的に取得され、アプリケーション・コンテキストの初期化にも使用されます。

関連項目:

- OCIインタフェース、ジョブ・キュー・プロセスまたはデータベース・リンクなど、外部リソースを介したローカル・アプリケーション・コンテキストの初期化に関する詳細は、データベース・セッション・ベースのアプリケーション・コンテキストの外部での初期化を参照してください
- Oracle Internet Directoryなどの集中化されたリソースを介したローカル・アプリケーション・コンテキストの初期化に関する詳細は、データベース・セッション・ベースのアプリケーション・コンテキストのグローバルな初期化を参照してください
- エンタープライズ・ユーザーの詳細は、『Oracle Databaseエンタープライズ・ユーザー・セキュリティ管理者ガイド』を参照してください。

親トピック: データベース・セッション・ベースのアプリケーション・コンテキストの使用

13.4 グローバル・アプリケーション・コンテキスト

グローバル・アプリケーション・コンテキストを使用して、Oracle Real Application Clusters環境などのデータベース・セッション間でアプリケーション値にアクセスできます。

- グローバル・アプリケーション・コンテキストについて
  グローバル・アプリケーション・コンテキストを使用すると、アプリケーション・コンテキスト値を複数のデータベース・セッション(Oracle RACインスタンスを含む)にわたってアクセス可能にできます。
- グローバル・アプリケーション・コンテキストの使用方法
  グローバル・アプリケーション・コンテキストには、3種類の一般的な使用方法があります。
- グローバル・アプリケーション・コンテキストのコンポーネント
  グローバル・アプリケーション・コンテキストはパッケージを使用して属性を管理し、中間層アプリケーションを使用してクライアント・セッションIDを管理します。
- Oracle Real Application Clusters環境でのグローバル・アプリケーション・コンテキスト
  Oracle RAC環境では、グローバル・アプリケーション・コンテキストがロードまたは変更された場合は、常に、既存のアクティブ・インスタンスによってのみ、そのことが認識されます。
- グローバル・アプリケーション・コンテキストの作成
  `CREATE CONTEXT` SQL文によって作成されたグローバル・アプリケーション・コンテキストはSYSスキーマに配置されます。
- グローバル・アプリケーション・コンテキストを管理するためのPL/SQLパッケージ
  `DBMS_SESSION` PL/SQLパッケージで、グローバル・アプリケーション・コンテキストを管理します。
- クライアント・セッションIDを管理するための中間層アプリケーションへのコールの埋込み
  中間層アプリケーションにコールを埋め込み、クライアント・セッションIDを管理できます。
- 例: クライアント・セッションIDを使用するグローバル・アプリケーション・コンテキストの作成
  このチュートリアルでは、クライアント・セッションIDを使用するグローバル・アプリケーション・コンテキストの作成を示しています。
- グローバル・アプリケーション・コンテキスト・プロセス
  簡単なグローバル・アプリケーション・コンテキストはデータベース・ユーザー・アカウントを使用し、ユーザー・セッションを作成します。グローバル・アプリケーション・コンテキストは軽量ユーザー用です。

親トピック: アプリケーション・コンテキストを使用したユーザー情報の取得

13.4.1 グローバル・アプリケーション・コンテキストについて

グローバル・アプリケーション・コンテキストを使用すると、アプリケーション・コンテキスト値を複数のデータベース・セッション(Oracle RACインスタンスを含む)にわたってアクセス可能にできます。

グローバル・アプリケーション・コンテキスト情報はシステム・グローバル領域(SGA: 共有グローバル領域でSGAと称する場合もあります)に格納されるため、3層アーキテクチャの中間層アプリケーションなど、セッションを使用しないアプリケーションに対して使用できます。

このようなアプリケーションでは、複数のユーザーがアプリケーションに対して認証され、通常は単一の識別情報としてデータベースに接続するため、セッション・ベースのアプリケーション・コンテキストを使用できません。グローバル・アプリケーション・コンテキストは、ユーザー・セッション単位ではなく一度に初期化されます。したがって、接続は接続プールから再利用されるため、パフォーマンスが向上します。

グローバル・アプリケーション・コンテキスト値は、`ALTER SYSTEM FLUSH GLOBAL_CONTEXT` SQL文を実行してクリアできます。

親トピック: グローバル・アプリケーション・コンテキスト

13.4.2 グローバル・アプリケーション・コンテキストの使用方法

グローバル・アプリケーション・コンテキストには、3種類の一般的な使用方法があります。

使用方法は次のとおりです。

- 全データベース・ユーザーを対象にしてアプリケーションの値をグローバルに共有する必要がある場合。たとえば、特定の状況に基づいてアプリケーションへのアクセスを禁止することがあります。この場合、アプリケーション・コンテキストが設定する値はユーザーに固有の値ではなく、ユーザーのプライベート・データに基づく値でもありません。アプリケーション・コンテキストは、ある状況を定義して、実行中のアプリケーション・モジュールのバージョンなどを示します。
- 複数のアプリケーション間を移動する必要があるデータベース・ユーザーがいる場合。この場合、ユーザーの移動先となる第2のアプリケーションには、第1のアプリケーションとは異なるアクセス要件があります。
- 非データベース・ユーザー(つまり、データベースに認識されていないユーザー)を認証する必要がある場合。データベース・アカウントを持たないこのタイプのユーザーは、通常、Webアプリケーション経由で接続プールを使用して接続します。このようなアプリケーションでは、One Big Application User認証モデルを使用して、複数のユーザーを単一のユーザーとしてデータベースに接続します。このタイプのユーザーを認証するには、そのユーザーのクライアント・セッションIDを使用します。

親トピック: グローバル・アプリケーション・コンテキスト

13.4.3 グローバル・アプリケーション・コンテキストのコンポーネント

グローバル・アプリケーション・コンテキストはパッケージを使用して属性を管理し、中間層アプリケーションを使用してクライアント・セッションIDを管理します。

- グローバル・アプリケーション・コンテキスト。`CREATE CONTEXT` SQL文に`ACCESSED GLOBALLY`句を指定して、グローバル・アプリケーション・コンテキストを作成します。この文は、アプリケーション・コンテキスト名を設定し、その名前に、アプリケーション・コンテキスト・データを設定するように設計されたPL/SQLプロシージャを関連付けます。グローバル・アプリケーション・コンテキストは、そのコンテキストを作成したセキュリティ管理者のデータベース・スキーマに作成および格納されます。

- 属性を設定するPL/SQLパッケージ。このパッケージには、DBMS_SESSION.SET_CONTEXTプロシージャを使用してグローバル・アプリケーション・コンテキストを設定するプロシージャが含まれている必要があります。SET_CONTEXTプロシージャには、この項に記載されている3種類すべての状況に適応するグローバル・アプリケーション・コンテキストを生成できるパラメータが用意されています。このPL/SQLパッケージは、データベース・サーバーで作成、格納および実行します。通常、このパッケージは、それを作成したセキュリティ管理者のスキーマに属します。
- クライアント・セッションIDを取得および設定する中間層アプリケーション。非データベース・ユーザー(認証にはクライアント・セッションIDが必要)の場合は、中間層アプリケーションでOracle Call Interface(OCI)コールを使用して、それぞれのセッション・データを取得および設定できます。DBMS_SESSION.SET_IDENTIFIERプロシージャを使用してクライアント・セッションIDを設定することもできます。クライアント・セッションIDを作成して非データベース・ユーザー名を保存する利点は、DBA_AUDIT_TRAIL、DBA_FGA_AUDIT_TRAIL、DBA_COMMON_AUDIT_TRAILデータ・ディクショナリ・ビューのCLIENT_ID列を問い合せてこのユーザーのアクティビティを監査できることです。

ノート:

DBMS_APPLICATION_INFO.SET_CLIENT_INFO 設定で値を上書きできることに注意してください。

関連トピック

- DBMS_SESSION PL/SQLパッケージを使用したクライアント識別子の設定とクリア

親トピック: グローバル・アプリケーション・コンテキスト

13.4.4 Oracle Real Application Clusters環境でのグローバル・アプリケーション・コンテキスト

Oracle RAC環境では、グローバル・アプリケーション・コンテキストがロードまたは変更された場合は、常に、既存のアクティブ・インスタンスによってのみ、そのことが認識されます。

Oracle RAC環境でグローバル・アプリケーション・コンテキスト値を設定すると、コンテキスト値をすべてのOracle RACインスタンスに一貫して伝播することによるパフォーマンスのオーバーヘッドが生じることに注意してください。

1つのOracle RACインスタンスでグローバル・アプリケーション・コンテキストをフラッシュすると(ALTER SYSTEM FLUSH GLOBAL_CONTEXT SQL文を使用して)、他のすべてのOracle RACインスタンスでも、すべてのグローバル・アプリケーション・コンテキストがフラッシュされます。

親トピック: グローバル・アプリケーション・コンテキスト

13.4.5 グローバル・アプリケーション・コンテキストの作成

CREATE CONTEXT SQL文によって作成されたグローバル・アプリケーション・コンテキストはSYSスキーマに配置されます。

- グローバル・アプリケーション・コンテキストの所有権
  グローバル・アプリケーション・コンテキストを所有できるのはSYSスキーマのみです。
- グローバル・アプリケーション・コンテキストの作成
  ローカル・アプリケーション・コンテキストと同様、グローバル・アプリケーション・コンテキストもセキュリティ管理者のデータベース・スキーマで作成および格納されます。

親トピック: グローバル・アプリケーション・コンテキスト

13.4.5.1 グローバル・アプリケーション・コンテキストの所有権

グローバル・アプリケーション・コンテキストを所有できるのはSYSスキーマのみです。

グローバル・アプリケーション・コンテキストの所有権については、CREATE ANY CONTEXT権限とDROP ANY CONTEXT権限を付与されたユーザーがそのグローバル・アプリケーション・コンテキストを作成および削除できた場合でも、SYSスキーマが所有しています。

Oracle Databaseは、作成したスキーマ・アカウントにコンテキストを関連付けますが、このユーザーを削除した場合、コンテキストはそれでもSYSスキーマの中に存在します。ユーザーSYSであれば、アプリケーション・コンテキストを削除できます。

親トピック: グローバル・アプリケーション・コンテキストの作成

13.4.5.2 グローバル・アプリケーション・コンテキストの作成

ローカル・アプリケーション・コンテキストと同様、グローバル・アプリケーション・コンテキストもセキュリティ管理者のデータベース・スキーマで作成および格納されます。

グローバル・アプリケーション・コンテキストを作成するには、CREATE ANY CONTEXTシステム権限が必要であり、DROP CONTEXT文でコンテキストを削除するにはDROP ANY CONTEXT権限が必要になります。

- グローバル・アプリケーション・コンテキストを作成するには、CREATE CONTEXT SQL文を使用してグローバル・アプリケーション・コンテキストを作成し、ACCESSED GLOBALLY句をこのSQL文に含めます。

たとえば:

```
CREATE OR REPLACE CONTEXT global_hr_ctx USING hr_ctx_pkg ACCESSED GLOBALLY CONTAINER
= ALL;
```

親トピック: グローバル・アプリケーション・コンテキストの作成

13.4.6 グローバル・アプリケーション・コンテキストを管理するためのPL/SQLパッケージ

DBMS_SESSION PL/SQLパッケージで、グローバル・アプリケーション・コンテキストを管理します。

- グローバル・アプリケーション・コンテキストを管理するパッケージについて
  グローバル・アプリケーション・コンテキストに関連付けるパッケージは、DBMS_SESSIONパッケージを使用してグローバル・アプリケーション・コンテキスト値の設定とクリアを行います。
- エディションがグローバル・アプリケーション・コンテキストのPL/SQLパッケージの結果に与える影響
  グローバル・アプリケーション・コンテキストのパッケージ、Oracle Virtual Private Databaseのパッケージ、およびファイングレイン監査ポリシーは、複数のエディションで使用できます。
- DBMS_SESSION.SET_CONTEXTのusernameおよびclient_idパラメータ
  DBMS_SESSION.SYS_CONTEXTプロシージャには、グローバル・アプリケーション・コンテキストに使用するclient_idおよびusernameパラメータが用意されています。
- 全データベース・ユーザーを対象としたグローバル・アプリケーション・コンテキスト値の共有
  全データベース・ユーザーを対象としてグローバル・アプリケーション値を共有して、データベース内のデータへのアクセス権を付与できます。
- 例: 全データベース・ユーザーを対象としてグローバル・アプリケーション値を管理するためのパッケージ
  CREATE PACKAGE文で、全データベース・ユーザーを対象としてグローバル・アプリケーション値を管理できます。
- アプリケーション間を移動するデータベース・ユーザーのグローバル・コンテキスト

アプリケーションによってアクセス要件が異なる場合でも、アプリケーション間を移動するデータベース・ユーザーにグローバル・アプリケーション・コンテキストを使用できます。

- 非データベース・ユーザーのグローバル・アプリケーション・コンテキスト
  非データベース・ユーザーがクライアント・セッションを開始すると、アプリケーション・サーバーでクライアント・セッションIDが生成されます。
- 例: 非データベース・ユーザーのグローバル・アプリケーション・コンテキスト値を管理するためのパッケージ
  CREATE PACKAGE文で、非データベース・ユーザーのグローバル・アプリケーション・コンテキスト値を管理できます。
- セッションをクローズする際のセッション・データのクリア
  アプリケーション・コンテキストはメモリー内に存在するので、ユーザーがセッションを終了したときは、client_identifierコンテキスト値をクリアする必要があります。

親トピック: グローバル・アプリケーション・コンテキスト

13.4.6.1 グローバル・アプリケーション・コンテキストを管理するパッケージについて

グローバル・アプリケーション・コンテキストに関連付けるパッケージは、DBMS_SESSIONパッケージを使用してグローバル・アプリケーション・コンテキスト値の設定とクリアを行います。

このプロシージャを使用するには、DBMS_SESSIONパッケージに対するEXECUTE権限が必要です。通常、このパッケージはセキュリティ管理者のデータベース・スキーマに作成および格納されます。DBMS_SESSIONパッケージはSYSスキーマの所有です。

ローカル・アプリケーション・コンテキストの設定に使用するPL/SQLパッケージとは異なり、ユーザー・セッション・データを取得するSYS_CONTEXT関数は挿入しません。グローバル・アプリケーション・コンテキストでは、接続しているすべてのユーザーにとって、セッションの所有者(USERENVコンテキストに記録される)は同じ所有者となるため、この関数を挿入する必要はありません。

プロシージャは、グローバル・アプリケーション・コンテキストのPL/SQLパッケージ内でいつでも実行できます。グローバル・アプリケーション・コンテキストに関連付けられているパッケージ・プロシージャを実行するために、ログイン・トリガーとログオフ・トリガーを作成する必要はありません。一般的に、パッケージ・プロシージャはデータベース・アプリケーション内から実行します。また、非データベース・ユーザーには、中間層アプリケーションを使用してクライアント・セッションIDを取得および設定します。

関連項目:

DBMS_SESSIONパッケージの詳細は、『Oracle Database PL/SQLパッケージおよびタイプ・リファレンス』を参照してください。

親トピック: グローバル・アプリケーション・コンテキストを管理するためのPL/SQLパッケージ

13.4.6.2 エディションがグローバル・アプリケーション・コンテキストのPL/SQLパッケージの結果に与える影響

グローバル・アプリケーション・コンテキストのパッケージ、Oracle Virtual Private Databaseのパッケージ、およびファイングレイン監査ポリシーは、複数のエディションで使用できます。

次のガイドラインに従ってください。

- PL/SQLパッケージの結果が、すべてのエディションで同じになるようにします。そのためには、エディションが使用可能になっていないユーザーのスキーマで、パッケージを作成します。エディションが使用可能になっていないユーザーを検索するには、DBA_USERSおよびUSER_USERSデータ・ディクショナリ・ビューを問い合せます。SYS、SYSTEM、およびDBA_REGISTRYデータ・ディクショナリ・ビューにリストされている他のデフォルトのOracle Database管理者アカウントは、エディションが使用不可であり、使用可能にすることもできません。

- PL/SQLパッケージの結果が、パッケージが実行されているエディションの現在の状態に依存するようにします。その結果は、パッケージが適用されるすべてのエディションで異なることがあります。この場合、エディションが使用可能になっているユーザーのスキーマで、パッケージを作成します。スキーマでエディションが使用可能な場合、各エディションに存在するパッケージの実コピーが異なり、各コピーの動作が異なる可能性があります。これは、次のようなシナリオにおいて役立ちます。
  - パッケージで新しいアプリケーション・コンテキストを使用する必要がある場合。
  - パッケージで、別のスキームを使用して入力値をエンコードする必要がある場合。
  - パッケージで、データベースにログインするユーザーに別の検証ルールを適用する必要がある場合。

  グローバル・アプリケーション・コンテキストを設定するPL/SQLパッケージでは、1つのgetter関数を使用して、アプリケーション・コンテキストのキー値のペアを読み取る、プリミティブなSYS_CONTEXTコールをラップします。このgetter関数は、アプリケーション・コンテキストのsetterプロシージャと同じパッケージに配置できます。この方法により、関連する概念を反映するように、アプリケーション・コンテキストのキー値をタグ付けできます。たとえば、setter関数が実際に存在するエディションをタグに使用できます。または、コンテキストを設定するセッションの現行エディションを使用できます、これは、SYS_CONTEXT('USERENV', 'CURRENT_EDITION_NAME')を使用して検索できます。このタグには、setter関数を適用する任意の概念を使用できます。

  関連項目:

  エディションの詳細は、『Oracle Database開発ガイド』を参照してください。

親トピック: グローバル・アプリケーション・コンテキストを管理するためのPL/SQLパッケージ

13.4.6.3 DBMS_SESSION.SET_CONTEXTのusernameおよびclient_idパラメータ

DBMS_SESSION.SYS_CONTEXTプロシージャには、グローバル・アプリケーション・コンテキストに使用するclient_idおよびusernameパラメータが用意されています。

これらの設定の組合せによって、作成可能なグローバル・アプリケーション・コンテキストのタイプがどのように制御されるかについて、表13-2で説明します。

表13-2 DBMS_SESSION.SET_CONTEXTのusernameおよびclient_idパラメータの設定

設定の組合せ	結果
username を NULL に設定 client_id を NULL に設定	この組合せでは、すべてのユーザーがアプリケーション・コンテキストにアクセスできます。詳細は、「全データベース・ユーザーを対象としたグローバル・アプリケーション・コンテキスト値の共有」を参照してください。 これらの設定は、データベース・セッション・ベースのアプリケーション・コンテキストにも使用されます。詳細は、「データベース・セッション・ベースのアプリケーション・コンテキストの使用」を参照してください。
username をある値に設定 client_id を NULL に設定	この組合せでは、username 設定が同じであるかぎり、複数のセッションによるアプリケーション・コンテキストへのアクセスが可能です。指定されているユーザー名は有効なデータベース・ユーザーであることが必要です。詳細は、「アプリケーション間を移動する

設定の組合せ	結果
	データベース・ユーザーのグローバル・コンテキスト」を参照してください。
username を NULL に設定 client_id をある値に設定	この組合せでは、client_id パラメータが同じ値に設定されているかぎり、複数のユーザー・セッションによる 1 つのアプリケーションへのアクセスが可能です。これによって、全ユーザーのセッションが、アプリケーション・コンテキスト値を参照できます。
username をある値に設定 client_id をある値に設定	この組合せでは、次の 2 つの場合が考えられます。 ● 軽量ユーザー。ユーザーにデータベース・アカウントがない場合は、指定されたユーザー名が接続プール所有者です。client_id 設定は、ログインしている非データベース・ユーザーに関連付けられます。 ● データベース・ユーザー。ユーザーがデータベース・ユーザーの場合、この組合せはステートレス Web セッションに使用できます。 SET_CONTEXT プロシージャの username パラメータを USER に設定すると、Oracle Database に用意されている USER 関数がコールされます。この USER 関数によって、アプリケーション・コンテキスト取得プロセスからセッション所有者が指定され、アプリケーション・コンテキストを設定したユーザーのみがコンテキストにアクセスできるようになります。USER 関数の詳細は、『Oracle Database SQL 言語リファレンス』を参照してください。 詳細は、「非データベース・ユーザーのグローバル・アプリケーション・コンテキスト」を参照してください。

親トピック: グローバル・アプリケーション・コンテキストを管理するためのPL/SQLパッケージ

13.4.6.4 全データベース・ユーザーを対象としたグローバル・アプリケーション・コンテキスト値の共有

全データベース・ユーザーを対象としてグローバル・アプリケーション値を共有して、データベース内のデータへのアクセス権を付与できます。

- 全データベース・ユーザーを対象としてグローバル・アプリケーションの値を共有するには、SET_CONTEXTプロシージャのnamespace、attributeおよびvalueパラメータを設定します。

関連トピック

- 例: 全データベース・ユーザーを対象としてグローバル・アプリケーション値を管理するためのパッケージ

親トピック: グローバル・アプリケーション・コンテキストを管理するためのPL/SQLパッケージ

13.4.6.5 例: 全データベース・ユーザーを対象としてグローバル・アプリケーション値を管理するためのパッケージ

CREATE PACKAGE文で、全データベース・ユーザーを対象としてグローバル・アプリケーション値を管理できます。

例13-7に、全データベース・ユーザーを対象としてグローバル・アプリケーション・コンテキストを設定およびクリアするパッケージの

作成方法を示します。

例13-7 全データベース・ユーザーを対象としてグローバル・アプリケーション値を管理するためのパッケージ

```
CREATE OR REPLACE PACKAGE hr_ctx_pkg
   AS
    PROCEDURE set_hr_ctx(sec_level IN VARCHAR2);
    PROCEDURE clear_hr_context;
   END;
  /
  CREATE OR REPLACE PACKAGE BODY hr_ctx_pkg
   AS
    PROCEDURE set_hr_ctx(sec_level IN VARCHAR2)
    AS
    BEGIN
     DBMS_SESSION.SET_CONTEXT(
      namespace  => 'global_hr_ctx',
      attribute  => 'job_role',
      value      => sec_level);
     END set_hr_ctx;

  PROCEDURE clear_hr_context
    AS
    BEGIN
     DBMS_SESSION.CLEAR_CONTEXT('global_hr_ctx', 'job_role');
    END clear_context;
  END;
 /
```

この例では、次のようになります。

- DBMS_SESSION.SET_CONTEXT ... END set_hr_ctxは、DBMS_SESSION.SET_CONTEXTプロシージャを使用して、namespace、attributeおよびvalueパラメータの値を設定します。sec_level値は、データベース・アプリケーションがhr_ctx_pkg.set_hr_ctxプロシージャを実行する際に指定されます。

  username値とclient_id値は設定されていません。そのためこれらはNULLです。これによりすべてのユーザー(データベース・ユーザー)が値にアクセスできることになり、サーバー全体として適切な設定になります。

- namespace => 'global_hr_ctx'は、SET_CONTEXTプロシージャで、namespaceをglobal_hr_ctxに設定します。
- attribute => 'job_role'は、job_role属性を作成します。
- value => sec_levelは、job_role属性の値をsec_levelに設定します。
- PROCEDURE clear_hr_contextは、コンテキスト値を消去するclear_hr_contextプロシージャを作成します。詳細は、「セッションをクローズする際のセッション・データのクリア」を参照してください。

通常、このプロシージャは1つのデータベース・アプリケーション内で実行します。たとえば、ログインしているすべてのユーザーが社員であり、「社員」というセキュリティ・レベルを使用する場合は、次の例のようにデータベース・アプリケーション内にコールを埋め込みます。

```
BEGIN
 hr_ctx_pkg.set_hr_ctx('clerk');
END;
/
```

プロシージャが正常に完了した場合は、次のようにしてアプリケーション・コンテキスト値をチェックできます。

```
SELECT SYS_CONTEXT('global_hr_ctx', 'job_role') job_role FROM DUAL;
JOB_ROLE
-----------
```

```
clerk
```

全データベース・ユーザーを対象としてグローバル・アプリケーション・コンテキスト値をクリアするには、次のプロシージャを実行します。

```
BEGIN
 hr_ctx_pkg.clear_hr_context;
END;
/
```

グローバル・コンテキスト値が実際にクリアされていることをチェックする場合、次のSELECT文では値は戻りません。

```
SELECT SYS_CONTEXT('global_hr_ctx', 'job_role') job_role FROM DUAL;
JOB_ROLE
-----------
```

権限が不十分であることを示すエラー・メッセージがOracle Databaseで返された場合は、グローバル・アプリケーション・コンテキストが正しく作成されていることを確認してください。また、DBA_CONTEXTデータベース・ビューを問い合せて、設定が正しいこと(作成したスキーマからプロシージャがコールされていることなど)を確認してください。

NULLが戻る場合は、誤ってクライアント識別子を設定した可能性があります。クライアント識別子をクリアするには、次のプロシージャを実行します。

```
EXEC DBMS_SESSION.CLEAR_IDENTIFIER;
```

親トピック: グローバル・アプリケーション・コンテキストを管理するためのPL/SQLパッケージ

13.4.6.6 アプリケーション間を移動するデータベース・ユーザーのグローバル・コンテキスト

アプリケーションによってアクセス要件が異なる場合でも、アプリケーション間を移動するデータベース・ユーザーにグローバル・アプリケーション・コンテキストを使用できます。

これを行うには、DBMS_SESSION.SET_CONTEXTプロシージャにusernameパラメータを含める必要があります。

このパラメータは、すべてのセッションを対象として同じスキーマを使用することを指定します。

次のDBMS_SESSION.SET_CONTEXTパラメータを使用できます。

- namespace
- attribute
- value
- username

Oracle Databaseは、username値を一致させることで、他のアプリケーションがアプリケーション・コンテキストを認識できるようにします。これによって、ユーザーは複数のアプリケーション間を移動できます。

client_id設定を省略すると、その値はデフォルトのNULLになります。これは、異なるアプリケーションで同じコンテキストを保持するデータベース・ユーザーに対して同じusernameが設定されている場合は、複数のセッションで値を参照できることを意味します。たとえば、各ユーザーを1つのジョブ・ロールに制限し、Oracle Virtual Private Databaseポリシーを使用してユーザー・アクセスを制御するアプリケーション一式を保持できます。

例13-8に、特定のユーザーが複数のアプリケーション間を移動できるように、usernameパラメータを設定する方法を示します。usernameパラメータを使用している部分は、boldで記載しています。

例13-8 複数のアプリケーション間を移動するグローバル・アプリケーション・コンテキスト値のパッケージ

```
CREATE OR REPLACE PACKAGE hr_ctx_pkg
```

```
 AS
   PROCEDURE set_hr_ctx(sec_level IN VARCHAR2, user_name IN VARCHAR2);
   PROCEDURE clear_hr_context;
  END;
 /
 CREATE OR REPLACE PACKAGE BODY hr_ctx_pkg
  AS
   PROCEDURE set_hr_ctx(sec_level IN VARCHAR2, user_name IN VARCHAR2)
   AS
    BEGIN
     DBMS_SESSION.SET_CONTEXT(
      namespace  => 'global_hr_ctx',
      attribute  => 'job_role',
      value      => sec_level,
      username   => user_name);
     END set_hr_ctx;

  PROCEDURE clear_hr_context
   AS
    BEGIN
     DBMS_SESSION.CLEAR_CONTEXT('global_hr_ctx');
    END clear_context;
 END;
/
```

通常は、次の例のようにコールを埋め込むことで、このプロシージャをデータベース・アプリケーション内で実行します。user_nameパラメータの値(この場合はscott)には、必ず有効なデータベース・ユーザー名を指定してください。

```
BEGIN
 hr_ctx_pkg.set_hr_ctx('clerk', 'scott');
END;
```

このタイプのグローバル・アプリケーション・コンテキストを安全に管理する方法は、自分のアプリケーション範囲内にコードを埋め込み、セキュア・アプリケーション・ロールをユーザーに付与することです。このコードでは、アプリケーション・コンテキストを設定するトラステッドPL/SQLパッケージに対してEXECUTE権限を指定する必要があります。つまり、アプリケーション(ユーザーではなく)がユーザーのコンテキストを設定します。

親トピック: グローバル・アプリケーション・コンテキストを管理するためのPL/SQLパッケージ

13.4.6.7 非データベース・ユーザーのグローバル・アプリケーション・コンテキスト

非データベース・ユーザーがクライアント・セッションを開始すると、アプリケーション・サーバーでクライアント・セッションIDが生成されます。

非データベース・ユーザーとはデータベースにとって未知のユーザーのことで、Webアプリケーション・ユーザーなどがこれに当たります。

このIDは、アプリケーション・サーバーで設定後にデータベース・サーバー側へ渡す必要があります。そのためには、DBMS_SESSION.SET_IDENTIFIERプロシージャを使用してクライアント・セッションIDを設定します。

コンテキストを設定するには、サーバー側のPL/SQLプロシージャのDBMS_SESSION.SET_CONTEXTプロシージャにclient_idパラメータを設定します。これでアプリケーション・コンテキストをグローバルに管理できますが、各クライアントには、各自が割り当てられているアプリケーション・コンテキストのみが表示されます。

ここで、client_id値は、グローバル・アプリケーション・コンテキストの正しい属性を取得および設定するためのキーです。クライアント識別子は中間層アプリケーションによって制御され、一度設定されるとクリアされるまでオープン状態のままです。

このタイプのアプリケーション・コンテキストを管理する一般的な方法は、session_id(client_identifier)の値をCookieに格納し、次のリクエストで戻されるように、エンド・ユーザーのHTMLページに送信する方法です。また、アプリケーション

の参照表にクライアント識別子を保持し、他のユーザーのために使用されたり、エンド・ユーザーのセッション・タイムアウトを実装するために利用されるのを防止する必要があります。

非データベース・ユーザーの場合は、次のSET_CONTEXTパラメータを構成します。

- namespace
- attribute
- value
- username
- client_id

関連トピック

- 例: クライアント・セッションIDを使用するグローバル・アプリケーション・コンテキストの作成
- ステップ2: 中間層アプリケーションを使用したクライアント・セッションIDの設定
- データベースに認識されないアプリケーション・ユーザーの識別でのクライアント識別子の使用

親トピック: グローバル・アプリケーション・コンテキストを管理するためのPL/SQLパッケージ

13.4.6.8 例: 非データベース・ユーザーのグローバル・アプリケーション・コンテキスト値を管理するためのパッケージ

CREATE PACKAGE文で、非データベース・ユーザーのグローバル・アプリケーション・コンテキスト値を管理できます。

例13-9に、このタイプのグローバル・アプリケーション・コンテキストを管理するパッケージの作成方法を示します。

例13-9 非データベース・ユーザーのグローバル・アプリケーション・コンテキスト値を管理するためのパッケージ

```
CREATE OR REPLACE PACKAGE hr_ctx_pkg
  AS
   PROCEDURE set_session_id(session_id_p IN NUMBER);
   PROCEDURE set_hr_ctx(sec_level_attr IN VARCHAR2,
      sec_level_val IN VARCHAR2);
   PROCEDURE clear_hr_session(session_id_p IN NUMBER);
   PROCEDURE clear_hr_context;
  END;
/
  CREATE OR REPLACE PACKAGE BODY hr_ctx_pkg
   AS
    session_id_global NUMBER;
  PROCEDURE set_session_id(session_id_p IN NUMBER)
   AS
   BEGIN
    session_id_global := session_id_p;
    DBMS_SESSION.SET_IDENTIFIER(session_id_p);
  END set_session_id;

  PROCEDURE set_hr_ctx(sec_level_attr IN VARCHAR2,
     sec_level_val IN VARCHAR2)
   AS
   BEGIN
    DBMS_SESSION.SET_CONTEXT(
     namespace  => 'global_hr_ctx',
     attribute  => sec_level_attr,
     value      => sec_level_val,
     username   => USER,
     client_id  => session_id_global);
   END set_hr_ctx;

  PROCEDURE clear_hr_session(session_id_p IN NUMBER)
   AS
```

```
  BEGIN
     DBMS_SESSION.SET_IDENTIFIER(session_id_p);
     DBMS_SESSION.CLEAR_IDENTIFIER;
  END clear_hr_session;

 PROCEDURE clear_hr_context
 AS
 BEGIN
  DBMS_SESSION.CLEAR_CONTEXT('global_hr_ctx', session_id_global);
 END clear_hr_context;
END;
/
```

この例では、次のようになります。

- `session_id_global NUMBER`は、クライアント・セッションIDを格納する`session_id_global`変数を作成します。この`session_id_global`変数は、グローバル・アプリケーション・コンテキストの属性を作成して値を割り当てるプロシージャも含めて、パッケージ定義全体から参照されます。これは、グローバル・アプリケーション・コンテキストの値が、この特定のセッションIDと常に関連付けられることを意味します。
- `PROCEDURE set_session_id ... END set_session_id`は、`set_session_id`プロシージャを作成します。これは、クライアント・セッションIDを`session_id_global`変数に書き込むプロシージャです。
- `PROCEDURE set_hr_ctx ... END set_hr_ctx`は、`set_hr_ctx`プロシージャを作成します。このプロシージャによりグローバル・アプリケーション・コンテキスト属性が作成されて、これらの属性に値を割り当てることができるようになります。このプロシージャ内では、次のようになります。
  - `username => USER`は、`username`値を指定します。この例では、Oracle Databaseによって指定される、コンテキスト取得プロセスからセッション所有者を追加するためのUSER関数を呼び出すことによって設定します。USER関数によって、アプリケーション・コンテキストを設定するユーザーのみがコンテキストにアクセスできるようになります。USER関数の詳細は、『Oracle Database SQL言語リファレンス』を参照してください。

    `NULL`(`username`パラメータのデフォルト)を指定すると、すべてのユーザーがコンテキストにアクセスできるようになります。

    `username`値と`client_id`値の両方を設定した場合は、2つの異なる使用目的が考えられます。軽量ユーザーの場合は、`username`パラメータに対して接続プール所有者(APPS_USERなど)を設定し、`client_id`に対してクライアント・セッションIDを設定します。ステートレスWebセッションを使用する場合は、ログインしたユーザーと同じデータベース・ユーザーを`user_name`パラメータに設定し、このユーザーが同じクライアント・セッションIDを保持していることを確認します。異なる`username`と`client_id`を設定した場合の動作の例は、「DBMS_SESSION.SET_CONTEXTのusernameおよびclient_idパラメータ」を参照してください。
  - `client_id => session_id_global`は、`client_id`値を指定します。この例では、この値を`session_id_global`変数に設定しています。これによって、ここで定義したコンテキスト設定が特定のクライアント・セッションID(つまり、`set_session_id`プロシージャを実行する際に設定されるID)に関連付けられます。`client_id`パラメータにデフォルトの`NULL`を指定すると、すべてのセッションでグローバル・アプリケーション・コンテキストの設定を使用できるようになります。
- `PROCEDURE clear_hr_session ... END clear_hr_session`は、クライアント・セッション識別子を消去する`clear_hr_session`プロシージャを作成します。AS句は、正しいセッションID、つまり`CREATE OR REPLACE PACKAGE BODY hr_ctx_pkg`プロシージャで定義されている変数`session_id_p`に格納されてい

るIDをクリアするように設定します。

- `PROCEDURE clear_hr_context ... END clear_hr_context`は、`clear_hr_context`プロシージャを作成します。このプロシージャは、`global_hr_ctx`変数によって定義された現行のユーザー・セッションのコンテキスト設定を消去します。詳細は、「セッションをクローズする際のセッション・データのクリア」を参照してください。

親トピック: グローバル・アプリケーション・コンテキストを管理するためのPL/SQLパッケージ

13.4.6.9 セッションをクローズする際のセッション・データのクリア

アプリケーション・コンテキストはメモリー内に存在するので、ユーザーがセッションを終了したときは、`client_identifier`コンテキスト値をクリアする必要があります。

この消去によってメモリーが解放され、残存している値を他のユーザーが誤って使用することがなくなります。

- ユーザーのセッション終了時にセッション・データをクリアするには、サーバー側のPL/SQLパッケージで、次のいずれかの方法を使用します。
  - ユーザーがセッションを終了したときにクライアント識別子をクリアする方法。
    `DBMS_SESSION.CLEAR_IDENTIFIER`プロシージャを使用します。たとえば:

    ```
    DBMS_SESSION.CLEAR_IDENTIFIER;
    ```

  - コンテキストをクリアしてもセッションを続行する方法。セッションを続行しながらコンテキストを消去する必要がある場合は、`DBMS_SESSION.CLEAR_CONTEXT`または`DBMS_SESSION.CLEAR_ALL_CONTEXT`プロシージャを使用します。たとえば:

    ```
    DBMS_SESSION.CLEAR_CONTEXT('my_ctx', 'my_attribute');
    ```

    `CLEAR_CONTEXT`プロシージャは現行ユーザーのコンテキストを消去します。たとえば、アプリケーション・サーバーを停止する必要がある場合など、すべてのユーザーのコンテキスト値を消去するには、`CLEAR_ALL_CONTEXT`プロシージャを使用します。

    グローバル・アプリケーション・コンテキスト値は消去されるまで使用可能です。したがって、他のセッションがこれらの値にアクセスしないように、`CLEAR_CONTEXT`または`CLEAR_ALL_CONTEXT`を使用してください。コンテキストの値が変更された場合、その内容はただちに反映され、`SYS_CONTEXT`ファンクションによって値を取得する後続のコールでは、最新の値が戻されます。

親トピック: グローバル・アプリケーション・コンテキストを管理するためのPL/SQLパッケージ

13.4.7 クライアント・セッションIDを管理するための中間層アプリケーションへのコールの埋込み

中間層アプリケーションにコールを埋め込み、クライアント・セッションIDを管理できます。

- 中間層アプリケーションを使用したクライアント・セッションIDの管理について
  アプリケーション・サーバーはクライアント・セッションIDを生成します。
- ステップ1: 中間層アプリケーションを使用したクライアント・セッションIDの取得
  ユーザーがクライアント・セッションを開始すると、アプリケーション・サーバーでクライアント・セッションIDが生成されます。
- ステップ2: 中間層アプリケーションを使用したクライアント・セッションIDの設定
  次に、中間層アプリケーションを使用してクライアント・セッションIDを設定します。
- ステップ3: 中間層アプリケーションを使用したセッション・データのクリア

アプリケーション・コンテキストは、すべてがメモリー内に存在します。

親トピック: グローバル・アプリケーション・コンテキスト

13.4.7.1 中間層アプリケーションを使用したクライアント・セッションIDの管理について

アプリケーション・サーバーはクライアント・セッションIDを生成します。

中間層アプリケーションから、クライアント・セッションIDを取得、設定、およびクリアできます。そのために、Oracle Call Interface (OCI)コールとDBMS_SESSION PL/SQLパッケージ・プロシージャのどちらかを中間層アプリケーション・コードに埋め込むことができます。

アプリケーションによってユーザーが認証され、クライアント識別子が設定されてカレント・セッションに設定されます。PL/SQLパッケージSET_CONTEXTによって、アプリケーション・コンテキストにclient_identifier値が設定されます。

関連トピック

- 非データベース・ユーザーのグローバル・アプリケーション・コンテキスト

親トピック: クライアント・セッションIDを管理するための中間層アプリケーションへのコールの埋込み

13.4.7.2 ステップ1: 中間層アプリケーションを使用したクライアント・セッションIDの取得

ユーザーがクライアント・セッションを開始すると、アプリケーション・サーバーでクライアント・セッションIDが生成されます。

このIDは、ユーザーのアクセス認証時に使用する目的で取得できます。

- このクライアントIDを取得するには、次のいずれかの文でOCIStmtExecuteコールを使用します。
  - `SELECT SYS_CONTEXT('userenv', 'client_identifier') FROM DUAL;`
  - `SELECT CLIENT_IDENTIFIER from V$SESSION;`
  - `SELECT value FROM session_context WHERE attribute='CLIENT_IDENTIFIER';`

たとえば、OCIStmtExecuteコールを使用してクライアント・セッションID値を取得するには:

```
oratext    clientid[31];
 OCIDefine  *defnp1 = (OCIDefine *) 0;
 OCIStmt    *statementhndle;
 oratext    *selcid = (oratext *)"SELECT SYS_CONTEXT('userenv',
            'client_identifier') FROM  DUAL";
OCIStmtPrepare(statementhndle, errhp, selcid,
  (ub4) strlen((char *) selcid), (ub4) OCI_NTV_SYNTAX, (ub4) OCI_DEFAULT);
OCIDefineByPos(statementhndle, &defnp1, errhp, 1, (dvoid *)clientid, 31,
  SQLT_STR, (dvoid *) 0, (ub2 *) 0, (ub2 *) 0, OCI_DEFAULT);
OCIStmtExecute(servhndle, statementhndle, errhp, (ub4) 1, (ub4) 0,
 (CONST OCISnapshot *) NULL, (OCISnapshot *) NULL, OCI_DEFAULT);
  printf("CLIENT_IDENTIFIER = %s ¥n", clientid);
```

この例では、次のようになります。

- oratext、OCIDefine、OCIStmtおよびoratextは、クライアント・セッションID、OCIDefineの参照コール、文ハンドルおよび使用するSELECT文を格納するための変数を作成します。
- OCIStmtPrepareは、文selcidを実行する準備を整えます。
- OCIDefineByPosは、クライアント・セッションIDの出力変数clientidを定義します。
- OCIStmtExecuteは、selcid変数に格納されている文を実行します。
- printfは、取得したクライアント・セッションIDに書式を設定した出力を表示します。

親トピック: クライアント・セッションIDを管理するための中間層アプリケーションへのコールの埋込み

13.4.7.3 ステップ2: 中間層アプリケーションを使用したクライアント・セッションIDの設定

次に、中間層アプリケーションを使用してクライアント・セッションIDを設定します。

- 中間層アプリケーションを使用したクライアント・セッションIDの設定について
  OCIStmtExecuteコールを使用してクライアント・セッションIDを取得した後は、このIDを設定できます。
- 中間層アプリケーションを使用したクライアント・セッションIDの設定
  Oracle Call InterfaceまたはDBMS_SESSION PL/SQLパッケージによって、中間層アプリケーションを使用してクライアント・セッションIDを設定できます。
- クライアント識別子の値のチェック
  OCIAttrSetおよびDBMS_SESSION.SET_IDENTIFIERでは、クライアント識別子の値をチェックできます。

親トピック: クライアント・セッションIDを管理するための中間層アプリケーションへのコールの埋込み

13.4.7.3.1 中間層アプリケーションを使用したクライアント・セッションIDの設定について

OCIStmtExecuteコールを使用してクライアント・セッションIDを取得した後は、このIDを設定できます。

サーバー側のPL/SQLパッケージのDBMS_SESSION.SET_CONTEXTプロシージャによってこのセッションIDを設定した後、必要に応じてアプリケーション・コンテキスト値を上書きします。

中間層アプリケーション・コードにより、クライアント・セッションID値(前の例でuser_idに書き込まれた値など)がサーバー側のDBMS_SESSION.SET_CONTEXTプロシージャで定義されているclient_id設定と一致することを確認する必要があります。アプリケーション・サーバー側では、次の順序でコールする必要があります。

1. 現行のクライアント・セッションIDを取得します。セッションにはこのIDがすでに設定されていますが、本当に正しい値かどうかを確認するほうが安全です。
2. 現行のクライアント・セッションIDをクリアします。消去することで、異なるエンド・ユーザーからのリクエストにサービスを提供するようにアプリケーションを準備します。
3. 新規クライアント・セッションIDまたはエンド・ユーザーに割当て済のクライアント・セッションIDを設定します。この設定によって、セッションは異なるグローバル・アプリケーション・コンテキストの値セットを使用するようになります。

親トピック: ステップ2: 中間層アプリケーションを使用したクライアント・セッションIDの設定

13.4.7.3.2 中間層アプリケーションを使用したクライアント・セッションIDの設定

Oracle Call InterfaceまたはDBMS_SESSION PL/SQLパッケージによって、中間層アプリケーションを使用してクライアント・セッションIDを設定できます。

- 次のいずれかの方法を使用して、アプリケーション・サーバー側にクライアント・セッションIDを設定します。
  - Oracle Call Interface。OCIAttrSet OCIコールにOCI_ATTR_CLIENT_IDENTIFIER属性を設定します。この属性は、エンド・ユーザーの識別情報を追跡するためのクライアント識別子をセッション・ハンドルに設定します。

    次の例は、OCIAttrSetでATTR_CLIENT_IDENTIFIERパラメータを使用する方法を示しています。user_id設定は、ログインしているユーザーのIDが格納されている変数を参照します。

```
OCIAttrSet((void *)session_handle, (ub4) OCI_HTYPE_SESSION,
           (void *) user_id, (ub4)strlen(user_id),
           OCI_ATTR_CLIENT_IDENTIFIER, error_handle);
```

- DBMS_SESSIONパッケージ。DBMS_SESSION.SET_IDENTIFIERプロシージャを使用して、グローバル・アプリケーション・コンテキストのクライアント識別子を設定します。たとえば、ログインしているユーザーのIDをuser_idという変数に格納している場合は、中間層アプリケーションのコードに次の行を入力することになります。

```
DBMS_SESSION.SET_IDENTIFIER(user_id);
```

ノート:

アプリケーションで CLIENT_IDENTIFIER として使用するセッション ID を生成する場合、そのセッション ID は適度にランダムで、ネットワーク上では暗号化によって保護されている必要があります。セッション ID がランダムでない場合は、不正なユーザーがセッション ID を推測し、他のユーザーのデータにアクセスする可能性があります。セッション ID がネットワーク上で暗号化されていない場合は、不正なユーザーがセッション ID を取得して、接続にアクセスする可能性があります。

ネットワーク・データの暗号化を使用して、セッション ID を暗号化できます。詳細は、Oracle Database のネイティブ・ネットワーク暗号化とデータ整合性の構成を参照してください。

親トピック: ステップ2: 中間層アプリケーションを使用したクライアント・セッションIDの設定

13.4.7.3.3 クライアント識別子の値のチェック

OCIAttrSetおよびDBMS_SESSION.SET_IDENTIFIERでは、クライアント識別子の値をチェックできます。

- クライアント識別子の値をチェックするには、次のいずれかの方法を使用します。
  - SYS_CONTEXT関数を使用してチェックするには:

```
SELECT SYS_CONTEXT('userenv', 'client_identifier') FROM DUAL;
```

  - V$SESSIONビューを問い合せてチェックするには:

```
SELECT CLIENT_IDENTIFIER from V$SESSION;
```

親トピック: ステップ2: 中間層アプリケーションを使用したクライアント・セッションIDの設定

13.4.7.4 ステップ3: 中間層アプリケーションを使用したセッション・データのクリア

アプリケーション・コンテキストは、すべてがメモリー内に存在します。

ユーザーがセッションを終了したときは、client_identifier値のコンテキストをクリアする必要があります。この消去によってメモリーが解放され、残存している値を他のユーザーが誤って使用することがなくなります。

- ユーザーのセッション終了時にセッション・データをクリアするには、中間層アプリケーションのコードで、次のいずれかの方法を使用します。
  - ユーザーがセッションを終了したときにクライアント識別子をクリアする方法。
    DBMS_SESSION.CLEAR_IDENTIFIERプロシージャを使用します。たとえば:

```
DBMS_SESSION.CLEAR_IDENTIFIER;
```

  - コンテキストをクリアしてもセッションを続行する方法。セッションを続行しながらコンテキストを消去する必要がある場合は、DBMS_SESSION.CLEAR_CONTEXTまたはDBMS_SESSION.CLEAR_ALL_CONTEXTプロ

シージャを使用します。たとえば:

```
DBMS_SESSION.CLEAR_CONTEXT(namespace, client_identifier, attribute);
```

CLEAR_CONTEXTプロシージャは現行ユーザーのコンテキストを消去します。たとえば、アプリケーション・サーバーを停止する必要がある場合など、すべてのユーザーのコンテキスト値を消去するには、CLEAR_ALL_CONTEXTプロシージャを使用します。

グローバル・アプリケーション・コンテキスト値は消去されるまで使用可能です。したがって、他のセッションがこれらの値にアクセスしないように、CLEAR_CONTEXTまたはCLEAR_ALL_CONTEXTを使用してください。

親トピック: クライアント・セッションIDを管理するための中間層アプリケーションへのコールの埋込み

13.4.8 例: クライアント・セッションIDを使用するグローバル・アプリケーション・コンテキストの作成

このチュートリアルでは、クライアント・セッションIDを使用するグローバル・アプリケーション・コンテキストの作成を示しています。

- このチュートリアルについて
  この例は、軽量ユーザーのアプリケーション用にクライアント・セッションIDを使用するグローバル・アプリケーション・コンテキストを作成する方法を示しています。
- ステップ1: ユーザー・アカウントの作成
  セキュリティ管理者はアプリケーション・コンテキストおよびそのパッケージを管理し、ユーザー・アカウントは接続プールを所有します。
- ステップ2: グローバル・アプリケーション・コンテキストの作成
  次に、グローバル・アプリケーション・コンテキストを作成します。
- ステップ3: グローバル・アプリケーション・コンテキストのパッケージの作成
  PL/SQLパッケージは、作成したグローバル・アプリケーション・コンテキストを管理します。
- ステップ4: 新規作成したグローバル・アプリケーション・コンテキストのテスト
  この時点で、このグローバル・アプリケーション・コンテキストとセッションIDの設定について動作を確認します。
- ステップ5: セッションIDの変更とグローバル・アプリケーション・コンテキストの再テスト
  次に、セッションIDをクリアして変更し、グローバル・アプリケーション・コンテキストを再テストします。
- ステップ6: このチュートリアルのコンポーネントの削除
  このチュートリアルのコンポーネントが不要になった場合、それらを削除できます。

親トピック: グローバル・アプリケーション・コンテキスト

13.4.8.1 このチュートリアルについて

この例は、軽量ユーザーのアプリケーション用にクライアント・セッションIDを使用するグローバル・アプリケーション・コンテキストを作成する方法を示しています。

非データベース・ユーザーの接続プールを使用したアクセスを制御する方法を示しています。マルチテナント環境を使用している場合、このチュートリアルは現在のPDBのみに適用されます。

親トピック: 例: クライアント・セッションIDを使用するグローバル・アプリケーション・コンテキストの作成

13.4.8.2 ステップ1: ユーザー・アカウントの作成

セキュリティ管理者はアプリケーション・コンテキストおよびそのパッケージを管理し、ユーザー・アカウントは接続プールを所有します。

1. SYSDBA管理権限を持つSYSとしてSQL*Plusにログインします。

```
sqlplus sys as sysdba
Enter password: password
```

2. マルチテナント環境で、適切なPDBに接続します。

   たとえば:

```
CONNECT SYS@my_pdb AS SYSDBA
Enter password: password
```

   利用可能なPDBを検索するには、DBA_PDBSデータ・ディクショナリ・ビューを問い合せます。現在のPDBを確認するには、show con_nameコマンドを実行します。

3. sysadmin_ctxローカル・ユーザー・アカウントを作成します。このアカウントは、グローバル・アプリケーション・コンテキストを管理します。

```
CREATE USER sysadmin_ctx IDENTIFIED BY password CONTAINER = CURRENT;
GRANT CREATE SESSION, CREATE ANY CONTEXT, CREATE PROCEDURE TO sysadmin_ctx;
GRANT EXECUTE ON DBMS_SESSION TO sysadmin_ctx;
```

   「パスワードの最低要件」のガイドラインに従って、passwordを安全なパスワードに置き換えます。

4. 接続プールの所有者にするローカル・データベース・アカウントapps_userを作成します。

```
CREATE USER apps_user IDENTIFIED BY password CONTAINER = CURRENT;
GRANT CREATE SESSION TO apps_user;
```

   passwordを安全なパスワードに置き換えます。

親トピック: 例: クライアント・セッションIDを使用するグローバル・アプリケーション・コンテキストの作成

13.4.8.3 ステップ2: グローバル・アプリケーション・コンテキストの作成

次に、グローバル・アプリケーション・コンテキストを作成します。

1. セキュリティ管理者sysadmin_ctxでログインします。

```
CONNECT sysadmin_ctx -- Or, CONNECT sysadmin_ctx@hrpdb
Enter password: password
```

2. cust_ctxグローバル・アプリケーション・コンテキストを作成します。

```
CREATE CONTEXT global_cust_ctx USING cust_ctx_pkg ACCESSED GLOBALLY;
```

   cust_ctxコンテキストが作成され、セキュリティ管理者sysadmin_ctxのスキーマに関連付けられます。ただし、アプリケーション・コンテキストはSYSスキーマが所有します。

親トピック: 例: クライアント・セッションIDを使用するグローバル・アプリケーション・コンテキストの作成

13.4.8.4 ステップ3: グローバル・アプリケーション・コンテキストのパッケージの作成

PL/SQLパッケージは、作成したグローバル・アプリケーション・コンテキストを管理します。

1. sysadmin_ctxとして、次のPL/SQLパッケージを作成します。

```
CREATE OR REPLACE PACKAGE cust_ctx_pkg
  AS
   PROCEDURE set_session_id(session_id_p IN NUMBER);
   PROCEDURE set_cust_ctx(sec_level_attr IN VARCHAR2,
     sec_level_val IN VARCHAR2);
```

```
   PROCEDURE clear_hr_session(session_id_p IN NUMBER);
   PROCEDURE clear_hr_context;
  END;
 /
CREATE OR REPLACE PACKAGE BODY cust_ctx_pkg
  AS
  session_id_global NUMBER;

 PROCEDURE set_session_id(session_id_p IN NUMBER)
  AS
  BEGIN
   session_id_global := session_id_p;
   DBMS_SESSION.SET_IDENTIFIER(session_id_p);
 END set_session_id;

 PROCEDURE set_cust_ctx(sec_level_attr IN VARCHAR2, sec_level_val IN VARCHAR2)
  AS
  BEGIN
   DBMS_SESSION.SET_CONTEXT(
    namespace  => 'global_cust_ctx',
    attribute  => sec_level_attr,
    value      => sec_level_val,
    username   => USER, -- Retrieves the session user, in this case, apps_user
    client_id  => session_id_global);
  END set_cust_ctx;

  PROCEDURE clear_hr_session(session_id_p IN NUMBER)
   AS
   BEGIN
     DBMS_SESSION.SET_IDENTIFIER(session_id_p);
     DBMS_SESSION.CLEAR_IDENTIFIER;
   END clear_hr_session;
 PROCEDURE clear_hr_context
  AS
  BEGIN
   DBMS_SESSION.CLEAR_CONTEXT('global_cust_ctx', session_id_global);
  END clear_hr_context;
 END;
/
```

このタイプのパッケージの動作の詳細は、例13-9を参照してください。

2. cust_ctx_pkgパッケージに対するEXECUTE権限を接続プール所有者apps_userに付与します。

```
GRANT EXECUTE ON cust_ctx_pkg TO apps_user;
```

親トピック: 例: クライアント・セッションIDを使用するグローバル・アプリケーション・コンテキストの作成

13.4.8.5 ステップ4: 新規作成したグローバル・アプリケーション・コンテキストのテスト

この時点で、このグローバル・アプリケーション・コンテキストとセッションIDの設定について動作を確認します。

1. 接続プール所有者(ユーザーapps_user)でSQL*Plusにログインします。

```
CONNECT apps_user -- Or, CONNECT apps_user@hrpdb
Enter password: password
```

2. 接続プール・ユーザーのログイン時に、アプリケーションでは、次のようにクライアント・セッション識別子が設定されます。

```
BEGIN
 sysadmin_ctx.cust_ctx_pkg.set_session_id(34256);
END;
/
```

3. クライアント・セッション識別子の値をテストします。

   a. セッションIDを設定します。

   ```
   EXEC sysadmin_ctx.cust_ctx_pkg.set_session_id(34256);
   ```

   b. セッションIDをチェックします。

   ```
   SELECT SYS_CONTEXT('userenv', 'client_identifier') FROM DUAL;
   ```

   次の出力が表示されます。

   ```
   SYS_CONTEXT('USERENV','CLIENT_IDENTIFIER')
   ------------------------------------------------
   34256
   ```

4. グローバル・アプリケーション・コンテキストを次のように設定します。

   ```
   EXEC sysadmin_ctx.cust_ctx_pkg.set_cust_ctx('Category', 'Gold Partner');
   EXEC sysadmin_ctx.cust_ctx_pkg.set_cust_ctx('Benefit Level', 'Highest');
   ```

   (実際には、ステップ2でクライアント・セッション識別子を設定した方法と同様に、中間層アプリケーションでグローバル・アプリケーション・コンテキスト値を設定することになります。)

5. 次のSELECT SYS_CONTEXT文を入力し、設定が正しいことをチェックします。

   ```
   col category format a13
   col benefit_level format a14
   SELECT SYS_CONTEXT('global_cust_ctx', 'Category') category,
   SYS_CONTEXT('global_cust_ctx', 'Benefit Level') benefit_level FROM DUAL;
   ```

   次の出力が表示されます。

   ```
   CATEGORY       BENEFIT_LEVEL
   -------------  --------------
   Gold Partner   Highest
   ```

apps_userがここで(クライアント・セッション34256内で)実行したのは、非データベース・ユーザーにかわってグローバル・アプリケーション・コンテキストを設定したことです。このコンテキストは、DBMS_SESSION.SET_CONTEXTのCategoryおよびBenefit Level属性をそれぞれGold PartnerおよびHighestに設定します。このコンテキストは、クライアントID 34256のユーザーapps_userに対してのみ存在します。非データベース・ユーザーがバックグラウンドでログインした場合、そのユーザーは、実際には接続プール・ユーザーapps_userでログインしたことになります。このようにして、非データベース・ユーザーは、Gold PartnerおよびHighestのコンテキスト値を使用できます。

データベース・ユーザーであるユーザーが、相応のアプリケーションを使用せずにログインすると仮定します。(たとえば、SQL*Plusを使用してログインします。)このユーザーは接続プール・ユーザー(apps_user)でログインしていないため、グローバル・アプリケーション・コンテキストは、該当しないユーザーに対して空で表示されます。これは、このコンテキストがapps_userセッションで作成および設定されていたためです。ユーザーがSELECT SYS_CONTEXT文を実行すると、次の出力が表示されます。

```
CATEGORY       BENEFIT_LEVEL
-------------  --------------
```

親トピック: 例: クライアント・セッションIDを使用するグローバル・アプリケーション・コンテキストの作成

13.4.8.6 ステップ5: セッションIDの変更とグローバル・アプリケーション・コンテキストの再テスト

次に、セッションIDをクリアして変更し、グローバル・アプリケーション・コンテキストを再テストします。

1. ユーザーapps_userとして、セッションIDをクリアします。

```
EXEC sysadmin_ctx.cust_ctx_pkg.clear_hr_session(34256);
```

2. グローバル・アプリケーション・コンテキストを再度チェックしてください。

```
SELECT SYS_CONTEXT('global_cust_ctx', 'Category') category,
SYS_CONTEXT('global_cust_ctx', 'Benefit Level') benefit_level FROM DUAL;
CATEGORY       BENEFIT_LEVEL
-------------  --------------
```

apps_userによってセッションIDがクリアされたため、グローバル・アプリケーション・コンテキストの設定は使用できません。

3. セッションIDに34256をリストアしてから、コンテキスト値をチェックします。

```
EXEC sysadmin_ctx.cust_ctx_pkg.set_session_id(34256);
SELECT SYS_CONTEXT('global_cust_ctx', 'Category') category,
SYS_CONTEXT('global_cust_ctx', 'Benefit Level') benefit_level FROM DUAL;
```

次の出力が表示されます。

```
CATEGORY       BENEFIT_LEVEL
-------------  --------------
Gold Partner   Highest
```

このように、セッションIDを34256に再設定すると、アプリケーション・コンテキスト値が元に戻ります。要約すると、グローバル・アプリケーション・コンテキストはこのユーザーに対して1度のみ設定しますが、クライアント・セッションIDはユーザーがログインするたびに設定する必要があります。

4. ここで、グローバル・アプリケーション・コンテキストの値をクリアした後、その値をチェックしてください。

```
EXEC sysadmin_ctx.cust_ctx_pkg.clear_hr_context;
SELECT SYS_CONTEXT('global_cust_ctx', 'Category') category,
SYS_CONTEXT('global_cust_ctx', 'Benefit Level') benefit_level FROM DUAL;
```

次の出力が表示されます。

```
CATEGORY       BENEFIT_LEVEL
-------------  --------------
```

この時点では、クライアント・セッションIDの34256は適切に設定されていますが、アプリケーション・コンテキストの設定はすでに存在していません。これで、以前に設定したアプリケーション・コンテキストの値を使用せずに、このユーザーのセッションを続行できます。

親トピック: 例: クライアント・セッションIDを使用するグローバル・アプリケーション・コンテキストの作成

13.4.8.7 ステップ6: このチュートリアルのコンポーネントの削除

このチュートリアルのコンポーネントが不要になった場合、それらを削除できます。

1. SYSDBA管理権限を持つSYSとして接続します。

```
CONNECT SYS AS SYSDBA -- Or, CONNECT SYS@mypdb AS SYSDBA
Enter password: password
```

2. グローバル・アプリケーション・コンテキストを削除します。

```
DROP CONTEXT global_cust_ctx;
```

このグローバル・アプリケーション・コンテキストはsysadmin_ctxが作成しましたが、SYSスキーマの所有となっていることに注意してください。

3. 2人のサンプル・ユーザーを削除します。

```
DROP USER sysadmin_ctx CASCADE;
DROP USER apps_user;
```

親トピック: 例: クライアント・セッションIDを使用するグローバル・アプリケーション・コンテキストの作成

13.4.9 グローバル・アプリケーション・コンテキスト・プロセス

簡単なグローバル・アプリケーション・コンテキストはデータベース・ユーザー・アカウントを使用し、ユーザー・セッションを作成します。グローバル・アプリケーション・コンテキストは軽量ユーザー用です。

- 単純なグローバル・アプリケーション・コンテキスト・プロセス
  簡単なグローバル・アプリケーション・コンテキストのプロセスでは、アプリケーションは、データベース・ユーザーを使用してユーザー・セッションを作成します。
- 軽量ユーザー用のグローバル・アプリケーション・コンテキスト・プロセス
  軽量ユーザーにグローバル・アプリケーション・コンテキストを設定できます。

親トピック: グローバル・アプリケーション・コンテキスト

13.4.9.1 単純なグローバル・アプリケーション・コンテキスト・プロセス

簡単なグローバル・アプリケーション・コンテキストのプロセスでは、アプリケーションは、データベース・ユーザーを使用してユーザー・セッションを作成します。

単純なグローバル・アプリケーション・コンテキスト・プロセスのコンテキスト属性の値は、SELECT文から取得できます。

クライアント識別子12345をクライアントSCOTTに割り当てたアプリケーション・サーバーAppSvrを想定します。AppSvrアプリケーションは、SCOTTユーザーを使用してセッションを作成します。(つまり、接続プールではありません。)コンテキスト属性に割り当てる値には、ユーザーの職責コードが入っている表でSELECT文を実行して取得された値など、あらゆる値を使用できます。アプリケーション・コンテキストは移入されると、メモリーに保存されます。そのため、職責コードを必要とするアクションはSYS_CONTEXTコールによってすぐに職責コードにアクセスできるようになり、表にアクセスするオーバーヘッドもありません。このケースのローカル・コンテキストに対するグローバル・コンテキストの唯一の利点は、SCOTTがアプリケーションを頻繁に変更して、各アプリケーションで同じコンテキストを使用したような場合です。

次のステップは、グローバル・アプリケーション・コンテキスト・プロセスでSCOTTのクライアント識別子を設定する方法を示しています。

1. 管理者は次の文を使用して、グローバル・コンテキスト・ネームスペースを作成します。

```
CREATE OR REPLACE CONTEXT hr_ctx USING hr.init ACCESSED GLOBALLY;
```

2. 管理者は、hr_ctxアプリケーション・コンテキスト用にPL/SQLパッケージを作成し、このクライアント識別子には、HRネームスペースに値13があるresponsibilityと呼ばれるアプリケーション・コンテキストがあることを示します。

```
CREATE OR REPLACE PROCEDURE hr.init
 AS
 BEGIN
  DBMS_SESSION.SET_CONTEXT(
    namespace => 'hr_ctx',
    attribute => 'responsibility',
    value     => '13',
    username  => 'SCOTT',
```

```
    client_id => '12345' );
 END;
/
```

このPL/SQLプロシージャはHRデータベース・スキーマに格納されていますが、通常は、セキュリティ管理者のスキーマに格納されます。

3. scottがAppSvrを使用してデータベースに接続するたびに、AppSvrアプリケーションによって、次のコマンドが発行され、接続クライアントの識別情報が指定されます。

```
EXEC DBMS_SESSION.SET_IDENTIFIER('12345');
```

4. データベース・セッション内にSYS_CONTEXT('hr_ctx','responsibility')コールがある場合、データベースによって、クライアント識別子12345がグローバル・コンテキストと照合され、値13が戻されます。

5. このデータベース・セッションの終了時には、AppSvrによって、次のプロシージャが発行され、クライアント識別子がクリアされます。

```
EXEC DBMS_SESSION.CLEAR_IDENTIFIER( );
```

6. アプリケーション・コンテキストによって使用されたメモリーを解放するために、AppSvrは次のプロシージャを発行します。

```
DBMS_SESSION.CLEAR_CONTEXT('hr_ctx', '12345');
```

CLEAR_CONTEXTが必要なのは、ユーザー・セッションが、明示的なログアウト、タイムアウトまたはAppSvrアプリケーションで判断される他の状況でアクティブでない場合です。

ノート:

セッションのクライアント識別子は、クリアされると NULL 値になります。したがって、後続の SYS_CONTEXT コールでは、SET_IDENTIFIER インタフェースを使用してクライアント識別子を再設定しないかぎり、クライアント識別子が NULL のアプリケーション・コンテキストのみが取得されます。

親トピック: グローバル・アプリケーション・コンテキスト・プロセス

13.4.9.2 軽量ユーザー用のグローバル・アプリケーション・コンテキスト・プロセス

軽量ユーザーにグローバル・アプリケーション・コンテキストを設定できます。

このアクセスは、他のユーザーがログイン時にグローバル・アプリケーション・コンテキストにアクセスできないように構成できます。

次のステップは、軽量ユーザー・アプリケーション用のグローバル・アプリケーション・コンテキスト・プロセスを示しています。軽量ユーザーrobertは、アプリケーションを介してデータベースに認識されていません。

1. 管理者は次の文を使用して、グローバル・コンテキスト・ネームスペースを作成します。

```
CREATE CONTEXT hr_ctx USING hr.init ACCESSED GLOBALLY;
```

2. HRアプリケーション・サーバーAppSvrが起動し、ユーザーappsmgrとしてHRデータベースへの複数の接続を確立します。

3. ユーザーrobertがHRアプリケーション・サーバーにログインします。

4. AppSvrがアプリケーションに対してrobertを認証します。

5. AppSvrがこの接続に対して一時的なセッションID12345を割り当てます(またはアプリケーション・ユーザーIDを使用します)。
6. セッションIDが、robertが使用するWebブラウザにCookieの一部として戻されるか、またはAppSvrによって保持されます。
7. AppSvrがhr.initパッケージをコールして、このクライアントのアプリケーション・コンテキストを初期化すると、次の文が発行されます。

```
DBMS_SESSION.SET_CONTEXT( 'hr_ctx', 'id', 'robert', 'APPSMGR', 12345 );
DBMS_SESSION.SET_CONTEXT( 'hr_ctx', 'dept', 'sales', 'APPSMGR', 12345 );
```

8. AppSvrがこのセッションにデータベース接続を割り当て、次の文を発行してセッションを初期化します。

```
DBMS_SESSION.SET_IDENTIFIER( 12345 );
```

9. このデータベース・セッション内のすべてのSYS_CONTEXTコールが、そのクライアントのセッションのみに属するアプリケーション・コンテキストの値を戻します。

   たとえば、SYS_CONTEXT('hr','id')は、robertという値を戻します。

10. セッションが終了すると、AppSvrは次の文を発行してクライアントの識別情報を消去します。

```
DBMS_SESSION.CLEAR_IDENTIFIER ( );
```

他のユーザーがデータベースにログインした場合でも、このユーザーはAppSvrによって設定されたグローバル・コンテキストにはアクセスできません。これは、AppSvrが、ユーザーAPPSMGRがログインしたアプリケーションのみがこのグローバル・コンテキストを認識できるように指定したためです。AppSvrが次の文を使用した場合、クライアントIDが12345に設定されたあらゆるユーザー・セッションがグローバル・コンテキストを認識できるようになります。

```
DBMS_SESSION.SET_CONTEXT( 'hr_ctx', 'id', 'robert', NULL , 12345 );
DBMS_SESSION.SET_CONTEXT( 'hr_ctx', 'dept', 'sales', NULL , 12345 );
```

USERNAMEをNULLに設定すると、異なるユーザーが同一のコンテキストを共有できます。

ノート:

グローバル・コンテキストの設定が異なると、セキュリティ上の意味が変わることを認識する必要があります。ユーザー名に NULL が指定されている場合は、すべてのユーザーがそのグローバル・コンテキストにアクセスできます。グローバル・コンテキストのクライアント ID に NULL が指定されている場合、そのグローバル・コンテキストにアクセスできるのは、初期化されていないクライアント ID を持つセッションです。セッションへのアクセスをログインしたユーザーのみに制限するには、NULL のかわりに USER を指定します。

セッションに設定されたクライアント識別子は、次のように問い合せることができます。

```
SELECT SYS_CONTEXT('USERENV','CLIENT_IDENTIFIER') FROM DUAL;
```

次の出力が表示されます。

```
SYS_CONTEXT('USERENV','CLIENT_IDENTIFIER')
------------------------------------------------
12345
```

セキュリティ管理者は、V$SESSIONビューのCLIENT_IDENTIFIERおよびUSERNAMEを問い合せることで、どのセッションに

クライアント識別子が設定されているかを確認できます。たとえば:

```
COL client_identifier format a18
SELECT CLIENT_IDENTIFIER, USERNAME from V$SESSION;
```

次の出力が表示されます。

```
CLIENT_IDENTIFIER   USERNAME
------------------  --------
12345               APPSMGR
```

グローバル・コンテキストの使用領域(バイト単位)をチェックするには、次の問合せを使用します。

```
SELECT SYS_CONTEXT('USERENV','GLOBAL_CONTEXT_MEMORY') FROM DUAL;
```

次の出力が表示されます。

```
SYS_CONTEXT('USERENV','GLOBAL_CONTEXT_MEMORY')
----------------------------------------------
584
```

関連項目:

USERENVアプリケーション・コンテキストの事前定義属性であるCLIENT_IDENTIFIERの使用の詳細は、次を参照してください。

- CLIENT_IDENTIFIER属性を使用したユーザー識別情報の保持
- Oracle Database SQL言語リファレンス
- Oracle Call Interfaceプログラマーズ・ガイド

親トピック: グローバル・アプリケーション・コンテキスト・プロセス

13.5 クライアント・セッション・ベースのアプリケーション・コンテキストの使用

クライアント・セッション・ベースのアプリケーション・コンテキストは、ユーザー・グローバル領域(UGA)に格納されます。

- クライアント・セッション・ベースのアプリケーション・コンテキストについて
  Oracle Call Interface (OCI)関数で、ユーザー・グローバル領域(UGA)のユーザー・セッション情報を設定およびクリアできます。
- CLIENTCONTEXTネームスペースへの値の設定
  Oracle Call Interface (OCI)で、CLIENTCONTEXTネームスペースを設定できます。
- CLIENTCONTEXTネームスペースの取得
  Oracle Call Interfaceを使用して、CLIEINTCONTEXTネームスペースを取得できます。
- 例: クライアント・セッション・ベース・コンテキストのクライアント・セッションID値の取得
  OCI OCIStmtExecuteコールで、クライアント・セッション・ベース・コンテキストのクライアント・セッションID値を取得できます。
- CLIENTCONTEXTネームスペースの設定のクリア
  Oracle Call Interfaceを使用して、CLIENTCONTEXTネームスペースをクリアできます。
- CLIENTCONTEXTネームスペースのすべての設定のクリア
  Oracle Call Interface (OCI)を使用して、CLIENTCONTEXTネームスペースをクリアできます。

親トピック: アプリケーション・コンテキストを使用したユーザー情報の取得

13.5.1 クライアント・セッション・ベースのアプリケーション・コンテキストについて

Oracle Call Interface (OCI)関数で、ユーザー・グローバル領域(UGA)のユーザー・セッション情報を設定およびクリアできます。

セッション・ベース・アプリケーション・コンテキストのこのタイプのアプリケーション・コンテキストの利点は、個々のアプリケーションが特定の非データベース・ユーザー・セッション・データを確認でき、このタスクをデータベースで実行しないことです。もう1つの利点は、アプリケーション・コンテキスト値を設定するコールはサーバーに対する次のコールに含まれるため、パフォーマンスが向上するという点です。

ただし、アプリケーション・コンテキストは、クライアント・セッション・ベースのアプリケーション・コンテキストではセキュリティを維持できなくなる点に注意が必要です。具体的には、アプリケーション・ユーザーがクライアント・アプリケーション・コンテキストを設定できるようになり、データベースでのチェックが一切実行されません。

クライアント・セッション・ベースのアプリケーション・コンテキストは、クライアント・アプリケーションに対してのみ構成します。クライアントの接続先データベース・サーバーには、設定を構成しません。データベース・サーバーのアプリケーション・コンテキスト設定は、クライアント・セッション・ベースのアプリケーション・コンテキストに影響を与えません。

クライアント・セッション・ベースのアプリケーション・コンテキストを構成するには、OCI関数`OCIAppCtxSet`を使用します。`CLIENTCONTEXT`ネームスペースを使用するクライアント・セッション・ベースのアプリケーション・コンテキストは、OCIクライアントまたはアプリケーション・コンテキストの既存の`DBMS_SESSION`パッケージで更新できます。このタイプのコンテキストでは、権限またはパッケージ・セキュリティのチェックは実行されません。

`CLIENTCONTEXT`ネームスペースを使用すると、1つのアプリケーション・トランザクションで、ユーザー・コンテキスト情報を変更することと、同じユーザー・セッション・ハンドルを使用して新規ユーザー・リクエストに対応することの両方が可能になります。`CLIENTCONTEXT`ネームスペースで属性の個々の値を設定またはクリアすることも、すべての値をクリアすることもできます。

- OCIクライアントでは`OCIAppCtx`関数を使用して、ネームスペース`OCISessionHandle`の可変長データを設定します。OCIネットワークの単一ラウンドトリップ転送は、1回のラウンドトリップで全情報をサーバーに送信します。サーバー側では、ネームスペースに対して`SYS_CONTEXT` SQL関数を使用することで、アプリケーション・コンテキスト情報を問い合せることができます。たとえば:
- JDBCクライアントでは、`oracle.jdbc.internal.OracleConnection`関数を使用してこれと同じことを実行します。

`CLIENTCONTEXT`ネームスペースはパッケージ・ベースのセキュリティで保護されていないため、すべてのユーザーがこのネームスペース内の情報を設定、クリアまたは収集できます。

関連項目:

クライアント・アプリケーション・コンテキストの詳細は、『Oracle Call Interfaceプログラマーズ・ガイド』を参照してください。

親トピック: クライアント・セッション・ベースのアプリケーション・コンテキストの使用

13.5.2 CLIENTCONTEXTネームスペースへの値の設定

Oracle Call Interface (OCI)で、`CLIENTCONTEXT`ネームスペースを設定できます。

- `CLIENTCONTEXT`ネームスペースに値を設定するには、次の構文で`OCIAppCTXSet`コマンドを使用します。

```
err = OCIAppCtxSet((void *) session_handle,(dvoid *)"CLIENTCONTEXT",(ub4) 13,
                   (dvoid *)attribute_name, length_of_attribute_name
                   (dvoid *)attribute_value, length_of_attribute_value, errhp,
                   OCI_DEFAULT);
```

詳細は、次のとおりです。

- session_handleは、OCISessionHandleネームスペースを表します。
- attribute_nameは、属性の名前です。たとえば、長さ14のresponsibilityを指定します。
- attribute_valueは、属性の値です。たとえば、長さ7のmanagerを指定します。

関連項目:

OCIAppCtxファンクションの詳細は、Oracle Call Interfaceプログラマーズ・ガイドを参照してください

親トピック: クライアント・セッション・ベースのアプリケーション・コンテキストの使用

13.5.3 CLIENTCONTEXTネームスペースの取得

Oracle Call Interfaceを使用して、CLIEINTCONTEXTネームスペースを取得できます。

- CLIENTCONTEXTネームスペースを取得するには、次のいずれかの文でOCIStmtExecuteコールを使用します。
  - SELECT SYS_CONTEXT('CLIENTCONTEXT', 'attribute-1') FROM DUAL;
  - SELECT VALUE FROM SESSION_CONTEXT WHERE NAMESPACE='CLIENTCONTEXT' AND ATTRIBUTE='attribute-1';

attribute-1の値は、CLIENTCONTEXTネームスペースですでに設定されているどんな属性値でも構いません。Oracle Databaseは設定された属性のみを取得します。設定済属性がない場合は、NULLを返します。通常、属性を設定するにはOCIAppCtxSetコールを使用します。さらに、DBMS_SESSION.SET_CONTEXTコールをOCIコードに埋め込んで属性値を設定できます。

親トピック: クライアント・セッション・ベースのアプリケーション・コンテキストの使用

13.5.4 例: クライアント・セッション・ベース・コンテキストのクライアント・セッションID値の取得

OCI OCIStmtExecuteコールで、クライアント・セッション・ベース・コンテキストのクライアント・セッションID値を取得できます。

例13-10に、OCIStmtExecuteコールを使用してクライアント・セッションIDを取得する方法を示します。

例13-10 クライアント・セッション・ベース・コンテキストのクライアント・セッションID値の取得

```
oratext    clientid[31];
OCIDefine  *defnp1 = (OCIDefine *) 0;
OCIStmt    *statementhndle;
oratext    *selcid = (oratext *)"SELECT SYS_CONTEXT('CLIENTCONTEXT',
            attribute) FROM  DUAL";

OCIStmtPrepare(statementhndle, errhp, selcid, (ub4) strlen((char *) selcid),
  (ub4) OCI_NTV_SYNTAX, (ub4) OCI_DEFAULT);

OCIDefineByPos(statementhndle, &defnp1, errhp, 1, (dvoid *)clientid, 31,
  SQLT_STR, (dvoid *) 0, (ub2 *) 0, (ub2 *) 0, OCI_DEFAULT);

OCIStmtExecute(servhndle, statementhndle, errhp, (ub4) 1, (ub4) 0,
```

```
  (CONST OCISnapshot *) NULL, (OCISnapshot *) NULL, OCI_DEFAULT);
 printf("CLIENT_IDENTIFIER = %s ¥n", clientid);
```

この例では、次のようになります。

- oratext、OCIDefine、OCIStmtおよびoratextは、クライアント・セッションID、OCIDefineの参照コール、文ハンドルおよび使用するSELECT文を格納するための変数を作成します。
- OCIStmtPrepareは、文selcidを実行する準備を整えます。
- OCIDefineByPosは、クライアント・セッションIDの出力変数clientidを定義します。
- OCIStmtExecuteは、selcid変数に格納されている文を実行します。
- printfは、取得したクライアント・セッションIDに書式を設定した出力を表示します。

親トピック: クライアント・セッション・ベースのアプリケーション・コンテキストの使用

13.5.5 CLIENTCONTEXTネームスペースの設定のクリア

Oracle Call Interfaceを使用して、CLIENTCONTEXTネームスペースをクリアできます。

- CLIENTCONTEXTの設定をクリアするには、次のいずれかのコマンドを使用して、値をNULLまたは空の文字列に設定します。
  - 次のコマンドは、空の文字列をゼロに設定します。

```
(void) OCIAppCtxSet((void *) session_handle, (dvoid *)"CLIENTCONTEXT", 13,
                    (dvoid *)attribute_name, length_of_attribute_name,
                    (dvoid *)0, 0,errhp
                    OCI_DEFAULT);
```

  - 次のコマンドは、空の文字列を空白値に設定します。

```
(void) OCIAppCtxSet((void *) session_handle, (dvoid *)"CLIENTCONTEXT", 13
                    (dvoid *)attribute_name, length_of_attribute_name,
                    (dvoid *)"", 0,errhp,
                    OCI_DEFAULT);
```

親トピック: クライアント・セッション・ベースのアプリケーション・コンテキストの使用

13.5.6 CLIENTCONTEXTネームスペースのすべての設定のクリア

Oracle Call Interface (OCI)を使用して、CLIENTCONTEXTネームスペースをクリアできます。

- ネームスペースを消去するには、次の構文でOCIAppCtxClearAllコマンドを使用します。

```
err = OCIAppCtxClearAll((void *) session_handle,
                        (dvoid *)"CLIENTCONTEXT", 13,
                         errhp,                          OCI_DEFAULT);
```

親トピック: クライアント・セッション・ベースのアプリケーション・コンテキストの使用

13.6 アプリケーション・コンテキストのデータ・ディクショナリ・ビュー

Oracle Databaseには、アプリケーション・コンテキストに関する情報を提供するデータ・ディクショナリ・ビューが用意されています。

表13-3に、これらのデータ・ディクショナリ・ビューを示します。

表13-3 アプリケーション・コンテキストに関する情報を表示するデータ・ディクショナリ・ビュー

ビュー	説明
ALL_CONTEXT	属性および値が DBMS_SESSION.SET_CONTEXT プロシージャを使用して指定された、カレント・セッション内のすべてのコンテキスト・ネームスペースが表示されます。ネームスペースと、対応付けられているスキーマおよび PL/SQL パッケージがリストされます。
ALL_POLICY_CONTEXTS	現行ユーザーがアクセス可能なシノニム、表およびビューに定義されている駆動コンテキストが表示されます。(駆動コンテキストは、仮想プライベート・データベース・ポリシーで使用されるコンテキストです。)
DBA_CONTEXT	データベース内のすべてのコンテキスト・ネームスペース情報が表示されます。このビューの列は、TYPE 列が含まれていること以外は ALL_CONTEXT ビューの各列と同じです。TYPE 列には、アプリケーション・コンテキストのアクセスまたは初期化の方法が表示されます。
DBA_OBJECTS	既存のアプリケーション・コンテキストの名前が表示されます。次のように、DBA_OBJECTS ビューの OBJECT_TYPE 列を問い合せます。 `SELECT OBJECT_NAME FROM DBA_OBJECTS WHERE OBJECT_TYPE ='CONTEXT';`
DBA_POLICY_CONTEXTS	DBMS_RLS.ADD_POLICY_CONTEXT プロシージャによってデータベースに追加されたすべての駆動コンテキストが表示されます。このビューの列は、ALL_POLICY_CONTEXTS の各列と同じです。
SESSION_CONTEXT	カレント・セッションに設定されているコンテキスト属性とその値が表示されます。
USER_POLICY_CONTEXTS	現行ユーザーが所有するシノニム、表およびビューに定義されている駆動コンテキストが表示されます。このビューの列は、OBJECT_OWNER 列以外は ALL_POLICY_CONTEXTS の各列と同じです。
V$CONTEXT	カレント PDB セッションに設定されている属性がリストされます。このビューに対する SELECT 権限が付与されていないユーザーは、このビューにアクセスできません。
V$SESSION	各カレント PDB セッションに関する詳細情報がリストされます。このビューに対する SELECT 権限が付与されていないユーザーは、このビューにアクセスできません。

ヒント:

アプリケーション・コンテキストを使用するアプリケーションの実行時にエラーが発生した場合は、これらのビューに加え、

データベース・トレース・ファイルも確認してください。USER_DUMP_DEST 初期化パラメータは、トレース・ファイルのディレクトリの場所を示します。このパラメータの値は、SQL*Plus で SHOW PARAMETER USER_DUMP_DEST を発行して確認できます。

関連トピック

- Oracle Databaseリファレンス
- Oracle Database SQLチューニング・ガイド

親トピック: アプリケーション・コンテキストを使用したユーザー情報の取得

14 Oracle Virtual Private Databaseを使用したデータ・アクセスの制御

Oracle Virtual Private Database (VPD)を使用すると、データにアクセスするユーザーをフィルタ処理できます。

- Oracle Virtual Private Databaseについて
  Oracle Virtual Private Database (VPD)には、ユーザーによるデータへのアクセスをフィルタ処理する場合の重要なメリットがあります。
- Oracle Virtual Private Databaseポリシーのコンポーネント
  VPDポリシーは関数を使用して動的WHERE句を生成し、ポリシーを使用して保護するオブジェクトに関数を付加します。
- Oracle Virtual Private Databaseのポリシーの構成
  DBMS_RLS PL/SQLパッケージで、Oracle Virtual Private Database (VPD)ポリシーを構成できます。
- 例: Oracle Virtual Private Databaseポリシーの作成
  このチュートリアルでは、簡単なデータベース・セッション・ベースのOracle Virtual Privateポリシーの作成方法と、ポリシー・グループの作成方法を説明します。
- 他のOracle機能でのOracle Virtual Private Databaseの使用
  Oracle Virtual Private DatabaseをOracleの他の機能と併用することの影響を理解しておく必要があります。
- Oracle Virtual Private Databaseのデータ・ディクショナリ・ビュー
  Oracle Databaseには、Oracle Virtual Private Databaseポリシーに関する情報を表示するデータ・ディクショナリ・ビューが用意されています。

親トピック: データへのアクセス制御

14.1 Oracle Virtual Private Databaseについて

Oracle Virtual Private Database (VPD)には、ユーザーによるデータへのアクセスをフィルタ処理する場合の重要なメリットがあります。

- Oracle Virtual Private Database
  Oracle Virtual Private Database(VPD)で、行および列レベルでデータベース・アクセスを制御するセキュリティ・ポリシーを作成します。
- Oracle Virtual Private Databaseポリシーを使用するメリット
  Oracle Virtual Private Databaseポリシーには、重要なメリットがあります。
- Oracle Virtual Private Databaseポリシーの作成者とは
  DBMS_RLS PL/SQLパッケージでは、VPDポリシーを作成できます。
- Oracle Virtual Private Databaseポリシー関数を実行するための権限
  Oracle Virtual Private Database (VPD)ポリシー関数を実行するには、正しい権限を使用する必要があります。
- Oracle Virtual Private Databaseでのアプリケーション・コンテキストの使用
  Oracle Virtual Private Databaseポリシーを使ったアプリケーション・コンテキストを使用できます。
- マルチテナント環境でのOracle Virtual Private Database
  アプリケーション・ルートで仮想プライベート・データベース・ポリシーを作成して、関連付けられたすべてのアプリケーションPDBで使用できます。

親トピック: Oracle Virtual Private Databaseを使用したデータ・アクセスの制御

14.1.1 Oracle Virtual Private Database

Oracle Virtual Private Database(VPD)で、行および列レベルでデータベース・アクセスを制御するセキュリティ・ポリシーを作成します。

ノート:

Oracle Database リリース 12c では、VPD にかわって Real Application Security (RAS)が採用されました。アプリケーションで行レベルおよび列レベルのアクセス制御が必要な新規プロジェクトには RAS を使用することをお薦めします。

基本的には、Oracle Virtual Private Databaseのセキュリティ・ポリシーが適用された表、ビューまたはシノニムに対して発行されるSQL文に、動的なWHERE句が追加されます。

Oracle Virtual Private Databaseを使用すると、データベース表、ビューまたはシノニムに対するセキュリティを直接詳細なレベルまで規定できます。これらのデータベース・オブジェクトにセキュリティ・ポリシーを直接付加すると、ユーザーがデータにアクセスするたびにポリシーが自動的に適用されるため、セキュリティを回避できません。

ユーザーがOracle Virtual Private Databaseポリシーで保護されている表、ビューまたはシノニムに直接的または間接的にアクセスすると、Oracle Databaseはユーザーの SQL文を動的に変更します。この変更は、セキュリティ・ポリシーを実装する関数によって戻されたWHERE条件(述語)に基づいて行われます。Oracle Databaseでは、関数内に記述された条件、または関数が戻す条件を使用して、動的かつユーザーに対して透過的に文が変更されます。Oracle Virtual Private Database ポリシーは、SELECT、INSERT、UPDATE、INDEXおよびDELETE文に適用できます。

たとえば、ユーザーが次の問合せを実行するとします。

```
SELECT * FROM OE.ORDERS;
```

Oracle Virtual Private Databaseポリシーにより、WHERE句の文が動的に追加されます。たとえば:

```
SELECT * FROM OE.ORDERS
 WHERE SALES_REP_ID = 159;
```

この例では、ユーザーは営業担当者159の受注のみを表示できます。

このユーザーのセッション情報(ユーザーのIDなど)に基づいてユーザーをフィルタ処理する場合は、WHERE句を作成してアプリケーション・コンテキストを使用できます。たとえば:

```
SELECT * FROM OE.ORDERS
 WHERE SALES_REP_ID = SYS_CONTEXT('USERENV','SESSION_USER');
```

ノート:

Oracle Virtual Private Database では、DDL(TRUNCATE 文や ALTER TABLE 文)のフィルタ処理はサポートされていません。

関連トピック

- Oracle Virtual Private Databaseの述語の監査

親トピック: Oracle Virtual Private Databaseについて

14.1.2 Oracle Virtual Private Databaseポリシーを使用するメリット

Oracle Virtual Private Databaseポリシーには、重要なメリットがあります。

- アプリケーションではなくデータベース・オブジェクトに基づくセキュリティ・ポリシー
  Oracle Virtual Private Databaseにはセキュリティ、簡易性、柔軟性という利点があります。
- Oracle Databaseによるポリシー関数の評価方法の制御
  ポリシー関数を複数回実行すると、パフォーマンスに影響を与える可能性があります。

親トピック: Oracle Virtual Private Databaseについて

14.1.2.1 アプリケーションではなくデータベース・オブジェクトに基づくセキュリティ・ポリシー

Oracle Virtual Private Databaseにはセキュリティ、簡易性、柔軟性という利点があります。

すべてのアプリケーションでアクセス制御を実装するのではなく、Oracle Virtual Private Databaseセキュリティ・ポリシーをデータベース表、ビューまたはシノニムに付加すると、次のようなメリットがあります。

- セキュリティ。データベース表、ビューまたはシノニムにポリシーを対応付けることで、アプリケーション・セキュリティの重大な問題を解決できます。たとえば、アプリケーションの使用を許可されているユーザーが、そのアプリケーションに対応付けられている権限を利用し、SQL*Plusなどの非定型の問合せツールを使用してデータベースを誤って変更してしまう可能性があります。ファイングレイン・アクセス・コントロールでは、セキュリティ・ポリシーを表、ビューまたはシノニムに直接付加することによって、ユーザーがどのような方法でデータにアクセスしても、同じセキュリティが施行されます。
- 簡潔性。セキュリティ・ポリシーは、表ベース、ビュー・ベースまたはシノニム・ベースのアプリケーションごとに繰り返し追加するのではなく、表、ビューまたはシノニムに1回のみ追加します。
- 柔軟性。SELECT文には1つのセキュリティ・ポリシーを、INSERT文には別のポリシーを、さらにUPDATE文およびDELETE文にはまた別のポリシーを指定できます。たとえば、人事部門の担当者には、その部門内のすべての社員のレコードに対するSELECT権限を付与し、名字がAからFで始まるその部門内の社員の給与のみを更新できるように指定できます。さらに、各表、ビューまたはシノニムに対して複数のポリシーを作成できます。

親トピック: Oracle Virtual Private Databaseポリシーを使用するメリット

14.1.2.2 Oracle Databaseによるポリシー関数の評価方法の制御

ポリシー関数を複数回実行すると、パフォーマンスに影響を与える可能性があります。

Oracle DatabaseがOracle Virtual Private Database述語をキャッシュする方法を構成することによって、ポリシー関数のパフォーマンスを制御できます。

次のオプションを使用できます。

- 各問合せについてポリシーを1回評価します(静的ポリシー)。
- ポリシー関数内のアプリケーション・コンテキストが変更された場合のみ、ポリシーを評価します(状況依存ポリシー)。
- 実行ごとにポリシーを評価します(動的ポリシー)。

関連トピック

- Oracle Virtual Private Databaseポリシー・タイプを使用したパフォーマンスの最適化

親トピック: Oracle Virtual Private Databaseポリシーを使用するメリット

14.1.3 Oracle Virtual Private Databaseポリシーの作成者とは

DBMS_RLS PL/SQLパッケージでは、VPDポリシーを作成できます。

DBMS_RLS PL/SQLパッケージのEXECUTE権限が付与されたユーザーは、Oracle Virtual Private Databaseポリシーを作成できます。すべての権限と同様、この権限は信頼できるユーザーにのみ付与してください。ユーザーに付与された権限を確認するには、DBA_SYS_PRIVSデータ・ディクショナリ・ビューに問い合せます。

親トピック: Oracle Virtual Private Databaseについて

14.1.4 Oracle Virtual Private Databaseポリシー関数を実行するための権限

Oracle Virtual Private Database (VPD)ポリシー関数を実行するには、正しい権限を使用する必要があります。

セキュリティを強化するために、Oracle Virtual Private Database ポリシー関数は、定義者権限で宣言されたかのように実行されます。

実行者権限として宣言すると、自分自身だけでなく、コードをメンテナンスする他のユーザーも混乱させてしまうため、実行者権限では宣言しないでください。

関連項目:

定義者権限の詳細は、『Oracle Database PL/SQL言語リファレンス』を参照してください。

親トピック: Oracle Virtual Private Databaseについて

14.1.5 Oracle Virtual Private Databaseでのアプリケーション・コンテキストの使用

Oracle Virtual Private Databaseポリシーを使ったアプリケーション・コンテキストを使用できます。

アプリケーション・コンテキストを作成すると、ユーザー情報が安全にキャッシュされます。キャッシュ環境を設定できるのは、指定されたアプリケーション・パッケージのみです。ユーザーやパッケージ外部からの変更はできません。さらに、データがキャッシュされるため、パフォーマンスが向上します。

たとえば、ORDERS_TAB表へのアクセスを顧客ID番号を基にして行うとします。顧客ID番号が必要になるたびにログインしたユーザーに問い合せるのではなく、アプリケーション・コンテキスト内に顧客ID番号を格納しておくことができます。このようにすると、顧客番号は必要なときにセッションで使用できます。

アプリケーション・コンテキストは、セキュリティ・ポリシーが複数のセキュリティ属性に基づく場合に特に役立ちます。たとえば、WHERE述語が4つの属性(従業員番号、コスト・センター、職位、支出制限など)に基づくポリシー関数は、この情報を取得するために複数の副問合せを実行する必要があります。このデータがアプリケーション・コンテキストを介して使用可能な場合、パフォーマンスは大きく向上します。

アプリケーション・コンテキストを使用すると、述語により規定される正しいセキュリティ・ポリシーを戻すことができます。たとえば、「顧客は自分の注文のみを参照でき、社員はすべての顧客のすべての注文を参照できる」というルールを規定している受注管理アプリケーションを想定します。この場合は、2つの異なるポリシーがあります。position属性でアプリケーション・コンテキストを定義できます。この属性はポリシー関数内でアクセスでき、その属性の値に基づいて正しい述語を戻します。そのため、職位がclerk (社員)であるユーザーはすべての注文を取得できるようにし、customer (顧客)であるユーザーは自分に関連するレコードのみ参照できるようにすることができます。

属性に対して特定の述語を戻すファイングレイン・アクセス・コントロール・ポリシーを設計するには、ポリシーを実装する関数内でアプリケーション・コンテキストにアクセスする必要があります。たとえば、顧客が自分のレコードのみを参照するように制限する必要があるとします。ユーザーは次の問合せを実行します。

```
SELECT * FROM orders_tab
```

ファイングレイン・アクセス・コントロールによって、この問合せはWHERE述語が含まれるように動的に変更されます。

```
SELECT * FROM orders_tab
  WHERE custno = SYS_CONTEXT ('order_entry', 'cust_num');
```

前述の例で、50,000人の顧客が存在し、各顧客が受け取る述語を同じにするとします。すべての顧客は同一のWHERE述語を共有し、自分の注文のみを参照できます。顧客間で異なるのは、顧客番号のみです。

アプリケーション・コンテキストを使用して、50,000人の顧客に適用されるポリシー関数内で1つのWHERE述語を返すことができます。結果として、顧客番号が実行時に評価されるため、顧客ごとに実行方法が異なる1つの共有カーソルが存在することになります。この値は、すべての顧客で異なります。このようにアプリケーション・コンテキストを使用することで、最適なパフォーマンスと行レベルのセキュリティが実現します。

SYS_CONTEXT関数がバインド変数のように動作するのは、SYS_CONTEXT引数が定数の場合のみです。

関連トピック

- アプリケーション・コンテキストを使用したユーザー情報の取得

親トピック: Oracle Virtual Private Databaseについて

14.1.6 マルチテナント環境でのOracle Virtual Private Database

アプリケーション・ルートで仮想プライベート・データベース・ポリシーを作成して、関連付けられたすべてのアプリケーションPDBで使用できます。

CDBの制限は、共有の状況依存ポリシーのほか、仮想プライベート・データベース・ポリシー関連のビューにも適用されます。マルチテナント環境全体に仮想プライベート・データベース・ポリシーを作成することはできません。

アプリケーション・コンテナについては、仮想プライベート・データベース・ポリシーを作成して、アプリケーション・ルートに属するすべてのPDBに共通ポリシーを適用することで、アプリケーション共通オブジェクトを保護できます。つまり、アプリケーション・ルートにアプリケーションをインストールすると、共通オブジェクトを保護するすべての共通仮想プライベート・データベース・ポリシーは、アプリケーション・コンテナのすべてのPDBに適用され、即座に実施されます。

次のことに注意してください。

- アプリケーション・ルートでは共通仮想プライベート・データベース・ポリシーおよびそれに関連するPL/SQLファンクションのみを作成し、それをアプリケーション共通オブジェクトに付加することができます。ファンクションがポリシーと同じ場所にない場合、実行時にエラーが発生します。
- 共通オブジェクトに適用される仮想プライベート・データベース・ポリシーは、アプリケーションPDBからアプリケーション共通オブジェクトにアクセスする場合、アプリケーション・コンテナに属するPDBで自動的に強制される共通ポリシーと見なされます。
- アプリケーション共通仮想プライベート・データベース・ポリシーは、アプリケーション共通オブジェクトのみを保護できます。
- アプリケーション・ルートのアプリケーション共通オブジェクトに適用され、すべてのアプリケーションPDBに適用される仮想プライベート・データベース・ポリシーは、共通仮想プライベート・データベース・ポリシーと見なされます。ローカル・データベース表に適用され、1つのPDBで実施されるポリシーは、ローカル仮想プライベート・データベース・ポリシーと見なされ

ます。

たとえば、ポリシーVPD_P1がアプリケーション・ルートのアプリケーション共通表T1に適用される場合は、共通ポリシーと見なされます。このポリシーは各アプリケーションPDBに強制されます。VPD_P1というポリシーがPDB1のT1というローカル表に適用される場合は、ローカル・ポリシーと見なされ、PDB1のみがその影響を受けます。VPD_P1というポリシーがアプリケーション・ルートのT1というローカル表に適用される場合も、アプリケーション・ルートのみが影響を受けるため、ローカル・ポリシーと見なされます。この概念は、仮想プライベート・データベース・ポリシーの有効化、無効化および削除などの他の操作にも適用されます。

- アプリケーション共通仮想プライベート・データベース・ポリシーはアプリケーション共通オブジェクトのみを保護し、ローカル仮想プライベート・データベース・ポリシーはローカル・オブジェクトのみを保護します。
- アプリケーション・コンテキストを使用している場合は、共通データベース・セッション・ベースのアプリケーション・コンテキストおよび共通グローバル・アプリケーション・コンテキスト・オブジェクトを共通仮想プライベート・データベース構成で使用するようにしてください。
- アプリケーション・コンテナの仮想プライベート・データベース・ポリシーは、アプリケーション・ルートに格納されます。PDBにはローカル・ポリシーのみが格納されます。PDBをアプリケーション・コンテナに接続する場合、共通ポリシーはローカル・ポリシーに変換されません。かわりに、Oracle Databaseがアプリケーション・ルートからそれらをロードし、ポリシーからローカルPDBの共通オブジェクトにアクセスがあると、ローカルPDBでそれらを強制します。

親トピック: Oracle Virtual Private Databaseについて

14.2 Oracle Virtual Private Databaseポリシーのコンポーネント

VPDポリシーは関数を使用して動的WHERE句を生成し、ポリシーを使用して保護するオブジェクトに関数を付加します。

- 動的なWHERE句を生成する関数
  Oracle Virtual Private Database (VPD)関数で、適用する制限を定義します。
- 保護するオブジェクトに関数を付加するポリシー
  Oracle Virtual Private Databaseポリシーで、VPD関数を表やビュー、シノニムに関連付けます。

親トピック: Oracle Virtual Private Databaseを使用したデータ・アクセスの制御

14.2.1 動的なWHERE句を生成する関数

Oracle Virtual Private Database (VPD)関数で、適用する制限を定義します。

Oracle Virtual Private Database (VPD)の動的なWHERE句(述語)を生成するには、これらの制限を定義した関数(プロシージャではなく)を作成する必要があります。この関数は、定義者権限関数です。

通常は、セキュリティ管理者が自分のスキーマにこの関数を作成します。別の関数のコールを含めたり、ログイン失敗を追跡するためのチェックを追加するなど、複雑な動作が必要な場合は、これらの関数を1つのパッケージ内に作成します。

関数では、次の動作が必要です。

- 引数としてスキーマ名、入力としてオブジェクト(表、ビューまたはシノニム)名を取る必要があります。これらの情報を保持するように入力パラメータを定義しますが、関数内でスキーマ名とオブジェクト名自体を指定しないでください。DBMS_RLSパッケージ(保護するオブジェクトに関数を付加するポリシーを参照)を使用して作成したポリシーによって、スキーマ名、およびポリシーが適用されるオブジェクトが提供されます。最初にスキーマのパラメータを作成し、その後にオブジェクトのパラメータを作成する必要があります。

- 生成するWHERE句述語の戻り値を提供する必要があります。WHERE句の戻り値は、常にVARCHAR2データ型です。
- 有効なWHERE句を生成する必要があります。このコードは、例: 単純なOracle Virtual Private Databaseポリシーの作成に示すように基本的なもので、WHERE句はログインするすべてのユーザーに共通です。

  ただし、ほとんどの場合、各ユーザー、ユーザーの各グループ、または保護するオブジェクトにアクセスする各アプリケーションに異なるWHERE句を設計する必要があります。たとえば、マネージャがログインする場合は、そのマネージャの権限に固有のWHERE句を生成できます。これを行うには、ユーザー・セッション情報にアクセスするアプリケーション・コンテキストをWHERE句の生成コードに組み込みます。チュートリアル: セッション・ベースのアプリケーション・コンテキスト・ポリシーの実装では、アプリケーション・コンテキストを使用するOracle Virtual Private Databaseポリシーの作成方法について説明します。

  アプリケーション・コンテキストを使用しないOracle Virtual Private Database関数を作成することもできますが、アプリケーション・コンテキストを使用すると、ユーザーのセッション属性(ユーザーIDなど)に基づいてユーザー・アクセスが安全に行われるため、より強固なOracle Virtual Private Databaseポリシーを作成できます。様々なタイプのアプリケーション・コンテキストの詳細は、「アプリケーション・コンテキストを使用したユーザー情報の取得」を参照してください。

  さらに、CまたはJavaコールを埋め込み、オペレーティング・システム情報にアクセスしたり、オペレーティング・システム・ファイルや他のソースからWHERE句を戻すことができます。
- 関連するポリシー関数内の表から選択しないでください。表へのポリシーの定義は可能ですが、表に定義されたポリシーから表を選択することはできません。
- 純粋関数である必要があります。VPD関数は、受け取ったアプリケーション・コンテキストおよび引数のみに依存して、WHERE句を生成するものでなければなりません。この関数がパッケージ変数に依存することは許可されません。

ノート:

関数を様々なエディションで実行する場合、関数の結果がすべてのエディションで同一か、または関数が実行されているエディションによって異なるかに関係なく、関数の結果を制御できます。詳細は、エディションがグローバル・アプリケーション・コンテキストの PL/SQL パッケージの結果に与える影響を参照してください。

親トピック: Oracle Virtual Private Databaseポリシーのコンポーネント

14.2.2 保護するオブジェクトに関数を付加するポリシー

Oracle Virtual Private Databaseポリシーで、VPD関数を表やビュー、シノニムに関連付けます。

このポリシーを作成するには、DBMS_RLSパッケージを使用します。SYSでない方は、DBMS_RLSパッケージを使用するためにEXECUTE権限を付与される必要があります。このパッケージには、ポリシーの管理とファイングレイン・アクセス・コントロールの設定が可能になるプロシージャが含まれています。たとえば、ポリシーを表に付加するには、DBMS_RLS.ADD_POLICYプロシージャを使用します。この設定の中で、たとえばユーザーがSELECT文やUPDATE文を表またはビューで発行するとポリシーが有効になるよう設定して、ファイングレイン・アクセス・コントロールを設定します。

Oracle Virtual Private Databaseポリシーの作成とは、関数を作成し、その関数を表またはビューに適用することを意味します。

関連トピック

- Oracle Virtual Private Databaseのポリシーの構成

- 例: Oracle Virtual Private Databaseポリシーの作成

親トピック: Oracle Virtual Private Databaseポリシーのコンポーネント

14.3 Oracle Virtual Private Databaseのポリシーの構成

DBMS_RLS PL/SQLパッケージで、Oracle Virtual Private Database (VPD)ポリシーを構成できます。

- Oracle Virtual Private Databaseポリシーについて
  Oracle Virtual Private Databaseポリシーで、VPD関数をデータベース表やビュー、シノニムに関連付けます。
- データベース表、ビューまたはシノニムへのポリシーの付加
  DBMS_RLS PL/SQLパッケージで、表、ビューまたはシノニムにポリシーを付加できます。
- 例: 表への単純なOracle Virtual Private Databaseポリシーの付加
  DBMS_RLS.ADD_POLICYプロシージャで、Oracle Virtual Private Database (VPD)ポリシーを表、ビューまたはシノニムに付加できます。
- 特定のSQL文に対するポリシーの規定
  Oracle Virtual Private Databaseポリシーは、SELECT、INSERT、UPDATE、INDEXおよびDELETE文に規定できます。
- 例: DBMS_RLS.ADD_POLICYを使用したSQL文の指定
  DBMS_RLS.ADD_POLICYプロシージャのstatement_typesパラメータで、ポリシーのSELECT文とINDEX文を指定できます。
- ポリシーを使用した列データ表示の制御
  セキュリティに関連する列が問合せで参照される際に行レベルのセキュリティを規定するポリシーを作成できます。
- Oracle Virtual Private Databaseポリシー・グループ
  Oracle Virtual Private Databaseポリシー・グループは、アプリケーションに適用できるVPDポリシーの名前付きコレクションです。
- Oracle Virtual Private Databaseポリシー・タイプを使用したパフォーマンスの最適化
  Oracle Virtual Private Database (VPD)の動的、静的または共有ポリシー・タイプを使用して、パフォーマンスを最適化できます。

親トピック: Oracle Virtual Private Databaseを使用したデータ・アクセスの制御

14.3.1 Oracle Virtual Private Databaseポリシーについて

Oracle Virtual Private Databaseポリシーで、VPD関数をデータベース表やビュー、シノニムに関連付けます。

この関数はOracle Virtual Private Database WHERE句のアクションを定義します。この関数を、Oracle Virtual Private Database (VPD)アクションが適用されるデータベース表に関連付ける必要があります。

これを行うには、Oracle Virtual Private Databaseポリシーを構成します。ポリシーとは、仮想プライベート・データベース関数を管理するメカニズムです。また、ポリシーを使用すると、ファイングレイン・アクセス・コントロールを追加できるため、たとえば、SQL文のタイプを指定したり、特定の表列にポリシーを適用できます。ユーザーがこのデータベース・オブジェクト内のデータにアクセスすると、ポリシーが自動的に有効になります。

表14-1に、DBMS_RLSパッケージに含まれているプロシージャを示します。

表14-1 DBMS_RLSのプロシージャ

プロシージャ	説明

プロシージャ	説明
個別ポリシーの処理用	-
DBMS_RLS.ADD_POLICY	表、ビューまたはシノニムにポリシーを追加します。
DBMS_RLS.ENABLE_POLICY	表、ビューまたはシノニムに事前に追加したポリシーを使用可能または使用禁止にします。
DBMS_RLS.ALTER_POLICY	属性とポリシーの関連付けまたは関連付けの解除のため、既存のポリシーを変更します。
DBMS_RLS.REFRESH_POLICY	静的ポリシー以外のポリシーに対応付けられたカーソルを無効にします。
DBMS_RLS.DROP_POLICY	表、ビューまたはシノニムからポリシーを削除します。
グループ・ポリシーの処理用	-
DBMS_RLS.CREATE_POLICY_GROUP	ポリシー・グループを作成します。
DBMS_RLS.ALTER_GROUPED_POLICY	ポリシー・グループを変更します。
DBMS_RLS.DELETE_POLICY_GROUP	ポリシー・グループを削除します。
DBMS_RLS.ADD_GROUPED_POLICY	特定のポリシー・グループにポリシーを追加します。
DBMS_RLS.ENABLE_GROUPED_POLICY	グループ内のポリシーを使用可能にします。
DBMS_RLS.REFRESH_GROUPED_POLICY	リフレッシュされたポリシーに対応付けられた SQL 文を再解析します。
DBMS_RLS.DISABLE_GROUPED_POLICY	グループ内のポリシーを使用禁止にします。
DBMS_RLS.DROP_GROUPED_POLICY	特定のグループに属するポリシーを削除します。
アプリケーション・コンテキストの処理用	-
DBMS_RLS.ADD_POLICY_CONTEXT	アクティブなアプリケーションのコンテキストを追加します。

プロシージャ	説明
DBMS_RLS.DROP_POLICY_CONTEXT	アプリケーションのコンテキストを削除します。

関連トピック

- Oracle Virtual Private Databaseポリシーのコンポーネント
- アプリケーション・コンテキストを使用したユーザー情報の取得

親トピック: Oracle Virtual Private Databaseのポリシーの構成

14.3.2 データベース表、ビューまたはシノニムへのポリシーの付加

DBMS_RLS PL/SQLパッケージで、表、ビューまたはシノニムにポリシーを付加できます。

- ポリシーをデータベース表、ビューまたはシノニムに付加するには、DBMS_RLS.ADD_POLICYプロシージャを使用します。

ポリシーを追加する表、ビューまたはシノニム、およびポリシーの名前を指定する必要があります。また、ポリシーが制御する文のタイプ(SELECT、INSERT、UPDATE、DELETE、CREATE INDEXまたはALTER INDEX)など、その他の情報も指定できます。

次のガイドラインに従ってください。

- 拡張データリンク・オブジェクトとしてビューを作成した場合は、ビューの基礎となるオブジェクトの場合と同じVPDポリシーをこのタイプのビュー適用することをお薦めします。
- VPDポリシーを追加するベース・オブジェクトに依存オブジェクトが含まれているかどうかを判別します。依存オブジェクトが含まれている場合は、そのVPDポリシーがベース・オブジェクトに追加されるとこれらのオブジェクトは無効になり、使用時には自動的に再コンパイルされます。

  あるいは、ALTER ... COMPILE文を使用してこれらを各自でプロアクティブに再コンパイルできます。依存オブジェクトを(そのベース・オブジェクトにVPDポリシーを追加することにより)を無効にすると、それらを再コンパイルする必要が生じ、これが原因でシステム全体のパフォーマンスが低下する可能性がある点に注意してください。依存オブジェクトが含まれるオブジェクトにVPDポリシーを追加する操作は、オフピーク時またはスケジュールされた停止時間のみに行うことをお薦めします。
- 1つのオブジェクトに作成できるポリシーの最大数は255であることに注意してください。

親トピック: Oracle Virtual Private Databaseのポリシーの構成

14.3.3 例: 表への単純なOracle Virtual Private Databaseポリシーの付加

DBMS_RLS.ADD_POLICYプロシージャで、Oracle Virtual Private Database (VPD)ポリシーを表、ビューまたはシノニムに付加できます。

例14-1に、DBMS_RLS.ADD_POLICYを使用して、secure_updateというOracle Virtual Private DatabaseポリシーをHR.EMPLOYEES表に付加する方法を示します。ポリシーに付加される関数はcheck_updatesです。

例14-1 表への単純なOracle Virtual Private Databaseポリシーの付加

```
BEGIN
 DBMS_RLS.ADD_POLICY(
  object_schema   => 'hr',
  object_name     => 'employees',
```

```
  policy_name     => 'secure_update',
  policy_function => 'check_updates',
...
```

関数がパッケージ内に作成されている場合は、パッケージ名を指定します。たとえば:

```
 policy_function => 'pkg.check_updates',
...
```

表へのポリシーの定義は可能ですが、表に定義されたポリシーから表を選択することはできません。

親トピック: Oracle Virtual Private Databaseのポリシーの構成

14.3.4 特定のSQL文に対するポリシーの規定

Oracle Virtual Private Databaseポリシーは、SELECT、INSERT、UPDATE、INDEXおよびDELETE文に規定できます。

- ポリシーに対してSQL文のタイプを指定するには、DBMS_RLS.ADD_POLICYプロシージャのstatement_typesパラメータを使用します。複数指定するには、それぞれをカンマで区切ります。リストは一重引用符で囲みます。

文のタイプを指定しない場合、Oracle DatabaseではデフォルトでSELECT、INSERT、UPDATEおよびDELETEが指定されますが、INDEXは指定されません。これらの文のタイプの組合せを入力できます。

statement_typesパラメータを指定する場合は、次の機能に注意してください。

- Virtual Private Databaseポリシーによる影響を受けるアプリケーション・コードとして、MERGE INTO文が含まれる場合があります。しかし、Virtual Private Databaseポリシーでは、statement_typesパラメータにINSERT、UPDATE、およびDELETEの3つの文すべてを含めてポリシーを正常に機能させる必要があります。あるいは、statement_typesパラメータを省略できます。
- 索引をメンテナンスする権限を持っているユーザーは、たとえこのユーザーがSELECTのような標準問合せでは完全な表アクセスを持っていない場合でも、すべての行データを参照できることに注意してください。たとえばユーザーは、列値を引数とする、ユーザー定義関数を含む関数ベースの索引を作成できます。索引作成時に、Oracle Databaseがあらゆる行の列値をユーザー関数へ渡して、索引を作成するユーザーが行データを使用できるようにします。INDEXをstatement_typesパラメータで指定することにより、Oracle Virtual Private Databaseポリシーを索引メンテナンス操作で適用できます。

親トピック: Oracle Virtual Private Databaseのポリシーの構成

14.3.5 例: DBMS_RLS.ADD_POLICYを使用したSQL文の指定

DBMS_RLS.ADD_POLICYプロシージャのstatement_typesパラメータで、ポリシーのSELECT文とINDEX文を指定できます。

例14-2に、これを行う方法を示します。

例14-2 DBMS_RLS.ADD_POLICYを使用したSQL文の指定

```
BEGIN
 DBMS_RLS.ADD_POLICY(
  object_schema   => 'hr',
  object_name     => 'employees',
  policy_name     => 'secure_update',
  policy_function => 'check_updates',
  statement_types => 'SELECT,INDEX');
END;
/
```

親トピック: Oracle Virtual Private Databaseのポリシーの構成

14.3.6 ポリシーを使用した列データ表示の制御

セキュリティに関連する列が問合せで参照される際に行レベルのセキュリティを規定するポリシーを作成できます。

- 列レベルのOracle Virtual Private Databaseのポリシー
  列レベルのポリシーによって、セキュリティに関連する列が問合せで参照される際の行レベルのセキュリティを規定できます。
- 例: 列レベルのOracle Virtual Private Databaseポリシーの作成
  CREATE FUNCTION文およびDBMS_RLS.ADD_POLICYプロシージャで、列レベルのOracle Virtual Private Databaseポリシーを構成できます。
- 問合せに関連する列の行のみの表示
  デフォルトで、列レベルのOracle Virtual Private Databaseでは、機密情報が含まれた列が問合せで参照された場合、戻される行の数が制限されます。
- 機密性の高い列をNULL値で表示するための列のマスク
  問合せで機密性の高い列を参照する場合、デフォルトの列レベルのOracle Virtual Private Databaseでは、戻される行の数が制限されます。
- 例: Oracle Virtual Private Databaseポリシーへの列のマスクの追加
  DBMS_RLS.ADD_POLICYプロシージャで、列レベルのOracle Virtual Private Database列のマスク動作を構成できます。

親トピック: Oracle Virtual Private Databaseのポリシーの構成

14.3.6.1 列レベルのOracle Virtual Private Databaseのポリシー

列レベルのポリシーによって、セキュリティに関連する列が問合せで参照される際の行レベルのセキュリティを規定できます。

列レベルのOracle Virtual Private Databaseポリシーは、表やビューに適用できますが、シノニムには適用できません。ポリシーを列に適用するには、`DBMS_RLS.ADD_POLICY`プロシージャの`SEC_RELEVANT_COLS`パラメータを使用して、セキュリティ関連の列を指定します。このパラメータによって、問合せで列が明示的または暗黙的に参照されるたびに、セキュリティ・ポリシーが適用されます。

たとえば、人事部門以外のユーザーは、通常、各自の社会保障番号を参照することのみが許可されます。販売担当者が次の問合せを開始するとします。

```
SELECT fname, lname, ssn FROM emp;
```

セキュリティ・ポリシーを実装する関数は述語`ssn='my_ssn'`を戻します。Oracle Databaseは次のように問合せをリライトして実行します。

```
SELECT fname, lname, ssn FROM emp
 WHERE ssn = 'my_ssn';
```

親トピック: ポリシーを使用した列データ表示の制御

14.3.6.2 例: 列レベルのOracle Virtual Private Databaseポリシーの作成

CREATE FUNCTION文およびDBMS_RLS.ADD_POLICYプロシージャで、列レベルのOracle Virtual Private Databaseポリシーを構成できます。

例14-3は、営業部門(部門番号30)のユーザーは他部門の従業員の給与を参照できないというOracle Virtual Private

Databaseポリシーを示します。このポリシーに関連する列は、salおよびcommです。最初にOracle Virtual Private Databaseポリシー関数を作成し、次にDBMS_RLS PL/SQLパッケージを使用して追加します。

例14-3 列レベルのOracle Virtual Private Databaseポリシーの作成

```
CREATE OR REPLACE FUNCTION hide_sal_comm (
 v_schema IN VARCHAR2,
 v_objname IN VARCHAR2)
RETURN VARCHAR2 AS
con VARCHAR2 (200);
BEGIN
 con := 'deptno=30';
 RETURN (con);
END hide_sal_comm;
```

次に、DBMS_RLS.ADD_POLICYプロシージャを使用して、ポリシーを構成します。

```
BEGIN
 DBMS_RLS.ADD_POLICY (
  object_schema     => 'scott',
  object_name       => 'emp',
  policy_name       => 'hide_sal_policy',
  policy_function   => 'hide_sal_comm',
  sec_relevant_cols => 'sal,comm');
END;
```

親トピック: ポリシーを使用した列データ表示の制御

14.3.6.3 問合せに関連する列の行のみの表示

デフォルトで、列レベルのOracle Virtual Private Databaseでは、機密情報が含まれた列が問合せで参照された場合、戻される行の数が制限されます。

これらのセキュリティ関連の列を指定するには、例14-3に示すように、DBMS_RLS.ADD_POLICYプロシージャのSEC_RELEVANT_COLUMNSパラメータを使用します。

たとえば、営業部門ユーザーはemp表に対してSELECT権限を持っており、この表は例14-3で作成された列レベルのOracle Virtual Private Databaseポリシーによって保護されている場合を想定します。ユーザー(たとえば、ユーザーSCOTT)は次の問合せを実行します。

```
SELECT ENAME, d.dname, JOB, SAL, COMM
 FROM emp e, dept d
 WHERE d.deptno = e.deptno;
```

データベースは次の行を戻します。

```
ENAME      DNAME          JOB              SAL       COMM
---------- -------------- --------- ---------- ----------
ALLEN      SALES          SALESMAN        1600        300
WARD       SALES          SALESMAN        1250        500
MARTIN     SALES          SALESMAN        1250       1400
BLAKE      SALES          MANAGER         2850
TURNER     SALES          SALESMAN        1500          0
JAMES      SALES          CLERK            950

6 rows selected.
```

表示されるのは、行のすべての列に対してユーザーがアクセス権を持っている行のみです。

親トピック: ポリシーを使用した列データ表示の制御

14.3.6.4 機密性の高い列をNULL値で表示するための列のマスク

問合せで機密性の高い列を参照する場合、デフォルトの列レベルのOracle Virtual Private Databaseでは、戻される行の数が制限されます。

列のマスク動作を利用すると、機密性の高い列を参照している場合にもすべての行が表示されます。ただし、機密性の高い列はNULL値で表示されます。列のマスク動作を有効にするには、DBMS_RLS.ADD_POLICYプロシージャのSEC_RELEVANT_COLS_optパラメータを設定します。

たとえば、前述の例に記載されている販売担当者の問合せ結果を考えてみます。列のマスクを使用すると、販売担当者自身の詳細と社会保障番号が格納されている行のみが表示されるのではなく、emp表からすべての行が表示されますが、ssn列の値はNULLになります。この動作は、行のサブセットのみを戻す他のあらゆるタイプのOracle Virtual Private Databaseポリシーとは基本的に異なることに注意してください。

列レベルのOracle Virtual Private Databaseに関するデフォルトのアクションとは対照的に、列のマスク動作ではすべての行が表示されますが、機密情報が含まれた列の値はNULLとして戻されます。ポリシーに列のマスク動作を含めるには、DBMS_RLS.ADD_POLICYプロシージャのSEC_RELEVANT_COLS_OPTパラメータをDBMS_RLS.ALL_ROWSに設定します。

列のマスクに関する考慮事項は次のとおりです。

- 列のマスクが適用されるのは、SELECT文に対してのみです。
- 通常のOracle Virtual Private Database述語とは異なり、ポリシー関数によって生成される列マスク条件は、単純なブール式であることが必要です。
- 計算を実行するアプリケーションや、NULL値が戻されることが想定されていないアプリケーションの場合は、標準の列レベルのOracle Virtual Private Databaseを使用して、列のマスク動作オプションSEC_RELEVANT_COLS_OPTではなくSEC_RELEVANT_COLSを指定します。
- オブジェクト・データ型(XMLtypeを含む)の列をsec_relevant_cols設定に含めないでください。この列型は、sec_relevant_cols設定ではサポートされていません。
- UPDATE AS SELECTとともに使用される列のマスク動作によって更新されるのは、ユーザーが表示を許可されている列のみです。
- 問合せによっては、列のマスク動作により一部の行が表示されない場合があります。たとえば:

  ```
  SELECT * FROM emp
   WHERE sal = 10;
  ```

  列のマスク動作オプションが設定されているため、salary列にNULL値が戻される場合は、この問合せを実行しても行が戻されない場合があります。

親トピック: ポリシーを使用した列データ表示の制御

14.3.6.5 例: Oracle Virtual Private Databaseポリシーへの列のマスクの追加

DBMS_RLS.ADD_POLICYプロシージャで、列レベルのOracle Virtual Private Database列のマスク動作を構成できます。

例14-4は、列レベルのOracle Virtual Private Database列のマスク動作を示しています。これは、例: 列レベルのOracle Virtual Private Databaseポリシーの作成と同じVPDポリシーを使用していますが、sec_relevant_cols_optをDBMS_RLS.ALL_ROWSとして指定しています。

例14-4 Oracle Virtual Private Databaseポリシーへの列のマスクの追加

```
BEGIN
 DBMS_RLS.ADD_POLICY(
   object_schema         => 'scott',
   object_name           => 'emp',
   policy_name           => 'hide_sal_policy',
   policy_function       => 'hide_sal_comm',
   sec_relevant_cols     =>' sal,comm',
   sec_relevant_cols_opt => dbms_rls.ALL_ROWS);
END;
```

emp表に対するSELECT権限を持つ営業部門のユーザー(たとえば、SCOTT)が、次の問合せを実行するとします。

```
SELECT ENAME, d.dname, job, sal, comm
 FROM emp e, dept d
 WHERE d.deptno = e.deptno;
```

データベースは、この問合せで指定されたすべての行を戻しますが、Oracle Virtual Private Databaseポリシーによって一部の値はマスクされています。

```
ENAME      DNAME          JOB              SAL       COMM
---------- -------------- --------- ---------- ----------
CLARK      ACCOUNTING     MANAGER
KING       ACCOUNTING     PRESIDENT
MILLER     ACCOUNTING     CLERK
JONES      RESEARCH       MANAGER
FORD       RESEARCH       ANALYST
ADAMS      RESEARCH       CLERK
SMITH      RESEARCH       CLERK
SCOTT      RESEARCH       ANALYST
WARD       SALES          SALESMAN        1250        500
TURNER     SALES          SALESMAN        1500          0
ALLEN      SALES          SALESMAN        1600        300
JAMES      SALES          CLERK            950
BLAKE      SALES          MANAGER         2850
MARTIN     SALES          SALESMAN        1250       1400

14 rows selected.
```

列のマスク動作により、営業部門のユーザーの問合せで要求された列がすべて戻されますが、営業部門以外の従業員に関してはsal列とcomm列がNULLになっています。

親トピック: ポリシーを使用した列データ表示の制御

14.3.7 Oracle Virtual Private Databaseポリシー・グループ

Oracle Virtual Private Databaseポリシー・グループは、アプリケーションに適用できるVPDポリシーの名前付きコレクションです。

- Oracle Virtual Private Databaseポリシー・グループについて
  複数のセキュリティ・ポリシーをまとめてグループ化して、1つのアプリケーションに適用できます。
- Oracle Virtual Private Databaseの新しいポリシー・グループの作成
  DBMS_RLS.ADD_GROUPED_POLICYプロシージャで、VPDポリシーをVPDポリシー・グループに追加します。
- SYS_DEFAULTポリシー・グループを使用したデフォルト・ポリシー・グループ
  セキュリティ・ポリシーのグループ内で、1つのセキュリティ・ポリシーをデフォルトのセキュリティ・ポリシーになるよう指定できます。
- 各表、ビューまたはシノニムに対する複数のポリシー

同一の表、ビューまたはシノニムに対して複数のポリシーを設定できます。

- データベースへの接続に使用されるアプリケーションの検証
  駆動コンテキストを実装するパッケージは、データベースへの接続に使用されるアプリケーションを正しく検証する必要があります。

親トピック: Oracle Virtual Private Databaseのポリシーの構成

14.3.7.1 Oracle Virtual Private Databaseポリシー・グループについて

複数のセキュリティ・ポリシーをまとめてグループ化して、1つのアプリケーションに適用できます。

ポリシー・グループとは、アプリケーションに属する一連のセキュリティ・ポリシーです。有効なポリシー・グループを示すようにアプリケーション・コンテキスト(駆動コンテキストまたはポリシー・コンテキストと呼ばれます)を指定できます。ユーザーが表、ビュー、またはシノニム列にアクセスすると、Oracle Databaseが駆動コンテキストを検索して、有効なポリシー・グループを特定します。ポリシー・グループに属しているすべての関連ポリシーが適用されます。

ポリシー・グループは、複数のセキュリティ・ポリシーを持つ複数のアプリケーションが同一の表、ビューまたはシノニムを共有する場合に便利です。ポリシー・グループによって、表、ビューまたはシノニムがアクセスされたときに有効にするポリシーを識別できます。

たとえば、ホスト環境で、A社が、B社およびC社に対してBENEFIT表をホスティングするとします。この表は、2つの異なるセキュリティ・ポリシーを持つ、Human Resources(人事管理)およびFinance(財務管理)という2つアプリケーションによってアクセスされます。Human Resourcesアプリケーションは社内の序列に基づいてユーザーを認可し、Financeアプリケーションは部門に基づいてユーザーを認可します。これらの2つのポリシーをBENEFIT表に統合するには、2つの企業が共同でポリシーを開発する必要がありますが、それは現実的ではありません。ベース・オブジェクトに一連の特定のポリシーを規定するアプリケーション・コンテキストを定義することにより、各アプリケーションがセキュリティ・ポリシーの集合を個別に実装できます。

これを行うには、セキュリティ・ポリシーをグループ化します。アプリケーション・コンテキストを参照することにより、実行時に有効にするポリシーのグループをOracle Databaseが判断します。サーバーは、そのポリシー・グループに属するすべてのポリシーを規定します。

親トピック: Oracle Virtual Private Databaseのポリシー・グループ

14.3.7.2 Oracle Virtual Private Databaseの新しいポリシー・グループの作成

DBMS_RLS.ADD_GROUPED_POLICYプロシージャで、VPDポリシーをVPDポリシー・グループに追加します。

有効にするポリシーを指定するには、DBMS_RLS.ADD_POLICY_CONTEXTプロシージャを使用して駆動コンテキストを追加できます。駆動コンテキストが不明なポリシー・グループを戻した場合は、エラーが戻されます。

駆動コンテキストが定義されていない場合は、すべてのポリシーが実行されます。同様に、駆動コンテキストがNULLである場合は、すべてのポリシー・グループのポリシーが規定されます。データにアクセスするアプリケーションは、セキュリティ設定モジュール(アプリケーション・コンテキストを設定するモジュール)を回避できないため、該当するすべてのポリシーが適用されます。

同一の表、ビューまたはシノニムに複数の駆動コンテキストを適用して、それぞれを個別に処理できます。これによって、複数のアクティブなポリシーを設定して規定できます。

たとえば、福利厚生アプリケーションと財務アプリケーションをホスティングするホスト企業があり、これらのアプリケーションが、いくつかのデータベース・オブジェクトを共有している場合を想定します。この2つのアプリケーションは、SYS_DEFAULTポリシー・グループのSUBSCRIBERポリシーを使用して、ホスティング用にストライプ化されます。データ・アクセスは、最初にサブスクライバIDごとにパーティション化され、次に、ユーザーが福利厚生アプリケーションと財務アプリケーションのどちらにアクセスしているか(駆動コンテキストによって決定される)によってパーティション化されます。ホスティング・サービスを使用するA社が、自社のデータ・アクセスにのみ関連したカスタム・ポリシーを適用するとします。この場合は、駆動コンテキスト(COMPANY A SPECIALなど)を追加し

て、追加の特殊ポリシー・グループの適用をA社のデータ・アクセスのみに限定できます。このポリシーはA社のみに関連しているため、SUBSCRIBERポリシーの下では適用できません。基本的なホスティング・ポリシーは、他のポリシーから分離した方がより効率的です。

親トピック: Oracle Virtual Private Databaseのポリシー・グループ

14.3.7.3 SYS_DEFAULTポリシー・グループを使用したデフォルト・ポリシー・グループ

セキュリティ・ポリシーのグループ内で、1つのセキュリティ・ポリシーをデフォルトのセキュリティ・ポリシーになるよう指定できます。

これは、セキュリティ・ポリシーをアプリケーション別に分割して常に有効になるようにする場合に便利です。デフォルト・セキュリティ・ポリシーを使用すると、開発者はすべての条件下で基礎となるセキュリティを規定できます。一方、アプリケーションごとにセキュリティ・ポリシーを分割(セキュリティ・グループを使用)すると、デフォルト・セキュリティ・ポリシーにアプリケーション固有のセキュリティ・レイヤーを追加できます。デフォルト・セキュリティ・ポリシーを実装するには、このデフォルト・セキュリティ・ポリシーをSYS_DEFAULTポリシー・グループに追加します。

このグループ内に定義されている、特定の表、ビューまたはシノニムに対するポリシーは、駆動コンテキストが指定するポリシー・グループとともに実行されます。前述のとおり、駆動コンテキストは、有効なポリシー・グループを指定するアプリケーション・コンテキストです。SYS_DEFAULTポリシー・グループには、ポリシーが含まれる場合と含まれない場合があります。SYS_DEFAULTポリシー・グループは削除できません。このポリシー・グループを削除すると、Oracle Databaseでエラーが発生します。

SYS_DEFAULTポリシー・グループに、複数のオブジェクトに対応付けられているポリシーを追加する場合、各オブジェクトには個別にSYS_DEFAULTポリシー・グループが対応付けられます。たとえば、scottスキーマのemp表に1つのSYS_DEFAULTポリシー・グループがある場合、scottスキーマのdept表には別のSYS_DEFAULTポリシー・グループが対応付けられます。これは、次のようにツリー構造に編成されます。

```
SYS_DEFAULT
  - policy1 (scott/emp)
  - policy3 (scott/emp)
SYS_DEFAULT
  - policy2 (scott/dept)
```

同一の名前を持つ複数のポリシー・グループを作成できます。特定のポリシー・グループを選択すると、対応付けられているスキーマとオブジェクト名が、画面右側のプロパティ・シートに表示されます。

親トピック: Oracle Virtual Private Databaseのポリシー・グループ

14.3.7.4 各表、ビューまたはシノニムに対する複数のポリシー

同一の表、ビューまたはシノニムに対して複数のポリシーを設定できます。

たとえば、受注用の基本アプリケーションがあり、社内の各部門にはそれぞれ独自のデータ・アクセス規則があるとします。基本アプリケーションのポリシー関数を作成しなおさなくても、部門固有のポリシー関数を表に追加できます。

表に適用されるすべてのポリシーは、AND構文で規定されます。そのため、CUSTOMERS表に3つのポリシーを適用している場合、各ポリシーが表に適用されます。データにアクセスするアプリケーションに応じて異なるポリシーが適用されるように、ポリシー・グループおよびアプリケーション・コンテキストを使用して、ファイングレイン・アクセス・コントロールの規定を分割できます。これによって、開発グループ間でポリシーを調整する必要がなくなり、アプリケーションの開発が容易になります。また、常に適用する(たとえば、ホスト環境でサブスクライバによりデータ分割を規定する)デフォルト・ポリシー・グループを保持することもできます。

親トピック: Oracle Virtual Private Databaseのポリシー・グループ

14.3.7.5 データベースへの接続に使用されるアプリケーションの検証

駆動コンテキストを実装するパッケージは、データベースへの接続に使用されるアプリケーションを正しく検証する必要があります。

Oracle Databaseは、コール・スタックをチェックして、駆動コンテキストを実装するパッケージによるコンテキスト属性の設定を確認しますが、パッケージ内では不適切な検証が発生する可能性があります。たとえば、データベース・ユーザーまたはエンタープライズ・ユーザーがデータベースに認識されているアプリケーションの場合、ユーザーには、駆動コンテキストを設定するパッケージに対するEXECUTE権限が必要です。BENEFITSアプリケーションの方がHRアプリケーションよりもアクセスが自由であることを理解しているユーザーについて考えてみます。(正しいポリシー・グループを駆動コンテキスト内に設定する)setctxプロシージャでは、実際に接続しているアプリケーションを判断するための検証が実行されないこと。つまり、このプロシージャでは、(3層システムに対する)着信接続のIPアドレス、またはユーザー・セッションのproxy_user属性がチェックされないこと。

ユーザーは、アクセスがより自由なBENEFITSポリシー・グループにコンテキストを設定する引数を駆動コンテキスト・パッケージに渡してから、かわりにHRアプリケーションにアクセスできます。setctxではアプリケーションに対してそれ以上の検証を行わないため、このユーザーは限定的なHRセキュリティ・ポリシーをバイパスしてしまいます。

一方、Oracle Virtual Private Databaseによるプロキシ認証を実装すると、ユーザーのかわりにデータベースに接続する中間層(およびアプリケーション)の識別情報を確認できます。これによって、データ・アクセスを仲介するアプリケーションごとに正しいポリシーが適用されます。

たとえば、開発者は、プロキシ認証機能を使用して、データベースに接続しているアプリケーション(中間層)がHRAPPSERVERであることを確認できます。このように、駆動コンテキストを実装するパッケージでは、ユーザー・セッションのproxy_userがHRAPPSERVERかどうかを検証できます。その場合、HRポリシー・グループを使用するよう駆動コンテキストを設定できます。proxy_userがHRAPPSERVERでない場合は、アクセスを拒否できます。

このような場合に、次の問合せが実行されるとします。

```
SELECT * FROM apps.benefit;
```

Oracle Databaseは、デフォルト・ポリシー・グループ(SYS_DEFAULT)およびアクティブなネームスペースHRのポリシーを選択します。この問合せは、内部で次のようにリライトされます。

```
SELECT * FROM apps.benefit
 WHERE company = SYS_CONTEXT('ID','MY_COMPANY')
 AND SYS_CONTEXT('ID','TITLE') = 'MANAGER';
```

親トピック: Oracle Virtual Private Databaseのポリシー・グループ

14.3.8 Oracle Virtual Private Databaseポリシー・タイプを使用したパフォーマンスの最適化

Oracle Virtual Private Database (VPD)の動的、静的または共有ポリシー・タイプを使用して、パフォーマンスを最適化できます。

- Oracle Virtual Private Databaseポリシー・タイプについて
  ポリシーのポリシー・タイプを指定すると、Oracle Virtual Private Databaseポリシーの実行パフォーマンスを最適化できます。
- ポリシー関数の自動再実行のための動的ポリシー・タイプ
  DYNAMICポリシー・タイプを指定すると、ユーザーが仮想プライベート・データベースで保護されたデータベース・オブジェクトにアクセスするたびに、ポリシー関数が実行されます。
- 例: DBMS_RLS.ADD_POLICYを使用したDYNAMICポリシーの作成

DBMS_RLS.ADD_POLICYプロシージャで、動的なOracle Virtual Private Databaseポリシーを作成できます。

- ポリシー関数の問合せごとの再実行を回避するための静的ポリシー
  静的ポリシー・タイプを指定すると、インスタンス内のすべてのユーザーに対して同じ述語が規定されます。
- 例: DBMS_RLS.ADD_POLICYを使用した静的ポリシーの作成
  DBMS_RLS.ADD_POLICYプロシージャで、静的なOracle Virtual Private Database (VPD)ポリシーを作成できます。
- 例: 複数オブジェクト間でポリシーを共有するための共有の静的ポリシー
  複数オブジェクト間でポリシーを共有するために、DBMS_RLS.ADD_POLICYプロシージャで共有の静的Oracle Virtual Private Databaseポリシーを作成できます。
- 静的ポリシーおよび共有の静的ポリシーを使用する場合
  静的ポリシーは、すべての問合せに同じ述語が必要で、高いパフォーマンスが不可欠なホスト環境などに最適です。
- 変更されるアプリケーション・コンテキスト属性の状況依存ポリシー
  状況依存ポリシーは、どの述語が問合せを実行しているかに応じて異なる述語を適用する必要がある場合に便利です。
- 例: DBMS_RLS.ADD_POLICYを使用した状況依存ポリシーの作成
  DBMS_RLS.ADD_POLICYプロシージャで、Oracle Virtual Private Database状況依存ポリシーを作成できます。
- 例: VPD状況依存ポリシーのキャッシュされた文のリフレッシュ
  DBMS_RLS.REFRESH_POLICY文で、Oracle Virtual Private Database状況依存ポリシーのキャッシュされた文をリフレッシュできます。
- 例: 既存の状況依存ポリシーの変更
  DBMS_RLS.ALTER_POLICYプロシージャで、Oracle Virtual Private Databaseポリシーを変更できます。
- 例: 共有の状況依存ポリシーの使用による複数オブジェクト間でのポリシーの共有
  複数オブジェクトがあるポリシーを共有するために、DBMS_RLS.ADD_POLICYプロシージャを使用して、共有の状況依存のOracle Virtual Private Databaseポリシーを作成できます。
- 状況依存ポリシーおよび共有の状況依存ポリシーを使用する場合
  状況依存ポリシーは、ユーザー・セッションごとに述語を変える必要はないが、ポリシーが異なるユーザーまたはグループに複数の異なる述語を規定する必要がある場合に使用します。
- 5種類のOracle Virtual Private Databaseポリシー・タイプの要約
  Oracle Virtual Private Databaseには、ユーザーのニーズ(ホスト環境での使用など)に基づいて、5種類のポリシー・タイプが用意されています。

親トピック: Oracle Virtual Private Databaseのポリシーの構成

14.3.8.1 Oracle Virtual Private Databaseポリシー・タイプについて

ポリシーのポリシー・タイプを指定すると、Oracle Virtual Private Databaseポリシーの実行パフォーマンスを最適化できます。

ポリシー・タイプを使用して、Oracle DatabaseがOracle Virtual Private Databaseポリシーの述語をキャッシュする方法を制御します。ポリシー関数を実行すると、システム・リソースを大量に消費する場合があるため、ポリシーに対してポリシー・タイプを設定することを検討してください。ポリシー関数の実行回数を最小限に抑えることで、データベースのパフォーマンスを最適化できます。

選択できるポリシー・タイプは、DYNAMIC、STATIC、SHARED_STATIC、CONTEXT_SENSITIVEおよびSHARED_CONTEXT_SENSITIVEの5種類です。これらのポリシー・タイプによって、ポリシーの述語を変更する頻度を正確に指定できます。ポリシー・タイプを指定するには、DBMS_RLS.ADD POLICYプロシージャのpolicy_typeパラメータを設定

します。

親トピック: Oracle Virtual Private Databaseポリシー・タイプを使用したパフォーマンスの最適化

14.3.8.2 ポリシー関数の自動再実行のための動的ポリシー・タイプ

DYNAMICポリシー・タイプを指定すると、ユーザーが仮想プライベート・データベースで保護されたデータベース・オブジェクトにアクセスするたびに、ポリシー関数が実行されます。

DBMS_RLS.ADD_POLICYプロシージャにポリシー・タイプを指定しない場合、ポリシーはデフォルトで動的になります。ポリシーを動的に構成するには、DBMS_RLS.ADD_POLICYプロシージャのpolicy_typeパラメータをDYNAMICに設定します。

動的ポリシー・タイプでは、静的ポリシーや状況依存ポリシー・タイプの場合と異なり、データベースのパフォーマンスは最適化されません。ただし、ポリシーを静的または状況依存に設定する前に、その都度実行されるDYNAMICポリシー・タイプでテストすることをお薦めします。最初にポリシー関数をDYNAMICポリシーでテストすると、何もキャッシュされていないため、ポリシー関数が各問合せに与える影響を調べることができます。これにより、パフォーマンスを最適化するために静的ポリシーまたは状況依存ポリシーを使用可能にする前に、関数が正常に機能することを確認できます。

文の実行開始時間と終了時間を測定するには、DBMS_UTILITY.GET_TIME関数を使用します。たとえば:

```
-- 1. Get the start time:
SELECT DBMS_UTILITY.GET_TIME FROM DUAL;
  GET_TIME
----------
   2312721
-- 2. Run the statement:
SELECT COUNT(*) FROM HR.EMPLOYEES;
  COUNT(*)
----------
       107
-- 3. Get the end time:
SELECT DBMS_UTILITY.GET_TIME FROM DUAL;
  GET_TIME
----------
   2314319
```

関連トピック

- ファンクション、プロシージャ、パッケージおよびトリガーの監査

親トピック: Oracle Virtual Private Databaseポリシー・タイプを使用したパフォーマンスの最適化

14.3.8.3 例: DBMS_RLS.ADD_POLICYを使用したDYNAMICポリシーの作成

DBMS_RLS.ADD_POLICYプロシージャで、動的なOracle Virtual Private Databaseポリシーを作成できます。

例14-5に、DYNAMICポリシー・タイプの作成方法を示します。

例14-5 DBMS_RLS.ADD_POLICYを使用したDYNAMICポリシーの作成

```
BEGIN
 DBMS_RLS.ADD_POLICY(
  object_schema   => 'hr',
  object_name     => 'employees',
  policy_name     => 'secure_update',
  policy_function => 'hide_fin',
  policy_type     => dbms_rls.DYNAMIC);
END;
/
```

親トピック: Oracle Virtual Private Databaseポリシー・タイプを使用したパフォーマンスの最適化

14.3.8.4 ポリシー関数の問合せごとの再実行を回避するための静的ポリシー

静的ポリシー・タイプを指定すると、インスタンス内のすべてのユーザーに対して同じ述語が規定されます。

Oracle Databaseでは静的ポリシーの述語がSGAに格納されるため、ポリシー関数は問合せごとに再実行されません。その結果、パフォーマンスが向上します。

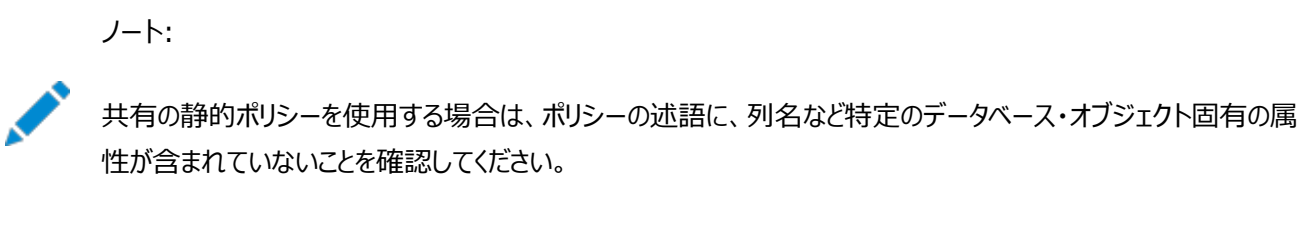
静的ポリシーを使用可能にするには、そのポリシーを複数のオブジェクト間で共有するかどうかに応じて、DBMS_RLS.ADD_POLICYプロシージャのpolicy_typeパラメータをSTATICまたはSHARED_STATICのいずれかに設定します。

述語が同じでも、同じカーソルを実行するたびに異なる行セットが生成される場合があります。これは、述語によるデータのフィルタ処理がSYS_CONTEXTやSYSDATEなどの属性によって異なるためです。

たとえば、ポリシーをSTATICまたはSHARED_STATICポリシー・タイプとして使用可能にする場合を想定します。この場合は、ポリシーで保護されたデータベース・オブジェクトに対して実行されるすべての問合せに次の述語が追加されます。

```
WHERE dept = SYS_CONTEXT ('hr_app','deptno')
```

述語は、問合せごとに変わりませんが、SYS_CONTEXTのセッション属性に基づいて問合せに適用されます。前述の例の述語では、ポリシーで保護されたデータベース・オブジェクトを問い合せているユーザーの部門番号がSYS_CONTEXTのdeptno属性と一致する行のみを戻します。



ノート:

共有の静的ポリシーを使用する場合は、ポリシーの述語に、列名など特定のデータベース・オブジェクト固有の属性が含まれていないことを確認してください。

関連トピック

- ファンクション、プロシージャ、パッケージおよびトリガーの監査

親トピック: Oracle Virtual Private Databaseポリシー・タイプを使用したパフォーマンスの最適化

14.3.8.5 例: DBMS_RLS.ADD_POLICYを使用した静的ポリシーの作成

DBMS_RLS.ADD_POLICYプロシージャで、静的なOracle Virtual Private Database (VPD)ポリシーを作成できます。

例14-6に、STATICポリシー・タイプの作成方法を示します。

例14-6 DBMS_RLS.ADD_POLICYを使用した静的ポリシーの作成

```
BEGIN
 DBMS_RLS.ADD_POLICY(
  object_schema   => 'hr',
  object_name     => 'employees',
  policy_name     => 'secure_update',
  policy_function => 'hide_fin',
  policy_type     => DBMS_RLS.STATIC);
END;
/
```

親トピック: Oracle Virtual Private Databaseポリシー・タイプを使用したパフォーマンスの最適化

14.3.8.6 例: 複数オブジェクト間でポリシーを共有するための共有の静的ポリシー

複数オブジェクト間でポリシーを共有するために、DBMS_RLS.ADD_POLICYプロシージャで共有の静的Oracle Virtual Private Databaseポリシーを作成できます。

たとえば、使用する財務データを含むHRスキーマの2番目の表に例14-6のポリシーを適用する場合、両方の表でSHARED_STATIC設定を使用します。

例14-7に、同じポリシーを共有する2つの表に対してSHARED_STATICポリシー・タイプを設定する方法を示します。

例14-7 複数オブジェクト間でポリシーを共有するための共有の静的ポリシーの作成

```
-- 1. Create a policy for the first table, employees:
BEGIN
 DBMS_RLS.ADD_POLICY(
  object_schema   => 'hr',
  object_name     => 'employees',
  policy_name     => 'secure_update',
  policy_function => 'hide_fin',
  policy_type     => dbms_rls.SHARED_STATIC);
END;
/
-- 2. Create a policy for the second table, fin_data:
BEGIN
 DBMS_RLS.ADD_POLICY(
  object_schema   => 'hr',
  object_name     => 'fin_data',
  policy_name     => 'secure_update',
  policy_function => 'hide_fin',
  policy_type     => dbms_rls.SHARED_STATIC);
END;
/
```

親トピック: Oracle Virtual Private Databaseポリシー・タイプを使用したパフォーマンスの最適化

14.3.8.7 静的ポリシーおよび共有の静的ポリシーを使用する場合

静的ポリシーは、すべての問合せに同じ述語が必要で、高いパフォーマンスが不可欠なホスト環境などに最適です。

このような環境では、ポリシー関数がすべての問合せに同じ述語を追加すると、ポリシー関数を再実行するたびに不要なオーバーヘッドがシステムに加わります。たとえば、競争相手である複数の顧客企業に関する市場調査データが含まれたデータ・ウェアハウスを考えてみます。このウェアハウスでは、各企業が自社の市場調査データのみを参照できるポリシーを規定する必要があり、このポリシーは次の述語で表現されます。

```
WHERE subscriber_id = SYS_CONTEXT('customer', 'cust_num')
```

アプリケーション・コンテキストに対してSYS_CONTEXTを使用すると、データベースでは、戻される行を動的に変更できます。関数を再実行する必要はなく、述語はSGAにキャッシュされるため、システム・リソースを節約でき、パフォーマンスが向上します。

親トピック: Oracle Virtual Private Databaseポリシー・タイプを使用したパフォーマンスの最適化

14.3.8.8 変更されるアプリケーション・コンテキスト属性の状況依存ポリシー

状況依存ポリシーは、どの述語が問合せを実行しているかに応じて異なる述語を適用する必要がある場合に便利です。

たとえば、マネージャには述語WHERE groupがmanagersに設定され、従業員には述語WHERE empno_ctxがemp_idに設定される場合を考えてみます。状況依存ポリシーでは、マネージャのログイン時にマネージャが確認する必要がある情報のみ表示し、従業員のログイン時に従業員が確認する必要がある情報のみ表示できます。このポリシーは、アプリケーション・コンテキストを使用して、使用する述語を決定します。

静的ポリシーとは対照的に、状況依存ポリシーは必ずしも述語をキャッシュしません。状況依存ポリシーの場合、データベースでは、述語は文の解析後に変更されると想定しています。ただし、ローカル・アプリケーション・コンテキストに変更がない場合、Oracle Databaseはユーザー・セッション内でポリシー関数を再実行しません。ユーザー・セッション中にアプリケーション・コンテキストの属性の変更がある場合、デフォルトでは、データベースはポリシー関数を再実行して、初期解析からの述語へのすべての変更を取得していることを確認します。これにより、関連付けられている属性が変更されていない場合、ポリシー関数の再実行が必要なくなります。namespaceおよびattributeパラメータを含めて、特定のアプリケーション・コンテキストへの評価を制限できます。

ポリシーでnamespaceおよびattributeパラメータを使用する場合、次のガイドラインに従います。

- 1つだけではなくnamespaceおよびattributeパラメータの両方を指定していることを確認します。
- ポリシーにDBMS_RLS.CONTEXT_SENSITIVEまたはSHARED_CONTEXT_SENSITIVEに設定されているpolicy_type引数があることを確認します。静的または動的ポリシーのnamespaceおよびattributeパラメータを使用できません。

仮想プライベート・データベースのポリシー関数に関連付けられている属性がない場合、Oracle Databaseは、アプリケーション・コンテキストの変更の状況依存関数を評価します。

共有の状況依存ポリシーは、複数のデータベース・オブジェクト間で共有できる点を除いて、通常の状況依存ポリシーと同じように動作します。このポリシー・タイプの場合、すべてのオブジェクトがUGAのポリシー関数を共有でき、この場合、述語はローカル・セッション・コンテキストが変更されるまでキャッシュされます。

関連トピック

- 例: 共有の状況依存ポリシーの使用による複数オブジェクト間でのポリシーの共有
- チュートリアル: セッション・ベースのアプリケーション・コンテキスト・ポリシーの実装
- 例: Oracle Virtual Private Databaseポリシー・グループの実装

親トピック: Oracle Virtual Private Databaseポリシー・タイプを使用したパフォーマンスの最適化

14.3.8.9 例: DBMS_RLS.ADD_POLICYを使用した状況依存ポリシーの作成

DBMS_RLS.ADD_POLICYプロシージャで、Oracle Virtual Private Database状況依存ポリシーを作成できます。

例14-8は、empno_ctxネームスペースおよびemp_id属性への変更についてのみポリシーが評価されるCONTEXT_SENSITIVEポリシーの作成方法を示しています。

例14-8 DBMS_RLS.ADD_POLICYを使用した状況依存ポリシーの作成

```
BEGIN
 DBMS_RLS.ADD_POLICY(
  object_schema   => 'hr',
  object_name     => 'employees',
  policy_name     => 'secure_update',
  policy_function => 'hide_fin',
  policy_type     => dbms_rls.CONTEXT_SENSITIVE,
  namespace       => 'empno_ctx',
  attribute       => 'emp_id');
END;
/
```

親トピック: Oracle Virtual Private Databaseポリシー・タイプを使用したパフォーマンスの最適化

14.3.8.10 例: VPD状況依存ポリシーのキャッシュされた文のリフレッシュ

DBMS_RLS.REFRESH_POLICY文で、Oracle Virtual Private Database状況依存ポリシーのキャッシュされた文をリ

フレッシュできます。

例14-9は、DBMS_RLS.REFRESH_POLICYプロシージャを実行して仮想プライベート・データベースの状況依存ポリシーに関連付けられているすべてのキャッシュされた文を手動でリフレッシュできることを示しています。

例14-9 VPD状況依存ポリシーのキャッシュされた文のリフレッシュ

```
BEGIN
 DBMS_RLS.REFRESH_POLICY(
  object_schema   => 'hr',
  object_name     => 'employees',
  policy_name     => 'secure_update');
END;
/
```

親トピック: Oracle Virtual Private Databaseポリシー・タイプを使用したパフォーマンスの最適化

14.3.8.11 例: 既存の状況依存ポリシーの変更

DBMS_RLS.ALTER_POLICYプロシージャで、Oracle Virtual Private Databaseポリシーを変更できます。

例14-10は、関連するコンテキスト属性が変更される場合にのみorder_update_polポリシー関数を実行するためにDBMS_RLS.ALTER_POLICY文を使用して既存の状況依存ポリシーを変更する方法を示しています。

例14-10 既存の状況依存ポリシーの変更

```
BEGIN
 DBMS_RLS.ALTER_POLICY(
  object_schema   => 'oe',
  object_name     => 'orders',
  policy_name     => 'order_update_pol',
  alter_option    =>  DBMS_RLS.ADD_ATTRIBUTE_ASSOCIATION,
  namespace       => 'empno_ctx',
  attribute       => 'emp_role');
END;
/
```

親トピック: Oracle Virtual Private Databaseポリシー・タイプを使用したパフォーマンスの最適化

14.3.8.12 例: 共有の状況依存ポリシーの使用による複数オブジェクト間でのポリシーの共有

複数オブジェクトがあるポリシーを共有するために、DBMS_RLS.ADD_POLICYプロシージャを使用して、共有の状況依存のOracle Virtual Private Databaseポリシーを作成できます。

例14-11は、複数の表でポリシーを共有する2つの共有の状況依存ポリシーを作成する方法およびempno_ctxネームスペースおよびemp_id属性への変更のみに評価を制限する方法を示しています。

例14-11 DBMS_RLS.ADD_POLICYを使用した共有の状況依存ポリシー

```
-- 1. Create a policy for the first table, employees:
BEGIN
 DBMS_RLS.ADD_POLICY(
  object_schema   => 'hr',
  object_name     => 'employees',
  policy_name     => 'secure_update',
  policy_function => 'hide_fin',
  policy_type     => dbms_rls.SHARED_CONTEXT_SENSITIVE,
  namespace       => 'empno_ctx',
  attribute       => 'emp_id');
END;
/
--2. Create a policy for the second table, fin_data:
```

```
BEGIN
 DBMS_RLS.ADD_POLICY(
  object_schema   => 'hr',
  object_name     => 'fin_data',
  policy_name     => 'secure_update',
  policy_function => 'hide_fin',
  policy_type     => dbms_rls.SHARED_CONTEXT_SENSITIVE,
  namespace       => 'empno_ctx',
  attribute       => 'emp_id');
END;
/
```

次のことに注意してください。

- 共有の状況依存ポリシーを使用する場合は、ポリシーの述語に、列名など特定のデータベース・オブジェクト固有の属性が含まれていないことを確認してください。
- 仮想プライベート・データベースの共有の状況依存ポリシーに関連付けられているすべてのキャッシュされた文を手動でリフレッシュするには、DBMS_RLS.REFRESH_GROUPED_POLICYプロシージャを実行します。

親トピック: Oracle Virtual Private Databaseポリシー・タイプを使用したパフォーマンスの最適化

14.3.8.13 状況依存ポリシーおよび共有の状況依存ポリシーを使用する場合

状況依存ポリシーは、ユーザー・セッションごとに述語を変える必要はないが、ポリシーが異なるユーザーまたはグループに複数の異なる述語を規定する必要がある場合に使用します。

たとえば、単一のポリシーを持つsales_history表を考えてみます。このポリシーでは、アナリストは自分の製品のみを参照でき、地域担当者は自分の地域のみを参照できます。この場合、データベースは、ユーザーのタイプが変わるたびにポリシー関数を再実行する必要があります。サーバーによるポリシー関数の再実行なしに、ユーザーがログインして保護されたオブジェクトに対していくつかのDML文を発行できる場合は、パフォーマンスが向上します。

ノート:

複数のクライアントが 1 つのデータベース・セッションを共有するセッション・プーリングの場合は、クライアント切替え時に中間層でコンテキストを再設定する必要があります。

親トピック: Oracle Virtual Private Databaseポリシー・タイプを使用したパフォーマンスの最適化

14.3.8.14 5種類のOracle Virtual Private Databaseポリシー・タイプの要約

Oracle Virtual Private Databaseには、ユーザーのニーズ(ホスト環境での使用など)に基づいて、5種類のポリシー・タイプが用意されています。

表14-2に、使用可能なポリシー・タイプの要約を示します。

表14-2 DBMS_RLS.ADD_POLICYのポリシー・タイプ

ポリシー・タイプ	ポリシー関数の実行	使用例	複数オブジェクト間での共有
DYNAMIC	ポリシーで保護されたデータベース・オブジェクトがアクセスされるたびに、ポリシー関数を	日中の特定時間は、ユーザーによるデータベース・オブジェクトへのアクセスを拒否する時間依存のポリシーなど、ポリシーの	いいえ

ポリシー・タイプ	ポリシー関数の実行	使用例	複数オブジェクト間での共有
	再実行します。	述語を問合せごとに生成する必要があるアプリケーション。	
STATIC	1 回。それから述語は SGA にキャッシュされます脚注 1	ビューの置換	いいえ
SHARED_STATIC	STATIC と同じ	同じ述語を複数のデータベース・オブジェクトに適用する必要があるデータ・ウェアハウスなどのホスト環境。	はい
CONTEXT_SENSITIVE	● 文の解析時。 ● 文の実行時(カーソルが最後に使用された後にローカル・アプリケーション・コンテキストが変更された場合)。	ポリシーによって異なるユーザーまたはグループに複数の述語が規定される、3 層セッション・プーリング・アプリケーション。	いいえ
SHARED_CONTEXT_SENSITIVE	データベース・セッションで最初にオブジェクトが参照されたとき。 述語はプライベート・セッション・メモリーである UGA にキャッシュされるため、ポリシー関数を複数のオブジェクト間で共有できます。	CONTEXT_SENSITIVE と同じですが、複数のオブジェクトがセッションの UGA からポリシー関数を共有できます。	はい

脚注1

述語が同じでも、同じカーソルを実行するたびに異なる行セットが生成される場合があります。これは、述語によるデータのフィルタ処理がSYS_CONTEXTやSYSDATEなどの属性によって異なるためです。

親トピック: Oracle Virtual Private Databaseポリシー・タイプを使用したパフォーマンスの最適化

14.4 例: Oracle Virtual Private Databaseポリシーの作成

このチュートリアルでは、簡単なデータベース・セッション・ベースのOracle Virtual Privateポリシーの作成方法と、ポリシー・グループの作成方法を説明します。

- 例: 単純なOracle Virtual Private Databaseポリシーの作成
  このチュートリアルでは、OEユーザー・アカウントを使用して単純なOracle Virtual Private Databaseポリシーを作成する方法を説明します。
- チュートリアル: セッション・ベースのアプリケーション・コンテキスト・ポリシーの実装
  このチュートリアルでは、データベース・セッション・ベースのアプリケーション・コンテキストを使用するOracle Virtual Private Databaseポリシーの作成方法を示します。

- 例: Oracle Virtual Private Databaseポリシー・グループの実装
  このチュートリアルでは、Oracle Virtual Private Databaseポリシー・グループの作成方法を示します。

親トピック: Oracle Virtual Private Databaseを使用したデータ・アクセスの制御

14.4.1 例: 単純なOracle Virtual Private Databaseポリシーの作成

このチュートリアルでは、OEユーザー・アカウントを使用して単純なOracle Virtual Private Databaseポリシーを作成する方法を説明します。

- このチュートリアルについて
  このチュートリアルでは、アクセスを営業担当者159が作成したOE.ORDERS表内の受注に制限するVPDポリシーの作成方法を示します。
- ステップ1: OEユーザー・アカウントがアクティブであることの確認
  まず、OEユーザー・アカウントが有効であることを確認する必要があります。
- ステップ2: ポリシー関数の作成
  次に、ポリシー関数を作成します。
- ステップ3: Oracle Virtual Private Databaseポリシーの作成
  ポリシー関数の作成後、その関数をVPDポリシーに関連付けます。
- ステップ4: ポリシーのテスト
  Oracle Virtual Private Databaseポリシーを作成した直後に、ポリシーは有効になります。
- ステップ5: このチュートリアルのコンポーネントの削除
  このチュートリアルのコンポーネントが不要になった場合、それらを削除できます。

親トピック: 例: Oracle Virtual Private Databaseポリシーの作成

14.4.1.1 このチュートリアルについて

このチュートリアルでは、アクセスを営業担当者159が作成したOE.ORDERS表内の受注に制限するVPDポリシーの作成方法を示します。

このポリシーは、基本的に次の文で表現されます。

```
SELECT * FROM OE.ORDERS;
```

この文を次のように変換します。

```
SELECT * FROM OE.ORDERS WHERE SALES_REP_ID = 159;
```

ノート:

マルチテナント環境を使用している場合、このチュートリアルは現在の PDB のみに適用されます。

親トピック: 例: 単純なOracle Virtual Private Databaseポリシーの作成

14.4.1.2 ステップ1: OEユーザー・アカウントがアクティブであることの確認

まず、OEユーザー・アカウントが有効であることを確認する必要があります。

1. SYSDBA管理権限を持つユーザーSYSとしてSQL*Plusにログインします。

   ```
   sqlplus sys as sysdba
   ```

```
Enter password: password
```

2. マルチテナント環境で、適切なPDBに接続します。

   たとえば:

   ```
   CONNECT SYS@hrpdb AS SYSDBA
   Enter password: password
   ```

   使用可能なPDBを検索するには、show pdbsコマンドを実行します。現在のPDBを確認するには、show con_nameコマンドを実行します。

3. OEのアカウント・ステータスを調べるには、DBA_USERSデータ・ディクショナリ・ビューを問い合せます。

   ```
   SELECT USERNAME, ACCOUNT_STATUS FROM DBA_USERS WHERE USERNAME = 'OE';
   ```

   ステータスはOPENである必要があります。DBA_USERSビューに、ユーザーOEがロックされて期限切れになっていると表示された場合は、次の文を入力して、OEアカウントのロックを解除し、新しいパスワードを作成します。

   ```
   ALTER USER OE ACCOUNT UNLOCK IDENTIFIED BY password;
   ```

   「パスワードの最低要件」のガイドラインに従って、passwordを安全なパスワードに置き換えます。セキュリティを向上させるため、以前のリリースのOracle Databaseで使用されたパスワードを再利用しないでください。

親トピック: 例: 単純なOracle Virtual Private Databaseポリシーの作成

14.4.1.3 ステップ2: ポリシー関数の作成

次に、ポリシー関数を作成します。

ユーザーSYSとして、次の関数を作成します。この関数はWHERE SALES_REP_ID = 159句を、OE.ORDERS表の任意のSELECT文に追加します。(最初の行のCREATE OR REPLACEの前にカーソルを置くことで、このテキストをコピーして貼り付けることができます。)

```
CREATE OR REPLACE FUNCTION auth_orders(
  schema_var IN VARCHAR2,
  table_var  IN VARCHAR2
 )
 RETURN VARCHAR2
 IS
  return_val VARCHAR2 (400);
 BEGIN
  return_val := 'SALES_REP_ID = 159';
  RETURN return_val;
 END auth_orders;
/
```

この例では、次のようになります。

- schema_varおよびtable_varは、スキーマ名(OE)および表名(ORDERS)を格納するために指定する入力パラメータを作成します。最初に、スキーマ用のパラメータを定義し、次に、オブジェクト(この例では表)用のパラメータを定義します。パラメータは常にこの順序で作成します。作成する仮想プライベート・データベース・ポリシーでは、OE.ORDERS表を指定するためにこれらのパラメータが必要です。
- RETURN VARCHAR2は、WHERE述語句に使用される文字列を返します。戻り値は常にVARCHAR2データ型になります。
- IS ... RETURN return_valは、WHERE SALES_REP_ID = 159述語の作成が含まれます。

親トピック: 例: 単純なOracle Virtual Private Databaseポリシーの作成

14.4.1.4 ステップ3: Oracle Virtual Private Databaseポリシーの作成

ポリシー関数の作成後、その関数をVPDポリシーに関連付けます。

- DBMS_RLSパッケージのADD_POLICYプロシージャを使用して、次のポリシーを作成します。

```
BEGIN
  DBMS_RLS.ADD_POLICY (
    object_schema    => 'oe',
    object_name      => 'orders',
    policy_name      => 'orders_policy',
    function_schema  => 'sys',
    policy_function  => 'auth_orders',
    statement_types  => 'select'
   );
 END;
/
```

この例では、次のようになります。

- `object_schema => 'oe'`は、保護するスキーマ(この例ではOE)を指定します。
- `object_name => 'orders'`は、保護するスキーマ内のオブジェクト(この例ではORDERS表)を指定します。
- `policy_name => 'orders_policy'`は、このポリシーの名前を`orders_policy`と指定します。
- `function_schema => 'sys'`は、`auth_orders`関数が作成されたスキーマを指定します。この例では、`auth_orders`はSYSスキーマに作成されています。ただし、通常は、セキュリティ管理者のスキーマに作成する必要があります。
- `policy_function => 'auth_orders'`は、ポリシーを規定する関数を指定します。この例では、「ステップ2: ポリシー関数の作成」で作成した`auth_orders`関数を指定します。
- `statement_types => 'select'`は、ポリシーを適用する操作を指定します。この例では、ユーザーが実行するすべてのSELECT文にポリシーが適用されます。

親トピック: 例: 単純なOracle Virtual Private Databaseポリシーの作成

14.4.1.5 ステップ4: ポリシーのテスト

Oracle Virtual Private Databaseポリシーを作成した直後に、ポリシーは有効になります。

ユーザー(スキーマの所有者を含む)が次にOE.ORDERSに対してSELECT文を実行すると、営業担当者159の受注のみにアクセスできます。

1. ユーザーOEとして接続します。

```
CONNECT oe -- Or, CONNECT OE@hrpdb
Enter password: password
```

2. 次のSELECT文を入力します。

```
SELECT COUNT(*) FROM ORDERS;
```

次の出力が表示されます。

```
 COUNT(*)
---------
```

```
         7
```

ポリシーはユーザーOEに対して有効です。ここに示すように、受注表の105行の内、7行のみが戻されます。

ただし、管理権限を持つユーザーは、表のすべての行にアクセスできます。

3. SYSDBA管理権限を持つユーザーSYSとして接続します。

```
CONNECT SYS AS SYSDBA -- Or, CONNECT SYS@hrpdb AS SYSDBA
Enter password: password
```

4. 次のSELECT文を入力します。

```
SELECT COUNT(*) FROM OE.ORDERS;
```

次の出力が表示されます。

```
 COUNT(*)
---------
       105
```

親トピック: 例: 単純なOracle Virtual Private Databaseポリシーの作成

14.4.1.6 ステップ5: このチュートリアルのコンポーネントの削除

このチュートリアルのコンポーネントが不要になった場合、それらを削除できます。

1. ユーザーSYSで、次のように関数とポリシーを削除します。

```
DROP FUNCTION auth_orders;
EXEC DBMS_RLS.DROP_POLICY('OE','ORDERS','ORDERS_POLICY');
```

2. OEアカウントをロックして期限切れにする必要がある場合は、次の文を入力します。

```
ALTER USER OE ACCOUNT LOCK PASSWORD EXPIRE;
```

親トピック: 例: 単純なOracle Virtual Private Databaseポリシーの作成

14.4.2 チュートリアル: セッション・ベースのアプリケーション・コンテキスト・ポリシーの実装

このチュートリアルでは、データベース・セッション・ベースのアプリケーション・コンテキストを使用するOracle Virtual Private Databaseポリシーの作成方法を示します。

- このチュートリアルについて
  この例では、データベース・セッション・ベースのアプリケーション・コンテキストを使用して、顧客が自分の注文のみを参照できるというポリシーを実装する方法を示しています。
- ステップ1: ユーザー・アカウントとサンプル表の作成
  まず、ユーザー・アカウントとサンプル表を作成します。
- ステップ2: データベース・セッション・ベースのアプリケーション・コンテキストの作成
  次に、データベース・セッション・ベースのアプリケーション・コンテキストを作成します。
- ステップ3: アプリケーション・コンテキストを設定するPL/SQLパッケージの作成
  アプリケーション・コンテキストの作成後、パッケージを作成してコンテキストを設定します。
- ステップ4: アプリケーション・コンテキストのPL/SQLパッケージを実行するログイン・トリガーの作成
  ログイン・トリガーがPL/SQLパッケージ・プロシージャを実行し、次回のユーザー・ログイン時にアプリケーション・コンテキストが設定されるようにします。

- ステップ5: ログイン・トリガーのテスト
  ログイン・トリガーは、sysadmin_vpd.orders_ctx_pkg.set_custnumプロシージャの実行時に、そのユーザーのアプリケーション・コンテキストを設定します。
- ステップ6: ユーザー・アクセスを自分の注文に制限するPL/SQLポリシー関数の作成
  次に、ユーザーの問合せの表示を制御するPL/SQL関数を作成します。
- ステップ7: 新しいセキュリティ・ポリシーの作成
  最後に、VPDセキュリティ・ポリシーを作成します。
- ステップ8: 新しいポリシーのテスト
  これで、すべてのコンポーネントが作成されたため、ポリシーをテストします。
- ステップ9: このチュートリアルのコンポーネントの削除
  このチュートリアルのコンポーネントが不要になった場合、それらを削除できます。

親トピック: 例: Oracle Virtual Private Databaseポリシーの作成

14.4.2.1 このチュートリアルについて

この例では、データベース・セッション・ベースのアプリケーション・コンテキストを使用して、顧客が自分の注文のみを参照できるというポリシーを実装する方法を示しています。

マルチテナント環境を使用している場合、このチュートリアルは現在のPDBのみに適用されます。

このチュートリアルでは、次の層のセキュリティを作成します。

1. ユーザーがログインするとき、データベース・セッション・ベースのアプリケーション・コンテキストによって顧客かどうかがチェックされます。顧客でないユーザーでもログインできますが、このユーザーはこの例で作成する注文表にアクセスできません。
2. ユーザーが顧客の場合はログインできます。顧客がログインした後、Oracle Virtual Private Databaseポリシーによって、このユーザーは自分の注文のみを参照できるように制限されます。
3. さらなる制限として、Oracle Virtual Private Databaseポリシーによって、ユーザーは注文を追加、変更、または削除できなくなります。

親トピック: チュートリアル: セッション・ベースのアプリケーション・コンテキスト・ポリシーの実装

14.4.2.2 ステップ1: ユーザー・アカウントとサンプル表の作成

まず、ユーザー・アカウントとサンプル表を作成します。

1. SQL*Plusを起動して、管理権限を持つユーザーとしてログインします。

```
sqlplus sys as sysdba
Enter password: password
```

2. マルチテナント環境で、適切なPDBに接続します。

   たとえば:

```
CONNECT SYS@hrpdb AS SYSDBA
Enter password: password
```

   利用可能なPDBを検索するには、DBA_PDBSデータ・ディクショナリ・ビューを問い合せます。現在のPDBを確認するには、show con_nameコマンドを実行します。

3. Oracle Virtual Private Databaseポリシーを管理する、次の管理ユーザーを作成します。

   次のSQL文では、このユーザーを作成してから、この例を終了するのに必要な権限をユーザーに付与します。

```
CREATE USER sysadmin_vpd IDENTIFIED BY password CONTAINER = CURRENT;
GRANT CREATE SESSION, CREATE ANY CONTEXT, CREATE PROCEDURE, CREATE TRIGGER,
ADMINISTER DATABASE TRIGGER TO sysadmin_vpd;
GRANT EXECUTE ON DBMS_SESSION TO sysadmin_vpd;
GRANT EXECUTE ON DBMS_RLS TO sysadmin_vpd;
```

「パスワードの最低要件」のガイドラインに従って、passwordを安全なパスワードに置き換えます。

4. 次のローカル・ユーザーを作成します。

```
CREATE USER tbrooke IDENTIFIED BY password CONTAINER = CURRENT;
CREATE USER owoods IDENTIFIED BY password CONTAINER = CURRENT;
GRANT CREATE SESSION TO tbrooke, owoods;
```

passwordを安全なパスワードに置き換えます。

5. この例で使用するサンプル・ユーザーSCOTTのアカウント・ステータスをチェックします。

```
SELECT USERNAME, ACCOUNT_STATUS FROM DBA_USERS WHERE USERNAME = 'SCOTT';
```

ステータスはOPENである必要があります。DBA_USERSビューに、ユーザーSCOTTがロックされて期限切れになっていると表示された場合は、次の文を入力して、SCOTTアカウントのロックを解除し、新しいパスワードを作成します。

```
ALTER USER SCOTT ACCOUNT UNLOCK IDENTIFIED BY password;
```

「パスワードの最低要件」のガイドラインに従って、passwordを安全なパスワードに置き換えます。セキュリティを向上させるため、以前のリリースのOracle Databaseで使用されたパスワードを再利用しないでください。

6. ユーザーSCOTTとして接続します。

```
CONNECT SCOTT -- Or, CONNECT SCOTT@hrpdb
Enter password: password
```

7. customers表を作成および移入します。

```
CREATE TABLE customers (
 cust_no    NUMBER(4),
 cust_email VARCHAR2(20),
 cust_name  VARCHAR2(20));
INSERT INTO customers VALUES (1234, 'TBROOKE', 'Thadeus Brooke');
INSERT INTO customers VALUES (5678, 'OWOODS', 'Oberon Woods');
```

ユーザーの電子メールIDを入力する際には、大文字で入力します。後でアプリケーション・コンテキストのPL/SQLパッケージを作成するときに、SYS_CONTEXTファンクションのSESSION_USERパラメータではユーザー名が大文字であると想定されます。大文字でないと、そのユーザー用のアプリケーション・コンテキストを設定できません。

8. ユーザーsysadmin_vpdには、ユーザーSCOTTと同様にcustomers表のSELECT権限が必要になるため、この権限をsysadmin_vpdに付与します。

```
GRANT READ ON customers TO sysadmin_vpd;
```

9. orders_tab表を作成して値を入力します。

```
CREATE TABLE orders_tab (
  cust_no  NUMBER(4),
  order_no NUMBER(4));
INSERT INTO orders_tab VALUES (1234, 9876);
INSERT INTO orders_tab VALUES (5678, 5432);
INSERT INTO orders_tab VALUES (5678, 4592);
```

10. ユーザーtbrookeとowoodsはorders_tab表を問い合せる必要があるため、これらのユーザーにREADオ

ブジェクト権限を付与します。

```
GRANT READ ON orders_tab TO tbrooke, owoods;
```

orders_tab受注表には、2名のサンプル顧客tbrookeとowoodsの購買レコードがあります。この段階では、これらの顧客がこの表を参照すると、すべての受注を参照できます。

親トピック: チュートリアル: セッション・ベースのアプリケーション・コンテキスト・ポリシーの実装

14.4.2.3 ステップ2: データベース・セッション・ベースのアプリケーション・コンテキストの作成

次に、データベース・セッション・ベースのアプリケーション・コンテキストを作成します。

1. ユーザーsysadmin_vpdで接続します。

```
CONNECT sysadmin_vpd -- Or, CONNECT sysadmin_vpd@hrpdb
Enter password: password
```

2. 次の文を入力します。

```
CREATE OR REPLACE CONTEXT orders_ctx USING orders_ctx_pkg;
```

この文は、orders_ctxアプリケーション・コンテキストを作成します。ユーザーsysadmin_vpdがこのコンテキストを作成してsysadmin_vpdスキーマに対応付けた場合でも、SYSスキーマがこのアプリケーション・コンテキストを所有することに注意してください。

親トピック: チュートリアル: セッション・ベースのアプリケーション・コンテキスト・ポリシーの実装

14.4.2.4 ステップ3: アプリケーション・コンテキストを設定するPL/SQLパッケージの作成

アプリケーション・コンテキストの作成後、パッケージを作成してコンテキストを設定します。

- ユーザーsysadmin_vpdとして、次のPL/SQLパッケージを作成します。このパッケージは、顧客tbrookeおよびowoodsがそれぞれのアカウントにログインすると、データベース・セッション・ベースのアプリケーション・コンテキストを設定します。

```
CREATE OR REPLACE PACKAGE orders_ctx_pkg IS
  PROCEDURE set_custnum;
 END;
/
CREATE OR REPLACE PACKAGE BODY orders_ctx_pkg IS
  PROCEDURE set_custnum
  AS
    custnum NUMBER;
  BEGIN
     SELECT cust_no INTO custnum FROM SCOTT.CUSTOMERS
        WHERE cust_email = SYS_CONTEXT('USERENV', 'SESSION_USER');
     DBMS_SESSION.SET_CONTEXT('orders_ctx', 'cust_no', custnum);
  EXCEPTION
   WHEN NO_DATA_FOUND THEN NULL;
  END set_custnum;
END;
/
```

この例では、次のようになります。

- custnum NUMBERは、顧客IDを保持するcustnum変数を作成します。
- SELECT cust_no INTO custnumは、SELECT文を実行し、scott.customers表のcust_no列データに格納されている顧客IDをcustnum変数にコピーします。

- `WHERE cust_email = SYS_CONTEXT('USERENV', 'SESSION_USER')`は、WHERE句を使用して、ログインしたユーザーのユーザー名と一致するすべての顧客IDを検索します。
- `DBMS_SESSION.SET_CONTEXT('orders_ctx', 'cust_no', custnum)`は、cust_no属性を作成した後、custnum変数に格納されている値に設定することで、orders_ctxアプリケーション・コンテキストの値を設定します。
- `EXCEPTION ... WHEN`は、`WHEN NO_DATA_FOUND`システム例外を追加して、`SELECT cust_no INTO custnum ...`文のSELECT文によって発生した可能性のある`no data found`エラーを捕捉します。

要約すると、`sysadmin_vpd.set_custnum`プロシージャは、セッション・ユーザーの顧客IDを選択してcustnum変数に格納することによって、ユーザーが登録済顧客かどうかを識別します。ユーザーが登録済顧客の場合、Oracle Databaseによってこのユーザーにアプリケーション・コンテキスト値が設定されます。ポリシー関数は、コンテキスト値を使用して、ユーザーが`orders_tab`表内のデータに対して持つアクセスを制御します。

親トピック: チュートリアル: セッション・ベースのアプリケーション・コンテキスト・ポリシーの実装

14.4.2.5 ステップ4: アプリケーション・コンテキストのPL/SQLパッケージを実行するログイン・トリガーの作成

ログイン・トリガーがPL/SQLパッケージ・プロシージャを実行し、次回のユーザー・ログイン時にアプリケーション・コンテキストが設定されるようにします。

- ユーザー`sysadmin_vpd`で、次のログイン・トリガーを作成します。

```
CREATE TRIGGER set_custno_ctx_trig AFTER LOGON ON DATABASE
 BEGIN
 sysadmin_vpd.orders_ctx_pkg.set_custnum;
 END;
/
```

関連トピック

- データベース・セッションのアプリケーション・コンテキスト・パッケージを実行するログオン・トリガー

親トピック: チュートリアル: セッション・ベースのアプリケーション・コンテキスト・ポリシーの実装

14.4.2.6 ステップ5: ログオン・トリガーのテスト

ログイン・トリガーは、`sysadmin_vpd.orders_ctx_pkg.set_custnum`プロシージャの実行時に、そのユーザーのアプリケーション・コンテキストを設定します。

1. ユーザー`tbrooke`として接続します。

```
CONNECT tbrooke -- For a CDB, connect to the PDB, e.g., @hrpdb
Enter password: password
```

2. 次の問合せを実行します。

```
SELECT SYS_CONTEXT('orders_ctx', 'cust_no') custnum FROM DUAL;
```

次の出力が表示されます。

```
EMP_ID
----------------------------------------------------------------
1234
```

親トピック: チュートリアル: セッション・ベースのアプリケーション・コンテキスト・ポリシーの実装

14.4.2.7 ステップ6: ユーザー・アクセスを自分の注文に制限するPL/SQLポリシー関数の作成

次のステップは、ユーザーの問合せの表示を制御するPL/SQL関数を作成します。

ログインしたユーザーがSELECT * FROM scott.orders_tab問合せを実行したときに、関数の出力がそのユーザーの発注に制限されるようにします。

1. ユーザーsysadmin_vpdで接続します。

```
CONNECT sysadmin_vpd -- Or, CONNECT sysadmin_vpd@hrpdb
Enter password: password
```

2. 次の関数を作成します。

```
CREATE OR REPLACE FUNCTION get_user_orders(
  schema_p   IN VARCHAR2,
  table_p    IN VARCHAR2)
 RETURN VARCHAR2
 AS
  orders_pred VARCHAR2 (400);
 BEGIN
  orders_pred := 'cust_no = SYS_CONTEXT(''orders_ctx'', ''cust_no'')';
 RETURN orders_pred;
END;
/
```

この関数は、「表示される注文がログインしたユーザーに属する場合」に変換されるWHERE述語を作成して返します。それからこのWHERE述語を、このユーザーがscott.orders_tab表に対して実行するあらゆる問合せに追加します。問合せが追加されると、この関数をorders_tab表に適用するOracle Virtual Private Databaseポリシーを作成できます。

親トピック: チュートリアル: セッション・ベースのアプリケーション・コンテキスト・ポリシーの実装

14.4.2.8 ステップ7: 新しいセキュリティ・ポリシーの作成

最後に、VPDセキュリティ・ポリシーを作成します。

- ユーザーsysadmin_vpdとして、DBMS_RLS.ADD_POLICYプロシージャを使用して、ポリシーを次のように作成します。

```
BEGIN
 DBMS_RLS.ADD_POLICY (
  object_schema    => 'scott',
  object_name      => 'orders_tab',
  policy_name      => 'orders_policy',
  function_schema  => 'sysadmin_vpd',
  policy_function  => 'get_user_orders',
  statement_types  => 'select',
  policy_type      => DBMS_RLS.CONTEXT_SENSITIVE,
  namespace        => 'orders_ctx',
  attribute        => 'cust_no');
END;
/
```

この文は、SCOTTスキーマで、orders_policyという名前のポリシーを作成して、顧客が自分の注文について問い合せるためのorders_tab表に適用します。get_user_orders関数が実装するこのポリシーは、sysadmin_vpdスキーマに格納されます。このポリシーは、さらに、ユーザーがSELECT文のみを発行するように制限します。namespaceおよびattributeパラメータは、以前に作成したアプリケーション・コンテキストを指定します。

14.4.2.9 ステップ8: 新しいポリシーのテスト

これで、すべてのコンポーネントが作成されたため、ポリシーをテストします。

1. ユーザーtbrookeとして接続します。

```
CONNECT tbrooke -- Or, CONNECT tbrooke@hrpdb
Enter password: password
```

ユーザーtbrookeは、アプリケーション・コンテキストに定義した要件を通過できるため、ログインできます。

2. ユーザーtbrookeで、購買レコードにアクセスします。

```
SELECT * FROM scott.orders_tab;
```

次の出力が表示されます。

```
   CUST_NO    ORDER_NO
----------  ----------
      1234        9876
```

ユーザーtbrookeは、2番目のテストを通過します。このユーザーは、scott.orders_tab表にある自分の注文のみにアクセスできます。

3. ユーザーowoodsとして接続し、購買レコードにアクセスします。

```
CONNECT owoods -- For a CDB, connect to the PDB, e.g., @hrpdb
Enter password: password
SELECT * FROM scott.orders_tab
```

次の出力が表示されます。

```
   CUST_NO    ORDER_NO
----------  ----------
      5678        5432
      5678        4592
```

ユーザーtbrookeとユーザーowoodsはログインでき、自分の注文のリストを参照できます。

次のことに注意してください。

- ユーザーの職位に基づいて複数の述語を作成できます。たとえば、営業担当者は自分の担当顧客のレコードのみを参照でき、受注入力担当はすべての顧客注文を参照できます。ユーザーの職位のコンテキスト値に基づいて別の述語を返すようにcustnum_sec機能を拡張できます。
- ファイングレイン・アクセス・コントロール・パッケージ内でアプリケーション・コンテキストを使用することによって、実際には解析済の文の中にバインド変数が指定されます。たとえば:

```
SELECT * FROM scott.orders_tab
WHERE cust_no = SYS_CONTEXT('order_entry', 'cust_num');
```

この文は完全に解析および最適化されますが、order_entryコンテキストに対するユーザーのcust_num属性値の評価は、実行時に行われます。これは、最適化された文には、その文を発行するユーザーごとに異なる形態で実行されるという利点があることを意味します。

ノート:

この例の関数のパフォーマンスは、cust_no に索引を作成するとさらに向上します。

- コンテキスト属性は、データベース表(複数も可)のデータ、またはLightweight Directory Access Protocol(LDAP)を使用するディレクトリ・サーバーのデータに基づいて設定できます。

ノート:

トリガーの詳細は、『Oracle Database PL/SQL 言語リファレンス』を参照してください。

動的に生成された述語の中でアプリケーション・コンテキストを使用するこの例と、述語の中で副問合せを使用するOracle Virtual Private Databaseポリシーについてを比較してください。

親トピック: チュートリアル: セッション・ベースのアプリケーション・コンテキスト・ポリシーの実装

14.4.2.10 ステップ9: このチュートリアルのコンポーネントの削除

このチュートリアルのコンポーネントが不要になった場合、それらを削除できます。

1. ユーザーSCOTTとして接続します。

```
CONNECT SCOTT -- Or, CONNECT SCOTT@hrpdb
Enter password: password
```

2. orders_tabおよびcustomers表を削除します。

```
DROP TABLE orders_tab;
DROP TABLE customers;
```

3. ユーザーSYSとしてAS SYSDBAで接続します。

```
CONNECT SYS AS SYSDBA -- Or, CONNECT SYS@hrpdb AS SYSDBA
Enter password: password
```

4. 次の文を実行して、この例で使用したコンポーネントを削除します。

```
DROP CONTEXT orders_ctx;
DROP USER sysadmin_vpd CASCADE;
DROP USER tbrooke;
DROP USER owoods;
```

親トピック: チュートリアル: セッション・ベースのアプリケーション・コンテキスト・ポリシーの実装

14.4.3 例: Oracle Virtual Private Databaseポリシー・グループの実装

このチュートリアルでは、Oracle Virtual Private Databaseポリシー・グループの作成方法を示します。

- このチュートリアルについて
  このチュートリアルでは、Oracle Virtual Private Database (VPD)を使用してポリシー・グループを作成する方法を示します。
- ステップ1: この例で使用するユーザー・アカウントと他のコンポーネントの作成
  まず、このチュートリアルのユーザー・アカウントと表を作成し、適切な権限を付与する必要があります。
- ステップ2: 2つのポリシー・グループの作成

次に、2人の非データベース・ユーザーprovider_aおよびprovider_bにそれぞれポリシー・グループを作成する必要があります。

- ステップ3: ポリシー・グループを制御するPL/SQLファンクションの作成
  ポリシー・グループには、ユーザーのデータ・アクセスをアプリケーションでどのように制御するかを定義する関数が必要です。
- ステップ4: 駆動アプリケーション・コンテキストの作成
  アプリケーション・コンテキストにより、ログインする非データベース・ユーザーが使用する必要があるポリシーが決定されます。
- ステップ5: PL/SQLファンクションのポリシー・グループへの追加
  必要な関数を作成したら、適切なポリシー・グループに関数を関連付ける必要があります。
- ステップ6: ポリシー・グループのテスト
  これで、2つのポリシー・グループをテストできます。
- ステップ7: このチュートリアルのコンポーネントの削除
  このチュートリアルのコンポーネントが不要になった場合、それらを削除できます。

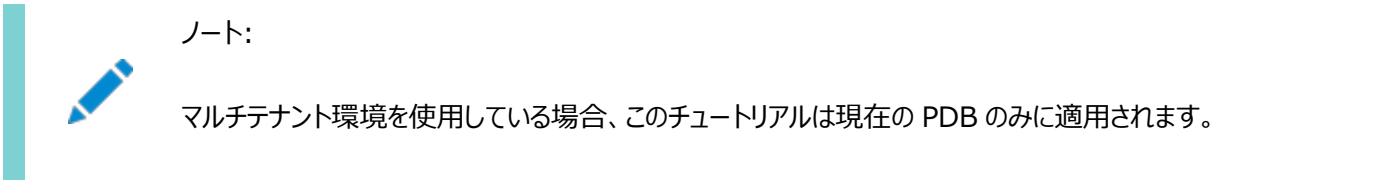
親トピック: 例: Oracle Virtual Private Databaseポリシーの作成

14.4.3.1 このチュートリアルについて

このチュートリアルでは、Oracle Virtual Private Database (VPD)を使用してポリシー・グループを作成する方法を示します。

アプリケーションで使用する一連のポリシーをグループ化する方法は、「Oracle Virtual Private Databaseのポリシー・グループ」を参照してください。非データベース・ユーザーがアプリケーションにログインすると、Oracle Databaseでは、適切なポリシー・グループ内で定義されたポリシーに基づいてユーザーにアクセス権が付与されます。

列レベルのアクセス制御の場合、各列または非表示の列セットが1つのポリシーで制御されます。この例では、2つの列セットを非表示にする必要があります。そのため、非表示にする列セットごとに1つずつ、2つのポリシーを作成する必要があります。ユーザーごとに必要なポリシーは1つのみのため、駆動アプリケーション・コンテキストによってポリシーが分割されます。



ノート:

マルチテナント環境を使用している場合、このチュートリアルは現在の PDB のみに適用されます。

親トピック: 例: Oracle Virtual Private Databaseポリシー・グループの実装

14.4.3.2 ステップ1: この例で使用するユーザー・アカウントと他のコンポーネントの作成

まず、このチュートリアルのユーザー・アカウントと表を作成し、適切な権限を付与する必要があります。

1. SYSDBA管理権限を持つユーザーSYSとしてログインします。

```
sqlplus sys as sysdba
Enter password: password
```

2. マルチテナント環境で、適切なPDBに接続します。

   たとえば:

```
CONNECT SYS@hrpdb AS SYSDBA
Enter password: password
```

   使用可能なPDBを検索するには、show pdbsコマンドを実行します。現在のPDBを確認するには、show

con_nameコマンドを実行します。

3. 次のローカル・ユーザーを作成します。

```
CREATE USER apps_user IDENTIFIED BY password CONTAINER = CURRENT;
GRANT CREATE SESSION TO apps_user;
CREATE USER sysadmin_pg  IDENTIFIED BY password CONTAINER = CURRENT;
GRANT CREATE SESSION, CREATE PROCEDURE, CREATE ANY CONTEXT TO sysadmin_pg;
```

「パスワードの最低要件」のガイドラインに従って、passwordを安全なパスワードに置き換えます。

4. ユーザーsysadmin_pgに次の権限を追加付与します。

```
GRANT EXECUTE ON DBMS_RLS TO sysadmin_pg;
```

5. ユーザーOEでログインします。

```
CONNECT OE -- Or, CONNECT OE@hrpdb
Enter password: password
```

OEアカウントがロックされて期限切れになっている場合、SYSDBA管理権限を持つユーザーSYSとして再接続してから、次の文を入力してアカウントのロックを解除し、新しいパスワードを指定します。

```
ALTER USER OE ACCOUNT UNLOCK IDENTIFIED BY password;
```

passwordを安全なパスワードに置き換えます。セキュリティを向上させるため、以前のリリースのOracle Databaseで使用されたパスワードを再利用しないでください。

6. product_code_names表を作成します。

```
CREATE TABLE product_code_names(
group_a     varchar2(32),
year_a      varchar2(32),
group_b     varchar2(32),
year_b      varchar2(32));
```

7. product_code_names表に値をいくつか挿入します。

```
INSERT INTO product_code_names values('Biffo','2008','Beffo','2004');
INSERT INTO product_code_names values('Hortensia','2008','Bunko','2008');
INSERT INTO product_code_names values('Boppo','2006','Hortensia','2003');
COMMIT;
```

8. product_code_names表のSELECT権限をapps_userユーザーに付与します。

```
GRANT SELECT ON product_code_names TO apps_user;
```

親トピック: 例: Oracle Virtual Private Databaseポリシー・グループの実装

14.4.3.3 ステップ2: 2つのポリシー・グループの作成

次に、2人の非データベース・ユーザーprovider_aおよびprovider_bにそれぞれポリシー・グループを作成する必要があります。

1. ユーザーsysadmin_pgで接続します。

```
CONNECT sysadmin_pg -- Or, CONNECT sysadmin_pg@hrpdb
Enter password: password
```

2. ユーザーprovider_aにより使用されるprovider_a_groupポリシー・グループを作成します。

```
BEGIN
```

```
 DBMS_RLS.CREATE_POLICY_GROUP(
 object_schema   => 'oe',
 object_name     => 'product_code_names',
 policy_group    => 'provider_a_group');
END;
/
```

3. ユーザーprovider_bにより使用されるprovider_b_groupポリシー・グループを作成します。

```
BEGIN
 DBMS_RLS.CREATE_POLICY_GROUP(
 object_schema   => 'oe',
 object_name     => 'product_code_names',
 policy_group    => 'provider_b_group');
END;
/
```

親トピック: 例: Oracle Virtual Private Databaseポリシー・グループの実装

14.4.3.4 ステップ3: ポリシー・グループを制御するPL/SQLファンクションの作成

ポリシー・グループには、ユーザーのデータ・アクセスをアプリケーションでどのように制御するかを定義する関数が必要です。

このポリシー・グループに作成する関数はユーザーprovider_aとprovider_bに適用されます。

1. ユーザーprovider_aがアクセスするデータを制限するvpd_function_provider_a関数を作成します。

```
CREATE OR REPLACE FUNCTION vpd_function_provider_a
 (schema in varchar2, tab in varchar2) return varchar2 as
  predicate  varchar2(8) default NULL;
  BEGIN
   IF LOWER(SYS_CONTEXT('USERENV','CLIENT_IDENTIFIER')) = 'provider_a'
    THEN predicate := '1=2';
   ELSE NULL;
  END IF;
  RETURN predicate;
END;
/
```

この関数では、ログインするユーザーが確かにユーザーprovider_aであるかどうかがチェックされます。間違いない場合、provider_aが表示できるのは、product_code_names表の列group_aおよびyear_a内のデータのみとなります。列group_bおよびyear_b内のデータは、provider_aに表示されません。つまり、predicate := '1=2'を設定すると関連列が非表示になります。「ステップ5: PL/SQLファンクションのポリシー・グループへの追加」では、これらの列をSEC_RELEVANT_COLSパラメータに指定します。

2. ユーザーprovider_aがアクセスするデータを制限するvpd_function_provider_b関数を作成します。

```
CREATE OR REPLACE FUNCTION vpd_function_provider_b
 (schema in varchar2, tab in varchar2) return varchar2 as
  predicate  varchar2(8) default NULL;
  BEGIN
   IF LOWER(SYS_CONTEXT('USERENV','CLIENT_IDENTIFIER')) = 'provider_b'
    THEN predicate := '1=2';
   ELSE NULL;
  END IF;
  RETURN predicate;
END;
/
```

vpd_function_provider_a関数と同様に、この関数では、ログインするユーザーが確かにユーザーprovider_bであるかどうかがチェックされます。間違いない場合、provider_bが表示できるのは、列group_b

およびyear_b内のデータのみであり、group_aおよびyear_a内のデータはprovider_bには表示されません。vpd_function_provider_a関数と同様に、predicate := '1=2'に指定すると、「ステップ5: PL/SQLファンクションのポリシー・グループへの追加」でSEC_RELEVANT_COLSパラメータに指定する関連列が非表示になります。

関連トピック

- 動的なWHERE句を生成する関数

親トピック: 例: Oracle Virtual Private Databaseポリシー・グループの実装

14.4.3.5 ステップ4: 駆動アプリケーション・コンテキストの作成

アプリケーション・コンテキストにより、ログインする非データベース・ユーザーが使用する必要があるポリシーが決定されます。

1. ユーザーsysadmin_pgで、次の駆動アプリケーション・コンテキストを作成します。

```
CREATE OR REPLACE CONTEXT provider_ctx USING provider_package;
```

2. アプリケーション・コンテキストに対してPL/SQL provider_packageパッケージを作成します。

```
CREATE OR REPLACE PACKAGE provider_package IS
 PROCEDURE set_provider_context (policy_group varchar2 default NULL);
END;
/
CREATE OR REPLACE PACKAGE BODY provider_package AS
 PROCEDURE set_provider_context (policy_group varchar2 default NULL) IS
 BEGIN
  CASE LOWER(SYS_CONTEXT('USERENV', 'CLIENT_IDENTIFIER'))
   WHEN 'provider_a' THEN
    DBMS_SESSION.SET_CONTEXT('provider_ctx','policy_group','PROVIDER_A_GROUP');
   WHEN 'provider_b' THEN
    DBMS_SESSION.SET_CONTEXT('provider_ctx','policy_group','PROVIDER_B_GROUP');
  END CASE;
 END set_provider_context;
END;
/
```

3. provider_ctxアプリケーション・コンテキストをproduct_code_names表に関連付け、名前を指定します。

```
BEGIN
 DBMS_RLS.ADD_POLICY_CONTEXT(
 object_schema  =>'oe',
 object_name    =>'product_code_names',
 namespace      =>'provider_ctx',
 attribute      =>'policy_group');
END;
/
```

4. apps_userアカウントにprovider_packageパッケージに対するEXECUTE権限を付与します。

```
GRANT EXECUTE ON provider_package TO apps_user;
```

親トピック: 例: Oracle Virtual Private Databaseポリシー・グループの実装

14.4.3.6 ステップ5: PL/SQLファンクションのポリシー・グループへの追加

必要な関数を作成したら、適切なポリシー・グループに関数を関連付ける必要があります。

1. vpd_function_provider_a関数をprovider_a_groupポリシー・グループに追加します。

```
BEGIN
```

```
 DBMS_RLS.ADD_GROUPED_POLICY(
 object_schema         => 'oe',
 object_name           => 'product_code_names',
 policy_group          => 'provider_a_group',
 policy_name           => 'filter_provider_a',
 function_schema       => 'sysadmin_pg',
 policy_function       => 'vpd_function_provider_a',
 statement_types       => 'select',
 policy_type           => DBMS_RLS.CONTEXT_SENSITIVE,
 sec_relevant_cols     => 'group_b,year_b',
 sec_relevant_cols_opt => DBMS_RLS.ALL_ROWS,
 namespace             => 'provider_ctx',
 attribute             => 'provider_group');
END;
/
```

sec_relevant_colsパラメータで指定したgroup_bおよびyear_b列は、ユーザーprovider_aに対して非表示になります。

2. vpd_function_provider_b関数をprovider_b_groupポリシー・グループに追加します。

```
BEGIN
 DBMS_RLS.ADD_GROUPED_POLICY(
 object_schema         => 'oe',
 object_name           => 'product_code_names',
 policy_group          => 'provider_b_group',
 policy_name           => 'filter_provider_b',
 function_schema       => 'sysadmin_pg',
 policy_function       => 'vpd_function_provider_b',
 statement_types       => 'select',
 policy_type           => DBMS_RLS.CONTEXT_SENSITIVE,
 sec_relevant_cols     => 'group_a,year_a',
 sec_relevant_cols_opt => DBMS_RLS.ALL_ROWS,
 namespace             => 'provider_ctx',
 attribute             => 'provider_group');
END;
/
```

sec_relevant_colsパラメータで指定したgroup_aおよびyear_a列は、ユーザーprovider_bに対して非表示になります。

親トピック: 例: Oracle Virtual Private Databaseポリシー・グループの実装

14.4.3.7 ステップ6: ポリシー・グループのテスト

これで、2つのポリシー・グループをテストできます。

1. ユーザーapps_userとして接続した後、次の文を入力して、後で作成する出力の書式が適切に設定されるようにします。

```
CONNECT apps_user -- Or, CONNECT apps_user@hrpdb
Enter password: password
col group_a format a16
col group_b format a16;
col year_a format a16;
col year_b format a16;
```

2. セッション識別子をprovider_aに設定します。

```
EXEC DBMS_SESSION.SET_IDENTIFIER('provider_a');
```

ここで、アプリケーションが識別子を設定します。識別子をprovider_aに設定すると、apps_userユーザーは、

provider_a_groupポリシー・グループ内の製品に使用可能な製品のみが表示されるユーザーとして設定されます。

3. provider_packageを実行し、コンテキストに基づいてポリシー・グループを設定します。

```
EXEC sysadmin_pg.provider_package.set_provider_context;
```

この時点で、アプリケーション・コンテキストが設定されたことを、次のように確認できます。

```
SELECT SYS_CONTEXT('USERENV', 'CLIENT_IDENTIFIER') AS END_USER FROM DUAL;
```

次の出力が表示されます。

```
END_USER
-----------------
provider_a
```

4. 次のSELECT文を入力します。

```
SELECT * FROM oe.product_code_names;
```

次の出力が表示されます。

```
GROUP_A          YEAR_A           GROUP_B          YEAR_B
---------------- ---------------- ---------------- ----------------
Biffo            2008
Hortensia        2008
Boppo            2006
```

5. クライアント識別子をprovider_bに設定し、次の文を入力します。

```
EXEC DBMS_SESSION.SET_IDENTIFIER('provider_b');
EXEC sysadmin_pg.provider_package.set_provider_context;
SELECT * FROM oe.product_code_names;
```

次の出力が表示されます。

```
GROUP_A          YEAR_A           GROUP_B          YEAR_B
---------------- ---------------- ---------------- ----------------
                                  Beffo            2004
                                  Bunko            2008
                                  Hortensia        2003
```

親トピック: 例: Oracle Virtual Private Databaseポリシー・グループの実装

14.4.3.8 ステップ7: このチュートリアルのコンポーネントの削除

このチュートリアルのコンポーネントが不要になった場合、それらを削除できます。

1. ユーザーOEとして接続します。

```
CONNECT OE -- Or, CONNECT OE@hrpdb
Enter password: password
```

2. product_code_names表を削除します。

```
DROP TABLE product_code_names;
```

3. SYSDBA管理権限を持つユーザーSYSとして接続します。

```
CONNECT SYS AS SYSDBA -- Or, CONNECT SYS@hrpdb AS SYSDBA
Enter password: password
```

4. このチュートリアルのアプリケーション・コンテキストとユーザーを削除します。

```
DROP CONTEXT provider_ctx;
DROP USER sysadmin_pg cascade;
DROP USER apps_user;
```

親トピック: 例: Oracle Virtual Private Databaseポリシー・グループの実装

14.5 他のOracle機能でのOracle Virtual Private Databaseの使用

Oracle Virtual Private DatabaseをOracleの他の機能と併用することの影響を理解しておく必要があります。

- Oracle Virtual Private Databaseポリシーとエディション
  エディションを扱う場合のOracle VPDの使用方法について理解しておく必要があります。
- VPD保護表に対するユーザーの問合せでのSELECT FOR UPDATE文
  原則として、ユーザーは、Virtual Private Database保護表を問い合せる場合にFOR UPDATE句を含めないようにします。
- Oracle Virtual Private Databaseポリシーと外部結合またはANSI結合
  Oracle Virtual Private Databaseでは、動的ビューを使用して、SQLをリライトします。
- Oracle Virtual Private Databaseセキュリティ・ポリシーとアプリケーション
  Oracle Virtual Private Databaseセキュリティ・ポリシーは、アプリケーション内ではなく、データベース内で適用されます。
- ファイングレイン・アクセス・コントロールのポリシー関数に対する自動再解析
  各ポリシーに対して最新の述語が使用されるように、ファイングレイン・アクセス・コントロール対応のオブジェクトに対する問合せでポリシー関数が実行されます。
- Oracle Virtual Private Databaseポリシーとフラッシュバック問合せ
  データベース上での操作では、直前にコミットされた使用可能データが使用されます。
- Oracle Virtual Private DatabaseとOracle Label Security
  Oracle Virtual Private DatabaseとOracle Label Securityは併用できますが、その場合、セキュリティの例外に注意してください。
- EXPDPユーティリティのaccess_methodパラメータを使用したデータのエクスポート
  VPDポリシーが定義されているオブジェクトからデータをエクスポートする場合は注意してください。
- ユーザー・モデルとOracle Virtual Private Database
  Oracle Virtual Private Databaseは、複数のタイプのユーザー・モデルで使用できます。

親トピック: Oracle Virtual Private Databaseを使用したデータ・アクセスの制御

14.5.1 Oracle Virtual Private Databaseポリシーとエディション

エディションを扱う場合のOracle VPDの使用方法について理解しておく必要があります。

アプリケーションをエディションベースの再定義用に準備し、アプリケーションで使用される各表をエディショニング・ビューで保護する場合は、これらの表を保護するVirtual Private Databaseポリシーをエディショニング・ビューに移動する必要があります。

エディション付きオブジェクトにVirtual Private Databaseポリシーが付加されている場合、そのオブジェクトが表示されるすべてのエディションにそのポリシーが適用されます。エディション付きオブジェクトが実現化されると、そのオブジェクトに付加されているVPDポリシーが新しい実際のオブジェクトに新たに付加されます。継承されたエディション付きオブジェクトに新たにVPDポリシーを適用すると、そのオブジェクトが実現化されます。

関連項目:

エディションの詳細は、『Oracle Database開発ガイド』を参照してください。

親トピック: 他のOracle機能でのOracle Virtual Private Databaseの使用

14.5.2 VPD保護表に対するユーザーの問合せでのSELECT FOR UPDATE文

原則として、ユーザーは、Virtual Private Database保護表を問い合せる場合にFOR UPDATE句を含めないようにします。

Virtual Private Databaseテクノロジは、VPDポリシー関数によって生成されたVPD述語を含むインライン・ビューに対するユーザーの問合せをリライトする処理に基づいています。このため、ビューに対する制限事項はVPD保護表にも同様に適用されます。VPD保護表に対するユーザーの問合せでSELECT文にFOR UPDATE句が含まれていると、ほとんどの場合、その問合せは動作しません。ただし、VPDによって生成されたインライン・ビューが非常に単純である場合、ユーザーの問合せが動作することがあります。

関連項目:

SELECT文のFOR UPDATE句の制限事項の詳細は、『Oracle Database SQL言語リファレンス』を参照してください。

親トピック: 他のOracle機能でのOracle Virtual Private Databaseの使用

14.5.3 Oracle Virtual Private Databaseポリシーと外部結合またはANSI結合

Oracle Virtual Private Databaseでは、動的ビューを使用して、SQLをリライトします。

SQLに外部結合操作またはANSI操作が含まれていると、一部のビューがマージされない場合や、一部の索引が使用されない場合があります。この問題は既知の最適化制限です。この問題に対処するには、SQLをリライトして外部結合操作やANSI操作が使用されないようにします。

親トピック: 他のOracle機能でのOracle Virtual Private Databaseの使用

14.5.4 Oracle Virtual Private Databaseセキュリティ・ポリシーとアプリケーション

Oracle Virtual Private Databaseセキュリティ・ポリシーは、アプリケーション内ではなく、データベース内で適用されます。

したがって、ユーザーが異なるアプリケーションを使用してデータにアクセスしようとしても、Oracle Virtual Private Databaseセキュリティ・ポリシーを回避できません。データベースにセキュリティ・ポリシーを作成するもう1つの利点は、複数のアプリケーションで各セキュリティ・ポリシーを保持するのではなく、1箇所でセキュリティ・ポリシーを保持できることです。このため、Oracle Virtual Private Databaseは、アプリケーション・ベースのセキュリティよりも強力なセキュリティを提供し、所有権のコストもより低くなります。

データにアクセスしているアプリケーションに応じて、異なるセキュリティ・ポリシーを規定する必要がある場合があります。Order Entry(受注管理)およびInventory(在庫管理)の2つのアプリケーションが、両方ともorders表にアクセスする場合を考えてみます。Inventoryアプリケーションで、製品の種類に基づいてアクセスを制限するポリシーを使用するとします。同時に、Order Entryアプリケーションでも、顧客番号に基づいてアクセスを制限するポリシーを使用するとします。

この場合、アプリケーションによるファイングレイン・アクセスの使用を分割する必要があります。分割しないと、2つのポリシーが自動的に連結され、目的とする結果が得られません。複数のポリシー・グループ、および特定のトランザクションに対して有効なポリシー・グループを判断する駆動アプリケーションのコンテキストを指定できます。また、データ・アクセスに対して必ず適用するデフォルト・ポリシーも指定できます。たとえば、ホスティングされたアプリケーションでは、データ・アクセスはサブスクライバIDによって制限

されます。

関連トピック

- 例: Oracle Virtual Private Databaseポリシー・グループの実装

親トピック: 他のOracle機能でのOracle Virtual Private Databaseの使用

14.5.5 ファイングレイン・アクセス・コントロールのポリシー関数に対する自動再解析

各ポリシーに対して最新の述語が使用されるように、ファイングレイン・アクセス・コントロール対応のオブジェクトに対する問合せでポリシー関数が実行されます。

たとえば、問合せを午前8時から午後5時の間に限定する時間ベースのポリシー関数の場合は、正午にカーソルの実行を解析すると、その時点でこのポリシー関数が実行され、問合せに対してポリシーが再度参照されます。カーソルが午前9時に解析された場合でも、そのカーソルが後で(たとえば、正午に)実行されると、Virtual Private Databaseポリシー関数が再び実行され、カーソルの実行が現時点(正午)でも引き続き許可されるようになります。これにより、常に最新のセキュリティ・チェックが実行されます。

Virtual Private Databaseポリシー関数の自動再実行は、ポリシーを追加するときに`DBMS_RLS.ADD_POLICY`設定の`STATIC_POLICY`を`TRUE`に設定した場合は行われません。この設定の場合、ポリシー関数は同じ述語を戻します。

親トピック: 他のOracle機能でのOracle Virtual Private Databaseの使用

14.5.6 Oracle Virtual Private Databaseポリシーとフラッシュバック問合せ

データベース上での操作では、直前にコミットされた使用可能データが使用されます。

フラッシュバック問合せ機能により、過去のどこかの時点でのデータベースの問合せが可能になります。

フラッシュバック問合せを使用するアプリケーションを作成するには、SQL問合せで`AS OF`句を使用して、時間とシステム変更番号(SCN)のどちらかを指定して、指定された時間からコミット済データに対して問い合せます。`DBMS_FLASHBACK` PL/SQLパッケージを使用することもできます。このPL/SQLパッケージでは必要なコードが増えますが、複数の操作を実行でき、これらの操作のすべてが同じ時点を参照します。

ただし、Oracle Virtual Private Databaseポリシーで保護されているデータベース・オブジェクトに対してフラッシュバック問合せを使用すると、現行のポリシーが過去のデータに適用されます。現行のOracle Virtual Private Databaseポリシーをフラッシュバック問合せのデータに適用すると、最新のビジネス・ポリシーが反映されるため、安全性が高まります。

関連項目:

- フラッシュバック問合せ機能、およびフラッシュバック問合せを使用するアプリケーションの作成方法の詳細は、『Oracle Database開発ガイド』を参照してください。
- `DBMS_FLASHBACK` PL/SQLパッケージの詳細は、『Oracle Database PL/SQLパッケージおよびタイプ・リファレンス』を参照してください。

親トピック: 他のOracle機能でのOracle Virtual Private Databaseの使用

14.5.7 Oracle Virtual Private DatabaseとOracle Label Security

Oracle Virtual Private DatabaseとOracle Label Securityは併用できますが、その場合、セキュリティの例外に注意し

てください。

- Oracle Virtual Private Databaseを使用したOracle Label Securityポリシーの規定
  Oracle Virtual Private Databaseポリシーを使用して、Oracle Label Securityユーザー認可に基づいて列または行レベルのアクセス制御を提供できます。
- Oracle Virtual Private DatabaseおよびOracle Label Securityの例外
  Oracle Virtual Private DatabaseおよびOracle Label Securityを使用する場合は、セキュリティ例外に注意してください。

親トピック: 他のOracle機能でのOracle Virtual Private Databaseの使用

14.5.7.1 Oracle Virtual Private Databaseを使用したOracle Label Securityポリシーの規定

Oracle Virtual Private Databaseポリシーを使用して、Oracle Label Securityユーザー認可に基づいて列または行レベルのアクセス制御を提供できます。

一般に、次のステップを実行する必要があります。

1. Oracle Label Securityのポリシーを作成したら、保護する必要がある表にこのポリシーを適用しないでください。(作成した仮想プライベート・データベース・ポリシーによって自動的に処理されます。)SA_SYSDBA.CREATE_POLICYプロシージャで、default_optionsパラメータをNO_CONTROLに設定します。
2. Oracle Label Securityのラベル・コンポーネントを作成し、通常どおりユーザーを認可します。
3. Oracle Virtual Private Databaseポリシーを作成する際には、次の操作を行います。
   - ポリシーについて作成するPL/SQLファンクションでは、Oracle Label SecurityのDOMINATESファンクションを使用して、ユーザーの認可をステップ2で作成したラベルと比較します。DOMINATESファンクションにより、ユーザーの認可が比較で使用されたラベルと等しい、またはラベルより機密性が高いかどうかが判別されます。ユーザー認可が通過した場合、ユーザーは列に対するアクセス権を付与されます。通過しない場合、ユーザーはアクセスを拒否されます。
   - 仮想プライベート・データベース・ポリシー定義で、保護する必要がある表にこのファンクションを適用します。DBMS_RLS.ADD_POLICYプロシージャで、機密性の高い列(SEC_RELEVANT_COLSパラメータ)および列のマスク(SEC_RELEVANT_COLS_OPTパラメータ)機能を使用して、Oracle Label Securityユーザー認可に基づいて列を表示または非表示にします。

関連項目:

支配ファンクションの詳細は、『Oracle Label Security管理者ガイド』を参照してください。

親トピック: Oracle Virtual Private DatabaseとOracle Label Security

14.5.7.2 Oracle Virtual Private DatabaseおよびOracle Label Securityの例外

Oracle Virtual Private DatabaseおよびOracle Label Securityを使用する場合は、次の例外に注意してください。

これらのセキュリティ例外を次に示します。

- データのエクスポート時、Oracle Virtual Private DatabaseおよびOracle Label Securityのポリシーはダイレクト・パス・エクスポート操作では規定されません。ダイレクト・パス・エクスポート操作では、Oracle Databaseはディスク

からバッファ・キャッシュにデータを読み込み、行をエクスポート・クライアントに直接転送します。

- Oracle Virtual Private DatabaseおよびOracle Label Securityのポリシーは、SYSスキーマのオブジェクトに適用できません。SYSユーザーおよびデータベースにDBA権限でアクセスするユーザー(CONNECT/AS SYSDBAなど)の場合、それらのユーザーのアクションにOracle Virtual Private DatabaseまたはOracle Label Securityのポリシーは適用されません。したがって、データベース・ユーザーSYSは、データベースからデータを抽出するために使用するエクスポート・モード、アプリケーションまたはユーティリティに関係なく、常にOracle Virtual Private DatabaseまたはOracle Label Securityの規定対象から除外されます。

  ただし、SYSDBAのアクションは、インストール時に監査を有効にし、監査証跡をオペレーティング・システムの保護位置に格納するように指定することによって監査できます。SYSユーザーはOracle Database Vaultを使用して詳細に監視できます。

- EXEMPT ACCESS POLICY権限を直接的またはデータベース・ロールを介して付与されているデータベース・ユーザーは、Oracle Virtual Private Databaseの規定対象から除外されます。システム権限EXEMPT ACCESS POLICYを持つユーザーは、すべてのSELECTまたはDML操作(INSERT、UPDATEおよびDELETE)において、ファイングレイン・アクセス・コントロール・ポリシーの対象から除外されます。これによって、インストールや、SYS以外のスキーマを介したデータベースのインポートとエクスポートなどの管理アクティビティが使いやすくなります。

  ただし、ポリシーを規定する次のオプションは、EXEMPT ACCESS POLICYが付与されている場合も規定されます。

  - INSERT_CONTROL、UPDATE_CONTROL、DELETE_CONTROL、WRITE_CONTROL、LABEL_UPDATEおよびLABEL_DEFAULT
  - Oracle Label SecurityのポリシーにALL_CONTROLオプションを指定した場合は、READ_CONTROLおよびCHECK_CONTROL以外のすべてが規定の対象として適用されます。

  EXEMPT ACCESS POLICYは、ファイングレイン・アクセス・コントロールを無効にするため、この権限は、ファイングレイン・アクセス・コントロールの規定を回避する正当な理由を持つユーザーに対してのみ付与する必要があります。この権限はWITH ADMIN OPTIONを使用して付与しないでください。これを使用すると、ユーザーが他のユーザーにEXEMPT ACCESS POLICY権限を譲渡して、ファイングレイン・アクセス・コントロールを回避する権限が伝播する可能性があります。

ノート:

- EXEMPT ACCESS POLICY 権限は、SELECT、INSERT、UPDATE および DELETE などのオブジェクト権限の規定には影響しません。これらのオブジェクト権限は、ユーザーに EXEMPT ACCESS POLICY 権限が付与されている場合も規定されます。
- Oracle Virtual Private Database が使用する SYS_CONTEXT 値は、フェイルオーバーのためのセカンダリ・データベースには伝播されません。

関連項目:

ダイレクト・パス・エクスポート操作の詳細は、『Oracle Databaseユーティリティ』を参照してください。

親トピック: Oracle Virtual Private DatabaseとOracle Label Security

14.5.8 EXPDPユーティリティのaccess_methodパラメータを使用したデータのエクスポート

VPDポリシーが定義されているオブジェクトからデータをエクスポートする場合は注意してください。

access_methodパラメータがdirect_pathに設定されたOracle Data Pump Export (EXPDP)ユーティリティを使用して、データをスキーマからエクスポートしようとするとき、このスキーマに仮想プライベート・データベース・ポリシーが定義されているオブジェクトが含まれていると、ORA-31696エラー・メッセージが表示される場合があり、エクスポート操作は失敗します。

エラー・メッセージは次のとおりです。

```
ORA-31696: unable to export/import TABLE_DATA:"schema.table" using client specified
DIRECT_PATH method
```

この問題は、スキーマ・レベルのエクスポートを、EXP_FULL_DATABASEロールを付与されていないユーザーとして実行する場合にのみ発生します。EXP_FULL_DATABASEロールを必要とするフル・データベース・エクスポート時には発生しません。EXP_FULL_DATABASEロールにはEXEMPT ACCESS POLICYシステム権限が含まれており、これにより仮想プライベート・データベース・ポリシーが無視されます。

潜在的な問題を見つけるには、EXPDP起動を再度試してください。ただし、access_methodパラメータをdirect_pathには設定しないでください。かわりに、automaticとexternal_tableのどちらかを使用してください。潜在的な問題とは、たとえば次のような権限の問題です。

```
ORA-39181: Only partial table data may be exported due to fine grain access control
on "schema_name"."object_name"
```

関連項目:

データ・ポンプ・エクスポートの使用に関する詳細は、『Oracle Databaseユーティリティ』を参照してください

親トピック: 他のOracle機能でのOracle Virtual Private Databaseの使用

14.5.9 ユーザー・モデルとOracle Virtual Private Database

Oracle Virtual Private Databaseは、複数のタイプのユーザー・モデルで使用できます。

これらのユーザー・モデルを次に示します。

- アプリケーション・ユーザーがデータベース・ユーザーでもある場合。Oracle Databaseでは、ユーザーがデータベース・ユーザーか、データベースに認識されないアプリケーション・ユーザーかに関係なく、アプリケーションはユーザーごとにファイングレイン・アクセス・コントロールを規定できます。アプリケーション・ユーザーがデータベース・ユーザーでもある場合、Oracle Virtual Private Databaseは次のように規定されます。ユーザーがデータベースに接続すると、アプリケーションは各セッションに対するアプリケーション・コンテキストを設定します（様々な種類のユーザー・セッション・データを取得するための多くのパラメータを提供する、デフォルトのUSERENVアプリケーション・コンテキスト・ネームスペースを使用できます)。各セッションは異なるユーザー名で開始されるため、各ユーザーに対して異なるファイングレイン・アクセス・コントロール条件を規定できます。
- OCIまたはJDBC/OCIを使用したプロキシ認証。プロキシ認証により、ユーザーごとに異なるファイングレイン・アクセス・コントロールが可能になります。これは、各セッション(OCIまたはJDBC/OCI)がそれぞれのアプリケーション・コンテキストを持つ個別のデータベース・セッションであるためです。
- エンタープライズ・ユーザー・セキュリティと統合されたプロキシ認証。エンタープライズ・ユーザー・セキュリティを使用してプ

ロキシ認証を統合した場合、Oracle Internet Directoryからユーザー・ロールおよび他の属性を取得してOracle Virtual Private Databaseポリシーを規定できます。(さらに、グローバルに初期化されたアプリケーション・コンテキストもこのディレクトリから取得できます。)

- ユーザーがOne Big Application Userとして接続する場合。すべてのユーザーのかわりにシングル・ユーザーとしてデータベースに接続するアプリケーションでは、ユーザーごとのファイングレイン・アクセス・コントロールを規定できます。この1セッションのユーザーは、「One Big Application User」と呼ばれます。ただし、アプリケーション開発者は、そのセッションのコンテキスト内で、個別のアプリケーション・ユーザー(たとえば、REALUSER)を表すグローバル・アプリケーション・コンテキスト属性を作成できます。すべてのデータベース・セッションおよびすべての監査レコードはOne Big Application Userに対して作成されますが、各セッションはエンド・ユーザーに応じて異なる属性を保持できます。このモデルは、ユーザーの数が限定されていて、セッションが再利用されないアプリケーションに最適です。このモデルの場合、各セッションは同一のデータベース・ユーザーとして作成されるため、ロールやデータベース監査の範囲が限定されます。
- Webベースのアプリケーション。Webベースのアプリケーションには、通常は何百人ものユーザーがいます。多数のユーザー要求に対するデータの取得をサポートするためにデータベースへの接続が持続的に行われる場合でも、このような接続は特定のWebベース・ユーザーに固有のものではありません。通常、Webベースのアプリケーションはスケーラビリティを提供するために、ユーザーごとに異なるセッションを保持するのではなく、接続を設定して再利用します。たとえば、WebユーザーJaneおよびAjitが中間層アプリケーションに接続すると、このアプリケーションは、2人のユーザーのかわりに、使用するデータベース・セッションを1つ確立します。通常、JaneもAjitもデータベースには認識されません。アプリケーションが接続するユーザー名を切り替えるため、どの時点でも、セッションを使用しているのはJaneまたはAjitのいずれかです。

  Oracle Virtual Private Databaseを使用すると、複数のグローバル・アプリケーション・コンテキストにアクセスする複数の接続が可能になり、接続プーリングが促進されます。この機能によって、個別ユーザー・セッションごとに別々のアプリケーション・コンテキストを確立する必要がなくなります。

表14-3に、Oracle Virtual Private Databaseをユーザー・モデルに適用する方法の要約を示します。

表14-3 様々なユーザー・モデルでのOracle Virtual Private Database

ユーザー・モデル	個別のデータベース接続	ユーザーごとの個別アプリケーション・コンテキスト	単一データベース接続	アプリケーションによるユーザー名の切替え
アプリケーション・ユーザーがデータベース・ユーザーでもある場合	はい	はい	いいえ	いいえ
OCI または JDBC/OCI を使用したプロキシ認証	はい	はい	いいえ	いいえ
エンタープライズ・ユーザー・セキュリティ脚注 2と統合されたプロキシ認証	いいえ	いいえ	はい	はい
One Big Application User	いいえ	いいえ脚注 3	いいえ	はい 2

ユーザー・モデル	個別のデータベース接続	ユーザーごとの個別アプリケーション・コンテキスト	単一データベース接続	アプリケーションによるユーザー名の切替え
Web ベースのアプリケーション	いいえ	いいえ	はい	はい

脚注2

ユーザー・ロールとその他の属性(グローバルに初期化されたアプリケーション・コンテキストも含む)は、Oracle Internet Directoryから取得してOracle Virtual Private Databaseを規定できます。

脚注3

アプリケーション開発者は、個別アプリケーション・ユーザー(たとえば、REALUSER)を表すグローバル・アプリケーション・コンテキスト属性を作成できます。その後、この属性を使用して、各セッション属性を制御したり、監査することができます。

関連トピック

- グローバル・アプリケーション・コンテキスト

親トピック: 他のOracle機能でのOracle Virtual Private Databaseの使用

14.6 Oracle Virtual Private Databaseのデータ・ディクショナリ・ビュー

Oracle Databaseには、Oracle Virtual Private Databaseポリシーに関する情報を表示するデータ・ディクショナリ・ビューが用意されています。

表14-4に、仮想プライベート・データベース固有のビューを示します

表14-4 VPDポリシーに関する情報を表示するデータ・ディクショナリ・ビュー

ビュー	説明
ALL_POLICIES	現行ユーザーがアクセス可能なオブジェクトに対するすべての Oracle Virtual Private Database セキュリティ・ポリシーが表示されます。
ALL_POLICY_ATTRIBUTES	ログインしたユーザーが VPD ポリシーの所有者であるか、VPD ポリシーが PUBLIC に属する場合のすべてのアプリケーション・コンテキスト・ネームスペース、属性および仮想プライベート・データベース・ポリシーの関連付けが表示されます。
ALL_POLICY_CONTEXTS	現行ユーザーがアクセス可能なシノニム、表およびビューに定義されている駆動コンテキストが表示されます。駆動コンテキストは、Oracle Virtual Private Database ポリシーで使用されるアプリケーション・コンテキストです。
ALL_POLICY_GROUPS	現行ユーザーがアクセス可能なシノニム、表およびビューに定義されている Oracle Virtual Private Database ポリシー・グループが表示されます。
ALL_SEC_RELEVANT_COLS	現行ユーザーがアクセス可能な表とビューに対するセキュリティ・ポリシーのセキュリティ関連列が表示されます。

ビュー	説明
DBA_POLICIES	データベース内のすべての Oracle Virtual Private Database セキュリティ・ポリシーが表示されます。
DBA_POLICY_ATTRIBUTES	状況依存および共有の状況依存仮想プライベート・データベース・ポリシーのすべてのアプリケーション・コンテキスト・ネームスペース、属性および仮想プライベート・データベース・ポリシーの関連付けが表示されます。
DBA_POLICY_GROUPS	データベース内のすべてのポリシー・グループが表示されます。
DBA_POLICY_CONTEXTS	データベース内のすべての駆動コンテキストが表示されます。このビューの列は、ALL_POLICY_CONTEXTS の各列と同じです。
DBA_SEC_RELEVANT_COLS	データベース内のすべてのセキュリティ・ポリシーのセキュリティ関連列が表示されます。
UNIFIED_AUDIT_TRAIL	統合監査およびファイングレイン監査のために、RLS_INFO 列に VPD 述語を取得します。
USER_POLICIES	現行ユーザーが所有するオブジェクトに対応付けられたすべての Oracle Virtual Private Database セキュリティ・ポリシーが表示されます。このビューには OBJECT_OWNER 列が表示されません。
USER_POLICY_ATTRIBUTES	仮想プライベート・データベース・ポリシーの所有者が現在のユーザーである場合のすべてのアプリケーション・コンテキスト・ネームスペース、属性および仮想プライベート・データベース・ポリシーの関連付けが表示されます。
USER_POLICY_CONTEXTS	現行ユーザーが所有するシノニム、表およびビューに定義されている駆動コンテキストが表示されます。このビューの列は、OBJECT_OWNER 列以外は ALL_POLICY_CONTEXTS の各列と同じです。
USER_SEC_RELEVANT_COLS	現行ユーザーが所有している表とビューに対するセキュリティ・ポリシーのセキュリティ関連列が表示されます。このビューの列は、OBJECT_OWNER 列以外は ALL_SEC_RELEVANT_COLS の各列と同じです。
USER_POLICY_GROUPS	現行ユーザーが所有するシノニム、表およびビューに定義されているポリシー・グループが表示されます。このビューには OBJECT_OWNER 列が表示されません。
V$VPD_POLICY	現在の PDB の場合、現在ライブラリ・キャッシュにキャッシュされているカーソルに対応付けられた、すべてのファイングレイン・セキュリティ・ポリシーと述語が表示されます。このビューは、SQL 文に適用されたポリシーを検索するのに便利です。

ヒント:

仮想プライベート・データベース・ポリシーを使用するアプリケーションでエラーが見つかった場合は、これらのビューに加え、データベース・トレース・ファイルも確認してください。USER_DUMP_DEST 初期化パラメータは、トレース・ファイルの現在の位置を示します。このパラメータの値は、SQL*Plus で SHOW PARAMETER USER_DUMP_DEST を発行して確認できます。

関連トピック

- Oracle Databaseリファレンス
- Oracle Database SQLチューニング・ガイド

親トピック: Oracle Virtual Private Databaseを使用したデータ・アクセスの制御

15 透過的機密データ保護の使用

透過的機密データ保護により、機密データを保持するデータベースのすべての表の列を確認できます。

- 透過的機密データ保護について
  透過的機密データ保護は、機密情報を保持する列を表から探し出して分類するための手段です。
- 透過的機密データ保護を使用する一般的なステップ
  Oracle Data RedactionでTSDPを使用するには、次の一般的なステップに従う必要があります。
- 透過的機密データ保護ポリシーのユースケース
  透過的機密データ保護には、次の利点があります。
- 透過的機密データ保護の使用に必要な権限
  透過的機密データ保護を使用するには、次のPL/SQLパッケージに対するEXECUTE権限が必要です。
- マルチテナント環境が透過的機密データ保護に影響を与えるしくみ
  マルチテナント環境では、透過的機密データ保護ポリシーを現在のPDBまたは現在のアプリケーションPDBのみに適用できます。
- 透過的機密データ保護ポリシーの作成
  機密タイプを作成し、保護する機密列を特定して、それらの列をADMからデータベースにインポートする必要があります。
- 透過的機密データ保護ポリシーの変更
  `DBMS_TSDP_PROTECT.ALTER_POLICY`プロシージャで、TSDPポリシーを変更できます。
- 透過的機密データ保護ポリシーの無効化
  `DBMS_TSDP_PROTECT.DISABLE_PROTECTION_COLUMN`プロシージャは1つまたはすべてのTSDPポリシーを無効にします。
- 透過的機密データ保護ポリシーの削除
  TSDPポリシー全体を削除するか、またはポリシーから条件有効オプションの組合せを削除できます。
- 事前定義のREDACT_AUDITポリシーを使用したバインド値のマスク
  事前定義の`REDACT_AUDIT`ポリシーはバインド値をマスクします。バインド値は、イベントの設定時にトレース・ファイルに表示されます。
- データ・リダクションでの透過的機密データ保護ポリシー
  Oracle Data Redaction機能を透過的機密データ保護ポリシーで使用できます。
- Oracle VPDポリシーでの透過的機密データ保護ポリシーの使用
  TSDPとOracle Virtual Private Databaseの保護を1つのポリシーに統合できます。
- 統合監査での透過的機密データ保護ポリシーの使用
  透過的機密データ保護および統合監査プロシージャを使用すると、これら2つの機能の保護を組み合せることができます。
- ファイングレイン監査での透過的機密データ保護ポリシーの使用
  透過的機密データ保護およびファイングレイン監査プロシージャを使用すると、これら2つの機能の保護を組み合せることができます。
- TDE列暗号化での透過的機密データ保護ポリシーの使用
  TSDPプロシージャと透過的データ暗号化の列暗号化文を使用すると、これら2つの機能の保護を組み合せることができます。
- 透過的機密データ保護のデータ・ディクショナリ・ビュー
  Oracle Databaseには、透過的機密データ保護ポリシーに関する情報を表示するデータ・ディクショナリ・ビューが用

意されています。

親トピック: データへのアクセス制御

15.1 透過的機密データ保護について

透過的機密データ保護は、機密情報を保持する列を表から探し出して分類するための手段です。

この機能を使用して、機密データを保持する列をデータベース表で特定し、このデータを分類して、指定されたクラスのこのデータ全体を保護するポリシーを作成できます。この種の機密データの例としては、クレジット・カード番号や社会保障番号などがあります。

TSDPポリシーは、次にOracle Data RedactionまたはOracle Virtual Private Databaseのいずれかの設定を使用して、これらの表の列の機密データを保護します。TSDPポリシーが保護する表の列レベルで適用され、クレジット・カード情報を含むすべてのNUMBERデータ型の列などの特定の列のデータ型を対象とします。分類するすべてのデータの均一のTSDPポリシーを作成して、コンプライアンス規制が変更される場合に必要に応じてこのポリシーを変更できます。オプションで、他のデータベースで使用するTSDPポリシーをエクスポートできます。

TSDPポリシーの利点は多くあります。多数のデータベースを使用する大きい組織全体にTSDPポリシーを簡単に作成および適用できます。これにより、TSDPポリシーが対象とするデータの保護を見積もることが可能なため、監査者にとって非常に便利です。TSDPは、似たセキュリティ制限を持つデータが多くあり、ポリシーをこのすべてのデータに一貫して適用する必要がある政府の環境に特に役立ちます。ポリシーは、リダクション、暗号化、そのアクセスの制御、そのアクセスの監査、監査証跡でのマスクを行うことができます。TSDPを使用しない場合、各リダクション・ポリシー、列レベルの暗号化構成および列ごとの仮想プライベート・データベース・ポリシーを構成する必要があります。

親トピック: 透過的機密データ保護の使用

15.2 透過的機密データ保護を使用する一般的なステップ

Oracle Data RedactionでTSDPを使用するには、次の一般的なステップに従う必要があります。

1. 機密タイプを作成して、保護する列のタイプを分類します。

   たとえば、すべての社会保障番号やクレジット・カード番号を分類するための機密タイプを作成できます。機密タイプを作成するには、DBMS_TSDP_MANAGE.ADD_SENSITIVE_TYPE PL/SQLプロシージャを使用するか、Enterprise Manager Cloud Controlアプリケーション・データ・モデルを使用します。アプリケーション・データ・モデルから1つの操作で複数の機密タイプを追加するには、DBMS_TSDP_MANAGE.IMPORT_SENSITIVE_TYPESプロシージャを使用できます。

2. 機密タイプに関連付けられる機密列のリストを識別します。

   このリストを決定および生成するには、次の方法のいずれかを使用できます。

   - DBMS_TSDP_MANAGE.ADD_SENSITIVE_COLUMNプロシージャは、機密列を個々に識別します。
   - Oracle Enterprise Manager Cloud Controlアプリケーション・データ・モデルにより、機密列のグループを識別できます。XML形式でこのリストの機密列を準備し、データベースにインポートします。

3. ステップ2のアプリケーション・データ・モデルを使用した場合、DBMS_TSDP_MANAGE.IMPORT_DISCOVERY_RESULTプロシージャを使用して、アプリケーション・データ・モデルから機密列のリストをデータベースにインポートします。
4. 使用するデータ・リダクションまたは仮想プライベート・データベースの設定を定義する無名ブロック内の

DBMS_TSDP_PROTECT.ADD_POLICYプロシージャを使用して、TSDPポリシーを作成します。

5. DBMS_TSDP_PROTECT.ASSOCIATE_POLICYプロシージャを使用して、TSDPポリシーを1つ以上の機密タイプと関連付けます。
6. DBMS_TSDP_PROTECT.ENABLE_PROTECTION_SOURCE、DBMS_TSDP_PROTECT.ENABLE_PROTECTION_COLUMNまたはDBMS_TSDP_PROTECT.ENABLE_PROTECTION_TYPEプロシージャを使用して、TSDPポリシー保護を有効化します。
7. オプションで、Oracle Data Pumpを使用してフル・データベース・エクスポートを実行して、TSDPポリシーを他のデータベースにエクスポートします。(TSDPポリシーを個々にエクスポートできません。)

親トピック: 透過的機密データ保護の使用

15.3 透過的機密データ保護ポリシーのユースケース

透過的機密データ保護には、次の利点があります。

これらの利点は次のとおりです。

- 機密データ保護を一度構成して、必要に応じてこの保護をデプロイします。 透過的機密データ保護ポリシーを構成して、実際にターゲット・データを指定することなくどのようにデータのクラス(たとえば、クレジット・カード列など)を保護する必要があるかを指定できます。つまり、透過的機密データ保護ポリシーを作成する場合、保護する実際のターゲット列の参照を含む必要がありません。透過的機密データ保護ポリシーは、データベースの機密列のリストおよび指定された機密タイプのポリシーの関連付けに基づいて、これらのターゲット列を確認します。これは、透過的機密データ保護ポリシーを作成した後に追加の機密データをデータベースに追加する場合に便利です。ポリシーを作成した後、単一のステップで機密データの保護を有効化できます(たとえば、ソース・データベース全体に基づく保護の有効化など)。新しいデータの機密タイプ、機密タイプおよびポリシーの関連付けは、機密データの保護方法を決定します。これによって、新しい機密データが追加される場合、現在の透過的機密データ保護ポリシーの要件を満たしているかぎり、保護を構成する必要がありません。
- 複数の機密列の保護を管理できます。適切な属性(ソース・データベースの識別、機密タイプ自体、特定のスキーマ、表、列など)に基づいて複数の機密列の保護を有効化または無効化できます。この精度により、データ・セキュリティの上位レベルの制御が提供されます。この機能の設計により、これらのコンプライアンス規制の範囲に該当する大きいデータ・セットの特定のコンプライアンス要件に基づいてデータ・セキュリティを管理できます。個別の列ごとではなく特定のカテゴリに基づいて、データ・セキュリティを構成できます。たとえば、クレジット・カード番号または社会保障番号の保護を構成できますが、このデータを含むデータベースの各列の保護を構成する必要はありません。
- Oracle Enterprise Manager Cloud Controlアプリケーション・データ・モデル(ADM)機能を使用して識別された機密列を保護できます。Cloud Control ADM機能を使用して、機密タイプを作成し、機密列のリストで検出できます。次に、このリストの機密列および対応する機密タイプをデータベースにインポートできます。そこから、この情報を使用して、透過的機密データ保護ポリシーを作成および管理できます。

親トピック: 透過的機密データ保護の使用

15.4 透過的機密データ保護の使用に必要な権限

透過的機密データ保護を使用するには、次のPL/SQLパッケージに対するEXECUTE権限が必要です。

これらの権限は次のとおりです。

- 機密列および機密タイプをデータベースにインポートして管理できるDBMS_TSDP_MANAGE。このパッケージのプロシージャは、実行者権限で実行されます。通常、アプリケーション・データベース管理者には、このパッケージの権限が付与されます。
- TSDPポリシーの作成に使用されるDBMS_TSDP_PROTECT。このパッケージのプロシージャは、実行者権限で実行されます。通常、セキュリティ・データベース管理者には、このパッケージの権限が付与されます。
- データ・リダクション・ポリシーを作成する場合のDBMS_REDACT。通常、セキュリティ・データベース管理者には、このパッケージの権限が付与されます。
- Oracle Virtual Private Database機能をTSDPポリシーに組み込む場合のDBMS_RLS。通常、セキュリティ・データベース管理者には、このパッケージの権限が付与されます。

業務分離を強化するために、アプリケーション・データベース管理者がTSDPポリシー作成の1つの領域(DBMS_TSDP_MANAGEパッケージの場合と同様)を制御するか、セキュリティ・データベース管理者(DBMS_TSDP_PROTECT、DBMS_REDACTおよびDBMS_RLSパッケージ用)が制御できるように、これらのパッケージが設計されています。

親トピック: 透過的機密データ保護の使用

15.5 マルチテナント環境が透過的機密データ保護に影響を与えるしくみ

マルチテナント環境では、透過的機密データ保護ポリシーを現在のPDBまたは現在のアプリケーションPDBのみに適用できます。

Enterprise Manager Cloud Controlアプリケーション・データ・モデルを使用している場合、PDB内のローカルおよび共通アプリケーション・オブジェクト(つまり、現在のPDB内で表示およびアクセスできる共通オブジェクト)に属する機密列を探すことができます。これにより、TSDPポリシーを使用して、PDBに対するローカル・オブジェクトと、PDBからアクセス可能な共通オブジェクトの両方を保護できます。

アプリケーション・ルート:

- アプリケーション・コンテナ全般:
  - アプリケーションのインストール、アップグレード、パッチ適用またはアンインストール操作用のスクリプトを作成する際、`ALTER PLUGGABLE DATABASE app_name BEGIN INSTALL`および`ALTER PLUGGABLE DATABASE app_name END INSTALL`ブロック内にSQL文を含めて、様々な操作を実行できます。これらのブロックにTSDP文を含めると、そのTSDP文は失敗します。ただし、これらのブロック外のスクリプトにTSDP文を含めることができます。
- アプリケーション・ルート:
  - アプリケーション共通オブジェクトと、アプリケーション・ルートのローカル・オブジェクトの両方でTSDP操作を実行できます。
  - アプリケーション・ルート・コンテナで定義されたTSDPポリシーは、アプリケーション・ルートに対するローカル・ポリシーであるかのように動作します。つまり、このポリシーが有効なのはアプリケーション・ルート・コンテナ内のみです。

アプリケーションPDB:

- アプリケーションPDBを保護するセキュリティ・ポリシーは、ローカル・アプリケーション・オブジェクトに対して実行されるTSDP操作に適用されます。
- アプリケーションPDBを保護するセキュリティ・ポリシーは、PDBからアクセス可能なアプリケーション共通オブジェクトに対

して実行されるTSDP操作に適用されます。ただし、アプリケーションPDB外のアプリケーション共通オブジェクトへのアクセスは、アプリケーションPDBを保護するセキュリティ・ポリシーによって拘束されません。

DBA_TSDP_POLICY_FEATUREデータ・ディクショナリ・ビューを問い合せて、TSDPポリシー、および関連付けられているセキュリティ機能のリストを確認できます。すべてのPDBを検索するには、DBA_PDBSビューを問い合せます。

親トピック: 透過的機密データ保護の使用

15.6 透過的機密データ保護ポリシーの作成

機密タイプを作成し、保護する機密列を特定して、それらの列をADMからデータベースにインポートする必要があります。

- ステップ1: 機密タイプの作成
  機密タイプは、機密として指定するデータのクラスです。
- ステップ2: 保護する機密列の識別
  機密列を定義したら、保護する列を特定します。
- ステップ3: ADMからデータベースへの機密列リストのインポート
  次に、ADMからデータベースに機密列リストをインポートします。
- ステップ4: 透過的機密データ保護ポリシーの作成
  機密列のリストを作成し、このリストをデータベースにインポートしたら、透過的機密データ保護ポリシーを作成します。
- ステップ5: ポリシーと機密タイプの関連付け
  DBMS_TSDP_PROTECT.ASSOCIATE_POLICYプロシージャで、TSDPポリシーと機密タイプを関連付けます。
- ステップ6: 透過的機密データ保護ポリシーの有効化
  保護されるソースの現在のデータベース、特定の表の列または特定の列タイプに対してTSDPポリシーを有効できます。
- ステップ7: 他のデータベースへのポリシーのエクスポート(オプション)
  別のデータベースに対してポリシーをエクスポートまたはインポートできます。

親トピック: 透過的機密データ保護の使用

15.6.1 ステップ1: 機密タイプの作成

機密タイプは、機密として指定するデータのクラスです。

たとえば、すべてのクレジット・カード番号のcredit_card_type機密タイプを作成できます。

- 機密タイプを作成するには、Enterprise Manager Cloud Controlアプリケーション・データ・モデルから機密タイプを作成するか、DBMS_TSDP_MANAGE.ADD_SENSITIVE_TYPE PL/SQLプロシージャを使用します。

  機密タイプを削除するには、DBMS_TSDP_MANAGE.DROP_SENSITIVE_TYPEプロシージャを使用します。

たとえば、機密タイプcredit_card_num_typeを作成するには:

```
BEGIN
 DBMS_TSDP_MANAGE.ADD_SENSITIVE_TYPE (
  sensitive_type  => 'credit_card_num_type',
  user_comment    => 'Type for credit card columns using a number data type');
END;
/
```

この例では、次のようになります。

- sensitive_type: 取得する機密タイプを説明する名前を作成します。この値は大/小文字を区別するので、後で参照する場合は、作成した大/小文字を使用していることを確認してください。

DBA_SENSITIVE_COLUMN_TYPESデータ・ディクショナリ・ビューを問い合せて、既存の機密タイプを確認できます。

- user_comment: オプションで、機密タイプの説明を入力します。

関連項目:

- アプリケーション・データ・モデルの詳細は、Oracle Database Testingガイドを参照してください。
- DBMS_TSDP_MANAGE.ADD_SENSITIVE_TYPE PL/SQLプロシージャの詳細は、『Oracle Database PL/SQLパッケージおよびタイプ・リファレンス』を参照してください。

親トピック: 透過的機密データ保護ポリシーの作成

15.6.2 ステップ2: 保護する機密列の識別

機密列を定義したら、保護する列を特定します。

保護する列を識別するには、定義した機密タイプに基づいて、Enterprise Manager Cloud Controlアプリケーション・データ・モデルを使用してこれらの列を識別するか、DBMS_TSDP_MANAGE.ADD_SENSITIVE_COLUMNプロシージャを使用できます。

データベースの機密列のリストから列を削除するには、DBMS_TSDP_MANAGE.DROP_SENSITIVE_COLUMNプロシージャを使用します。

1. 使用する機密タイプを確認します。

   たとえば:

```
SELECT NAME FROM DBA_SENSITIVE_COLUMN_TYPES;
NAME
--------------------
credit_card_num_type
```

2. DBMS_TSDP_MANAGE.ADD_SENSITIVE_COLUMNプロシージャを実行して、機密タイプと表の列を関連付けます。機密タイプの作成に使用された大/小文字を使用してsensitive_typeパラメータを入力していることを確認します。

   たとえば:

```
BEGIN
 DBMS_TSDP_MANAGE.ADD_SENSITIVE_COLUMN(
 schema_name         => 'OE',
 table_name          => 'CUST_CC',
 column_name         => 'CREDIT_CARD',
 sensitive_type      => 'credit_card_num_type',
 user_comment        => 'Sensitive column addition of credit_card_num_type');
END;
/
```

親トピック: 透過的機密データ保護ポリシーの作成

15.6.3 ステップ3: ADMからデータベースへの機密列リストのインポート

次に、ADMからデータベースに機密列リストをインポートします。

- アプリケーション・データ・モデルを使用して機密列のリストを作成した場合、DBMS_TSDP_MANAGE.IMPORT_DISCOVERY_RESULTプロシージャを実行して、このリストをデータベースにイン

ポートします。

DBMS_TSDP_MANAGE.ADD_SENSITIVE_COLUMNプロシージャを使用してこれらの列を識別した場合、このステップを回避できます。

たとえば、Cloud Controlアプリケーション・データ・モデルを現在のデータベースにインポートするには:

```
BEGIN
 DBMS_TSDP_MANAGE.IMPORT_DISCOVERY_RESULT (
 discovery_result      => xml_adm_result,
 discovery_source      => 'ADM_Demo');
END;
/
```

この例では、次のようになります。

- discovery_resultは、機密列および関連付けられている機密タイプのリストを表します。このリストはXML形式です。
- discover_sourceは、discovery_result設定で参照された機密列のリストを含むアプリケーション・データ・モデルの名前を表します。Enterprise Manager Cloud Controlのデータ検出およびモデリング・ページからアプリケーション・データ・モデルのリストを確認できます。(このページにアクセスするには、「エンタープライズ」メニューから、「クオリティ管理」、「データ検出およびモデリング」の順に選択します。「機密列」タブで、機密列および関連付けられているタイプのリストを確認できます。)

親トピック: 透過的機密データ保護ポリシーの作成

15.6.4 ステップ4: 透過的機密データ保護ポリシーの作成

機密列のリストを作成し、このリストをデータベースにインポートしたら、透過的機密データ保護ポリシーを作成します。

- 透過的機密データ保護ポリシーの作成について
  DBMS_TSDP_PROTECT.ADD_POLICYプロシージャで、透過的機密データ保護ポリシーを作成します。
- 透過的機密データ保護ポリシーの作成
  部分的な数値データ型ベースのデータ・リダクション・ポリシーを使用する透過的機密データ保護ポリシーを作成できます。
- Oracle Data Redactionまたは仮想プライベート・データベース機能オプションの設定
  TSDP機能オプションは、透過的機密データ保護ポリシーに使用するOracle Data Redactionまたは仮想プライベート・データベース設定を示します。
- 透過的機密データ保護ポリシーの条件の設定
  オプションで、透過的機密データ保護ポリシーの条件を指定できます。
- DBMS_TSDP_PROTECT.ADD_POLICYプロシージャの指定
  DBMS_TSDP_PROTECT.ADD_POLICYプロシージャでTSDPポリシーに名前を付与し、FEATURE_OPTIONSおよびPOLICY_CONDITIONSの設定を実行します。

親トピック: 透過的機密データ保護ポリシーの作成

15.6.4.1 透過的機密データ保護ポリシーの作成について

DBMS_TSDP_PROTECT.ADD_POLICYプロシージャで、透過的機密データ保護ポリシーを作成します。

機密列の特定後、アプリケーション・データ・モデルを使用して機密列のリストを作成し、このリストをデータベースにインポートしたら、透過的機密データ保護ポリシーを作成します。透過的機密データ保護ポリシーを作成するには、使用する仮想プライベー

ト・データベースまたはOracle Data Redaction設定にポリシーを構成し、これらの設定を`DBMS_TSDP_PROTECT.ADD_POLICY`で定義された透過的機密データ保護ポリシーに適用します。

次の構成要素を持つ無名ブロックを定義して、ポリシーを作成できます。

- ポリシーにOracle Data Redactionを使用している場合、部分的なデータ・リダクションなど、使用するデータ・リダクションのタイプの指定
- ポリシーにOracle Virtual Private Databaseを使用している場合、使用するVPD設定の指定
- ポリシーを有効にする場合にテストする条件。たとえば、ポリシーを有効にする前に満たす必要がある列のデータ型など。
- `DBMS_TSDP_PROTECT.ADD_POLICY`プロシージャを使用して、これらの構成要素を結合する名前付きの透過的機密データ保護ポリシー

機密タイプが作成された後、SYSスキーマに配置されます。

関連トピック

- チュートリアル: 仮想プライベート・データベース保護を使用するTSDPポリシーの作成

親トピック: ステップ4: 透過的機密データ保護ポリシーの作成

15.6.4.2 透過的機密データ保護ポリシーの作成

部分的な数値データ型ベースのデータ・リダクション・ポリシーを使用する透過的機密データ保護ポリシーを作成できます。

例15-1に、このタイプのポリシーを作成する方法を示します。

- ポリシーを作成するには、例15-1に示すように、`DBMS_TSDP_PROTECT.ADD_POLICY`プロシージャを使用します。

例15-1 透過的機密データ保護ポリシーの作成

```
DECLARE
  redact_feature_options DBMS_TSDP_PROTECT.FEATURE_OPTIONS;
  policy_conditions DBMS_TSDP_PROTECT.POLICY_CONDITIONS;
BEGIN
  redact_feature_options ('expression') :=
   'SYS_CONTEXT(''USERENV'',''SESSION_USER'') =''APPUSER''';
  redact_feature_options ('function_type') := 'DBMS_REDACT.PARTIAL';
  redact_feature_options ('function_parameters') := '0,1,6';
  policy_conditions(DBMS_TSDP_PROTECT.DATATYPE) := 'NUMBER';
  policy_conditions(DBMS_TSDP_PROTECT.LENGTH) := '16';
 DBMS_TSDP_PROTECT.ADD_POLICY ('redact_partial_cc',
  DBMS_TSDP_PROTECT.REDACT,redact_feature_options,
   policy_conditions);
END;
/
```

この例では、次のようになります。

- `redact_feature_options DBMS_TSDP_PROTECT.FEATURE_OPTIONS`は、`FEATURE_OPTIONS`データ型を使用する変数`redact_feature_options`を作成します。詳細は、Oracle Data Redactionまたは仮想プライベート・データベース機能オプションの設定を参照してください。
- `policy_conditions DBMS_TSDP_PROTECT.POLICY_CONDITIONS`は、`POLICY_CONDITIONS`データ型を使用する変数`policy_conditions`を作成します。詳細は、透過的機密データ保護ポリシーの条件の設定を参照してください。
- `redact_feature_options`の3行目は、データ・リダクション・ポリシー設定を`redact_feature_option`変

数に書き込みます。この例では、データ・リダクション・ポリシーをユーザーAPPUSERに適用して、数値データ型の部分的なデータ・リダクションとしてポリシーを定義します。この場合のfunction_parametersパラメータの使用方法の詳細は、『Oracle Database Advanced Securityガイド』を参照してください。

- policy_conditionsの2行目は、TSDPポリシー条件を保護されたNUMBERデータ型の列のpolicy_conditions変数(つまり、データ型および長さ)に書き込みます。
- DBMS_TSDP_PROTECT.ADD_POLICYは、redact_partial_cc TSDPポリシーを作成するDBMS_TSDP_PROTECT.ADD_POLICYプロシージャを実行します。詳細は、DBMS_TSDP_PROTECT.ADD_POLICYプロシージャの指定を参照してください。

VPD用の類似したポリシーの例を確認する場合は、ステップ4: 透過的機密データ保護ポリシーの作成および有効化を参照してください。

親トピック: ステップ4: 透過的機密データ保護ポリシーの作成

15.6.4.3 Oracle Data Redactionまたは仮想プライベート・データベース機能オプションの設定

TSDP機能オプションは、透過的機密データ保護ポリシーに使用するOracle Data Redactionまたは仮想プライベート・データベース設定を示します。

- Data Redactionの場合、redact_feature_options変数名を使用して機能オプションを定義し、型にはデータ型VARCHAR2(TSDP_PARAM_MAX)の結合配列であるDBMS_TSDP_PROTECT.FEATURE_OPTIONS型を使用する必要があります。DBMS_REDACT.ADD_POLICYパラメータに対応するパラメータと値のペアでこれらのオプションを初期化します。

たとえば、部分的なデータ・リダクションを使用するTSDPポリシーを指定する場合、例15-1に次のパラメータと値のペアを示します。

```
redact_feature_options ('function_type')       := 'DBMS_REDACT.PARTIAL';
```

保護された列の数値データ型を使用する部分的なデータ・リダクション・ポリシーの場合、例15-1では、次の追加のパラメータと値のペアを指定しています。

```
redact_feature_options ('expression')          := 'expression';
redact_feature_options ('function_parameters') := 'values';
```

同様に、仮想プライベート・データベースの場合、vpd_feature_options変数を使用して、VPD機能オプションを定義します。たとえば:

```
vpd_feature_options ('statement_types')   := 'SELECT, INSERT, UPDATE, DELETE';
```

関連項目:

- データ・リダクション・ポリシー作成パラメータの詳細は、『Oracle Database Advanced Securityガイド』を参照してください。
- 使用できるVPDパラメータの詳細は、「TSDPポリシーに使用されるDBMS_RLS.ADD_POLICYパラメータ」を参照してください。

親トピック: ステップ4: 透過的機密データ保護ポリシーの作成

15.6.4.4 透過的機密データ保護ポリシーの条件の設定

オプションで、透過的機密データ保護ポリシーの条件を指定できます。

ただし、条件を省略する場合、DECLARE変数の次の行を引き続き含む必要があります。(この場合、policy_conditionsのデフォルト値は、空の結合配列です。)

```
policy_conditions SYS.DBMS_TSDP_PROTECT.POLICY_CONDITIONS;
```

- 条件を定義するには、policy_conditions変数名を使用し、型にはデータ型VARCHAR2(TSDP_PARAM_MAX)の結合配列であるDBMS_TSDP_PROTECT.POLICY_CONDITIONS型を使用します。2つの条件が単一のターゲット機密列で満たされていることを確認します。ターゲット列のプロパティは、列に適用する対応するDBMS_TSDP_PROTECT.FEATURE_OPTIONS設定のすべての条件プロパティを満たす必要があります

例15-1は、次のポリシー条件を示しています。

```
policy_conditions(DBMS_TSDP_PROTECT.DATATYPE) := 'NUMBER';
policy_conditions(DBMS_TSDP_PROTECT.LENGTH)   := '16';
```

オプションで、POLICY_CONDITIONS設定の次の1つ以上のキーを指定できます。

- DBMS_TSDP_PROTECT.DATATYPEでは、データ型を指定できます。
- DBMS_TSDP_PROTECT.LENGTHでは、DBMS_TSDP_PROTECT.DATATYPEキーのデータ型の長さを指定できます。
- DBMS_TSDP_PROTECT.PARENT_SCHEMAでは、ポリシーを特定のスキーマに制限できます。この設定を省略すると、ポリシーはデータベースのすべてのスキーマに適用されます。
- DBMS_TSDP_PROTECT.PARENT_TABLEでは、ポリシーをDBMS_TSDP_PROTECT.PARENT_SCHEMAキーで指定された表に制限できます。この設定を省略すると、ポリシーは指定されたスキーマ内のすべての表に適用されます。

親トピック: ステップ4: 透過的機密データ保護ポリシーの作成

15.6.4.5 DBMS_TSDP_PROTECT.ADD_POLICYプロシージャの指定

DBMS_TSDP_PROTECT.ADD_POLICYプロシージャでTSDPポリシーに名前を付与し、FEATURE_OPTIONSおよびPOLICY_CONDITIONSの設定を実行します。

ポリシーでは、redact_feature_optionsとpolicy_conditions設定が連携します。ポリシーがターゲット・オブジェクトで有効な場合(DBMS_TSDP_PROTECT.ENABLE_PROTECTION*プロシージャを使用)、redact_feature_options設定は、対応するpolicy_condition設定を満たしている場合のみ適用されます。次のパラメータを入力します。

- 透過的機密データ保護ポリシーの名前を付け、必要な設定を実行するプロシージャを指定するには、次のパラメータを含めます。
  - policy_nameは、TSDPポリシーの名前を作成します。入力する名前は、作成時に使用された大/小文字の区別が使用され、データベースに格納されます。たとえば、redact_partial_ccを入力した場合、データベースはredact_partial_ccではなくredact_partial_ccとして格納します。
  - security_featureは、TSDPポリシーが使用するセキュリティ機能を表します。DBMS_TSDP_PROTECT.REDACTを入力して、Oracle Data Redactionを指定します。
  - policy_enable_optionsは、DBMS_TSDP_PROTECT.FEATURE_OPTIONS型に定義された変

数を表します。

- policy_apply_conditionは、DBMS_TSDP_PROTECT.POLICY_CONDITIONS型に定義された変数を表します。

例15-1は、次のポリシー・セットを示しています。

```
DBMS_TSDP_PROTECT.ADD_POLICY('redact_partial_cc', DBMS_TSDP_PROTECT.REDACT,
redact_feature_options, policy_conditions);
```

親トピック: ステップ4: 透過的機密データ保護ポリシーの作成

15.6.5 ステップ5: ポリシーと機密タイプの関連付け

DBMS_TSDP_PROTECT.ASSOCIATE_POLICYプロシージャで、TSDPポリシーと機密タイプを関連付けます。

1. 使用する機密タイプを確認します。

   たとえば、すべての機密タイプのリストを確認するには:

   ```
   SELECT NAME FROM DBA_SENSITIVE_COLUMN_TYPES ORDER BY NAME;
   NAME
   --------------------
   credit_card_num_type
   ```

2. DBMS_TSDP_PROTECT.ASSOCIATE_POLICYプロシージャを実行して、ポリシーと機密列タイプを関連付けます。

   たとえば:

   ```
   BEGIN
    DBMS_TSDP_PROTECT.ASSOCIATE_POLICY(
    policy_name         => 'redact_partial_cc',
    sensitive_type      => 'credit_card_num_type',
    associate           => true);
   END;
   /
   ```

   次の問合せは、credit_card_num_typeとredact_partial_ccポリシーが関連付けられたことを示します。

   ```
   SELECT POLICY_NAME, SENSITIVE_TYPE FROM DBA_TSDP_POLICY_TYPE ORDER BY
   SENSITIVE_TYPE;
   POLICY_NAME       SENSITIVE_TYPE
   ----------------- --------------------
   redact_partial_cc  credit_card_num_type
   ```

親トピック: 透過的機密データ保護ポリシーの作成

15.6.6 ステップ6: 透過的機密データ保護ポリシーの有効化

保護されるソースの現在のデータベース、特定の表の列または特定の列タイプに対してTSDPポリシーを有効にすることができます。

- 保護されたソースの現在のデータベースの保護の有効化
  保護されるソースの現在のデータベースに対して透過的機密データ保護を有効にできます。
- 特定の表の列の保護の有効化
  表の特定の列に対して透過的機密データ保護を有効にできます。
- 特定の列タイプの保護の有効化
  VARCHAR2データ型を使用するすべての列など、特定の列タイプに対して透過的機密データ保護を有効にできます。

親トピック: 透過的機密データ保護ポリシーの作成

15.6.6.1 保護されたソースの現在のデータベースの保護の有効化

保護されるソースの現在のデータベースに対して透過的機密データ保護を有効にすることができます。

保護を無効化する必要がある場合、DBMS_TSDP_PROTECT.DISABLE_PROTECTION_SOURCEプロシージャを実行します。

- このタイプの保護を有効にするには、DBMS_TSDP_PROTECT.ENABLE_PROTECTION_SOURCEプロシージャを実行します。

たとえば、orders_dbデータベースの透過的機密データ保護ポリシーを有効化するには。

```
BEGIN
 DBMS_TSDP_PROTECT.ENABLE_PROTECTION_SOURCE(
  discovery_source       => 'orders_db');
END;
/
```

親トピック: ステップ6: 透過的機密データ保護ポリシーの有効化

15.6.6.2 特定の表の列の保護の有効化

表の特定の列に対して透過的機密データ保護を有効にすることができます。

表ごとに1つのポリシーのみ有効化できることに注意してください。保護を無効化する必要がある場合、DBMS_TSDP_PROTECT.DISABLE_PROTECTION_COLUMNプロシージャを実行します。

- このタイプの保護を有効にするには、DBMS_TSDP_PROTECT.ENABLE_PROTECTION_COLUMNプロシージャを実行します。

たとえば、特定の表の列の透過的機密データ保護ポリシーredact_partial_ccを有効化するには:

```
BEGIN
 DBMS_TSDP_PROTECT.ENABLE_PROTECTION_COLUMN(
  schema_name          => 'OE',
  table_name           => 'CUST_CC',
  column_name          => 'CREDIT_CARD',
  policy               => 'redact_partial_cc');
END;
/
```

「ORA-45622: ポリシーの強制中に警告が生成されました」エラーが表示される場合、ポリシーの構成をチェックします。この例では、この列がNUMBERデータ型で長さが16の場合、redact_partial_ccポリシーが列で有効です。OE.CUST_CC.CREDIT_CARD列がredact_partial_ccポリシーに関連付けられていても、この列が条件(データ型および長さ)を満たさない場合はポリシーが無効です。

親トピック: ステップ6: 透過的機密データ保護ポリシーの有効化

15.6.6.3 特定の列タイプの保護の有効化

VARCHAR2データ型を使用するすべての列など、特定の列タイプに対して透過的機密データ保護を有効にすることができます。

保護を無効化する必要がある場合、DBMS_TSDP_PROTECT.DISABLE_PROTECTION_TYPEプロシージャを実行します。

- このタイプの保護を有効にするには、DBMS_TSDP_PROTECT.ENABLE_PROTECTION_TYPEプロシージャを実

行します。

たとえば、credit_card_num_type機密タイプを使用するすべての列の透過的機密データ保護を有効化するには:

```
BEGIN
 DBMS_TSDP_PROTECT.ENABLE_PROTECTION_TYPE(
  sensitive_type           => 'credit_card_num_type');
END;
/
```

親トピック: ステップ6: 透過的機密データ保護ポリシーの有効化

15.6.7 ステップ7: 他のデータベースへのポリシーのエクスポート(オプション)

別のデータベースに対してポリシーをエクスポートまたはインポートできます。

- 別のデータベースに対してTSDPポリシーをエクスポートまたはインポートするには、Oracle Data Pumpを使用して、ポリシーを含むデータベースのフル・エクスポートまたはインポートを実行します。

エクスポートおよびインポート操作が透過的機密データ保護ポリシーだけでなくデータベース全体に適用されるので注意してください。

関連項目:

- Oracle Data Pumpの使用方法の詳細は、『Oracle Databaseユーティリティ』を参照してください。
- Oracle Database Vault環境のOracle Data Pumpの使用の詳細は、『Oracle Database Vault管理者ガイド』を参照してください。

親トピック: 透過的機密データ保護ポリシーの作成

15.7 透過的機密データ保護ポリシーの変更

DBMS_TSDP_PROTECT.ALTER_POLICYプロシージャで、TSDPポリシーを変更できます。

透過的データ保護ポリシーを変更する場合、データ・リダクション設定の変更方法を定義して、これらの変更を透過的機密データ保護ポリシー自体に適用する必要があります。

DBA_TSDP_POLICY_FEATUREデータ・ディクショナリ・ビューを問い合せて、既存のポリシーおよび保護定義のリストを確認できます。

- 透過的機密データ保護ポリシーを変更するには、DBMS_TSDP_PROTECT.ALTER_POLICYプロシージャを使用します。

たとえば、既存の透過的機密データ保護ポリシーを変更するには:

```
DECLARE
  redact_feature_options SYS.DBMS_TSDP_PROTECT.FEATURE_OPTIONS;
  policy_conditions SYS.DBMS_TSDP_PROTECT.POLICY_CONDITIONS;
  BEGIN
  redact_feature_options ('expression') :=
   'SYS_CONTEXT(''USERENV'',''SESSION_ USER'') =''APPUSER''';
  redact_feature_options ('function_type') := 'DBMS_REDACT.PARTIAL';
  redact_feature_options ('function_parameters') := '9,1,6';
  policy_conditions(DBMS_TSDP_PROTECT.DATATYPE) := 'NUMBER';
  policy_conditions(DBMS_TSDP_PROTECT.LENGTH) := '22';
 DBMS_TSDP_PROTECT.ALTER_POLICY ('redact_partial_cc',
  redact_feature_options, policy_conditions);
```

```
END;
/
```

この例では、次のようになります。

- `redact_feature_options SYS.DBMS_TSDP_PROTECT.FEATURE_OPTIONS`は、`FEATURE_OPTIONS`データ型を使用する変数`redact_feature_options`を作成します。
- `policy_conditions SYS.DBMS_TSDP_PROTECT.POLICY_CONDITIONS`は、`POLICY_CONDITIONS`データ型を使用する変数`policy_conditions`を作成します。
- `redact_feature_options ... redact_feature_options`は、データ・リダクション・ポリシー設定を`redact_feature_option`変数に書き込みます。この例では、データ・リダクション・ポリシーをユーザーAPPUSERに適用して、数値データ型の部分的なデータ・リダクションとしてポリシーを定義します。この場合の`function_parameters`パラメータの使用方法の詳細は、『Oracle Database Advanced Securityガイド』を参照してください。
- `policy_conditions ... policy_conditions`は、TSDPポリシー条件を保護されたNUMBERデータ型の列の`policy_conditions`変数(つまり、データ型および長さ)に書き込みます。
- `DBMS_TSDP_PROTECT.ALTER_POLICY ...`は、`redact_partial_cc` TSDPポリシーを変更して`redact_feature_options`および`policy_conditions`変数に設定された定義を使用する`DBMS_TSDP_PROTECT.ALTER_POLICY`プロシージャを実行します。

親トピック: 透過的機密データ保護の使用

15.8 透過的機密データ保護ポリシーの無効化

`DBMS_TSDP_PROTECT.DISABLE_PROTECTION_COLUMN`プロシージャは1つまたはすべてのTSDPポリシーを無効にします。

1. `DBA_TSDP_POLICY_PROTECTION`データ・ディクショナリ・ビューを問い合せて、保護された列および関連付けられた透過的機密データ保護ポリシーを確認します。

   たとえば:

   ```
   SELECT COLUMN_NAME, TSDP_POLICY FROM DBA_TSDP_POLICY_PROTECTION WHERE
   TABLE_NAME = 'CUST_CC';
   COLUMN_NAME    TSDP_POLICY
   ------------   ------------------
   CREDIT_CARD    redact_partial_cc
   ```

2. `DBMS_TSDP_PROTECT.DISABLE_PROTECTION_COLUMN`プロシージャを実行します。

   たとえば、CUST_CC表の`CREDIT_CARD`列の`redact_partial_cc`ポリシーを無効化するには:

   ```
   BEGIN
    DBMS_TSDP_PROTECT.DISABLE_PROTECTION_COLUMN(
      schema_name          => 'OE',
      table_name           => 'CUST_CC',
      column_name          => 'CREDIT_CARD',
      policy               => 'redact_partial_cc');
   END;
   /
   ```

   このプロシージャの%ワイルドカードを使用して、複数の項目を指定できます。たとえば、`CREDIT`で始まる列の保護を無効化するには、次のように入力できます。

```
BEGIN
 DBMS_TSDP_PROTECT.DISABLE_PROTECTION_COLUMN(
  schema_name          => 'OE',
  table_name           => 'CUST_CC',
  column_name          => 'CREDIT%',
  policy               => 'redact_partial_cc');
END;
/
```

表のすべての透過的機密データ保護ポリシーを無効化するには、policyパラメータを省略できます。たとえば:

```
BEGIN
 DBMS_TSDP_PROTECT.DISABLE_PROTECTION_COLUMN(
  schema_name          => 'OE',
  table_name           => 'CUST_CC',
  column_name          => '%');
END;
/
```

親トピック: 透過的機密データ保護の使用

15.9 透過的機密データ保護ポリシーの削除

TSDPポリシー全体を削除するか、またはポリシーから条件有効オプションの組合せを削除できます。

ポリシーが1つの条件有効オプションの組合せのみ持つ場合、Oracle Databaseはポリシー全体を削除します。削除する前にポリシーを無効化する必要はありませんが、関連付けられている機密列、機密タイプの順に削除する必要があります。

1. DBA_TSDP_POLICY_FEATUREデータ・ディクショナリ・ビューのPOLICY_NAME列を問い合せて、削除するポリシーを確認します。

   ```
   SELECT POLICY_NAME FROM DBA_TSDP_POLICY_FEATURE;
   POLICY_NAME
   -----------------
   redact_partial_cc
   ```

   透過的機密データ保護データ・ディクショナリ・ビューを問い合せるには、SELECT_CATALOG_ROLEロールが付与されている必要があります。

2. このポリシーに関連付けられている機密列を確認します。

   たとえば:

   ```
   SELECT COLUMN_NAME FROM DBA_TSDP_POLICY_PROTECTION WHERE TSDP_POLICY =
   'redact_partial_cc';
   COLUMN_NAME
   -----------------
   CREDIT_CARD
   ```

3. この機密列を削除します。

   たとえば:

   ```
   BEGIN
    DBMS_TSDP_MANAGE.DROP_SENSITIVE_COLUMN (
      schema_name         => 'OE',
      table_name          => 'CUST_CC',
      column_name         => 'CREDIT_CARD');
   END;
   /
   ```

4. このポリシーに関連付けられている機密タイプを確認します。

   たとえば:

```
SELECT SENSITIVE_TYPE FROM DBA_TSDP_POLICY_TYPE WHERE POLICY_NAME =
'redact_partial_cc';
SENSITIVE_TYPE
--------------------
credit_card_num_type
```

5. この機密タイプを削除します。

   たとえば:

```
BEGIN
 DBMS_TSDP_MANAGE.DROP_SENSITIVE_TYPE (   sensitive_type     =>
'credit_card_num_type');END;
/
```

6. DBMS_TSDP_PROTECT.DROP_POLICYプロシージャを実行して、ポリシーを削除します。

   たとえば、ポリシーを完全に削除するには:

```
BEGIN
 DBMS_TSDP_PROTECT.DROP_POLICY(
   policy_name     => 'redact_partial_cc');
END;
/
```

   ポリシーからデフォルトの条件有効オプションの組合せを削除するには:

```
DECLARE
   policy_conditions DBMS_TSDP_PROTECT.POLICY_CONDITIONS;
BEGIN
   DBMS_TSDP_PROTECT.DROP_POLICY ('redact_partial_cc', policy_conditions);
END;
/
```

   特定の条件に基づいてポリシーからデフォルトの条件有効オプションを削除するには:

```
DECLARE
   policy_conditions DBMS_TSDP_PROTECT.POLICY_CONDITIONS;
BEGIN
   policy_conditions (DBMS_TSDP_PROTECT.DATATYPE) := 'NUMBER';
   DBMS_TSDP_PROTECT.DROP_POLICY ('redact_partial_cc', policy_conditions);
END;
/
```

親トピック: 透過的機密データ保護の使用

15.10 事前定義のREDACT_AUDITポリシーを使用したバインド値のマスク

事前定義のREDACT_AUDITポリシーはバインド値をマスクします。バインド値は、イベントの設定時にトレース・ファイルに表示されます。

- REDACT_AUDITポリシーについて
  事前定義のREDACT_AUDIT透過的機密データ保護ポリシーはバインド値をマスクします。
- 機密列と関連付けられた変数

バインド変数は、条件やSELECTアイテム、INSERTまたはUPDATE操作のある機密列の使用に影響します。

- ビューでの機密列のバインド変数の動作
  ビューの列が機密列を参照する場合、ビューの問合せに表示されるバインド変数が機密とみなされます。
- REDACT_AUDITポリシーの無効化
  デフォルトでは、REDACT_AUDITポリシーがすべての機密列に有効です。
- REDACT_AUDITポリシーの有効化
  特定の機密列、またはデータベース内のすべての列に対してREDACT_AUDITポリシーを有効化できます。

親トピック: 透過的機密データ保護の使用

15.10.1 REDACT_AUDITポリシーについて

事前定義のREDACT_AUDIT透過的機密データ保護ポリシーはバインド値をマスクします。

SQL文で使用されるバインド変数のバインド値は、監査の構成時に監査レコードに表示できます。同様に、バインド値は、適切なイベントの設定時にトレース・ファイルに表示できます。V$SQL_BIND_DATA動的ビューを問い合せる場合、バインド値も表示できます。

REDACT_AUDIT透過的機密データ保護ポリシーは、監査レコード、トレース・ファイルおよびV$SQL_BIND_DATAビューの問合せでアスタリスク(*)としてデータを表示します。デフォルトでは、REDACT_AUDITポリシーがデータベースの機密タイプに関連付けられます。機密として列を識別する場合、デフォルトではREDACT_AUDITポリシーが有効です。

REDACT_AUDITポリシーを無効化および有効化できますが、変更または削除できません。

親トピック: 事前定義のREDACT_AUDITポリシーを使用したバインド値のマスク

15.10.2 機密列と関連付けられた変数

バインド変数は、条件やSELECTアイテム、INSERTまたはUPDATE操作のある機密列の使用に影響します。

- 機密列に関連付けられた変数について
  TSDPポリシーで変数を機密列に関連付けることができます。
- 条件式のバインド変数および機密列
  WHERE句を含むSQL問合せに機密列を含めることができます。
- 同じSELECT項目に表示されるバインド変数および機密列
  SELECT項目の列が機密である場合、SELECT項目のすべてのバインドが機密とみなされます。
- INSERTまたはUPDATE操作の機密列に割り当てられる式のバインド変数
  複数のバインド変数を1つのINSERTまたはUPDATE文の異なる列に割り当てることができます。

親トピック: 事前定義のREDACT_AUDITポリシーを使用したバインド値のマスク

15.10.2.1 機密列に関連付けられた変数について

TSDPポリシーで変数を機密列に関連付けることができます。

バインド変数が機密列と同じ比較条件にある場合、機密列とともにSELECT文にある場合、または機密列を含むINSERTまたはUPDATE操作にある場合、そのバインド変数は機密とみなされるか、または機密列と関連付けられます。

親トピック: 機密列に関連付けられている変数

15.10.2.2 条件式のバインド変数および機密列

WHERE句を含むSQL問合せに機密列を含めることができます。

WHERE句を含むSQL問合せには、比較演算子(=、IS、IS NOT、LIKE、BETWEEN、INなど)とともに使用したり、副問合せで使用する機密列およびバインド変数を含めることができます

次の比較問合せでは、VAR1および機密列SALARYが比較条件>を使用して比較される式に表示されているため、VAR1のバインド値がマスクされます。

```
SELECT EMPLOYEE_ID FROM HR.EMPLOYEES WHERE SALARY > :VAR1;
```

次の問合せでは、VAR1、VAR2および機密列SALARYが比較等価条件=を使用する式に表示されているため、VAR1およびVAR2のバインド値がマスクされます。

```
SELECT EMPLOYEE_ID FROM HR.EMPLOYEES WHERE SALARY + :VAR1 = TO_NUMBER(:VAR2,
'9G999D99');
```

浮動小数点条件の場合、機密列およびバインド変数が評価される式に表示されます。次の例では、VAR1および機密列SALARYがIS NOT NAN条件の式に表示されているため、VAR1のバインド値がマスクされます。

```
SELECT COUNT( ) FROM HR.EMPLOYEES WHERE (SALARY * :VAR1) IS NOT NAN;
```

パターン一致条件では、機密列およびバインド変数が引数として表示されます。次の例では、VAR1および機密列LAST_NAMEがLIKE条件の引数であるため、VAR1のバインド値がマスクされます。

```
SELECT LAST_NAME FROM HR.EMPLOYEES WHERE LAST_NAME LIKE :VAR1;
```

BETWEEN条件の場合、機密列およびバインド変数が引数である式に表示されます。次の例では、VAR1、VAR2およびSALARYがBETWEEN条件の引数である式に表示されているため、VAR1およびVAR2のバインド値がマスクされます。

```
SELECT EMPLOYEE_ID FROM HR.EMPLOYEES WHERE SALARY BETWEEN :VAR1 AND :VAR2;
```

次の例では、機密列およびバインド変数がIN条件の引数です。ここでは、VAR1、VAR2および機密列SALARYがIN条件の引数として表示されているため、VAR1およびVAR2のバインド値がマスクされます。

```
SELECT COUNT( ) FROM HR.EMPLOYEES WHERE SALARY IN ( :VAR1, :VAR2);
```

条件に引数としてネストされた副問合せがある場合、ネストされた副問合せに表示されるバインド変数および機密列は条件に関連付けられているとみなされません。次の問合せでは、機密列SALARY列および副問合せは、次より大きい条件>の式です。

```
SELECT EMPLOYEE_ID FROM HR.EMPLOYEES WHERE SALARY > (SELECT SALARY FROM HR.EMPLOYEES
WHERE MANAGER_ID = :VAR1);
```

ただし、変数VAR1およびMANAGER_IDが条件=を使用して比較される式に表示されるため、変数VAR1が列MANAGER_IDに関連付けられます。MANAGER_IDが機密列ではないため、変数VAR1は機密とみなされません。変数VAR1は、機密列SALARYと関連付けられているとみなされません。

論理条件、モデル条件、多重集合条件、XML条件、複合条件、IS OF型条件およびEXISTS条件の場合、バインド変数および機密列が相互に関連付けられることはありません。これは、これらの条件の構造または性質のためです。

親トピック: 機密列に関連付けられている変数

15.10.2.3 同じSELECT項目に表示されるバインド変数および機密列

SELECT項目の列が機密である場合、SELECT項目のすべてのバインドが機密とみなされます。

たとえば、HR.EMPLOYEES.SALARYおよびHR.EMPLOYEES.COMMISSION_PCTが機密列であるとします。次の問合せでは、機密列SALARYと同じSELECT項目に表示されているため、バインド変数VAR1が機密とみなされ、バインド値がマスクされます。

```
SELECT (SALARY * :VAR1) AS BONUS AS FROM HR.EMPLOYEES WHERE EMPLOYEE_ID = :VAR2;
```

次の例では、SALARYと同じSELECT項目に表示されているため、バインド変数VAR1が機密とみなされます。機密列COMMISSION_PCTと同じSELECT項目に表示されているため、VAR2が機密とみなされます。

```
SELECT (SALARY * :VAR1), (COMMISSION_PCT * :VAR2), (EMPNO + :VAR3) AS BONUS AS FROM
PAYROLL.ACCOUNT;
```

親トピック: 機密列に関連付けられている変数

15.10.2.4 INSERTまたはUPDATE操作の機密列に割り当てられる式のバインド変数

複数のバインド変数を1つのINSERTまたはUPDATE文の異なる列に割り当てることができます。

次のINSERT文を考えてみます。

```
INSERT INTO PAYROLL.ACCOUNT (ACCOUNT_NUM, SALARY) VALUES (:VAR1 * :VAR2 , :VAR3);
```

このINSERT文では、次が発生します。

- バインド変数VAR1およびVAR2が式(:VAR1 * :VAR2)に表示され、機密列ACCOUNT_NUMに割り当てられます。
- バインド変数VAR3が機密列SALARYに割り当てられます。

次のUPDATE文を考えてみます。

```
UPDATE PAYROLL.ACCOUNT SET ACCOUNT_NUM = :VAR1, SALARY = :VAR2;
```

このUPDATE文では、次が発生します。

- バインド変数VAR1が機密列ACCOUNT_NUMに割り当てられます。
- バインド変数VAR2が機密列SALARYに割り当てられます。

親トピック: 機密列に関連付けられている変数

15.10.3 ビューでの機密列のバインド変数の動作

ビューの列が機密列を参照する場合、ビューの問合せに表示されるバインド変数が機密とみなされます。

たとえば、HR.EMPLOYEES表のSALARY列を機密と識別するとします。次に、ビューEMPLOYEES_VIEWを次のように作成します。

```
CREATE OR REPLACE VIEW HR.EMPLOYEES_VIEW AS SELECT * FROM HR.EMPLOYEES;
```

ユーザーがSQL文でこのビューからSALARY列を参照する場合、SALARY列に関連付けられているバインド変数が機密とみなされ、バインド値がマスクされます。

```
SELECT EMPLOYEE_ID FROM HR.EMPLOYEES_VIEW WHERE SALARY = :VAR1;
```

この場合、機密列HR.EMPLOYEES.SALARYを参照するHR.EMPLOYEES_VIEW.SALARY列に関連付けられているため、バインド変数VAR1がマスクされます。

親トピック: 事前定義のREDACT_AUDITポリシーを使用したバインド値のマスク

15.10.4 REDACT_AUDITポリシーの無効化

デフォルトでは、REDACT_AUDITポリシーがすべての機密列に有効です。

特定の機密列またはすべての機密列に対して無効化できますが、必要に応じて再度有効化できます。REDACT_AUDITポリシーを変更または削除できないことに注意してください。

- REDACT_AUDITポリシーを無効化するには、DBMS_TSDP_PROTECT.DISABLE_PROTECTION_COLUMNプロシージャを使用します。

たとえば、HR.EMPLOYEESのSALARY列のREDACT_AUDITポリシーを無効化するには:

```
BEGIN
 DBMS_TSDP_PROTECT.DISABLE_PROTECTION_COLUMN(
  schema_name          => 'HR',
  table_name           => 'EMPLOYEES',
  column_name          => 'SALARY',
  policy               => 'REDACT_AUDIT');
END;
/
```

次の例は、現在のデータベースのすべての機密列のREDACT_AUDITポリシーを無効化する方法を示します。

```
BEGIN
 DBMS_TSDP_PROTECT.DISABLE_PROTECTION_COLUMN(
  policy               => 'REDACT_AUDIT');
END;
/
```

親トピック: 事前定義のREDACT_AUDITポリシーを使用したバインド値のマスク

15.10.5 REDACT_AUDITポリシーの有効化

特定の機密列、またはデータベース内のすべての列に対してREDACT_AUDITポリシーを有効化できます。

- REDACT_AUDITポリシーを有効化するには、DBMS_TSDP_PROTECT.DISABLE_PROTECTION_COLUMNプロシージャを使用します。

たとえば、HR.EMPLOYEESのSALARY列のREDACT_AUDITポリシーを再有効化するには:

```
BEGIN
 DBMS_TSDP_PROTECT.ENABLE_PROTECTION_COLUMN(
  schema_name          => 'HR',
  table_name           => 'EMPLOYEES',
  column_name          => 'SALARY',
  policy               => 'REDACT_AUDIT');
END;
/
```

次の例は、現在のデータベースのすべての機密列のREDACT_AUDITポリシーを有効化する方法を示します。

```
BEGIN
 DBMS_TSDP_PROTECT.ENSABLE_PROTECTION_COLUMN(
  policy               => 'REDACT_AUDIT');
END;
/
```

親トピック: 事前定義のREDACT_AUDITポリシーを使用したバインド値のマスク

15.11 データ・リダクションでの透過的機密データ保護ポリシー

Oracle Data Redaction機能を透過的機密データ保護ポリシーで使用できます。

Data Redactionの関数タイプ、関数パラメータおよび式をTSDPポリシーの定義に使用できます。たとえば、TSDPポリシーの

有効化を設定して、FULLまたはPARTIALデータ・リダクションを使用できます。この章では、TSDPポリシーを管理する例としてデータ・リダクションを使用します。

関連項目:

- データ・リダクション関数タイプを使用するTSDPポリシーの作成方法の例は、透過的機密データ保護ポリシーの作成を参照してください
- Oracle Data Redactionの詳細は、『Oracle Database Advanced Securityガイド』を参照してください。

親トピック: 透過的機密データ保護の使用

15.12 Oracle VPDポリシーでの透過的機密データ保護ポリシーの使用

TSDPとOracle Virtual Private Databaseの保護を1つのポリシーに統合できます。

- TSDPポリシーとOracle Virtual Private Databaseポリシーの併用について
  Oracle Virtual Private Database保護を透過的機密データ保護ポリシーに組み込むには、DBMS_TSDP_PROTECTおよびDBMS_RLSパッケージを使用する必要があります。
- TSDPポリシーに使用されるDBMS_RLS.ADD_POLICYパラメータ
  Oracle Databaseには、TSDPポリシーの動作を調整するための一連のパラメータが用意されています。
- チュートリアル: 仮想プライベート・データベース保護を使用するTSDPポリシーの作成
  このチュートリアルでは、Oracle Virtual Private Database保護を透過的機密データ保護ポリシーに組み込む方法を示します。

親トピック: 透過的機密データ保護の使用

15.12.1 TSDPポリシーとOracle Virtual Private Databaseポリシーの併用について

Oracle Virtual Private Database保護を透過的機密データ保護ポリシーに組み込むには、DBMS_TSDP_PROTECTおよびDBMS_RLSパッケージを使用する必要があります。

この機能の手順は次のとおりです。

1. 適切な述語を使用して、VPDポリシー関数を作成します。後で、TSDPポリシーを作成する場合、DBMS_TSDP_PROTECT.ADD_POLICYプロシージャのfeature_optionsパラメータにDBMS_RLS.ADD_POLICYプロシージャのpolicy_function設定を使用して、このVPDポリシー関数を参照します。
2. VPDポリシー関数と似ている必要なVPD設定を使用して、TSDPポリシーを作成します。

   TSDPポリシーは、DBMS_RLS.ADD_POLICYプロシージャのパラメータ設定を使用して、VPD保護を提供します。表15-1に、これらのパラメータを示します。DBMS_RLS.ADD_GROUPED_POLICYポリシーのパラメータはサポートされていないので注意してください。
3. DBMS_TSDP_PROTECT.ASSOCIATE_POLICYプロシージャを使用して、TSDPポリシーを必要な機密タイプと関連付けます。
4. 次に、DBMS_TSDP_PROTECT.ENABLE_PROTECTION_*プロシージャのいずれかを使用して、TSDP保護を有効化します。

5. TSDPポリシーを有効化します。この時点で、Oracle Databaseは、ステップ1で作成した関数を使用する内部VPDポリシーを作成します。

   内部ポリシーの名前は、ORA$VPDで始まり、識別子と続きます(たとえば、ORA$VPD_6J6L3RSJSN2VAN0XFなど)。このポリシーは、DBA_POLICIESデータ・ディクショナリ・ビューのPOLICY_NAME列に問い合せて確認できます。

6. ユーザーが表を問い合せる場合、列の出力は、現在設定されているVPD保護およびTSDPポリシーに基づきます。

7. この列のTSDPポリシーを無効化するまで、これらの保護は残ります。この時点で、不要になるため、Oracle Databaseが内部VPDポリシーを自動的に削除します。TSDPポリシーを再度有効化すると、内部VPDポリシーが再作成されます。

関連トピック

- 動的なWHERE句を生成する関数

親トピック: Oracle VPDポリシーでの透過的機密データ保護ポリシーの使用

15.12.2 TSDPポリシーに使用されるDBMS_RLS.ADD_POLICYパラメータ

Oracle Databaseには、TSDPポリシーの動作を調整するための一連のパラメータが用意されています。

表15-1に、DBMS_TSDP_PROTECT.ADD_POLICYまたはDBMS_TSDP_PROTECT.ALTER_POLICYプロシージャを使用する場合にFEATURE_OPTIONSパラメータで許可できるDBMS_RLS.ADD_POLICYパラメータを示します。

表15-1 TSDPポリシーに使用されるDBMS_RLS.ADD_POLICYパラメータ

パラメータ	説明	デフォルト
function_schema	ポリシー・ファンクションのスキーマ(NULL の場合は現行のデフォルト・スキーマ)。function_schema が指定されていない場合は、現在のユーザーのスキーマと想定されます。	NULL
policy_function	ポリシーの述語を生成するファンクションの名前。関数がパッケージ内で定義される場合、パッケージの名前(たとえば、my_package.my_function など)を含む必要があります。	NULL
statement_types	ポリシーを適用する文タイプ。INDEX、SELECT、INSERT、UPDATE または DELETE を任意に組み合せることができます。デフォルトでは、これらの中から INDEX 以外のほとんどのタイプが適用されます。	NULL
update_check	文タイプ INSERT または UPDATE に対するオプションの引数。update_check を TRUE に設定すると、Oracle Database は、INSERT または UPDATE 操作後に値に対してポリシーをチェックします。 チェックは、ポリシー定義に含まれるセキュリティ関連列にのみ適用されます。つまり、ポリシーで定義されているセキュリティ関連の列が INSERT または UPDATE 文で追加または更新された場合にのみ、	FALSE

パラメータ	説明	デフォルト
	INSERT 操作または UPDATE 操作が失敗します。	
static_policy	この値が TRUE に設定されている場合は、オブジェクトにアクセスするすべてのユーザー(ただし、SYS または EXEMPT ACCESS POLICY 権限を持つ特権ユーザーを除く)に対して、静的ポリシーのポリシー・ファンクションが同一の述語文字列を生成するものと Oracle Database はみなします。	FALSE
policy_type	デフォルト値の NULL は、policy_type が static_policy パラメータの値によって決定されることを意味します。この中から指定したポリシー・タイプは、static_policy の値よりも優先されます。	NULL
long_predicate	デフォルト値の FALSE は、ポリシー・ファンクションから最大 4000 バイト長の述語が戻されることを意味します。TRUE は、述語テキスト文字列の長さを最大 32K バイトにできることを意味します。long_predicate パラメータを使用する前に存在するポリシーは、32K 制限を保持します。	FALSE
sec_relevant_cols_opt	このパラメータを指定すると、透過的機密データ保護は、DBMS_RLS.ADD_POLICY プロシージャの sec_relevant_cols パラメータに対して保護が有効な機密列を入力します。 sec_relevant_cols_opt の許可された値は次のとおりです。 ● NULL は、sec_relevant_cols で定義されたフィルタ処理を有効にします。 ● DBMS_RLS.ALL_ROWS はすべての行を表示しますが、機密情報の列の値があると、列の値が sec_relevant_cols パラメータでフィルタ処理されるため、NULL が表示されます。	NULL

関連トピック

- データベース表、ビューまたはシノニムへのポリシーの付加

親トピック: Oracle VPDポリシーでの透過的機密データ保護ポリシーの使用

15.12.3 チュートリアル: 仮想プライベート・データベース保護を使用するTSDPポリシーの作成

このチュートリアルでは、Oracle Virtual Private Database保護を透過的機密データ保護ポリシーに組み込む方法を示し

ます。

- ステップ1: hr_appuserユーザー・アカウントの作成
  まず、サンプル・ユーザー・アカウントを作成し、このユーザーに適切な権限を付与する必要があります。
- ステップ2: 機密列の識別
  サンプル・ユーザーtsdp_adminとして、保護する機密データを特定します。
- ステップ3: Oracle Virtual Private Database関数の作成
  TSDPはOracle VPDポリシー関数を、TSDPポリシーが有効になったときに自動作成されるVPDポリシーに関連付けます。
- ステップ4: 透過的機密データ保護ポリシーの作成および有効化
  VPDポリシー関数の作成後、その関数を透過的機密データ保護ポリシーに関連付けることができます。
- ステップ5: 透過的機密データ保護ポリシーのテスト
  これで、透過的機密データ保護ポリシーをテストする準備ができました。
- ステップ6: このチュートリアルのコンポーネントの削除
  このチュートリアルのコンポーネントが不要になった場合、それらを削除できます。

親トピック: Oracle VPDポリシーでの透過的機密データ保護ポリシーの使用

15.12.3.1 ステップ1: hr_appuserユーザー・アカウントの作成

まず、サンプル・ユーザー・アカウントを作成し、このユーザーに適切な権限を付与する必要があります。

1. SYSDBA管理権限を持つユーザーSYSとして、データベース・インスタンスにログインします。

```
sqlplus sys as sysdba
Enter password: password
```

2. マルチテナント環境を使用している場合は、適切なプラガブル・データベース(PDB)に接続します。

   たとえば:

```
CONNECT SYS@hrpdb AS SYSDBA
Enter password: password
```

   使用可能なPDBを検索するには、show pdbsコマンドを実行します。現在のPDBを確認するには、show con_nameコマンドを実行します。

3. 次のユーザー・アカウントを作成します。

```
GRANT CREATE SESSION TO hr_appuser IDENTIFIED BY password;
GRANT CREATE SESSION TO tsdp_admin IDENTIFIED BY password;
```

   「パスワードの最低要件」のガイドラインに従って、passwordを安全なパスワードに置き換えます。

4. ユーザーtsdp_adminに次の権限を付与します。

```
GRANT CREATE PROCEDURE TO tsdp_admin;
GRANT EXECUTE ON DBMS_TSDP_MANAGE TO tsdp_admin;
GRANT EXECUTE ON DBMS_TSDP_PROTECT TO tsdp_admin;
GRANT EXECUTE ON DBMS_RLS to tsdp_admin;
```

5. ユーザーSCOTTとして接続します。

```
CONNECT SCOTT -- Or, CONNECT SCOTT@hrpdb
Enter password: password
```

6. hr_appuserにEMP表のREADオブジェクト権限を付与します。

```
GRANT READ ON EMP TO hr_appuser;
```

親トピック: チュートリアル: 仮想プライベート・データベース保護を使用するTSDPポリシーの作成

15.12.3.2 ステップ2: 機密列の識別

サンプル・ユーザーtsdp_adminとして、保護する機密データを特定します。

1. ユーザーtsdp_adminとして接続します。

```
CONNECT tsdp_admin -- Or, CONNECT tsdb_admin@hrpdb
Enter password: password
```

2. salary_type機密タイプを作成します。

```
BEGIN
 DBMS_TSDP_MANAGE.ADD_SENSITIVE_TYPE (
  sensitive_type  => 'salary_type',
  user_comment    => 'Type for SCOTT.EMP column');
END;
/
```

3. salary_type機密タイプとSCOTT.EMP表を関連付けます。

```
BEGIN
 DBMS_TSDP_MANAGE.ADD_SENSITIVE_COLUMN (
 schema_name         => 'SCOTT',
 table_name          => 'EMP',
 column_name         => 'SAL',
 sensitive_type      => 'salary_type',
 user_comment        => 'Sensitive column addition of SALARY_TYPE');
END;
/
```

親トピック: チュートリアル: 仮想プライベート・データベース保護を使用するTSDPポリシーの作成

15.12.3.3 ステップ3: Oracle Virtual Private Database関数の作成

TSDPはOracle VPDポリシー関数を、TSDPポリシーが有効になったときに自動作成されるVPDポリシーに関連付けます。

- VPDポリシー関数を作成するには、CREATE OR REPLACE FUNCTIONプロシージャを次のように使用します。

```
CREATE OR REPLACE FUNCTION vpd_function (
  v_schema IN VARCHAR2,
  v_objname IN VARCHAR2)
 RETURN VARCHAR2 AS
BEGIN
 RETURN 'SYS_CONTEXT(''USERENV'',''SESSION_USER'') = ''HR_APPUSER''';
END vpd_function;
/
```

親トピック: チュートリアル: 仮想プライベート・データベース保護を使用するTSDPポリシーの作成

15.12.3.4 ステップ4: 透過的機密データ保護ポリシーの作成および有効化

VPDポリシー関数の作成後、その関数を透過的機密データ保護ポリシーに関連付けることができます。

1. 透過的機密データ保護ポリシーを作成します。

```
DECLARE
 vpd_feature_options DBMS_TSDP_PROTECT.FEATURE_OPTIONS;
 policy_conditions DBMS_TSDP_PROTECT.POLICY_CONDITIONS;
BEGIN
```

```
 vpd_feature_options ('policy_function') := 'vpd_function';
 vpd_feature_options ('sec_relevant_cols_opt') := 'DBMS_RLS.ALL_ROWS';
 dbms_tsdp_protect.add_policy('tsdp_vpd', DBMS_TSDP_PROTECT.VPD,
vpd_feature_options, policy_conditions);
END;
/
```

この例では、vpd_feature_optionsパラメータは、DBMS_RLS.ADD_POLICYプロシージャからsec_relevant_cols_optパラメータを参照します。TSDPポリシーを有効にすると、自動的に作成されるVPDポリシーは、DBMS_RLS.ADD_POLICYのsec_relevant_colsパラメータをTSDPによってVPDポリシーが有効になる機密列の名前に設定します。sec_relevant_cols_optパラメータを使用しなかった場合、TSDPはDBMS_RLS.ADD_POLICY sec_relevant_cols_optパラメータを使用していません。

2. tsdp_vpd1 TSDPポリシーとsalary_type機密タイプを関連付けます。

```
BEGIN
 DBMS_TSDP_PROTECT.ASSOCIATE_POLICY(
 policy_name         => 'tsdp_vpd',
 sensitive_type      => 'salary_type',
 associate           => TRUE);
END;
/
```

3. 保護を有効化して、SALARY_TYPEとして識別されるすべての列の仮想プライベート・データベース・ポリシーを規定します。

```
BEGIN
 DBMS_TSDP_PROTECT.ENABLE_PROTECTION_TYPE(
  sensitive_type           => 'salary_type');
END;
/
```

親トピック: チュートリアル: 仮想プライベート・データベース保護を使用するTSDPポリシーの作成

15.12.3.5 ステップ5: 透過的機密データ保護ポリシーのテスト

これで、透過的機密データ保護ポリシーをテストする準備ができました。

1. ユーザーhr_appuserとして接続します。

```
CONNECT hr_appuser -- Or, CONNECT hr_appuser@hrpdb
Enter password: password
```

2. 次のようにSCOTT.EMP表を問い合せます。

```
SELECT SAL, COMM, EMPNO FROM SCOTT.EMP;
```

次のような出力結果が表示されます。

```
    SAL   COMM   EMPNO
--------- ------ ----------
    800          7369
   1600    300   7499
   1250    500   7521
   2975          7566
   1250   1400   7654
   2850          7698
   2450          7782
   3000          7788
   5000          7839
   1500     0    7844
   1100          7876
```

```
    950          7900
   3000          7902
   1300          7934
14 rows selected.
```

vpd_function関数により、ユーザーhr_appuserはEMP表のSAL列の給与を確認できます。

3. ユーザーSCOTTとして接続し、同じ問合せを実行します。

```
CONNECT SCOTT -- Or, CONNECT SCOTT@hrpdb
Enter password: password
SELECT SAL, COMM, EMPNO FROM SCOTT.EMP;
```

次のような出力結果が表示されます。

```
    SAL   COMM   EMPNO
--------- ------ ----------
                 7369
           300   7499
           500   7521
                 7566
          1400   7654
                 7698
                 7782
                 7788
                 7839
             0   7844
                 7876
                 7900
                 7902
                 7934
14 rows selected.
```

SCOTTがEMP表を所有していても、vpd_function関数は、この表のSAL列の給与を表示しません。

親トピック: チュートリアル: 仮想プライベート・データベース保護を使用するTSDPポリシーの作成

15.12.3.6 ステップ6: このチュートリアルのコンポーネントの削除

このチュートリアルのコンポーネントが不要になった場合、それらを削除できます。

1. ユーザーtsdp_adminとして接続します。

```
CONNECT tsdp_admin -- Or, CONNECT tsdp_admin@hrpdb
Enter password: password
```

2. 表示されている順序で次の文を実行します。

```
BEGIN
 DBMS_TSDP_MANAGE.DROP_SENSITIVE_COLUMN (
   schema_name         => 'SCOTT',
   table_name          => 'EMP',
   column_name         => 'SAL');
END;
/

BEGIN
 DBMS_TSDP_MANAGE.DROP_SENSITIVE_TYPE(
 sensitive_type      => 'salary_type');
END;
/
BEGIN
 DBMS_TSDP_PROTECT.DROP_POLICY(
   policy_name      => 'tsdp_vpd');
```

```
END;
/
```

3. ユーザーSYSTEMで接続します。

```
CONNECT SYSTEM -- Or, CONNECT SYSTEM@hrpdb
Enter password: password
```

4. tsdp_adminおよびhr_appuserアカウントを削除します。

```
DROP USER tsdp_admin CASCADE;
DROP USER hr_appuser
```

親トピック: チュートリアル: 仮想プライベート・データベース保護を使用するTSDPポリシーの作成

15.13 統合監査での透過的機密データ保護ポリシーの使用

透過的機密データ保護および統合監査プロシージャを使用すると、これら2つの機能の保護を組み合せることができます。

- 統合監査ポリシーでのTSDPポリシーの使用について
  統合監査を使用してオブジェクトのアクションを監査するように、透過的機密データ保護ポリシーを構成できます。
- TSDPポリシーに使用される統合監査ポリシーの設定
  監査ポリシー設定は、DBMS_TSDP_PROTECT.ADD_POLICYまたはDBMS_TSDP_PROTECT.ALTER_POLICYプロシージャのPOLICY_ENABLE_OPTIONSパラメータで使用できます。

親トピック: 透過的機密データ保護の使用

15.13.1 統合監査ポリシーでのTSDPポリシーの使用について

統合監査を使用してオブジェクトのアクションを監査するように、透過的機密データ保護ポリシーを構成できます。

DBMS_TSDP_PROTECT.ADD_POLICYおよびDBMS_TSDP_PROTECT.ALTER_POLICYプロシージャでは、CREATE AUDIT POLICY、ALTER AUDIT POLICY、AUDIT POLICYおよびCOMMENT SQL文の設定を指定できます。TSDPポリシーを使用すると、INSERTまたはDELETE操作など、ポリシー内のオブジェクト固有のオプションに対するアクション監査オプションを作成できます。システム全体の監査オプションはサポートされていません。したがって、監査対象のオブジェクト型は常にTABLEです。標準のアクション(INSERTなど)のみが許可されます。コンポーネント・アクション(Oracle Label Securityまたは他のOracle Database機能に対するポリシーの作成など)はサポートされていません。

この機能の手順は次のとおりです。

1. 必要な統合監査設定を使用して、TSDPポリシーを作成します。

   TSDPポリシーでは、CREATE AUDIT POLICY、AUDIT POLICYおよびCOMMENT文のパラメータ設定を使用します。これらの設定は、TSDPポリシーに使用される統合監査ポリシーの設定にリストされています。

2. DBMS_TSDP_PROTECT.ASSOCIATE_POLICYプロシージャを使用して、TSDPポリシーを必要な機密タイプと関連付けます。

3. 次に、DBMS_TSDP_PROTECT.ENABLE_PROTECTION_*プロシージャのいずれかを使用して、TSDP保護を有効化します。

4. TSDPポリシーを有効化します。TSDPポリシー有効化プロセスの一環として、Oracle Databaseでは統合監査ポリシーが内部的に作成され、ステップ1でDBMS_TSDP_PROTECT.ADD_POLICYプロシージャに指定したターゲット・

ユーザーおよびロールのリストに対して、そのポリシーが有効化されます。

内部ポリシーの名前は、ORA$UNIFIED_AUDIT_で始まり、その後にランダムな英数文字列が続きます(たとえば、ORA$UNIFIED_AUDIT_6J6L3RSJSN2VAN0XFなど)。このポリシーは、AUDIT_UNIFIED_POLICIESデータ・ディクショナリ・ビューのPOLICY_NAME列に問い合せて確認できます。この内部的に作成されたTSDP統合監査ポリシーが強制されるユーザーとロールの名前を確認するには、AUDIT_UNIFIED_ENABLED_POLICIESビューを問い合せます。

5. TSDPポリシーによって保護されている表に対してユーザーがアクションの実行を試みると、TSDP統合監査ポリシー構成に基づいて、このオブジェクト・アクセスについて統合監査レコードが統合監査証跡に書き込まれます。その後、UNIFIED_AUDIT_TRAILビューを問い合せて、TSDP統合監査ポリシーの実施により作成された統合監査レコードを確認できます。

6. この列のTSDPポリシーを無効化するまで、これらの保護は残ります。この時点で、Oracle Databaseでは内部ポリシーが不要になるため、自動的に無効化され、削除されます。(統合監査ポリシーは、削除前に無効化する必要があります。)TSDPポリシーを再度有効化すると、内部ポリシーが再作成されます。

親トピック: 統合監査での透過的機密データ保護ポリシーの使用

15.13.2 TSDPポリシーに使用される統合監査ポリシーの設定

監査ポリシー設定は、DBMS_TSDP_PROTECT.ADD_POLICYまたはDBMS_TSDP_PROTECT.ALTER_POLICYプロシージャのPOLICY_ENABLE_OPTIONSパラメータで使用できます。

これらの監査ポリシー設定は、AUDIT、CREATE AUDIT POLICYおよびALTER AUDIT POLICY文からの設定です。

次の表に、これらの設定を示します。

表15-2 TSDPポリシーに使用される統合監査ポリシーの設定

パラメータ	説明	デフォルト
ACTION_AUDIT_OPTIONS	SQL アクションのカンマ区切りリストを含む文字列。 該当するアクション: ALTER、AUDIT、COMMENT、DELETE、FLASHBACK、GRANT、INDEX、INSERT、LOCK、RENAME、SELECT、UPDATE これらのアクションをすべて監査するようにポリシーを構成するには、キーワード ALL を指定します。	ALL
AUDIT_CONDITION	SYS_CONTEXT (namespace, attribute) operation value-list この構文では、operation に、次の任意の演算子を使用できます: IN,| NOT IN, =,	NULL

パラメータ	説明	デフォルト
	<, >, or <> 監査条件に一重引用符が含まれている場合、1 つではなく 2 つの一重引用符を指定し、SYS_CONTEXT を一重引用符で囲みます。たとえば: `'SYS_CONTEXT(''USERENV'', ''CLIENT_IDENTIFIER'') = ''myclient'''`	
EVALUATE_PER	次のいずれかです。 ● STATEMENT ● SESSION ● INSTANCE	STATEMENT
ENTITY_NAME	ユーザーまたはロールのカンマ区切りリストを含む文字列。このパラメータを省略すると、すべてのユーザーに対して監査ポリシーが有効になります。	NULL (つまり、すべてのデータベース・ユーザー)
ENABLE_OPTION	ENTITY_NAME パラメータが使用されている場合にのみ適用されます。これは、ENTITY_NAME が BY ユーザー・リスト、EXCEPT ユーザー・リスト、BY USERS WITH GRANTED ROLES ロール・リストのいずれであるかを指定します。有効な設定は次のとおりです。 ● BY ● EXCEPT ● BY USERS WITH GRANTED ROLES	BY
UNIFIED_AUDIT_POLICY_COMMENT	作成される統合監査ポリシーを示す文字列。	NULL

親トピック: 統合監査での透過的機密データ保護ポリシーの使用

15.14 ファイングレイン監査での透過的機密データ保護ポリシーの使用

透過的機密データ保護およびファイングレイン監査プロシージャを使用すると、これら2つの機能の保護を組み合せることができます。

- ファイングレイン監査でのTSDPポリシーの使用について
  ファイングレイン監査のために透過的機密データ保護ポリシーを構成できます。
- TSDPポリシーに使用されるファイングレイン監査パラメータ
  DBMS_FGA.ADD_POLICY設定は、DBMS_TSDP_PROTECT.ADD_POLICYまたはDBMS_TSDP_PROTECT.ALTER_POLICYプロシージャのPOLICY_ENABLE_OPTIONSパラメータで使用できます。

親トピック: 透過的機密データ保護の使用

15.14.1 ファイングレイン監査でのTSDPポリシーの使用について

ファイングレイン監査のために透過的機密データ保護ポリシーを構成できます。

DBMS_TSDP_PROTECT.ADD_POLICYおよびDBMS_TSDP_PROTECT.ALTER_POLICYプロシージャを使用すると、DBMS_FGA.ADD_POLICYプロシージャの設定を指定できます。

この機能の手順は次のとおりです。

1. 必要なファイングレイン監査設定を使用して、TSDPポリシーを作成します。

   TSDPポリシーでは、DBMS_FGA.ADD_POLICYプロシージャのパラメータ設定が使用されます。これらの設定は、TSDPポリシーに使用されるファイングレイン監査パラメータにリストされています。

2. DBMS_TSDP_PROTECT.ASSOCIATE_POLICYプロシージャを使用して、TSDPポリシーを必要な機密タイプと関連付けます。

3. 次に、DBMS_TSDP_PROTECT.ENABLE_PROTECTION_*プロシージャのいずれかを使用して、TSDP保護を有効化します。

4. TSDPポリシーを有効化します。TSDPポリシー有効化プロセスの一環として、Oracle Databaseでは、ステップ1でDBMS_TSDP_PROTECT.ADD_POLICYプロシージャに指定したファイングレイン監査ポリシーが内部的に作成されます。

   内部ポリシーの名前は、ORA$FGA_で始まり、その後にランダムな英数文字列が続きます(たとえば、ORA$FGA_6J6L3RSJSN2VAN0XFなど)。このポリシーは、DBA_POLICIESデータ・ディクショナリ・ビューのPOLICY_NAME列に問い合せて確認できます。

5. TSDPポリシーによって保護されている表に対してユーザーがアクションの実行を試みると、ポリシー構成に基づいて、このオブジェクト・アクセスについてDBA_FGA_AUDIT_TRAILデータ・ディクショナリ・ビューにファイングレイン監査レコードが生成されます。

6. この列のTSDPポリシーを無効化するまで、これらの保護は残ります。この時点で、Oracle Databaseでは内部ポリシーが不要になるため、自動的に削除されます。TSDPポリシーを再度有効化すると、内部ポリシーが再作成されます。

親トピック: ファイングレイン監査での透過的機密データ保護ポリシーの使用

15.14.2 TSDPポリシーに使用されるファイングレイン監査パラメータ

DBMS_FGA.ADD_POLICY設定は、DBMS_TSDP_PROTECT.ADD_POLICYまたはDBMS_TSDP_PROTECT.ALTER_POLICYプロシージャのPOLICY_ENABLE_OPTIONSパラメータで使用できます。

次の表に、これらの設定を示します。

表15-3 TSDPポリシーに使用されるファイングレイン監査ポリシーの設定

パラメータ	説明	デフォルト
audit_condition	次の構文を使用して、監視状況を示すブール値を指定します。 operator value たとえば: < 1000	NULL
handler_schema	イベント・ハンドラを含むスキーマ。デフォルトの NULL の場合、現在のスキーマが使用されます。	NULL
handler_module	イベント・ハンドラのファンクション名。必要に応じて、パッケージ名を含めます。このファンクションは、問合せの監査条件と一致する最初の行が処理された後でのみ実行されます。例外が発生してプロシージャが異常終了すると、ユーザーの SQL 文も異常終了します。	NULL
statement_types	次の文タイプのいずれかを指定できます: INSERT、UPDATE、SELECT または DELETE。	SELECT
audit_trail	データベースを完全な統合監査にまだ移行していない場合は、この設定を使用して、監査レコードの書込み先(データベースの場合は DB、または XML レコードの場合は XML)を設定できます。この設定では、FGA_LOG$システム表の LSQLTEXT 列および LSQLBIND 列にデータを移入するかどうかについても指定します。 完全な統合監査が有効になっている場合は、Oracle Database によってこのパラメータは無視され、監査レコードが統合監査証跡に書	NULL

パラメータ	説明	デフォルト
	き込まれます。	
object_schema	機密列に対応するスキーマ	機密列が含まれているスキーマ
object_name	機密列が含まれている表	機密列が含まれているオブジェクト(表またはビュー)
policy_name	内部ファイングレイン監査ポリシーに対してシステムが生成した名前	システムが生成した内部ファイングレイン監査ポリシー名
audit_column	機密列	機密列
audit_column_opts	すべての列または特定の列の監査を指定します	DBMS_FGA.ANY_COLUMN
enable	TSDP ポリシーの有効化ステータス(TRUE または FALSE)	TRUE
policy_owner	DBMS_TSDP_PROTECT.ENABLE_PROTECTION_*プロシージャを起動するユーザー	現行ユーザー

親トピック: ファイングレイン監査での透過的機密データ保護ポリシーの使用

15.15 TDE列暗号化での透過的機密データ保護ポリシーの使用

TSDPプロシージャと透過的データ暗号化の列暗号化文を使用すると、これら2つの機能の保護を組み合せることができます。

- TDE列暗号化でのTSDPポリシーの使用について
  TSDPポリシーで、透過的データ暗号化を使用する列の暗号化を有効にできます。
- TSDPポリシーに使用されるTDE列暗号化ENCRYPT句の設定
  CREATE TABLEおよびALTER TABLE文のENCRYPT句の設定は、DBMS_TSDP_PROTECT.ADD_POLICYまたはDBMS_TSDP_PROTECT.ALTER_POLICYプロシージャのPOLICY_ENABLE_OPTIONSパラメータで使用できます。

親トピック: 透過的機密データ保護の使用

15.15.1 TDE列暗号化でのTSDPポリシーの使用について

TSDPポリシーで、透過的データ暗号化を使用する列の暗号化を有効にできます。

DBMS_TSDP_PROTECT.ADD_POLICYおよびDBMS_TSDP_PROTECT.ALTER_POLICYプロシージャを使用すると、CREATE TABLEまたはALTER TABLE文のENCRYPT句の設定を指定できます。

この機能の手順は次のとおりです。

1. DBMS_TSDP_PROTECT.ADD_POLICYプロシージャを使用して、TSDPポリシーを作成できます。ADD_POLICYプロシージャでは、SECURITY_FEATUREパラメータをDBMS_TSDP_PROTECT.COLUMN_ENCRYPTIONに設定することで、列暗号化のポリシーを構成できます。この設定により、TSDPポリシーがオブジェクトに対して有効になっている場合、機密列の暗号化が有効になります。
2. 必要な表暗号化設定を使用して、TSDPポリシーを作成します。

   TSDPポリシーでは、CREATE TABLEまたはALTER TABLE SQL文のパラメータ設定が使用されます。これらの設定は、TSDPポリシーに使用されるTDE列暗号化ENCRYPT句の設定にリストされています。
3. DBMS_TSDP_PROTECT.ASSOCIATE_POLICYプロシージャを使用して、TSDPポリシーを必要な機密タイプと関連付けます。
4. 次に、DBMS_TSDP_PROTECT.ENABLE_PROTECTION_*プロシージャのいずれかを使用して、TSDP保護を有効化します。
5. TSDPポリシーを有効化します。この時点で、Oracle Databaseでは、この手順の最初に作成した表暗号化設定を使用する内部TSDPポリシーが作成されます。

   内部ポリシーの名前は、ORA$TDECE_で始まり、その後にランダムな英数文字列が続きます(たとえば、ORA#TDECE_6J6L3RSJSN2VAN0XFなど)。このポリシーは、DBA_TSDP_POLICY_PROTECTIONビューのTSDP_POLICY列に問い合せて確認できます。
6. ポリシーによって保護されている表に対してユーザーがアクションの実行を試みると、実行中のTDE列の保護およびTSDPポリシーの両方に基づいて、列に対する出力が行われます。TSDPポリシーを有効化した後、列が暗号化されているかどうかを確認するには、DBA_ENCRYPTED_COLUMNSビューのENCRYPTION_ALG列を問い合せます。
7. この列のTSDPポリシーを無効化するまで、これらの保護は残ります。その時点で、Oracle Databaseでは、機密列が暗号化されるように、機密列が含まれている表に対してALTER TABLE文が内部的に発行されます。TSDPポリシーを再有効化すると、TSDPでは列に対してALTER TABLE文がENCRYPT句とともに内部的に実行されます。

ノート:

同じ列に対して、異なる暗号化アルゴリズムを指定した 2 つのポリシーを作成できます。この場合、2 つのアルゴリズムのうち、より強いアルゴリズムが機密列に対して実施されます。

親トピック: TDE列暗号化での透過的機密データ保護ポリシーの使用

15.15.2 TSDPポリシーに使用されるTDE列暗号化ENCRYPT句の設定

CREATE TABLEおよびALTER TABLE文のENCRYPT句の設定は、DBMS_TSDP_PROTECT.ADD_POLICYまたはDBMS_TSDP_PROTECT.ALTER_POLICYプロシージャのPOLICY_ENABLE_OPTIONSパラメータで使用できます。

次の表に、これらの設定を示します。

表15-4 TSDPポリシーに使用されるTDE列暗号化ENCRYPTの設定

パラメータ	説明	デフォルト
encrypt_algorithm	使用可能な値	AES192

パラメータ	説明	デフォルト
	• 3DES168 • AES128 • AES192 • AES256 • ARIA128 • ARIA192 • ARIA256 • SEED128 • GOST256	
salt	使用可能な値: • SALT • NO SALT	SALT
integrity_algorithm	使用可能な値: • SHA-1 • NOMAC	SHA-1

親トピック: TDE列暗号化での透過的機密データ保護ポリシーの使用

15.16 透過的機密データ保護のデータ・ディクショナリ・ビュー

Oracle Databaseには、透過的機密データ保護ポリシーに関する情報を表示するデータ・ディクショナリ・ビューが用意されています。

表15-5に、これらのビューを示します。これらのビューを使用する前に、SELECT_CATALOG_ROLEロールを付与する必要があります。

表15-5 透過的機密データ保護ビュー

ビュー	説明
DBA_DISCOVERY_SOURCE	透過的機密データの保護ポリシーに関するインポート情報の検出を説明します。
DBA_SENSITIVE_COLUMN_TYPES	現在のデータベースに定義されている機密列のタイプを説明します。

ビュー	説明
DBA_SENSITIVE_DATA	データベースの機密列を説明します。
DBA_TSDP_IMPORT_ERRORS	検出結果のインポート中に発生したエラーに関する情報を表示します。エラー・コード、スキーマ名、表名、列名および機密タイプに関する情報を表示します。
DBA_TSDP_POLICY_CONDITION	透過的機密データ保護ポリシーおよび条件マッピングを示します。このビューは、条件のプロパティと値のペアもリストします。
DBA_TSDP_POLICY_FEATURE	透過的機密データ保護ポリシーのセキュリティ機能マッピングを表示します。(現時点で、Oracle Data Redaction および Oracle Virtual Private Database のみサポートされます。)
DBA_TSDP_POLICY_PARAMETER	透過的機密データ保護ポリシーのパラメータを示します
DBA_TSDP_POLICY_PROTECTION	透過的機密データ保護を通じて保護されている列のリストを表示します
DBA_TSDP_POLICY_TYPE	ポリシーと機密列タイプのマッピングを表示します

関連トピック

- Oracle Databaseリファレンス

親トピック: 透過的機密データ保護の使用

16 データ・ディクショナリ内の機密性の高い資格証明データの暗号化

データ・ディクショナリ内に格納されているパスワードなどの機密性の高い資格証明情報を暗号化できます。

- データ・ディクショナリ内の機密性の高い資格証明データの暗号化について
  データ・ディクショナリSYS.LINK$およびSYS.SCHEDULER$_CREDENTIALシステム表は、ユーザー・パスワードなどの機密性の高い資格証明データを格納します。
- マルチテナント・オプションが機密データの暗号化に与える影響
  マルチテナント環境では、アプリケーション・ルートから、および個々のプラガブル・データベース(PDB)内で、機密のデータ・ディクショナリ情報を暗号化できます。
- システム表内の機密性の高い資格証明データの暗号化
  ALTER DATABASE DICTIONARY文は、SYS.LINK$システム表およびSYS.SCHEDULER$_CREDENTIALシステム表内の機密性の高い資格証明データを暗号化できます。
- SYS.LINK$システム表内の機密性の高い資格証明データのキーの更新
  ALTER DATABASE DICTIONARY文を使用して、データ・ディクショナリSYS.LINK$およびSYS.SCHEDULER$_CREDENTIALシステム表内の機密性の高い資格証明データのキーを更新できます。
- システム表内の機密性の高い資格証明データの削除
  ALTER DATABASE DICTIONARY文で、SYS.LINK$およびSYS.SCHEDULER$_CREDENTIAL内の既存の資格証明を無効にし、それらの表に対する今後の資格証明エントリを不明瞭化できます。
- キーストアの消失後のデータベース・リンクの機能の復元
  TDEキーストアおよびそのマスター暗号化キーが誤って失われると、データベース・リンクが悪影響を受ける可能性があります。
- 暗号化データ・ディクショナリ資格証明のデータ・ディクショナリ・ビュー
  Oracle Databaseには、データ・ディクショナリ内の機密性の高い資格証明データの暗号化に関する情報を提供するデータ・ディクショナリ・ビューのセットが用意されています。

親トピック: データへのアクセス制御

16.1 データ・ディクショナリ内の機密性の高い資格証明データの暗号化について

データ・ディクショナリSYS.LINK$およびSYS.SCHEDULER$_CREDENTIALシステム表にはユーザー・パスワードなどの機密性の高い資格証明データが格納されています。

SYS.LINK$表には、データベース・リンクに関する情報が格納されます。SYS.SCHEDULER$_CREDENTIALには、Oracle Schedulerイベントに関する情報が格納されます。デフォルトでは、これらの表に格納されている機密性の高い資格証明データは不明瞭化されています。

ALTER DATABASE DICTIONARY文を使用して、SYS.LINK$表およびSYS.SCHEDULER$_CREDENTIAL表に格納されているデータを手動で暗号化できます。この機能は透過的データ暗号化(TDE)を使用しますが、暗号化を実行するにはAdvanced Securityオプション・ライセンスは必要なく、SYSKM管理権限を持っている必要があります。TDEは、AES256(Advanced Encryption Standard)アルゴリズムを使用して暗号化を実行します。暗号化は、TDEを使用して暗号化されたその他のデータと同じ動作を行います。

最適なセキュリティ・プラクティスとして、この機密性の高い資格証明データを暗号化することをお薦めします。データ・ディクショナリの資格証明のステータスを確認するために、DICTIONARY_CREDENTIALS_ENCRYPTデータ・ディクショナリ・ビューを問い合せることができます。

親トピック: データ・ディクショナリでの機密性の高い資格証明データの暗号化

16.2 マルチテナント・オプションが機密データの暗号化に及ぼす影響

マルチテナント環境では、アプリケーション・ルートから、および個々のプラガブル・データベース(PDB)内の機密データ・ディクショナリ情報を暗号化できます。

SYS.LINK$システム表およびSYS.SCHEDULER$_CREDENTIALシステム表内の機密性の高い資格証明データの暗号化、キー更新、または復号化を行う場合は、プロセスの完了後に影響を受けたPDBを同期させる必要があります。これを行う方法については、これらのトピックについて取り上げている手順を参照してください。

親トピック: データ・ディクショナリでの機密性の高い資格証明データの暗号化

16.3 システム表内の機密性の高い資格証明データの暗号化

ALTER DATABASE DICTIONARY文は、SYS.LINK$システム表およびSYS.SCHEDULER$_CREDENTIALシステム表内の機密性の高い資格証明データを暗号化できます。

ENCRYPT CREDENTIALS句を指定したALTER DATABASE DICTIONARY文を実行してSYS.LINK$およびSYS.SCHEDULER$_CREDENTIALを暗号化する前に、データベースにオープン・キーストアおよび暗号化キーが必要です。資格証明データ暗号化プロセスは、不明瞭化されたパスワードの不明瞭化を解除してから、パスワードを暗号化します。この暗号化は、このプロセスが完了した後にユーザーが行う可能性のある今後のすべてのパスワード変更に適用されます。

1. SYSKM管理権限を付与されたユーザーとしてデータベース・インスタンスに接続します。
   たとえば:

   ```
   CONNECT hr_admin AS SYSKM
   Enter password: password
   ```

   マルチテナント環境では、アプリケーション・ルートまたはプラガブル・データベース(PDB)に接続します。
2. 必要に応じて、キーストアを作成して開き、次に暗号化キーを設定します。
   V$ENCRYPTION_WALLET動的ビューを問い合せて、キーストアのステータスを確認できます。
   ADMINISTER KEY MANAGEMENT文を使用してこれらの3個のタスクを実行します。たとえば:

   ```
   ADMINISTER KEY MANAGEMENT CREATE KEYSTORE '/etc/ORACLE/WALLETS/orcl' IDENTIFIED
   BY password;
   ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN IDENTIFIED BY "password";
   ADMINISTER KEY MANAGEMENT SET ENCRYPTION KEY IDENTIFIED BY "password" WITH
   BACKUP;
   ```

   マルチテナント環境で、現在の場所がアプリケーション・ルートの場合はCONTAINER = ALL句を含めます。これは統合モードのPDBのキーストア操作に適用されます。分離モードであるPDBの場合、PDB内から文を実行します。
3. ALTER DATABASE DICTIONARY文を実行して、データを暗号化します。
   たとえば:

   ```
   ALTER DATABASE DICTIONARY ENCRYPT CREDENTIALS;
   ```

   アプリケーション・ルートで、関連付けられたPDBに暗号化を適用するには、CONTAINER = ALL句を含めます。

```
ALTER DATABASE DICTIONARY ENCRYPT CREDENTIALS CONTAINER = ALL;
```

4. アプリケーション・ルートから暗号化を実行した場合、関連付けられたPDBを同期します。

```
ALTER PLUGGABLE DATABASE APPLICATION APP$CDB$SYSTEM SYNC;
```

親トピック: データ・ディクショナリでの機密性の高い資格証明データの暗号化

16.4 SYS.LINK$システム表内の機密性の高い資格証明データのキー更新

ALTER DATABASE DICTIONARY文を使用して、データ・ディクショナリSYS.LINK$およびSYS.SCHEDULER$_CREDENTIALシステム表内の機密性の高い資格証明データのキーを更新できます。

この機密性の高い資格証明データのキー更新を行うには、REKEY CREDENTIALS句を指定してALTER DATABASE DICTIONARY文を実行する必要があります。キー更新操作は列暗号化を使用しますが、他のTDEマスター暗号化キーには影響しません。

1. SYSKM管理権限を付与されたユーザーとしてデータベース・インスタンスに接続します。
   たとえば:

```
CONNECT hr_admin AS SYSKM
Enter password: password
```

   マルチテナント環境では、アプリケーション・ルートまたはプラガブル・データベース(PDB)に接続します。
2. 必要に応じて、キーストアを作成して開き、次に暗号化キーを設定します。
   V$ENCRYPTION_WALLET動的ビューを問い合せて、キーストアのステータスを確認できます。
   ADMINISTER KEY MANAGEMENT文を使用してこれらの3個のタスクを実行します。たとえば:

```
ADMINISTER KEY MANAGEMENT CREATE KEYSTORE '/etc/ORACLE/WALLETS/orcl' IDENTIFIED
BY password;
ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN IDENTIFIED BY "password";
ADMINISTER KEY MANAGEMENT SET ENCRYPTION KEY IDENTIFIED BY "password" WITH
BACKUP;
```

   マルチテナント環境で、現在の場所がアプリケーション・ルートの場合はCONTAINER = ALL句を含めます。
3. ALTER DATABASE DICTIONARY文を実行してデータのキー更新を行います。
   たとえば:

```
ALTER DATABASE DICTIONARY REKEY CREDENTIALS;
```

   アプリケーション・ルートで、関連付けられたPDBに暗号化を適用するには、CONTAINER = ALL句を含めます。

```
ALTER DATABASE DICTIONARY REKEY CREDENTIALS CONTAINER = ALL;
```

4. アプリケーション・ルートからキー更新操作を実行した場合、関連付けられたPDBを同期します。

```
ALTER PLUGGABLE DATABASE APPLICATION APP$CDB$SYSTEM SYNC;
```

親トピック: データ・ディクショナリでの機密性の高い資格証明データの暗号化

16.5 システム表内の機密性の高い資格証明データの削除

ALTER DATABASE DICTIONARY文は、SYS.LINK$およびSYS.SCHEDULER$_CREDENTIAL内の既存の資格証明を無効にし、それらの表に対する今後の資格証明エントリを不明瞭化できます。

この資格証明データを削除するには、DELETE CREDENTIALS句を指定してALTER DATABASE DICTIONARY文を実行する必要があります。この文は主に透過的データ暗号化(TDE)キーストアの消失からデータベース・リンクをリカバリする必要がある場合に使用されます。

1. SYSKM管理権限を付与されたユーザーとしてデータベース・インスタンスに接続します。
   たとえば:

   ```
   CONNECT hr_admin AS SYSKM
   Enter password: password
   ```

   マルチテナント環境では、アプリケーション・ルートまたはプラガブル・データベース(PDB)に接続します。
2. 必要に応じて、キーストアを作成して開き、次に暗号化キーを設定します。
   V$ENCRYPTION_WALLET動的ビューを問い合せて、キーストアのステータスを確認できます。
   ADMINISTER KEY MANAGEMENT文を使用してこれらの3個のタスクを実行します。たとえば:

   ```
   ADMINISTER KEY MANAGEMENT CREATE KEYSTORE '/etc/ORACLE/WALLETS/orcl' IDENTIFIED
   BY password;
   ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN IDENTIFIED BY "password";
   ADMINISTER KEY MANAGEMENT SET ENCRYPTION KEY IDENTIFIED BY "password" WITH
   BACKUP;
   ```

   マルチテナント環境で、現在の場所がアプリケーション・ルートの場合はCONTAINER = ALL句を含めます。
3. ALTER DATABASE DICTIONARY文を実行してパスワード資格証明を削除します。
   たとえば:

   ```
   ALTER DATABASE DICTIONARY DELETE CREDENTIALS KEY;
   ```

   アプリケーション・ルートで、関連付けられたPDB内のSYS.LINK$およびSYS.SCHEDULER$_CREDENTIALパスワード資格証明を削除するには、CONTAINER = ALL句を含めます。

   ```
   ALTER DATABASE DICTIONARY DELETE CREDENTIALS CONTAINER = ALL;
   ```
4. アプリケーション・ルートからの資格証明の削除を実行した場合、関連付けられたPDBを同期します。

   ```
   ALTER PLUGGABLE DATABASE APPLICATION APP$CDB$SYSTEM SYNC;
   ```

関連トピック

- キーストアの消失後のデータベース・リンクの機能の復元

親トピック: データ・ディクショナリでの機密性の高い資格証明データの暗号化

16.6 キーストアの消失後のデータベース・リンクの機能の復元

TDEキーストアおよびそのマスター暗号化キーが誤って失われると、データベース・リンクが悪影響を受ける可能性があります。

TDEキーストアとマスター暗号化キーが失われた場合、暗号化されたパスワードを使用して認証されている既存のデータベース・リンクが使用できなくなります。

1. SYSKM管理権限を付与されていてALTER DATABASE LINKシステム権限を持つユーザーとしてデータベース・インスタンスに接続します。
   たとえば:

   ```
   CONNECT hr_admin AS SYSKM
   Enter password: password
   ```

マルチテナント環境では、アプリケーション・ルートまたはプラガブル・データベース(PDB)に接続します。

2. SYS.LINK$システム表から暗号化された資格証明を削除します。

```
ALTER DATABASE DICTIONARY DELETE CREDENTIALS KEY;
```

アプリケーション・ルートからの削除を実行している場合、CONTAINER = ALL句を含めます。

```
ALTER DATABASE DICTIONARY DELETE CREDENTIALS CONTAINER = ALL;
```

3. キーストアを作成して開き、次に暗号化キーを設定します。

たとえば:

```
ADMINISTER KEY MANAGEMENT CREATE KEYSTORE '/etc/ORACLE/WALLETS/orcl' IDENTIFIED
BY password;
ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN IDENTIFIED BY "password";
ADMINISTER KEY MANAGEMENT SET ENCRYPTION KEY IDENTIFIED BY "password" WITH
BACKUP;
```

マルチテナント環境で、現在の場所がアプリケーション・ルートの場合はCONTAINER = ALL句を含めます。

4. SYS.LINK$およびSYS.SCHEDULER$_CREDENTIALのパスワード資格証明を暗号化します。

```
ALTER DATABASE DICTIONARY ENCRYPT CREDENTIALS;
```

アプリケーション・ルートから暗号化を実行している場合、CONTAINER = ALL句を含めます。

```
ALTER DATABASE DICTIONARY ENCRYPT CREDENTIALS CONTAINER = ALL;
```

5. データベース・リンクに関連付けられているユーザーのパスワードを使用して、ALTER DATABASE DICTIONARY DELETE CREDENTIALS KEY文の影響を受けたデータベース・リンクのパスワードをリセットします。

たとえば:

```
ALTER DATABASE LINK database_link_name CONNECT TO user_id IDENTIFIED BY
password;
```

既存のデータベース・リンクおよびその所有者を検索するには、DBA_DB_LINKSデータ・ディクショナリ・ビューを問い合せます。

6. アプリケーション・ルートからの資格証明の削除を実行した場合、関連付けられたPDBを同期します。

```
ALTER PLUGGABLE DATABASE APPLICATION APP$CDB$SYSTEM SYNC;
```

親トピック: データ・ディクショナリでの機密性の高い資格証明データの暗号化

16.7 暗号化データ・ディクショナリ資格証明のデータ・ディクショナリ・ビュー

Oracle Databaseには、データ・ディクショナリ内の機密性の高い資格証明データの暗号化に関する情報を提供する一連のデータ・ディクショナリ・ビューが用意されています。

表16-1に、データ・ディクショナリ・ビューを示します。

表16-1 暗号化データ・ディクショナリ資格証明のデータ・ディクショナリ・ビュー

ビュー	説明
ALL_DB_LINKS	現行のユーザーがアクセスできるデータベース・リンクを示します。VALID列の値がYESの場合、データベース・リンクが使用可能であることを示

ビュー	説明
	します。
DBA_DB_LINKS	データベース内のデータベース・リンクをすべて示します。VALID 列の値が YES の場合、データベース・リンクが使用可能であることを示します。(このビューは、SYS または DBA ロールを付与されているユーザーなどの管理ユーザーのみ使用できます。)
DICTIONARY_CREDENTIALS_ENCRYPT	ディクショナリ資格証明のステータスを示します。ENFORCEMENT 列は、資格証明が暗号化されている場合は ENABLED を、資格証明が暗号化されていない場合は DISABLED をリストします。
USER_DB_LINKS	現行のユーザーが所有するデータベース・リンクを示します。VALID 列の値が YES の場合、データベース・リンクが使用可能であることを示します。

関連トピック

- Oracle Databaseリファレンス

親トピック: データ・ディクショナリでの機密性の高い資格証明データの暗号化

17 手動によるデータ暗号化

DBMS_CRYPTO PL/SQLパッケージを使用して、データを手動で暗号化できます。

- 暗号化で解決しないセキュリティの問題
  データの暗号化には多くのメリットがありますが、多くのデメリットもあります。
- データ暗号化の課題
  暗号化によってセキュリティが強化される場合、いくつかの技術的な課題が伴います。
- DBMS_CRYPTOパッケージを使用したデータ暗号化ストレージ
  DBMS_CRYPTOパッケージは、セキュリティ問題に取り組むための様々な方法を提供します。
- DBMS_CRYPTOパッケージを使用した非対称キー操作
  DBMS_CRYPTOパッケージには、暗号化、復号化、署名および検証のための非対称キー操作を実行できる4つのファンクションが用意されています。
- OFBモードで暗号化された暗号文のOracle Databaseリリース11gでの使用
  Oracle Databaseリリース11gでは、出力フィードバック(OFB)を使用するように暗号文が構成されても、かわりに電子コードブック(ECB)モードが使用されました。
- データの暗号化APIの使用例
  データ暗号化APIの使用例として、DBMS_CRYPTO.SQLプロシージャの使用、AES 256ビット・データの暗号化およびBLOBデータの暗号化などがあります。
- 暗号化データのデータ・ディクショナリ・ビュー
  Oracle Databaseには、暗号化されたデータに関する情報を検索できるデータ・ディクショナリ・ビューが用意されています。

親トピック: データへのアクセス制御

17.1 暗号化で解決しないセキュリティの問題

データの暗号化には多くのメリットがありますが、多くのデメリットもあります。

- 原則1: 暗号化はアクセス制御の問題を解決しない
  データを暗号化するとき、暗号化によってアクセス制御の構成が妨げられないように注意する必要があります。
- 原則2: 暗号化は不正な管理者からデータを保護しない
  Oracle Database VaultなどのOracle機能を使用して、不正なデータベース管理者からデータベースを保護できます。
- 原則3: すべてのデータを暗号化してもデータは保護されない
  通常、データの暗号化によってセキュリティが強化されるのであれば、すべてのデータを暗号化することにより、データはすべて保護されると考えますが、これは誤りです。

親トピック: 手動によるデータ暗号化

17.1.1 原則1: 暗号化はアクセス制御の問題を解決しない

データを暗号化するとき、暗号化によってアクセス制御の構成が妨げられないように注意する必要があります。

ほとんどの組織で、データのアクセスを、そのデータを参照する必要があるユーザーのみに制限する必要があります。たとえば、人事管理システムでは、従業員には各自の雇用レコードのみを参照できるように制限し、従業員の上司には直属の部下の雇用レコードを参照できるようにする場合があります。人事部門の担当者が複数の従業員の従業員レコードを参照する必要がある

場合もあります。

一般的には、アクセス制御メカニズムを使用して、データを見る必要がある人にデータ・アクセスを制限するセキュリティ・ポリシーに対処できます。Oracle Databaseは、強力で独自に評価されたアクセス制御メカニズムを何年もの間提供してきました。仮想プライベート・データベースにより、アクセス制御を詳細なレベルまで規定できます。

人事部門のレコードは機密情報とみなされるため、セキュリティをより強化するためにこの情報はすべて暗号化する必要があると考えがちです。しかし、暗号化によってきめ細かいアクセス制御を規定できなくなるため、データ・アクセスの妨げとなる場合があります。たとえば、従業員、その上司および人事部門の担当者すべてが、従業員レコードにアクセスする必要がある場合があります。従業員データがすべて暗号化されている場合、この3人は暗号化されていない形式のデータにアクセスできる必要があります。したがって、従業員、上司および人事部門の担当者はデータを復号化するために同じ暗号化キーを共有する必要があります。このように、アクセス制御の向上という観点では、暗号化によってセキュリティが強化されず、アプリケーションの正しい機能や効率的な機能の妨げとなる場合があります。さらに、システムの複数のユーザー間で暗号化キーの送信および共有を安全に行うことは難しいという別の問題もあります。

格納データを暗号化するときの基本原則は、暗号化によってアクセス制御が妨げられないようにすることです。たとえば、empに対するSELECT権限を持つユーザーが、基本的には参照できるすべてのデータへの参照を暗号化メカニズムによって制限されないようにする必要があります。また、ユーザーが表のすべての暗号化データを参照する必要がある場合は、あるキーを使用して表の一部を暗号化し、別のキーを使用して表の他の部分を暗号化するメリットはほとんどありません。この場合、ユーザーがデータを参照できるように、データを復号化するオーバーヘッドが増加するのみです。アクセス制御が適切に実装されている場合は、暗号化によってデータベース自体の中でセキュリティが強化されることはほとんどありません。データベース内のデータにアクセスする権限を持つユーザーについては、暗号化しても権限は変わりません。したがって、アクセス制御の問題解決に暗号化を使用しないでください。

親トピック: 暗号化で解決しないセキュリティの問題

17.1.2 原則2: 暗号化は不正な管理者からデータを保護しない

Oracle Database VaultなどのOracle機能を使用して、不正なデータベース管理者からデータベースを保護できます。

不正なユーザーがパスワードを推測して上位(データベース管理者)の権限を取得する可能性を懸念して、この脅威から保護するために格納データを暗号化することを考える組織もあります。

ただし、この問題の正しい解決策は、データベース管理者アカウントを保護し、その他の権限付きアカウントのデフォルトのパスワードを変更することです。データベースに侵入する最も簡単な方法は、管理者が変更しないままにしている権限付きアカウントのデフォルトのパスワードを使用することです。その一例がSYS/CHANGE_ON_INSTALLです。

不正なユーザーがDBA権限を取得すると、データベースに対して多くの破壊的な行為を実行できますが、暗号化ではそれらの行為の多くからデータベースを保護できません。たとえば、データを破壊または削除する行為、ユーザー・データをファイル・システムにエクスポートし、そのデータを電子メールで自分自身に戻してパスワード・クラッカを実行するなどの行為があります。

データベース管理者は通常すべての権限を所有しているため、データベース管理者がデータベース内のすべてのデータを参照できることについて懸念を持つ組織もあります。そのような組織では、データベース管理者はデータベースを管理するのみで、そのデータベースに含まれるデータを参照できないようにする必要があると考えます。また、1人のユーザーに非常に多くの権限が集中することを懸念し、DBA機能を分割するか、または2人でアクセスするルールを規定する方が望ましいと考える組織もあります。

すべてのデータ(または大量のデータ)を暗号化すると、前述の問題が解決すると考えがちですが、このような不正行為からデータを保護するためのより優れた方法があります。たとえば、Oracle DatabaseではDBA権限の制限付きの分割がサポートされています。Oracle Databaseでは、SYSDBAユーザーとSYSOPERユーザーに対するネイティブなサポートが提供されています。SYSDBAはすべての権限を所有しますが、SYSOPERは制限付き権限セット(データベースの起動および停止など)を所有します。

さらに、複数のシステム権限を網羅する、より小さなロールを作成することもできます。たとえば、jr_dbaロールには、すべてのシステム権限を含めるのではなく、準データベース管理者に適切なロールのみ(CREATE TABLE、CREATE USERなど)を含めます。

Oracle Databaseは、SYS(またはSYS権限を持つユーザー)によって実行されたアクションを監査し、オペレーティング・システムの保護位置にその監査証跡を格納できます。このモデルを使用すると、オペレーティング・システムに対してルート権限を持つ別の監査人が、SYSによって実行されたすべてのアクションを監査できるため、監査人は、すべてのデータベース管理者が実行したアクションに対する責任をデータベース管理者自身に持たせることができます。

データベース管理者のアクセス権と制御権をOracle Database Vaultを使用して調整することもできます。

データベース管理者は信頼される立場にいます。諜報機関のような最も機密性の高いデータを扱う組織においても、通常、データベース管理者機能は分割されません。かわりに、データベース管理者には信頼性が必要なため、厳しく管理されます。定期的な監査によって不適切なアクティビティを明らかにできます。

格納データの暗号化が、データベースの管理を妨げないことが重要です。そうでない場合は、より大きいセキュリティ問題になる可能性があります。たとえば、データを暗号化することでそのデータが破損した場合セキュリティ問題となり、データ自体を解釈できず、修復不可能になる可能性があります。

暗号化を使用すると、データベース管理者(または権限を持つその他のユーザー)がデータベース内のデータを参照する機能を制限できます。ただし、これは、データベース管理者の権限を適切に管理すること、または強力なシステム権限の使用を制御することの代用にはなりません。信頼できないユーザーが重要な権限を持っている場合、そのユーザーが組織に多くの脅威をもたらすこととなり、暗号化されていないクレジット・カード番号を参照されるよりもはるかに深刻な状況に陥る場合もあります。

関連項目:

Oracle Database Vaultを使用したデータベース管理者のアクセス権と制御権の調整の詳細は、『Oracle Database Vault管理者ガイド』を参照してください。

親トピック: 暗号化で解決しないセキュリティの問題

17.1.3 原則3: すべてのデータを暗号化してもデータは保護されない

通常、データの暗号化によってセキュリティが強化されるのであれば、すべてのデータを暗号化することにより、データはすべて保護されると考えますが、これは誤りです。

前述の2つの原則で説明したとおり、暗号化ではアクセス制御の問題に適切に対処できないため、暗号化によって通常のアクセス制御を妨げないことが重要になります。さらに、本番データベース全体を暗号化すると、読取り、更新または削除を行うために、すべてのデータを復号化する必要があります。暗号化は、本来、パフォーマンス集中型の操作であるため、すべてのデータを暗号化するとパフォーマンスに大きい影響を与えます。

可用性はセキュリティの重要な側面です。データの暗号化によって、データを使用できなくなったり、パフォーマンスが低下して可用性に悪影響を与えた場合は、すべてのデータを暗号化することで新たなセキュリティ問題が発生することになります。セキュリティの適切な手続きとして、暗号化キーを定期的に変更する必要がありますが、このときデータベースにアクセスできなくなります。これも可用性に悪影響を与えます。キーを変更する場合は、データを復号化して新しいキーで再度暗号化するまでの間、データベースにアクセスできなくなります。

格納データをオフラインで暗号化することは、メリットもあります。たとえば、組織が半年から1年単位で遠隔地にオフラインでバックアップを格納する場合があります。もちろん、保護の第1段階は、物理的なアクセス制御を確立することによって、データを格納する施設を保護することです。さらに、このデータを格納する前に暗号化すると、メリットがあります。このデータはオンラインでア

クセスされないため、パフォーマンスについて考慮する必要はありません。Oracleデータベースにはこの機能が備わっていませんが、暗号化サービスを提供するベンダーがあります。この方法を検討する組織は、バックアップ・データの大規模な暗号化を実施する前に、プロセスを徹底的にテストする必要があります。オフラインで格納する前に、暗号化されたデータを正常に復号化でき、正しく再インポートできることを確認する必要があります。

親トピック: 暗号化で解決しないセキュリティの問題

17.2 データ暗号化の課題

暗号化によってセキュリティが強化される場合、いくつかの技術的な課題が伴います。

- 暗号化され索引付けされたデータ
  暗号化データが索引付けされている場合は、特別な問題が発生します。
- 生成された暗号化キー
  暗号化されたデータの安全性は、データの暗号化に使用されるキーの安全性に依存します。
- 転送された暗号化キー
  暗号化キーがアプリケーションによってデータベースに渡される場合は、暗号化キーを暗号化する必要があります。
- 暗号化キーの格納
  データベースまたはオペレーティング・システムに暗号化キーを格納できます。
- 暗号化キーの変更の重要性
  慎重なセキュリティの手続きでは、暗号化キーを定期的に変更するように指示されます。
- バイナリ・ラージ・オブジェクトの暗号化
  一部のデータ型は、暗号化に手間がかかります。

親トピック: 手動によるデータ暗号化

17.2.1 暗号化され索引付けされたデータ

暗号化データが索引付けされている場合は、特別な問題が発生します。

たとえば、ある会社が国民識別番号(米国社会保障番号(SSN)など)を各従業員の従業員番号として使用しているとします。会社は従業員番号を機密性の高いデータと考えているため、`employees`表の`employee_number`列のデータを暗号化します。`employee_number`列には一意の値が含まれているため、データベース設計者は、パフォーマンスを向上するためにそのデータを索引付けします。

ただし、`DBMS_CRYPTO` (または別のメカニズム)を使用して列のデータを暗号化した場合、その列の索引にも暗号化された値が含まれます。列の索引に暗号化された値が含まれている場合、この索引は等価性チェック(たとえば、`SELECT * FROM emp WHERE employee_number = '987654321'`)には使用できますが、それ以外の目的には事実上使用できません。索引付けされたデータを暗号化しないでください。

国民識別番号は一意のIDとして使用しないことをお薦めします。かわりに、`CREATE SEQUENCE`文を使用して、一意の識別番号を生成してください。国民識別番号を使用しない理由は、次のとおりです。

- 国民識別番号の乱用に関連するプライバシの問題があること(識別情報の盗難など)
- 国民識別番号に一部の重複があること(例: 米国社会保障番号)

親トピック: データ暗号化の課題

17.2.2 生成された暗号化キー

暗号化されたデータの安全性は、データの暗号化に使用されるキーの安全性に依存します。

暗号化キーは、安全な暗号化キー生成方法を使用して安全に生成する必要があります。Oracle Databaseは、DBMS_CRYPTOのRANDOMBYTESファンクションを使用した完全な乱数の生成を提供しています。(このファンクションは、非推奨になっている以前のDBMS_OBFUSCATION_TOOLKITのGetKeyプロシージャによって提供されていた機能にかわるものです。)DBMS_CRYPTOは、以前にRSA Securityによって認証された安全な乱数ジェネレータ(RNG)をコールします。

ノート:

DBMS_RANDOM パッケージは使用しないでください。DBMS_RANDOM パッケージは擬似乱数を生成します。「セキュリティのための乱雑性についての推奨事項(RFC-1750)」には、擬似乱数プロセスを使用して機密の数値を生成すると、真のセキュリティを維持できなくなると述べられています。

キーの値を暗号化する場合は、必ず正しいバイト数を指定してください。たとえば、ENCRYPT_AES128暗号化アルゴリズムの場合は、16バイトのキーを指定する必要があります。

親トピック: データ暗号化の課題

17.2.3 転送された暗号化キー

暗号化キーがアプリケーションによってデータベースに渡される場合は、暗号化キーを暗号化する必要があります。

暗号化しないと、キーの転送時に、侵入者がキーにアクセスできる可能性があります。ネットワーク・データの暗号化では、暗号化キーを含む送信中のデータはすべて、改ざんまたは傍受から保護されます。

関連トピック

- Oracle Databaseのネイティブ・ネットワーク暗号化とデータ整合性の構成

親トピック: データ暗号化の課題

17.2.4 暗号化キーの格納

データベースまたはオペレーティング・システムに暗号化キーを格納できます。

- 暗号化キーの格納について
  暗号化キーの格納は、暗号化で最も重要で難しい問題の1つです。
- データベースへの暗号化キーの格納
  データベースに暗号化キーを格納しても、データベース管理者が暗号化データにアクセスできなくなるとはかぎりません。
- オペレーティング・システムへの暗号化キーの格納
  暗号化キーをオペレーティング・システムのフラット・ファイルに格納する場合、PL/SQLからコールアウトを実行して暗号化キーを取得できます。
- ユーザー自身による暗号化キーの管理
  ユーザーにキーを提供させる場合は、ユーザー自身がそのキーに責任を持つことを想定しています。
- 透過的データベース暗号化および表領域暗号化を使用した手動暗号化
  透過的データベース暗号化および表領域暗号化は、暗号化された表および表領域に対するキーを自動的に管理して、安全な暗号化を提供します。

親トピック: データ暗号化の課題

17.2.4.1 暗号化キーの格納について

暗号化キーの格納は、暗号化で最も重要で難しい問題の1つです。

対称キーで暗号化されたデータを復元するには、データを復号化しようとする承認済のアプリケーションまたはユーザーがキーにアクセスできる必要があります。同時に、アクセス権のない暗号化データに不正アクセスしようとするユーザーは、キーにアクセスできないようにする必要があります。

親トピック: 暗号化キーの格納

17.2.4.2 データベースへの暗号化キーの格納

データベースに暗号化キーを格納しても、データベース管理者が暗号化データにアクセスできなくなるとは限りません。

すべての権限を持つデータベース管理者は暗号化キーを含む表にもアクセスできます。ただし、明確な意図のない詮索好きなユーザーまたはオペレーティング・システム上のデータベース・ファイルを破壊しようとするユーザーに対しては、適切なセキュリティ対策になる可能性があります。

簡単な例として、従業員データを含む表(EMP)を作成するとします。列の1つに格納されている各従業員の社会保障番号(SSN)を暗号化します。従業員のSSNは、別の列に格納されているキーを使用して暗号化できます。ただし、表全体に対するSELECT権限を持っているユーザーは、暗号化キーを取得し、対応するSSNを復号化できます。

この暗号化スキーマは簡単に侵害できるようにみえますが、多少の取り組みで、非常に強力な不正侵入対策ソリューションを作成できます。たとえば、この方法を使用してSSNを暗号化する前に、employee_numberのデータをさらに変換するテクニックを使用することで、SSNを暗号化できます。これは、employee_numberと従業員の誕生日のXOR(排他的論理和)演算を使用して値の妥当性を判別するような単純なテクニックの場合もあります。

別の保護として、暗号化を実行するPL/SQLパッケージの本体を(WRAPユーティリティを使用して)ラップして、コードを不明瞭化(スクランブル化)する方法もあります。WRAPユーティリティは入力SQLファイルを処理し、その中のPL/SQLユニットを不明瞭化します。たとえば、次のコマンドではkeymanage.sqlファイルを入力として使用しています。

```
wrap iname=/mydir/keymanage.sql
```

開発者は次に、ラップされたパッケージに含まれるキーを使用して、パッケージ内のファンクションにDBMS_CRYPTOパッケージ・コールをコールさせることができます。

Oracle Databaseでは、動的に生成されたPL/SQLコードを不明瞭化できます。DBMS_DDLパッケージには、動的に生成されたPL/SQLプログラム・ユニットを不明瞭化するために使用できるサブプログラムが2つ含まれています。たとえば、次のブロックでは、DBMS_DDL.CREATE_WRAPPEDプロシージャを使用して、動的に生成されたPL/SQLコードをラップします。

```
BEGIN
......
SYS.DBMS_DDL.CREATE_WRAPPED(function_returning_PLSQL_code());
......
END;
```

ラップは解読不可能ではありませんが、侵入者が暗号化キーにアクセスするのはかなり困難になります。各暗号化データの値ごとに異なるキーが提供されている場合でも、キーの値をパッケージ内に埋め込まないでください。かわりに、キーの管理を実行するパッケージをラップ(つまり、データ変換またはパディング)してください。

関連項目:

WRAPコマンドライン・ユーティリティと動的ラップのためのDBMS_DDLサブプログラムの詳細は、『Oracle Database PL/SQLパッケージおよびタイプ・リファレンス』を参照してください。

また、データをラップするかわりに暗号化キーを別の表に格納し、プロシージャを使用してキー表に対するコールをエンベロープする方法もあります。キー表は、主キーと外部キー関係を使用してデータ表に結合できます。たとえば、employee_numberは、従業員情報と暗号化されたSSNを格納するemployees表の主キーです。また、employee_number列は、従業員のSSNの暗号化キーを格納するssn_keys表に対する外部キーです。ssn_keys表に格納されたキーも使用前に(XOR演算により)変換できるため、キー自体が暗号化されずに格納されることはありません。プロシージャをラップすると、キーを使用前に変換する方法を隠すことができます。

この方法のメリットは次のとおりです。

- 表への直接アクセス権を持つユーザーは、暗号化されていない機密データを参照することも、キーを取得してデータを復号化することもできません。
- 復号化されたデータへのアクセスは、暗号化されたデータの選択、キー表からの復号化キーの取得、およびデータの復号化に使用するキーの変換を実行するプロシージャを介して制御できます。
- データ変換アルゴリズムは、プロシージャをラップし、プロシージャ・コードを不明瞭化することによって、偶発的な傍受から隠されます。
- データ表とキー表の両方に対するSELECT権限があっても、キーは使用前に変換されるため、この権限を持つユーザーがそのデータを復号化できる保証はありません。

この方法のデメリットは、キー表とデータ表の両方に対するSELECT権限があり、キー変換アルゴリズムを導出できるユーザーが、暗号化スキームを解読できることです。

前述の方法は完全ではありませんが、クリアテキストで格納されている機密情報を簡単に取得できないように保護するには十分です。

親トピック: 暗号化キーの格納

17.2.4.3 オペレーティング・システムへの暗号化キーの格納

暗号化キーをオペレーティング・システムのフラット・ファイルに格納する場合、PL/SQLからコールアウトを実行して暗号化キーを取得できます。

ただし、オペレーティング・システムにキーを格納し、それに対してコールアウトを行う場合、データはオペレーティング・システムでの保護と同じ程度にのみ保護されます。

セキュリティ上の主な懸念が、オペレーティング・システムからデータベースに侵入される可能性があるということである場合、オペレーティング・システムにキーを格納することは、データベース自体にキーを格納することより、侵入者にとっては暗号化されたデータを取得しやすくなります。

親トピック: 暗号化キーの格納

17.2.4.4 ユーザー自身による暗号化キーの管理

ユーザーにキーを提供させる場合は、ユーザー自身がそのキーに責任を持つことを想定しています。

ヘルプ・デスクへのコールの40%が、パスワードを忘れたユーザーからのコールであることから、ユーザーが暗号化キーを管理することのリスクは明らかです。多くの場合、ユーザーは暗号化キーを忘れるか、キーを書き留めておくため、セキュリティ上の弱点が生じます。ユーザーが暗号化キーを忘れたり、会社を退職した場合、データは復元できなくなります。

ユーザーにキーを提供する、またはユーザーがキーを管理する場合は、ネイティブ・ネットワーク暗号化を使用して、キーがクリアテ

キストでクライアントからサーバーに渡されないようにする必要があります。また、キー・アーカイブ・メカニズムを開発する必要があります。これも困難なセキュリティの問題です。キー・アーカイブおよびバックドアは、暗号化によって解決しようとしているセキュリティ上の弱点を生み出すことになります。

親トピック: 暗号化キーの格納

17.2.4.5 透過的データベース暗号化および表領域暗号化を使用した手動暗号化

透過的データベース暗号化および表領域暗号化は、暗号化された表および表領域に対するキーを自動的に管理して、安全な暗号化を提供します。

アプリケーションで、メディアに格納されている機密性の高い列データの保護を必要とする場合は、これらの2種類の暗号化を使用することで、これを簡単にすばやく行えます。

関連項目:

透過的データ暗号化の詳細は、『Oracle Database Advanced Securityガイド』を参照してください。

親トピック: 暗号化キーの格納

17.2.5 暗号化キーの変更の重要性

慎重なセキュリティの手続きでは、暗号化キーを定期的に変更するように指示されます。

格納データの場合、これを実行するには、データを定期的に暗号解除し、適切な別のキーを使用して再度暗号化する必要があります。

暗号化キーの変更は、おそらくデータへのアクセスがない間に実行しますが、また別の課題が生まれます。クレジット・カード番号を暗号化するWebベースのアプリケーションでは、暗号化キーを切り替える間はアプリケーション全体を停止できないため、特に問題になります。

親トピック: データ暗号化の課題

17.2.6 バイナリ・ラージ・オブジェクトの暗号化

一部のデータ型は、暗号化に手間がかかります。

たとえば、Oracle Databaseでは、非常に大規模なオブジェクト(たとえば、数GB)をデータベースに格納するバイナリ・ラージ・オブジェクト(BLOB)をサポートしています。BLOBは、列として内部的に格納するか、または外部ファイルに格納できます。

関連トピック

- BLOBデータの暗号化および復号化プロシージャの例

親トピック: データ暗号化の課題

17.3 DBMS_CRYPTOパッケージを使用したデータ暗号化ストレージ

DBMS_CRYPTOパッケージは、セキュリティ問題に取り組むための様々な方法を提供します。

暗号化は、複数のセキュリティへの脅威に対処できる理想的なソリューションではありませんが、データベースに格納する前に機密性の高いデータを選択的に暗号化することによって、セキュリティを強化できます。このようなデータの例としては、クレジット・カード番号や国民識別番号などがあります。

Oracle Databaseには、格納データの暗号化および復号化のためのPL/SQLパッケージDBMS_CRYPTOが用意されています。このパッケージは、Advanced Encryption Standard(AES)の暗号化アルゴリズムも含めて、複数の業界標準暗号化およびハッシング・アルゴリズムをサポートしています。AESは、データ暗号化規格(DES)にかわる規格として、National Institute of Standards and Technology(NIST)によって承認されました。

DBMS_CRYPTOパッケージは、RAWおよびイメージやサウンドなどのラージ・オブジェクト(LOB)を含むOracle Databaseの一般的なデータ型の暗号化および復号化に使用できます。特に、BLOBとCLOBをサポートしています。さらに、様々なデータベース文字セット間でデータを暗号化するためのグローバリゼーション・サポートも提供します。

次の暗号化アルゴリズムがサポートされています。

- Advanced Encryption Standard(AES)
- SHA-2暗号化ハッシュ設定:
  - HASH_SH256
  - HASH_SH384
  - HASH_SH512
- SHA-2メッセージ認証コード(MAC)

DBMS_CRYPTOではブロック暗号修飾子も提供されています。Public Key Cryptographic Standard(PKCS)#5を含む複数のパディング・オプションおよびCipher Block Chaining(CBC)を含む4つのブロック暗号連鎖モードから選択できます。パディングは8バイトの倍数で実行する必要があります。

ノート:

- DES は National Institute of Standards and Technology(NIST)の推奨対象ではなくなりました。
- SHA-1 を使用する方が MD5 より安全性が高くなります。(MD5 は、Oracle Database 21c 以降では非推奨となっています)。

  Oracle Database 21c 以降、古い暗号化およびハッシュ・アルゴリズムは非推奨になりました。非推奨になったアルゴリズムとしては、MD4、MD5、DES、3DES および RC4 関連のアルゴリズムがあります。古い安全性の低い暗号化アルゴリズムの削除により、これらの API が誤って使用されるのを防ぎます。セキュリティ要件を満たすために、AES などの最新の暗号化アルゴリズムを使用することをお薦めします。

  Oracle Database 21c 以降、古い暗号化およびハッシュ・アルゴリズムは非推奨になりました。

  この非推奨の結果、非推奨のアルゴリズムの使用が指定されているかどうかを確認するために、ネットワーク暗号化構成を確認することをお薦めします。いずれかが見つかった場合は、AES などの最新の暗号の使用に切り替えます。詳細は、「ネイティブ・ネットワーク暗号化のセキュリティの向上」を参照してください。
- SHA-2 を使用する方が SHA-1 より安全性が高くなります。
- キー付き MD5 は、無防備ではありません。

表17-1に、DBMS_CRYPTOパッケージ機能の概要を示します。

表17-1 DBMS_CRYPTOパッケージ機能の概要

機能	DBMS_CRYPTOでサポートされている機能
ブロック暗号連鎖モード	CBC、CFB、ECB、OFB
暗号化アルゴリズム	AES
暗号化ハッシュ・アルゴリズム	SHA-1、SHA-2、HASH_SH256、HASH_SH384、HASH_SH512
暗号化擬似乱数ジェネレータ	RAW、NUMBER、BINARY_INTEGER
データベース型	RAW、CLOB、BLOB
キー・ハッシュ(MAC)・アルゴリズム	HMAC_MD5、HMAC_SH1、HMAC_SH256、HMAC_SH384、HMAC_SH512
パディング形式	PKCS5、複数ゼロ

表17-2に、サポートされているSHAハッシュ関数を示します。これらの関数の多くはRSA環境で使用できます。

表17-2 SHAハッシュ・アルゴリズム

ハッシュ・アルゴリズム	説明
SIGN_RSA_PKCS1_OAEP_SHA256	Public Key Cryptographic Standard、SHA 256 ビット・ハッシュ関数および OAEP パディングを使用する RSA
SIGN_SHA1_ECDSA	楕円曲線デジタル署名アルゴリズムを使用する SHA ハッシュ関数
SIGN_SHA1_RSA	RSA を使用する SHA ハッシュ関数
SIGN_SHA1_RSA_X931	RSA および X931 パディングを使用する SHA ハッシュ関数
SIGN_SHA224_ECDSA	楕円曲線デジタル署名アルゴリズムを使用する SHA 224 ビット・ハッシュ関数
SIGN_SHA224_RSA	RSA を使用する SHA 224 ビット・ハッシュ関数
SIGN_SHA256_ECDSA	楕円曲線デジタル署名アルゴリズムを使用する SHA 256 ビット・ハッシュ関数
SIGN_SHA256_RSA	RSA を使用する SHA 256 ビット・ハッシュ関数

ハッシュ・アルゴリズム	説明
SIGN_SHA256_RSA_X931	RSA および X931 パディングを使用する SHA 256 ビット・ハッシュ関数
SIGN_SHA384_ECDSA	楕円曲線デジタル署名アルゴリズムを使用する SHA 384 ビット・ハッシュ関数
SIGN_SHA384_RSA	RSA を使用する SHA 384 ビット・ハッシュ関数
SIGN_SHA384_RSA_X931	RSA および X931 パディングを使用する SHA 384 ビット・ハッシュ関数
SIGN_SHA512_ECDSA	楕円曲線デジタル署名アルゴリズムを使用する SHA 512 ビット・ハッシュ関数
SIGN_SHA512_RSA	RSA を使用する SHA 384 ビット・ハッシュ関数
SIGN_SHA512_RSA_X931	RSA および X931 パディングを使用する SHA 384 ビット・ハッシュ関数

表17-3に、サポートされる暗号化および復号化アルゴリズムを示します。

表17-3 暗号化および復号化アルゴリズム

アルゴリズム	説明
PKENCRYPT_ECDH	楕円曲線 Diffie Hellman
PKENCRYPT_RSA_PKCS1_OAEP	PKCS1 および OAEP パディングを使用する RSA 公開キー暗号システム

表17-4に、サポートされているその他のアルゴリズムを示します。

表17-4 その他のアルゴリズム

アルゴリズム	説明
KEY_TYPE_RSA	RSA キーのタイプ
SIGN_ECDSA	楕円曲線デジタル署名アルゴリズム

DBMS_CRYPTOは、新しいシステムと既存のシステムの両方に対応する広範囲のアルゴリズムをサポートしています。3DES_2KEYおよびMD4は下位互換性を維持するために提供されていますが、3DES、AESまたはSHA-1を使用するとセキュリティをより強化できます。3DES_2KEYおよびMD4の使用はお薦めしません。

DBMS_CRYPTOパッケージには、比較の際に便利な暗号チェックサム機能(MD5)と、安全な乱数を生成する機能(RANDOMBYTES関数)が含まれています。安全な乱数生成は暗号化の重要な部分です。予測可能なキーは簡単に推定されるキーであり、キーを簡単に推定できることによってデータが簡単に復号化される可能性があります。ほとんどの暗号解読が、総当たり解析(可能性のあるすべてのキーを繰り返す)によってではなく、脆弱なキーや適切に格納されていないキーを見つけ出すことによって行われています。

ノート:

DBMS_RANDOM は暗号化キーの生成には不適切なため、使用しないでください。

キーの管理はプログラムによって実行されます。つまり、アプリケーション(またはファンクションのコール側)が、暗号化キーを提供する必要があります。これは、アプリケーション開発者がキーを安全に格納し、取得する方法を検討する必要があることを意味します。様々なキー管理方法の相対的なメリットとデメリットについては、後続の項で説明します。DESアルゴリズム自体の有効なキーの長さは56ビットです。

親トピック: 手動によるデータ暗号化

17.4 DBMS_CRYPTOパッケージを使用した非対称キー操作

DBMS_CRYPTOパッケージには、暗号化、復号化、署名および検証のための非対称キー操作を実行できる4つのファンクションが用意されています。

非対称キー操作(公開キー暗号化とも呼ばれる)では、公開キーと秘密キーを使用してメッセージを暗号化および復号化し、不正アクセスから保護します。

非対称キー操作ファンクションは次のとおりです。

- PKDECRYPTは、キー・アルゴリズムおよび暗号化アルゴリズムで支援される秘密キーを使用してRAWデータを復号化します。
- PKENCRYPTは、キー・アルゴリズムおよび暗号化アルゴリズムで支援される公開キーを使用してRAWデータを暗号化します。
- SIGNは、キー・アルゴリズムおよび署名アルゴリズムで支援された秘密キーを使用してRAWデータに署名します。
- VERIFYは、署名、キー・アルゴリズムおよび署名アルゴリズムで支援される公開キーを使用してRAWデータを検証します。

関連トピック

- Oracle Database PL/SQLパッケージ・プロシージャおよびタイプ・リファレンス

親トピック: 手動によるデータ暗号化

17.5 OFBモードで暗号化された暗号文のOracle Databaseリリース11gでの使用

Oracle Databaseリリース11gでは、出力フィードバック(OFB)を使用するように暗号文が構成されても、かわりに電子コードブック(ECB)モードが使用されました。

Oracle Databaseリリース11gで、DBMS_CRYPTO.CHAIN_OFB暗号ブロック連鎖修飾子を設定して、出力フィードバック(OFB)モードを使用するように暗号文の暗号化を構成すると、Oracle Bug 13001552のため、その構成で電子コードブック(ECB)モードが誤って使用されました。この不具合はOracle Databaseリリース12cで修正されています。したがって、Oracle Databaseリリース11gからリリース12cにアップグレードした後、リリース11gでOFBノードを使用して暗号化された暗号文は、Oracle Databaseリリース12cでは修正されたOFBモードで正しく復号化されません。

この問題を解決するには:

1. DBMS_CRYPTO PL/SQLパッケージに対するEXECUTE権限があるユーザーとして、データベースにログインします。
2. DBMS_CRYPTO.CHAIN_ECBブロック暗号連鎖修飾子を使用して、暗号文を復号化します。

次の例のdbmscrypto11.sqlは、Oracle Databaseリリース11gでの誤った動作を示します。

```
dbmscrypto11.sql:
set serveroutput on
declare
  l_mod_ofb pls_integer;
  l_mod_ecb pls_integer;
  v_key raw(32);
  v_iv  raw(16);
  v_test_in raw(16);
  v_ciphertext raw(16);
  v_test_out_ECB raw(16);
  v_test_out_OFB raw(16);
begin
  l_mod_ofb := dbms_crypto.ENCRYPT_AES256
       + dbms_crypto.CHAIN_OFB
       + DBMS_CRYPTO.PAD_NONE ;
  l_mod_ecb := dbms_crypto.ENCRYPT_AES256
       + dbms_crypto.CHAIN_ECB
       + DBMS_CRYPTO.PAD_NONE ;
  v_key := hextoraw
  ('603deb1015ca71be2b73aef0857d77811f352c073b6108d72d9810a30914dff4');
  v_iv :=   hextoraw('000102030405060708090A0B0C0D0E0F');
  v_test_in := hextoraw('6bc1bee22e409f96e93d7e117393172a');
  v_ciphertext := dbms_crypto.encrypt(src => v_test_in,
                                TYP => l_mod_ofb,
                                key => v_key,
                                iv => v_iv);
  v_test_out_ECB := dbms_crypto.decrypt(src => v_ciphertext,
                                TYP => l_mod_ecb,
                                key => v_key,
                                iv => v_iv);
  v_test_out_OFB := dbms_crypto.decrypt(src => v_ciphertext,
                                TYP => l_mod_ofb,
                                key => v_key,
                                iv => v_iv);
  dbms_output.put_line
  ('Input plaintext                      : '||rawtohex(v_test_in));
  dbms_output.put_line
  ('11g: Ciphertext (encrypt in OFB mode): '||rawtohex(v_ciphertext));
  dbms_output.put_line
  ('11g: Output of decrypt in ECB mode   : '||rawtohex(v_test_out_ECB));
  dbms_output.put_line
  ('11g: Output of decrypt in OFB mode   : '||rawtohex(v_test_out_OFB));
end;
/
```

生成される出力は次のようになります。

```
SQL> @dbmscrypto11.sql
Input plaintext                      : 6BC1BEE22E409F96E93D7E117393172A
11g: Ciphertext (encrypt in OFB mode): F3EED1BDB5D2A03C064B5A7E3DB181F8
11g: Output of decrypt in ECB mode   : 6BC1BEE22E409F96E93D7E117393172A
11g: Output of decrypt in OFB mode   : 6BC1BEE22E409F96E93D7E117393172A
```

この出力は、Oracle Databaseリリース11gではOFBモードが誤ってECBモードになるため、OFBまたはECBモードのどちらで復号化しても正しいプレーンテキストになることを示しています。

次の例のdbmscrypto12from11.sqlは、Oracle Databaseリリース11gからリリース12cにアップグレードした後、リリース11gのOFBモードで暗号化した暗号文を適切に復号化するには、OFBモードではなく、ECBモードを使用する必要があるこ

とを示します。

```
dbmscrypto12from11.sql:
set serveroutput on
declare
  l_mod_ofb pls_integer;
  l_mod_ecb pls_integer;
  v_key raw(32);
  v_iv  raw(16);
  v_test_in raw(16);
  v_ciphertext11 raw(16);
  v_test_out_ECB raw(16);
  v_test_out_OFB raw(16);
begin
  l_mod_ofb := dbms_crypto.ENCRYPT_AES256
       + dbms_crypto.CHAIN_OFB
       + DBMS_CRYPTO.PAD_NONE ;
  l_mod_ecb := dbms_crypto.ENCRYPT_AES256
       + dbms_crypto.CHAIN_ECB
       + DBMS_CRYPTO.PAD_NONE ;
  v_key := hextoraw
  ('603deb1015ca71be2b73aef0857d77811f352c073b6108d72d9810a30914dff4');
  v_iv :=   hextoraw('000102030405060708090A0B0C0D0E0F');
  v_test_in := hextoraw('6bc1bee22e409f96e93d7e117393172a');
  v_ciphertext11 := hextoraw('F3EED1BDB5D2A03C064B5A7E3DB181F8');
  v_test_out_ECB := dbms_crypto.decrypt(src => v_ciphertext11,
                                TYP => l_mod_ecb,
                                key => v_key,
                                iv => v_iv);
  v_test_out_OFB := dbms_crypto.decrypt(src => v_ciphertext11,
                                TYP => l_mod_ofb,
                                key => v_key,
                                iv => v_iv);
  dbms_output.put_line
  ('Input plaintext (to 11g)             : '||rawtohex(v_test_in));
  dbms_output.put_line
  ('11g: Ciphertext (encrypt in OFB mode): '||rawtohex(v_ciphertext11));
  dbms_output.put_line
  ('12c: Output of decrypt in ECB mode   : '||rawtohex(v_test_out_ECB));
  dbms_output.put_line
  ('12c: Output of decrypt in OFB mode   : '||rawtohex(v_test_out_OFB));
end;
/
```

生成される出力は次のようになります。

```
SQL> @dbmscrypto12from11.sql
Input plaintext (to 11g)             : 6BC1BEE22E409F96E93D7E117393172A
11g: Ciphertext (encrypt in OFB mode): F3EED1BDB5D2A03C064B5A7E3DB181F8
12c: Output of decrypt in ECB mode   : 6BC1BEE22E409F96E93D7E117393172A
12c: Output of decrypt in OFB mode   : 4451EBE041EB29E191BBA0E9D67FAEB2
```

Oracle Databaseリリース11gからリリース12cへのアップグレードを予定している場合は、復号化操作でECBモードを使用するように、OFBモードが指定されたすべてのスクリプトを編集してください。この方法によって、スクリプトはリリース11gとリリース12c以降の両方で使用できるため、ビジネスの継続性を確保できます。

親トピック: 手動によるデータ暗号化

17.6 データの暗号化APIの使用例

データ暗号化APIの使用例として、DBMS_CRYPTO.SQLプロシージャの使用、AES 256ビット・データの暗号化およびBLOBデータの暗号化などがあります。

- データ暗号化プロシージャの例
  DBMS_CRYPTO.SQL PL/SQLプログラムを使用してデータを暗号化できます。
- AES 256ビット・データ暗号化および復号化プロシージャの例
  PL/SQLブロックを使用して、事前定義された変数を暗号化および復号化できます。
- BLOBデータの暗号化および復号化プロシージャの例
  BLOBデータを暗号化できます。

親トピック: 手動によるデータ暗号化

17.6.1 データ暗号化プロシージャの例

DBMS_CRYPTO.SQL PL/SQLプログラムを使用してデータを暗号化できます。

このサンプル・コードでは、次の処理を実行します。

- 文字列(VARCHAR2型)をRAWデータ型に変換した後、DESを使用して暗号化します。

  DBMS_CRYPTOパッケージの暗号化および復号化ファンクションとプロシージャはRAWデータ型にのみ機能するため、このステップが必要となります。
- SHA-1アルゴリズムを使用した160ビット・ハッシュの作成方法を示します。
- MD5アルゴリズムを使用したキー依存の一方向ハッシュであるMACの計算方法を示します。

次に、DBMS_CRYPTO.SQLプロシージャを示します。

```
DECLARE
    input_string     VARCHAR2(16) := 'tigertigertigert';
    raw_input        RAW(128) :=
UTL_RAW.CAST_TO_RAW(CONVERT(input_string,'AL32UTF8','US7ASCII'));
    key_string       VARCHAR2(8)  := 'scottsco';
    raw_key          RAW(128) :=
UTL_RAW.CAST_TO_RAW(CONVERT(key_string,'AL32UTF8','US7ASCII'));
    encrypted_raw    RAW(2048);
    encrypted_string VARCHAR2(2048);
    decrypted_raw    RAW(2048);
    decrypted_string VARCHAR2(2048);
-- Begin testing Encryption:
BEGIN
    dbms_output.put_line('> Input String                    : ' ||
    CONVERT(UTL_RAW.CAST_TO_VARCHAR2(raw_input),'US7ASCII','AL32UTF8'));
    dbms_output.put_line('> ========= BEGIN TEST Encrypt =========');
    encrypted_raw := dbms_crypto.Encrypt(
        src => raw_input,
        typ => DBMS_CRYPTO.DES_CBC_PKCS5,
        key => raw_key);
        dbms_output.put_line('> Encrypted hex value             : ' ||
        rawtohex(UTL_RAW.CAST_TO_RAW(encrypted_raw)));
decrypted_raw := dbms_crypto.Decrypt(
        src => encrypted_raw,
        typ => DBMS_CRYPTO.DES_CBC_PKCS5,
        key => raw_key);
    decrypted_string :=
    CONVERT(UTL_RAW.CAST_TO_VARCHAR2(decrypted_raw),'US7ASCII','AL32UTF8');
dbms_output.put_line('> Decrypted string output          : ' ||
        decrypted_string);
if input_string = decrypted_string THEN
    dbms_output.put_line('> String DES Encyption and Decryption successful');
END if;
dbms_output.put_line('');
dbms_output.put_line('> ========= BEGIN TEST Hash =========');
    encrypted_raw := dbms_crypto.Hash(
```

```
        src => raw_input,
        typ => DBMS_CRYPTO.HASH_SH1);
dbms_output.put_line('> Hash value of input string       : ' ||
        rawtohex(UTL_RAW.CAST_TO_RAW(encrypted_raw)));
dbms_output.put_line('> ========= BEGIN TEST Mac =========');
    encrypted_raw := dbms_crypto.Mac(
        src => raw_input,
        typ => DBMS_CRYPTO.HMAC_MD5,
        key => raw_key);
dbms_output.put_line('> Message Authentication Code      : ' ||
        rawtohex(UTL_RAW.CAST_TO_RAW(encrypted_raw)));
dbms_output.put_line('');
dbms_output.put_line('> End of DBMS_CRYPTO tests  ');
END;
/
```

親トピック: データの暗号化APIの使用例

17.6.2 AES 256ビット・データ暗号化および復号化プロシージャの例

PL/SQLブロックを使用して、事前定義された変数を暗号化または復号できます。

次の例では、事前定義の変数`input_string`が、Cipher Block ChainingとPKCS #5パディングを使用するAES 256ビット・アルゴリズムを使用しています。

```
declare
   input_string       VARCHAR2 (200) := 'Secret Message';
   output_string      VARCHAR2 (200);
   encrypted_raw      RAW (2000);             -- stores encrypted binary text
   decrypted_raw      RAW (2000);             -- stores decrypted binary text
   num_key_bytes      NUMBER := 256/8;        -- key length 256 bits (32 bytes)
   key_bytes_raw      RAW (32);               -- stores 256-bit encryption key
   encryption_type    PLS_INTEGER :=          -- total encryption type
                         DBMS_CRYPTO.ENCRYPT_AES256
                       + DBMS_CRYPTO.CHAIN_CBC
                       + DBMS_CRYPTO.PAD_PKCS5;
begin
   DBMS_OUTPUT.PUT_LINE ('Original string: ' || input_string);
   key_bytes_raw := DBMS_CRYPTO.RANDOMBYTES (num_key_bytes);
   encrypted_raw := DBMS_CRYPTO.ENCRYPT
      (
         src => UTL_I18N.STRING_TO_RAW (input_string, 'AL32UTF8'),
         typ => encryption_type,
         key => key_bytes_raw
      );
    -- The encrypted value in the encrypted_raw variable can be used here:
   decrypted_raw := DBMS_CRYPTO.DECRYPT
      (
         src => encrypted_raw,
         typ => encryption_type,
         key => key_bytes_raw
      );
   output_string := UTL_I18N.RAW_TO_CHAR (decrypted_raw, 'AL32UTF8');
   DBMS_OUTPUT.PUT_LINE ('Decrypted string: ' || output_string);
end;
```

親トピック: データの暗号化APIの使用例

17.6.3 BLOBデータの暗号化および復号化プロシージャの例

BLOBデータを暗号化できます。

次に、BLOBデータを暗号化および復号化するためのPL/SQLプログラム(`blob_test.sql`)のサンプルを示します。このサン

プル・コードは次の操作を実行し、ステップごとに進捗状況(または問題)を出力します。

- BLOB列の表を作成します。
- RAWの値をその表に挿入します。
- RAWデータを暗号化します。
- 暗号化されたデータを復号化します。

`blob_test.sql`プロシージャは次のとおりです。

```
-- 1. Create a table for BLOB column:
create table table_lob (id number, loc blob);
-- 2. Insert 3 empty lobs for src/enc/dec:
insert into table_lob values (1, EMPTY_BLOB());
insert into table_lob values (2, EMPTY_BLOB());
insert into table_lob values (3, EMPTY_BLOB());
set echo on
set serveroutput on
declare
    srcdata    RAW(1000);
    srcblob    BLOB;
    encrypblob BLOB;
    encrypraw  RAW(1000);
    encrawlen  BINARY_INTEGER;
    decrypblob BLOB;
    decrypraw  RAW(1000);
    decrawlen  BINARY_INTEGER;

    leng       INTEGER;
begin

    -- RAW input data 16 bytes
    srcdata := hextoraw('6D6D6D6D6D6D6D6D6D6D6D6D6D6D6D6D');

    dbms_output.put_line('---');
    dbms_output.put_line('input is ' || srcdata);
    dbms_output.put_line('---');

    -- select empty lob locators for src/enc/dec
    select loc into srcblob from table_lob where id = 1;
    select loc into encrypblob from table_lob where id = 2;
    select loc into decrypblob from table_lob where id = 3;

    dbms_output.put_line('Created Empty LOBS');
    dbms_output.put_line('---');

    leng := DBMS_LOB.GETLENGTH(srcblob);
    IF leng IS NULL THEN
        dbms_output.put_line('Source BLOB Len NULL ');
    ELSE
        dbms_output.put_line('Source BLOB Len ' || leng);
    END IF;

    leng := DBMS_LOB.GETLENGTH(encrypblob);
    IF leng IS NULL THEN
        dbms_output.put_line('Encrypt BLOB Len NULL ');
    ELSE
        dbms_output.put_line('Encrypt BLOB Len ' || leng);
    END IF;

    leng := DBMS_LOB.GETLENGTH(decrypblob);
    IF leng IS NULL THEN
        dbms_output.put_line('Decrypt  BLOB Len NULL ');
    ELSE
```

```
        dbms_output.put_line('Decrypt BLOB Len ' || leng);
    END IF;

    -- 3. Write source raw data into blob:
    DBMS_LOB.OPEN (srcblob, DBMS_LOB.lob_readwrite);
    DBMS_LOB.WRITEAPPEND (srcblob, 16, srcdata);
    DBMS_LOB.CLOSE (srcblob);

    dbms_output.put_line('Source raw data written to source blob');
    dbms_output.put_line('---');

    leng := DBMS_LOB.GETLENGTH(srcblob);
    IF leng IS NULL THEN
        dbms_output.put_line('source BLOB Len NULL ');
    ELSE
        dbms_output.put_line('Source BLOB Len ' || leng);
    END IF;

    /*
    * Procedure Encrypt
    * Arguments: srcblob -> Source BLOB
    *            encrypblob -> Output BLOB for encrypted data
    *            DBMS_CRYPTO.AES_CBC_PKCS5 -> Algo : AES
    *                                         Chaining : CBC
    *                                         Padding : PKCS5
    *            256 bit key for AES passed as RAW
    *                ->
    hextoraw('000102030405060708090A0B0C0D0E0F101112131415161718191A1B1C1D1E1F')
    *            IV (Initialization Vector) for AES algo passed as RAW
    *                -> hextoraw('00000000000000000000000000000000')
    */

    DBMS_CRYPTO.Encrypt(encrypblob,
                srcblob,
                DBMS_CRYPTO.AES_CBC_PKCS5,
                hextoraw
('000102030405060708090A0B0C0D0E0F101112131415161718191A1B1C1D1E1F'),
                hextoraw('00000000000000000000000000000000'));


    dbms_output.put_line('Encryption Done');
    dbms_output.put_line('---');

    leng := DBMS_LOB.GETLENGTH(encrypblob);
    IF leng IS NULL THEN
        dbms_output.put_line('Encrypt BLOB Len NULL');
    ELSE
        dbms_output.put_line('Encrypt BLOB Len ' || leng);
    END IF;

    -- 4. Read encrypblob to a raw:
    encrawlen := 999;

    DBMS_LOB.OPEN (encrypblob, DBMS_LOB.lob_readwrite);
    DBMS_LOB.READ (encrypblob, encrawlen, 1, encrypraw);
    DBMS_LOB.CLOSE (encrypblob);

    dbms_output.put_line('Read encrypt blob to a raw');
    dbms_output.put_line('---');

    dbms_output.put_line('Encrypted data is (256 bit key) ' || encrypraw);
    dbms_output.put_line('---');

    /*
    * Procedure Decrypt
    * Arguments: encrypblob -> Encrypted BLOB to decrypt
    *            decrypblob -> Output BLOB for decrypted data in RAW
```

```
    *            DBMS_CRYPTO.AES_CBC_PKCS5 -> Algo : AES
    *                                         Chaining : CBC
    *                                         Padding : PKCS5
    *            256 bit key for AES passed as RAW (same as used during Encrypt)
    *                ->
    hextoraw('000102030405060708090A0B0C0D0E0F101112131415161718191A1B1C1D1E1F')
    *            IV (Initialization Vector) for AES algo passed as RAW (same as
                 used during Encrypt)
    *                -> hextoraw('00000000000000000000000000000000')
    */

    DBMS_CRYPTO.Decrypt(decrypblob,
                encrypblob,
                DBMS_CRYPTO.AES_CBC_PKCS5,
                hextoraw
           ('000102030405060708090A0B0C0D0E0F101112131415161718191A1B1C1D1E1F'),
                hextoraw('00000000000000000000000000000000'));

    leng := DBMS_LOB.GETLENGTH(decrypblob);
    IF leng IS NULL THEN
        dbms_output.put_line('Decrypt BLOB Len NULL');
    ELSE
        dbms_output.put_line('Decrypt BLOB Len ' || leng);
    END IF;

    -- Read decrypblob to a raw
    decrawlen := 999;

    DBMS_LOB.OPEN (decrypblob, DBMS_LOB.lob_readwrite);
    DBMS_LOB.READ (decrypblob, decrawlen, 1, decrypraw);
    DBMS_LOB.CLOSE (decrypblob);

    dbms_output.put_line('Decrypted data is (256 bit key) ' || decrypraw);
    dbms_output.put_line('---');

    DBMS_LOB.OPEN (srcblob, DBMS_LOB.lob_readwrite);
    DBMS_LOB.TRIM (srcblob, 0);
    DBMS_LOB.CLOSE (srcblob);

    DBMS_LOB.OPEN (encrypblob, DBMS_LOB.lob_readwrite);
    DBMS_LOB.TRIM (encrypblob, 0);
    DBMS_LOB.CLOSE (encrypblob);

    DBMS_LOB.OPEN (decrypblob, DBMS_LOB.lob_readwrite);
    DBMS_LOB.TRIM (decrypblob, 0);
    DBMS_LOB.CLOSE (decrypblob);

end;
/
truncate table table_lob;
drop table table_lob;
```

親トピック: データの暗号化APIの使用例

17.7 暗号化データのデータ・ディクショナリ・ビュー

Oracle Databaseには、暗号化されたデータに関する情報を検索できるデータ・ディクショナリ・ビューが用意されています。

表17-5 暗号化データに関する情報を表示するデータ・ディクショナリ・ビュー

ビュー	説明

ビュー	説明
ALL_ENCRYPTED_COLUMNS	ユーザーがアクセスできるすべての表にあるすべての暗号化列の暗号化アルゴリズム情報が表示されます。
DBA_ENCRYPTED_COLUMNS	データベースにあるすべての暗号化列の暗号化アルゴリズム情報が表示されます。
USER_ENCRYPTED_COLUMNS	ユーザー・スキーマのすべての表にあるすべての暗号化列の暗号化アルゴリズム情報が表示されます。
V$ENCRYPTED_TABLESPACES	暗号化されている現在のプラガブル・データベース(PDB)表領域に関する情報が表示されます。
V$ENCRYPTION_WALLET	透過的データ暗号化のウォレットのステータスおよびウォレットの場所に関する情報が表示され、現在の PDB のみに適用されます。
V$RMAN_ENCRYPTION_ALGORITHMS	現在の PDB でサポートされている暗号化アルゴリズムが表示されます。

関連トピック

- Oracle Databaseリファレンス

親トピック: 手動によるデータ暗号化

第IV部 ネットワーク上のデータの保護

第IV部では、ネットワーク上のデータの保護方法について説明します。

- Oracle Databaseのネイティブ・ネットワーク暗号化とデータ整合性の構成
  サーバーとクライアントの両方について、Oracle Net Services固有のデータ暗号化およびデータ整合性を構成できます。
- シンJDBCクライアント・ネットワークの構成
  Oracle Databaseのネイティブ・ネットワーク暗号化および厳密認証を使用すると、シンJava Database Connectivity (JDBC)クライアントは、Oracleデータベースに安全に接続できます。

18 Oracle Databaseのネイティブ・ネットワーク暗号化とデータ整合性の構成

サーバーとクライアントの両方について、Oracle Net Services固有のデータ暗号化およびデータ整合性を構成できます。

- Oracle Databaseのネイティブ・ネットワーク暗号化とデータ整合性について
  Oracle Databaseでは、ネットワークで送信されるデータを暗号化できます。
- Oracle Databaseのネイティブ・ネットワーク暗号化のデータ整合性
  ネットワーク・データを暗号化すると、ネットワーク上で転送される平文データを不正なユーザーが閲覧できなくなり、データのプライバシが保護されます。
- ネイティブ・ネットワーク暗号化のセキュリティの向上
  Oracleでは、Oracle Databaseサーバーとクライアントの両方のネイティブ・ネットワーク暗号化のセキュリティを強化するパッチを提供しています。
- データの整合性アルゴリズムのサポート
  データ整合性アルゴリズムは、サード・パーティ攻撃およびメッセージ・リプレイ攻撃から保護します。SHA-2をお薦めしますが、SHA-1 (非推奨)およびMD5は下位互換性のために維持されています。
- Diffie-Hellmanベースのキー交換
  Diffie-Hellmanキー交換アルゴリズムを使用して、マルチユーザー環境でデータを保護できます。
- データの暗号化および整合性の構成
  Oracle Database固有のOracle Net Servicesによる暗号化および整合性は、Oracle Net Servicesがインストール済であることを前提としています。

親トピック: ネットワーク上のデータの保護

18.1 Oracle Databaseのネイティブ・ネットワーク暗号化とデータ整合性について

Oracle Databaseでは、ネットワークで送信されるデータを暗号化できます。

- Oracle Databaseのネイティブ・ネットワーク暗号化と整合性の仕組み
  Oracle Databaseは、ネットワーク間を移動するデータが保護されるように、ネイティブ・データ・ネットワークの暗号化および整合性を提供します。
- Advanced Encryption Standard
  Oracle Databaseでは、米国連邦情報処理標準(FIPS)暗号化アルゴリズムであるAdvanced Encryption Standard (AES)がサポートされています。
- Triple-DES暗号化
  Triple-DES (3DES)暗号化は、DESアルゴリズムにメッセージ・データを3回渡して暗号化します。
- ネイティブ・ネットワーク暗号化とTransport Layer Securityの間の選択
  Oracleには、ネットワーク上のデータを暗号化する方法として、ネイティブ・ネットワーク暗号化とTransport Layer Security (TLS)の2つがあります。

親トピック: Oracle Databaseのネイティブ・ネットワーク暗号化とデータ整合性の構成

18.1.1 Oracle Databaseのネイティブ・ネットワーク暗号化と整合性の仕組み

Oracle Databaseは、ネットワーク間を移動するデータが保護されるように、ネイティブ・データ・ネットワークの暗号化および整合性を提供します。

セキュアな暗号システムの目的は、キーに基づいて、平文データを解読不能な暗号文に変換することです。正しいキーがなければ、暗号文を元の平文に変換することはきわめて困難(計算上は不可能)です。

対称暗号システムでは、同じデータの暗号化と復号化の両方に同じキーを使用します。Oracle Databaseは、Advanced Encryption Standard (AES)の対称暗号システムを提供して、Oracle Net Servicesトラフィックの機密保護を図ります。

親トピック: Oracle Databaseのネイティブ・ネットワーク暗号化とデータ整合性について

18.1.2 Advanced Encryption Standard

Oracle Databaseでは、米国連邦情報処理標準(FIPS)暗号化アルゴリズムであるAdvanced Encryption Standard (AES)がサポートされています。

AESは、ネットワーク上で機密データを保護するために、あらゆる米国政府組織および企業で使用できます。この暗号化アルゴリズムでは、128ビット、192ビットおよび256ビットの3つの標準のキーの長さが定義されています。いずれのバージョンも、外部暗号ブロック連鎖(CBC)モードで稼働します。暗号化メソッドの1つ。先行するすべてのブロックに依存する暗号ブロックの暗号化を行い、ブロック再生攻撃からデータを保護します。無許可の復号化が段階的に困難になるように設計されています。
Oracle Databaseでは、外部暗号ブロック連鎖が使用されています。これは、内部暗号ブロックよりも安全性が高く、実質的なパフォーマンスの低下を伴わないためです。

ノート:

AESアルゴリズムが改善されました。より強力なアルゴリズムを使用するようにOracle Database環境を移行するには、My Oracle Supportノート2118136.2で説明されているパッチをダウンロードしてインストールします。

親トピック: Oracle Databaseのネイティブ・ネットワーク暗号化とデータ整合性について

18.1.3 Triple-DES暗号化

Triple-DES (3DES)暗号化は、DESアルゴリズムにメッセージ・データを3回渡して暗号化します。

ノート:

このリリースでは、DES、DES40、3DES112 および 3DES168 アルゴリズムは非推奨です。より強力なアルゴリズムを使用するように Oracle Database 環境を移行するには、My Oracle Support ノート 2118136.2 で説明されているパッチをダウンロードしてインストールします。

3DESは高度なメッセージ・セキュリティを提供しますが、パフォーマンスの低下を伴います。パフォーマンス低下の度合いは、暗号化を実行するプロセッサの速度によって異なります。3DESは、標準のDESアルゴリズムに比べ、データ・ブロックの暗号化に通常3倍の時間を要します。

3DESには2つのキーを使用するバージョンと3つのキーを使用するバージョンがあり、それぞれ有効なキーの長さは112ビットと168ビットです。いずれのバージョンも外部暗号ブロック連鎖(CBC)モードで稼働します。

Oracle DatabaseおよびSecure Network Servicesで利用可能なDES40アルゴリズムは、秘密キーを事前処理することによって有効キー・ビットを40とするDESの一種です。米国の輸出法が厳しかったときに米国およびカナダ以外の顧客を対象にDESベースの暗号化を提供する目的で設計されました。DES40、DESおよび3DESはすべて輸出のために使用できます。DES40は、海外顧客向けに下位互換性を維持するために引き続きサポートされています。

親トピック: Oracle Databaseのネイティブ・ネットワーク暗号化とデータ整合性について

18.1.4 ネイティブ・ネットワーク暗号化とTransport Layer Securityの間の選択

Oracleには、ネットワーク上のデータを暗号化する方法として、ネイティブ・ネットワーク暗号化とTransport Layer Security (TLS)の2つがあります。

どちらの方法にもメリットとデメリットがあります。

表18-1 ネイティブ・ネットワーク暗号化とTransport Layer Securityの比較

-	ネイティブ・ネットワーク暗号化	Transport Layer Security
メリット	● `sqlnet.ora` 構成ファイル内のパラメータで構成されます。 ● ほとんどの場合、クライアント構成の変更は必要ありません。 ● 証明書は必要ありません。 ● ネイティブ・ネットワーク暗号化をサポートしていないクライアントは、非互換性が緩和されている間、暗号化されていない接続に戻ることができます。	● 移動中のデータを暗号化するための業界標準です。 ● 第三者攻撃を防ぐために、サーバー接続に対する否認防止を提供します。 ● データベース・ユーザー認証に使用できます。
デメリット	● 非標準の Oracle 独自の実装を使用します。 ● サーバー接続の否認防止を提供しません(つまり、第三者攻撃に対する保護なし)。	● クライアントおよびサーバーの変更が必要です。 ● 証明書はサーバーでは必要であり、クライアントではオプションです。ただし、クライアントには、サーバーの証明書を発行した認証局のトラステッド・ルート証明書が必要です。 ● 証明書は最終的に失効します。

親トピック: Oracle Databaseのネイティブ・ネットワーク暗号化とデータ整合性について

18.2 Oracle Databaseのネイティブ・ネットワーク暗号化のデータ整合性

ネットワーク・データを暗号化すると、ネットワーク上で転送される平文データを不正なユーザーが閲覧できなくなり、データのプライバシが保護されます。

また、Oracle Databaseでは、2つの形態の攻撃からデータを保護できます。

表18-2に、これらの攻撃についての情報を示します。

表18-2 ネットワーク攻撃の2つの形態

攻撃形態	説明
データ変更攻撃	データ変更攻撃とは、不正なユーザーが転送中のデータを傍受し、データを変更して再転送することです。たとえば、銀行への$100 の預入れを傍受して、金額を$10,000 に変更し、その水増し金額を再転送することをデータ変更攻撃といいます。
再生攻撃	再生攻撃とは、有効なデータ全体を反復的に再送することです。たとえば、銀行からの$100 の払戻しを傍受し、その払戻しを 10 回再転送して、最終的に$1,000 受け取るといった攻撃です。

親トピック: Oracle Databaseのネイティブ・ネットワーク暗号化とデータ整合性の構成

18.3 ネイティブ・ネットワーク暗号化のセキュリティの向上

Oracleでは、Oracle Databaseサーバーとクライアントの両方のネイティブ・ネットワーク暗号化のセキュリティを強化するパッチを提供しています。

- ネイティブ・ネットワーク暗号化のセキュリティの向上について
  Oracleのこのパッチは、暗号化アルゴリズムおよびチェックサム・アルゴリズムを更新し、脆弱な暗号化アルゴリズムおよびチェックサム・アルゴリズムを非推奨にします。
- ネイティブ・ネットワーク暗号化へのセキュリティ改善更新の適用
  Oracle Databaseサーバーおよびクライアントにパッチを適用することに加え、サーバーおよびクライアントの`sqlnet.ora`のパラメータを設定する必要があります。

親トピック: Oracle Databaseのネイティブ・ネットワーク暗号化とデータ整合性の構成

18.3.1 ネイティブ・ネットワーク暗号化のセキュリティの向上について

Oracleのこのパッチは、暗号化アルゴリズムおよびチェックサム・アルゴリズムを更新し、脆弱な暗号化アルゴリズムおよびチェックサム・アルゴリズムを非推奨にします。

このパッチは、My Oracle Supportノート2118136.2からダウンロードでき、サーバーとクライアントとの接続を強化し、ネイティブ・ネットワーク暗号化アルゴリズムおよびチェックサム・アルゴリズムの脆弱性を修正します。これにより、安全性の低い古い暗号化アルゴリズムおよびチェックサム・アルゴリズムの無効化が容易になる2つのパラメータが追加されます。このパッチをOracle Databaseサーバーおよびクライアントに適用することをお薦めします。

このパッチは、Oracle Databaseリリース11.2以降に適用されます。このパッチは、スタンドアロン、マルチテナント、プライマリ・スタンバイ、Oracle Real Application Clusters (Oracle RAC)、およびデータベース・リンクを使用する環境に適用できます。

改善されたサポート対象アルゴリズムは次のとおりです。

- 暗号化アルゴリズム: AES128、AES192およびAES256
- チェックサム・アルゴリズム: SHA1、SHA256、SHA384およびSHA512

非推奨であり、パッチ適用後に使用をお薦めしていない脆弱なアルゴリズムは、次のとおりです。

- 暗号化アルゴリズム: DES、DES40、3DES112、3DES168、RC4_40、RC4_56、RC4_128およびRC4_256
- チェックサム・アルゴリズム: MD5

実行する一般的な手順としては、まず、Oracle Database環境におけるサポート対象外アルゴリズムへの参照をサポート対象アルゴリズムに置換し、サーバーにパッチを適用し、クライアントにパッチを適用し、最後に、サーバーとクライアントとの正しい接続が再度有効になるようにsqlnet.oraのパラメータを設定します。

このパッチは次の領域に影響を与えますが、これらに限定されるわけではありません。

- JDBCネットワーク暗号化関連の構成設定
- Oracle Net Managerを使用して構成した暗号化パラメータおよび整合性パラメータ
- Transport Layer Security (TLS)のSSL_CIPHER_SUITEパラメータ設定
- SecureFiles LOBの暗号化列
- データベース常駐接続プーリング(DRCP)の構成
- Oracle Call Interface (Oracle OCI)、ODP.NETの構成に使用される暗号化設定

関連トピック

- 『Oracle Database JDBC開発者ガイド』
- Oracle Net Managerを使用した暗号化および整合性パラメータの構成
- ネイティブ・ネットワーク暗号化とTransport Layer Securityの間の選択

親トピック: ネイティブ・ネットワーク暗号化のセキュリティの向上

18.3.2 ネイティブ・ネットワーク暗号化へのセキュリティ改善更新の適用

Oracle Databaseサーバーおよびクライアントにパッチを適用することに加え、サーバーおよびクライアントのsqlnet.oraのパラメータを設定する必要があります。

次のステップは、次に示す順序で実行してください。

1. パッチをインストールするサーバーとクライアントをバックアップします。
2. My Oracle Supportにログインし、My Oracle Supportノート2118136.2で説明されているパッチをダウンロードします。
   My Oracle Supportは、次のURLにあります。

   https://support.oracle.com
3. サーバーにパッチを適用します。
   My Oracle Supportノート2118136.2の手順に従って、サーバーにパッチを適用します。後述のステップで、同じパッチをクライアントに適用します。
4. クライアントにパッチを適用します。

   パッチを適用する必要があるクライアントを決定します。

   My Oracle Supportノート2118136.2の手順に従って、各クライアントにパッチを適用します。
5. 各クライアントのsqlnet.oraファイル内で、非推奨になったアルゴリズムが定義されている場合は、それらのアルゴリズムをすべて削除します。
   次のパラメータが定義されていないか、アルゴリズムがリストされていない場合は、このステップを省略できます。
   - SQLNET.ENCRYPTION_TYPES_CLIENT
   - SQLNET.CRYPTO_CHECKSUM_TYPES_CLIENT
6. サーバーのsqlnet.oraファイル内で、非推奨になったアルゴリズムが定義されている場合は、それらのアルゴリズムをすべて削除します。
   次のパラメータが定義されていないか、アルゴリズムがリストされていない場合は、このステップを省略できます。
   - SQLNET.ENCRYPTION_TYPES_SERVER

- SQLNET.CRYPTO_CHECKSUM_TYPES_SERVER

7. サーバー上のセキュリティを最大限に高めるには、sqlnet.oraの次のパラメータを設定します。
   - SQLNET.ENCRYPTION_SERVER = REQUIRED
   - SQLNET.ENCRYPTION_TYPES_SERVER = (AES256)
   - SQLNET.CRYPTO_CHECKSUM_SERVER = REQUIRED
   - SQLNET.CRYPTO_CHECKSUM_TYPES_SERVER = (SHA512)
   - SQLNET.ALLOW_WEAK_CRYPTO_CLIENTS = FALSE
8. クライアント上のセキュリティを最大限に高めるには、sqlnet.oraの次のパラメータを設定します。
   - SQLNET.ENCRYPTION_CLIENT = REQUIRED
   - SQLNET.ENCRYPTION_TYPES_CLIENT = (AES256)
   - SQLNET.CRYPTO_CHECKSUM_CLIENT = REQUIRED
   - SQLNET.CRYPTO_CHECKSUM_TYPES_CLIENT = (SHA512)
   - SQLNET.ALLOW_WEAK_CRYPTO = FALSE
9. 各クライアントのsqlnet.oraファイル内で、ステップ5および6に従って、非推奨のアルゴリズムをすべてサーバーおよびクライアントから削除し、クライアントがパッチ未適用のサーバーと通信できないように、パラメータSQLNET.ALLOW_WEAK_CRYPTO = FALSEを設定します。

   SQLNET.ALLOW_WEAK_CRYPTOパラメータがFALSEに設定されている場合、脆弱なアルゴリズムの使用をクライアントが試みると、サーバーでORA-12269: クライアントで脆弱な暗号化/暗号チェックサム・バージョンが使用されていますというエラーが発生します。脆弱なアルゴリズムを使用しているサーバー(またはプロキシ)に接続しているクライアントは、ORA-12268: サーバーで脆弱な暗号化/暗号チェックサム・バージョンが使用されていますというエラーを受信します。
10. サーバーのsqlnet.oraファイルで、ステップ9に従って、SQLNET.ALLOW_WEAK_CRYPTO = FALSEのすべてのクライアントを更新した後、パラメータSQLNET.ALLOW_WEAK_CRYPTO_CLIENTS = FALSEを設定します。このパラメータにより、パッチ適用済のサーバーがパッチ未適用のクライアントと通信できないようにします。

    SQLNET.ALLOW_WEAK_CRYPTOパラメータがFALSEに設定されている場合、脆弱なアルゴリズムの使用をクライアントが試みると、サーバーでORA-12269: クライアントで脆弱な暗号化/暗号チェックサム・バージョンが使用されていますというエラーが発生します。脆弱なアルゴリズムを使用しているサーバー(またはプロキシ)に接続しているクライアントは、ORA-12268: サーバーで脆弱な暗号化/暗号チェックサム・バージョンが使用されていますというエラーを受信します。

    データベース・リンクを使用すると、最初のデータベース・サーバーはクライアントとして機能し、2番目のサーバーに接続します。したがって、SQLNET.ALLOW_WEAK_CRYPTOをFALSEに設定する前に、すべてのサーバーに完全にパッチが適用され、サポートされていないアルゴリズムが削除されていることを確認してください。

親トピック: ネイティブ・ネットワーク暗号化のセキュリティの向上

18.4 データの整合性アルゴリズムのサポート

データ整合性アルゴリズムは、サード・パーティ攻撃およびメッセージ・リプレイ攻撃から保護します。SHA-2をお薦めしますが、SHA-1 (非推奨)およびMD5は下位互換性のために維持されています。

ノート:

MD5 は、このリリースでは非推奨です。より強力なアルゴリズムを使用するように Oracle Database 環境を移行するには、My Oracle Support ノート 2118136.2 で説明されているパッチをダウンロードしてインストールします。

これらのハッシュ・アルゴリズムは、データがなんらかの方法で変更された場合に変更されるチェックサムを作成します。この保護機

能は暗号化プロセスとは独立して動作するため、暗号化の有無に関係なくデータ整合性を保つことができます。

関連トピック

- クライアントとサーバーでの整合性の構成

親トピック: Oracle Databaseのネイティブ・ネットワーク暗号化とデータ整合性の構成

18.5 Diffie-Hellmanベースのキー交換

Diffie-Hellmanキー交換アルゴリズムを使用して、マルチユーザー環境でデータを保護できます。

マルチユーザー環境では、安全なキーを配布することは困難です。Oracle Databaseでは、一般的なDiffie-Hellmanキー交換アルゴリズムを使用して、暗号化およびデータ整合性の両面において安全なキーの配布を実現します。

暗号化を使用して暗号データを保護するときは、キーを頻繁に変更して、キーの安全性が損なわれた場合の影響を最小限に抑える必要があります。そのため、Oracle Databaseのキー管理機能では、セッションごとにセッション・キーが変更されます。

Diffie-Hellmanキー交換アルゴリズムは、セキュアでないチャネルを介して通信する2つのパーティが、それらのパーティのみが知っているランダムな数字を合意させる方法です。Oracle Databaseでは、セッション・キーの生成にDiffie-Hellmanキー交換アルゴリズムが使用されています。

クライアントとサーバーは、Diffie-Hellmanによって生成されるセッション・キーを使用して通信を開始します。サーバーに対するクライアントの認証時に、両者のみが認識する共有秘密鍵が確立されます。Oracle Databaseでは、その共有秘密キーとDiffie-Hellmanセッション・キーを組み合せることで、第三者の攻撃を阻止するためのさらに強力なセッション・キーを生成します。

> ノート:
>
> Oracle Database で使用可能なよりセキュアな認証接続を使用することをお薦めします。エンタープライズ・ユーザー・セキュリティのために Oracle Internet Directory に接続するために RC4 で匿名 Diffie-Hellman を使用する場合は、別のアルゴリズム接続を使用するように移行する必要があります。TLS 一方向または証明書を使用した相互認証のいずれかを使用することをお薦めします。

親トピック: Oracle Databaseのネイティブ・ネットワーク暗号化とデータ整合性の構成

18.6 データの暗号化および整合性の構成

Oracle Database固有のOracle Net Servicesによる暗号化および整合性は、Oracle Net Servicesがインストール済であることを前提としています。

- 暗号化および整合性のアクティブ化について
  どのネットワーク接続でも、クライアントとサーバーの両方が複数の暗号化アルゴリズムと複数の整合性アルゴリズムをサポートできます。
- 暗号化および整合性のネゴシエーションについて
  データの暗号化および整合性を使用するシステム上の`sqlnet.ora`ファイルには、`REJECTED`、`ACCEPTED`、`REQUESTED`および`REQUIRED`パラメータの一部またはすべてが含まれている必要があります。
- Oracle Net Managerを使用した暗号化および整合性パラメータの構成
  Oracle Net Managerを使用して、暗号化および整合性パラメータを設定または変更できます。

親トピック: Oracle Databaseのネイティブ・ネットワーク暗号化とデータ整合性の構成

18.6.1 暗号化および整合性のアクティブ化について

どのネットワーク接続でも、クライアントとサーバーの両方が複数の暗号化アルゴリズムと複数の整合性アルゴリズムをサポートできます。

接続が確立されるときに、sqlnet.oraファイルで指定されているアルゴリズムの中から、使用するアルゴリズムをサーバーが選択します。サーバーは、クライアントとサーバーの両方で使用できるアルゴリズム間で一致するものを検索し、サーバー側のリストで最初にあって、クライアント側のリストにも出現するアルゴリズムを選択します。接続の一方の側がアルゴリズム・リストを指定していない場合、その側でインストールされているすべてのアルゴリズムを使用できます。いずれかの側でインストールされていないアルゴリズムを指定すると、エラー・メッセージORA-12650が表示され、接続が失敗します。

暗号化パラメータと整合性パラメータを定義するには、ネットワーク上のクライアントとサーバーのsqlnet.oraファイルを変更します。

利用可能な暗号化アルゴリズムの一部またはすべて、および利用可能な整合性アルゴリズムの一方または両方を構成できます。各接続セッションに使用できるのは、1つの暗号化アルゴリズムと1つの整合性アルゴリズムのみです。

> ノート:
>
> Oracle Database では、クライアントとサーバーで利用可能なアルゴリズムのうち、最初の暗号化アルゴリズムと最初の整合性アルゴリズムが自動的に選択されます。ネゴシエーションの優先順にアルゴリズムとキーの長さを選択することをお薦めします(つまり、最も強力なキーの長さを最初に選択します)。

関連項目:

- 有効な暗号化アルゴリズムをリストする表18-4
- 使用可能な整合性アルゴリズムのリストは、『Oracle Database Advanced Securityガイド』を参照してください。
- データ暗号化および整合性パラメータ

親トピック: データの暗号化および整合性の構成

18.6.2 暗号化および整合性のネゴシエーションについて

データの暗号化および整合性を使用するシステム上のsqlnet.oraファイルには、REJECTED、ACCEPTED、REQUESTED、REQUIREDパラメータの一部またはすべてが含まれている必要があります。

- 暗号化および整合性のネゴシエーションの値について
  Oracle Net Managerを使用して、暗号化と整合性の構成パラメータに4つの値を指定できます。
- REJECTED構成パラメータ
  REJECTED値は、他方が要求している場合でも、セキュリティ・サービスを無効にします。
- ACCEPTED構成パラメータ
  ACCEPTED値は、他方が必要としている場合または要求している場合に、セキュリティ・サービスを有効にします。
- REQUESTED構成パラメータ
  REQUESTED値は、他方が許可している場合にセキュリティ・サービスを有効にします。

- REQUIRED構成パラメータ
  REQUIRED値は、セキュリティ・サービスを有効にする、または接続を禁止します。

親トピック: データの暗号化および整合性の構成

18.6.2.1 暗号化および整合性のネゴシエーションの値について

Oracle Net Managerを使用して、暗号化と整合性の構成パラメータに4つの値を指定できます。

次の4つの値はセキュリティの低い順で記載されています。暗号化および整合性を使用しているシステムのクライアントとサーバーのプロファイル・ファイル(sqlnet.ora)でこれらを使用する必要があります。

値REJECTEDは、クライアントとサーバーの間の通信に最小レベルのセキュリティを提供し、値REQUIREDは、最高レベルのネットワーク・セキュリティを提供します。

- REJECTED
- ACCEPTED
- REQUESTED
- REQUIRED

各パラメータのデフォルト値はACCEPTEDです。

Oracle Databaseサーバーおよびクライアントは、デフォルトではACCEPT暗号化接続に設定されます。これは、接続の片側のみ(サーバー側またはクライアント側)を構成するだけで、接続ペアに対して目的の暗号化および整合性設定を有効化できることを意味します。

したがって、たとえば、数多くのOracleクライアントがOracleデータベースに接続する場合も、サーバー側でsqlnet.oraに適切な変更を加えることによって、すべての接続に対して必要な暗号化および整合性設定を構成できます。クライアントごとに個別に構成の変更を実装する必要はありません。

表18-3に、クライアントとサーバーの構成パラメータを各種組み合せたときに、セキュリティ・サービスが有効化されるかどうかを示します。サーバーまたはクライアントでREQUIREDが指定されている場合は、共通のアルゴリズムが存在しないと、接続が失敗します 。それ以外の場合は、サービスが有効化されていて、共通のサービス・アルゴリズムが存在しないと、サービスが無効化されます。

表18-3 暗号化とデータ整合性のネゴシエーション

クライアントの設定	サーバーの設定	暗号化とデータのネゴシエーション
REJECTED	REJECTED	OFF
ACCEPTED	REJECTED	OFF
REQUESTED	REJECTED	OFF
REQUIRED	REJECTED	接続失敗
REJECTED	ACCEPTED	OFF
ACCEPTED	ACCEPTED	OFF 脚注 1

クライアントの設定	サーバーの設定	暗号化とデータのネゴシエーション
REQUESTED	ACCEPTED	ON
REQUIRED	ACCEPTED	ON
REJECTED	REQUESTED	OFF
ACCEPTED	REQUESTED	ON
REQUESTED	REQUESTED	ON
REQUIRED	REQUESTED	ON
REJECTED	REQUIRED	接続失敗
ACCEPTED	REQUIRED	ON
REQUESTED	REQUIRED	ON
REQUIRED	REQUIRED	ON

脚注1

この値のデフォルトはOFFです。ユーザーがOracle Net Managerを使用するか、sqlnet.oraファイルを変更することによってこのパラメータを変更しないかぎり、暗号化とデータ整合性は有効化されません。

親トピック: 暗号化および整合性のネゴシエーションについて

18.6.2.2 REJECTED構成パラメータ

REJECTED値は、他方が要求している場合でも、セキュリティ・サービスを無効にします。

このシナリオでは、セキュリティ・サービスの使用が許可されないことを接続元で指定します。接続先がREQUIREDに設定されている場合、エラー・メッセージORA-12650が表示されて接続が終了します。接続先がREQUESTED、ACCEPTEDまたはREJECTEDに設定されている場合、エラーは発生せずに、セキュリティ・サービスが無効のまま接続が継続されます。

親トピック: 暗号化および整合性のネゴシエーションについて

18.6.2.3 ACCEPTED構成パラメータ

ACCEPTED値は、他方が必要としている場合または要求している場合に、セキュリティ・サービスを有効にします。

このシナリオでは、接続元からはセキュリティ・サービスを要求しませんが、接続先がREQUIREDまたはREQUESTEDに設定されている場合は、セキュリティ・サービスが有効化されます。接続先がREQUIREDまたはREQUESTEDに設定されていて、該当する暗号化アルゴリズムまたは整合性アルゴリズムが見つかると、エラーは発生せずに、セキュリティ・サービスが有効のまま接続が継続されます。接続先がREQUIREDに設定されていて、該当するアルゴリズムが見つからない場合、エラー・メッセージORA-12650が表示されて接続が終了します。

接続先がREQUESTEDに設定されていて、該当するアルゴリズムが見つからない場合、または接続先がACCEPTEDまたはREJECTEDに設定されている場合、エラーは発生せずに、セキュリティ・サービスが無効のまま接続が継続されます。

親トピック: 暗号化および整合性のネゴシエーションについて

18.6.2.4 REQUESTED構成パラメータ

REQUESTED値は、他方が許可している場合にセキュリティ・サービスを有効にします。

このシナリオでは、接続元がセキュリティ・サービスの使用を希望します(必須ではない)。接続先でACCEPTED、REQUESTEDまたはREQUIREDが指定されている場合、セキュリティ・サービスが有効化されます。接続先に該当するアルゴリズムがある必要があります。見つからない場合、セキュリティ・サービスは有効化されません。接続先でREQUIREDが指定されていて、該当するアルゴリズムが見つからない場合、接続は失敗します 。

親トピック: 暗号化および整合性のネゴシエーションについて

18.6.2.5 REQUIRED構成パラメータ

REQUIRED値は、セキュリティ・サービスを有効にする、または接続を禁止します。

このシナリオでは、接続元がセキュリティ・サービスの有効化を強制します。接続先でREJECTEDが指定されている場合、または接続先に互換性のあるアルゴリズムが見つからない場合、接続は失敗します。

親トピック: 暗号化および整合性のネゴシエーションについて

18.6.3 Oracle Net Managerを使用した暗号化および整合性パラメータの構成

Oracle Net Managerを使用して、暗号化および整合性パラメータを設定または変更できます。

- クライアントとサーバーでの暗号化の構成
  クライアントとサーバーで暗号化を構成するには、Oracle Net Managerを使用します。
- クライアントとサーバーでの整合性の構成
  Oracle Net Managerを使用して、クライアントとサーバーの両方でネットワーク整合性を構成できます。
- 異なるユーザーに対するOracleネイティブ暗号化とSSL認証の両方の同時有効化
  SQLNET.ENCRYPTION_CLIENTおよびSQLNET.ENCRYPTION_SERVERの設定に応じて、異なるユーザーに対してOracleネイティブ暗号化とSSL認証の両方を同時に許可するようにOracle Databaseを構成できます。

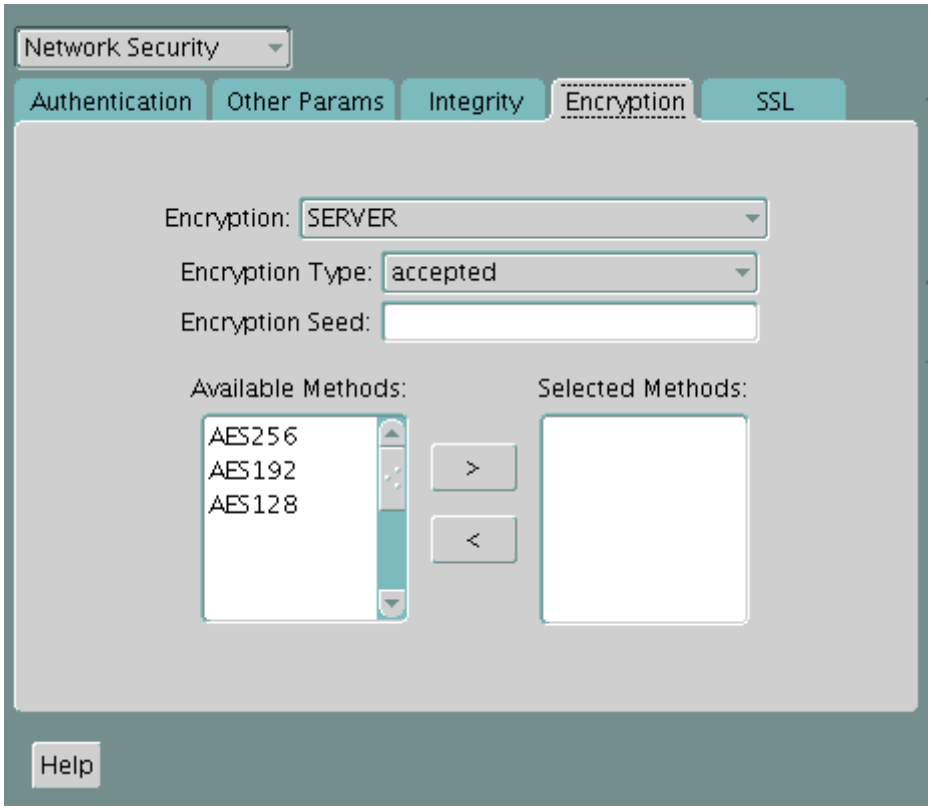
親トピック: データの暗号化および整合性の構成

18.6.3.1 クライアントとサーバーでの暗号化の構成

クライアントとサーバーで暗号化を構成するには、Oracle Net Managerを使用します。

1. Oracle Net Managerを起動します。
   - (UNIX) $ORACLE_HOME/binから、コマンドラインで次のコマンドを入力します。
     ```
     netmgr
     ```
   - (Windows)「スタート」→「プログラム」→「Oracle - HOME_NAME」→「Configuration and Migration Tools」→「Net Manager」を選択します。
2. 「Oracle Netの構成」を展開し、「ローカル」から「プロファイル」を選択します。
3. 「ネーミング」リストから、「ネットワーク・セキュリティ」を選択します。

   ネットワーク・セキュリティのタブ付きウィンドウが表示されます。
4. 「暗号化」タブを選択します。



5. 「暗号化」ボックスからCLIENTまたはSERVERオプションを選択します。
6. 「暗号化タイプ」リストから、次のいずれかを選択します。
   - REQUESTED
   - REQUIRED
   - ACCEPTED
   - REJECTED
7. (オプション)「暗号化シード」フィールドに、10から70字のランダムな文字を入力します。クライアントの暗号化シードは、サーバーの暗号化シードとは別のものにします。
8. 「使用可能なメソッド」リストで暗号化アルゴリズムを選択します。右矢印(>)を選択して「選択メソッド」リストに移動します。追加の方式を使用する場合は、それぞれ同じ手順を繰り返します。
9. 「ファイル」→「ネットワーク構成の保存」を選択します。sqlnet.oraファイルが更新されます。
10. 同じ手順を繰り返して、もう一方のシステムで暗号化を構成します。2つのシステムのsqlnet.oraファイルに、次のエントリが含まれている必要があります。
    - サーバー:

```
SQLNET.ENCRYPTION_SERVER = [accepted | rejected | requested | required]
SQLNET.ENCRYPTION_TYPES_SERVER = (valid_encryption_algorithm
[,valid_encryption_algorithm])
```

    - クライアント:

```
SQLNET.ENCRYPTION_CLIENT = [accepted | rejected | requested | required]
SQLNET.ENCRYPTION_TYPES_CLIENT = (valid_encryption_algorithm
[,valid_encryption_algorithm])
```

表18-4に、有効な暗号化アルゴリズムと対応する有効な値を示します。

表18-4 有効な暗号化アルゴリズム

アルゴリズム名	有効な値
AES 256 ビット・キー	AES256
AES 192 ビット・キー	AES192
AES 128 ビット・キー	AES128

親トピック: Oracle Net Managerを使用した暗号化および整合性パラメータの構成

18.6.3.2 クライアントとサーバーでの整合性の構成

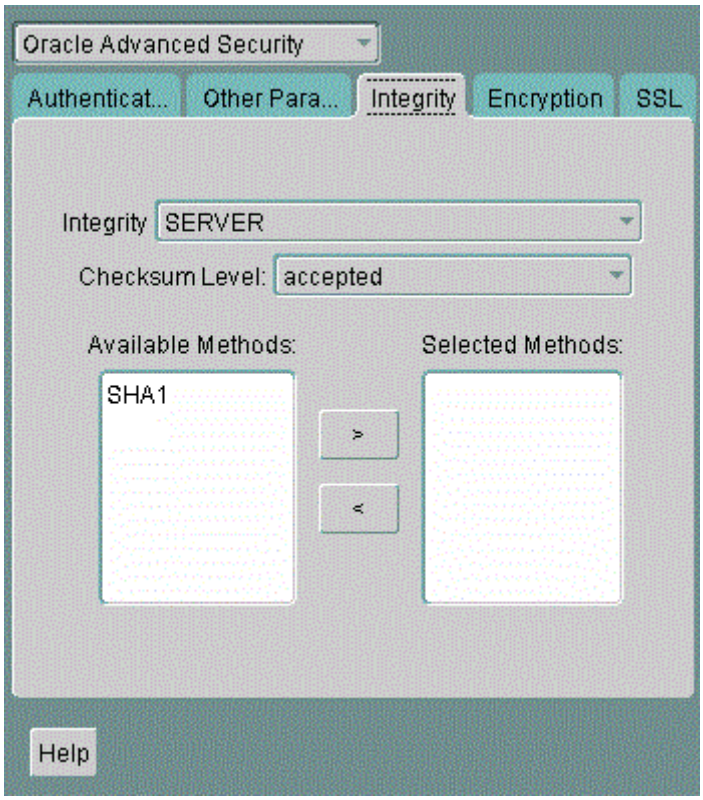
Oracle Net Managerを使用して、クライアントとサーバーの両方でネットワーク整合性を構成できます。

1. Oracle Net Managerを起動します。
   - (UNIX) `$ORACLE_HOME/bin`から、コマンドラインで次のコマンドを入力します。

     ```
     netmgr
     ```

   - (Windows)「スタート」→「プログラム」→「Oracle - HOME_NAME」→「Configuration and Migration Tools」→「Net Manager」を選択します。
2. 「Oracle Netの構成」を展開し、「ローカル」から「プロファイル」を選択します。
3. 「ネーミング」リストから、「ネットワーク・セキュリティ」を選択します。

   ネットワーク・セキュリティのタブ付きウィンドウが表示されます。
4. 「整合性」タブを選択します。



5. いずれのシステムを構成しているかに応じて、「整合性」ボックスから「サーバー」または「クライアント」を選択します。
6. 「チェックサム・レベル」リストから、次のチェックサム・レベル値のいずれかを選択します。
   - REQUESTED

- REQUIRED
- ACCEPTED
- REJECTED

7. 「使用可能なメソッド」リストで整合性アルゴリズムを選択します。右矢印(>)を選択して「選択メソッド」リストに移動します。追加の方式を使用する場合は、それぞれ同じ手順を繰り返します。
8. 「ファイル」→「ネットワーク構成の保存」を選択します。

   sqlnet.oraファイルが更新されます。

9. 同じ手順を繰り返して、もう一方のシステムで整合性を構成します。

   2つのシステムのsqlnet.oraファイルに、次のエントリが含まれている必要があります。

   - サーバー:

     ```
     SQLNET.CRYPTO_CHECKSUM_SERVER = [accepted | rejected | requested |
     required]
     SQLNET.CRYPTO_CHECKSUM_TYPES_SERVER = (valid_crypto_checksum_algorithm
     [,valid_crypto_checksum_algorithm])
     ```

   - クライアント:

     ```
     SQLNET.CRYPTO_CHECKSUM_CLIENT = [accepted | rejected | requested |
     required]
     SQLNET.CRYPTO_CHECKSUM_TYPES_CLIENT = (valid_crypto_checksum_algorithm
     [,valid_crypto_checksum_algorithm])
     ```

ユーザーが使用できる有効な整合性/チェックサムのアルゴリズムは次のとおりです。

- SHA1
- SHA256
- SHA384
- SHA512

関連トピック

- Oracle Database Advanced Securityガイド

親トピック: Oracle Net Managerを使用した暗号化および整合性パラメータの構成

18.6.3.3 異なるユーザーに対するOracleネイティブ暗号化とSSL認証の両方の同時有効化

SQLNET.ENCRYPTION_CLIENTおよびSQLNET.ENCRYPTION_SERVERの設定に応じて、異なるユーザーに対してOracleネイティブ暗号化とSSL認証の両方を同時に許可するようにOracle Databaseを構成できます。

- 異なるユーザーに対するOracleネイティブ暗号化とSSL認証の両方の同時有効化について
  デフォルトでは、Oracle Databaseでは、異なるユーザーに対するOracleネイティブ暗号化とTransport Layer Security (SSL)認証の両方を同時に使用することはできません。
- 異なるユーザーに対するOracleネイティブ暗号化とSSL認証の両方の同時構成
  IGNORE_ANO_ENCRYPTION_FOR_TCPSパラメータを使用して、Oracleネイティブ暗号化とTransport Layer Security (SSL)認証の両方を同時に使用できるようにします。

親トピック: Oracle Net Managerを使用した暗号化および整合性パラメータの構成

18.6.3.3.1 異なるユーザーに対するOracleネイティブ暗号化とSSL認証の両方の同時有効化について

デフォルトでは、Oracle Databaseでは、異なるユーザーに対するOracleネイティブ暗号化とTransport Layer Security (SSL)認証の両方を同時に使用することはできません。

Oracleネイティブ暗号化(Advanced Networking Option (ANO)暗号化とも呼ばれる)とTLS認証の併用は、二重暗号化と呼ばれます。

一部のユーザーにユーザー名とパスワードの使用によるサーバーへの接続を可能にし、他のユーザーにTLS証明書の使用によるそのサーバーへの認証を可能にするように、TCPリスナーとTCPSリスナーの両方を構成する必要がある場合があります。このような場合、パスワードベースの認証とTLS認証の両方を構成する必要があります。以前のリリースの回避策では、SQLNET.ENCRYPTION_SERVERパラメータをrequestedに設定していました。SQLNET.ENCRYPTION_SERVERがrequiredに設定されている必要がある場合、SQLNET.ENCRYPTION_CLIENTとSQLNET.ENCRYPTION_SERVERの両方のIGNORE_ANO_ENCRYPTION_FOR_TCPSパラメータをTRUEに設定できます。デフォルトでは、これはFALSEに設定されています。

IGNORE_ANO_ENCRYPTION_FOR_TCPSをTRUEに設定すると、クライアントはすべての発信TCPS接続のSQLNET.ENCRYPTION_CLIENTパラメータに設定されている値を無視します。このパラメータを使用すると、TCPSクライアントの使用に矛盾がある場合に、データベースでSQLNET.ENCRYPTION_CLIENTまたはSQLNET.ENCRYPTION_SERVERの設定を、これら2つのパラメータがrequiredに設定されていても無視できます。

親トピック: 異なるユーザーに対するOracleネイティブ暗号化とSSL認証の両方の同時有効化

18.6.3.3.2 異なるユーザーに対するOracleネイティブ暗号化とSSL認証の両方の同時構成

IGNORE_ANO_ENCRYPTION_FOR_TCPSパラメータを使用して、Oracleネイティブ暗号化とTransport Layer Security (SSL)認証の両方を同時に使用できるようにします。

サーバーではIGNORE_ANO_ENCRYPTION_FOR_TCPSをsqlnet.oraファイルで設定する必要があり、クライアントではsqlnet.oraファイルまたはtnsnames.oraファイルのいずれかで設定できます。

1. データベース・サーバーにログインします
2. sqlnet.oraファイルの場所に移動します。
   デフォルトでは、sqlnet.oraはORACLE_BASE/network/adminディレクトリにあります。sqlnet.oraファイルは、環境変数TNS_ADMINで指定されたディレクトリに配置される場合もあります。
3. sqlnet.oraで、現在のSQLNET.ENCRYPTION_SERVERの設定がrequiredかrequestedかを確認します。
4. SQLNET.ENCRYPTION_SERVERがrequiredに設定されている場合は、
   SQLNET.IGNORE_ANO_ENCRYPTION_FOR_TCPSをsqlnet.oraに追加してからTRUEに設定します。

   ```
   IGNORE_ANO_ENCRYPTION_FOR_TCPS=TRUE
   ```

5. sqlnet.oraファイルを保存して終了します。
6. クライアントにログインします。
   クライアントの場合、sqlnet.oraファイルまたはtnsnames.oraファイルのいずれかに値を設定できます。
   - sqlnet.oraの値の設定: SQLNET.ENCRYPTION_CLIENTパラメータがrequiredに設定されているか確認します。SQLNET.ENCRYPTION_CLIENTの場合は、sqlnet.oraファイルを編集して次の設定を行います。

     ```
     IGNORE_ANO_ENCRYPTION_FOR_TCPS=TRUE
     ```

   - tnsnames.oraの値の設定: デフォルトでは、tnsnames.oraはsqlnet.oraと同じ場所にあります。

SQLNET.ENCRYPTION_CLIENTがsqlnet.oraでrequiredに設定されている場合は、TNS_ALIAS設定のSECURITY部分でIGNORE_ANO_ENCRYPTION_FOR_TCPS=TRUEを設定します。たとえば:

```
test_tls=
    (DESCRIPTION =
       (ADDRESS=(PROTOCOL=tcps)(HOST=)(PORT=1750))
       (CONNECT_DATA=(SID=^ORACLE_SID^))
       (SECURITY=(IGNORE_ANO_ENCRYPTION_FOR_TCPS=TRUE))
     )
```

親トピック: 異なるユーザーに対するOracleネイティブ暗号化とSSL認証の両方の同時有効化

19 シンJDBCクライアント・ネットワークの構成

Oracle Databaseのネイティブ暗号化および厳密認証を使用すると、シンJava Database Connectivity (JDBC)クライアントは、Oracleデータベースに安全に接続できます。

- Java実装について
  Oracle Databaseには、ネイティブ・ネットワーク暗号化と厳密認証のJava実装が用意されています。
- Java Database Connectivityのサポート
  業界標準のJavaインタフェースであるJDBCは、Javaプログラムからリレーショナル・データベースに接続するためのJava標準です。
- シンJDBCの機能
  シンJDBCドライバは、厳密認証、データ暗号化、データ整合性チェックなど、セキュリティ機能を備えています。
- 実装の概要
  サーバー側で、アルゴリズムのネゴシエーションおよびキーの生成は、Oracle Database固有の暗号化とまったく同様に機能します。
- Java暗号化コードの不明瞭化
  Java暗号化コードの不明瞭化では、不明瞭化ソフトウェアを使用して、暗号化機能と複号化機能を含むJavaクラスおよびメソッドを保護します。
- シンJDBCネットワーク実装の構成パラメータ
  クライアントのシンJDBCネットワーク実装では、暗号化や整合性、認証サービスを制御するパラメータを提供します。

親トピック: ネットワーク上のデータの保護

19.1 Java実装について

Oracle Databaseには、ネイティブ・ネットワーク暗号化と厳密認証のJava実装が用意されています。

Oracle Databaseのネイティブ・ネットワーク暗号化および厳密認証のJava実装では、Oracle Databaseのネイティブ・ネットワーク暗号化および厳密認証が構成されているOracle Databaseと通信するシンJDBCクライアントに対して、ネットワーク認証、暗号化および整合性の保護が提供されます。

関連項目:

JDBCの詳細および例は、『Oracle Database JDBC開発者ガイド』を参照してください。

親トピック: シンJDBCクライアント・ネットワークの構成

19.2 Java Database Connectivityのサポート

業界標準のJavaインタフェースであるJDBCは、Javaプログラムからリレーショナル・データベースに接続するためのJava標準です。

JDBC標準はSun Microsystemsによって規定され、オラクル社では独自のJDBCドライバによってこの標準を実装および拡張しています。

Oracle JDBCドライバは、Oracleデータベースと通信するJava Database Connectivity (JDBC)アプリケーションを作成するために使用されます。Oracleでは、CベースのOracle Netクライアントの最上部に構築されたシックJDBCドライバ、および

ダウンロード可能なアプレットをサポートするシン(Pure Java)JDBCドライバの2つのタイプのJDBCドライバが実装されます。JDBCに対するOracleの拡張機能には、次の機能が含まれます。

- データ・アクセスおよび操作
- LOBアクセスおよび操作
- Oracleオブジェクト型マッピング
- オブジェクト参照アクセスおよび操作
- 配列アクセスおよび操作
- アプリケーション・パフォーマンスの向上

親トピック: シンJDBCクライアント・ネットワークの構成

19.3 シンJDBCの機能

シンJDBCドライバは、厳密認証、データ暗号化、データ整合性チェックなど、セキュリティ機能を備えています。

シンJDBCドライバは、インターネットで使用されるダウンロード可能なアプレットとともに使用するように設計されているため、オラクル社では、シン・クライアントとともに使用するOracle Databaseのネイティブ・ネットワーク暗号化および厳密認証、暗号化および整合性のアルゴリズムがJavaで100%実装されるように設計しました。

Oracle Databaseには、シンJDBCのための次の機能が備えられています。

- 厳密認証
- データの暗号化
- データ整合性チェック
- シンJDBCクライアントからOracle RDBMSへの接続の保護
- 開発者が安全な通信チャネルでデータを転送するアプレットを作成するための機能
- Java Server Pages (JSP)を持つ中間層サーバーからOracle RDBMSへの接続の保護
- 現在のリリースのOracle Databaseから旧バージョンのOracle Databaseへの接続の保護

Oracle JDBCシン・ドライバは、Oracle DatabaseのSSL実装およびRADIUSやKerberosなどのサード・パーティの認証方式をサポートしています。シンJDBCによるRADIUS、Kerberos、SSLなどの認証方式のサポートは、Oracle Database 11g リリース1 (11.1)で導入されました。

Oracle Databaseのネイティブ・ネットワーク暗号化および厳密認証のJava実装では、次の暗号化アルゴリズムのJavaバージョンが提供されます。

- AES256: AES 256ビット・キー
- AES192: AES 192ビット・キー
- AES128: AES 128ビット・キー

ノート:

前述のアルゴリズムのリストで、CBC は暗号ブロック連鎖モードのことです。

シンJDBCによるAdvanced Encryption Standard (AES)のサポートは、Oracle Database 12cリリース1 (12.1)で導入されました。

また、この実装により、Secure Hash Algorithm (SHA1)およびMessage Digest 5 (MD5)を使用したシンJDBCのデータ整合性チェックが実行されます。シンJDBCによるSHA1のサポートは、Oracle Database 11g リリース1 (11.1)で導入されました。

関連項目:

シンJDBCクライアントに対する認証、暗号化および整合性の構成の詳細は、『Oracle Database JDBC開発者ガイド』を参照してください。

ノート:

MD5 は、このリリースでは非推奨です。より強力なアルゴリズムを使用するように Oracle Database 環境を移行するには、My Oracle Support ノート 2118136.2 で説明されているパッチをダウンロードしてインストールします。

親トピック: シンJDBCクライアント・ネットワークの構成

19.4 実装の概要

サーバー側で、アルゴリズムのネゴシエーションおよびキーの生成は、Oracle Database固有の暗号化とまったく同様に機能します。

この機能により、クライアントとサーバーの下位および上位互換性が維持されます。

クライアント側では、アルゴリズムのネゴシエーションおよびキーの生成は、OCIクライアントとまったく同じ方法で行われます。クライアントとサーバーは、従来のOracle Netクライアントと同様の方法で、暗号化アルゴリズムのネゴシエーション、乱数の生成、Diffie-Hellmanを使用したセッション・キーの交換を行い、Oracle Password Protocolを使用します。シンJDBCには、Oracle Netクライアントがpure Javaで完全に実装されています。

親トピック: シンJDBCクライアント・ネットワークの構成

19.5 Java暗号化コードの不明瞭化

Java暗号化コードの不明瞭化では、不明瞭化ソフトウェアを使用して、暗号化機能と複号化機能を含むJavaクラスおよびメソッドを保護します。

Javaバイト・コードの不明瞭化は、Javaプログラムの形式で作成された知的財産を保護するためによく使用されるプロセスです。これによって、コード内のJavaシンボルが変更されます。プロセスは、元のプログラム構造をそのまま保持し、意図した動作を隠すためにクラス、メソッドおよび変数の名前を変更する一方でプログラムが正常に稼働するようにする。不明瞭化されていないJavaコードは再コンパイルして読むことができますが、不明瞭化されたJavaコードは再コンパイルが難しく、米国政府の輸出規制を満たすことができます。

親トピック: シンJDBCクライアント・ネットワークの構成

19.6 シンJDBCネットワーク実装の構成パラメータ

クライアントのシンJDBCネットワーク実装では、暗号化や整合性、認証サービスを制御するパラメータを提供します。

- シンJDBCネットワーク実装の構成パラメータについて
  JDBCネットワーク実装の構成パラメータでは、クライアントとサーバー間接続で使用するセキュリティのレベルなど、ネットワーク設定を制御します。
- クライアント暗号化レベルのパラメータ
  `CONNECTION_PROPERTY_THIN_NET_ENCRYPTION_LEVEL`パラメータは、クライアントがサーバーとのネゴシエートに使用するセキュリティのレベルを定義します。
- クライアント暗号化選択リストのパラメータ
  `CONNECTION_PROPERTY_THIN_NET_ENCRYPTION_TYPES`パラメータは、使用する暗号化アルゴリズムを定義します。
- クライアント整合性レベルのパラメータ
  `CONNECTION_PROPERTY_THIN_NET_CHECKSUM_LEVEL`パラメータは、データ整合性のためにサーバーとネゴシエートする際のセキュリティのレベルを定義します。
- クライアント整合性選択リストのパラメータ
  `CONNECTION_PROPERTY_THIN_NET_CHECKSUM_TYPES`パラメータは、使用するデータ整合性アルゴリズムを定義します。
- クライアント認証サービスのパラメータ
  `CONNECTION_PROPERTY_THIN_NET_AUTHENTICATION_SERVICES`パラメータは、使用する認証サービスを決定します。
- AnoServices定数
  `oracle.net.ano.AnoServices`インタフェースには、JDBCシン・ドライバによってサポートされる暗号化、認証およびチェックサムのアルゴリズムの名前が含まれます。

親トピック: シンJDBCクライアント・ネットワークの構成

19.6.1 シンJDBCネットワーク実装の構成パラメータについて

JDBCネットワーク実装の構成パラメータでは、クライアントとサーバー間接続で使用するセキュリティのレベルなど、ネットワーク設定を制御します。

いくつかの構成パラメータを含むプロパティ・クラス・オブジェクトは、Oracle Databaseのネイティブ・ネットワーク暗号化および厳密認証インタフェースに渡されます。

Oracle Databaseに関連する接続プロパティを含むすべてのJDBC接続プロパティは、`oracle.jdbc.OracleConnection`インタフェースで定数として定義されます。次のリストに、それらの接続プロパティの一部を列挙します。

関連項目:

構成パラメータおよび構成例の詳細は、『Oracle Database JDBC開発者ガイド』を参照してください。

親トピック: シンJDBCネットワーク実装の構成パラメータ

19.6.2 クライアント暗号化レベルのパラメータ

CONNECTION_PROPERTY_THIN_NET_ENCRYPTION_LEVELパラメータは、クライアントがサーバーとのネゴシエートに使用するセキュリティのレベルを定義します。

表19-1に、このパラメータの属性を示します。

表19-1 CONNECTION_PROPERTY_THIN_NET_ENCRYPTION_LEVELの属性

属性	説明
パラメータ・タイプ	文字列
パラメータ・クラス	静的
設定できる値	REJECTED、ACCEPTED、REQUESTED、REQUIRED
デフォルト値	ACCEPTED
構文	prop.setProperty(OracleConnection.CONNECTION_PROPERTY_THIN_NET_ENCRYPTION_LEVEL,level); prop は Properties クラスのオブジェクトです。
例	prop.setProperty(OracleConnection.CONNECTION_PROPERTY_THIN_NET_ENCRYPTION_LEVEL,"REQUIRED"); prop は Properties クラスのオブジェクトです。

親トピック: シンJDBCネットワーク実装の構成パラメータ

19.6.3 クライアント暗号化選択リストのパラメータ

CONNECTION_PROPERTY_THIN_NET_ENCRYPTION_TYPESパラメータは、使用する暗号化アルゴリズムを定義します。

表19-2に、このパラメータの属性を示します。

表19-2 CONNECTION_PROPERTY_THIN_NET_ENCRYPTION_TYPESの属性

属性	説明
パラメータ・タイプ	文字列
パラメータ・クラス	静的
設定できる値	AES256 (AES 256 ビット・キー)、AES192 (AES 192 ビット・キー)、AES128 (AES 128 ビット・キー)

属性	説明
構文	`prop.setProperty(OracleConnection.CONNECTION_PROPERTY_THIN_NET_ENCRYPTION_TYPES,algorithm);` `prop` は `Properties` クラスのオブジェクトです。
例	`prop.setProperty(OracleConnection.CONNECTION_PROPERTY_THIN_NET_ENCRYPTION_TYPES, "( AES256, AES192 )");` `prop` は `Properties` クラスのオブジェクトです。

親トピック: シンJDBCネットワーク実装の構成パラメータ

19.6.4 クライアント整合性レベルのパラメータ

CONNECTION_PROPERTY_THIN_NET_CHECKSUM_LEVELパラメータは、データ整合性のためにサーバーとネゴシエートする際のセキュリティのレベルを定義します。

表19-3に、このパラメータの属性を示します。

表19-3 CONNECTION_PROPERTY_THIN_NET_CHECKSUM_LEVELの属性

属性	説明
パラメータ・タイプ	文字列
パラメータ・クラス	静的
設定できる値	REJECTED、ACCEPTED、REQUESTED、REQUIRED
デフォルト値	ACCEPTED
構文	`prop.setProperty(OracleConnection.CONNECTION_PROPERTY_THIN_NET_CHECKSUM_LEVEL,level);` `prop` は `Properties` クラスのオブジェクトです。
例	`prop.setProperty(OracleConnection.CONNECTION_PROPERTY_THIN_NET_CHECKSUM_LEVEL,"REQUIRED");` `prop` は `Properties` クラスのオブジェクトです。

親トピック: シンJDBCネットワーク実装の構成パラメータ

19.6.5 クライアント整合性選択リストのパラメータ

CONNECTION_PROPERTY_THIN_NET_CHECKSUM_TYPESパラメータは、使用するデータ整合性アルゴリズムを定義します。

表19-4に、このパラメータの属性を示します。

表19-4 CONNECTION_PROPERTY_THIN_NET_CHECKSUM_TYPESの属性

属性	説明
パラメータ・タイプ	文字列
パラメータ・クラス	静的
設定できる値	SHA1
構文	`prop.setProperty(OracleConnection.CONNECTION_PROPERTY_THIN_NET_CHECKSUM_TYPES, algorithm);` `prop` は `Properties` クラスのオブジェクトです。
例	`prop.setProperty(OracleConnection.CONNECTION_PROPERTY_THIN_NET_CHECKSUM_TYPES,"( SHA1 )");` `prop` は `Properties` クラスのオブジェクトです。

親トピック: シンJDBCネットワーク実装の構成パラメータ

19.6.6 クライアント認証サービスのパラメータ

`CONNECTION_PROPERTY_THIN_NET_AUTHENTICATION_SERVICES`パラメータは、使用する認証サービスを決定します。

Table 19-5に、このパラメータの属性を示します。

表19-5 CONNECTION_PROPERTY_THIN_NET_AUTHENTICATION_SERVICESの属性

属性	説明
パラメータ・タイプ	文字列
パラメータ・クラス	静的
設定できる値	`RADIUS`、`KERBEROS`、`SSL`
構文	`prop.setProperty(OracleConnection.CONNECTION_PROPERTY_THIN_NET_AUTHENTICATION_SERVICES,authentication);` `prop` は `Properties` クラスのオブジェクトです。
例	`prop.setProperty(OracleConnection.CONNECTION_PROPERTY_THIN_NET_AUTHENTICATION_SERVICES,"( RADIUS, KERBEROS,`

属性	説明
	SSL)"); propはPropertiesクラスのオブジェクトです。

親トピック: シンJDBCネットワーク実装の構成パラメータ

19.6.7 AnoServices定数

oracle.net.ano.AnoServicesインタフェースには、JDBCシン・ドライバによってサポートされる暗号化、認証およびチェックサムのアルゴリズムの名前が含まれます。

次の定数がoracle.net.ano.AnoServicesインタフェースに含まれます。

```
// ---- SUPPORTED ENCRYPTION ALG -----
public static final String ENCRYPTION_RC4_40 = "RC4_40";
public static final String ENCRYPTION_RC4_56 = "RC4_56";
public static final String ENCRYPTION_RC4_128 = "RC4_128";
public static final String ENCRYPTION_RC4_256 = "RC4_256";
public static final String ENCRYPTION_DES40C = "DES40C";
public static final String ENCRYPTION_DES56C = "DES56C";
public static final String ENCRYPTION_3DES112 = "3DES112";
public static final String ENCRYPTION_3DES168 = "3DES168";
public static final String ENCRYPTION_AES128 = "AES128";
public static final String ENCRYPTION_AES192 = "AES192";
public static final String ENCRYPTION_AES256 = "AES256";
// ---- SUPPORTED INTEGRITY ALG ----
public static final String CHECKSUM_MD5 = "MD5";
public static final String CHECKSUM_SHA1 = "SHA1";
// ---- SUPPORTED AUTHENTICATION ADAPTORS ----
public static final String AUTHENTICATION_RADIUS = "RADIUS";
public static final String AUTHENTICATION_KERBEROS = "KERBEROS";
```

ノート:

DES40、3DES112、3DES168、MD5、RC4_40、RC4_56、RC4_128およびRC4-256アルゴリズムは、このリリースでは非推奨です。より強力なアルゴリズムを使用するようにOracle Database環境を移行するには、My Oracle Supportノート2118136.2で説明されているパッチをダウンロードしてインストールします。

これらの定数を使用して、暗号化、整合性および認証のパラメータを設定できます。例19-1に、その1つの例を示します。

例19-1 JDBCクライアント・コード内でのAnoServices定数の使用

```
import java.sql.*;
import java.util.Properties;import oracle.jdbc.*;
import oracle.net.ano.AnoServices;
/**
 * JDBC thin driver demo: new security features in 11gR1.
 *
 * This program attempts to connect to the database using the JDBC thin
 * driver and requires the connection to be encrypted with either AES256 or AES192
 * and the data integrity to be verified with SHA1.
 *
 * In order to activate encryption and checksumming in the database you need to
 * modify the sqlnet.ora file. For example:
 *
 *   SQLNET.ENCRYPTION_TYPES_SERVER = (AES256,AES192,AES128)
```

```
 *   SQLNET.ENCRYPTION_SERVER = accepted
 *   SQLNET.CRYPTO_CHECKSUM_TYPES_SERVER= (SHA1)
 *   SQLNET.CRYPTO_CHECKSUM_SERVER = accepted
 *
 * This output of this program is:
 *   Connection created! Encryption algorithm is: AES256, data integrity algorithm
 *   is: SHA1
 *
 */
public class DemoAESAndSHA1
{
  static final String USERNAME= "hr";
  static final String PASSWORD= "hr";
  static final String URL =
"jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=somehost.us.example.com)
(PORT=5561))"
+"(CONNECT_DATA=(SERVICE_NAME=itydemo.regress.rdbms.dev.us.example.com)))";

  public static final void main(String[] argv)
  {
    DemoAESAndSHA1 demo = new DemoAESAndSHA1();
    try
    {
      demo.run();
    }catch(SQLException ex)
    {
      ex.printStackTrace();
    }
  }
  void run() throws SQLException
  {
    OracleDriver dr = new OracleDriver();
    Properties prop = new Properties();
    // We require the connection to be encrypted with either AES256 or AES192.
    // If the database doesn't accept such a security level, then the connection
    // attempt will fail.
    prop.setProperty(

OracleConnection.CONNECTION_PROPERTY_THIN_NET_ENCRYPTION_LEVEL,AnoServices.ANO_REQUIR
ED);
    prop.setProperty(
      OracleConnection.CONNECTION_PROPERTY_THIN_NET_ENCRYPTION_TYPES,      "( " +
AnoServices.ENCRYPTION_AES256 + "," +AnoServices.ENCRYPTION_AES192 + ")");
    // We also require the use of the SHA1 algorithm for data integrity checking.
    prop.setProperty(

OracleConnection.CONNECTION_PROPERTY_THIN_NET_CHECKSUM_LEVEL,AnoServices.ANO_REQUIRED
);
    prop.setProperty(
      OracleConnection.CONNECTION_PROPERTY_THIN_NET_CHECKSUM_TYPES,      "( " +
AnoServices.CHECKSUM_SHA1 + " )");

    prop.setProperty("user",DemoAESAndSHA1.USERNAME);
    prop.setProperty("password",DemoAESAndSHA1.PASSWORD);
    OracleConnection oraConn =
 (OracleConnection)dr.connect(DemoAESAndSHA1.URL,prop);

    System.out.println("Connection created! Encryption algorithm is:
"+oraConn.getEncryptionAlgorithmName()    +", data integrity algorithm is:
"+oraConn.getDataIntegrityAlgorithmName());

    oraConn.close();
  }

}
```

親トピック: シンJDBCネットワーク実装の構成パラメータ

第V部 厳密認証の管理

第V部では、厳密認証の管理方法について説明します。

- 厳密認証の概要
  厳密認証では、データベースにログインするユーザーの識別情報を検証するために、Transport Layer Security (TLS)などのツールがサポートされます。
- 厳密認証の管理ツール
  ネイティブ・ネットワーク暗号化および公開キー・インフラストラクチャ資格証明には、厳密認証の一連の管理ツールを使用できます。
- Kerberos認証の構成
  Kerberosは信頼できるサード・パーティ認証システムであり、共有秘密鍵に基づき、サード・パーティがセキュアであることを前提とします。
- Transport Layer Security認証の構成
  Transport Layer Security認証を使用するようにOracle Databaseを構成できます。
- RADIUS認証の構成
  RADIUSは、リモート認証およびアクセスを実現するために広く使用されているクライアント/サーバー・セキュリティ・プロトコルです。
- 厳密認証の使用のカスタマイズ
  Oracle Databaseのネイティブ・ネットワーク暗号化および厳密認証のもとで、複数の認証方法を構成できます。

20 厳密認証の概要

厳密認証では、データベースにログインするユーザーの識別情報を検証するために、Transport Layer Security (TLS)などのツールがサポートされます。

- 厳密認証とは
  認証は、データベースにログインしようとしているユーザーの識別情報を証明するために使用されます。
- 集中化された認証とシングル・サインオン
  シングル・サインオンでは、ユーザーは1つのパスワードで複数のアカウントとアプリケーションにアクセスできます。
- 集中化されたネットワーク認証の動作
  集中化されたネットワーク認証システムは、Oracleサーバー、認証サーバー、およびOracleサーバーに接続するユーザーの間で機能します。
- サポートされている厳密認証方式
  Oracle Databaseは、業界で標準的な認証方式をサポートしています。
- Oracle Databaseのネイティブ・ネットワークの暗号化/厳密認証アーキテクチャ
  Oracle Databaseのネイティブ・ネットワーク暗号化および厳密認証アーキテクチャは、Oracleデータベース・サーバーまたはクライアントのインストールを補完します。
- 厳密認証のシステム要件
  Kerberos、RADIUSおよびTransport Layer Security (TLS)には、厳密認証のための一連のシステム要件があります。
- Oracle Databaseのネイティブ・ネットワーク暗号化および厳密認証の制限事項
  Oracleアプリケーションは、Oracle Databaseのネイティブ・ネットワークの暗号化および厳密認証をサポートしています。

親トピック: 厳密認証の管理

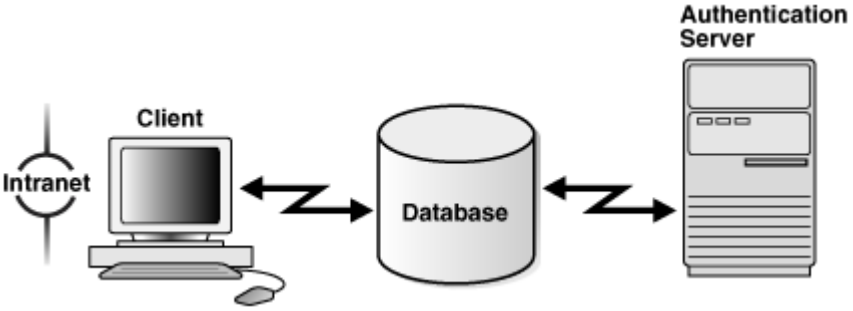
20.1 厳密認証とは

認証は、データベースにログインしようとしているユーザーの識別情報を証明するために使用されます。

分散環境では、ユーザーIDを認証することが必須であり、これを行わないとネットワーク・セキュリティの信頼性はほぼありません。パスワードは、認証で最も一般的な手段です。Oracle Databaseでは、Oracle認証アダプタを使用して厳密認証を有効にし、Oracle認証アダプタは、デジタル証明書を使用したSSLを含む様々なサード・パーティ認証サービスをサポートします。

図20-1は、サード・パーティの認証サーバーを使用するように構成されたOracle Databaseインスタンスでのユーザー認証を示しています。ネットワークのすべてのメンバー(クライアントからサーバー、サーバーからサーバー、ユーザーからクライアントとサーバーの両方)を1箇所で集中的に認証する方法は、メンバーのIDを偽造するネットワーク・ノードの脅威に対処する効果的な方法の1つです。

図20-1 Oracle認証アダプタによる厳密認証



親トピック: 厳密認証の概要

20.2 集中化された認証とシングル・サインオン

シングル・サインオンでは、ユーザーは1つのパスワードで複数のアカウントとアプリケーションにアクセスできます。

集中化された認証では、シングル・サインオン(SSO)の利点もユーザーに提供されます。

シングル・サイン・オンでは、ユーザーがログインする必要があるのは1回のみであり、ユーザー名とパスワードを再度入力しなくても、他の任意のサービスに自動的に接続できます。シングル・サインオンを使用すると、ユーザーは複数のパスワードを記憶したり管理する必要がなくなるため、複数のサービスへのログインに費やす時間を削減できます。

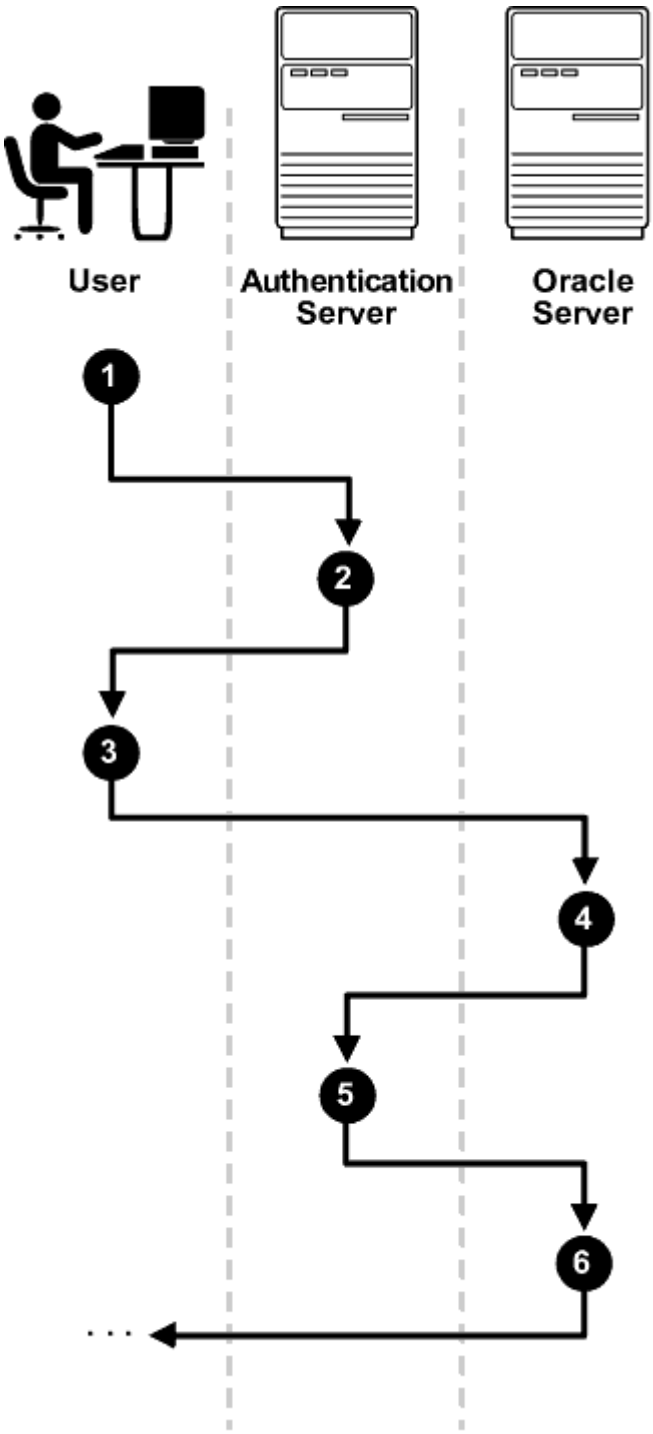
親トピック: 厳密認証の概要

20.3 集中化されたネットワーク認証の動作

集中化されたネットワーク認証システムは、Oracleサーバー、認証サーバー、およびOracleサーバーに接続するユーザーの間で機能します。

次の図は、集中化されたネットワーク認証サービスの一般的な動作方法を表します。

図20-2 ネットワーク認証サービスでのユーザーの認証方法



次のステップは、集中化されたネットワーク認証プロセスがどのように動作するかを記述します。

1. ユーザー(クライアント)が認証サービスを要求し、トークンまたはパスワードなどの識別情報を入力します。
2. 認証サーバーによりユーザーの識別情報が検証され、チケットまたは資格証明がクライアントに渡されます。チケットまたは資格証明には期限がある場合があります。
3. クライアントは、これらの資格証明をサービス・リクエスト(データベースへの接続など)と一緒にOracleサーバーに渡します。
4. サーバーは資格証明を認証サーバーに渡し、認証を行います。
5. 認証サーバーは資格証明をチェックし、Oracleサーバーに通知します。
6. 認証サーバーが資格証明を受け入れた場合、Oracleサーバーはユーザーを認証します。認証サーバーが資格証明を拒否した場合、認証は失敗し、サービス・リクエストは拒否されます。

親トピック: 厳密認証の概要

20.4 サポートされている厳密認証方式

Oracle Databaseは、業界で標準的な次の認証方式をサポートしています。

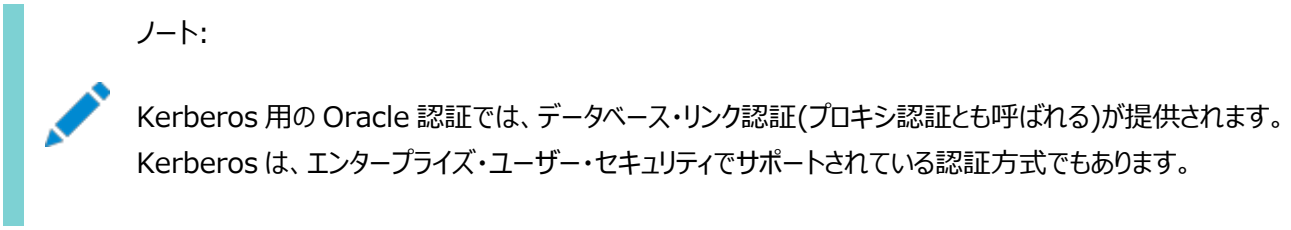
- Kerberosについて
  Oracle DatabaseはKerberosをサポートしており、Oracleユーザーにシングル・サインオンおよび集中化された認証の利点を提供します。
- Remote Authentication Dial-In User Service (RADIUS)について
  RADIUSは、最も広く知られている、リモートでの認証とアクセスを可能にするクライアント/サーバー・セキュリティ・プロトコルです。
- Transport Layer Securityについて
  Transport Layer Security (TLS)は、ネットワーク接続を保護するための業界標準プロトコルです。

親トピック: 厳密認証の概要

20.4.1 Kerberosについて

Oracle DatabaseはKerberosをサポートしており、Oracleユーザーにシングル・サインオンおよび集中化された認証の利点を提供します。

Kerberosは、共有秘密を使用するサード・パーティの認証システムです。サード・パーティがセキュアであることを保障し、シングル・サインオン機能、集中化されたパスワード・ストレージ、データベース・リンク認証、拡張されたPCセキュリティを提供します。Kerberosは、Kerberos認証サーバーを使用して認証を行います。このアダプタの構成および使用の詳細は、「Kerberos認証の構成」を参照してください。



ノート:

Kerberos 用の Oracle 認証では、データベース・リンク認証(プロキシ認証とも呼ばれる)が提供されます。Kerberos は、エンタープライズ・ユーザー・セキュリティでサポートされている認証方式でもあります。

親トピック: サポートされている厳密認証方式

20.4.2 Remote Authentication Dial-In User Service (RADIUS)について

RADIUSは、最も広く知られている、リモートでの認証とアクセスを可能にするクライアント/サーバー・セキュリティ・プロトコルです。

Oracle Databaseは、この標準をクライアント/サーバー・ネットワーク環境で使用して、RADIUSプロトコルをサポートするあらゆる認証方式の使用を可能にします。RADIUSは、トークン・カードやスマートカードなど、いろいろな認証メカニズムで使用できます。

- スマート・カード。RADIUS準拠のスマートカードは、クレジット・カードに似たハードウェア・デバイスであり、メモリーとプロセッサを備えています。クライアント・ワークステーションにあるスマートカード・リーダーで読み取られます。
- トークン・カード。トークン・カード(Secure IDまたはRADIUSに準拠)は、いくつかの異なるメカニズムによって使いやすくできます。一部のトークン・カードは、認証サービスと同期された1回かぎりのパスワードを動的に表示します。サーバーは、認証サービスにアクセスすることにより、トークン・カードによって提供されるパスワードを任意の時点で検証できます。トークン・カードにはキーパッドを持ち、チャレンジ・レスポンス・ベースで動作するものがあります。この場合、サーバーは

ユーザーがトークン・カードに入力するチャレンジ(番号)を提供します。トークン・カードはレスポンス(チャレンジから暗号的に導出される別の番号)を提供し、ユーザーはそれを入力してサーバーに送信します。

SecurIDトークンはRADIUSアダプタを介して使用できます。

関連トピック

- RADIUS認証の構成

親トピック: サポートされている厳密認証方式

20.4.3 Transport Layer Securityについて

Transport Layer Security (TLS)は、ネットワーク接続を保護するための業界標準プロトコルです。

TLSは、認証、データの暗号化およびデータの整合性を提供します。

TLSプロトコルは、公開キー・インフラストラクチャ(PKI)の基盤です。認証のために、TLSはX.509v3標準に準拠したデジタル証明書と公開キー/秘密キー・ペアを使用します。

公開キー・ページと秘密キー・ページでは、暗号化および復号化に2つの数値のセットが使用されます。1つは秘密キーと呼ばれ、もう1つは公開キーと呼ばれます。通常、公開キーは広範に使用可能であるが、秘密キーはそれぞれの所有者が保持する。数学的な関連性はあるが、一般には公開キーから秘密キーを導出するのは計算上不可能とみなされている。公開キーと秘密キーは、公開キー暗号化アルゴリズムまたは公開キー暗号方式とも呼ばれる非対称型暗号化アルゴリズムでのみ使用される。キーのペアの公開キーまたは秘密キーを使用して暗号化されたデータは、キーのペアによってそれに関連付けられているキーで復号化できます。ただし、公開キーで暗号化されたデータを同じ公開キーで復号化することはできず、秘密キーで暗号化されたデータを同じ秘密キーで復号化することはできない。

Oracle DatabaseのTLSは、任意のクライアントと任意のサーバー間の通信を保護するために使用できます。TLSは、サーバーのみ、クライアントのみ、またはクライアントとサーバーの両方に認証を提供するように構成できます。また、Oracle Databaseでサポートされている他の認証方式(データベース・ユーザー名とパスワード、RADIUSおよびKerberos)と組み合せて、TLS機能を構成することもできます。

PKI実装をサポートするために、Oracle DatabaseにはTLS以外に次の機能があります。

- Oracleウォレット(PKI資格証明を格納できる)
- Oracle Wallet Manager (Oracleウォレットを管理するために使用できる)(Oracle Wallet Managerは Oracle Database 21cでは非推奨です。Oracle Wallet Managerを使用するかわりに、コマンドライン・ツール`orapki`および`mkstore`を使用することをお薦めします。
- 証明書失効リスト(CRL)による証明書の検証
- ハードウェア・セキュリティ・モジュールのサポート

関連トピック

- Transport Layer Security認証の構成
- 厳密認証の使用のカスタマイズ

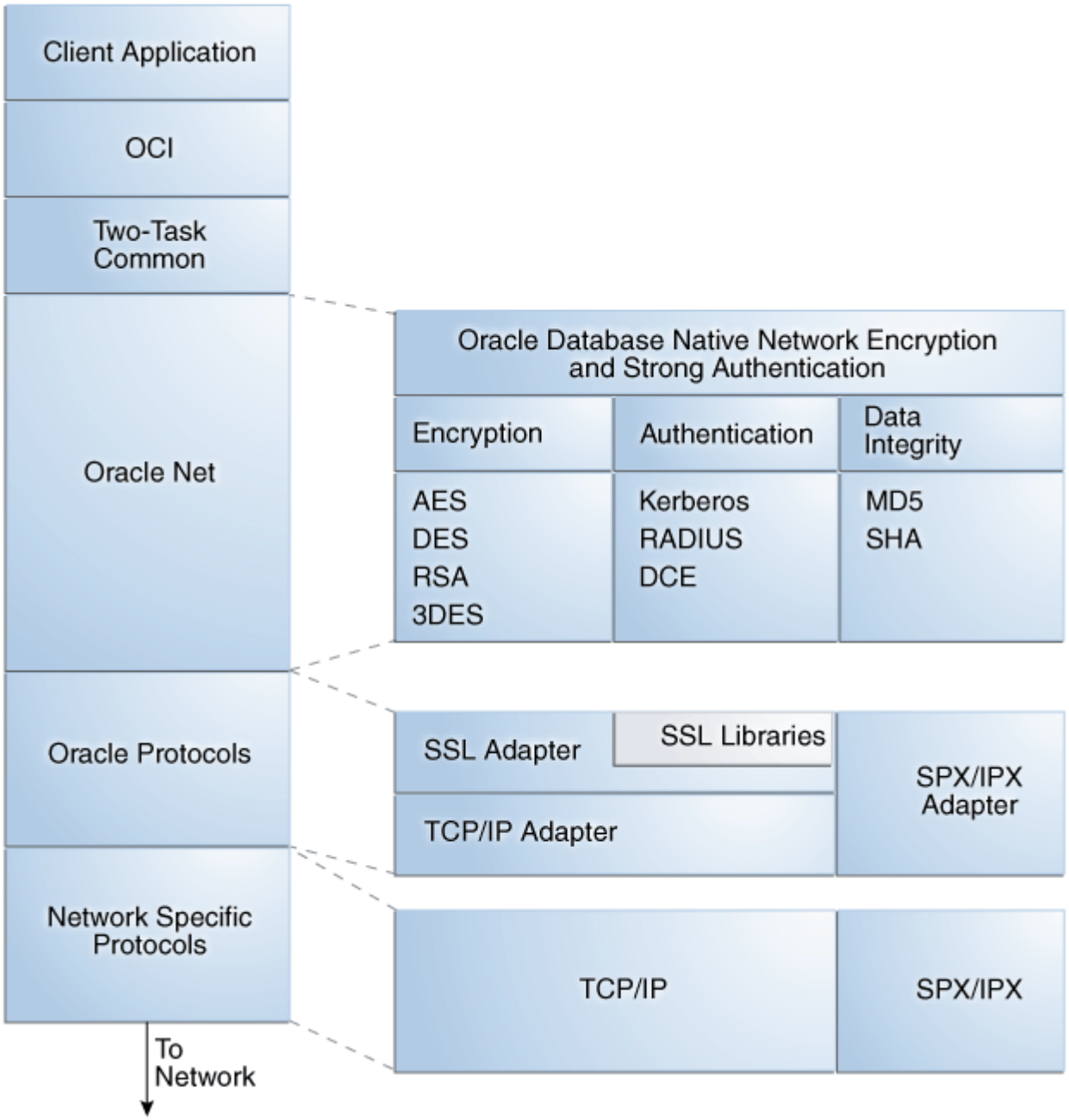
親トピック: サポートされている厳密認証方式

20.5 Oracle Databaseのネイティブ・ネットワークの暗号化/厳密認証アーキテクチャ

Oracle Databaseのネイティブ・ネットワーク暗号化および厳密認証アーキテクチャは、Oracleデータベース・サーバーまたはクライアントのインストールを補完します。

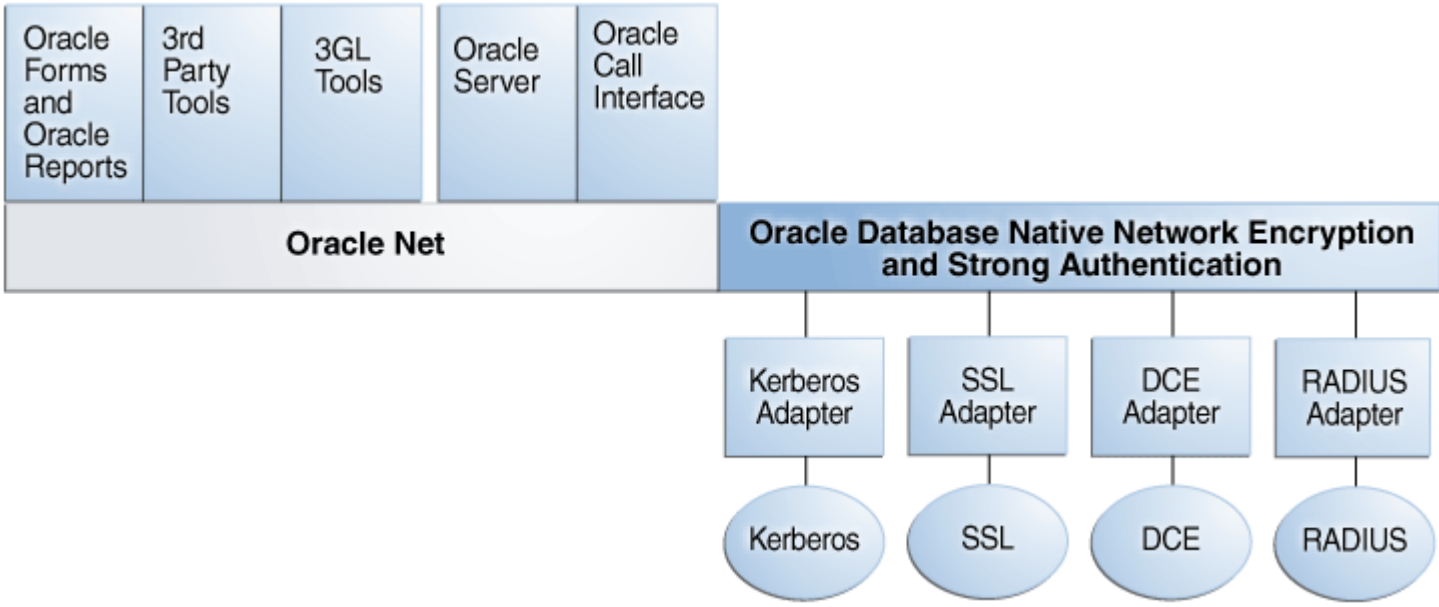
次の図は、Oracleネットワーク環境におけるこのアーキテクチャを示しています。

図20-3 Oracleネイティブ・ネットワーク暗号化および厳密認証アーキテクチャ



Oracle Databaseでは、既存のOracleプロトコル・アダプタと同様のアダプタによって認証がサポートされます。図20-4に示すように、認証アダプタによってOracle Netインタフェースが統合され、既存のアプリケーションを変更しなくてもそれらのアプリケーションで新しい認証システムを透過的に利用できるようになります。

図20-4 認証アダプタを使用したOracle Net Services



関連項目:

Oracleネットワーク環境でのスタック通信の詳細は『Oracle Database Net Services管理者ガイド』を参照してください。

親トピック: 厳密認証の概要

20.6 厳密認証のシステム要件

Kerberos、RADIUSおよびTransport Layer Security (TLS)には、厳密認証のための一連のシステム要件があります。

表20-1に、厳密認証のためのTLSシステム要件を示します。

表20-1 認証方式とシステム要件

認証方式	システム要件
Kerberos	● MIT Kerberos バージョン 5、リリース 1.8 以上。 ● Kerberos 認証サーバーを、物理的に安全なシステムにインストールする必要があります。
RADIUS	● Internet Engineering Task Force (IETF) RFC #2138 Remote Authentication Dial In User Service (RADIUS)および RFC #2139 RADIUS Accounting の規格に準拠する RADIUS サーバー。 ● チャレンジ・レスポンス認証を有効にするには、JavaSoft の Java Development Kit のリリース 1.1 で指定されている Java Native Interface をサポートするオペレーティング・システムで RADIUS を実行する必要があります。
TLS	● Oracle Database 10g と互換性のあるウォレット。

親トピック: 厳密認証の概要

20.7 Oracle Databaseのネイティブ・ネットワーク暗号化および厳密認証の制限事項

Oracleアプリケーションは、Oracle Databaseのネイティブ・ネットワーク暗号化および厳密認証をサポートしています。

ただし、Oracle Databaseのネイティブ・ネットワーク暗号化および厳密認証ではデータを安全に送信するためにOracle Net Servicesが必要であるため、Microsoft Windowsで実行されるOracleの財務管理アプリケーション、人事管理アプリケーションおよび生産管理アプリケーションの一部では、これらの外部認証機能はサポートされません。

Oracle Display Manager (ODM)ではOracle Net Servicesが使用されないため、これらの製品のODMを使用する部分では、Oracle Databaseのネイティブ・ネットワーク暗号化および厳密認証は利用されません。

親トピック: 厳密認証の概要

21 厳密認証の管理ツール

ネイティブ・ネットワーク暗号化および公開キー・インフラストラクチャ資格証明には、厳密認証の一連の管理ツールを使用できます。

- 構成ツールと管理ツールについて
  構成ツールと管理ツールで、Oracle Net Servicesの暗号化、整合性(チェックサム)および厳密認証方式を管理します。
- ネイティブ・ネットワーク暗号化ツールと厳密認証構成ツール
  Oracle Net Servicesで、標準の暗号化アルゴリズムを使用してデータを暗号化し、Kerberos、RADIUS、SSLなどの厳密認証方式を構成できます。
- 公開キー・インフラストラクチャ資格証明管理ツール
  公開キー・インフラストラクチャ(PKI)によって提供されるセキュリティは、PKI資格証明をどの程度効果的に格納、管理および検証しているかに依存します。
- 厳密認証管理者の義務
  セキュリティ管理者のほとんどのタスクでは、Oracleデータベースとの間の接続を保護する必要があります。

親トピック: 厳密認証の管理

21.1 構成ツールと管理ツールについて

構成ツールと管理ツールで、Oracle Net Servicesの暗号化、整合性(チェックサム)および厳密認証方式を管理します。

厳密認証方式の構成には、KerberosやRADIUSなどのサード・パーティ・ソフトウェアを含めることができ、Transport Layer Security (TLS)でデジタル証明書を使用するための公開キー・インフラストラクチャの構成や管理が必要になることがあります。

親トピック: 厳密認証の管理ツール

21.2 ネイティブ・ネットワーク暗号化ツールと厳密認証構成ツール

Oracle Net Servicesで、標準の暗号化アルゴリズムを使用してデータを暗号化し、Kerberos、RADIUS、SSLなどの厳密認証方式を構成できます。

- Oracle Net Managerについて
  Oracle Net Managerで、Oracle Net Servicesをローカル・クライアントやサーバー・ホスト上のOracleホーム向けに構成します。
- Kerberosアダプタ・コマンドライン・ユーティリティ
  Kerberosアダプタは、Kerberos資格証明を取得、キャッシュ、表示および削除するコマンドライン・ユーティリティを提供します。

親トピック: 厳密認証の管理ツール

21.2.1 Oracle Net Managerについて

Oracle Net Managerで、Oracle Net Servicesをローカル・クライアントやサーバー・ホスト上のOracleホーム向けに構成します。

グラフィカル・ユーザー・インタフェース・ツールであるOracle Net Managerを使用して、ネーミング、リスナー、一般的なネットワーク設定などのOracle Net Servicesを構成できますが、Oracle Netプロトコルを使用する次の機能を構成することもでき

ます。

- 厳密認証(Kerberos、RADIUSおよびTransport Layer Security)
- ネイティブ・ネットワーク暗号化(RC4、DES、3DESおよびAES)
- データ整合性のためのチェックサム(MD5、SHA-1、SHA-2)

ノート:

このリリースでは、DES、3DES112、3DES168、MD5 および RC4 アルゴリズムは非推奨です。より強力なアルゴリズムを使用するように Oracle Database 環境を移行するには、My Oracle Support ノート 2118136.2 で説明されているパッチをダウンロードしてインストールします。

親トピック: ネイティブ・ネットワーク暗号化ツールと厳密認証構成ツール

21.2.2 Kerberosアダプタ・コマンドライン・ユーティリティ

Kerberosアダプタは、Kerberos資格証明を取得、キャッシュ、表示および削除するコマンドライン・ユーティリティを提供します。

次の表では、これらのユーティリティについて簡単に説明しています。

表21-1 Kerberosアダプタ・コマンドライン・ユーティリティ

ユーティリティ名	説明
`okinit`	Key Distribution Center (KDC) から Kerberos チケットを取得し、それをユーザーの資格証明キャッシュに格納します。
`oklist`	指定された資格証明キャッシュ内の Kerberos チケットのリストを表示します。
`okdstry`	指定された資格証明キャッシュから Kerberos 資格証明を削除します。
`okcreate`	KDC またはサービス・エンドポイントから、キー表の作成を自動化します。

ノート:

Cybersafe アダプタは、このリリースからサポートされなくなりました。かわりに、Oracle の Kerberos アダプタを使用する必要があります。Kerberos アダプタを使用する場合は、Cybersafe KDC (Trust Broker)による Kerberos 認証が引き続きサポートされます。

関連トピック

- Kerberos認証アダプタのユーティリティ

親トピック: ネイティブ・ネットワーク暗号化ツールと厳密認証構成ツール

21.3 公開キー・インフラストラクチャ資格証明管理ツール

公開キー・インフラストラクチャ(PKI)によって提供されるセキュリティは、PKI資格証明をどの程度効果的に格納、管理および検証しているかに依存します。

- Oracle Wallet Managerについて
  ウォレットの所有者とセキュリティ管理者はOracle Wallet Managerを使用して、Oracleウォレット内のセキュリティ資格証明を管理および編集します。
- orapkiユーティリティについて
  `orapki`ユーティリティで、証明書失効リスト(CRL)の管理、Oracleウォレットの作成と管理、および署名付き証明書の作成を行います。

親トピック: 厳密認証の管理ツール

21.3.1 Oracle Wallet Managerについて

ウォレットの所有者とセキュリティ管理者はOracle Wallet Managerを使用して、Oracleウォレット内のセキュリティ資格証明を管理および編集します。

ウォレットはパスワードで保護されたコンテナで、認証情報や、Transport Layer Securityに必要な秘密キー、証明書および信頼できる証明書を含む署名証明書が格納されます。Oracle Wallet Managerを使用して次のタスクを実行できます。

- 公開キーと秘密キーのペアの生成
- ユーザー資格証明書の格納および管理
- 証明書リクエストの生成
- 資格証明の格納および管理(ルート・キー証明書および証明連鎖)
- ウォレットのLDAPディレクトリへのアップロードおよびLDAPディレクトリからのダウンロード
- ハードウェア・セキュリティ・モジュールの資格証明書を格納するウォレットの作成

ノート:

Oracle Database の以前のリリースでは、Oracle Wallet Manager を使用して、透過的データ暗号化のウォレットを構成できます。このリリースでは、かわりに `ADMINISTER KEY MANAGEMENT SQL` 文を使用できます。

関連トピック

- Oracle Database Advanced Securityガイド

親トピック: 公開キー・インフラストラクチャ資格証明管理ツール

21.3.2 orapkiユーティリティについて

`orapki`ユーティリティで、証明書失効リスト(CRL)の管理、Oracleウォレットの作成と管理、および署名付き証明書の作成を行います。

このコマンドライン・ユーティリティの基本構文は次のとおりです。

```
orapki module command -option_1 argument ... -option_n argument
```

たとえば、次のコマンドを実行すると、machine1.us.example.comにインストールされ、ポート389を使用するOracle Internet Directoryのインスタンスの証明書失効リスト(CRL)サブツリーにあるすべてのCRLがリストされます。

```
orapki crl list -ldap machine1.us.example.com:389
```

ノート:

透過的データ暗号化を構成する orapki の使用は非推奨です。かわりに、ADMINISTER KEY MANAGEMENT SQL 文を使用します。

関連トピック

- 証明書失効リストの管理
- 公開キー・インフラストラクチャ(PKI)要素の管理

親トピック: 公開キー・インフラストラクチャ資格証明管理ツール

21.4 厳密認証管理者の義務

セキュリティ管理者のほとんどのタスクでは、Oracleデータベースとの間の接続を保護する必要があります。

次の表に、厳密認証、タスクの実行に使用するツール、およびタスクのドキュメントの場所へのリンクに対して責任を負うセキュリティ管理者の主要なタスクを示します。

表21-2 セキュリティ管理者/DBAの一般的な構成および管理タスク

タスク	使用するツール	関連項目
データベース・サーバーとクライアント間の暗号化された Oracle Net 接続を構成します。	Oracle Net Manager	クライアントとサーバーでの暗号化の構成
データベース・サーバーとクライアント間の Oracle Net 接続のチェックサムを構成します。	Oracle Net Manager	クライアントとサーバーでの整合性の構成
RADIUS 認証を受け入れるようにデータベース・クライアントを構成します。	Oracle Net Manager	ステップ 1A: Oracle クライアントでの RADIUS の構成
RADIUS 認証を受け入れるようにデータベースを構成します。	Oracle Net Manager	ステップ 1B: Oracle データベース・サーバーでの RADIUS の構成
RADIUS ユーザーを作成し、データベース・セッションへのアクセス権を付与します。	SQL*Plus	ステップ 2: ユーザーの作成とアクセス権の付与

タスク	使用するツール	関連項目
データベース・クライアントとサーバーに Kerberos 認証を構成します。	Oracle Net Manager	ステップ 6: Kerberos 認証の構成
Kerberos データベース・ユーザーを作成します。	• `kadmin.local` • Oracle Net Manager	• ステップ 7: Kerberos ユーザーの作成 • ステップ 8: 外部認証された Oracle ユーザーの作成
資格証明キャッシュ内の Kerberos 資格証明を管理します。	• `okinit` • `oklist` • `okdstry` • `okcreate`	• 初期チケットを取得するための okinit ユーティリティ・オプション • 資格証明を表示するための oklist ユーティリティ・オプション • キャッシュ・ファイルから資格証明を削除するための okdstry ユーティリティのオプション
データベース・クライアントまたはサーバーのウォレットを作成します。	Oracle Wallet Manager	Oracle Database エンタープライズ・ユーザー・セキュリティ管理者ガイド
認証局(CA)に TLS 認証用のユーザー証明書をリクエストします	Oracle Wallet Manager	• 証明書リクエストを追加するには、『Oracle Database エンタープライズ・ユーザー・セキュリティ管理者ガイド』を参照してください。 • Oracle ウォレットにユーザー証明書をインポートするには、『Oracle Database エンタープライズ・ユーザー・セキュリティ管理者ガイド』を参照してください。
ユーザー証明書とそれに関連する信頼できる証明書(CA 証明書)をウォレットにインポートします。	Oracle Wallet Manager	• 信頼できる証明書をインポートするには、『Oracle Database エンタープライズ・ユーザー・セキュリティ管理者ガイド』を参照してください。 • Oracle ウォレットにユーザー証明書をインポートするには、『Oracle

タスク	使用するツール	関連項目
		Database エンタープライズ・ユーザー・セキュリティ管理者ガイド』を参照してください。
データベース・クライアントの TLS 接続を構成します	Oracle Net Manager	ステップ 2: クライアントでの Transport Layer Security の構成
データベース・サーバーの TLS 接続を構成します	Oracle Net Manager	ステップ 1: サーバーでの Transport Layer Security の構成
証明書失効リスト(CRL)を使用した証明書の検証を有効化します。	Oracle Net Manager	証明書失効リストによる証明書検証の構成

親トピック: 厳密認証の管理ツール

22 Kerberos認証の構成

Kerberosは信頼できるサード・パーティ認証システムであり、共有秘密鍵に基づき、サード・パーティがセキュアであることを前提とします。

- Kerberos認証の有効化
  Oracle Databaseに対してKerberos認証を有効にするには、これをインストールしてから一連の構成ステップに従います。
- Kerberos認証アダプタのユーティリティ
  Oracle Kerberos認証アダプタのユーティリティは、Oracle Kerberos認証サポートがインストールされたOracleクライアントで使用するように設計されています。
- Kerberosによって認証されたOracleデータベース・サーバーへの接続
  Kerberosの構成後は、ユーザー名やパスワードを使用しないでOracleデータベース・サーバーに接続できます。
- Windows 2008ドメイン・コントローラKDCとの相互運用性の構成
  Oracle DatabaseをMicrosoft Windows 2008ドメイン・コントローラのキー配布センター(KDC)と相互作用するように構成できます。
- Kerberos認証フォールバック動作の構成
  Kerberos認証の障害に備え、フォールバック動作(パスワードベースの認証)を構成できます。
- Oracle Kerberos認証の構成のトラブルシューティング
  一般的なKerberos構成の問題に対するガイドラインを示します。

親トピック: 厳密認証の管理

22.1 Kerberos認証の有効化

Oracle Databaseに対してKerberos認証を有効にするには、これをインストールしてから一連の構成ステップに従います。

- ステップ1: Kerberosのインストール
  Kerberosバージョン5をインストールしてください。
- ステップ2: Oracleデータベース・サーバーに対するサービス・プリンシパルの構成
  Kerberosを使用して自己を認証するクライアントの識別情報をOracleデータベース・サーバーで検証できるようにするには、Oracle Databaseのサービス・プリンシパルを作成する必要があります。
- ステップ3: Kerberosからのサービス・キー表の抽出
  次に、Kerberosからサービス・キー表を抽出し、Oracleデータベース・サーバー/Kerberosクライアント・システムにコピーします。
- ステップ4: OracleデータベースとOracleクライアントのインストール
  Kerberosからサービス・キー表を抽出した後、OracleデータベースおよびOracleクライアントをインストールします。
- ステップ5: Oracle Net ServicesとOracle Databaseの構成
  Oracleデータベース・サーバーとクライアントのインストール後、サーバーとクライアントでOracle Net Servicesを構成できます。
- ステップ6: Kerberos認証の構成
  Oracleデータベース・サーバーおよびクライアントの`sqlnet.ora`ファイルで必須パラメータを設定する必要があります。
- ステップ7: Kerberosユーザーの作成
  管理ツールがインストールされているKerberos認証サーバーで、Kerberosユーザーを作成する必要があります。

- ステップ8: 外部認証されたOracleユーザーの作成
  次に、外部認証されるOracleユーザーを作成します。
- ステップ9: Kerberos/Oracleユーザーの初期チケットの取得
  データベースに接続するには、Key Distribution Center (KDC)に初期チケットを要求する必要があります。

関連項目:

KerberosユーザーをKerberos認証のエンタープライズ・ユーザーに移行する方法の詳細は、『Oracle Databaseエンタープライズ・ユーザー・セキュリティ管理者ガイド』を参照してください。

親トピック: Kerberos認証の構成

22.1.1 ステップ1: Kerberosのインストール

Kerberosバージョン5をインストールしてください。

Kerberosの構築とインストールに関するノートの供給元配布資料に詳細が記載されています。Kerberosのインストール後、POWERシステム(64ビット)でIBM AIXを使用する場合、Kerboros 5が推奨の認証方式であることを確認する必要があります。

1. 認証サーバーとして動作するシステムにKerberosをインストールします。

   ノート:

   32 ビット・バージョンの Oracle Database からアップグレードした後に初めて Kerberos 認証アダプタを使用すると、エラー・メッセージ「ORA-01637: パケット受信に失敗しました。」が表示されます。

   回避策: 64 ビット・バージョンのデータベースにアップグレードした後、Kerberos 外部認証方式を使用する前に、コンピュータ上の/usr/tmp/oracle_service_name.RC という名前のファイルを確認して削除します。

2. POWERシステム(64ビット)上のIBM AIXの場合、認証方式をチェックします。

   たとえば:

   ```
   /usr/bin/lsauthent
   ```

   次のような出力が表示されます。

   ```
   Standard Aix
   ```

3. Kerberos 5を推奨方式として構成します。

   たとえば:

   ```
   /usr/bin/chauthent -k5 -std
   ```

   このコマンドは、Kerberos 5を推奨方式(k5)、標準AIXを2番目の方式(std)として設定します。

4. Kerberos 5が現在推奨方式になっていることを確認するには、新しい構成をチェックします。

   ```
   /usr/bin/lsauthent
   ```

```
Kerberos 5
Standard Aix
```

親トピック: Kerberos認証の有効化

22.1.2 ステップ2: Oracleデータベース・サーバーに対するサービス・プリンシパルの構成

Kerberosを使用して自己を認証するクライアントの識別情報をOracleデータベース・サーバーで検証できるようにするには、Oracle Databaseのサービス・プリンシパルを作成する必要があります。

1. 次の形式を使用して、サーバー・プリンシパルの名前を決定します。

   ```
   kservice/kinstance@REALM
   ```

   サービス・プリンシパル内の各フィールドで次の値を指定します。

サービス・プリンシパル・フィールド	説明
`kservice`	Oracle サービスを表す、大/小文字を区別する文字列。データベース・サービス名と同じでもかまいません。
`kinstance`	通常は、Oracle Database が実行されているシステムの完全修飾 DNS 名。
`REALM`	サービス・プリンシパルが登録されている Kerberos レルムの名前。`REALM` は常に大文字である必要があり、通常は DNS ドメイン名です。

   この項のユーティリティ名は実行可能プログラムです。ただし、Kerberosユーザー名`krbuser`およびレルム`EXAMPLE.COM`は単なる例です。

   たとえば、`kservice`が`oracle`、Oracle Databaseが実行されているシステムの完全修飾名を`dbserver.example.com`、レルムを`EXAMPLE.COM`とします。この場合、プリンシパル名は次のようになります。

   ```
   oracle/dbserver.example.com@EXAMPLE.COM
   ```

2. `kadmin.local`を実行してサーバー・プリンシパルを作成します。UNIXでは、次の構文を使用して、rootユーザーとしてこのコマンドを実行します。

   ```
   # cd /kerberos-install-directory/sbin
   # ./kadmin.local
   ```

   たとえば、`oracle/dbserver.example.com@EXAMPLE.COM`という名前のプリンシパルを、Kerberosが認識するサーバー・プリンシパルのリストに追加するには、次のように入力します。

   ```
   kadmin.local:addprinc -randkey oracle/dbserver.example.com@EXAMPLE.COM
   ```

親トピック: Kerberos認証の有効化

22.1.3 ステップ3: Kerberosからのサービス・キー表の抽出

次に、Kerberosからサービス・キー表を抽出し、Oracleデータベース・サーバー/Kerberosクライアント・システムにコピーします。

たとえば、`dbserver.example.com`のサービス・キー表を抽出するには、次の手順を実行します。

1. ドメイン管理権限があることを確認します。
2. 次のように入力してサービス・キー表を抽出します。

```
kadmin.local:  ktadd -k /tmp/keytab oracle/dbserver.example.com
Entry for principal oracle/dbserver.example.com with kvno 2,
encryption type AES-256 CTS mode with 96-bit SHA-1 HMAC added to keytab WRFILE:
WRFILE:/tmp/keytab
kadmin.local:  exit
```

3. サービス・キー表をチェックするには、次のコマンドを入力します。

```
oklist -k -t /tmp/keytab
```

4. サービス・キー表を抽出した後、古いエントリに加えて新しいエントリが表にあることを確認します。

   新しいエントリがない場合、またはさらに追加する必要がある場合は、`kadmin.local`を使用して追加します。

   `ktadd`を使用するときにレルムを入力しない場合、Kerberosサーバーのデフォルトのレルムが使用されます。`kadmin.local`は、`localhost`で実行されているKerberosサーバーに接続されます。

5. Kerberosサービス・キー表がKerberosクライアントと同じシステム上にある場合は、移動できます。Kerberosサービス・キー表がKerberosクライアントと異なるシステム上にある場合は、FTPなどのプログラムを使用してファイルを転送する必要があります。FTPを使用する場合は、ファイルをバイナリ・モードで転送します。

   次の例は、UNIXプラットフォームでサービス・キー表を移動する方法を示しています。

```
# mv /tmp/keytab /etc/v5srvtab
```

   サービス・ファイルのデフォルト名は、`/etc/v5srvtab`です。

6. Oracleデータベース・サーバー実行可能ファイルの所有者がサービス・キー表(前の例の`/etc/v5srvtab`)を読み取ることができることを確認します。

   そのためには、ファイル所有者をOracleユーザーに設定するか、またはファイルをOracleが属するグループに対して読取り可能にします。

   ファイルをすべてのユーザーに対して読取り可能にしないでください。そのことによってセキュリティ侵害が発生する場合があります。

親トピック: Kerberos認証の有効化

22.1.4 ステップ4: Oracleデータベース・サーバーとOracleクライアントのインストール

Kerberosからサービス・キー表を抽出した後、Oracleデータベース・サーバーおよびクライアントをインストールします。

- Oracleデータベース・サーバーおよびクライアント・ソフトウェアのインストールの詳細は、Oracle Databaseオペレーティング・システムに固有のインストール・ドキュメントを参照してください。

親トピック: Kerberos認証の有効化

22.1.5 ステップ5: Oracle Net ServicesとOracle Databaseの構成

Oracleデータベース・サーバーとクライアントのインストール後、サーバーとクライアントでOracle Net Servicesを構成できます。

- Oracleデータベース・サーバーおよびクライアントでの Oracle Net Servicesの構成の詳細は、次のドキュメントを参照してください。

- オペレーティング・システム固有のOracle Databaseインストール関連ドキュメント
- Oracle Database Net Services管理者ガイド

親トピック: Kerberos認証の有効化

22.1.6 ステップ6: Kerberos認証の構成

Oracleデータベース・サーバーおよびクライアントの`sqlnet.ora`ファイルで必須パラメータを設定する必要があります。

> ノート:
>
> マルチテナント環境では、`sqlnet.ora` ファイルの設定はすべてのプラガブル・データベース(PDB)に適用されることに注意してください。ただし、Kerberos を使用する場合は、すべての PDB を 1 つの KDC で認証する必要があるという意味ではありません。`sqlnet.ora` ファイルおよび Kerberos 構成ファイルの設定では複数の KDC をサポートできます。

- ステップ6A: クライアントとデータベース・サーバーでのKerberosの構成
  最初に、クライアントとデータベース・サーバーでKerberos認証サービス・パラメータを構成する必要があります。
- ステップ6B: 初期化パラメータの設定
  次に、`OS_AUTHENT_PREFIX`初期化パラメータを設定します。
- ステップ6C: sqlnet.oraパラメータの設定(オプション)
  セキュリティを強化するために、必須パラメータに加えて、オプションの`sqlnet.ora`パラメータを設定できます。

親トピック: Kerberos認証の有効化

22.1.6.1 ステップ6A: クライアントとデータベース・サーバーでのKerberosの構成

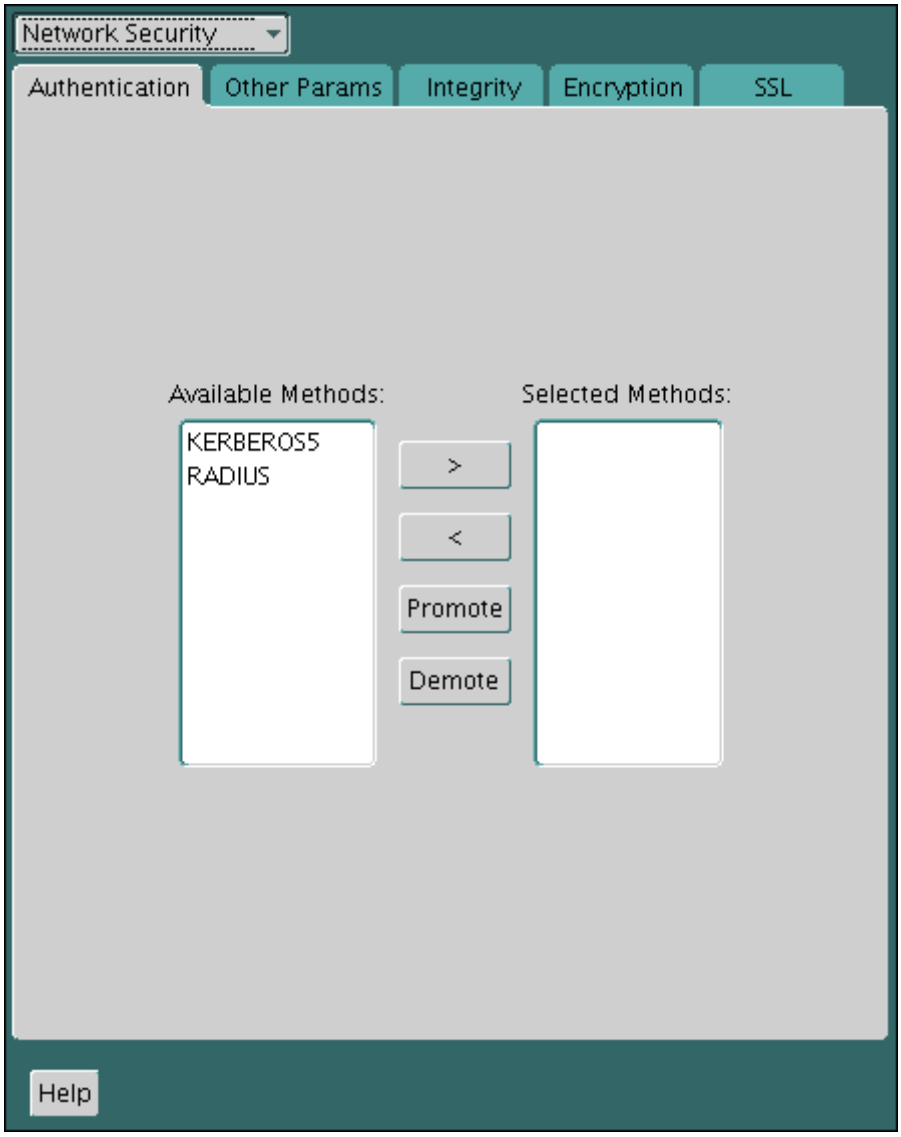
最初に、クライアントとデータベース・サーバーでKerberos認証サービス・パラメータを構成する必要があります。

1. Oracle Net Managerを起動します。
   - (UNIX) `$ORACLE_HOME/bin`から、コマンドラインで次のコマンドを入力します。

     ```
     netmgr
     ```

   - (Windows)「スタート」→「プログラム」→「Oracle - HOME_NAME」→「Configuration and Migration Tools」→「Net Manager」を選択します。
2. 「Oracle Netの構成」を展開し、「ローカル」から「プロファイル」を選択します。
3. 「ネーミング」リストから、「ネットワーク・セキュリティ」を選択します。

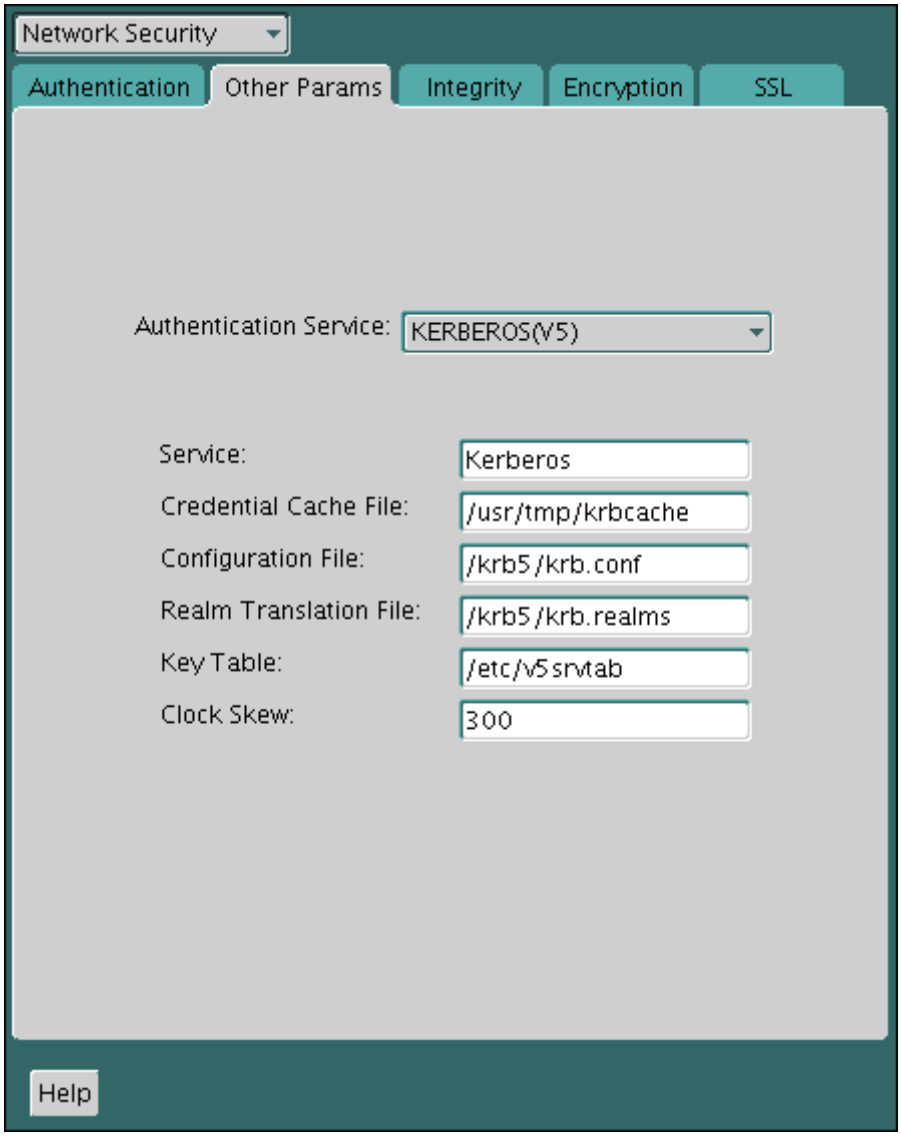
   ネットワーク・セキュリティのタブ付きウィンドウが表示されます。
4. 「認証」タブを選択します。



5. 「使用可能なメソッド」リストから、「KERBEROS5」を選択します。
   Kerberosクロスレルム認証はKERBEROS5またはKERKBEROS5PREアダプタによる制約委任の使用ではサポートされないということに注意してください。
6. 右矢印(＞)をクリックして、「KERBEROS5」を「選択メソッド」リストに移動します。
7. 選択したメソッドを使用する順に並べます。

   そのためには、「選択メソッド」リストでメソッドを選択し、「上へ」または「下へ」をクリックしてリスト内に配置します。たとえば、最初に使用するサービスをKERBER0S5にするには、リストの先頭に移動します。

8. 「その他のパラメータ」タブを選択します。
9. 「認証サービス」リストから「KERBEROS(V5)」を選択します。
10. 「サービス」フィールドに「Kerberos」と入力します。



このフィールドでは、Kerberosのサービス・チケットを取得するためにOracle Databaseが使用するサービスの名前を定義します。このフィールドに値を入力すると、他のフィールドに入力できるようになります。

11. オプションで、次のフィールドの値を入力します。
    - 資格証明キャッシュ・ファイル
    - 構成ファイル
    - レルム変換ファイル
    - キー表
    - 時間誤差

    構成するフィールドとパラメータの詳細は、Oracle Net Managerオンライン・ヘルプおよび「ステップ6C: sqlnet.oraパラメータの設定(オプション)」を参照してください。

12. 「ファイル」メニューから、「ネットワーク構成の保存」を選択します。

    sqlnet.oraファイルは、前述のステップで行った可能性のあるオプションの選択だけでなく、次のエントリでも更新されます。

```
SQLNET.AUTHENTICATION_SERVICES=(KERBEROS5)
SQLNET.AUTHENTICATION_KERBEROS5_SERVICE=kservice
```

親トピック: ステップ6: Kerberos認証の構成

22.1.6.2 ステップ6B: 初期化パラメータの設定

次に、OS_AUTHENT_PREFIX初期化パラメータを設定します。

1. init.oraファイルを探します。

   デフォルトでは、init.oraファイルは、LinuxおよびUNIXシステムの場合はORACLE_HOME/dbsディレクトリ(または同じデータ・ファイルの場所)、Windowsの場合はORACLE_HOME¥databaseディレクトリにあります。

2. init.oraファイルで、init.ora初期化パラメータ・ファイルのOS_AUTHENT_PREFIXの値をnullに設定します。

   たとえば:

   ```
   OS_AUTHENT_PREFIX=""
   ```

   Kerberosユーザー名は長くてもかまわないためこの値をnullに設定しますが、Oracleユーザー名は30バイトまでに制限されています。このパラメータをnullに設定すると、OPS$のデフォルト値が上書きされます。

ノート:

30 バイトを超える Kerberos ユーザー名を持つ外部データベース・ユーザーを作成できます。詳細は、「ステップ 8: 外部認証された Oracle ユーザーの作成」を参照してください。

親トピック: ステップ6: Kerberos認証の構成

22.1.6.3 ステップ6C: sqlnet.oraパラメータの設定(オプション)

セキュリティを強化するために、必須パラメータに加えて、オプションのsqlnet.oraパラメータを設定できます。

- オプションで、クライアントおよびOracleデータベース・サーバーの両方に、次の表に示すパラメータを設定します。

表22-1 Kerberos固有のsqlnet.oraパラメータ

パラメータ	説明
SQLNET.KERBEROS5_CC_NAME=pathname_to_credentials_cache_file|OS_MEMORY	Kerberos 資格証明キャッシュ(CC)ファイルへの完全パス名を指定します。デフォルト値は、オペレーティング・システムによって異なります。UNIX では、/tmp/krb5cc_userid です。 OS_MEMORY オプションを使用すると、OS 管理メモリーの資格証明キャッシュが資格証明キャッシュ・ファイルに使用されるように指定されます。このオプションは、すべてのプラットフォームでサポートされます。 SQLNET.KERBEROS5_CC_NAME の値は、次の形式を使用して指定できます。 ● SQLNET.KERBEROS5_CC_NAME=complete_path_to_cc_file たとえば: SQLNET.KERBEROS5_CC_NAME=/tmp/kcache

パラメータ	説明
	SQLNET.KERBEROS5_CC_NAME=D:¥tmp¥kcache ● SQLNET.KERBEROS5_CC_NAME=FILE:complete_path_to_cc_file たとえば: SQLNET.KERBEROS5_CC_NAME=FILE:/tmp/kcache ● SQLNET.KERBEROS5_CC_NAME=OSMSFT:// Windows を実行し、Microsoft KDC を使用している場合は、この値を使用します。 このパラメータは KRB5CCNAME 環境変数を使用して設定することもできますが、sqlnet.ora ファイルで設定する値は、KRB5CCNAME で設定する値よりも優先されます。 たとえば: SQLNET.KERBEROS5_CC_NAME=/usr/tmp/krbcache
SQLNET.KERBEROS5_CLOCKSKEW=number_of_seconds_accepted_as_network_delay	このパラメータは、Kerberos 資格証明を期限切れとみなすまでの秒数を指定します。これは、資格証明がクライアントまたはデータベース・サーバーによって実際に受け取られるときに使用されます。また、再生攻撃を受けないように資格証明を格納する必要があるかどうかを Oracle データベース・サーバーが判断するときにも、使用されます。デフォルトは 300 秒です。 たとえば: SQLNET.KERBEROS5_CLOCKSKEW=1200
SQLNET.KERBEROS5_CONF=pathname_to_Kerberos_configuration_file|AUTO_DISCOVER	このパラメータは、Kerberos 構成ファイルへの完全パス名を指定します。構成ファイルには、デフォルトの KDC (key distribution center)のレルムが含まれており、レルムを KDC ホストにマッピングします。デフォルトは、オペレーティング・システムによって異なります。UNIX では、/krb5/krb.conf です。 構成ファイルのかわりに AUTO_DISCOVER オプションを使用すると、Kerberos クライアントで KDC を自動検出できます。 たとえば: SQLNET.KERBEROS5_CONF=/krb/krb.conf SQLNET.KERBEROS5_CONF=AUTO_DISCOVER

パラメータ	説明
SQLNET.KERBEROS5_CONF_LOCATION=path_to_Kerberos_configuration_directory	このパラメータは、Kerberos 構成ファイルがシステムによって作成され、クライアントで指定する必要がないことを示します。構成ファイルは、DNS 参照を使用してデフォルトの KDC のレルムを取得し、レルムを KDC ホストにマッピングします。 たとえば: SQLNET.KERBEROS5_CONF_LOCATION=/krb
SQLNET.KERBEROS5_KEYTAB=path_to_Kerberos_principal/key_table	このパラメータは、Kerberos プリンシパル/秘密キー・マッピング・ファイルへの完全パス名を指定します。これは、Oracle データベース・サーバーがキーを抽出し、クライアントから受信する認証情報を複号化するために使用されます。デフォルトは、オペレーティング・システムによって異なります。UNIX では、/etc/v5srvtab です。 たとえば: SQLNET.KERBEROS5_KEYTAB=/etc/v5srvtab
SQLNET.KERBEROS5_REALMS=path_to_Kerberos_realm_translation_file	このパラメータは、Kerberos レルム変換ファイルへの完全パス名を指定します。変換ファイルを使用して、ホスト名またはドメイン名をレルムにマッピングします。デフォルトは、オペレーティング・システムによって異なります。UNIX では、/etc/krb.realms です。 たとえば: SQLNET.KERBEROS5_REALMS=/krb5/krb.realms

親トピック: ステップ6: Kerberos認証の構成

22.1.7 ステップ7: Kerberosユーザーの作成

管理ツールがインストールされているKerberos認証サーバーで、Kerberosユーザーを作成する必要があります。

レルムはすでに存在している必要があります。

ノート:

この項のユーティリティ名は実行可能プログラムです。ただし、Kerberos ユーザー名 krbuser およびレルム EXAMPLE.COM は単なる例です。システムによって異なる場合があります。

- /krb5/admin/kadmin.localをrootとして実行して、krbuserなどの新しいKerberosユーザーを作成します。

  たとえば、UNIX固有のKerberosユーザーを作成するとします。

```
# /krb5/admin/kadmin.local
kadmin.local: addprinc krbuser
Enter password for principal: "krbuser@example.com": (password does not
display)
Re-enter password for principal: "krbuser@example.com": (password does not
display)
kadmin.local: exit
```

親トピック: Kerberos認証の有効化

22.1.8 ステップ8: 外部認証されたOracleユーザーの作成

次に、外部認証されるOracleユーザーを作成します。

1. CREATE USER権限を持つユーザーとしてSQL*Plusにログインします。

   ```
   sqlplus sec_admin - Or, CONNECT sec_admin@hrpdb
   Enter password: password
   ```

2. OS_AUTHENT_PREFIXがnull ("")に設定されていることを確認します。
3. Kerberosユーザーに対応するOracle Databaseユーザー・アカウントを作成します。Oracleユーザー名を大文字で入力して、その名前を二重引用符で囲みます。

   たとえば:

   ```
   CREATE USER krbuser IDENTIFIED EXTERNALLY AS 'krbuser@example.com';
   GRANT CREATE SESSION TO krbuser;
   ```

ノート:

データベース管理者は、2 つのデータベース・ユーザーが同じ Kerberos プリンシパル名で外部から識別されないことを確認する必要があります。

親トピック: Kerberos認証の有効化

22.1.9 ステップ9: Kerberos/Oracleユーザーの初期チケットの取得

データベースに接続するには、Key Distribution Center (KDC)に初期チケットを要求する必要があります。

- 初期チケットを要求するには、クライアントで次のコマンドを実行します。

  ```
  % okinit username
  ```

  データベース・リンク間で使用可能な資格証明を有効にするには、-fオプションを含めて、プロンプトが表示された場合はKerberosパスワードを指定します。

  ```
  % services/okinit -f
  Password for krbuser@EXAMPLE.COM:(password does not display)
  ```

okinit: Cannot contact any KDC for requested realmなどのエラーが発生した場合は、kerberos 5エントリがあるかどうか/etc/servicesファイルを確認します。たとえば:

```
kerberos        88/tcp          kerberos5 krb5  # Kerberos v5
kerberos        88/udp          kerberos5 krb5  # Kerberos v5
```

親トピック: Kerberos認証の有効化

22.2 Kerberos認証アダプタのユーティリティ

Oracle Kerberos認証アダプタのユーティリティは、Oracle Kerberos認証サポートがインストールされたOracleクライアントで使用するように設計されています。

- 初期チケットを取得するためのokinitユーティリティ・オプション
  okinitユーティリティでは、Kerberosチケットを取得し、キャッシュします。
- 資格証明を表示するためのoklistユーティリティ・オプション
  oklistユーティリティは保持しているチケットのリストを表示します。
- キャッシュ・ファイルから資格証明を削除するためのokdstryユーティリティのオプション
  okdstry (okdestroy)ユーティリティはキャッシュ・ファイルから資格証明を削除します。
- キー表の作成を自動化するためのokcreateユーティリティのオプション
  okcreateユーティリティは、KDCまたはサービス・エンドポイントから、キー表の作成を自動化します。

親トピック: Kerberos認証の構成

22.2.1 初期チケットを取得するためのokinitユーティリティ・オプション

okinitユーティリティでは、Kerberosチケットを取得し、キャッシュします。

通常は、このユーティリティを使用してチケット認可チケットを取得し、ユーザーが入力したパスワードを使用してkey distribution center (KDC)からの資格証明を復号化します。チケット認可チケットは、ユーザーの資格証明キャッシュに格納されます。

次の表に、okinitで使用できるオプションを示します。表に示されている機能を使用するには、sqlnet.oraのSQLNET.KERBEROS5_CONF_MITパラメータをTRUEに設定する必要があります。(SQLNET.KERBEROS5_CONF_MITは非推奨ですが、okinitの下位互換性のため現在も維持されていることに注意してください。)

表22-2 okinitユーティリティのオプション

オプション	説明
-f | -F	転送可能なチケットまたは転送不可のチケットを要求します。データベース・リンクをたどる場合は、このオプションが必要です。
-l lifetime	チケット認可チケットおよびすべての後続チケットの存続期間を指定します。デフォルトで、チケット認可チケットは 8 時間有効ですが、存続期間がより短いまたは長い資格証明を指定することもできます。KDC はこのオプションを無視するか、各サイトで指定できる時間を制限することができます。次の例に示すように、存続期間の値は、w (週)、d (日)、h (時間)、m (分)または s (秒)で修飾された数字で構成される文字列です。 `okinit -l 2wld6h20m30s` この例では、存続期間が 2 週間と 1 日 6 時間 20 分 30 秒のチケット認可チケットが要求されます。
-s start_time	チケットが有効になるまでの遅延の期間を指定します。チケットは、無効なフラグ・セットを使用し

オプション	説明
	て発行されます。
-r renewable_life	合計存続期間が renewable_life の更新可能なチケットを要求します
-p | -P	プロキシ可能なチケットまたはプロキシ不可のチケットを要求します
-a	ホストのローカル・アドレスに制限されているチケットを要求します
-A	アドレスによって制限されていないチケットを要求します
-E	プリンシパル名を企業名として扱います
-v	キャッシュ内のチケット認可チケットを KDC に渡して検証するよう要求します。チケットが要求された期間内のものであれば、キャッシュは検証済のチケットで置き換えられます。
-R	チケット認可チケットの更新を要求します
-k [-t keytab_file]	ローカル・ホストのキー表内のキーから取得されたチケットを要求します
-n	匿名処理を要求します
-C	プリンシパル名の正規化を要求し、要求されたものとは異なるクライアント・プリンシパルで KDC が応答できるようにします
-c cache_name	キャッシュの名前をキャッシュの場所として指定します。UNIX では、デフォルトは /tmp/krb5cc_uid です。代替資格証明キャッシュは、sqlnet.ora ファイルで SQLNET.KERBEROS5_CC_NAME パラメータを使用して指定することもできます。
-I input_cache	すでにチケットが含まれている資格証明キャッシュの名前を指定します。そのチケットを取得する際、チケットを取得した方法に関する情報がキャッシュ内に格納されていれば、同じ情報を使用して、新しい資格証明を取得する方法に影響を及ぼします。
-T armor_cache	KDC でサポートされている場合、このキャッシュは要求を保護するために使用され、オフラインの辞書攻撃を防ぎ、追加で事前認証メカニズムを使用することを可能にします。
-X attribute[=value	事前認証属性と値を指定します。次のいずれかの値を指定します。 ● X509_user_identity=value は、ユーザーの X509 識別情報を確認する場所を指定します ● X509_anchors=value は、信頼できる X509 アンカー情報を確認する場所を指

オプション	説明
	定します ● flag_RSA_PROTOCOL[=yes]は、デフォルトの Diffie-Hellman プロトコルではなく、RSA の使用を指定します
-?	コマンドライン・オプションのリストを表示します。

親トピック: Kerberos認証アダプタのユーティリティ

22.2.2 資格証明を表示するためのoklistユーティリティ・オプション

oklistユーティリティは保持しているチケットのリストを表示します。

次の表に、oklistの使用可能なオプションを示します。表に示されている機能を使用するには、sqlnet.oraのSQLNET.KERBEROS5_CONF_MITパラメータをTRUEに設定する必要があります。(SQLNET.KERBEROS5_CONF_MITは非推奨ですが、oklistの下位互換性のため現在も維持されていることに注意してください。)

表22-3 oklistユーティリティのオプション

オプション	説明
-f	資格証明のフラグを表示します。関連するフラグは次のとおりです。 ● I: 資格証明がチケット認可チケットです。 ● F: 資格証明が転送可能です。 ● f: 資格証明が転送済です。
-c	代替資格証明キャッシュを指定します。UNIX では、デフォルトは/tmp/krb5cc_uid です。代替資格証明キャッシュは、sqlnet.ora ファイルで SQLNET.KERBEROS5_CC_NAME パラメータを使用して指定することもできます。
-k	UNIX 上のサービス表のエントリ(デフォルト/etc/v5srvtab)をリストします。代替サービス表は、sqlnet.ora ファイルで SQLNET.KERBEROS5_KEYTAB パラメータを使用して指定することもできます。
-e	資格証明キャッシュ内の各資格証明のセッション・キーとチケットの暗号化タイプ、またはキー表ファイル内の各キーの暗号化タイプを表示します。
-l	キャッシュ・コレクションが使用可能な場合、コレクション内に存在するキャッシュを要約した表を表示します。

オプション	説明
-A	キャッシュ・コレクションが使用可能な場合、コレクション内のすべてのキャッシュの内容を表示します。
-s	出力を生成せずにユーティリティを実行します。キャッシュが読取り不可または期限切れの場合、ユーティリティはステータス 1 で終了します。それ以外の場合はステータス 0 です
-a	資格証明内のアドレスのリストを表示します
-n	逆引きのアドレスのかわりに数字のアドレスを表示します
-C	klist によって検出された場合、資格証明のキャッシュに格納されている構成データをリストします。デフォルトでは、構成データはリストされません。
-t	キー表ファイル内の各キー表入力の時間入力のタイムスタンプを表示します
-K	キー表ファイル内の各キー表入力の暗号化キーの値を表示します
-V	Kerberos バージョンを表示して終了します。

フラグ表示オプション(-f)によって、次のような追加情報が表示されます。

```
% oklist -f
04-Aug-2015 21:57:51   28-Aug-2015 05:58:14
krbtgt/EXAMPLE.COM@EXAMPLE.COM
Flags: FI
```

親トピック: Kerberos認証アダプタのユーティリティ

22.2.3 キャッシュ・ファイルから資格証明を削除するためのokdstryユーティリティのオプション

okdstry (okdestroy)ユーティリティはキャッシュ・ファイルから資格証明を削除します。

次の表に、okdstryの使用可能なオプションを示します。表に示されている機能を使用するには、sqlnet.oraのSQLNET.KERBEROS5_CONF_MITパラメータをTRUEに設定する必要があります。(SQLNET.KERBEROS5_CONF_MITは非推奨ですが、okdstryの下位互換性のため現在も維持されていることに注意してください。)

表22-4 okdstryユーティリティのオプション

オプション	説明
–A	キャッシュ・コレクションが使用可能な場合、コレクション内のキャッシュをすべて破棄します

オプション	説明
–q	音なしで実行します。通常、okdstry はユーザーのチケットの破棄に失敗すると、ビープ音で通知します。このフラグによりこの動作が抑制されます。
–c cache_name	資格証明(チケット)のキャッシュの名前および場所として cache_name を使用します。UNIX では、デフォルトは /tmp/krb5cc_uid です。代替資格証明キャッシュは、sqlnet.ora ファイルで SQLNET.KERBEROS5_CC_NAME パラメータを使用して指定することもできます。

親トピック: Kerberos認証アダプタのユーティリティ

22.2.4 キー表の作成を自動化するためのokcreateユーティリティのオプション

okcreateユーティリティは、KDCまたはサービス・エンドポイントから、キー表の作成を自動化します。

次の表に、okcreateの使用可能なオプションを示します。

表22-5 キー表の作成を自動化するためのokcreateユーティリティのオプション

オプション	説明
-name service_name	キー表を取得する対象となる、Kerberos を使用するサービスのサービス名を指定します。デフォルトは oracle です。
–hosts path-to_hosts_list	キー表を取得する対象となるホストをカンマ区切りのリストで指定するか、ホストのリストが含まれるテキスト・ファイルへのパスを指定します。デフォルトは none です。
–out path_to_output	結果のキー表を格納するための出力パスを指定します。デフォルトはカレント・ディレクトリです。 このディレクトリは、ルート・ユーザーのみがアクセスできるようにします。キー表ファイルは、ネットワークを介してクリアテキストで送信しないでください。
–k	KDC に対して操作を実行する場合に使用します。–s を使用している場合は、このオプションを使用しないでください。
–s	Kerberos を使用するサービスに対して操作を実行する場合に使用します。–k を使用している場合は、このオプションを使用しないでください。

オプション	説明
`-u KDC_username`	KDC のユーザー名を指定します。この設定は、Kerberos を使用するサービス・エンドポイントのみで使用します。 –s を指定し、この設定を省略すると、okcreate のプロンプトは、`KDCuser@KDCmachine` になります。
`-r`	Kerberos レルムを指定します
–p	Kerberos プリンシパルを指定します
`-q`	Kerberos 問合せを指定します
–d	KDC データベース名を指定します
–e	作成された新しいキーに使用する salt リストを指定します
–m	KDC マスター・パスワードを要求するよう指定します

親トピック: Kerberos認証アダプタのユーティリティ

22.3 Kerberosによって認証されたOracleデータベース・サーバーへの接続

Kerberosの構成後は、ユーザー名やパスワードを使用しないでOracleデータベース・サーバーに接続できます。

- 次の構文を使用して、ユーザー名やパスワードを入力しないでデータベースに接続します。

```
$ sqlplus /@net_service_name
```

ここで、`net_service_name`は、Oracle Net Servicesのサービス名です。たとえば:

```
$ sqlplus /@oracle_dbname
```

関連項目:

外部認証の詳細は、『Oracle Database Heterogeneous Connectivityユーザーズ・ガイド』を参照してください。

親トピック: Kerberos認証の構成

22.4 Windows 2008ドメイン・コントローラKDCとの相互運用性の構成

Oracle DatabaseをMicrosoft Windows 2008ドメイン・コントローラのキー配布センター(KDC)と相互作用するように構成できます。

- Microsoft Windows Serverドメイン・コントローラKDCとの相互運用性の構成について
  Oracle DatabaseはMIT Kerberosに準拠します。

- ステップ1: Windows 2008ドメイン・コントローラのためのOracle Kerberosクライアントの構成
  Microsoft Windows 2008ドメイン・コントローラKDCと相互運用するようにOracle Kerberosクライアントを構成できます。
- ステップ2: OracleクライアントのためのMicrosoft Windows Serverドメイン・コントローラKDCの構成
  次に、Oracleクライアントと相互運用するようにMicrosoft Windows Serverドメイン・コントローラKDCを構成します。
- ステップ3: Microsoft Windows Serverドメイン・コントローラKDCのためのOracleデータベースの構成
  Oracleデータベースがインストールされているホスト・コンピュータでドメイン・コントローラに対してOracleデータベースを構成する必要があります。
- ステップ4: Kerberos/Oracleユーザーの初期チケットの取得
  クライアントがデータベースに接続するには、初期チケットを要求する必要があります。

親トピック: Kerberos認証の構成

22.4.1 Microsoft Windows Serverドメイン・コントローラKDCとの相互運用性の構成について

Oracle Databaseは、MIT Kerberosに準拠しています。

そのためOracle Databaseは、Microsoft Windows Serverドメイン・コントローラ上のKerberosキー発行センター(KDC)から発行されたチケットとの相互運用が可能です。このプロセスにより、OracleデータベースでのKerberos認証が可能になります。

親トピック: Windows 2008ドメイン・コントローラKDCとの相互運用性の構成

22.4.2 ステップ1: Windows 2008ドメイン・コントローラのためのOracle Kerberosクライアントの構成

Windows 2008ドメイン・コントローラKDCと相互運用するようにOracle Kerberosクライアントを構成できます。

- ステップ1A: クライアントKerberos構成ファイルの作成
  Windows 2008ドメイン・コントローラをKerberos KDCとして参照する一連のクライアントKerberos構成ファイルを構成する必要があります。
- ステップ1B: sqlnet.oraファイルでのOracle構成パラメータの指定
  Microsoft Windows Serverドメイン・コントローラKerberos Key Distribution Center (KDC)と相互運用するようにOracleクライアントを構成するには、クライアントとデータベース・サーバーでKerberosを構成する場合と同じ`sqlnet.ora`ファイルのパラメータを使用します。
- ステップ1C: tnsnames.oraを使用した追加のKerberosプリンシパルの指定(オプション)
  Oracle Databaseクライアントから接続するように追加のKerberosプリンシパル・ユーザーを構成できます。
- ステップ1D: リスニング・ポート番号の指定
  Microsoft Windows Serverドメイン・コントローラKDCはUDP/TCPポート88でリスニングします。

親トピック: Windows 2008ドメイン・コントローラKDCとの相互運用性の構成

22.4.2.1 ステップ1A: クライアントKerberos構成ファイルの作成

Windows 2008ドメイン・コントローラをKerberos KDCとして参照するKerberosクライアント構成ファイルを構成する必要があります。

- krb.confおよびkrb5.realmsファイルを作成します。Oracle Databaseに用意されているデフォルトのkrb5.confファイルは、自分のサイトにあわせて変更する必要があります。
  krb5.confファイルは、SQLNET.KERBEROS_CONFパラメータによって示された場所にあります。
  たとえば、Windows 2008ドメイン・コントローラがsales3854.us.example.comという名前のノード上で実行されていることを前提とします。
  - krb.confファイル

    たとえば:

    ```
    SALES3854.US.EXAMPLE.COM
    SALES3854.US.EXAMPLE.COM
    sales3854.us.example.com admin server
    ```

  - krb5.confファイル

    たとえば:

    ```
    [libdefaults]
    default_realm=SALES.US.EXAMPLE.COM
    [realms]
    SALES.US.EXAMPLE.COM= { kdc=sales3854.us.example.com:88 }
    [domain_realm]
    .us.example.com=SALES.US.EXAMPLE.COM
    ```

  - krb5.realmsファイル

    たとえば:

    ```
    us.example.com SALES.US.EXAMPLE.COM
    ```

親トピック: ステップ1: Windows 2008ドメイン・コントローラのためのOracle Kerberosクライアントの構成

22.4.2.2 ステップ1B: sqlnet.oraファイルでのOracle構成パラメータの指定

Microsoft Windows Serverドメイン・コントローラKey Distribution Center(KDC)と相互運用するようにOracleクライアントを構成するには、クライアントおよびデータベース・サーバーでKerberosの構成用に使用されるものと同じsqlnet.oraファイルのパラメータを使用します。

- クライアントのsqlnet.oraファイルで次のパラメータを設定します。

  ```
  SQLNET.KERBEROS5_CONF=pathname_to_Kerberos_configuration_file
  SQLNET.KERBEROS5_CONF_MIT=TRUE
  SQLNET.AUTHENTICATION_KERBEROS5_SERVICE=Kerberos_service_name
  SQLNET.AUTHENTICATION_SERVICES=(BEQ,KERBEROS5)
  ```

  次のことに注意してください。

  - SQLNET.KERBEROS5_CONF_MITパラメータは非推奨となりましたが、okint、oklistおよびokdstryユーティリティの下位互換性のため維持されています。
  - Windows Serverオペレーティング・システムはMIT Kerberosバージョン5に基づくセキュリティ・サービスとのみ相互運用するように設計されているため、SQLNET.KERBEROS5_CONF_MITパラメータがTRUEに設定されていることを確認してください。
  - 複数のKerberosプリンシパル・ユーザーを使用する場合は、それらを接続文字列の一部またはtnsnames.oraに指定できます。

関連トピック

- ステップ6A: クライアントとデータベース・サーバーでのKerberosの構成
- ステップ1C: tnsnames.oraを使用した追加のKerberosプリンシパルの指定(オプション)

親トピック: ステップ1: Windows 2008ドメイン・コントローラのためのOracle Kerberosクライアントの構成

22.4.2.3 ステップ1C: tnsnames.oraを使用した追加のKerberosプリンシパルの指定(オプション)

Oracle Databaseクライアントから接続するように追加のKerberosプリンシパル・ユーザーを構成できます。

- KERBEROS5_CC_NAMEおよびKERBEROS5_PRINCIPAL設定をtnsnames.ora接続文字列に追加します。KERBEROS5_CC_NAMEは、追加のKerberosユーザーおよびプリンシパルすべてに必須ですが、KERBEROS5_PRINCIPAL設定はオプションです。Oracle Databaseでは、資格証明キャッシュから取得された値に対してKERBEROS5_PRINCIPALがチェックされます。2つの値が一致しない場合、ユーザーは認証されません。

  たとえば:

```
krbuser1 =
(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=hostname)(PORT=port_number))
(CONNECT_DATA=(SERVICE_NAME=db.example.com))
(SECURITY=(KERBEROS5_CC_NAME = /tmp/krbuser1/krb.cc)
          (KERBEROS5_PRINCIPAL = krbprinc1@example.com)))
krbuser2 =
(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=hostname)(PORT=port_number))
(CONNECT_DATA=(SERVICE_NAME=db.example.com))
(SECURITY=(KERBEROS5_CC_NAME = /tmp/krbuser2/krb.cc)
          (KERBEROS5_PRINCIPAL = krbprinc2@example.com)))
```

関連トピック

- 『Oracle Database Net Servicesリファレンス・ガイド』

親トピック: ステップ1: Windows 2008ドメイン・コントローラのためのOracle Kerberosクライアントの構成

22.4.2.4 ステップ1D: リスニング・ポート番号の指定

Microsoft Windows Serverドメイン・コントローラKDCはUDP/TCPポート88でリスニングします。

- kerberos5のシステム・ファイル・エントリが、UDP/TCPポート88に設定されていることを確認してください。

  UNIX環境では、/etc/servicesファイルの最初のkerberos5エントリが88に設定されていることを確認してください。

親トピック: ステップ1: Windows 2008ドメイン・コントローラのためのOracle Kerberosクライアントの構成

22.4.3 ステップ2: OracleクライアントのためのMicrosoft Windows Serverドメイン・コントローラKDCの構成

次に、Oracleクライアントと相互運用するようにWindows Serverドメイン・コントローラKDCを構成します。

- ステップ2A: ユーザー・アカウントの作成
  Microsoft Windows Serverドメイン・コントローラのKDCにユーザー・アカウントを作成する必要があります。
- ステップ2B: Oracleデータベースのプリンシパル・ユーザー・アカウントおよびキー表の作成
  ユーザー・アカウントの作成後、Oracle Databaseプリンシパル・ユーザー・アカウントを作成します。

関連項目:

Active Directoryでユーザーを作成する方法の詳細は、Microsoft社のドキュメントを参照してください。

親トピック: Windows 2008ドメイン・コントローラKDCとの相互運用性の構成

22.4.3.1 ステップ2A: ユーザー・アカウントの作成

Microsoft Windows Serverドメイン・コントローラのKDCにユーザー・アカウントを作成する必要があります。

- Microsoft Windows Serverドメイン・コントローラで、Microsoft Active DirectoryにOracleクライアントの新規ユーザー・アカウントを作成します。

親トピック: ステップ2: OracleクライアントのためのMicrosoft Windows Serverドメイン・コントローラKDCの構成

22.4.3.2 ステップ2B: Oracle Databaseのプリンシパル・ユーザー・アカウントおよびキー表の作成

ユーザー・アカウントの作成後、Oracle Databaseプリンシパル・ユーザー・アカウントを作成します。

Windows Serverドメイン・コントローラでこのアカウントを作成した後、`okcreate`ユーティリティを使用して、これをプリンシパル・キー表に登録する必要があります。このユーティリティを同じKDCに対して実行して、すべてのサービス・キー表を作成できます(それらを個々に作成せずに済みます)。または、KDCに接続するサービス・エンドポイントから`okcreate`を実行し、必要なコマンドを実行してから、結果のキー表をサービス・エンドポイントにコピーすることもできます。

1. Microsoft Active DirectoryでOracleデータベースの新規ユーザー・アカウントを作成します。

   たとえば、Oracleデータベースがホスト`sales3854.us.example.com`で実行されている場合は、Active Directoryを使用してユーザー名が`sales3854.us.example.com`のユーザーを作成します。

   Active Directoryでは、ユーザーを`host/hostname.dns.com` (`oracle/sales3854.us.example.com`など)として作成しないでください。Microsoft社のKDCは、MIT KDCのようにマルチパートの名前をサポートしていません。MIT KDCは、すべてのプリンシパルをユーザー名として扱うため、マルチパートの名前をサービス・プリンシパルに使用できます。ただし、Microsoft社のKDCでは使用できません。

2. `okcreate`コマンドを実行して、このユーザー・アカウントを使用するキー表を作成します。構文は次のとおりです。

```
okcreate (-s [-u KDCuser@KDCmachine] | -k)
  [-name service_name] [-hosts path_to_host_list]
  [-out path_to_output] [-r realm] [-p principal]
  [-q query] [-d dbname] [-e enc:salt...] [-m]
  [-x db_args]
```

   たとえば:

```
okcreate -s -u kdcuser1@kdcmachine1 -name oracle
  -hosts sales3854.us.example.com
  -out /OSsecured/keytablocation
```

3. 抽出した`keytab`ファイルをOracleデータベースがインストールされているホスト・コンピュータにコピーします。

   たとえば、前のステップで作成した`keytab`を`/krb5/v5srvtab`にコピーできます。

親トピック: ステップ2: OracleクライアントのためのMicrosoft Windows Serverドメイン・コントローラKDCの構成

22.4.4 ステップ3: Microsoft Windows Serverドメイン・コントローラKDCのためのOracleデータベースの構成

Oracleデータベースがインストールされているホスト・コンピュータでドメイン・コントローラに対してOracleデータベースを構成する必要があります。

- ステップ3A: sqlnet.oraファイルでの構成パラメータの設定
  最初に、データベースの構成パラメータを設定する必要があります。
- ステップ3B: 外部認証されたOracleユーザーの作成
  構成パラメータの設定後、外部認証されるOracleユーザーを作成します。

親トピック: Windows 2008ドメイン・コントローラKDCとの相互運用性の構成

22.4.4.1 ステップ3A: sqlnet.oraファイルでの構成パラメータの設定

最初に、データベースの構成パラメータを設定する必要があります。

- データベース・サーバーのsqlnet.oraファイルで、次のパラメータの値を指定します。

```
SQLNET.KERBEROS5_CONF=pathname_to_Kerberos_configuration_file
SQLNET.KERBEROS5_KEYTAB=pathname_to_Kerberos_principal/key_table
SQLNET.KERBEROS5_CONF_MIT=TRUE
SQLNET.AUTHENTICATION_KERBEROS5_SERVICE=Kerberos_service_name
SQLNET.AUTHENTICATION_SERVICES=(BEQ,KERBEROS5)
```

ノート:

- SQLNET.KERBEROS5_CONF_MIT パラメータは非推奨となりましたが、okint、oklist および okdstry ユーティリティの下位互換性のため維持されています。
- Windows Server オペレーティング・システムは MIT Kerberos バージョン 5 に基づくセキュリティ・サービスとのみ相互運用するように設計されているため、SQLNET.KERBEROS5_CONF_MIT パラメータが TRUE に設定されていることを確認してください。
- sqlnet.ora ファイル内の設定はすべての PDB に適用されるということに注意してください。ただし、Kerberos を使用する場合は、すべての PDB を 1 つの KDC で認証する必要があるという意味ではありません。sqlnet.ora ファイルおよび Kerberos 構成ファイルの設定では複数の KDC をサポートできます。

親トピック: ステップ3: Microsoft Windows Serverドメイン・コントローラKDCのためのOracleデータベースの構成

22.4.4.2 ステップ3B: 外部認証されたOracleユーザーの作成

構成パラメータの設定後、外部認証されるOracleユーザーを作成します。

- ステップ8: 外部認証されたOracleユーザーの作成の手順に従って、外部認証されたOracleユーザーを作成します。

  ユーザー名はすべて大文字で作成します(たとえば、ORAKRB@SALES.US.EXAMPLE.COM)。

関連項目:

Oracle Net Managerを使用してsqlnet.oraファイルのパラメータを設定する方法の詳細は、ステップ6: Kerberos認証の構成を参照してください。

親トピック: ステップ3: Microsoft Windows Serverドメイン・コントローラKDCのためのOracleデータベースの構成

22.4.5 ステップ4: Kerberos/Oracleユーザーの初期チケットの取得

クライアントがデータベースに接続するには、初期チケットを要求する必要があります。

1. 初期チケットを要求するには、ステップ9: Kerberos/Oracleユーザーの初期チケットの取得のタスク情報に従います。ユーザーは、Windows固有のキャッシュを使用する場合、`okinit`コマンドを使用して初期チケットを明示的に要求する必要はありません。

   Oracleクライアントが Microsoft Windows Server以上で実行されている場合、Kerberosチケットはユーザーが Windowsにログインしたときに自動的に取得されます。

   システムのKerberosチケット情報を表示するために使用できる`Kerbtray.exe`ユーティリティの詳細は、Microsoft社のドキュメントを参照してください。

2. `tnsnames.ora`に追加したKerberosプリンシパル・ユーザーごとに、クライアントで`okinit`コマンドを実行します。たとえば:

   ```
   okinit krbprinc1@example.com
   ```

親トピック: Windows 2008ドメイン・コントローラKDCとの相互運用性の構成

22.5 Kerberos認証フォールバック動作の構成

Kerberos認証の障害に備え、フォールバック動作(パスワードベースの認証)を構成できます。

OracleクライアントでKerberos認証を構成してOracleデータベースを認証した後に、パスワード・ベースの認証にフォールバックする必要が発生する場合があります。たとえば、ユーザー・データベースのリンクをOracleデータベースに固定している場合などです。

- Kerberos認証からパスワード・ベースの認証へのフォールバックを有効にするには、クライアントとサーバーの両方で`sqlnet.ora`内の`SQLNET.FALLBACK_AUTHENTICATION`パラメータを`TRUE`に設定します。
  このパラメータのデフォルト値は`FALSE`です。これは、デフォルトでは、Kerberos認証に失敗すると接続が失敗することを意味します。

関連項目:

`SQLNET.FALLBACK_AUTHENTICATION`パラメータの詳細は、『Oracle Database Net Servicesリファレンス』を参照してください。

親トピック: Kerberos認証の構成

22.6 Oracle Kerberos認証の構成のトラブルシューティング

一般的なKerberos構成の問題に対するガイドラインを示します。

一般的な問題は次のとおりです。

- `okinit`を使用してチケット認証チケットを取得できない場合:
  - `krb.conf`ファイルを調べて、デフォルトのレルムが正しいことを確認します。
  - レルムに対して指定されているホスト上でKDCが実行されていることを確認します。

        - KDCにユーザー・プリンシパルのエントリがあること、およびパスワードが一致していることを確認します。
        - krb.confファイルおよびkrb.realmsファイルがOracleによって読取り可能であることを確認します。
        - TNS_ADMIN環境変数が、sqlnet.ora構成ファイルを含むディレクトリを指していることを確認します。
    - 初期チケットはあるが接続できない場合:
        - 接続を試みた後で、サービス・チケットを確認します。
        - データベース・サーバー側のsqlnet.oraファイルに、Kerberosによって認識されるサービスに対応するサービス名があることを確認します。
        - 関連するすべてのシステムでクロックのずれが数分以内に設定されていることを確認するか、sqlnet.oraファイルでSQLNET.KERBEROS5_CLOCKSKEWパラメータを変更します。
    - サービス・チケットはあるが接続できない場合:
        - クライアントおよびデータベース・サーバー上でクロックを確認します。
        - v5srvtabファイルが正しい場所にあり、Oracleによって読取り可能であることを確認します。sqlnet.oraパラメータを忘れずに設定してください。
        - データベース・サーバー側のsqlnet.oraファイルで指定されているサービスに対してv5srvtabファイルが生成されていることを確認します。
    - 問題がないと考えられるにもかかわらず、発行した問合せが失敗する場合:
        - 初期チケットが転送可能であることを確認します。okinitユーティリティを実行して初期チケットを取得している必要があります。
        - 資格証明の有効期限を確認します。資格証明の有効期限が切れている場合は、接続をクローズし、okinitを実行して新しい初期チケットを取得します。

- 一般的なKerberos構成の問題
  Oracleは、一般的なKerberos構成の問題についてのガイダンスを提供しています。
- Kerberos構成のORA-12631エラー
  ORA-12631: ユーザー名の検索に失敗しました。エラーは、Kerberos認証に不正なプリンシパルまたは間違った形式のプリンシパルが使用されていることが原因で発生する可能性があります
- Kerberos構成のORA-28575エラー
  ORA-28575: 外部プロシージャ・エージェントへのRPC接続をオープンできません。エラーは、クライアントがリモートのときにEXTPROCプロセスが生成されると発生することがあります。
- Kerberos構成のORA-01017エラー
  ORA-01017: ユーザー名/パスワードが無効です。ログオンは拒否されました。エラーは、okinitが失敗し、SQL*Plus接続に有効なチケットがない場合に発生する可能性があります。
- Kerberos okinit操作のトレースの有効化
  KRB5_TRACE環境変数を使用すると、Kerberos okinit操作をトレースできます。

親トピック: Kerberos認証の構成

22.6.1 一般的なKerberos構成の問題

一般的なKerberos構成の問題に対するガイドラインを示します。

一般的な問題は次のとおりです。

- okinitを使用してチケット認証チケットを取得できない場合:
  - krb.confファイルを調べて、デフォルトのレルムが正しいことを確認します。
  - レルムに対して指定されているホスト上でKDCが実行されていることを確認します。
  - KDCにユーザー・プリンシパルのエントリがあること、およびパスワードが一致していることを確認します。
  - krb.confファイルおよびkrb.realmsファイルがOracleによって読取り可能であることを確認します。
  - TNS_ADMIN環境変数が、sqlnet.ora構成ファイルを含むディレクトリを指していることを確認します。
- 初期チケットはあるが接続できない場合は、次を試してください。
  - 接続を試みた後で、サービス・チケットを確認します。
  - データベース・サーバー側のsqlnet.oraファイルに、Kerberosによって認識されるサービスに対応するサービス名があることを確認します。
  - 関連するすべてのシステムでクロックのずれが数分以内に設定されていることを確認するか、sqlnet.oraファイルでSQLNET.KERBEROS5_CLOCKSKEWパラメータを変更します。
- サービス・チケットはあるが接続できない場合:
  - クライアントおよびデータベース・サーバー上でクロックを確認します。
  - v5srvtabファイルが正しい場所にあり、Oracleによって読取り可能であることを確認します。sqlnet.oraパラメータを忘れずに設定してください。
  - データベース・サーバー側のsqlnet.oraファイルで指定されているサービスに対してv5srvtabファイルが生成されていることを確認します。
- まったく問題がないように見えていても、別の問合せを発行したときに失敗する場合は、次の事項を試してください。
  - 初期チケットが転送可能であることを確認します。okinitユーティリティを実行して初期チケットを取得している必要があります。
  - 資格証明の有効期限を確認します。資格証明の有効期限が切れている場合は、接続をクローズし、okinitを実行して新しい初期チケットを取得します。

親トピック: Oracle Kerberos認証の構成のトラブルシューティング

22.6.2 Kerberos構成のORA-12631エラー

ORA-12631: ユーザー名の検索に失敗しました。エラーは、Kerberos認証に不正なプリンシパルまたは間違った形式のプリンシパルが使用されていることが原因で発生する可能性があります

出力で、Wrong principal in requestのsqlnetサーバー・トレース・ファイルを確認します。

この問題を修正するには、krb5.confファイルを編集して、[domain_realm]の設定を確認します。これらの設定では大文字と小文字が区別されるため、domain_realm名が正しいとしても、小文字の場合は正しく解析されません。この設定が大文字になっていることを確認します。たとえば:

```
[domain_realm]
.country.<DOMAIN_NAME> = SECWIN.LOCAL
country.<DOMAIN_NAME> = SECWIN.LOCAL
```

親トピック: Oracle Kerberos認証の構成のトラブルシューティング

22.6.3 Kerberos構成のORA-28575エラー

ORA-28575: 外部プロシージャ・エージェントへのRPC接続をオープンできません。エラーは、クライアントがリモートのときにEXTPROCプロセスが生成されると発生することがあります。

外部プロシージャ・コールにはKerberos認証は不要です。この問題を修正するには、sqlnet.oraファイルのKERBEROS5およびKERBEROS5PREパラメータの前にBEQを追加します。

親トピック: Oracle Kerberos認証の構成のトラブルシューティング

22.6.4 Kerberos構成のORA-01017エラー

ORA-01017: ユーザー名/パスワードが無効です。ログオンは拒否されました。エラーは、okinitが失敗し、SQL*Plus接続に有効なチケットがない場合に発生する可能性があります。

okinitトレース・ファイルに、次のエラーが示されます。

```
nauk5l_sendto_kdc: entry
snauk5l_sendto_kdc: exit
snauk5l_sendto_kdc: exit
nauk5la_get_in_tkt: Returning 25: Additional pre-authentication required
.
snauk5l_sendto_kdc: exit
snauk5l_sendto_kdc: exit
nauk5la_get_in_tkt: Returning 24: Preauthentication failed
.
nauk5la_get_in_tkt: exit
nauk5zi_kinit: Getting TGT failed: Preauthentication failed
.
nauk5fq_free_principal: entry
nauk5fq_free_principal: exit
nauk5fq_free_principal: entry
nauk5fq_free_principal: exit
nauk5zi_kinit: Returning 24: Preauthentication failed
.
nauk5zi_kinit: exit
```

この問題を解決するには:

1. krb5.confファイルで、default_tkt_enctypesパラメータを設定します。これにより、クライアントからリクエストされる暗号化のタイプを制御できます。たとえば:

   ```
   default_tgs_enctypes = aes256-cts-hmac-sha1-96
   default_tkt_enctypes = aes256-cts-hmac-sha1-96
   ```

2. 次のオプションを指定して、okinitをテストします。

   ```
   okinit user_name
   ```

   DES暗号化アルゴリズムがActive Directoryサーバーに実装されていないと、okinitは失敗します。

   ```
   okinit user_name
   Kerberos Utilities for Solaris: Version 23.0.0.0.0 - Production on 15-MAY-2023
   11:50:39
   Copyright (c) 1996, 2023 Oracle. All rights reserved.
   Password for user_name@domain:
   okinit: KDC has no support for encryption type
   okinit user_name
   ```

```
Kerberos Utilities for Solaris: Version 23.0.0.0.0 - Production on 15-MAY-2023
11:50:39
Copyright (c) 1996, 2023 Oracle. All rights reserved.
Password for user_name@domain:
okinit: Preauthentication failed
```

ただし、次が成功します。

```
okinit user_name
Kerberos Utilities for Solaris: Version 23.0.0.0.0 - Production on 15-MAY-2023
11:50:39
Copyright (c) 1996, 2023 Oracle. All rights reserved.
Password for user_name@domain:
```

oklistユーティリティは、チケットからユーザー・プリンシパルをリスト表示します。有効なチケットが存在する場合は、通常の方法で接続できます。okinitが正常に完了したら、次に示すように、ユーザー名やパスワードを使用せずにOracle Databaseサーバーに接続できます。

```
% sqlplus /@service_name
```

親トピック: Oracle Kerberos認証の構成のトラブルシューティング

22.6.5 Kerberos okinit操作に対するトレースの有効化

KRB5_TRACE環境変数を使用すると、Kerberos okinit操作をトレースできます。

この方法は、krb.confのdefault_tkt_enctypes設定を使用して設定された暗号化タイプを検証するために使用できます。

1. KRB5_TRACE環境変数に対してexportコマンドを実行します。
   たとえば、krb5.trcという名前のトレース・ファイルの場合:

   ```
   export KRB5_TRACE="/oracle/work/krb5.trc"
   ```

2. 次のように、okinitコマンドを実行します。

   ```
   okinit user_name
   ```

   次のような出力が表示されます。

   ```
   Kerberos Utilities for Linux: Version 23.0.0.0.0 - Development on 15-MAY-2023
   21:37:39
   Copyright (c) 1996, 2023 Oracle. All rights reserved.
   Configuration file : /oracle/work/krb/krb.conf.
   Password for user_name@US.EXAMPLE.COM:
   pfitch@sales_us:/oracle/work/
   ```

3. grepコマンドを使用して、トレース・ファイルのdefault_tkt_enctype設定を検索します。
   たとえば:

   ```
   /oracle/work/fgrep aes256-cts krb5.trc
   [4072148] 1683321391.149999: Selected etype info: etype aes256-cts, salt
   "US.EXAMPLE.COMoratst", params ""
   [4072148] 1683321393.375503: AS key obtained from gak_fct: aes256-cts/95C0
   [4072148] 1683321393.375504: Decrypted AS reply; session key is: aes256-
   cts/40F6
   [4072182] 1683321415.915360: Selected etype info: etype aes256-cts, salt
   "US.EXAMPLE.COMoratst", params ""
   [4072182] 1683321417.701784: AS key obtained from gak_fct: aes256-cts/95C0
   [4072182] 1683321417.701785: Decrypted AS reply; session key is: aes256-
   cts/859E
   ```

```
[4075441] 1683322653.162464: Selected etype info: etype aes256-cts, salt
"US.EXAMPLE.COMoratst", params ""
[4075441] 1683322656.084028: AS key obtained from gak_fct: aes256-cts/1938
[4075455] 1683322659.360899: Selected etype info: etype aes256-cts, salt
"US.EXAMPLE.COMoratst", params ""
[4075455] 1683322661.242404: AS key obtained from gak_fct: aes256-cts/95C0
[4075455] 1683322661.242405: Decrypted AS reply; session key is: aes256-
cts/3580
```

親トピック: Oracle Kerberos認証の構成のトラブルシューティング

23 Transport Layer Security認証の構成

Transport Layer Security認証を使用するようにOracle Databaseを構成できます。

- Transport Layer SecurityおよびSecure Sockets Layer
  以前はSecure Sockets Layer (SSL)と呼ばれていたTransport Layer Security (TLS)は、安全なネットワーク接続を目的に、Netscape社によって設計されました。
- Oracle DatabaseでのTransport Layer Securityを使用した認証
  Transport Layer Securityは、暗号化およびデータ・アクセス・コントロールなど、Oracle Databaseの中核機能と連携します。
- Oracle環境におけるTransport Layer Securityの機能: TLSハンドシェイク
  Transport Layer Securityによるネットワーク接続を開始する場合、クライアントとサーバーは、認証を行う前にTLSハンドシェイクを実行します。
- Oracle環境における公開キー・インフラストラクチャ
  公開キー・インフラストラクチャ(PKI)は、組織全体のセキュリティ基盤を提供するネットワーク・コンポーネントの基質であり、信頼アサーションに基づいています。
- Transport Layer Securityと他の認証方式の併用
  TLSをデータベースのユーザー名とパスワード、RADIUSおよびKerberosと同時に使用するようにOracle Databaseを構成できます。
- Transport Layer Securityとファイアウォール
  Oracle Databaseは、アプリケーション・プロキシベースのファイアウォールとステートフル・パケット・インスペクション・ファイアウォールの2つをサポートしています。
- Transport Layer Security使用時の問題
  他のOracle製品との通信や、サポート対象の認証方式および暗号化方式のタイプといった、TLSの使用に関する問題に注意する必要があります。
- クライアント・ウォレットを使用しないTransport Layer Security接続
  データベース・サーバーに対して共通のルート証明書を使用するTransport Layer Security (TLS)接続には、クライアント・ウォレットは必要ありません。
- クライアント・ウォレットを使用するTransport Layer Security接続
  Transport Layer Securityは、サーバーで構成してから、クライアントで構成する必要があります。
- Oracle Real Application Clusters環境でのTransport Layer Security接続
  Oracle RACツールを使用してOracle Real Application Clusters (Oracle RAC)環境でTransport Layer Security (TLS)接続を構成し、Oracle Database構成ファイルを変更できます。
- Microsoft証明書ストアを使用したクライアント認証および暗号化のためのトランスポート・レイヤー・セキュリティの構成
  Microsoft証明書ストア(MCS)でこの構成を実行するには、`orapki`コマンドライン・ツールを使用して証明書を生成し、Oracleウォレットを操作します。
- Transport Layer Security構成のトラブルシューティング
  Oracle Database SSLアダプタの使用中に一般的なエラーが発生する場合があります。
- 証明書失効リストによる証明書の検証
  オラクル社では、証明書失効リストを使用して証明書を検証できるツールを提供しています。
- ハードウェア・セキュリティ・モジュールを使用するためのシステムの構成
  Oracle Databaseでは、RSA Security社のPKCS #11仕様に準拠したAPIを使用するハードウェア・セキュリ

ティ・モジュールがサポートされています。

親トピック: 厳密認証の管理

23.1 Transport Layer SecurityおよびSecure Sockets Layer

以前はSecure Sockets Layer (SSL)と呼ばれていたTransport Layer Security (TLS)は、安全なネットワーク接続を目的に、Netscape社によって設計されました。

- Transport Layer SecurityとSecure Sockets Layerの違い
  Transport Layer Security (TLS)は、Secure Sockets Layer (SSL)バージョン3.0の増分バージョンです。
- マルチテナント環境でのTransport Layer Securityの使用
  Transport Layer Security (TLS)はアプリケーション・コンテナのマルチテナント環境で使用できます。

親トピック: Transport Layer Security認証の構成

23.1.1 Transport Layer SecurityとSecure Sockets Layerの違い

Transport Layer Security (TLS)は、Secure Sockets Layer (SSL)バージョン3.0の増分バージョンです。

SSLは最初にNetscape社によって開発されましたが、Internet Engineering Task Force (IETF)がその開発を引き継ぎ、名前をTransport Layer Security (TLS)に変更しました。TLSはIETF標準です。

Oracle DatabaseではTLSが実装されているため、*Oracle Database*セキュリティ・ガイドでは、Secure Sockets LayerおよびSSLのかわりに、Transport Layer SecurityおよびTLSという用語が使用されています。ただし、Oracle Databaseライブラリ内の他のドキュメントでは、以前のSecure Socket LayerおよびSSLという用語が引き続き使用されている場合があります。これらのプロトコルの使用方法または構成方法に違いがある場合、*Oracle Database*セキュリティ・ガイドでは、記述内容がSSLに該当するか、TLSに該当するかを明記します。

Oracle Databaseソフトウェアでは、古い用語の一部が引き続き使用されています。たとえば、`netmgr`ツールでは、引き続きSecure Socket LayerおよびSSLの用語が使用されています。`SSL_SERVER_CERT_DN`などの多くのSSLパラメータでは、古い用語が使用されています。暗号スイートの名前とエラー・メッセージ内の言い回しにも、SSL用語が使用されています。ただし、これらのすべての機能は、Transport Layer Securityと連携して、適用されます。

親トピック: Transport Layer SecurityおよびSecure Sockets Layer

23.1.2 マルチテナント環境でのTransport Layer Securityの使用

Transport Layer Security (TLS)はアプリケーション・コンテナのマルチテナント環境で使用できます。

アプリケーション・コンテナのマルチテナント環境でTransport Layer Security (TLS)を使用する場合、各PDBで独自のウォレットと独自のTLS認証用の証明書を使用できることを確認する必要があります。

1. ウォレットを使用するPDBに接続します。
2. ウォレットを`wallet`ディレクトリのサブディレクトリに置きます。サブディレクトリの名前は、ウォレットを使用するPDBのGUIDです。
   各PDBには個別の`sqlnet.ora`ファイルがないため、これを行う必要があります。たとえば、`sqlnet.ora`の`WALLET_LOCATION`パラメータが次のように設定されているとします。

```
(SOURCE=(METHOD=FILE)(METHOD_DATA=
   (DIRECTORY=/home/oracle/wallet)))
```

各PDBのウォレットを/home/oracle/wallet/PDB_GUIDディレクトリに置きます。既存のPDBおよびそのGUIDを確認するには、DBA_PDBSデータ・ディクショナリ・ビューを問い合せます。

WALLET_LOCATIONパラメータが指定されていない場合、デフォルトのウォレット・パスのリーフ・サブディレクトリにPDBウォレットを置く必要があります。サブディレクトリの名前はPDBのGUID、リーフ・サブディレクトリの名前はTLSです。たとえば:

```
$ORACLE_BASE/admin/db_unique_name/PDB_GUID/TLS
```

または、ORACLE_BASE環境変数が設定されていない場合は、Oracleホームを使用できます。

```
$ORACLE_HOME/admin/db_unique_name/PDB_GUID/TLS
```

これらのデフォルトの場所は、LDAPの認証用にウォレットを特定するためにOracle Enterprise User Securityによって使用されるデフォルトに対応します。

PDBで個別の証明書を使用できるようにするには、$WALLET_LOCATION/PDB_GUID/TLSディレクトリの下にサブディレクトリを作成し、このサブディレクトリにウォレットをコピーします。

3. PDBをクローズしてから再度オープンします。

```
ALTER PLUGGABLE DATABASE pdb_name CLOSE IMMEDIATE;
ALTER PLUGGABLE DATABASE pdb_name OPEN;
```

親トピック: Transport Layer SecurityおよびSecure Sockets Layer

23.2 Oracle DatabaseでのTransport Layer Securityを使用した認証

Transport Layer Securityは、暗号化およびデータ・アクセス・コントロールなど、Oracle Databaseの中核機能と連携します。

Oracle DatabaseのTLS機能を使用してクライアントとサーバー間の通信を保護すると、次の操作を実行できます

- TLSを使用したクライアントとサーバー間の接続の暗号化
- TLS通信用に構成されたOracleデータベース・サーバーに対するクライアントまたはサーバー(Oracle Application Server 10gなど)の認証

TLS機能は、単独で使用するか、Oracle Databaseでサポートされている他の認証方式と組み合せて使用できます。たとえば、TLSから提供されている暗号化をKerberosから提供されている認証と組み合せて使用できます。TLSでは、次の認証モードがサポートされます。

- サーバーのみ、クライアントに対して自己認証を行います。
- クライアントとサーバーは、互いに自己認証を行います。
- クライアントもサーバーも、互いに自己認証を行わず、TLS暗号化機能を単独で使用します

関連項目:

TLSの詳細は、Internet Engineering Task Forceが公開しているTLSプロトコルのバージョン3.0を参照してください

親トピック: Transport Layer Security認証の構成

23.3 Oracle環境におけるTransport Layer Securityの機能: TLSハンドシェイク

Transport Layer Securityによるネットワーク接続を開始する場合、クライアントとサーバーは、認証を行う前にTLSハンドシェイクを実行します。

ハンドシェイク・プロセスは次のようになります。

1. クライアントとサーバーは、どの暗号スイートを使用するかを設定します。これには、データ転送に使用する暗号化アルゴリズムも含まれます。
2. サーバーは証明書をクライアントに送信し、クライアントは、サーバーの証明書が信頼できるCAによって署名されていることを検証します。このステップにより、サーバーの身元が検証されます。
3. 同様に、クライアント認証が必要な場合は、クライアントが自分自身の証明書をサーバーに送信し、サーバーは、クライアントの証明書が信頼できるCAによって署名されているかどうかを検証します。
4. クライアントとサーバーが、公開キー暗号化を使用してキー情報を交換します。この情報に基づき、双方でセッション・キーが生成されます。キーは少なくとも二者(通常はクライアントとサーバー)によって共有され、単一の通信セッション中のデータ暗号化に使用されます。セッション・キーは通常、ネットワーク・トラフィックを暗号化するために使用されます。クライアントとサーバーはセッションの開始時にセッション・キーをネゴシエーションすることができ、そのキーはそのセッションの関係者間のすべてのネットワーク・トラフィックを暗号化するために使用されます。クライアントとサーバーが新しいセッションで再び通信する場合は、新しいセッション・キーをネゴシエーションします。クライアントとサーバーとの間の以降の通信はすべて、このセッション・キーと、ネゴシエーションにより決定した暗号スイートを使用して、暗号化または復号化されます。

認証プロセスは次のとおりです。

1. クライアントで、ユーザーがTLSを使用してサーバーへのOracle Net接続を開始します。
2. TLSは、クライアントとサーバー間のハンドシェイクを実行します。
3. ハンドシェイクが成功すると、サーバーは、そのデータベースにアクセスするために必要な認可をユーザーが所有していることを検証します。

親トピック: Transport Layer Security認証の構成

23.4 Oracle環境における公開キー・インフラストラクチャ

公開キー・インフラストラクチャ(PKI)は、組織全体のセキュリティ基盤を提供するネットワーク・コンポーネントの基質であり、信頼アサーションに基づいています。

- 公開キーの暗号化について
  従来の秘密キーまたは対称キーの暗号化では、安全な通信を確立する目的で2者以上が共有する1つの秘密キーが必要です。
- Oracle環境における公開キー・インフラストラクチャ・コンポーネント
  Oracle環境の公開キー・インフラストラクチャ(PKI)のコンポーネントには、認証局、証明書、証明書失効リストおよびウォレットなどがあります。

親トピック: Transport Layer Security認証の構成

23.4.1 公開キーの暗号化について

従来の秘密キーまたは対称キーの暗号化では、安全な通信を確立する目的で2者以上が共有する1つの秘密キーが必要です。

このキーは、ユーザー間で送信される保護メッセージの暗号化および復号化に使用されるため、各ユーザーへは事前に安全な方法でキーを配布する必要があります。この方法における問題点は、キーを安全に転送し、格納することが困難なことです。

公開キーの暗号化による公開キー/秘密キーのペアおよびキー配布のための安全な方法を採用することで、この問題は解決します。対応する秘密キーの所有者のみが復号化できるメッセージは、自由に使用できる公開キーによって暗号化できます。秘密キーは、その他のセキュリティ資格証明とともに、ウォレットと呼ばれる暗号化されたコンテナに、安全に格納されます。

公開キーのアルゴリズムでは、メッセージの秘密は保証されますが、安全な通信は必ずしも保証されません。その理由は、通信者間の識別が検証されないためです。安全な通信を確立するには、メッセージの暗号化に使用される公開キーが相手の受信者に実際に属していることを確認することが重要です。そうしないと、第三者が通信を傍受し、公開キーのリクエストに割り込み、正当なキーを独自の公開キーに置き換えることが可能になります(第三者攻撃)。

この攻撃を回避するには、公開キーの所有者の確認(認証と呼ばれるプロセス)が必要です。認証は、通信する双方の委託するサード・パーティの認証局(CA)によって行われます。

CAは、エンティティの名前、公開キーおよび他のセキュリティ資格証明を含む公開キー証明書を発行します。通常、このような資格証明には、CA名、CAの署名および証明書の有効日(開始日、終了日)が含まれています。

CAでは独自の秘密キーを使用してメッセージを暗号化します。一方、そのメッセージの復号化には公開キーが使用されるため、メッセージがCAによって暗号化されたものであるかどうかが確認されます。CA公開キーは広く一般に知られているため、アクセスするたびに認証する必要はありません。このようなCA公開キーはウォレットに格納されます。

親トピック: Oracle環境における公開キー・インフラストラクチャ

23.4.2 Oracle環境における公開キー・インフラストラクチャ・コンポーネント

Oracle環境の公開キー・インフラストラクチャ(PKI)のコンポーネントには、認証局、証明書、証明書失効リストおよびウォレットなどがあります。

- 認証局
  認証局(CA)は、ユーザー、データベース、管理者、クライアント、サーバーなどのエンティティの識別情報を証明する、信頼できるサード・パーティです。
- 証明書
  証明書は、エンティティの公開キーが、信頼できる認証局(CA)によって署名されたときに作成されます。
- 証明書失効リスト
  公開キー・ペアをユーザーIDにバインドする証明書にCAが署名すると、証明書は指定された期間有効になります。
- ウォレット
  ウォレットは、認証および署名資格証明(TLSで必要な秘密キー、証明書、信頼できる証明書など)の格納に使用されるコンテナです。
- ハードウェア・セキュリティ・モジュール
  SSLのハードウェア・セキュリティ・モジュールには、様々な機能を処理するデバイスや、暗号情報を格納するハードウェア・デバイスなどがあります。

親トピック: Oracle環境における公開キー・インフラストラクチャ

23.4.2.1 認証局

認証局(CA)は、ユーザー、データベース、管理者、クライアント、サーバーなどのエンティティの識別情報を証明する、信頼できるサード・パーティです。

エンティティが証明書を要求すると、CAはその識別情報を検証し、CAの秘密キーを使用して署名した証明書を発行します。

CAごとに、証明書の発行時の識別情報要件が異なる可能性があります。CAの中には、要求者の識別情報をドライバのライセンスを使用して確認したり、要求者の指紋で識別情報を確認したり、証明書リクエスト・フォームが認証されていることを必要とするものがあります。

CAは、公開キーが含まれている独自の証明書を発行します。各ネットワーク・エンティティには、信頼できるCA証明書のリストがあります。通信する前に、ネットワーク・エンティティは証明書を交換し、相互の証明書がそれぞれの信頼できるCA証明書リストのいずれかのCAによって署名されていることを確認します。

ネットワーク・エンティティは、同じCAから、または異なるCAから証明書を取得できます。

関連トピック

- ウォレット

親トピック: Oracle環境における公開キー・インフラストラクチャ・コンポーネント

23.4.2.2 証明書

証明書は、エンティティの公開キーが、信頼できる認証局(CA)によって署名されたときに作成されます。

この証明書は、そのエンティティの識別情報が正しいこと、および公開キーがそのエンティティに実際に属していることを保証します。

証明書には、エンティティの名前、公開キー、有効期限、さらにシリアル番号および証明連鎖情報が含まれています。(証明書チェーンは、エンド・ユーザーまたはサブスクライバの証明書とその認証局の証明書を含む、順序付けられた証明書のリストです。)また、証明書に関連付けられた権限に関する情報が含まれている場合もあります。

ネットワーク・エンティティが証明書を受信すると、それが信頼できる証明書、つまり信頼できる認証局によって発行および署名された証明書であるか検証します。証明書は、期限切れになるか失効するまで有効です。

親トピック: Oracle環境における公開キー・インフラストラクチャ・コンポーネント

23.4.2.3 証明書失効リスト

公開キー・ペアをユーザーIDにバインドする証明書にCAが署名すると、証明書は指定された期間有効になります。

ただし、ユーザー名の変更や秘密キーの漏えいなどの特定のイベントが発生した場合は、有効期間が終了する前に証明書が無効になることがあります。この場合は、CAによって証明書が失効され、そのシリアル番号が証明書失効リスト(CRL)に追加されます。CAは、特定の公開キーが、関連付けられているユーザーIDの確認に使用できなくなると、定期的にCRLを発行してユーザーに警告します。

サーバーまたはクライアントはOracle環境でユーザー証明書を受信すると、その有効期限、署名および失効ステータスをチェックして証明書を検証できます。証明書失効ステータスは、発行されたCRLに照らして検証することでチェックされます。証明書失効ステータス・チェックが有効になっている場合、サーバーはこの機能の構成方法に応じて適切なCRLを検索します。サーバーは、次の場所で(この順番で) CRLを検索します。

1. ローカル・ファイル・システム
2. Oracle Internet Directory
3. CRL配布ポイント(CRL DP)。証明書が発行されたときの、CRL Distribution Point (CRL DP) X.509バージョ

ン3の証明書拡張で指定されている場所。CRL DPは、X.509バージョン3証明書標準で指定されるオプションの拡張子であり、証明書の失効情報が格納される区分CRLの位置を示します。通常、この拡張子の値はURLの形式です。CRL DPによって、1つの認証局ドメイン内の失効情報を複数のCRLにポストできます。CRL DPによって、失効情報はより管理しやすい部分に細分化され、CRLが膨大に増加するのが回避されるため、パフォーマンスが向上します。たとえば、CRL DPを証明書に指定し、その証明書の失効情報をダウンロードできる、Webサーバー上のファイルを指すようにできます。

ノート:

他の Oracle 製品で CRL を使用するには、その製品のドキュメントを参照してください。CRL による証明書の検証の実装は、Oracle Database 12c リリース 1 (12.1)以降の SSL アダプタでのみ使用できます。

関連トピック

- 証明書失効リストによる証明書の検証

親トピック: Oracle環境における公開キー・インフラストラクチャ・コンポーネント

23.4.2.4 ウォレット

ウォレットは、認証および署名資格証明(TLSで必要な秘密キー、証明書、信頼できる証明書など)の格納に使用されるコンテナです。

Oracle環境では、TLS通信を行うどのエンティティにも、X.509バージョン3の証明書、秘密キーおよび信頼できる証明書のリストが必要です。ただし、Diffie-Hellmanは例外です。

セキュリティ管理者は、サーバーのセキュリティ資格証明の管理にOracle Wallet Managerを使用します。ウォレットの所有者は、クライアントのセキュリティ資格証明の管理に使用します。具体的には、Oracle Wallet Managerは次のことを行うために使用します。

- 公開キーと秘密キーのペアの生成および証明書リクエストの作成
- 秘密キーと一致するユーザー証明書の格納
- 信頼できる証明書の構成

関連項目:

- Oracle Wallet Managerの詳細は、『Oracle Databaseエンタープライズ・ユーザー・セキュリティ管理者ガイド』を参照してください。
- Oracleウォレットの新規作成の詳細は、『Oracle Databaseエンタープライズ・ユーザー・セキュリティ管理者ガイド』を参照してください。
- Oracleウォレットでの信頼できる証明書の管理の詳細は、『Oracle Databaseエンタープライズ・ユーザー・セキュリティ管理者ガイド』を参照してください。

親トピック: Oracle環境における公開キー・インフラストラクチャ・コンポーネント

23.4.2.5 ハードウェア・セキュリティ・モジュール

SSLのハードウェア・セキュリティ・モジュールには、様々な機能を処理するデバイスや、暗号情報を格納するハードウェア・デバイ

スなどがあります。

Oracle Databaseでは、これらのデバイスを次の機能に使用します。

- 秘密キーなどの暗号情報の格納
- 他のトランザクションに応答できるようにCPUを解放して、サーバーにおけるRSA操作の負荷を軽減する暗号操作の実行

暗号情報は、次の2つのタイプのハードウェア・デバイスに格納できます。

- (サーバー側)キーがボックスに格納され、トークンを使用して管理されるハードウェア・ボックス。
- (クライアント側)トークンへの秘密キーの格納をサポートするスマートカード・リーダー

Oracle環境では、RSA Security社の公開キー暗号規格(PKCS)#11仕様に準拠しているAPIを使用したハードウェア・デバイスがサポートされています。

ノート:

現在、SafeNET、nCipher および Utimaco デバイスは Oracle Database で認定されています。

関連トピック

- ハードウェア・セキュリティ・モジュールを使用するためのシステムの構成

親トピック: Oracle環境における公開キー・インフラストラクチャ・コンポーネント

23.5 Transport Layer Securityと他の認証方式の併用

TLSをデータベースのユーザー名とパスワード、RADIUSおよびKerberosと同時に使用するようにOracle Databaseを構成できます。

- アーキテクチャ: Oracle DatabaseとTransport Layer Security
  Oracle DatabaseとTLSとの連携のアーキテクチャを理解することが重要です。
- Transport Layer Securityと他の認証方式の併用
  Transport Layer Securityは、Oracle Databaseでサポートされているその他の認証方式との併用が可能です。

親トピック: Transport Layer Security認証の構成

23.5.1 アーキテクチャ: Oracle DatabaseとTransport Layer Security

Oracle DatabaseとTLSとの連携のアーキテクチャを理解することが重要です。

Transport Layer SecurityアーキテクチャのOracle Databaseの実装を示す図20-4では、Oracle DatabaseがTLSの上位のセッション・レイヤーで動作し、トランスポート・レイヤーでTCP/IPを使用していることを示しています。セッション・レイヤーは、プレゼンテーション・レイヤー・エンティティが必要とするサービスを提供するネットワーク・レイヤーであり、プレゼンテーション・レイヤー・エンティティがダイアログの編成と同期およびデータ交換の管理を行えるようにします。このレイヤーは、クライアントとサーバー間でネットワーク・セッションを確立、管理および終了する。トランスポート・レイヤーは、データ・フロー制御とエラー・リカバリ方式を通じてエンドツーエンドの信頼性を維持するネットワーキング・レイヤーです。Oracle Net Servicesは、トランスポート・レイヤーにOracleプロトコル・サポートを使用します。

このように機能が分離されているため、TLSを他のサポートされているプロトコルと同時に利用できます。

関連トピック

- Oracle Database Net Services管理者ガイド

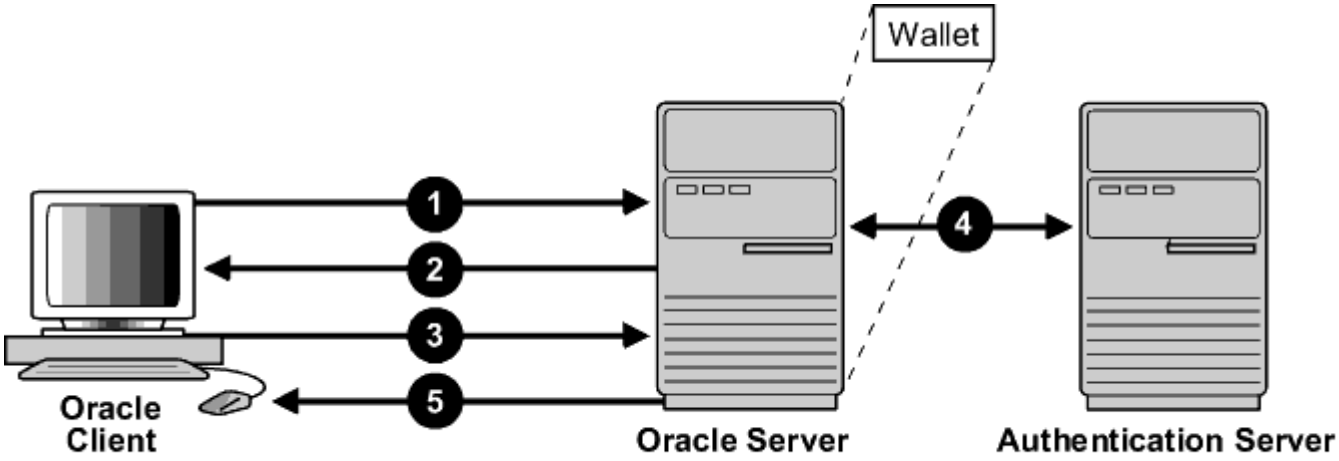
親トピック: Transport Layer Securityと他の認証方式の併用

23.5.2 Transport Layer Securityと他の認証方式の併用

Transport Layer Securityは、Oracle Databaseでサポートされているその他の認証方式との併用が可能です。

図23-1に、Transport Layer Securityが別の認証方式と併用されている構成を示します。

図23-1 Transport Layer Securityと他の認証方式との関係



この例では、最初のハンドシェイク(サーバー認証)を確立するためにTransport Layer Securityが使用され、クライアントの認証に別の認証方式が使用されています。このプロセスは、次のとおりです。

1. クライアントがOracleデータベース・サーバーに接続しようとします。
2. Transport Layer Securityによってハンドシェイクが実行され、その間にサーバーがクライアントに対して自己認証を行い、クライアントとサーバーの両方が使用する暗号スイートを設定します。
3. Transport Layer Securityハンドシェイクが正常に完了すると、ユーザーはデータベースにアクセスしようとします。
4. Oracleデータベース・サーバーは、KerberosやRADIUSなどの非TLS認証方式を使用して、認証サーバーでユーザーを認証します。
5. 認証サーバーによる検証が完了すると、Oracleデータベース・サーバーによってアクセス権と認可がユーザーに付与され、ユーザーはTLSを使用してデータベースに安全にアクセスできるようになります。

関連トピック

- Oracle環境におけるTransport Layer Securityの機能: TLSハンドシェイク

親トピック: Transport Layer Securityと他の認証方式の併用

23.6 Transport Layer Securityとファイアウォール

Oracle Databaseは、アプリケーション・プロキシベースのファイアウォールとステートフル・パケット・インスペクション・ファイアウォールの2つをサポートしています。

ファイアウォールは次のとおりです。

- アプリケーション・プロキシベースのファイアウォール: Network Associates GauntletやAxent Raptorなど。
- ステートフル・パケット・インスペクション・ファイアウォール: Check Point Firewall-1やCisco PIX Firewallなど。

TLSを有効にした場合、ステートフル・インスペクション・ファイアウォールは暗号化されたパケットを復号化しないため、アプリケーション・プロキシ・ファイアウォールと同じように動作します。

ファイアウォールは暗号化されたトラフィックを検査しません。ファイアウォールは、イントラネット・サーバーのTLSポート宛てのデータを検出すると、アクセス・ルールに基づいてターゲットIPアドレスを確認し、許可されたTLSポートに対してはTLSパケットを通過させ、他のすべてのパケットは拒否します。

親トピック: Transport Layer Security認証の構成

23.7 Transport Layer Security使用時の問題

他のOracle製品との通信や、サポート対象の認証方式および暗号化方式のタイプといった、TLSの使用に関する問題に注意する必要があります。

TLSを使用する場合は、次のことを考慮してください。

- TLSを使用すると、Oracle Internet Directoryなどの他のOracle製品との安全な通信が可能になります。
- TLSでは認証と暗号化の両方がサポートされているため、クライアント/サーバー接続が標準のOracle Net TCP/IPトランスポート(ネイティブ暗号化を使用)より多少遅くなります。
- 各TLS認証モードには、構成設定が必要となります。

ノート:

TLS 暗号化を構成する場合は、非 TLS 暗号化を無効にする必要があります。

関連トピック

- ハードウェア・セキュリティ・モジュールを使用するためのシステムの構成
- 厳密認証およびネイティブ・ネットワーク暗号化の無効化

親トピック: Transport Layer Security認証の構成

23.8 クライアント・ウォレットを使用しないTransport Layer Security接続

データベース・サーバーに対して共通のルート証明書を使用するTransport Layer Security (TLS)接続には、クライアント・ウォレットは必要ありません。

- クライアント・ウォレットを使用しないTransport Layer Security接続について
  環境が特定の要件を満たしている場合、クライアント・ウォレットを使用しないTransport Layer Security (TLS)接続を構成できます。
- クライアント・ウォレットを使用しないTransport Layer Security接続の構成
  クライアント・ウォレットを使用しないTransport Layer Security (TLS)を構成する前に、データベースでクライアント認証が必要ないことを確認する必要があります。

親トピック: Transport Layer Security認証の構成

23.8.1 クライアント・ウォレットを使用しないTransport Layer Security接続について

環境が特定の要件を満たしている場合、クライアント・ウォレットを使用しないTransport Layer Security (TLS)接続を構成できます。

環境が次の要件を満たしている場合は、クライアント・ウォレットを使用しないTLS接続を使用することを検討してください。

- クライアント証明書は、データベースへのユーザー認証の手段として使用されません。TLS接続を確立するために必要なのはサーバー証明書のみです。
- サーバー証明書は、システムのデフォルトの証明書ストアに使用可能な証明書(共通のルート証明書)を持つ認証局(CA)によって発行されました。
- Oracleデータベース・サーバーは、TLS接続を許可するように構成されています。(`SSL_CLIENT_AUTHENTICATION=FALSE`を設定)。

データベース・サーバーに対するルート証明書がローカルのシステム証明書ストア内にすでに存在していれば、これが最も一般的なタイプの構成になります。この構成は、クラウド・データベースとオンプレミス・データベースの両方に使用できます。この構成により、クライアントで独自のウォレットを構成することなく、サーバー証明書を検証できます。

次のことに注意してください。

- CおよびInstant Clientのデータベース・ドライバ(およびSQL*Plus)の場合、ウォレットレス機能はMicrosoft WindowsおよびLinuxのx64でのみ使用できます。
- JDBCシン・ドライバの場合、ウォレットレス機能はすべてのプラットフォームで使用できます。

関連トピック

- Oracle Database Net Services管理者ガイド

親トピック: クライアント・ウォレットを使用しないTransport Layer Security接続

23.8.2 クライアント・ウォレットを使用しないTransport Layer Security接続の構成

クライアント・ウォレットを使用しないTransport Layer Security (TLS)を構成する前に、データベースでクライアント認証が必要ないことを確認する必要があります。

CおよびInstant Clientのデータベース・ドライバ(およびSQL*Plus)の場合、ウォレットレス機能はMicrosoft WindowsおよびLinuxのx64でのみ使用できます。JDBCシン・ドライバの場合、ウォレットレス機能はすべてのプラットフォームで使用できます。

1. Oracleデータベースが存在するサーバーにログインします。
2. `sqlnet.ora`ファイルで次の設定を確認します。
   - `AUTHENTICATION_SERVICES=(tcps)`。これにより、データベース接続にTLSが必要になります。
   - `SSL_CLIENT_AUTHENTICATION=FALSE`。デフォルトは`TRUE`で、この場合はmTLS (クライアント・ウォレットでクライアント証明書を必要とする相互TLS)が必要になります。このパラメータを`FALSE`に設定すると、クライアントからのTLS接続とmTLS接続の両方が有効になります。`TRUE`に設定されている場合は、常にmTLSです。

   デフォルトでは、`sqlnet.ora`ファイルは、`$ORACLE_HOME/dbs`ディレクトリ、または`TNS_ADMIN`環境変数によって設定されている場所にあります。

3. WALLET_ROOTシステム・パラメータまたはWALLET_LOCATIONのsqlnet.oraパラメータで定義されたデフォルトの場所にサーバー・ウォレットが存在することを確認します。
4. listener.oraファイルを確認して、TLSが指定されていることを確認します。

```
LISTENER = (ADDRESS=(PROTOCOL=tcps)(HOST=)(PORT=1234))
```

5. WALLET_ROOTシステム・パラメータまたはWALLET_LOCATIONのsqlnet.oraパラメータで設定されたデフォルトの場所にリスナーのウォレットも存在することを確認します。
   新しいクライアント接続を作成する場合は、次の設定が含まれるようにlistener.oraファイルを編集します。

```
ADDRESS=(PROTOCOL=tcps)
```

   デフォルトでは、listener.oraは$ORACLE_HOME/network/adminディレクトリに配置されます。

6. Oracleデータベースのクライアントにログインします。
7. クライアントのsqlnet.oraファイルとtnsnames.oraファイルを変更します。
   - sqlnet.oraファイルのSQLNET.SSL_CLIENT_AUTHENTICATIONの設定を編集します。

     SQLNET.SSL_CLIENT_AUTHENTICATION=FALSEを設定します(デフォルトはTRUEであるため)。FALSEにした場合は、クライアントで、TLSまたはmTLSのどちらかを使用して接続を作成できます。FALSEに設定すると、クライアント側のプライベート証明書に関する情報は送信されなくなります。これは、すべての接続に適用されるため、tnsnames.ora接続文字列のSSL_CLIENT_AUTHENTICATIONパラメータは同じパラメータ設定を使用して変更できます。SSL_CLIENT_AUTHENTICATION=TRUEの場合は、mTLSのみを構成できます。この設定はオプションです。

   - 複数のデータベースに接続し、その一部がクライアント・ウォレットでmTLSを必要とする場合は、次のようにクライアント・ウォレットの有無で異なる接続を設定するための2つのオプションがあります。
     - オプション1: コマンド・ウォレットのsqlnet.oraでWALLET_LOCATIONを設定します。その後、sqlnet.oraの設定をオーバーライドするために、接続文字列でWALLET_LOCATIONを(tnsnames.oraまたは直接コマンドラインで)使用します。接続に別のウォレットの場所を指定することも、システム・デフォルトのキーストアを使用するように接続に指示することもできます。次のパラメータを使用して、ウォレットの場所をシステム・デフォルトのキーストアに変更します。

       ```
       net_service_name = (DESCRIPTION=(ADDRESS = (PROTOCOL=tcps)
       (HOST=host_name)(PORT=port)) (SECURITY=(WALLET_LOCATION=SYSTEM))
       (CONNECT_DATA=(SERVICE_NAME=service_name)))
       ```

       デフォルトの証明書ストアは、Linuxの場合は/etc/pki/tls/cert.pem、Microsoft Windowsの場合はMicrosoft証明書ストアにあります。証明書ストアのデフォルトの場所は変更できません。デフォルトでは、tnsnames.oraは$ORACLE_HOME/network/adminディレクトリに配置されます。

     - オプション2: クライアント・ウォレットを使用する必要がある接続の一部としてのみWALLET_LOCATIONを指定します。sqlnet.oraでは、WALLET_LOCATIONを指定しません。クライアント・ウォレットを使用する必要のない接続は、sqlnet.oraファイルでWALLET_LOCATIONが指定されていない場合、自動的にローカル・デフォルトのシステム・キーストアが使用されます。たとえば:

       ```
       net_service_name = (DESCRIPTION=(ADDRESS = (PROTOCOL=tcps)
       (HOST=host_name)(PORT=port))
       (SECURITY=(WALLET_LOCATION=wallet_file_directory))
       (CONNECT_DATA=(SERVICE_NAME=service_name)))
       ```

8. SQL*Plusで、データベース接続がTLSを使用しているかどうかを確認するには、次の問合せを実行して接続を調べます。

```
SELECT SYS_CONTEXT ('USERENV', 'NETWORK_PROTOCOL') FROM DUAL;
```

次のような出力が表示されます。

```
SYS_CONTEXT('USERENV','NETWORK_PROTOCOL')
------------------------------------------------------------------
tcps
```

関連トピック

- Oracle Databaseリファレンス

親トピック: クライアント・ウォレットを使用しないTransport Layer Security接続

23.9 クライアント・ウォレットを使用するTransport Layer Security接続

Transport Layer Securityは、サーバーで構成してから、クライアントで構成する必要があります。

- ステップ1: サーバーでのTransport Layer Securityの構成
  インストール中、Oracleデータベース・サーバーとOracleクライアントに、Oracleウォレットの場所を除くTLSパラメータのデフォルトが設定されます。
- ステップ2: クライアントでのTransport Layer Securityの構成
  SSLをクライアントで構成する場合、サーバーDNを構成して、TLS付きTCP/IPをクライアントで使用します。
- ステップ3: データベース・インスタンスへのログイン
  構成の完了後、データベースにログインします。

親トピック: Transport Layer Security認証の構成

23.9.1 ステップ1: サーバーでのTransport Layer Securityの構成

インストール中、Oracleデータベース・サーバーとOracleクライアントに、Oracleウォレットの場所を除くTLSパラメータのデフォルトが設定されます。

- ステップ1A: サーバーでのウォレット作成の確認
  次のステップに進む前に、ウォレットが作成されていることと、ウォレットに証明書があることを確認する必要があります。
- ステップ1B: サーバーでのデータベース・ウォレット・ロケーションの指定
  次に、ウォレットのサーバー上の場所を指定します。
- ステップ1C: サーバーでのTransport Layer Security暗号スイートの設定(オプション)
  オプションで、Transport Layer Security暗号スイートを設定できます。
- ステップ1D: サーバーでの必要なTransport Layer Securityバージョンの設定(オプション)
  SSL_VERSIONパラメータでは、サーバーが通信するシステムで実行する必要があるTLSのバージョンを定義します。
- ステップ1E: サーバーでのTransport Layer Securityクライアント認証の設定(オプション)
  SSL_CLIENT_AUTHENTICATIONパラメータは、クライアントがTLSを使用して認証されるかどうかを制御します。
- ステップ1F: サーバーでの認証サービスとしてのTransport Layer Securityの設定(オプション)
  sqlnet.oraファイルのSQLNET.AUTHENTICATION_SERVICESパラメータでは、TLS認証サービスを設定します。
- ステップ1G: サーバーおよびクライアントでのSSLv3の無効化(オプション)
  SSLv3はSecure Sockets Layerバージョン3のことです。

- ステップ1H: Transport Layer Security付きTCP/IPを使用するリスニング・エンドポイントのサーバーでの作成
  TLS付きTCP/IPを使用するリスニング・エンドポイントをサーバーで構成できます。
- ステップ1H: データベースの再起動
  サーバーでのTransport Layer Securityの構成を完了するには、データベースを再起動する必要があります。

親トピック: クライアント・ウォレットを使用するTransport Layer Security接続

23.9.1.1 ステップ1A: サーバーでのウォレット作成の確認

次のステップに進む前に、ウォレットが作成されていることと、ウォレットに証明書があることを確認する必要があります。

1. Oracle Wallet Managerを起動します。
   - (UNIXの場合) `$ORACLE_HOME/bin`から、次のコマンドを入力します。

     ```
     owm
     ```

   - (Windowsの場合)「スタート」→「プログラム」→「Oracle - HOME_NAME」→「Integrated Management Tools」→「Wallet Manager」の順に選択します。

   Oracle Wallet Managerは、Oracle Database 21cでは非推奨です。Oracle Wallet Managerを使用するかわりに、コマンドライン・ツール`orapki`および`mkstore`を使用することをお薦めします。
2. 「ウォレット」メニューから「オープン」を選択します。

   ウォレットにステータスがReadyの証明書が含まれ、自動ログインがオンになっている必要があります。自動ログインがオンになっていない場合は、「ウォレット」メニューから選択し、ウォレットを再度保存します。自動ログインがオンになります。

関連トピック

- Oracle Databaseエンタープライズ・ユーザー・セキュリティ管理者ガイド

親トピック: ステップ1: サーバーでのTransport Layer Securityの構成

23.9.1.2 ステップ1B: サーバーでのデータベース・ウォレット・ロケーションの指定

次に、ウォレットのサーバー上の場所を指定します。

1. Oracle Net Managerを起動します。
   - (UNIX) `$ORACLE_HOME/bin`から、コマンドラインで次のコマンドを入力します。

     ```
     netmgr
     ```

   - (Windows)「スタート」→「プログラム」→「Oracle - HOME_NAME」→「Configuration and Migration Tools」→「Net Manager」を選択します。
2. 「Oracle Netの構成」を展開し、「ローカル」から「プロファイル」を選択します。
3. 「ネーミング」リストから、「ネットワーク・セキュリティ」を選択します。

   ネットワーク・セキュリティのタブ付きウィンドウが表示されます。
4. 「SSL」タブ(TLSに適用される)を選択し、「SSL構成: サーバー」を選択します。
5. 「ウォレット・ディレクトリ」ボックスで、Oracleウォレットが配置されているディレクトリを入力するか、「参照」をクリックし、ファイル・システムを検索してディレクトリを探します。

   エンタープライズ・ユーザー・セキュリティ用にデータベースとディレクトリ間のTLS接続を構成する場合は、Database Configuration Assistantによって自動的にデータベース・ウォレットが作成され、データベースがディレクトリに登録さ

れます。そのウォレットを使用して、TLSで認証されたエンタープライズ・ユーザー・セキュリティのデータベースPKI資格証明を格納する必要があります。

重要:

- Oracle Wallet Managerを使用してウォレットを作成します。Oracleウォレットの新規作成の詳細は、『Oracle Databaseエンタープライズ・ユーザー・セキュリティ管理者ガイド』を参照してください。
- Oracle Net Managerを使用して、`sqlnet.ora`ファイルにウォレット・ロケーションを設定します。マルチテナント環境では、`sqlnet.ora`ファイルの設定はすべてのプラガブル・データベース(PDB)に適用されることに注意してください。

ウォレットを作成したとき、および`sqlnet.ora`ファイルに場所を設定したときと同じ場所を入力してください。

6. 「ファイル」メニューから、「ネットワーク構成の保存」を選択します。

`sqlnet.ora`ファイルと`listener.ora`ファイルが次のエントリで更新されます。

```
wallet_location =
 (SOURCE=
  (METHOD=File)
  (METHOD_DATA=
   (DIRECTORY=wallet_location)))
```

ノート:

リスナーでは、`listener.ora` ファイルに定義されているウォレットを使用します。任意のデータベース・ウォレットを使用できます。Net Manager を使用してサーバーの SSL が構成されている場合、ウォレット・ロケーションは `listener.ora` ファイルと `sqlnet.ora` ファイルに入力されています。`listener.ora` ファイルは、Oracle クライアントには関係ありません。

リスナーが独自のウォレットを所有するようにリスナーのウォレット・ロケーションを変更するには、`listener.ora` を編集して新しい場所を入力します。

親トピック: ステップ1: サーバーでのTransport Layer Securityの構成

23.9.1.3 ステップ1C: サーバーでのTransport Layer Security暗号スイートの設定(オプション)

オプションで、Transport Layer Security暗号スイートを設定できます。

- Transport Layer Security暗号スイートについて
  暗号スイートは、ネットワーク・エンティティ間のメッセージ交換に使用される認証、暗号化およびデータ整合性アルゴリズムのセットです。
- TLS暗号スイートの認証、暗号化、整合性およびTLSバージョン
  Oracle Databaseでは、Oracle Databaseをインストールするとデフォルトで設定される一連の暗号スイートがサポートされています。
- データベース・サーバーのTransport Layer Security暗号スイートの指定
  最初に、データベース・サーバーのTransport Layer Security暗号スイートを指定する必要があります。

親トピック: ステップ1: サーバーでのTransport Layer Securityの構成

23.9.1.3.1 Transport Layer Security暗号スイートについて

暗号スイートは、ネットワーク・エンティティ間のメッセージ交換に使用される認証、暗号化およびデータ整合性アルゴリズムのセットです。

Transport Layer Securityハンドシェイク時に、2つのエンティティがネゴシエートし、メッセージを送受信するときに使用する暗号スイートを確認します。

Oracle Databaseをインストールすると、Transport Layer Security暗号スイートがデフォルトで設定され、リスト順にネゴシエートされます。デフォルトの順序は、`SSL_CIPHER_SUITES`パラメータを設定して上書きできます。`SSL_CIPHER_SUITES`パラメータの設定は必ずカッコで囲んでください(例: `SSL_CIPHER_SUITES=(tls_rsa_with_aes_128_cbc_sha256)`)。そうしないと、暗号スイート設定が正しく解析されません。

暗号スイートの優先順位を設定できます。クライアントが使用する暗号スイートに関してサーバーとネゴシエートする場合、設定されている優先順位に従います。暗号スイートの優先順位を設定する場合は、次の点を考慮してください。

- 互換性。正常に接続するには、互換性のある暗号スイートを使用するようにサーバーとクライアントを構成する必要があります。
- 暗号の優先順位と強度。最高レベルのセキュリティを確保するために、最も強力なものから最も弱いものの順に暗号スイートの優先順位を設定します。
- 使用するセキュリティ・レベル。
- パフォーマンスへの影響。

ノート:

Diffie-Hellman 匿名認証について:

- この暗号スイートを使用するようにサーバーを設定する場合は、同じ暗号スイートをクライアントにも設定する必要があります。設定しない場合、接続に失敗します。
- Diffie-Hellman 匿名を使用する暗号スイートを使用する場合、`SSL_CLIENT_AUTHENTICATION` パラメータを `FALSE` に設定して、サーバーで TLS クライアント認証を構成する必要があります。
- 既知の不具合があり、クライアントを認証しない DH_ANON を含む暗号スイートを使用する場合でも、OCI クライアントでウォレットが必要です。

関連トピック

- TLS暗号スイートの認証、暗号化、整合性およびTLSバージョン
- ステップ1E: サーバーでのTransport Layer Securityクライアント認証の設定(オプション)

親トピック: ステップ1C: サーバーでのTransport Layer Security暗号スイートの設定(オプション)

23.9.1.3.2 TLS暗号スイートの認証、暗号化、整合性およびTLSバージョン

Oracle Databaseでは、Oracle Databaseをインストールするとデフォルトで設定される一連の暗号スイートがサポートされています。

表23-1に、各暗号スイートで使用される認証、暗号化およびデータ整合性のタイプを示します。

表23-1 Transport Layer Security暗号スイート

暗号スイート	認証	暗号化	データ整合性	TLSの互換性
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256	ECDHE_ECDSA	AES 128 GCM	SHA256 (SHA-2)	TLS 1.2 のみ
TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA	ECDHE_ECDSA	AES 128 CBC	SHA-1	TLS 1.0 以降
TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256	ECDHE_ECDSA	AES 128 CBC	SHA256 (SHA-2)	TLS 1.2 のみ
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA	ECDHE_ECDSA	AES 256 CBC	SHA-1	TLS 1.0 以降
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384	ECDHE_ECDSA	AES 256 CBC	SHA384 (SHA-2)	TLS 1.2 のみ
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384	ECDHE_ECDSA	AES 256 GCM	SHA384 (SHA-2)	TLS 1.2 のみ
TLS_RSA_WITH_AES_128_CBC_SHA256	RSA	AES 128 CBC	SHA256 (SHA-2)	TLS 1.2 のみ
TLS_RSA_WITH_AES_128_GCM_SHA256	RSA	AES 128 GCM	SHA256 (SHA-2)	TLS 1.2 のみ
TLS_RSA_WITH_AES_128_CBC_SHA	RSA	AES 128 CBC	SHA-1	TLS 1.0 のみ
TLS_RSA_WITH_AES_256_CBC_SHA	RSA	AES 256 CBC	SHA-1	TLS 1.0 以降
TLS_RSA_WITH_AES_256_CBC_SHA256	RSA	AES 256 CBC	SHA256 (SHA-2)	TLS 1.2 のみ
TLS_RSA_WITH_AES_256_GCM_SHA384	RSA	AES 256 GCM	SHA384 (SHA-2)	TLS 1.2 のみ

表23-2に、使用可能な暗号スイートを示しますが、これらは通信者の認証を提供しないため、第三者攻撃に対して無防備

になる可能性があることに注意してください。機密データを保護する場合は、これらの暗号スイートを使用しないことをお薦めします。ただし、これらは、通信者が匿名を維持する場合や、相互認証によって発生するオーバーヘッドを望まない場合に有効です。

表23-2 SSL_DH Transport Layer Security暗号スイート

暗号スイート	認証	暗号化	データ整合性	TLSの互換性
SSL_DH_anon_WITH_3DES_EDE_CBC_SHA	DH anon	3DES EDE CBC	SHA-1	TLS 3.0 以降

親トピック: ステップ1C: サーバーでのTransport Layer Security暗号スイートの設定(オプション)

23.9.1.3.3 データベース・サーバーのTransport Layer Security暗号スイートの指定

最初に、データベース・サーバーのTransport Layer Security暗号スイートを指定する必要があります。

1. Oracle Net Managerを起動します。
   - (UNIX) $ORACLE_HOME/binから、コマンドラインで次のコマンドを入力します。

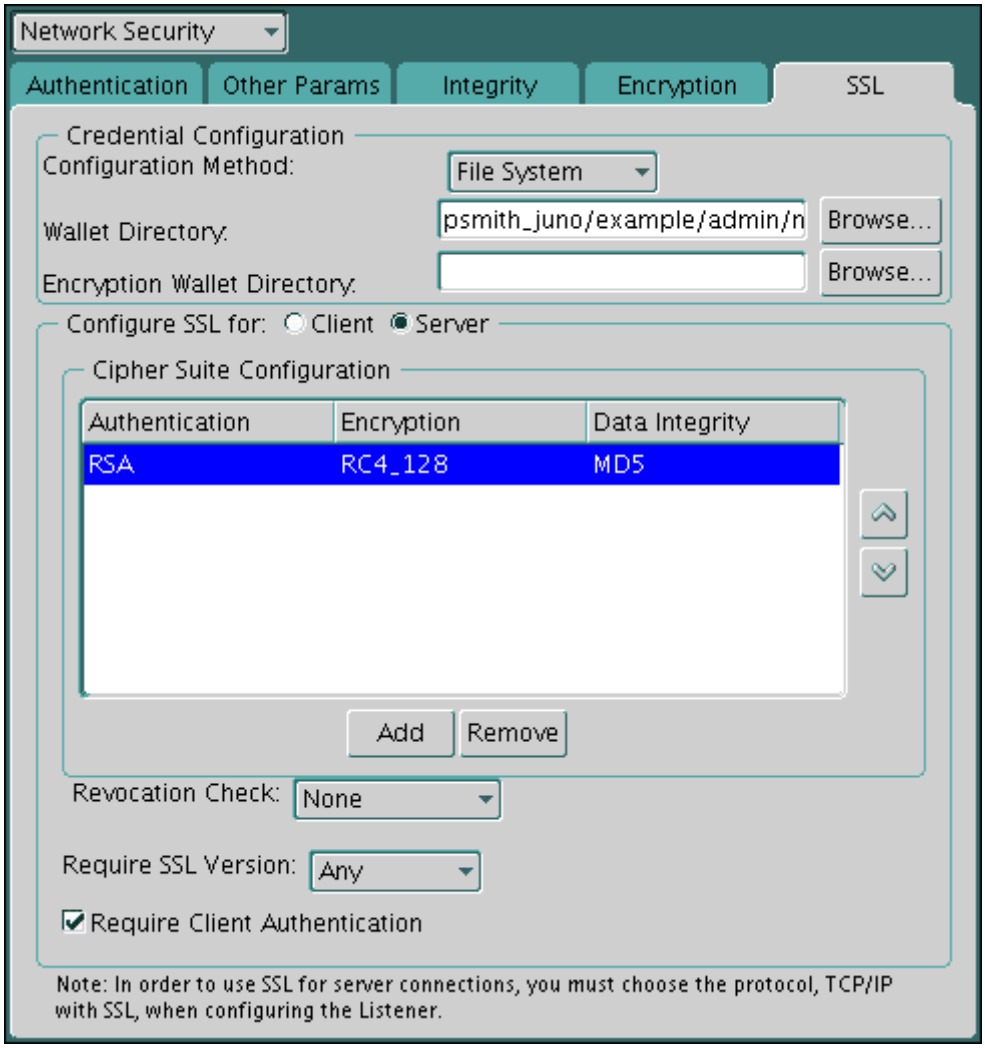
     ```
     netmgr
     ```

   - (Windows)「スタート」→「プログラム」→「Oracle - HOME_NAME」→「Configuration and Migration Tools」→「Net Manager」を選択します。
2. 「Oracle Netの構成」を展開し、「ローカル」から「プロファイル」を選択します。
3. 「ネーミング」リストから、「ネットワーク・セキュリティ」を選択します。

   ネットワーク・セキュリティのタブ付きウィンドウが表示されます。
4. 「SSL」タブを選択し、「SSL構成: サーバー」を選択します。
5. 「暗号スイートの構成」領域で、「追加」をクリックします。

   使用可能な暗号スイートがダイアログ・ボックスに表示されます。米国内用の暗号スイートを表示するには、米国内用の暗号スイートの表示チェック・ボックスをクリックします。
6. スイートを選択し、「OK」をクリックします。

   「暗号スイートの構成」リストが更新されます。



7. 上矢印と下矢印を使用して、暗号スイートの優先順位を設定します。
8. 「ファイル」メニューから、「ネットワーク構成の保存」を選択します。

   sqlnet.oraファイルが次のエントリで更新されます。

```
SSL_CIPHER_SUITES= (SSL_cipher_suite1 [,SSL_cipher_suite2])
```

親トピック: ステップ1C: サーバーでのTransport Layer Security暗号スイートの設定(オプション)

23.9.1.4 ステップ1D: サーバーでの必要なTransport Layer Securityバージョンの設定(オプション)

SSL_VERSIONパラメータでは、サーバーが通信するシステムで実行する必要があるTLSのバージョンを定義します。

オプションで、sqlnet.oraファイルまたはlistener.oraファイルでSSL_VERSIONパラメータを設定できます。

これらのシステムで有効なバージョンを使用するように設定できます。sqlnet.oraにおけるこのパラメータのデフォルト設定はundeterminedで、これは、「ネットワーク・セキュリティ」ウィンドウの「SSL」タブのリストから「任意」を選択すると設定されます。

1. 「必要なSSLバージョン」リストのデフォルトは「任意」です。

   このデフォルトを受け入れるか、使用するSSLバージョンを選択します。

2. 「ファイル」メニューから、「ネットワーク構成の保存」を選択します。

   「任意」を選択した場合は、sqlnet.oraファイルが次のエントリで更新されます。

```
SSL_VERSION=UNDETERMINED
```

ノート:

SSL 2.0 はサーバー側でサポートされていません。

関連トピック

- 『Oracle Database Net Servicesリファレンス・ガイド』

親トピック: ステップ1: サーバーでのTransport Layer Securityの構成

23.9.1.5 ステップ1E: サーバーでのTransport Layer Securityクライアント認証の設定(オプション)

SSL_CLIENT_AUTHENTICATIONパラメータは、クライアントがTLSを使用して認証されるかどうかを制御します。

このパラメータはサーバーのsqlnet.oraファイルで設定する必要があります。SSL_CLIENT_AUTHENTICATIONパラメータのデフォルト値はTRUEです。

Diffie-Hellman匿名認証(DH_anon)を含む暗号スイートを使用する場合は、SSL_CLIENT_AUTHENTICATIONをFALSEに設定できます。

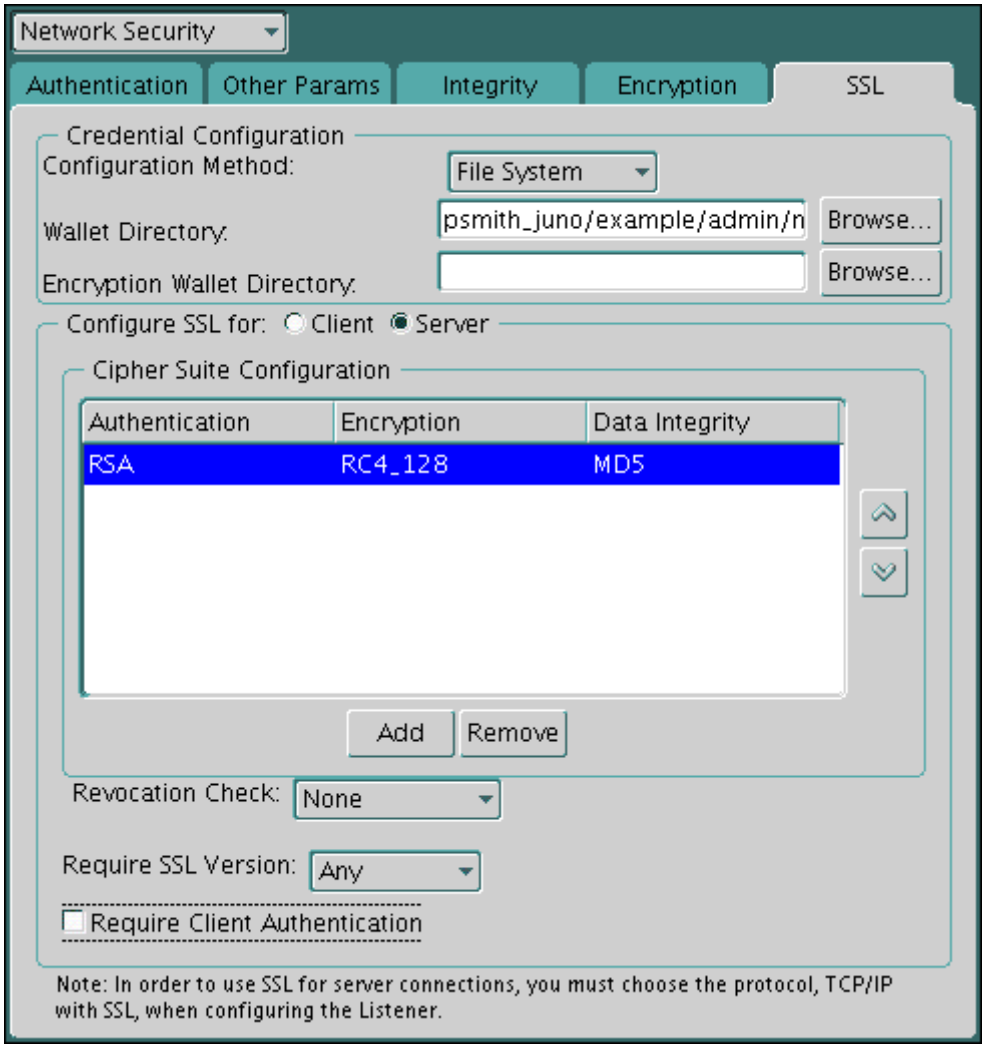
また、KerberosやRADIUSなどのOracle Databaseでサポートされている非SSL認証方式を使用してクライアントがサーバーに対して自己認証を行うようにする場合も、このパラメータをFALSEに設定できます。

ノート:

既知の不具合があり、クライアントを認証しない DH_ANON を含む暗号スイートを使用する場合でも、OCI クライアントでウォレットが必要です。

サーバーでSSL_CLIENT_AUTHENTICATIONをFALSEに設定するには:

1. Oracle Net Managerの「SSL」ページで、「クライアントの認証が必要」の選択を解除します。



2. 「ファイル」メニューから、「ネットワーク構成の保存」を選択します。

   `sqlnet.ora`ファイルが次のエントリで更新されます。

```
SSL_CLIENT_AUTHENTICATION=FALSE
```

親トピック: ステップ1: サーバーでのTransport Layer Securityの構成

23.9.1.6 ステップ1F: サーバーでの認証サービスとしてのTransport Layer Securityの設定(オプション)

`sqlnet.ora`ファイルの`SQLNET.AUTHENTICATION_SERVICES`パラメータでは、TLS認証サービスを設定します。

このパラメータは、TLS認証とOracle Databaseでサポートされている別の認証方式を組み合せて使用する場合に設定します。たとえば、サーバーがTLSを使用してクライアントに対して自己認証を行い、クライアントがKerberosを使用してサーバーに対して自己認証を行う場合に、このパラメータを使用します。

- サーバーに`SQLNET.AUTHENTICATION_SERVICES`パラメータを設定するには、テキスト・エディタを使用して`sqlnet.ora`ファイルでこのパラメータにTLS付きTCP/IP(TCPS)を追加します。たとえば、SSL認証とRADIUS認証を組み合せて使用する場合は、このパラメータを次のように設定します。

```
SQLNET.AUTHENTICATION_SERVICES = (TCPS, radius)
```

TLS認証と別の認証方式を組み合せて使用しない場合は、このパラメータを設定しません。

親トピック: ステップ1: サーバーでのTransport Layer Securityの構成

23.9.1.7 ステップ1G: サーバーおよびクライアントでのSSLv3の無効化(オプション)

SSLv3はSecure Sockets Layerバージョン3のことです。

Secure Sockets Layerバージョン3(SSLv3)をサポートするアプリケーションは、Transport Layer Security (TLS)の最新のバージョンを使用している場合でも、Padding Oracle On Downgraded Legacy Encryption(POODLE)攻撃に対して脆弱です。POODLE攻撃を防ぐには、サーバーとクライアントの両方でADD_SSLV3_TO_DEFAULT sqlnet.oraパラメータをFALSEに設定する必要があります。ADD_SSLV3_TO_DEFAULTは、SSL_VERSIONパラメータが設定されていない場合にのみ適用されます。(SSLバージョンのデフォルト・リストが使用されることを意味します)。

1. データベース・サーバーまたはクライアント・サーバーにログインします。
2. sqlnet.oraパラメータ・ファイル(デフォルトで$ORACLE_HOME/network/adminディレクトリにある)を次のように編集して、ADD_SSLV3_TO_DEFAULTパラメータを含むようにします。

   ```
   ADD_SSLV3_TO_DEFAULT=false
   ```

   ADD_SSLV3_TO_DEFAULTはデフォルトでFALSEに設定され、SSL_VERSIONが明示的に設定されている場合は効果がありません。したがって、デフォルトでは、SSL_VERSIONを明示的に設定しないと、SSLv3接続は許可されません。ADD_SSLV3_TO_DEFAULTをTRUEに設定すると、SSLv3接続がデフォルトで動作を続行できるようになります。

親トピック: ステップ1: サーバーでのTransport Layer Securityの構成

23.9.1.8 ステップ1H: Transport Layer Security付きTCP/IPを使用するリスニング・エンドポイントのサーバーでの作成

TLS付きTCP/IPを使用するリスニング・エンドポイントをサーバーで構成できます。

1. listener.oraファイルでリスナーを構成します。一般的なOracle Netクライアントには、ポート番号2484を使用することをお薦めします
2. データベースを再起動します。

関連トピック

- 『Oracle Database Net Servicesリファレンス・ガイド』

親トピック: ステップ1: サーバーでのTransport Layer Securityの構成

23.9.1.9 ステップ1H: データベースの再起動

サーバーでのTransport Layer Securityの構成を完了するには、データベースを再起動する必要があります。

- 次に例を示します。

  ```
  SHUTDOWN IMMEDIATE
  STARTUP
  ```

親トピック: ステップ1: サーバーでのTransport Layer Securityの構成

23.9.2 ステップ2: クライアントでのTransport Layer Securityの構成

SSLをクライアントで構成する場合、サーバーDNを構成して、TLS付きTCP/IPをクライアントで使用します。

- ステップ2A: クライアント・ウォレット作成の確認
  クライアントでウォレットが作成されていることと、クライアントに有効な証明書があることを確認する必要があります。

- ステップ2B: サーバーDN一致の構成とクライアントでのTLS付きTCP/IPの使用
  次に、サーバーDN一致を構成し、クライアントでTransport Layer Security (TLS)付きのTCP/IPを使用します。
- ステップ2C: 必要なクライアントTLS構成の指定(ウォレット・ロケーション)
  Oracle Net Managerを使用して、必要なクライアントTLS構成を指定できます。
- ステップ2D: クライアントのTransport Layer Security暗号スイートの設定(オプション)
  オプションで、Transport Layer Security暗号スイートを設定できます。Oracle Databaseにはデフォルトの暗号スイート設定が用意されています。
- ステップ2E: 必要なTLSバージョンのクライアントでの設定(オプション)
  SSL_VERSIONパラメータでは、クライアントが通信するシステムで実行する必要があるTLSのバージョンを定義します。
- ステップ2F: クライアントにおける認証サービスとしてのTLSの設定(オプション)
  sqlnet.oraファイルのSQLNET.AUTHENTICATION_SERVICESパラメータでは、TLS認証サービスを設定します。
- ステップ2G: クライアントでの認証に使用する証明書の指定(オプション)
  証明書が複数ある場合は、sqlnet.oraファイルのSQLNET.SSL_EXTENDED_KEY_USAGEパラメータで正しい証明書を指定できます。
- ステップ2H: 接続でTransport Layer Securityが使用されていることの確認
  動的ビューV$SESSIONとV$SESSION_CONNECT_INFOを問い合せると、クライアント接続でTransport Layer Security (TLS)が使用されていることを確認できます。
- ステップ2I: データベースの再起動
  クライアントでのTransport Layer Securityの構成を完了するには、データベースを再起動する必要があります。

親トピック: クライアント・ウォレットを使用するTransport Layer Security接続

23.9.2.1 ステップ2A: クライアント・ウォレット作成の確認

クライアントでウォレットが作成されていることと、クライアントに有効な証明書があることを確認する必要があります。

- Oracle Wallet Managerを使用して、「ウォレット」メニュー→「開く」の順に選択し、ウォレットが作成されていることを確認します。
  Oracle Wallet Managerを使用して、各認証局に関連付けられたOracleウォレット内の使用していない信頼できる証明書を削除することをお薦めします。

関連トピック

- ステップ1A: サーバーでのウォレット作成の確認
- Oracle Databaseエンタープライズ・ユーザー・セキュリティ管理者ガイド

親トピック: ステップ2: クライアントでのTransport Layer Securityの構成

23.9.2.2 ステップ2B: サーバーDN一致の構成とクライアントでのTLS付きTCP/IPの使用

次に、サーバーDN一致を構成し、クライアントでTransport Layer Security (TLS)付きのTCP/IPを使用します。

- サーバーDN一致の構成とクライアントでのTLS付きTCP/IPの使用について
  サーバー証明書の証明書チェーンの検証に加えて、サーバーDN一致を使用して追加のチェックを実行できます。
- サーバーDN一致の構成とクライアントでのTLS付きTCP/IPの使用
  サーバーDN一致を構成し、クライアントでTLS付きのTCP/IPを使用するようにtnsnames.oraファイルとlistener.oraファイルを編集する必要があります。

親トピック: ステップ2: クライアントでのTransport Layer Securityの構成

23.9.2.2.1 サーバーDN一致の構成とクライアントでのTLS付きTCP/IPの使用について

サーバー証明書の証明書チェーンの検証に加えて、サーバーDN一致を使用して追加のチェックを実行できます。

サーバーDN一致を含み、クライアントでTLS付きのTCP/IPを使用するようにOracle Net Service名を構成できます。これを実行するには、クライアント・ネットワーク構成ファイルで、サーバーDNマッチングおよびTCP/IPをTLS接続できるように、プロトコルとしてサーバーの識別名(DN)とTCPSを指定する必要があります。サーバーDN一致はオプションですが、クライアントにセキュリティのレイヤーが追加されるため、お薦めします。これにより、クライアントはサーバーに対してこのチェックを実行できます。

2022年から組織単位(OU)フィールドが削除されるCA証明書形式の変更により、SSL_SERVER_DN_MATCHをTRUEに設定した場合は、サーバー証明書DNを更新する必要がある場合があります。DNからOUが削除された新しいサーバー証明書を受け取ったら、新しいDNと一致するようにクライアントのSSL_SERVER_CERT_DNパラメータを更新する必要があります。

部分DN一致または完全DN一致のいずれかを構成できます。SSL_SERVER_DN_MATCHパラメータをTRUEに設定すると、部分DN一致が自動的に実行されます。次に、クライアントはDN情報のサーバー証明書をチェックします。完全DN一致により、クライアントはサーバーの完全なDNと照合できます。完全DN一致を実行する場合は、SSL_SERVER_CERT_DNパラメータにサーバーのDNを指定する必要があります。

部分または完全DN一致のいずれかを使用すると、ホスト証明書の作成方法と管理方法に基づいて柔軟性が向上します。たとえば、クライアントが`hostname=finance.us.example.com`を使用してサーバーへの接続を試みるとします。部分DN一致では、クライアントはサーバーの証明書をチェックして、`CN=finance`がサーバーのDNであることを確認します。部分DN一致の場合、ホスト名(`finance`)のみがチェックされ、完全修飾ドメイン名(`finance.us.example.com`)はチェックされません。完全DN一致と部分DN一致の両方について、SSL_SERVER_DN_MATCHパラメータをTRUEに設定する必要があります。

クライアント・ネットワーク構成ファイルの`tnsnames.ora`を手動で編集して、サーバーのDNとTCP/IPをSSLプロトコルに指定する必要があります。`tnsnames.ora`ファイルは、クライアントまたはLDAPディレクトリに配置できます。サーバーに配置される場合は、通常、`listener.ora`ファイルと同じディレクトリに存在します。`tnsnames.ora`ファイルは、通常、TNS_ADMIN環境変数で指定された設定にあります。TNS_ADMINが設定されていない場合、`tnsnames.ora`は次のディレクトリの場所に存在します。

- (UNIXの場合) `$ORACLE_HOME/network/admin/`
- (Windowsの場合) `ORACLE_BASE¥ORACLE_HOME¥network¥admin¥`

親トピック: ステップ2B: サーバーDN一致の構成とクライアントでのTLS付きTCP/IPの使用

23.9.2.2.2 サーバーDN一致の構成とクライアントでのTLS付きTCP/IPの使用

サーバーDN一致を構成し、クライアントでTLS付きのTCP/IPを使用するように`tnsnames.ora`ファイルと`listener.ora`ファイルを編集する必要があります。

1. クライアントの`tnsnames.ora`ファイルで、SSL_SERVER_CERT_DNパラメータを検索し、次を実行します。
    - 完全DN一致を使用する場合は、次のようにSSL_SERVER_CERT_DNを完全なDNに設定します。

      ```
      (SECURITY=
      (SSL_SERVER_CERT_DN="finance,cn=OracleContext,c=us,o=example"))
      ```

      クライアントは、この情報を使用して、各サーバーに予定しているDNのリストを取得して、サーバーのDNとそのサービス名が確実に一致するようにします。次の例では、`tnsnames.ora`ファイルでの`finance`データベースのエントリを示しています。

      ```
      finance=
      (DESCRIPTION=
      (ADDRESS_LIST=
      ```

```
(ADDRESS= (PROTOCOL = tcps) (HOST = finance) (PORT = 1575)))
(CONNECT_DATA=
(SERVICE_NAME= finance.us.example.com))
(SECURITY=
(SSL_SERVER_CERT_DN="cn=finance,cn=OracleContext,c=us,o=example"))
```

デフォルトで、tnsnames.oraとlistener.oraファイルは$ORACLE_HOME/network/adminディレクトリ(UNIXシステムの場合)またはORACLE_HOME¥network¥admin (Windowsの場合)にあります。

- 部分DN一致を使用する場合は、tnsnames.oraにSSL_SERVER_CERT_DNパラメータを含めないでください。

2. クライアントのtnsnames.oraファイルに、ADDRESSパラメータのPROTOCOLとしてtcpsと入力します。

   この指定により、クライアントはTLS付きTCP/IPを使用して、SERVICE_NAMEパラメータに指定されたデータベースに接続します。次に、tnsnames.oraファイル内の接続プロトコルとしてTLS付きTCP/IPを指定するエントリも示します。

```
LISTENER=
(DESCRIPTION_LIST=
(DESCRIPTION=
(ADDRESS= (PROTOCOL = tcps) (HOST = finance) (PORT = 1575))))
```

3. listener.oraファイルに、ADDRESSパラメータのPROTOCOLとしてtcpsと入力します。

親トピック: ステップ2B: サーバーDN一致の構成とクライアントでのTLS付きTCP/IPの使用

23.9.2.3 ステップ2C: 必要なクライアントTLS構成の指定(ウォレット・ロケーション)

Oracle Net Managerを使用して、必要なクライアントTLS構成を指定できます。

1. Oracle Net Managerを起動します。
   - (UNIX) $ORACLE_HOME/binから、コマンドラインで次のコマンドを入力します。

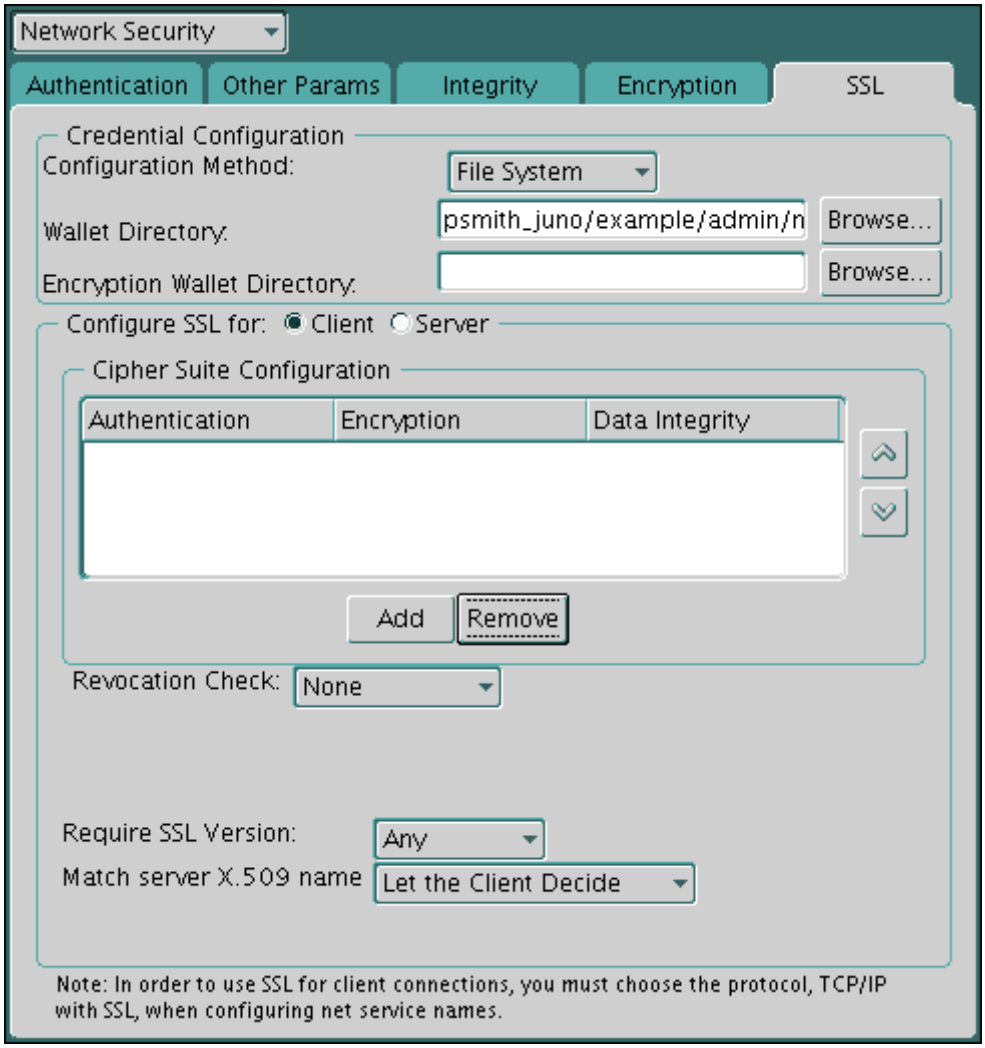
```
netmgr
```

   - (Windows)「スタート」→「プログラム」→「Oracle - HOME_NAME」→「Configuration and Migration Tools」→「Net Manager」を選択します。

2. 「Oracle Netの構成」を展開し、「ローカル」から「プロファイル」を選択します。
3. 「ネーミング」リストから、「ネットワーク・セキュリティ」を選択します。

   ネットワーク・セキュリティのタブ付きウィンドウが表示されます。

4. 「SSL」タブを選択します。
5. 「SSL構成: クライアント」を選択します。



6. 「ウォレット・ディレクトリ」ボックスで、Oracleウォレットが配置されているディレクトリを入力するか、「参照」をクリックし、ファイル・システムを検索してディレクトリを探します。
7. 「サーバーX.509の名前に一致」リストから、次のいずれかのオプションを選択します。
   - はい: サーバーの識別名(DN)がそのサービス名と一致している必要があります。TLSによって証明書がサーバーからのものであることが確認され、一致した場合は接続が成功します。

     このチェックは、デフォルト設定であるRSA暗号が選択されている場合にのみ実行できます。
   - いいえ(デフォルト): TLSによってDNとサービス名が一致するかどうかが確認されますが、強制はされません。結果に関係なく接続は成功しますが、一致しない場合はエラーが記録されます。
   - クライアントで決定: デフォルトを有効にします。

     「いいえ」を選択すると、次のアラートが表示されます。

     ```
     Security Alert
     Not enforcing the server X.509 name match allows a server to potentially
     fake its identity. Oracle recommends selecting YES for this option so
     that connections are refused when there is a mismatch.
     ```

8. 「ファイル」メニューから、「ネットワーク構成の保存」を選択します。

   クライアントの`sqlnet.ora`ファイルが次のエントリで更新されます。

```
SSL_CLIENT_AUTHENTICATION =TRUE
wallet_location =
 (SOURCE=
  (METHOD=File)
  (METHOD_DATA=
```

```
   (DIRECTORY=wallet_location)))
SSL_SERVER_DN_MATCH=(ON/OFF)
```

関連トピック

- Transport Layer Security X.509サーバー照合パラメータ

親トピック: ステップ2: クライアントでのTransport Layer Securityの構成

23.9.2.4 ステップ2D: クライアントのTransport Layer Security暗号スイートの設定(オプション)

オプションで、Transport Layer Security暗号スイートを設定できます。Oracle Databaseにはデフォルトの暗号スイート設定が用意されています。

- クライアントのTransport Layer Security暗号スイートの設定について
  暗号スイートは、ネットワーク・エンティティ間のメッセージ交換に使用される認証、暗号化およびデータ整合性アルゴリズムのセットです。
- クライアントのTransport Layer Security暗号スイートの設定
  Oracle Net Managerを使用して、クライアントTLS暗号スイートを設定できます。

親トピック: ステップ2: クライアントでのTransport Layer Securityの構成

23.9.2.4.1 クライアントのTransport Layer Security暗号スイートの設定について

暗号スイートは、ネットワーク・エンティティ間のメッセージ交換に使用される認証、暗号化およびデータ整合性アルゴリズムのセットです。

SSLハンドシェイク時に、2つのエンティティがネゴシエートし、メッセージを送受信するときに使用する暗号スイートを確認します。

Oracle Databaseをインストールすると、TLS暗号スイートがデフォルトで設定されます。2つのエンティティが接続をネゴシエートするときに、この表のリスト順で暗号スイートが試行されます。デフォルトは、SSL_CIPHER_SUITESパラメータを設定して上書きできます。たとえば、Oracle Net Managerを使用して暗号スイートSSL_RSA_WITH_RC4_128_SHAを追加する場合、デフォルト設定の他のすべての暗号スイートは無視されます。

暗号スイートの優先順位を設定できます。クライアントが使用する暗号スイートに関してサーバーとネゴシエートする場合、設定されている優先順位に従います。暗号スイートの優先順位を設定する場合は、次の点を考慮してください。

- 使用するセキュリティ・レベル。たとえば、AES暗号化はDESよりも強力です。
- パフォーマンスへの影響。たとえば、Triple-DES暗号化は、DESよりも低速です。
- 管理要件。クライアントに対して選択された暗号スイートは、サーバーに必要な暗号スイートと互換性がある必要があります。たとえば、Oracle Call Interface (OCI)ユーザーの場合、サーバーはクライアントに自己認証を求めます。この場合、証明書の交換を許可しないDiffie-Hellman匿名認証を利用する暗号スイートは使用できません。

通常、暗号スイートは最も強力なものから弱いものの順に優先順位を設定します。

現在サポートされているTransport Layer Security暗号スイートは、Oracle Databaseのインストール時にデフォルトで設定されます。表では、各暗号スイートで使用される認証、暗号化およびデータ整合性のタイプも示されています。

ノート:

sqlnet.ora ファイルで SSL_CLIENT_AUTHENTICATION パラメータを true に設定する場合は、Diffie-

Hellman 匿名認証を使用するすべての暗号スイートを無効にします。設定しない場合、接続に失敗します。

関連トピック

- TLS暗号スイートの認証、暗号化、整合性およびTLSバージョン

親トピック: ステップ2D: クライアントのTransport Layer Security暗号スイートの設定(オプション)

23.9.2.4.2 クライアントのTransport Layer Security暗号スイートの設定

Oracle Net Managerを使用して、クライアントTLS暗号スイートを設定できます。

1. Oracle Net Managerを起動します。
   - (UNIX) `$ORACLE_HOME/bin`から、コマンドラインで次のコマンドを入力します。

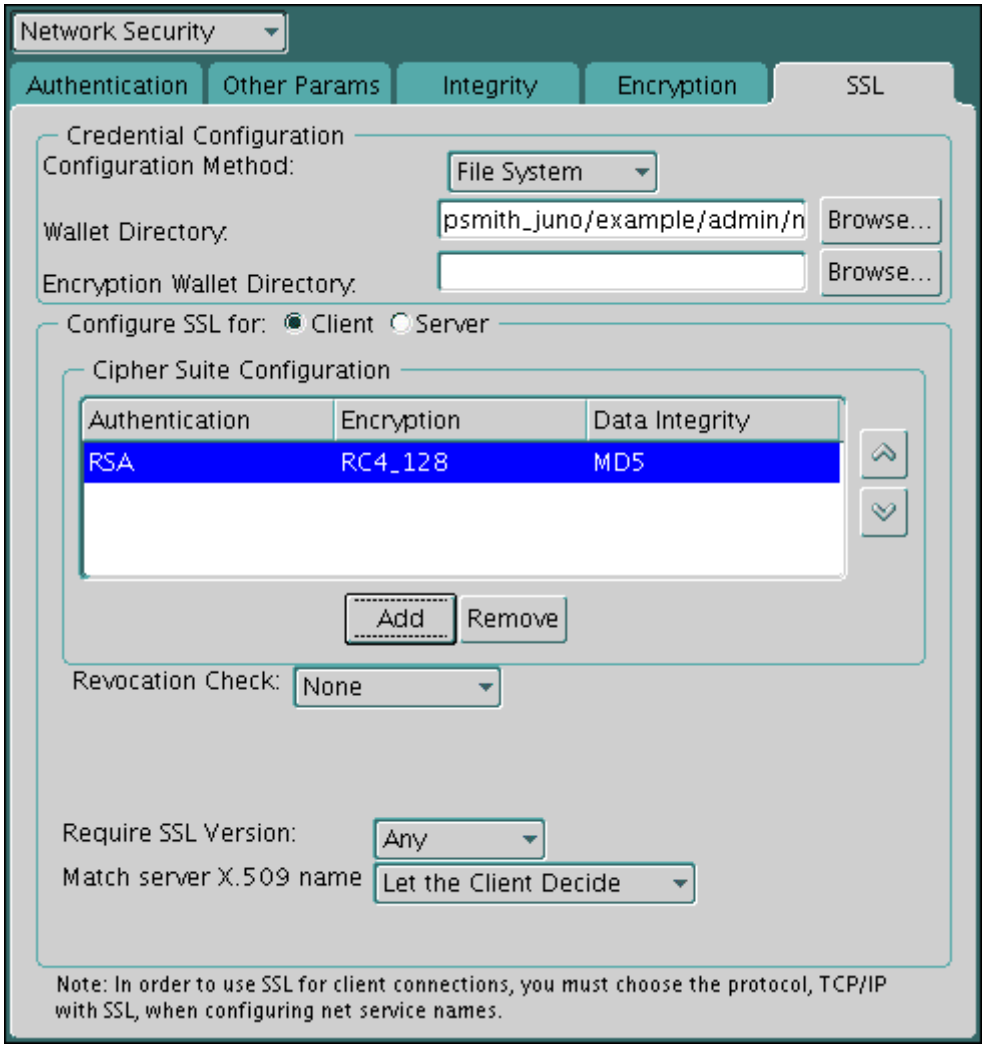
     ```
     netmgr
     ```

   - (Windows)「スタート」→「プログラム」→「Oracle - HOME_NAME」→「Configuration and Migration Tools」→「Net Manager」を選択します。
2. 「Oracle Netの構成」を展開し、「ローカル」から「プロファイル」を選択します。
3. 「ネーミング」リストから、「ネットワーク・セキュリティ」を選択します。

   ネットワーク・セキュリティのタブ付きウィンドウが表示されます。
4. 「SSL」タブを選択します。
5. 「暗号スイートの構成」領域で、「追加」をクリックします。

   使用可能な暗号スイートがダイアログ・ボックスに表示されます。
6. スイートを選択し、「OK」をクリックします。

   次に示すように、「暗号スイートの構成」リストが更新されます。



7. 上矢印と下矢印を使用して、暗号スイートの優先順位を設定します。
8. 「ファイル」メニューから、「ネットワーク構成の保存」を選択します。

   sqlnet.oraファイルが次のエントリで更新されます。

```
SSL_CIPHER_SUITES= (SSL_cipher_suite1 [,SSL_cipher_suite2])
```

親トピック: ステップ2D: クライアントのTransport Layer Security暗号スイートの設定(オプション)

23.9.2.5 ステップ2E: 必要なTLSバージョンのクライアントでの設定(オプション)

SSL_VERSIONパラメータでは、クライアントが通信するシステムで実行する必要があるTLSのバージョンを定義します。

sqlnet.oraファイルでSSL_VERSIONパラメータを設定する必要があります。これらのシステムで有効なバージョンを使用するように設定できます。

sqlnet.oraにおけるこのパラメータのデフォルト設定はundeterminedで、これは、「ネットワーク・セキュリティ」ウィンドウの「SSL」タブのリストから「任意」を選択すると設定されます。「任意」を選択すると、TLS 1.0が最初に試行され、その後、TLS 3.0、TLS 2.0の順に試行されます。クライアントのTLSバージョンがサーバーで使用されているバージョンと互換性があることを確認します。

1. 「必要なSSLバージョン」リストで、構成するSSLバージョンを選択します。

   デフォルトの設定は「任意」です。

2. 「ファイル」メニューから、「ネットワーク構成の保存」を選択します。

   sqlnet.oraファイルが更新されます。「任意」を選択した場合は、次のエントリで更新されます。

```
SSL_VERSION=UNDETERMINED
```

親トピック: ステップ2: クライアントでのTransport Layer Securityの構成

23.9.2.6 ステップ2F: クライアントにおける認証サービスとしてのTLSの設定(オプション)

sqlnet.oraファイルのSQLNET.AUTHENTICATION_SERVICESパラメータでは、TLS認証サービスを設定します。

- SQLNET.AUTHENTICATION_SERVICESパラメータについて
  SQLNET.AUTHENTICATION_SERVICESパラメータは、TLS認証とOracle Databaseでサポートされている別の認証方式の併用を可能にします。
- SQLNET.AUTHENTICATION_SERVICESパラメータの設定
  sqlnet.oraファイルでSQLNET.AUTHENTICATION_SERVICESパラメータを設定できます。

親トピック: ステップ2: クライアントでのTransport Layer Securityの構成

23.9.2.6.1 SQLNET.AUTHENTICATION_SERVICESパラメータについて

SQLNET.AUTHENTICATION_SERVICESパラメータは、TLS認証とOracle Databaseでサポートされている別の認証方式の併用を可能にします。

たとえば、サーバーがTLSを使用してクライアントに対して自己認証を行い、クライアントがRADIUSを使用してサーバーに対して自己認証を行う場合に、このパラメータを使用します。

SQLNET.AUTHENTICATION_SERVICESパラメータを設定するには、sqlnet.oraファイル(他のネットワーク構成ファイルと同じディレクトリにあります)を編集する必要があります。

プラットフォームに応じて、sqlnet.oraファイルは次のディレクトリにあります。

- (UNIXの場合) $ORACLE_HOME/network/admin
- (Windowsの場合) ORACLE_BASE¥ORACLE_HOME¥network¥admin¥

親トピック: ステップ2F: クライアントにおける認証サービスとしてのTLSの設定(オプション)

23.9.2.6.2 SQLNET.AUTHENTICATION_SERVICESパラメータの設定

sqlnet.oraファイルでSQLNET.AUTHENTICATION_SERVICESパラメータを設定できます。

- クライアントのSQLNET.AUTHENTICATION_SERVICESパラメータを設定するには、テキスト・エディタを使用して、sqlnet.oraファイルのこのパラメータにTLS付きTCP/IP (TCPS)を追加します。

  たとえば、TLS認証とRADIUS認証を組み合せて使用する場合は、このパラメータを次のように設定します。

  ```
  SQLNET.AUTHENTICATION_SERVICES = (TCPS, radius)
  ```

TLS認証と別の認証方式を組み合せて使用しない場合は、このパラメータを設定しません。

親トピック: ステップ2F: クライアントにおける認証サービスとしてのTLSの設定(オプション)

23.9.2.7 ステップ2G: クライアントでの認証に使用する証明書の指定(オプション)

証明書が複数ある場合は、sqlnet.oraファイルのSQLNET.SSL_EXTENDED_KEY_USAGEパラメータで正しい証明書を指定できます。

- SQLNET.SSL_EXTENDED_KEY_USAGEパラメータについて
  sqlnet.oraファイルのSQLNET.SSL_EXTENDED_KEY_USAGEパラメータは、データベース・サーバー認証時に使

用する証明書を指定します。

- SQLNET.SSL_EXTENDED_KEY_USAGEパラメータの設定
  SQLNET.SSL_EXTENDED_KEY_USAGEを設定して、クライアント認証を設定できます。

親トピック: ステップ2: クライアントでのTransport Layer Securityの構成

23.9.2.7.1 SQLNET.SSL_EXTENDED_KEY_USAGEパラメータについて

sqlnet.oraファイルのSQLNET.SSL_EXTENDED_KEY_USAGEパラメータは、データベース・サーバー認証時に使用する証明書を指定します。

セキュリティ・モジュールに複数の証明書があるが、1つの証明書のみにクライアント認証の拡張キー使用方法フィールドがあり、その証明書がまさに、データベースの認証に使用する証明書である場合、SQLNET.SSL_EXTENDED_KEY_USAGEパラメータを設定します。

たとえば、スマートカードに複数の証明書があり、そのうちの1つのみにclient authenticationの拡張キー使用方法フィールドがあり、その証明書Cをデータベースの認証に使用する場合、このパラメータを設定します。Windowsクライアントでこのパラメータをclient authenticationに設定すると、MSCAPI証明書の選択ボックスが表示され、証明書Cが自動的に、サーバーへのクライアントのTransport Layer Security認証に使用されます。

親トピック: ステップ2G: クライアントでの認証に使用する証明書の指定(オプション)

23.9.2.7.2 SQLNET.SSL_EXTENDED_KEY_USAGEパラメータの設定

SQLNET.SSL_EXTENDED_KEY_USAGEを設定して、クライアント認証を設定できます。

- クライアントのSQLNET.SSL_EXTENDED_KEY_USAGEパラメータを設定するには、sqlnet.oraファイルを編集し、次の行を含めます。

```
SQLNET.SSL_EXTENDED_KEY_USAGE = "client authentication"
```

証明書のフィルタ処理を使用しない場合は、sqlnet.oraファイルからSQLNET.SSL_EXTENDED_KEY_USAGEパラメータ設定を削除します。

親トピック: ステップ2G: クライアントでの認証に使用する証明書の指定(オプション)

23.9.2.8 ステップ2H: 接続でTransport Layer Securityが使用されていることの確認

動的ビューのV$SESSIONとV$SESSION_CONNECT_INFOを問い合せると、クライアント接続がTransport Layer Security (TLS)を使用していることを確認できます。

- SQL*Plusで、次の問合せを実行します。

```
 SELECT SYS_CONTEXT ('USERENV', 'NETWORK_PROTOCOL') FROM DUAL;
```

次のような出力が表示されます。

```
SYS_CONTEXT('USERENV','NETWORK_PROTOCOL')
------------------------------------------------------------------------
tcps
```

親トピック: ステップ2: クライアントでのTransport Layer Securityの構成

23.9.2.9 ステップ2I: データベースの再起動

クライアントでのTransport Layer Securityの構成を完了するには、データベースを再起動する必要があります。

- 次に例を示します。

```
SHUTDOWN IMMEDIATE
STARTUP
```

親トピック: ステップ2: クライアントでのTransport Layer Securityの構成

23.9.3 ステップ3: データベース・インスタンスへのログイン

構成の完了後、データベースにログインします。

- SQL*Plusを起動して、次のいずれかの接続コマンドを入力します。
  - クライアントにTransport Layer Security認証を使用する場合は(`sqlnet.ora`ファイルで`SSL_CLIENT_AUTHENTICATION=true`を設定)、次のように入力します。

    ```
    CONNECT/@net_service_name
    ```

  - Transport Layer Security認証を使用しない場合は(`sqlnet.ora`ファイルで`SSL_CLIENT_AUTHENTICATION=false`を設定)、次のように入力します。

    ```
    CONNECT username@net_service_name
    Enter password: password
    ```

関連トピック

- 証明書失効リストによる証明書の検証

親トピック: クライアント・ウォレットを使用するTransport Layer Security接続

23.10 Oracle Real Application Clusters環境でのTransport Layer Security接続

Oracle RACツールを使用してOracle Database構成ファイルを変更することで、Oracle Real Application Clusters (Oracle RAC)環境でTransport Layer Security (TLS)接続を構成できます。

- ステップ1: TCPSプロトコル・エンドポイントの構成
  Oracle Real Application Clusters (Oracle RAC)では、クライアントは3つのスキャン・リスナーのいずれかにアクセスし、その後、データベース・リスナーにルーティングされます。Transport Layer Security (TLS)をサポートするには、これらのリスナーすべてにTCPSプロトコル・エンドポイントが必要です。
- ステップ2: 各ノードでLOCAL_LISTENERパラメータが正しく設定されていることの確認
  Oracle Agentは、各ノードで`LOCAL_LISTENER`パラメータを自動的に設定しますが、正しいことを再確認する必要があります。
- ステップ3: Transport Layer Securityウォレットおよび証明書の作成
  クラスタ用、およびTLS経由でクラスタに接続するクライアント用のTransport Layer Security (TLS)ウォレットおよび証明書を作成する必要があります。
- ステップ4: Oracle RACクラスタの各ノードでのウォレットの作成
  クラスタ・ウォレットの作成後、Oracle Real Applications (Oracle RAC)クラスタの各ノードにコピーできます。
- ステップ5: listener.oraおよびsqlnet.oraファイルでのウォレットの場所の定義
  データベース・サーバーおよびリスナーがウォレットにアクセスできるようにするには、`listener.ora`および`sqlnet.ora`ファイルでウォレットの場所を定義する必要があります。

- ステップ6: データベース・インスタンスおよびリスナーの再起動
  ウォレットが配置され、*.oraファイルが編集された状態で、データベース・サーバーおよびリスナー・プロセスを再起動して、新しい設定を取得する必要があります。
- ステップ7: クラスタ・ノード構成のテスト
  クラスタ・ノード構成をテストするには、ノードの接続記述子を作成し、このノードへの接続を試行します。
- ステップ8: リモート・クライアント構成のテスト
  Oracle Real Applications (Oracle RAC)クラスタ・ノードでウォレットをテストした後、リモート・クライアント構成をテストする準備ができます。

親トピック: Transport Layer Security認証の構成

23.10.1 ステップ1: TCPSプロトコル・エンドポイントの構成

Oracle Real Application Clusters (Oracle RAC)では、クライアントは3つのSCANリスナーのいずれかにアクセスし、それからデータベース・リスナーにルーティングされます。Transport Layer Security (TLS)をサポートするには、これらのリスナーすべてにTCPSプロトコル・エンドポイントが必要です。

1. Oracle RACデータベースをホストするクラスタにログインします。
2. リスナー・リソースをチェックして、TCPエンドポイントをサポートしているかどうかを確認します。
   たとえば:

   ```
   $ srvctl config listener -h
   ```

   次のような出力が表示されます。

   ```
   Name: LISTENER
   Subnet: 192.0.2.195
   Type: type
   Owner: pfitch
   Home: Grid_home
   End points: TCP:1521
   ```

   次のコマンドは、スキャン・リスナーに関する情報を表示します。

   ```
   $ srvctl config scan_listener -h
   ```

   次のような出力が表示されます。

   ```
   SCAN Listener LISTENER_SCAN1 exists. Port: TCP:1529
   Registration invited nodes:
   Registration invited subnets:
   SCAN Listener is enabled.
   SCAN Listener is individually enabled on nodes:
   SCAN Listener is individually disabled on nodes:
   ```

3. データベース・リスナーにTCPSエンドポイントを追加します。
   たとえば:

   ```
   $ srvctl modify listener -endpoints "TCP:port_1/TCPS:port_2"
   ```

4. リスナー構成を確認します。
   たとえば:

   ```
   $ srvctl config listener
   Name: LISTENER
   Network: 1, Owner: oracle
   Home: CRS_home
   ```

```
End points: TCP:port_1/TCPS:port_2
$ lsnrctl status
Listening Endpoints Summary...
(DESCRIPTION=(ADDRESS=(PROTOCOL=ipc)(KEY=LISTENER)))
(DESCRIPTION=(ADDRESS=(PROTOCOL=tcps)(HOST=IP_address)(PORT=port_2)))
(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=IP_address)(PORT=port_1)))
```

5. スキャン・リスナーにTCPSエンドポイントを追加します。

   たとえば:

```
$ srvctl modify scan_listener -endpoints "TCP:port_1/TCPS:port_2"
```

6. SCANリスナーの構成を確認します。

   たとえば:

```
$ srvctl config scan_listener
SCAN Listener LISTENER_SCAN1 exists. Port: TCP:port_1/TCPS:port_2
SCAN Listener LISTENER_SCAN2 exists. Port: TCP:port_1/TCPS:port_2
SCAN Listener LISTENER_SCAN3 exists. Port: TCP:port_1/TCPS:port_2
$ lsnrctl status listener_scan3
Listening Endpoints Summary...
(DESCRIPTION=(ADDRESS=(PROTOCOL=ipc)(KEY=LISTENER_SCAN3)))
(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=IP_address)(PORT=port_1)))
(DESCRIPTION=(ADDRESS=(PROTOCOL=tcps)(HOST=IP_address)(PORT=port_2)))
```

親トピック: Oracle Real Application Clusters環境でのTransport Layer Security接続

23.10.2 ステップ2: 各ノードでLOCAL_LISTENERパラメータが正しく設定されていることの確認

Oracle Agentは、各ノードでLOCAL_LISTENERパラメータを自動的に設定しますが、正しいことを再確認する必要があります。

1. 任意のOracle Real Application Clusters (Oracle RAC)ノードにログインします。
2. SQL*Plusで、SYSDBA管理権限を持つユーザーとして、LOCAL_LISTENERパラメータを確認します。

```
show parameter local_listener;
```

   次のような出力が表示されます。

```
NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------------------------
local_listener                       string      (DESCRIPTION=(ADDRESS_LIST=

(ADDRESS=(PROTOCOL=TCPS)(HOST=IP_address)
                                                 (PORT=port_2))))
```

3. 出力が希望と異なる場合は、各Oracle RACインスタンスを再起動します。

親トピック: Oracle Real Application Clusters環境でのTransport Layer Security接続

23.10.3 ステップ3: Transport Layer Securityウォレットおよび証明書の作成

クラスタ用、およびTLS経由でクラスタに接続するクライアント用のTransport Layer Security (TLS)ウォレットおよび証明書を作成する必要があります。

- 証明書が必要なOracle Real Application Clustersコンポーネント

  Transport Layer Security (TLS)接続を構成する場合、Oracle Real Application Clusters (Oracle

RAC)の特定のコンポーネントに証明書が必要です。

- Transport Layer Securityウォレットおよび証明書の作成
  Transport Layer Securityウォレットおよび証明書を作成するには、最初にルートCA証明書を作成し、次にクラスタ・ウォレットおよびクライアント・ウォレットを作成します。

親トピック: Oracle Real Application Clusters環境でのTransport Layer Security接続

23.10.3.1 証明書が必要なOracle Real Application Clustersコンポーネント

Oracle Real Application Clusters (Oracle RAC)の特定のコンポーネントでは、Transport Layer Security (TLS)接続を構成するときに証明書が必要です。

- 各クラスタ・ノード(サーバー)およびリスナーには、ユーザー証明書およびCA証明書を含むウォレットが必要です。
- 一方向のTLSが構成されている場合、クライアントはリスナーおよびサーバーのCA証明書(ウォレットまたはシステムの証明書ストア内)のみを必要とします。
- mTLSが構成されている場合、クライアントには、ユーザー証明書とリスナーおよびサーバーのCA証明書を含むウォレットが必要です。

親トピック: ステップ3: Transport Layer Securityウォレットおよび証明書の作成

23.10.3.2 ステップ3: Transport Layer Securityウォレットおよび証明書の作成

Transport Layer Securityウォレットおよび証明書を作成するには、最初にルートCA証明書を作成し、次にクラスタ・ウォレットおよびクライアント・ウォレットを作成します。

1. ルートCA証明書を作成します。
   a. 任意のOracle Real Application Clusters (Oracle RAC)クラスタ・ノードにログインします。
   b. `orapki`ユーティリティを使用して、CAのディレクトリにCAウォレットを作成します。

```
$ orapki wallet create -wallet CA_home_wallet_file_directory
```

   c. CAウォレット用の自己署名ルート証明書を作成します。
   たとえば:

```
$ orapki wallet add -wallet CA_home_wallet_file_directory -self_signed -
dn "CN=test CA,O=test,C=c" -keysize 2048 -validity 3650 -sign_alg sha256
Enter wallet password: password
```

   d. ウォレットからルートCA証明書を抽出します。
   このルート証明書はクラスタ・ウォレットまたはクライアント・ウォレットで信頼できるCA証明書として使用され、PKCS#12ウォレットを作成するユーザーに対し配布または公開できます。たとえば:

```
$ orapki wallet export -wallet CA_home_wallet_file_directory -dn "CN=test
CA,O=test,C=c" -cert testCAroot.cer
Enter wallet password: password
```

   この段階で、`CA_home_wallet_file_directory`ディレクトリには新しいウォレット(`ewallet.p12`)と証明書(`testCAroot.cer`)が含まれます。

   構成を確認するには:

```
 $ orapki wallet display -wallet CA_home_wallet_file_directory -summary
```

   次のような出力が表示されます。

```
Requested Certificates:
```

```
User Certificates:
Subject:        CN=test CA,O=test,C=c
Trusted Certificates:
Subject:        CN=test CA,O=test,C=c
```

2. クラスタ・ウォレットを作成します。

   次に、この手順の残りのステップに従って、ユーザー証明書リクエストに署名し、使用している環境の様々なエンティティおよびプロセスに対して承認されたデジタル・ユーザー証明書を提供する準備ができます。テスト環境内で公開キー・インフラストラクチャ機能に関与している各エンティティに対してこの手順を繰り返します。有効なウォレットは、ルートCA証明書および署名付きユーザー証明書で構成されます。

   a. CAホーム・ディレクトリとは異なる場所にウォレットを作成します。

      たとえば:

      ```
      $ orapki wallet create -wallet cluster_wallet_file_directory
      Enter password: password
      Enter password again: password
      ```

   b. ユーザー・アイデンティティ(ユーザーdn)および証明書リクエストを作成します。

      たとえば:

      ```
      $ orapki wallet add -wallet cluster_wallet_file_directory -dn
      "CN=testuser" -keysize 2048
      Enter wallet password: password
      $ orapki wallet export -wallet cluster_wallet_file_directory -dn
      "CN=testuser" -request cluster_wallet_file_directory/testuser.req
      Enter wallet password: password
      ```

      この段階では、cluster_wallet_file_directoryディレクトリにSSOウォレット(cwallet.sso)、ウォレット(ewallet.p12)および証明書リクエスト(testuser.req)が含まれます。証明書リクエストは、前述で生成されたCAによって署名されます。

      たとえば:

      ```
      $ orapki cert create -wallet cluster_wallet_file_directory -request
      cluster_wallet_file_directory/testuser.req -cert
      user_wallet_file_directory/testuser.cer -validity 3650 -sign_alg sha256
      Enter wallet password: password
      ```

      これで、cluster_wallet_file_directoryディレクトリにtestuser.req証明書リクエスト・ファイルが含まれます。

   c. ルート証明書(testCAroot.cer)および署名付きユーザー証明書(testuser.cer)をユーザー・ウォレットにインポートします。

      たとえば:

      ```
      $ orapki wallet add -wallet cluster_wallet_file_directory -trusted_cert -
      cert CA_home_wallet_file_directory/testCAroot.cer -pwd
      Enter wallet password: user_password
      $ orapki wallet add -wallet cluster_wallet_file_directory -user_cert -
      cert cluster_wallet_file_directory/testuser.cer
      Enter wallet password: user_password
      ```

   d. 完成したクラスタ・ウォレットを確認します。

      たとえば:

      ```
      $ orapki wallet display -wallet cluster_wallet_file_directory -summary
      Requested Certificates:
      User Certificates:
      ```

```
Subject:        CN=testuser
Trusted Certificates:
Subject:        CN=test CA,O=test,C=c
```

この時点で、完成したクラスタ・ウォレットをクラスタの各ノードにコピーする準備ができます。

3. クライアント・ウォレットを作成します。
   a. ルート証明書(`testCAroot.cer`)を使用してクライアント・ウォレットを作成します。
      TLS接続を成功させるには、クライアントはサーバーの証明書のCA証明書のみが必要です。
      たとえば:

```
$ orapki wallet create -wallet client_wallet_file_directory -auto_login
$ orapki wallet add -wallet client_wallet_file_directory -trusted_cert -
cert CA_home_wallet_file_directory/testCAroot.cer
```

   b. 完成したクライアント・ウォレットを確認します。
      たとえば:

```
$ orapki wallet display -wallet client_wallet_file_directory -summary
Requested Certificates:
User Certificates:
Trusted Certificates:
Subject:        CN=test CA,O=test,C=c
```

親トピック: ステップ3: Transport Layer Securityウォレットおよび証明書の作成

23.10.4 ステップ4: Oracle RACクラスタの各ノードでのウォレットの作成

クラスタ・ウォレットを作成した後、それをOracle Real Applications (Oracle RAC)クラスタの各ノードにコピーできます。

各ノードが、Oracle Real Application Clusters (Oracle RAC)データベース・サーバー(プロセス・モニター)と、通常はGIホームから実行されるスキャン・リスナーおよびローカル・リスナーの両方からアクセスできることを確認します。

1. 前のセクションで作成したPKCS#12ウォレット(`ewallet.p12`)ファイルをクラスタ内の各ノードにコピーします。
2. 各ノードで、自動ログイン・ウォレット(`cwallet.sso`)を作成します。
   `cwallet.sso`ファイルは、`ewallet.p12`の不明瞭化されたミラー・コピーであり、データベース・サーバーとそのリスナーがアクセスするファイルです。Oracle RACクラスタに`cwallet.sso`を作成する場合は、これを`ewallet.p12`ファイルとともに各ノードのウォレット・ディレクトリにコピーできます。また、`ewallet.p12`ファイルがすでに配置されている場合は、各ノードに`cwallet.sso`ファイルを個別に作成することもできます。`ewallet.p12`ファイルと同じ場所で次のコマンドを実行します:

```
$ orapki wallet create -wallet wallet_file_location -auto_login
Enter wallet password: ewallet_password
```

関連トピック

- 証明書が必要なOracle Real Application Clustersコンポーネント

親トピック: Oracle Real Application Clusters環境でのTransport Layer Security接続

23.10.5 ステップ5: listener.oraおよびsqlnet.oraファイルへのウォレット・ロケーションの定義

データベース・サーバーおよびリスナーがウォレットにアクセスできるようにするには、`listener.ora`および`sqlnet.ora`ファイルにウォレットの場所を定義する必要があります。

1. すべてのノードのGridホームにあるlistener.oraファイルを変更します。

```
SSL_CLIENT_AUTHENTICATION = FALSE
WALLET_LOCATION =
  (SOURCE =
    (METHOD = FILE)
    (METHOD_DATA =
      (DIRECTORY = wallet_file_location)
```

2. 各クラスタ・ノードのOracle DatabaseホームおよびGridホームのsqlnet.oraファイルに、次の情報を追加します。

```
SQLNET.AUTHENTICATION_SERVICES = (BEQ, TCP, TCPS)
SSL_CLIENT_AUTHENTICATION = FALSE
WALLET_LOCATION =
  (SOURCE =
    (METHOD = FILE)
    (METHOD_DATA =
      (DIRECTORY = wallet_file_location)
    )
  )
```

親トピック: Oracle Real Application Clusters環境でのTransport Layer Security接続

23.10.6 ステップ6: データベース・インスタンスおよびリスナーの再起動

ウォレットを配置して*.oraファイルを編集した後、データベース・サーバーおよびリスナー・プロセスを再起動して新しい設定を取得する必要があります。

再起動プロセスにより、前にLOCAL_LISTENERパラメータを設定したOracle Real Application Clusters (Oracle RAC)インスタンスも有効になります。

- 任意のクラスタ・ノードで、srvctlユーティリティを使用して、データベース・サーバーおよびリスナー・プロセスを再起動します。

  たとえば:

```
$ srvctl stop listener
$ srvctl start listener
$ srvctl stop scan_listener
$ srvctl start scan_listener
$ srvctl stop database -d db_name
$ srvctl start database -d db_name
```

親トピック: Oracle Real Application Clusters環境でのTransport Layer Security接続

23.10.7 ステップ7: クラスタ・ノード構成のテスト

クラスタ・ノード構成をテストするには、ノードの接続記述子を作成し、このノードへの接続を試行します。

1. 任意のクラスタ・ノードで、スキャン・リスナーTCPSエンドポイントを使用する接続記述子をtnsnames.oraファイルに作成します。

   たとえば、dbsslというTCPSエンドポイントの場合:

```
DBSSL =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCPS)(HOST = scan_name)(PORT = port_2))
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = service_name)
    )
  )
```

2. SQL*Plusを使用して、このTCPSエンドポイントへの接続を試みます。
   たとえば:

```
sqlplus user_name/@dbssl
Enter password: password
```

親トピック: Oracle Real Application Clusters環境でのTransport Layer Security接続

23.10.8 ステップ8: リモート・クライアント構成のテスト

Oracle Real Applications (Oracle RAC)クラスタ・ノードでウォレットをテストした後、リモート・クライアント構成をテストする準備ができます。

1. クラスタ・ノード上のすべてのリモート・クライアント`sqlnet.ora`ファイルで、ウォレット・ディレクトリを定義します。

```
WALLET_LOCATION =
  (SOURCE =
    (METHOD = FILE)
    (METHOD_DATA =
      (DIRECTORY = wallet_file_location)
    )
  )
```

2. 前にSSLウォレットおよび証明書を作成したときに作成したクライアント・ウォレットを、クライアント・ウォレット・ディレクトリに移動します。

```
$ wallet create -wallet wallet_file_location -auto_login
Enter wallet password: password
```

   `wallet_file_location`には、`ewallet.p12`ファイルと`cwallet.sso`ファイルが必要です。

3. `tnsnames.ora`ファイルに、スキャン・リスナーのTCPSエンドポイントを使用する接続記述子を作成します。
   たとえば:

```
DBSSL =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCPS)(HOST = scan_name)(PORT = port_2))
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = service_name)
    )
  )
```

4. SQL*Plusを使用して、このTCPSエンドポイントへの接続を試みます。
   たとえば:

```
sqlplus user_name/@dbssl
Enter password: password
```

親トピック: Oracle Real Application Clusters環境でのTransport Layer Security接続

23.11 Microsoft証明書ストアを使用したクライアント認証および暗号化のためのトランスポート・レイヤー・セキュリティの構成

この構成をMicrosoft Certificate Store (MCS)で実行するには、`orapki`コマンドライン・ツールを使用して証明書を生成し、Oracleウォレットを操作します。

- Microsoft証明書ストアを使用したクライアント認証および暗号化のためのトランスポート・レイヤー・セキュリティの構成について
  このタイプの構成は、Common Access CardおよびPIVカード認証の基盤です。
- ステップ1: サーバー・ウォレットの作成および構成
  orapkiを使用して、サーバー・ウォレットおよびサーバーの自己署名証明書を作成する必要があります。
- ステップ2: クライアント・ウォレットの作成および構成
  orapkiを使用して、クライアント・ウォレットおよび証明書リクエストを作成する必要があります。
- ステップ3: Oracle Databaseでの外部ユーザーの作成
  クライアントおよびサーバー接続で使用する外部ユーザーを作成する必要があります。
- ステップ4: サーバーのlistener.oraファイルの構成
  次に、サーバーのlistener.oraファイルを確認してから再起動する必要があります。
- ステップ5: サーバーのsqlnet.oraファイルの構成
  sqlnet.oraファイルが、前に作成したサーバー・ウォレットを指していることを確認する必要があります。
- ステップ6: Microsoft証明書ストアへのクライアント・ウォレットのインポート
  このインポート操作を実行するには、Microsoft管理コンソール(MMC)を使用する必要があります。
- ステップ7: クライアントのsqlnet.oraファイルの構成
  クライアント・ウォレットにMicrosoft証明書ストアを使用するようにクライアントのsqlnet.oraファイルを構成する必要があります。
- ステップ8: Oracle Databaseの構成
  Oracleデータベースで、OS_AUTHENT_PREおよびREMOTE_OS_AUTHパラメータを構成します。
- ステップ9: クライアントおよびサーバー接続のテスト
  Microsoft証明書ストア構成を完了したら、サーバー接続をテストする必要があります。

親トピック: Transport Layer Security認証の構成

23.11.1 Microsoft証明書ストアを使用したクライアント認証および暗号化のためのトランスポート・レイヤー・セキュリティの構成について

このタイプの構成は、Common Access CardおよびPIVカード認証の基盤です。

Common Access CardおよびPIVカードとともに提供されるソフトウェア・ライブラリが、必要な証明書を透過的にMicrosoft証明書ストアにロードできるかぎり、構成するTransport Layer Security (TLS)認証は透過的に実行されます。

PIVカードにロードされるユーザー証明書のすべての署名証明書は、サーバー・レベルのTLS構成の一部としてサーバーのウォレットに手動でロードする必要があることに注意してください。

親トピック: Microsoft証明書ストアを使用したクライアント認証および暗号化のためのトランスポート・レイヤー・セキュリティの構成

23.11.2 ステップ1: サーバー・ウォレットの作成および構成

サーバー・ウォレットおよびサーバーの自己署名証明書を作成するには、orapkiを使用する必要があります。

1. Oracle Databaseサーバーにログインします。
2. サーバー・ウォレットのディレクトリを作成します。
   たとえば:

```
mkdir /home/oracle/wallet_tls/server
```

3. このディレクトリに移動します。

```
cd /home/oracle/wallet_tls/server
```

4. サーバー・ウォレットを作成します。

```
orapki wallet create -wallet . -auto_login -pwd password
```

5. ディレクトリを確認します。

   たとえば:

```
ls -la
```

   次のような出力が表示されます。

```
total 16
drwxr-xr-x. 2 oracle oinstall 4096 Oct 28 07:18 .
drwxr-xr-x. 6 oracle oinstall 4096 Oct 28 07:17 ..
-rw-------. 1 oracle oinstall 120 Oct 28 07:18 cwallet.sso
-rw-rw-rw-. 1 oracle oinstall 0 Oct 28 07:18 cwallet.sso.lck
-rw-------. 1 oracle oinstall 75 Oct 28 07:18 ewallet.p12
-rw-rw-rw-. 1 oracle oinstall 0 Oct 28 07:18 ewallet.p12.lck
```

6. サーバーの自己署名証明書を作成します。

```
orapki wallet add -wallet . -dn "cn=server" -self_signed -keysize 2048 -
sign_alg sha256 -validity 365 -pwd password
```

親トピック: Microsoft証明書ストアを使用したクライアント認証および暗号化のためのトランスポート・レイヤー・セキュリティの構成

23.11.3 ステップ2: クライアント・ウォレットの作成および構成

クライアント・ウォレットおよび証明書リクエストを作成するには、orapkiを使用する必要があります。

1. Oracle Databaseクライアントにログインします。
2. クライアント・ウォレットのディレクトリを作成します。

   たとえば:

```
mkdir /home/oracle/wallet_tls/client
```

3. このディレクトリに移動します。

```
cd /home/oracle/wallet_tls/client
```

4. クライアント・ウォレットを作成します。

```
orapki wallet create -wallet . -auto_login -pwd password
```

5. ユーザー証明書のリクエストを作成し、リクエストをエクスポートします。

```
orapki wallet add -wallet . -dn "cn=client" -keysize 2048 -sign_alg sha256 -pwd
password
orapki wallet export -wallet . -dn "cn=client" -request req.txt -pwd password
```

6. クライアント・ディレクトリからサーバー・ディレクトリに証明書リクエストをコピーします。

   たとえば:

```
cp req.txt ../server/
cd ../server/
```

7. クライアントの証明書に署名し、サーバーのCA証明書もエクスポートします。

   たとえば:

```
orapki cert create -wallet . -request req.txt -cert sign.txt -validity 1000 -
pwd password
orapki wallet export -wallet . -dn "cn=server" -cert server.txt
cp server.txt ../client
cp sign.txt ../client
orapki wallet add -wallet . -trusted_cert -cert server.txt -pwd password
orapki wallet add -wallet . -user_cert -cert sign.txt -pwd password
cp sign.txt server.txt ../client/
cd ../client
```

親トピック: Microsoft証明書ストアを使用したクライアント認証および暗号化のためのトランスポート・レイヤー・セキュリティの構成

23.11.4 ステップ3: Oracle Databaseでの外部ユーザーの作成

クライアントおよびサーバー接続で使用する外部ユーザーを作成する必要があります。

1. ユーザーを作成して権限を付与できるユーザーとして、この外部ユーザー・アカウントを使用するPDBにログインします。
2. 外部ユーザーを作成します。

   たとえば:

```
CREATE USER tlsuser IDENTIFIED EXTERNALLY AS 'cn=client';
```

3. このアカウントにCONNECT権限を付与します。

```
GRANT CONNECT TO tlsuser;
```

親トピック: Microsoft証明書ストアを使用したクライアント認証および暗号化のためのトランスポート・レイヤー・セキュリティの構成

23.11.5 ステップ4: サーバーのlistener.oraファイルの構成

次に、サーバーの`listener.ora`ファイルを確認してから再起動する必要があります。

1. Oracle Databaseサーバーにログインします。
2. サーバーの`listener.ora`ファイルをチェックして、正しく構成されていることを確認します。

   たとえば:

```
 cat /u01/app/oracle/product/release/dbhome_1/network/admin/listener.or
```

   次のような出力が表示されます。

```
LISTENERBOS =
     (DESCRIPTION_LIST =
       (DESCRIPTION =
         (ADDRESS = (PROTOCOL = TCP)(HOST = domain.com)(PORT = 1529))
        )
       (DESCRIPTION =
         (ADDRESS = (PROTOCOL = TCPS)(HOST = domain.com)(PORT = 1530))
         )
        )
WALLET_LOCATION =
      (SOURCE =
        (METHOD = File)
          (METHOD_DATA =
           (DIRECTORY = /home/oracle/wallet_tls/server))
```

3. リスナーを再起動し、データベースがこのリスナーに登録されているかどうかを確認します。

```
su - oracle
./lsnrctl start
```

次のような出力が表示されます。

```
Listener Parameter File
/u01/app/oracle/product/release/dbhome_1/network/admin/listener.ora
Listener Log File /u01/app/oracle/diag/tnslsnr/service/instance/alert/log.xml
Listening Endpoints Summary...
(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=domain.com)(PORT=1523)))
(DESCRIPTION=(ADDRESS=(PROTOCOL=tcps)(HOST=domain.com)(PORT=1525)))
  Services Summary...
   Service "service" has 1 instance(s).
    Instance "instance", status READY, has 1 handler(s) for this service...
   Service "serviceDB" has 1 instance(s).
    Instance "instance", status READY, has 1 handler(s) for this service...
 The command completed successful
```

親トピック: Microsoft証明書ストアを使用したクライアント認証および暗号化のためのトランスポート・レイヤー・セキュリティの構成

23.11.6 ステップ5: サーバーのsqlnet.oraファイルの構成

sqlnet.oraファイルが、前に作成したサーバー・ウォレットを指していることを確認する必要があります。

1. Oracle Databaseサーバーにログインします。
2. sqlnet.oraファイルを確認して、サーバー・ウォレットを指していることを確認します。

   たとえば:

```
NAMES.DIRECTORY_PATH=(TNSNAMES)
SQLNET.AUTHENTICATION_SERVICES=(BEQ,TCPS)
SSL_CLIENT_AUTHENTICATION = TRUE
WALLET_LOCATION =
 (SOURCE =
  (METHOD = FILE)
    (METHOD_DATA =
      (DIRECTORY = /home/oracle/wallet_tls/server)
    )
)
```

親トピック: Microsoft証明書ストアを使用したクライアント認証および暗号化のためのトランスポート・レイヤー・セキュリティの構成

23.11.7 ステップ6: Microsoft証明書ストアへのクライアント・ウォレットのインポート

このインポート操作を実行するには、Microsoft管理コンソール(MMC)を使用する必要があります。

1. MMC (mmc.exe)を起動します。
2. 「ファイル」、「スナップインの追加と削除」の順に選択します。
3. 「証明書」、「追加」の順に選択します。
4. 「ユーザー アカウント」、「完了」、「OK」の順に選択します。
5. 「コンソール ルート」、「証明書 - 現在のユーザー」、「個人」、「証明書」の順に移動します。
6. 「すべてのタスク」を右クリックし、「インポート」、「次へ」、「参照」の順に選択します。
7. 接続に必要な証明書が含まれている証明書ファイル(ewallet.p12など)を選択します。
8. 「開く」、「次へ」の順に選択します。

9. ウォレットのパスワードを入力します。
10. 「このキーをエクスポート可能としてマークする」チェック・ボックスを選択します。
11. 「すべての拡張プロパティを含める」チェック・ボックスを選択します。
12. 「すべての証明書を次のストアに配置する: 個人」を選択します。
13. 「次へ」、「完了」の順に選択します。
14. 「コンソール ルート」に移動してから、「証明書 - 現在のユーザー」、「個人」、「証明書」の順に選択して、クライアントの証明書がMYストアに追加されていることを確認します。
15. 「コンソール ルート」に移動してから、「証明書 - 現在のユーザー」、「信頼されたルート証明機関」、「証明書」の順に選択して、CA証明書がROOTストアに追加されていることを確認します。

親トピック: Microsoft証明書ストアを使用したクライアント認証および暗号化のためのトランスポート・レイヤー・セキュリティの構成

23.11.8 ステップ7: クライアントのsqlnet.oraファイルの構成

クライアント・ウォレットにMicrosoft証明書ストアを使用するようにクライアントの`sqlnet.ora`ファイルを構成する必要があります。

1. Oracle Databaseクライアントにログインします。
2. クライアント側の`sqlnet.ora`ファイルを確認します。
   たとえば:

```
WALLET_LOCATION = (SOURCE = (METHOD=MCS))
```

親トピック: Microsoft証明書ストアを使用したクライアント認証および暗号化のためのトランスポート・レイヤー・セキュリティの構成

23.11.9 ステップ8: Oracle Databaseの構成

Oracleデータベースで、OS_AUTHENT_PREおよびREMOTE_OS_AUTHパラメータを構成します。

1. ALTER SYSTEMシステム権限を持つユーザーとして、Oracle DatabaseサーバーでSQL*Plusにログインします。
2. OS_AUTHENT_PREおよびREMOTE_OS_AUTHパラメータを設定します。

```
ALTER SYSTEM SET REMOTE_OS_AUTHENT=FALSE SCOPE=SPFILE;
ALTER SYSTEM SET OS_AUTHENT_PREFIX='' SCOPE=SPFILE;
```

3. データベース・インスタンスを再起動します。

親トピック: Microsoft証明書ストアを使用したクライアント認証および暗号化のためのトランスポート・レイヤー・セキュリティの構成

23.11.10 ステップ9: クライアントとサーバーの接続のテスト

Microsoft証明書ストアの構成が完了したら、およびサーバー接続をテストする必要があります。

1. TLS接続にMCSが使用されていることを確認するには、クライアントの`sqlnet.ora`ファイルに次の行を追加することでクライアント・トレースを有効にします。

```
trace_level_client=16
trace_directory_client=trace_directory
DIAG_ADR_ENABLED=OFF
```

2. SQL*Plusを使用してサーバーに接続してから、証明書がMCSから正常にロードされていることを確認します。

```
nztwOpenWallet: [enter]
nztwOpenWallet: WRL mcs:, type = 24
nztwOpenWallet: Loading the EXTKS provider for MCS type wallet
nztwOpenWallet: [exit] OK
```

親トピック: Microsoft証明書ストアを使用したクライアント認証および暗号化のためのトランスポート・レイヤー・セキュリティの構成

23.12 Transport Layer Security構成のトラブルシューティング

Oracle Database SSLアダプタの使用中に一般的なエラーが発生する場合があります。

Oracle Netトレースを有効にして、エラーの原因を特定することが必要になる場合があります。トレース・パラメータを設定してOracle Netトレースを有効にする方法の詳細は、『Oracle Database Net Services管理者ガイド』を参照してください。

ORA-28759: ファイルのオープンに失敗しました

原因: 指定されたファイルをオープンできませんでした。通常、このエラーはウォレットが見つからないために発生します。

処置: 次の点を確認します。

- `sqlnet.ora` ファイルに正しいウォレット・ロケーションが指定されていることを確認します。これは、ウォレットを保存したディレクトリと同じにする必要があります。
- Oracle Net トレースを有効にして、開くことができないファイルの名前とその原因を調べます。
- ウォレットを保存したときに自動ログインが有効になっていたことを確認します。『Oracle Database エンタープライズ・ユーザー・セキュリティ管理者ガイド』を参照してください。

ORA-28786: 暗号化された秘密キーの復号化に失敗しました

原因: 暗号化された秘密キーの復号化に不正なパスワードが使用されました。このことは、多くの場合、自動ログイン・ウォレットが使用されていないために発生します。

処置: Oracle Wallet Manager を使用して、ウォレットの自動ログイン機能をオンにします。次に、ウォレットを再度保存します。『Oracle Database エンタープライズ・ユーザー・セキュリティ管理者ガイド』を参照してください。

自動ログイン機能が使用されていない場合は、正しいパスワードを入力します。

ORA-28858: SSL プロトコル・エラーが発生しました

原因: これは、2 つのプロセス間の TLS ハンドシェイク・ネゴシエーション中に発生する可能性がある一般的なエラーです。

処置: Oracle Net トレースを使用可能にし、再接続してトレース出力を生成します。トレース出力を Oracle カスタマ・サポートに連絡してください。

ORA-28859 SSL でネゴシエーションに失敗しました

原因: TLS プロトコルの一部である 2 つのプロセス間のネゴシエーション中にエラーが発生しました。このエラーは、両プロセスの接続で同じ暗号スイートがサポートされていない場合に発生する可能性があります。

処置: 次の点を確認します。

- Oracle Net Manager を使用して、クライアントとサーバーの両方の TLS バージョンが一致しているか、互換性があることを確認します。たとえば、サーバーで TLS 3.0 のみを受け入れ、クライアントで TLS 1.1 のみを受け入れる場合、TLS 接続は失敗します。
- Oracle Net Manager を使用して、クライアントとサーバーの両方で構成されている暗号スイートをチェックし、両方に互換性のある暗号スイートが設定されていることを確認します。

  エラーが繰り返される場合は、Oracle Net トレースを有効にし、再接続してください。トレース出力を Oracle カスタマ・サポートに連絡してください。

関連項目:

クライアントとサーバーで互換性のある暗号スイートを設定する方法の詳細は、ステップ 2D: クライアントの Transport Layer Security 暗号スイートの設定(オプション)を参照してください

ノート:

暗号スイートを構成しない場合は、使用可能なすべての暗号スイートが有効になります。

ORA-28862: SSL 接続に失敗しました。

原因: このエラーは、ピアが接続をクローズしたために発生しました。

処置: 次の点を確認します。

- ウォレットが検出されるように、`sqlnet.ora` ファイルに正しいウォレット・ロケーションが指定されていることを確認します。
- Oracle Net Manager を使用して、`sqlnet.ora` ファイルに暗号スイートが正しく設定されて

いることを確認します。このエラーは、sqlnet.ora が手動で編集され、暗号スイート名の綴りが誤っている場合に発生することがあります。暗号スイート名で大/小文字が区別される文字列照合が使用されていることを確認します。

- Oracle Net Manager を使用して、クライアントとサーバーの両方の TLS バージョンが一致しているか、互換性があることを確認します。このエラーは、サーバーとクライアントに指定されている TLS バージョンが一致しないために発生することがあります。たとえば、サーバーで TLS 3.0 のみを受け入れ、クライアントで TLS 1.0 のみを受け入れる場合、TLS 接続は失敗します。

- 診断情報の詳細を確認するために、ピアで Oracle Net トレースを有効にします。

ORA-28865: SSL 接続はクローズしました。

原因: 基礎となるトランスポート層でエラーが発生したか、ピア・プロセスが突然停止したため、TLS 接続がクローズしました。

処置: 次の点を確認します。

- Oracle Net Manager を使用して、クライアントとサーバーの両方の TLS バージョンが一致しているか、互換性があることを確認します。このエラーは、サーバーとクライアントに指定されている TLS バージョンが一致しないために発生することがあります。たとえば、サーバーで TLS 3.0 のみを受け入れ、クライアントで TLS 1.0 のみを受け入れる場合、TLS 接続は失敗します。

- Diffie-Hellman 匿名暗号スイートを使用し、サーバーの listener.ora ファイルで SSL_CLIENT_AUTHENTICATION パラメータが true に設定されている場合、クライアントは証明書をサーバーに渡しません。サーバーがクライアントの証明書を受信しない場合、クライアントを認証できないため接続がクローズされます。これを解決するには、別の暗号スイートを使用するか、この listener.ora パラメータを false に設定します。

- Oracle Net トレースを有効にし、トレース出力にネットワーク・エラーがないか確認します。

- 詳細は、「ORA-28862: SSL 接続に失敗しました。」の「処置」を参照してください。

ORA-28868: ピア証明連鎖のチェックに失敗しました。

原因: ピアが証明連鎖を提示したときに、その証明連鎖のチェックに失敗しました。この失敗の原因として、いくつかの問題が考えられます。

- 連鎖内のいずれかの証明書が期限切れになっています。

- 連鎖内のいずれかの証明書の認証局がトラスト・ポイントとして認識されていません。

- いずれかの証明書の署名を検証できません。

処置: Oracle Wallet Manager を使用してウォレットを開き、次のことを確認するには、『Oracle Database エンタープライズ・ユーザー・セキュリティ管理者ガイド』を参照してください。

- ウォレットにインストールされているすべての証明書が使用可能である(期限切れでない)こと。

- ピア証明連鎖からの認証局の証明書が、ウォレットに信頼できる証明書として追加されています。Oracle Wallet Manager を使用して、信頼できる証明書をインポートするには、『Oracle Database エンタープライズ・ユーザー・セキュリティ管理者ガイド』を参照してください。

ORA-28885: 必須のキー使用方法のある証明書が見つかりません。

原因: 証明書は、適切な X.509 バージョン 3 のキー使用方法の拡張機能を使用して作成されていません。

処置: Oracle Wallet Manager を使用して、証明書のキー使用方法をチェックします。キー使用方法の値の詳細は、『Oracle Database エンタープライズ・ユーザー・セキュリティ管理者ガイド』を参照してください。

ORA-29024: 証明書の検証に失敗しました

原因: 反対側から送信された証明書の妥当性チェックに失敗しました。このエラーは、証明書が期限切れか、失効済か、または他の理由で無効になっている場合に発生する可能性があります。

処置: 次の点を確認します。

- 証明書をチェックして、有効かどうかを確認します。必要に応じて、新規の証明書を取得するか、送信者に証明書にエラーがあることを警告するか、または再送します。

- サーバーのウォレットに、クライアントの証明書を検証するための適切なトラスト・ポイントがあることを確認します。トラスト・ポイントがない場合は、Oracle Wallet Manager を使用して、適切なトラスト・ポイントをウォレットにインポートします。信頼できる証明書のインポートの詳細は、『Oracle Database エンタープライズ・ユーザー・セキュリティ管理者ガイド』を参照してください。

- 証明書が失効していないこと、および証明書失効リスト(CRL)チェックがオンになっていることを確認します。詳細は、証明書失効リストによる証明書検証の構成を参照してください

ORA-29223: 証明連鎖を作成できませんでした

原因: インストールする証明書の既存のトラスト・ポイントを使用して証明書チェーンを作成できません。通常、このエラーは、ピアによって完全な連鎖が提供されず、連鎖を完了するのに適切なトラス

ト・ポイントがない場合に戻されます。

処置: Oracle Wallet Manager を使用して、連鎖の完了に必要なトラスト・ポイントをインストールします。信頼できる証明書のインポートの詳細は、『Oracle Database エンタープライズ・ユーザー・セキュリティ管理者ガイド』を参照してください。

親トピック: Transport Layer Security認証の構成

23.13 証明書失効リストによる証明書の検証

オラクル社では、証明書失効リストを使用して証明書を検証できるツールを提供しています。

- 証明書失効リストによる証明書検証について
  指定の証明書を指定のコンテキストで使用できるかどうかを判定するプロセスは、証明書の検証と呼ばれます。
- 使用するCRLの選択方法
  使用するトラスト・ポイントすべてについてCRLが必要です。
- CRLチェックの動作の仕組み
  Oracle Databaseでは証明書の失効状態をCRLに照らして確認します。
- 証明書失効リストによる証明書検証の構成
  `sqlnet.ora`ファイルを編集して、証明書失効リストによる証明書検証を構成できます。
- 証明書失効リストの管理
  証明書失効リストの管理では、証明書失効チェックを有効にする前に、CRLが正しい書式であることを確認する必要があります。
- CRL証明書検証のトラブルシューティング
  CRLに対して証明書を検証するかどうかを判断するには、Oracle Netトレースを有効にできます。
- 証明書検証に関連するOracle Netトレース・ファイルのエラー・メッセージ
  証明書検証に関連するトレース・メッセージが生成されます。

親トピック: Transport Layer Security認証の構成

23.13.1 証明書失効リストによる証明書検証について

指定の証明書を指定のコンテキストで使用できるかどうかを判定するプロセスは、証明書の検証と呼ばれます。

証明書の検証には、次が該当するかどうかの判定が含まれます。

- 信頼できる認証局(CA)にデジタル署名された証明書があるかどうか。
- 証明書のデジタル署名が、証明書自体の別個に計算されたハッシュ値および証明書の署名者の(CAの)公開キーに対応しているかどうか。
- 証明書が期限切れになっていないかどうか。
- 証明書が失効していないかどうか。

Transport Layer Securityネットワーク・レイヤーは、自動的に最初の3つの検証チェックを実行しますが、証明書が失効していないことを確認するには、証明書失効リスト(CRL)のチェックを構成する必要があります。CRLは、失効した証明書のリストを含む署名付きデータ構造体です。これは通常、元の証明書を発行したエンティティと同じエンティティによって発行され署名されます。

親トピック: 証明書失効リストによる証明書の検証

23.13.2 使用するCRLの選択方法

使用するトラスト・ポイントすべてについてCRLが必要です。

トラスト・ポイントは、一定の信頼レベルで資格を与えられたサード・パーティ・アイデンティティからの信頼できる証明書です。

通常、信頼する認証局はトラスト・ポイントと呼ばれます。

親トピック: 証明書失効リストによる証明書の検証

23.13.3 CRLチェックの動作の仕組み

Oracle Databaseでは証明書の失効状態をCRLに照らして確認します。

CRLはファイル・システム・ディレクトリ(Oracle Internet Directory)に存在するか、または証明書のCRL配布ポイント(CRL DP)拡張で指定された場所からダウンロードして入手します。

通常、CRL定義は数日間有効です。CRLをローカル・ファイル・システムまたはディレクトリに格納する場合、CRLを定期的に更新する必要があります。 CRL配布ポイント(CRL DP)を使用する場合、証明書が使用されるたびにCRLがダウンロードされるため、CRLを定期的に更新する必要はありません。

サーバーは、次の場所で(ここに示されている順番で)CRLを検索します。システムは、証明書のCAのDNと一致するCRLを検出すると、検索を停止します。

1. ローカル・ファイル・システム

   システムは、まず`sqlnet.ora`ファイルの`SSL_CRL_FILE`パラメータを確認し、それから`SSL_CRL_PATH`パラメータを確認します。システムは、これらの2つのパラメータが指定されていない場合、ウォレット・ロケーションで任意のCRLをチェックします。

   ノート: CRLをローカル・ファイル・システムに格納する場合、`orapki`ユーティリティを使用してCRLを定期的に更新する必要があります(証明書検証用ハッシュ値によるCRLの名前変更など)。

2. Oracle Internet Directory

   サーバーは、ローカル・ファイル・システムでCRLを検出できず、`ldap.ora`ファイルでディレクトリ接続情報が構成されている場合、このディレクトリを検索します。これは、CAの識別名(DN)およびCRLサブツリーのDNを使用してCRLサブツリーを検索します。

   ディレクトリでCRLを検索するためには、サーバーに、適切に構成された`ldap.ora`ファイルが存在している必要があります。Oracle Internet Directoryのドメイン・ネーム・システム(DNS)検出機能は使用できません。また、CRLをディレクトリに格納する場合、`orapki`ユーティリティを使用してCRLを定期的に更新する必要があることに注意してください。

3. CRL DP

   CAが、証明書が発行されたときのCRL DP X.509、バージョン3、証明書拡張子内の位置を指定すると、その証明書用の失効情報を含む適切なCRLがダウンロードされます。現在、Oracle Databaseは、LDAPを介したCRLのダウンロードをサポートしています。

   次のことに注意してください。

   - パフォーマンス上の理由から、ユーザー証明書のみがチェックされます。

- CRLをローカル・ファイル・システムではなくディレクトリに格納することをお薦めします。

関連トピック

- Oracle Internet DirectoryへのCRLのアップロード
- 証明書検証用ハッシュ値によるCRLの名前変更

親トピック: 証明書失効リストによる証明書の検証

23.13.4 証明書失効リストによる証明書検証の構成

sqlnet.oraファイルを編集して、証明書失効リストによる証明書検証を構成できます。

- 証明書失効リストによる証明書検証の構成について
  証明書失効ステータス・チェックを有効にするには、sqlnet.oraファイルでSSL_CERT_REVOCATIONパラメータをREQUIREDまたはREQUESTEDに設定する必要があります。
- クライアントまたはサーバー用の証明書失効ステータス・チェックの有効化
  クライアントまたはサーバー用の証明書失効ステータス・チェックを有効にできます。
- 証明書失効ステータス・チェックの無効化
  証明書失効ステータス・チェックを無効にできます。

親トピック: 証明書失効リストによる証明書の検証

23.13.4.1 証明書失効リストによる証明書検証の構成について

証明書失効ステータス・チェックを有効にするには、sqlnet.oraファイルでSSL_CERT_REVOCATIONパラメータをREQUIREDまたはREQUESTEDに設定する必要があります。

証明書失効ステータス・チェックを有効にするには、sqlnet.oraファイルでSSL_CERT_REVOCATIONパラメータをREQUIREDまたはREQUESTEDに設定する必要があります。

デフォルトでは、このパラメータはNONEに設定され、証明書失効ステータス・チェックはオフになっています。

ノート:

CRL をローカル・ファイル・システムまたは Oracle Internet Directory に格納する場合は、コマンドライン・ユーティリティ orapki を使用して、ファイル・システム内の CRL の名前を変更するか、CRL をディレクトリにアップロードします。

関連トピック

- 証明書失効リストの管理

親トピック: 証明書失効リストによる証明書検証の構成

23.13.4.2 クライアントまたはサーバー用の証明書失効ステータス・チェックの有効化

クライアントまたはサーバー用の証明書失効ステータス・チェックを有効にできます。

1. Oracle Net Managerを起動します。
   - (UNIX) $ORACLE_HOME/binから、コマンドラインで次のコマンドを入力します。

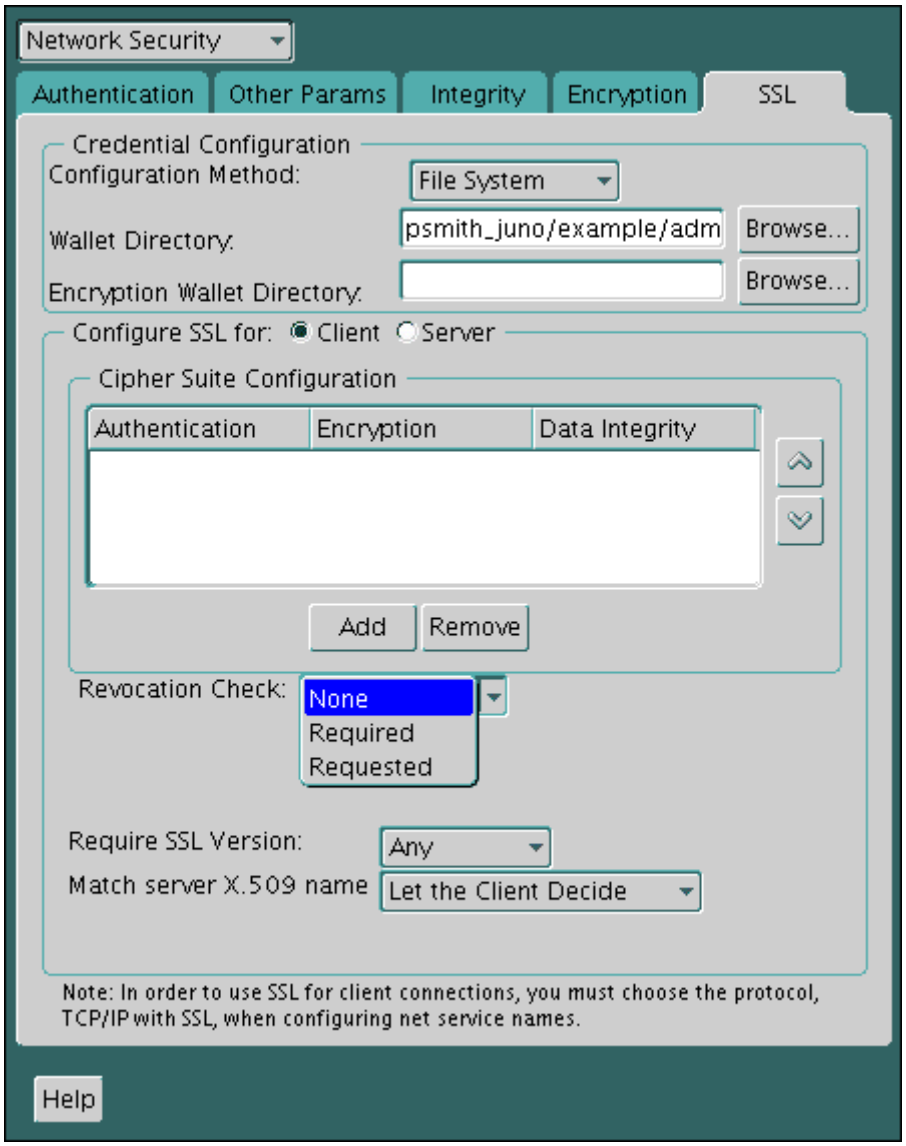
     ```
     netmgr
     ```

- (Windows)「スタート」→「プログラム」→「Oracle - HOME_NAME」→「Configuration and Migration Tools」→「Net Manager」を選択します。

2. 「Oracle Netの構成」を展開し、「ローカル」から「プロファイル」を選択します。
3. 「ネーミング」リストから、「ネットワーク・セキュリティ」を選択します。

   ネットワーク・セキュリティのタブ付きウィンドウが表示されます。

4. 「SSL」タブを選択します。
5. 「失効チェック」リストから次のいずれかのオプションを選択します。



- 必須: 証明書失効ステータス・チェックが必須です。証明書が失効しているかCRLが見つからない場合、TLS接続は拒否されます。証明書がまだ失効していないことが確認された場合にのみ、TLS接続は許可されます。
- リクエスト済: CRLが使用可能な場合に、証明書失効ステータス・チェックを実行します。証明書が失効している場合、TLS接続は拒否されます。CRLが見つからない場合や、証明書がまだ失効していない場合、TLS接続は許可されます。

  パフォーマンス上の理由から、ユーザー証明書の失効のみがチェックされます。

6. (オプション)CRLがローカル・ファイル・システムに格納されている場合は、次のフィールドのいずれか、または両方を設定して、格納場所を指定します。これらのフィールドは、「失効チェック」が「必須」または「リクエスト済」に設定されている場合にのみ使用できます。

- 証明書失効リスト・パス: CRLが格納されているディレクトリへのパスを入力するか、または「参照」をクリックし、ファイル・システムを検索してディレクトリを探します。このパスを指定すると、sqlnet.oraファイルのSSL_CRL_PATHパラメータが設定されます。このパラメータでパスを指定しない場合、デフォルトはウォレット・ディレクトリです。DERエンコードされた(バイナリ形式) CRLおよびPEMエンコードされた(BASE64) CRLの両方がサポートされています。
- 証明書失効リスト・ファイル: 包括的CRLファイル(PEMエンコードされた(BASE64) CRLが優先度の順に1つのファイルに連結されたもの)へのパスを入力するか、または「参照」をクリックし、ファイル・システムを検索してディレクトリを探します。このファイルを指定すると、sqlnet.oraファイルのSSL_CRL_FILEパラメータが設定されます。このパラメータを設定した場合、ファイルが指定された場所に存在する必要があります。存在しない場合、アプリケーションは起動中にエラーになります。

  「証明書失効リスト・パス」を設定してCRLをローカル・ファイル・システム・ディレクトリに格納する場合は、orapkiユーティリティを使用して、システムが特定できるようにCRLの名前を変更する要があります。

7. (オプション)CRLをOracle Internet Directoryからフェッチする場合は、ディレクトリ・サーバーとポート情報をldap.oraファイルで指定する必要があります。

   ldap.oraファイルを構成する場合は、ディレクトリに非TLSポートのみを指定する必要があります。CRLダウンロードはTLSプロトコルの一部として実行され、TLS接続内でのTLS接続の確立はサポートされません。

   Oracle Internet Directoryの非TLSポートが無効になっている場合、Oracle DatabaseのCRL機能は動作しません。

8. 「ファイル」→「ネットワーク構成の保存」を選択します。sqlnet.oraファイルが更新されます。

関連トピック

- 証明書検証用ハッシュ値によるCRLの名前変更

親トピック: 証明書失効リストによる証明書検証の構成

23.13.4.3 証明書失効ステータス・チェックの無効化

証明書失効ステータス・チェックを無効にできます。

1. Oracle Net Managerを起動します。
   - (UNIX) $ORACLE_HOME/binから、コマンドラインで次のコマンドを入力します。

     ```
     netmgr
     ```

   - (Windows)「スタート」→「プログラム」→「Oracle - HOME_NAME」→「Configuration and Migration Tools」→「Net Manager」を選択します。
2. 「Oracle Netの構成」を展開し、「ローカル」から「プロファイル」を選択します。
3. 「ネーミング」リストから、「ネットワーク・セキュリティ」を選択します。

   ネットワーク・セキュリティのタブ付きウィンドウが表示されます。

4. 「SSL」タブを選択します。
5. 「失効チェック」リストから「なし」を選択します。
6. 「ファイル」メニューから、「ネットワーク構成の保存」を選択します。

   sqlnet.oraファイルが次のエントリで更新されます。

   ```
   SSL_CERT_REVOCATION=NONE
   ```

関連トピック

- CRL証明書検証のトラブルシューティング

親トピック: 証明書失効リストによる証明書検証の構成

23.13.5 証明書失効リストの管理

証明書失効リストの管理では、証明書失効チェックを有効にする前に、CRLが正しい書式であることを確認する必要があります。

- 証明書失効リストの管理について
  Oracle Databaseには、証明書の管理を実行するために使用できるコマンドライン・ユーティリティ`orapki`があります。
- CRLを管理するコマンドのorapkiヘルプの表示
  CRLの管理に使用可能な`orapki`コマンドをすべて表示できます。
- 証明書検証用ハッシュ値によるCRLの名前変更
  システムは、証明書を検証するとき、証明書を作成したCAが発行するCRLを見つける必要があります。
- Oracle Internet DirectoryへのCRLのアップロード
  ディレクトリでCRLを発行すると、企業全体でのCRL検証が可能になり、個々のアプリケーションに固有のCRLを構成する必要がなくなります。
- Oracle Internet Directoryに格納されているCRLの一覧表示
  `orapki`を使用してディレクトリに格納されているすべてのCRLのリストを表示できます。特定のCRLを検出してローカル・コンピュータで表示またはダウンロードする際に、このリストを参照すると便利です。
- Oracle Internet DirectoryでのCRLの表示
  Oracle Internet Directory CRLは要約された形式で表示できるほか、CRLの失効した証明書のリストを要求することもできます。
- Oracle Internet DirectoryからのCRLの削除
  `orapki`を使用してディレクトリからCRLを削除するユーザーは、ディレクトリ・グループ`CRLAdmins`のメンバーである必要があります。

親トピック: 証明書失効リストによる証明書の検証

23.13.5.1 証明書失効リストの管理について

Oracle Databaseには、証明書の管理を実行するために使用できるコマンドライン・ユーティリティ`orapki`があります。

証明書失効ステータス・チェックを有効にするには、まず、使用するCAから受信したCRLがコンピュータで使用できるフォームである(ハッシュ値で名前変更されている)こと、またはコンピュータで使用できる場所にある(ディレクトリにアップロードされている)ことを確認してください。

LDAPコマンド行ツールを使用して、Oracle Internet DirectoryでCRLを管理することもできます。

ノート:

CRL は、正常な検証を行うために、(期限切れになる前に)定期的に更新する必要があります。このタスクは、スクリプトで `orapki` コマンドを使用して自動化できます。

親トピック: 証明書失効リストの管理

23.13.5.2 CRLを管理するコマンドのorapkiヘルプの表示

CRLの管理に使用可能なorapkiコマンドをすべて表示できます。

- CRLの管理に使用可能なorapkiコマンドとそのオプションをすべての表示するには、コマンドラインに次のように入力します。

```
orapki crl help
```

ノート:

-summary、-complete または-wallet コマンド・オプションの使用は、常にオプションです。これらのコマンド・オプションが指定されていなくても、コマンドは実行されます。

親トピック: 証明書失効リストの管理

23.13.5.3 証明書検証用ハッシュ値によるCRLの名前変更

システムは、証明書を検証するとき、証明書を作成したCAが発行するCRLを見つける必要があります。

システムは、証明書の発行者名とCRLにある発行者名を照合して適切なCRLを検出します。

Oracle Net Managerの「証明書失効リスト・パス」フィールドにCRL格納場所を指定する(sqlnet.oraファイルのSSL_CRL_PATHパラメータを設定する)場合、orapkiユーティリティを使用して発行者名を表すハッシュ値を使用してCRLを名前変更します。ハッシュ値を作成すると、サーバーでCRLをロードできます。

UNIXオペレーティング・システムでは、orapkiによってCRLへのシンボリック・リンクが作成されます。Windowsオペレーティング・システムでは、CRLファイルのコピーが作成されます。どちらの場合も、orapkiによって作成されたシンボリック・リンクまたはコピーは、発行者名のハッシュ値を使用して名前が付けられます。その後、システムが証明書を検証するとき、同じハッシュ関数が使用されて、適切なCRLをロードできるようにリンク(またはコピー)の名前が計算されます。

- オペレーティング・システムに応じて、次のコマンドのいずれかを入力して、ファイル・システムに格納されているCRLの名前を変更します。
  - UNIXファイル・システムに格納されているCRLの名前を変更するには:

```
orapki crl hash -crl crl_filename [-wallet wallet_location] -symlink
crl_directory [-summary]
```

  - Windowsファイル・システムに格納されているCRLの名前を変更するには:

```
orapki crl hash -crl crl_filename [-wallet wallet_location] -copy
crl_directory [-summary]
```

この指定では、crl_filenameはCRLファイルの名前、wallet_locationはCRLを発行したCAの証明書を含むウォレットの場所、crl_directoryはCRLがあるディレクトリです。

-walletおよび-summaryの使用はオプションです。-walletを指定すると、CRLの名前を変更する前に、ツールによってCAの証明書に対するCRLの有効性が確認されます。-summaryオプションを指定すると、ツールによってCRL発行者名が表示されます。

親トピック: 証明書失効リストの管理

23.13.5.4 Oracle Internet DirectoryへのCRLのアップロード

ディレクトリでCRLを発行すると、企業全体でのCRL検証が可能になり、個々のアプリケーションに固有のCRLを構成する必要がなくなります。

すべてのアプリケーションで、一元的に管理可能なディレクトリに格納されたCRLを使用できるので、CRL管理および使用の管理オーバーヘッドが大幅に減ります。orapkiを使用してディレクトリにCRLをアップロードするユーザーは、ディレクトリ・グループCRLAdmins(cn=CRLAdmins,cn=groups,%s_OracleContextDN%)のメンバーである必要があります。これらのCRLには企業全体でアクセスできるため、これは特権操作です。この管理ディレクトリ・グループに追加するには、ディレクトリ管理者に連絡してください。

- CRLをディレクトリにアップロードするには:

  ```
  orapki crl upload -crl crl_location -ldap hostname:ssl_port -user username [-
  wallet wallet_location] [-summary]
  ```

  この指定では、crl_locationはCRLがあるファイル名またはURL、hostnameおよびssl_port (認証なしのTLSポート)はディレクトリがインストールされているシステムに対するもの、usernameはCRLをCRLサブツリーに追加する権限があるディレクトリ・ユーザー、wallet_locationはCRLを発行したCAの証明書を含むウォレットの場所です。

-walletおよび-summaryの使用はオプションです。-walletを指定すると、CRLをディレクトリにアップロードする前に、ツールによってCAの証明書に対するCRLの有効性が確認されます。-summaryオプションを指定すると、CRL発行者名、およびCRLがディレクトリに格納されているLDAPエントリがツールによって出力されます。

次の例では、orapkiユーティリティを使用してCRLをアップロードする方法を示しています。

```
orapki crl upload -crl /home/user1/wallet/crldir/crl.txt -ldap host1.example.com:3533
-user cn=orcladmin
```

ノート:

- orapki ユーティリティを使用すると、この操作の実行時にディレクトリ・パスワードの入力を求めるプロンプトが表示されます。
- Diffie-Hellman ベースの TLS サーバーが実行されているディレクトリの SSL ポートを指定していることを確認してください。これは、認証を実行しない TLS ポートです。サーバー認証も相互認証 TLS ポートも、orapki ユーティリティではサポートされていません。

親トピック: 証明書失効リストの管理

23.13.5.5 Oracle Internet Directoryに格納されているCRLの一覧表示

orapkiを使用してディレクトリに格納されているすべてのCRLのリストを表示できます。特定のCRLを検出してローカル・コンピュータで表示またはダウンロードする際に、このリストを参照すると便利です。

このコマンドを使用すると、CRLを発行したCA(発行者)およびディレクトリのCRLサブツリー内の場所(DN)が表示されます。

- Oracle Internet DirectoryでCRLを一覧表示するには:

  ```
  orapki crl list -ldap hostname:ssl_port
  ```

  hostnameおよびssl_portは、ディレクトリがインストールされているシステムに対するものです。これは、前の項で説

明した認証なしのディレクトリのSSLポートであることに注意してください。

親トピック: 証明書失効リストの管理

23.13.5.6 Oracle Internet DirectoryでのCRLの表示

Oracle Internet Directory CRLは要約された形式で表示できるほか、CRLの失効した証明書のリストを要求することもできます。

Oracle Internet Directoryに格納されているCRLを要約された形式で表示するか、CRLの失効した証明書の完全なリストを要求できます。サマリー・リストには、CRL発行者名およびその有効期間が記載されています。完全なリストには、そのCRLに含まれるすべての失効した証明書のリストが記載されています。

- Oracle Internet DirectoryでCRLのサマリー・リストを表示するには:

```
orapki crl display -crl crl_location [-wallet wallet_location] -summary
```

この指定で、`crl_location`はディレクトリ内のCRLの場所です。`orapki crl list`コマンドを使用すると表示されるリストからCRLの位置を貼り付けると便利です。

Oracle Internet Directoryに格納されている指定したCRLに含まれているすべての失効した証明書のリストを表示するには、コマンドラインに次のように入力します。

```
orapki crl display -crl crl_location [-wallet wallet_location] -complete
```

たとえば、`orapki`コマンドを次のように入力します。

```
orapki crl display -crl $T_WORK/pki/wlt_crl/nzcrl.txt -wallet
$T_WORK/pki/wlt_crl -complete
```

次の出力が生成されます。これには、CRL発行者のDN、発行日、次の更新日、および含まれる失効した証明書が示されます。

```
issuer = CN=root,C=us, thisUpdate = Sun Nov 16 10:56:58 PST 2003, nextUpdate =
Mon Sep 30 11:56:58 PDT 2013, revokedCertificates = {(serialNo =
153328337133459399575438325845117876415, revocationDate - Sun Nov 16 10:56:58
PST 2003)}
CRL is valid
```

`-wallet`オプションを使用すると、`orapki crl display`コマンドにより、CAの証明書に対してCRLが検証されます。

`-complete`オプションを選択すると、CRLのサイズによっては、表示に時間がかかる場合があります。

Oracle Directory Manager(Oracle Internet Directoryに付属するグラフィカル・ユーザー・インタフェース)を使用して、ディレクトリ内のCRLを表示することもできます。CRLは、次のディレクトリの場所に格納されます。

```
cn=CRLValidation,cn=Validation,cn=PKI,cn=Products,cn=OracleContext
```

関連トピック

- Oracle Internet Directoryに格納されているCRLの一覧表示

親トピック: 証明書失効リストの管理

23.13.5.7 Oracle Internet DirectoryからのCRLの削除

`orapki`を使用してディレクトリからCRLを削除するユーザーは、ディレクトリ・グループ`CRLAdmins`のメンバーである必要があり

ます。

- CRLをディレクトリから削除するには:

```
orapki crl delete -issuer issuer_name -ldap host:ssl_port -user username [-
summary]
```

ここで、issuer_nameはCRLを発行したCAの名前、hostnameおよびssl_portはディレクトリがインストールされているシステムに対するもの、usernameはCRLをCRLサブツリーから削除する権限があるディレクトリ・ユーザーです。このポートは、認証を使用しないディレクトリのSSLポートであることが必要です。

-summaryオプションを使用すると、削除されたCRL LDAPエントリがツールによって印刷されます。

たとえば、orapkiコマンドを次のように入力します。

```
orapki crl delete -issuer "CN=root,C=us" -ldap machine1:3500 -user cn=orcladmin
-summary
```

次の出力が生成されます。これには、ディレクトリ内の削除されたCRLの場所が示されます。

```
Deleted CRL at cn=root
cd45860c.rN,cn=CRLValidation,cn=Validation,cn=PKI,cn=Products,cn=OracleContext
```

関連トピック

- Oracle Internet DirectoryへのCRLのアップロード

親トピック: 証明書失効リストの管理

23.13.6 CRL証明書検証のトラブルシューティング

CRLに対して証明書を検証するかどうかを判断するには、Oracle Netトレースを有効にできます。

CRLを使用して失効した証明書を検証すると、Oracle Netトレース・ファイルに次のエントリが記録され、entryとexitの間にエラー・メッセージは記録されません。

```
nzcrlVCS_VerifyCRLSignature: entry
nzcrlVCS_VerifyCRLSignature: exit
nzcrlVCD_VerifyCRLDate: entry
nzcrlVCD_VerifyCRLDate: exit
nzcrlCCS_CheckCertStatus: entry
nzcrlCCS_CheckCertStatus: Certificate is listed in CRL
nzcrlCCS_CheckCertStatus: exit
```

ノート:

証明書検証に失敗した場合は、SSL ハンドシェイク時にピアに「ORA-29024: 証明書の検証に失敗しました」というメッセージが表示されます。

関連トピック

- 証明書検証に関連するOracle Netトレース・ファイルのエラー・メッセージ
- Oracle Database Net Services管理者ガイド

親トピック: 証明書失効リストによる証明書の検証

23.13.7 証明書検証に関連するOracle Netトレース・ファイルのエラー・メッセージ

証明書検証に関連するトレース・メッセージが生成されます。

これらのトレースメッセージは、Oracle Netトレース・ファイルの`entry`エントリと`exit`エントリの間に記録されることがあります。Oracle SSLは複数の場所でCRLを検索するため、トレースに複数のエラーが記録されることがあります。

エラーの解決方法の詳細は、次の表示される可能性があるエラー・メッセージのリストを確認できます。

RSA ステータスで CRL の署名検証に失敗しました

原因: CRL の署名を検証できません。

処置: ダウンロードした CRL がピアの CA によって発行されたものであることと、ダウンロード時に破損しなかったことを確認します。CRL は、`orapki` ユーティリティによって検証されてから、ハッシュ値で名前が変更されるかディレクトリにアップロードされます。

CRL 管理に `orapki` を使用する方法の詳細は、「証明書失効リストの管理」を参照してください。

RSA ステータスで CRL の日付検証に失敗しました

原因: 現在の時刻が次の更新フィールド内の時刻より後になっています。このエラーは、CRL DP を使用している場合には表示されません。CRL は次の順番で検索されます。

1. ファイル・システム
2. Oracle Internet Directory
3. CRL DP

この検索で最初に見つかった CRL が最新であるとはかぎりません。

処置: CRL を最新のコピーで更新します。

CRL が見つかりませんでした

原因: 構成されている場所に CRL がありませんでした。構成で証明書検証が必須に指定されている場合、エラーORA-29024 が戻されます。

処置: 次のステップを実行して、構成で指定されている CRL の場所が正しいことを確認します。

1. Oracle Net Manager を使用して、正しい CRL の場所が構成されているかどうかを確認します。証明書失効リストによる証明書検証の構成を参照してください
2. 必要に応じて、`orapki` ユーティリティを使用してシステムで使用する CRL を次のように構成し

ます。

- ローカル・ファイル・システムに格納されている CRL の場合は、証明書検証用ハッシュ値による CRL の名前変更を参照してください
- ディレクトリに格納されている CRL の場合は、Oracle Internet Directory への CRL のアップロードを参照してください

Oracle Internet Directory のホスト名またはポート番号が設定されていません

原因: Oracle Internet Directory の接続情報が設定されていません。これは致命的エラーではありません。続いて CRL DP が検索されます。

処置: CRL を Oracle Internet Directory に格納する場合は、Oracle Net Configuration Assistant を使用して、Oracle ホームの `ldap.ora` ファイルを作成し、構成します。

CRL DP からの CRL のフェッチ: CRL が見つかりません

原因: CRL 配布ポイント(CRL DP)を使用して CRL をフェッチできませんでした。このことは、証明書に CRL DP 拡張で指定された場所がないか、CRL DP 拡張で指定された URL が正しくない場合に発生します。

処置: 認証局が証明書の CRL DP 拡張で指定された URL に CRL を発行することを確認します。

CRL を手動でダウンロードします。次に、それをローカル・ファイル・システムに格納するか、Oracle Internet Directory に格納するかに応じて、次のステップを実行します。

CRL をローカル・ファイル・システムに格納する場合:

1. Oracle Net Manager を使用して、CRL ディレクトリまたはファイルへのパスを指定します。証明書失効リストによる証明書検証の構成を参照してください
2. `orapki` ユーティリティを使用して、システムで使用する CRL を構成します。証明書検証用ハッシュ値による CRL の名前変更を参照してください

CRL を Oracle Internet Directory に格納する場合:

1. Oracle Net Configuration Assistant を使用して、ディレクトリ接続情報を含む `ldap.ora` ファイルを作成し、構成します。
2. `orapki` ユーティリティを使用して、CRL をディレクトリにアップロードします。Oracle Internet Directory への CRL のアップロードを参照してください

親トピック: 証明書失効リストによる証明書の検証

23.14 ハードウェア・セキュリティ・モジュールを使用するためのシステムの構成

Oracle Databaseでは、RSA Security社のPKCS #11仕様に準拠したAPIを使用するハードウェア・セキュリティ・モジュールがサポートされています。

通常、これらのハードウェア・デバイスは、トークンまたはスマートカードの秘密キーを安全に格納および管理するため、または暗号処理を高速化するために使用されます。

- TLSでハードウェア・セキュリティ・モジュールを使用するための一般的なガイドライン
  Oracle Databaseでハードウェア・セキュリティ・モジュールを使用する場合は、次のガイドラインに従ってください。
- nCipherハードウェア・セキュリティ・モジュールを使用するためのシステムの構成
  暗号化処理でnCipherハードウェア・セキュリティ・モジュールを使用するようにシステムを構成できます。
- SafeNETハードウェア・セキュリティ・モジュールを使用するためのシステムの構成
  暗号化処理でSafeNETハードウェア・セキュリティ・モジュールを使用するようにシステムを構成できます。
- ハードウェア・セキュリティ・モジュールの使用時のトラブルシューティング
  ハードウェア・セキュリティ・モジュールに関するトラブルシューティングのアドバイスを提供します。

親トピック: Transport Layer Security認証の構成

23.14.1 TLSでハードウェア・セキュリティ・モジュールを使用するための一般的なガイドライン

Oracle Databaseでハードウェア・セキュリティ・モジュールを使用する場合は、次のガイドラインに従ってください。

1. 必要なハードウェア、ソフトウェアおよびPKCS #11ライブラリを取得するには、ハードウェア・デバイス・ベンダーにお問い合せください。
2. 使用しているハードウェア・セキュリティ・モジュールに適したハードウェア、ソフトウェアおよびライブラリをインストールします。
3. ハードウェア・セキュリティ・モジュールのインストールをテストして、正常に動作することを確認します。詳細は、使用するデバイスのドキュメントを参照してください。
4. 秘密キーをトークンに格納する場合は、Oracle Wallet Managerを使用してPKCS11タイプのウォレットを作成し、PKCS #11ライブラリへの絶対パス(ライブラリ名を含む)を指定します。Oracle PKCS11ウォレットには、秘密キーにアクセスするためのトークンを指す情報が格納されます。

PKCS #11情報が格納されたウォレットは、任意のOracleウォレットを使用する場合と同様に使用できます。ただし、秘密キーをハードウェア・デバイスに格納し、暗号操作もそのデバイスで実行する場合を除きます。

関連トピック

- Oracle Databaseエンタープライズ・ユーザー・セキュリティ管理者ガイド

親トピック: ハードウェア・セキュリティ・モジュールを使用するためのシステムの構成

23.14.2 nCipherハードウェア・セキュリティ・モジュールを使用するためのシステムの構成

暗号化処理でnCipherハードウェア・セキュリティ・モジュールを使用するようにシステムを構成できます。

- nCipherハードウェア・セキュリティ・モジュールを使用するためのシステムの構成について
  nCipher社製のハードウェア・セキュリティ・モジュールは、Oracle Databaseで動作することが証明されています。

- nCipherハードウェア・セキュリティ・モジュールに必要なOracleコンポーネント
  nCipherハードウェア・セキュリティ・モジュールを使用するには、特別なコンポーネントのセットが必要です。
- nCipherハードウェア・セキュリティ・モジュールをインストールするためのディレクトリ・パス要件
  nCipherハードウェア・セキュリティ・モジュールはnCipher PKCS #11ライブラリを使用します。

親トピック: ハードウェア・セキュリティ・モジュールを使用するためのシステムの構成

23.14.2.1 nCipherハードウェア・セキュリティ・モジュールを使用するためのシステムの構成について

nCipher社製のハードウェア・セキュリティ・モジュールは、Oracle Databaseで動作することが証明されています。

これらのモジュールでは、キーを安全に格納し、暗号処理の負荷を軽減できます。これらのデバイスには、主に次の利点があります。

- 暗号処理の負荷が軽減され、サーバーが他の要求に応答できるようになります。
- デバイス上の秘密キーの記憶領域が保護されます。
- スマートカードを使用してキーを管理できます。

ノート:

Oracle Database で使用する認定済のハードウェアおよびソフトウェアを入手するには、nCipher の代理店にお問い合せください。

親トピック: nCipherハードウェア・セキュリティ・モジュールを使用するためのシステムの構成

23.14.2.2 nCipherハードウェア・セキュリティ・モジュールに必要なOracleコンポーネント

nCipherハードウェア・セキュリティ・モジュールを使用するには、特別なコンポーネントのセットが必要です。

これらのコンポーネントは次のとおりです。

- nCipherハードウェア・セキュリティ・モジュール
- サポートしているnCipher PKCS #11ライブラリ

  次のプラットフォーム固有のPKCS#11ライブラリが必要です。

  - `libcknfast.so`ライブラリ(UNIX 32ビットの場合)
  - `libcknfast-64.so`ライブラリ(UNIX 64ビットの場合)
  - `cknfast.dll`ライブラリ(Windowsの場合)

ノート:

ハードウェア・セキュリティ・モジュールまたはセキュア・アクセラレータをインストールし、必要なライブラリを入手するには、nCipher の代理店にお問い合せください。

これらの作業は、Oracle Database で nCipher ハードウェア・セキュリティ・モジュールを使用する前に実施する必要があります。

親トピック: nCipherハードウェア・セキュリティ・モジュールを使用するためのシステムの構成

23.14.2.3 nCipherハードウェア・セキュリティ・モジュールをインストールするためのディレクトリ・パス要件

nCipherハードウェア・セキュリティ・モジュールはnCipher PKCS #11ライブラリを使用します。

セキュア・アクセラレータを使用するには、Oracle Wallet Managerを使用してウォレットを作成するときに、nCipher PKCS #11ライブラリが格納されているディレクトリへの絶対パス(ライブラリ名を含む)を指定する必要があります。これにより、実行時にライブラリをロードできます。

通常、nCipherカードは次の場所にインストールされます。

- `/opt/nfast` (UNIXの場合)
- `C:¥nfast` (Windowsの場合)

nCipher PKCS #11ライブラリは、通常のインストールでは次の場所に配置されます。

- `/opt/nfast/toolkits/pkcs11/libcknfast.so`(UNIX 32ビットの場合)
- `/opt/nfast/toolkits/pkcs11/libcknfast-64.so`(UNIX 64ビットの場合)
- `C:¥nfast¥toolkits¥pkcs11¥cknfast.dll` (Windowsの場合)

ノート:

Oracle Database の 32 ビット・リリースを使用する場合は 32 ビット・ライブラリ・バージョンを使用し、Oracle Database の 64 ビット・リリースを使用する場合は 64 ビット・ライブラリ・バージョンを使用します。たとえば、Oracle Database for Solaris Operating System (SPARC 64 ビット)には、64 ビット nCipher PKCS #11 ライブラリを使用します。

親トピック: nCipherハードウェア・セキュリティ・モジュールを使用するためのシステムの構成

23.14.3 SafeNETハードウェア・セキュリティ・モジュールを使用するためのシステムの構成

暗号化処理でSafeNETハードウェア・セキュリティ・モジュールを使用するようにシステムを構成できます。

- SafeNETハードウェア・セキュリティ・モジュールを使用するためのシステムの構成について
  SafeNET社製のハードウェア・セキュリティ・モジュールは、Oracle Databaseで動作することが証明されています。
- SafeNET Luna SAハードウェア・セキュリティ・モジュールに必要なOracleコンポーネント
  SafeNET Luna SAハードウェア・セキュリティ・モジュールを使用するには、特別なコンポーネントのセットが必要です。
- SafeNETハードウェア・セキュリティ・モジュールをインストールするためのディレクトリ・パス要件
  SafeNETハードウェア・セキュリティ・モジュールはSafeNET PKCS #11ライブラリを使用します。

親トピック: ハードウェア・セキュリティ・モジュールを使用するためのシステムの構成

23.14.3.1 SafeNETハードウェア・セキュリティ・モジュールを使用するためのシステムの構成について

SafeNET社製のハードウェア・セキュリティ・モジュールは、Oracle Databaseで動作することが証明されています。

これらのモジュールでは、キーを安全に格納し、暗号処理の負荷を軽減できます。これらのデバイスには、主に次の利点がありま

す。

- 暗号処理の負荷が軽減され、サーバーが他の要求に応答できるようになります。
- デバイス上の秘密キーの記憶領域が保護されます。

ノート:

Oracle Database で使用する認定済のハードウェアおよびソフトウェアを入手するには、SafeNET の代理店にお問い合せください。

親トピック: SafeNETハードウェア・セキュリティ・モジュールを使用するためのシステムの構成

23.14.3.2 SafeNET Luna SAハードウェア・セキュリティ・モジュールに必要なOracleコンポーネント

SafeNET Luna SAハードウェア・セキュリティ・モジュールを使用するには、特別なコンポーネントのセットが必要です。

これらのコンポーネントは次のとおりです。

- SafeNET Luna SAハードウェア・セキュリティ・モジュール
- サポートしているSafeNET Luna SA PKCS #11ライブラリ

  次のプラットフォーム固有のPKCS#11ライブラリが必要です。

  - `libCryptoki2.so`ライブラリ(UNIXの場合)
  - `cryptoki.dll`ライブラリ(Windowsの場合)

ノート:

ハードウェア・セキュリティ・モジュールまたはセキュア・アクセラレータをインストールし、必要なライブラリを入手するには、SafeNET の代理店にお問い合せください。

これらの作業は、Oracle Database で SafeNET ハードウェア・セキュリティ・モジュールを使用する前に実施する必要があります。

親トピック: SafeNETハードウェア・セキュリティ・モジュールを使用するためのシステムの構成

23.14.3.3 SafeNETハードウェア・セキュリティ・モジュールをインストールするためのディレクトリ・パス要件

SafeNETハードウェア・セキュリティ・モジュールはSafeNET PKCS #11ライブラリを使用します。

セキュア・アクセラレータを使用するには、Oracle Wallet Managerを使用してウォレットを作成するときに、SafeNET PKCS #11ライブラリが格納されているディレクトリへの絶対パス(ライブラリ名を含む)を指定する必要があります。これにより、実行時にライブラリをロードできます。

通常、SafeNET Luna SAクライアントは次の場所にインストールされます。

- `/usr/lunasa` (UNIXの場合)
- `C:¥Program Files¥LunaSA` (Windowsの場合)

SafeNET Luna SA PKCS #11ライブラリは、通常のインストールでは次の場所に配置されます。

- `/usr/lunasa/lib/libCryptoki2.so` (UNIXの場合)
- `C:¥Program Files¥LunaSA¥cryptoki2.dll` (Windowsの場合)

親トピック: SafeNETハードウェア・セキュリティ・モジュールを使用するためのシステムの構成

23.14.4 ハードウェア・セキュリティ・モジュールの使用時のトラブルシューティング

ハードウェア・セキュリティ・モジュールに関するトラブルシューティングのアドバイスを提供します。

- Oracle Netトレース・ファイルのエラー
  使用されているモジュールを検出するには、Oracle Netトレースをオンにします。
- ハードウェア・セキュリティ・モジュールの使用に関連するエラー・メッセージ
  PKCS #11ハードウェア・セキュリティ・モジュールの使用に関連するエラーが表示される場合があります。

親トピック: ハードウェア・セキュリティ・モジュールを使用するためのシステムの構成

23.14.4.1 Oracle Netトレース・ファイルのエラー

使用されているモジュールを検出するには、Oracle Netトレースをオンにします。

ウォレットにPKCS #11情報が含まれ、モジュールの秘密キーが使用されている場合は、Oracle Netトレースに次のエントリが記録され、`entry`と`exit`の間にエラー・メッセージは記録されません。

```
nzpkcs11_Init: entry
nzpkcs11CP_ChangeProviders: entry
nzpkcs11CP_ChangeProviders: exit
nzpkcs11GPK_GetPrivateKey: entry
nzpkcs11GPK_GetPrivateKey: exit
nzpkcs11_Init: exit
...
nzpkcs11_Decrypt: entry
nzpkcs11_Decrypt: exit
nzpkcs11_Sign: entry
nzpkcs11_Sign: exit
```

関連項目:

トレース・パラメータを設定してOracle Netトレースを有効にする方法の詳細は、『Oracle Database Net Services管理者ガイド』を参照してください。

親トピック: ハードウェア・セキュリティ・モジュールの使用時のトラブルシューティング

23.14.4.2 ハードウェア・セキュリティ・モジュールの使用に関連するエラー・メッセージ

PKCS #11ハードウェア・セキュリティ・モジュールの使用に関連するエラーが表示される場合があります。

ORA-43000: PKCS11 ライブラリが見つかりません

原因: ウォレットの作成時に指定された場所に PKCS #11 ライブラリが見つかりません。これは、ウォレットの作成後にライブラリが移動された場合にのみ発生します。

処置: PKCS #11 ライブラリをウォレットが作成されたときの場所にコピーします。

ORA-43001: PKCS11 トークンが見つかりません

原因: ウォレットの作成に使用されたスマートカードがハードウェア・セキュリティ・モジュールのスロットにありません。

処置: ウォレットの作成時に使用されたスマートカードがハードウェア・セキュリティ・モジュールのスロットにあることを確認します。

ORA-43002: PKCS11 パスワードが無効です

原因: これは、ウォレットの作成時に指定されたパスワードが正しくない場合、またはウォレットの作成後に PKCS #11 デバイス・パスワードを変更し、Oracle Wallet Manager を使用してウォレットで更新しなかった場合に発生する可能性があります。

処置: 原因に応じて、次のいずれかの処置を行います。

ウォレットの作成時にこのエラーが表示された場合は、パスワードが正しいことを確認し、再入力します。

ウォレットの作成後にパスワードを変更した場合は、Oracle Wallet Manager を使用してウォレットを開き、新しいパスワードを入力します。

関連項目:

ハードウェア・セキュリティ証明書を格納するためにOracleウォレットを作成する方法の詳細は、『Oracle Databaseエンタープライズ・ユーザー・セキュリティ管理者ガイド』を参照してください。

ノート:

nCipher ログ・ファイルは、モジュールがインストールされているディレクトリの次の場所にあります。

```
/log/logfile
```

関連項目:

nCipherデバイスとSafeNETデバイスのトラブルシューティングの詳細は、nCipherとSafeNETのドキュメントを参照してください。

親トピック: ハードウェア・セキュリティ・モジュールの使用時のトラブルシューティング

24 RADIUS認証の構成

RADIUSは、リモート認証およびアクセスを実現するために広く使用されているクライアント/サーバー・セキュリティ・プロトコルです。

- RADIUS認証の構成について
  Oracle Databaseネットワークでは、RADIUS標準をサポートする任意の認証方法を使用できます。
- RADIUSの構成要素
  RADIUSには、構成設定を管理できる一連の認証の構成要素があります。
- RADIUS認証モード
  ユーザー認証は、同期認証モードまたはチャレンジ・レスポンス(非同期)認証モードのいずれかで行われます。
- RADIUS認証、認可およびアカウンティングの有効化
  RADIUS認証、認可およびアカウンティングを有効にするには、Oracle Net Managerを使用します。
- RADIUSを使用したデータベースへのログイン
  RADIUSを使用してデータベースにログインするには、同期認証モードまたはチャレンジ・レスポンス・モードのいずれかを使用します。
- RSA ACE/Server構成チェックリスト
  RSA ACE/Server RADIUSサーバーを使用する場合は、最初の接続を試みる前に、このサーバーのホスト・エージェントとSecurIDトークンを確認してください。

親トピック: 厳密認証の管理

24.1 RADIUS認証の構成について

Oracle Databaseネットワークでは、RADIUS標準をサポートする任意の認証方法を使用できます。

RADIUSプロトコルをインストールおよび構成するときには、トークン・カードやスマート・カードといったRADIUS標準もサポートされます。Oracle Databaseは、クライアント/サーバー・ネットワーク環境でRADIUSを使用します。さらに、RADIUSを使用すると、OracleクライアントまたはOracleデータベース・サーバーを変更することなく、認証方式を変更できます。

エンド・ユーザーから見ると、認証手続き全体は透過的です。ユーザーがOracleデータベース・サーバーにアクセスしようとすると、RADIUSクライアントとして機能するOracleデータベース・サーバーはRADIUSサーバーに通知します。RADIUSサーバーは次の処理を行います。

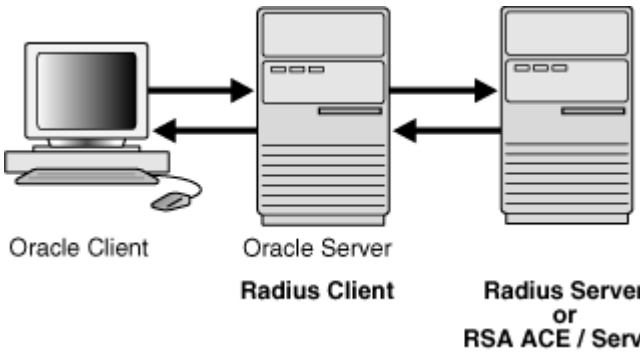
- ユーザーのセキュリティ情報を検索します。
- 適切な認証サーバー(複数可)とOracleデータベース・サーバーの間で認証および認可情報の受渡しを行います。
- Oracleデータベース・サーバーへのユーザー・アクセスを許可します。
- ユーザーがいつ、どのくらいの頻度で、どのくらいの時間、Oracleデータベース・サーバーに接続していたかなどのセッション情報をログに記録します。

ノート:

Oracle Database では、データベース・リンクを介した RADIUS 認証はサポートされません。

図24-1に、Oracle Database-RADIUS環境を示します。

図24-1 Oracle環境でのRADIUS



Oracle DatabaseサーバーはRADIUSクライアントとして機能し、OracleクライアントとRADIUSサーバーの間で情報の受渡しを行います。同様に、RADIUSサーバーはOracleデータベース・サーバーと適切な認証サーバーの間で情報の受渡しを行います。

RADIUSサーバーのベンダーが認証サーバーのベンダーでもある場合も多くあります。この場合、RADIUSサーバーで認証を処理できます。たとえば、RSA ACE/Serverは、RADIUSサーバーであり認証サーバーでもあります。したがって、ユーザーのパス・コードが認証されます。

ノート:

RSA Security 社の認証製品である SecurID は、Oracle Database で直接的にはサポートされていませんが、RADIUS 準拠として認定されています。したがって、RADIUS のもとで SecurID を実行できます。

詳細は、RSA Security 社の SecurID のドキュメントを参照してください。

関連項目:

`sqlnet.ora`ファイルの詳細は、Oracle Database Net Servicesリファレンス を参照してください

親トピック: RADIUS認証の構成

24.2 RADIUSの構成要素

RADIUSには、構成設定を管理できる一連の認証の構成要素があります。

表24-1に、認証の構成要素を示します。

表24-1 RADIUS認証の構成要素

構成要素	格納される情報
Oracle クライアント	RADIUS による通信のための構成設定
Oracle データベース・サーバー/RADIUS クライアント	Oracle クライアントと RADIUS サーバーの間で情報の受渡しを行うための構成設定 秘密キー・ファイル
RADIUS サーバー	すべてのユーザーの認証および認可情報

構成要素	格納される情報
	各クライアントの名前または IP アドレス 各クライアントの共有シークレット すでに認証されているユーザーが再接続しなくても別のログイン・オプションを選択できるようにする無制限の数のメニュー・ファイル
認証サーバー(複数可)	パス・コードや PIN などのユーザー認証情報(使用中の認証方式によって異なる) ノート: RADIUS サーバーは認証サーバーになることもできます。

親トピック: RADIUS認証の構成

24.3 RADIUS認証モード

ユーザー認証は、同期認証モードまたはチャレンジ・レスポンス(非同期)認証モードのいずれかで行われます。

- 同期認証モード
  同期モードでは、RADIUSでパスワードやSecurIDトークン・カードなどの様々な認証方式を使用できます。
- チャレンジ・レスポンス(非同期)認証モード
  システムで非同期モードが使用されている場合、ユーザーは、SQL*Plus CONNECT文字列でユーザー名とパスワードを入力する必要はありません。

親トピック: RADIUS認証の構成

24.3.1 同期認証モード

同期モードでは、RADIUSでパスワードやSecurIDトークン・カードなどの様々な認証方式を使用できます。

- 同期認証モードの順序
  同期認証モードは6ステップの順序です。
- 例: SecurIDトークン・カードによる同期認証
  SecurID認証では、各ユーザーにトークン・カードあり、60秒ごとに変わる動的番号が表示されます。

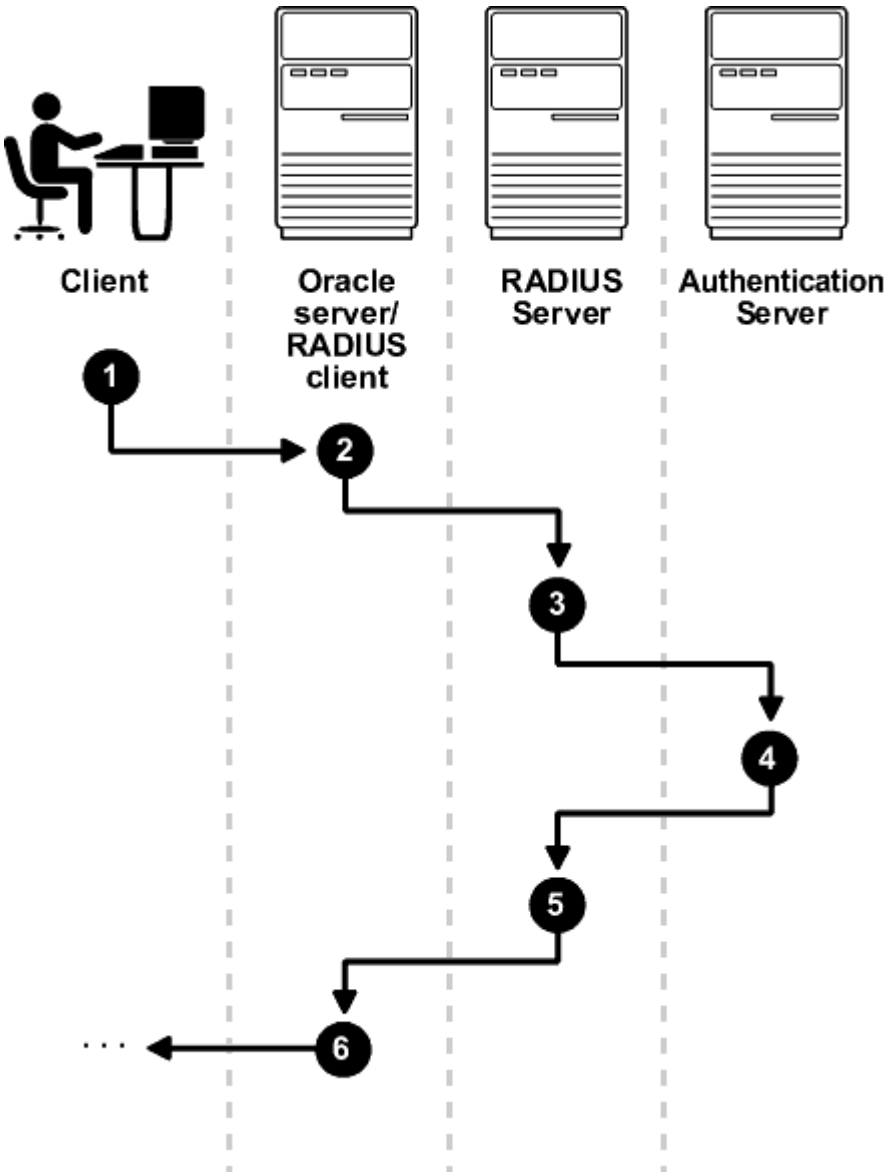
親トピック: RADIUS認証モード

24.3.1.1 同期認証モードの順序

同期認証モードは6ステップの順序です。

図24-2に、同期認証が行われる順序を示します。

図24-2 同期認証の順序



次のステップは、同期認証の順序を示しています。

1. ユーザーが、接続文字列、パス・コードまたはその他の値を入力してログインします。クライアント・システムは、このデータをOracleデータベース・サーバーに渡します。
2. RADIUSクライアントとして機能するOracleデータベース・サーバーは、Oracleクライアントからのデータを RADIUS サーバーに渡します。
3. RADIUSサーバーは、検証のためにデータをスマートカードやSecurID ACEなどの適切な認証サーバーに渡します。
4. 認証サーバーは、アクセス受入れまたはアクセス拒否メッセージをRADIUSサーバーに返します。
5. RADIUSサーバーは、このレスポンスをOracleデータベース・サーバー/RADIUSクライアントに渡します。
6. Oracleデータベース・サーバー/RADIUSクライアントは、レスポンスをOracleクライアントに返します。

親トピック: 同期認証モード

24.3.1.2 例: SecurIDトークン・カードによる同期認証

SecurID認証では、各ユーザーにトークン・カードあり、60秒ごとに変わる動的番号が表示されます。

Oracleデータベース・サーバー/RADIUSクライアントにアクセスするために、ユーザーは個人識別番号(PIN)とユーザーの SecurIDカード上に現在表示されている動的番号の両方を含む有効なパス・コードを入力します。Oracleデータベース・サー

バーは、Oracleクライアントからのこの認証情報をRADIUSサーバー(この場合は、検証のための認証サーバー)に渡します。認証サーバー(RSA ACE/Server)によってユーザーが検証されると、受入れパケットがOracleデータベース・サーバーに送られ、次に、Oracleクライアントに渡されます。これでユーザーは認証され、適切な表やアプリケーションにアクセスできます。

関連項目:

RSA Security社から提供されるドキュメント

親トピック: 同期認証モード

24.3.2 チャレンジ・レスポンス(非同期)認証モード

システムで非同期モードが使用されている場合、ユーザーは、SQL*Plus CONNECT文字列でユーザー名とパスワードを入力する必要はありません。

- チャレンジ・レスポンス(非同期)認証モードの順序
  チャレンジ・レスポンス(非同期)認証モードは12ステップの順序です。
- 例: スマートカードによる非同期認証
  スマートカード認証では、ユーザーは、スマートカードを読み取るスマートカード・リーダーにスマートカードを挿入することによってログインします。
- 例: ActivCardトークンによる非同期認証
  ActivCardトークンの1つに、キーパッドを備えた、動的パスワードを表示するハンドヘルド・デバイスがあります。

親トピック: RADIUS認証モード

24.3.2.1 チャレンジ・レスポンス(非同期)認証モードの順序

チャレンジ・レスポンス(非同期)認証モードは12ステップの順序です。

ノート:

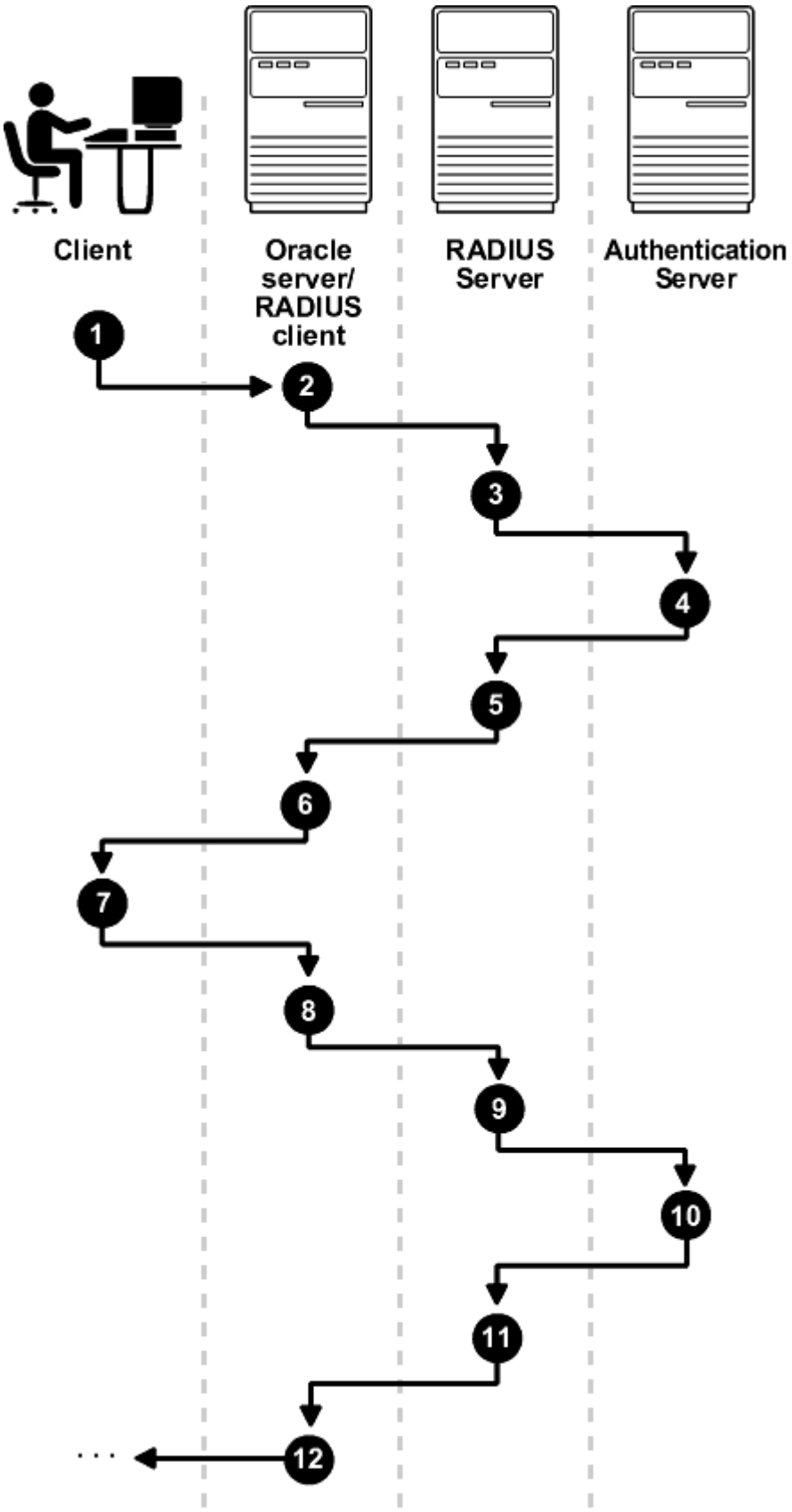
データベース・サーバーへのクライアント接続に関係なく、Microsoft Windows プラットフォームで実行されているデータベース・サーバーではチャレンジ・レスポンス(非同期)認証モードはサポートされていません。

図24-3に、チャレンジ・レスポンス(非同期)認証が行われる順序を示します。

ノート:

RADIUS サーバーが認証サーバーである場合、図 24-3 のステップ 3、4、5 およびステップ 9、10、11 は結合されます。

図24-3 非同期認証の順序



次のステップは、非同期認証の順序を示しています。

1. ユーザーが、Oracleデータベース・サーバーへの接続を開始します。クライアント・システムは、データをOracleデータベース・サーバーに渡します。
2. RADIUSクライアントとして機能するOracleデータベース・サーバーは、Oracleクライアントからのデータを RADIUSサーバーに渡します。
3. RADIUSサーバーは、データをスマートカード、SecurID ACE、トークン・カード・サーバーなどの適切な認証サーバー

に渡します。

4. 認証サーバーは、ランダム番号などのチャレンジをRADIUSサーバーに送信します。
5. RADIUSサーバーは、チャレンジをOracleデータベース・サーバー/RADIUSクライアントに渡します。
6. Oracleデータベース・サーバー/RADIUSクライアントは、それをOracleクライアントに渡します。グラフィカル・ユーザー・インタフェースで、ユーザーにチャレンジが表示されます。
7. ユーザーは、チャレンジに対するレスポンスを入力します。レスポンスを作成するために、ユーザーは、たとえば、受け取ったチャレンジをトークン・カードに入力できます。トークン・カードによって、グラフィカル・ユーザー・インタフェースに入力する動的なパスワードが提供されます。Oracleクライアントは、ユーザーのレスポンスをOracleデータベース・サーバー/RADIUSクライアントに渡します。
8. Oracleデータベース・サーバー/RADIUSクライアントは、ユーザーのレスポンスをRADIUSサーバーに送信します。
9. RADIUSサーバーは、検証のためにユーザーのレスポンスを適切な認証サーバーに渡します。
10. 認証サーバーは、アクセス受入れまたはアクセス拒否メッセージをRADIUSサーバーに返します。
11. RADIUSサーバーは、レスポンスをOracleデータベース・サーバー/RADIUSクライアントに渡します。
12. Oracleデータベース・サーバー/RADIUSクライアントは、レスポンスをOracleクライアントに渡します。

親トピック: チャレンジ・レスポンス(非同期)認証モード

24.3.2.2 例: スマートカードによる非同期認証

スマートカード認証では、ユーザーは、スマートカードを読み取るスマートカード・リーダーにスマートカードを挿入することによってログインします。

スマートカードは、情報を格納するための集積回路が埋め込まれた、クレジット・カードのようなプラスチック・カードです。

Oracleクライアントは、Oracleデータベース・サーバー/RADIUSクライアントおよびRADIUSサーバーを経由して、スマートカードに含まれるログイン情報を認証サーバーに送信します。認証サーバーは、RADIUSサーバーおよびOracleデータベース・サーバーを経由してOracleクライアントにチャレンジを戻し、認証情報をユーザーに要求します。この情報は、PINやスマートカードに含まれている他の認証情報などです。

Oracleクライアントは、Oracleデータベース・サーバーおよびRADIUSサーバーを経由して、ユーザーのレスポンスを認証サーバーに送信します。ユーザーが有効な番号を入力した場合、認証サーバーはRADIUSサーバーおよびOracleデータベース・サーバーを経由して、受入れパケットをOracleクライアントに戻します。これでユーザーは認証され、適切な表やアプリケーションへのアクセスが認可されます。ユーザーが誤った情報を入力した場合、認証サーバーはユーザーのアクセスを拒否するメッセージを戻します。

親トピック: チャレンジ・レスポンス(非同期)認証モード

24.3.2.3 例: ActivCardトークンによる非同期認証

ActivCardトークンの1つに、キーパッドを備えた、動的パスワードを表示するハンドヘルド・デバイスがあります。

ユーザーがパスワードを入力してOracleデータベース・サーバーにアクセスしようとすると、情報はOracleデータベース・サーバー/RADIUSクライアントおよびRADIUSサーバーを経由して、適切な認証サーバーに渡されます。認証サーバーは、RADIUSサーバーおよびOracleデータベース・サーバーを経由して、クライアントにチャレンジを戻します。ユーザーがそのチャレンジをトークンに入力すると、ユーザーがレスポンスで送信する番号がトークンに表示されます。

次に、Oracleクライアントは、Oracleデータベース・サーバーおよびRADIUSサーバーを経由して、ユーザーのレスポンスを認証

サーバーに送信します。ユーザーが有効な番号を入力した場合、認証サーバーはRADIUSサーバーおよびOracleデータベース・サーバーを経由して、受入れパケットをOracleクライアントに戻します。これでユーザーは認証され、適切な表やアプリケーションへのアクセスが認可されます。ユーザーが誤ったレスポンスを入力した場合、認証サーバーはユーザーのアクセスを拒否するメッセージを戻します。

親トピック: チャレンジ・レスポンス(非同期)認証モード

24.4 RADIUS認証、認可およびアカウンティングの有効化

RADIUS認証、認可およびアカウンティングを有効にするには、Oracle Net Managerを使用します。

- ステップ1: RADIUS認証の構成
  RADIUS認証を構成するには、Oracleクライアントで最初に認証を構成してからサーバーで構成する必要があります。その後、追加のRADIUS機能を構成できます。
- ステップ2: ユーザーの作成とアクセス権の付与
  RADIUS認証の完了後、RADIUS構成を担当するOracle Databaseユーザーを作成する必要があります。
- ステップ3: 外部RADIUS認可の構成(オプション)
  Oracleデータベースに接続する必要があるRADIUSユーザーに対して、Oracleサーバー、Oracleクライアントおよび RADIUSサーバーを構成する必要があります。
- ステップ4: RADIUSアカウンティングの構成
  RADIUSアカウンティングは、Oracleデータベース・サーバーへのアクセスに関する情報をログに記録し、RADIUSアカウンティング・サーバー上のファイルに格納します。
- ステップ5: RADIUSクライアント名のRADIUSサーバー・データベースへの追加
  選択するRADIUSサーバーは、RADIUS標準に準拠している必要があります。
- ステップ6: RADIUSとともに使用する認証サーバーの構成
  RADIUSクライアント名をRADIUSサーバー・データベースに追加した後、RADIUSを使用するように認証サーバーを構成できます。
- ステップ7: 認証サーバーとともに使用するRADIUSサーバーの構成
  RADIUSで使用するように認証サーバーを構成した後、その認証サーバーを使用するようにRADIUSサーバーを構成できます。
- ステップ8: マッピング・ロールの構成
  RADIUSサーバーでベンダー・タイプ属性がサポートされている場合は、ロールをRADIUSサーバーに格納して管理できます。

親トピック: RADIUS認証の構成

24.4.1 ステップ1: RADIUS認証の構成

RADIUS認証を構成するには、Oracleクライアントで最初に認証を構成してからサーバーで構成する必要があります。その後、追加のRADIUS機能を構成できます。

ノート:

特に示されていないかぎり、これらの構成タスクを実行するには、Oracle Net Manager を使用するか、任意のテキスト・エディタを使用して `sqlnet.ora` ファイルを変更します。マルチテナント環境では、`sqlnet.ora` ファイル

の設定はすべてのプラガブル・データベース(PDB)に適用されることに注意してください。

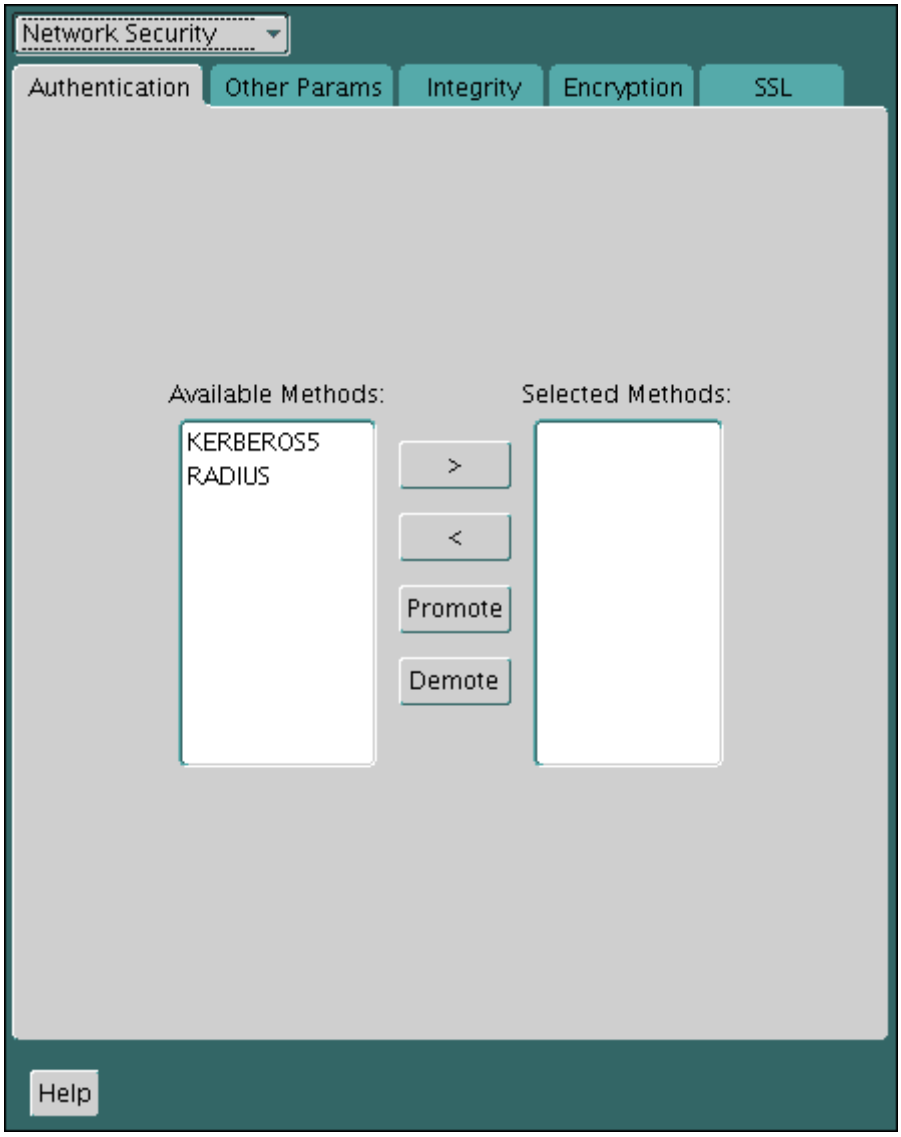
- ステップ1A: OracleクライアントでのRADIUSの構成
  Oracle Net Managerを使用して、OracleクライアントでRADIUSを構成できます。
- ステップ1B: Oracleデータベース・サーバーでのRADIUSの構成
  RADIUSキーを保持するファイルを作成して、Oracleデータベース・サーバー上にこのファイルを格納する必要があります。次に、`sqlnet.ora`ファイルで適切なパラメータを構成する必要があります。
- ステップ1C: その他のRADIUS機能の構成
  デフォルト設定を変更し、チャレンジ・レスポンス・モードを構成して、代替RADIUSサーバーのパラメータを設定できます。

親トピック: RADIUS認証、認可およびアカウンティングの有効化

24.4.1.1 ステップ1A: OracleクライアントでのRADIUSの構成

Oracle Net Managerを使用して、OracleクライアントでRADIUSを構成できます。

1. Oracle Net Managerを起動します。
   - (UNIX) `$ORACLE_HOME/bin`から、コマンドラインで次のコマンドを入力します。
     ```
     netmgr
     ```
   - (Windows)「スタート」→「プログラム」→「Oracle - HOME_NAME」→「Configuration and Migration Tools」→「Net Manager」を選択します。
2. 「Oracle Netの構成」を展開し、「ローカル」から「プロファイル」を選択します。
3. 「ネーミング」リストから、「ネットワーク・セキュリティ」を選択します。
   ネットワーク・セキュリティのタブ付きウィンドウが表示されます。
4. 「認証」タブを選択します。(デフォルトで選択されています。)



5. 「使用可能なメソッド」リストから、RADIUSを選択します。
6. 右矢印(>)を選択して、RADIUSを「選択メソッド」リストに移動します。

   使用するその他のメソッドを同じ方法で移動します。
7. 「選択メソッド」リストでメソッドを選択し、「上へ」または「下へ」をクリックしてリスト内での位置を指定して、選択したメソッドを使用に必要な順序に並べます。

   たとえば、RADIUSをリストの最も上に置いて、使用される最初のサービスにします。
8. 「ファイル」メニューから、「ネットワーク構成の保存」を選択します。

   `sqlnet.ora`ファイルが次のエントリで更新されます。

```
SQLNET.AUTHENTICATION_SERVICES=(RADIUS)
```

親トピック: ステップ1: RADIUS認証の構成

24.4.1.2 ステップ1B: Oracleデータベース・サーバーでのRADIUSの構成

RADIUSキーを保持するファイルを作成して、Oracleデータベース・サーバー上にこのファイルを格納する必要があります。次に、`sqlnet.ora`ファイルで適切なパラメータを構成する必要があります。

- ステップ1B(1): OracleデータベースでのRADIUS秘密キー・ファイルの作成
  最初に、RADIUS秘密キー・ファイルを作成する必要があります。

- ステップ1B(2): サーバー(sqlnet.oraファイル)でのRADIUSパラメータの構成
  RADIUS秘密キー・ファイルの作成後、`sqlnet.ora`ファイルで適切なパラメータを構成します。
- ステップ1B(3): Oracleデータベース・サーバー初期化パラメータの設定
  sqlnet.oraファイルの構成後、`init.ora`初期化ファイルを構成する必要があります。

親トピック: ステップ1: RADIUS認証の構成

24.4.1.2.1 ステップ1B(1): OracleデータベースサーバーでのRADIUS秘密キー・ファイルの作成

最初に、RADIUS秘密キー・ファイルを作成する必要があります。

1. RADIUSサーバーからRADIUS秘密キーを取得します。

   RADIUSクライアントごとに、RADIUSサーバーの管理者が共有秘密キーを作成します。長さは16文字以下である必要があります。

2. Oracleデータベース・サーバーで、次のディレクトリを作成します。

   - (UNIX) `$ORACLE_HOME/network/security`
   - (Windows) `ORACLE_BASE¥ORACLE_HOME¥network¥security`

3. RADIUSサーバーからコピーした共有シークレットを格納するために、ファイル`radius.key`を作成します。このファイルをステップ2で作成したディレクトリに配置します。

4. 共有秘密キーをコピーし、このキーのみを、Oracleデータベース・サーバーに作成した`radius.key`ファイルに貼り付けます。

5. セキュリティのために、`radius.key`のファイル権限を読取り専用に変更し、Oracle所有者のみがアクセスできるようにします。

   Oracleは、ファイル・システムに依存してこのファイルを秘密にします。

   関連項目:

   秘密キーの取得の詳細は、RADIUSサーバーの管理ドキュメントを参照してください。

親トピック: ステップ1B: OracleデータベースサーバーでのRADIUSの構成

24.4.1.2.2 ステップ1B(2): サーバー(sqlnet.oraファイル)でのRADIUSパラメータの構成

RADIUS秘密キー・ファイルの作成後、`sqlnet.ora`ファイルで適切なパラメータを構成します。

1. Oracle Net Managerを起動します。

   - (UNIX) `$ORACLE_HOME/bin`から、コマンドラインで次のコマンドを入力します。

     ```
     netmgr
     ```

   - (Windows)「スタート」→「プログラム」→「Oracle - HOME_NAME」→「Configuration and Migration Tools」→「Net Manager」を選択します。

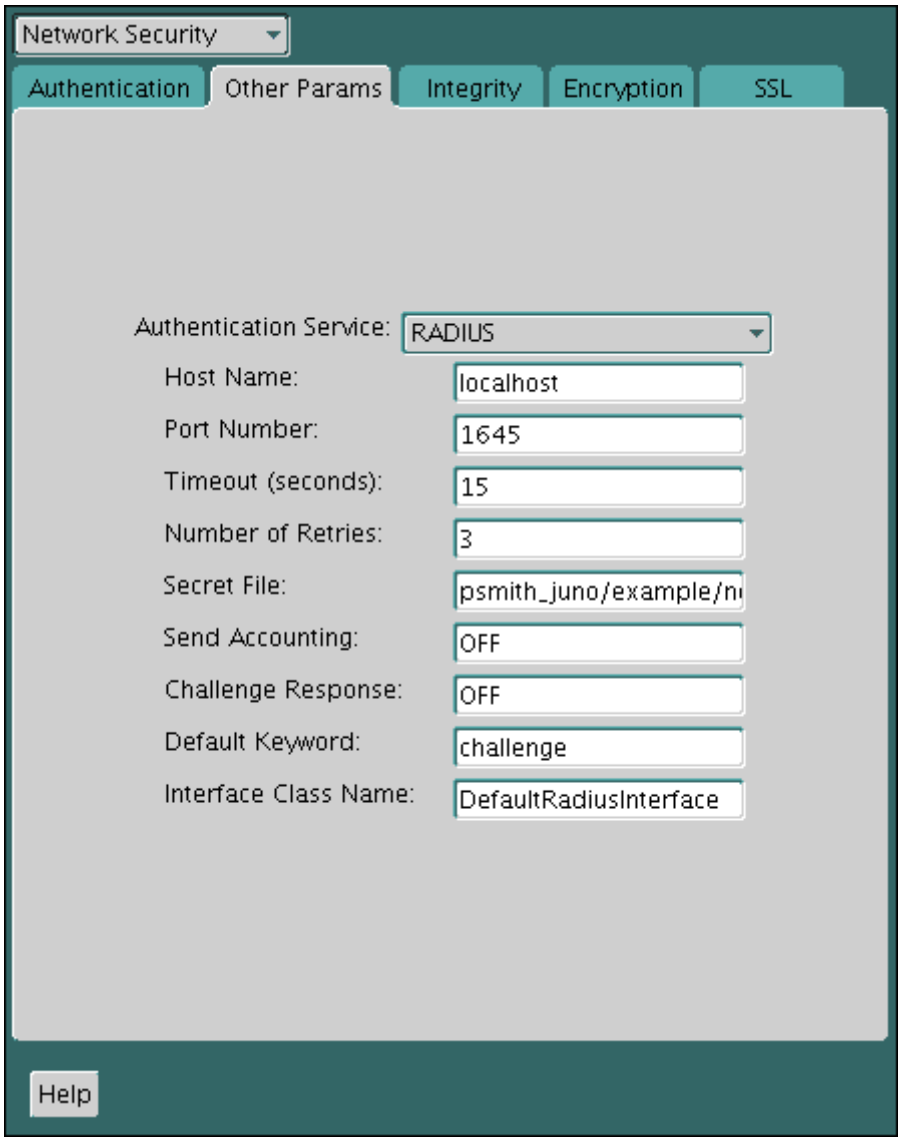
2. 「Oracle Netの構成」を展開し、「ローカル」から「プロファイル」を選択します。

3. 「ネーミング」リストから、「ネットワーク・セキュリティ」を選択します。

   ネットワーク・セキュリティのタブ付きウィンドウが表示されます。

4. 「認証」タブを選択します。
5. 「使用可能なメソッド」リストから、RADIUSを選択します。
6. 右矢印(>)を選択して、RADIUSを「選択メソッド」リストに移動します。
7. 選択したメソッドを使用に必要な順序に並べるには、「選択メソッド」リストでメソッドを選択し、「上へ」または「下へ」を選択してリスト内での位置を指定します。

   たとえば、RADIUSを使用される最初のサービスにする場合は、リストの最も上に置きます。
8. 「その他のパラメータ」タブを選択します。
9. 「認証サービス」リストから、RADIUSを選択します。
10. 「ホスト名」フィールドで、デフォルトのプライマリRADIUSサーバーとしてlocalhostを受け入れるか、または別のホスト名を入力します。



11. 「シークレット・ファイル」フィールドのデフォルト値が有効であることを確認します。
12. 「ファイル」メニューから、「ネットワーク構成の保存」を選択します。

`sqlnet.ora`ファイルは次のエントリで更新されます。

```
SQLNET.AUTHENTICATION_SERVICES=RADIUS
SQLNET.RADIUS_AUTHENTICATION=RADIUS_server_{hostname|IP_address}
```

ノート:

IP_address は、インターネット・プロトコル・バージョン 4 (IPv4)またはインターネット・プロトコル・バージョン 6 (IPv6)のアドレスです。RADIUS アダプタでは、IPv4 ベースと IPv6 ベースの両方のサーバーがサポートされています。

親トピック: ステップ1B: Oracleデータベース・サーバーでのRADIUSの構成

24.4.1.2.3 ステップ1B(3): Oracleデータベース・サーバー初期化パラメータの設定

sqlnet.oraファイルの構成後、init.ora初期化ファイルを構成する必要があります。

1. 次の設定をinit.oraファイルに追加します。

```
OS_AUTHENT_PREFIX=""
```

デフォルトでは、init.oraファイルは、LinuxおよびUNIXシステムの場合はORACLE_HOME/dbsディレクトリ(または同じデータ・ファイルの場所)、Windowsの場合はORACLE_HOME¥databaseディレクトリにあります。

2. データベースを再起動します。

たとえば:

```
SQL> SHUTDOWN
SQL> STARTUP
```

関連項目:

初期化パラメータの設定の詳細は、『Oracle Databaseリファレンス』を参照してください。

親トピック: ステップ1B: Oracleデータベース・サーバーでのRADIUSの構成

24.4.1.3 ステップ1C: その他のRADIUS機能の構成

デフォルト設定を変更し、チャレンジ・レスポンス・モードを構成して、代替RADIUSサーバーのパラメータを設定できます。

- ステップ1C(1): デフォルト設定の変更
  Oracle Net Managerを使用して、デフォルトのRADIUS設定を変更できます。
- ステップ1C(2): チャレンジ・レスポンス・モードの構成
  チャレンジ・レスポンス・モードを構成するには、トークン・カードから取得する動的パスワードなどの情報を指定する必要があります。
- ステップ1C(3): 代替RADIUSサーバーのパラメータの設定
  代替RADIUSサーバーを使用する場合、追加パラメータを設定する必要があります。

親トピック: ステップ1: RADIUS認証の構成

24.4.1.3.1 ステップ1C(1): デフォルト設定の変更

Oracle Net Managerを使用して、デフォルトのRADIUS設定を変更できます。

1. Oracle Net Managerを起動します。
   - (UNIX) $ORACLE_HOME/binから、コマンドラインで次のコマンドを入力します。

```
netmgr
```

- (Windows)「スタート」→「プログラム」→「Oracle - HOME_NAME」→「Configuration and Migration Tools」→「Net Manager」を選択します。

2. 「Oracle Netの構成」を展開し、「ローカル」から「プロファイル」を選択します。
3. 「ネーミング」リストから、「ネットワーク・セキュリティ」を選択します。

   ネットワーク・セキュリティのタブ付きウィンドウが表示されます。
4. 「その他のパラメータ」タブをクリックします。
5. 「認証サービス」リストから、RADIUSを選択します。
6. 次のフィールドについてデフォルト設定を変更します。
   - ポート番号: プライマリRADIUSサーバーのリスニング・ポートを指定します。デフォルト値は1645です。
   - タイムアウト(秒): Oracleデータベース・サーバーがプライマリRADIUSサーバーからの応答を待機する時間を指定します。デフォルトは15秒です。
   - 再試行回数: Oracleデータベース・サーバーがプライマリRADIUSサーバーにメッセージを再送する回数を指定します。デフォルトは3回の再試行です。RADIUSアカウンティングの構成の詳細は、「ステップ4: RADIUSアカウンティングの構成」を参照してください。
   - シークレット・ファイル: Oracleデータベース・サーバー上の秘密キーの場所を指定します。このフィールドは、秘密キー自体ではなく、秘密キー・ファイルの場所を指定します。秘密キーの指定の詳細は、「ステップ1B(1): OracleデータベースでのRADIUS秘密キー・ファイルの作成」を参照してください。
7. 「ファイル」メニューから、「ネットワーク構成の保存」を選択します。

   sqlnet.oraファイルは次のエントリで更新されます。

```
SQLNET.RADIUS_AUTHENTICATION_PORT=(PORT)
SQLNET.RADIUS_AUTHENTICATION_TIMEOUT=(NUMBER OF SECONDS TO WAIT FOR response)
SQLNET.RADIUS_AUTHENTICATION_RETRIES=(NUMBER OF TIMES TO RE-SEND TO RADIUS
server)
SQLNET.RADIUS_SECRET=(path/radius.key)
```

親トピック: ステップ1C: その他のRADIUS機能の構成

24.4.1.3.2 ステップ1C(2): チャレンジ・レスポンス・モードの構成

チャレンジ・レスポンス・モードを構成するには、トークン・カードから取得する動的パスワードなどの情報を指定する必要があります。

RADIUSアダプタでは、このインタフェースはJavaベースであり、最適なプラットフォーム独立性を提供します。

ノート:

認証デバイスのサード・パーティ・ベンダーは、その独自のデバイスに合せてこのグラフィカル・ユーザー・インタフェースをカスタマイズする必要があります。たとえば、スマートカード・ベンダーは Java インタフェースをカスタマイズして、Oracle クライアントがスマートカードから動的パスワードなどのデータを読み取るようにします。スマートカードは、チャレンジを受け取ると、PIN などの追加情報をユーザーに要求することで応答します。

チャレンジ・レスポンス・モードを構成するには:

1. JDK 1.1.7またはJRE 1.1.7を使用する場合は、JAVA_HOME環境変数を、Oracleクライアントが実行されるシステム上のJREまたはJDKの場所に設定します。

   - UNIXでは、プロンプトで次のコマンドを入力します。

     ```
     % setenv JAVA_HOME /usr/local/packages/jre1.1.7B
     ```

   - Windowsでは、「スタート」→「設定」→「コントロール パネル」→「システム」→「環境」の順に選択し、JAVA_HOME変数を次のように設定します。

     ```
     c:¥java¥jre1.1.7B
     ```

   このステップは、他のJDK/JREバージョンでは必要ありません。

2. Oracle Net Managerを起動します。

   - (UNIX) $ORACLE_HOME/binから、コマンドラインで次のコマンドを入力します。

     ```
     netmgr
     ```

   - (Windows)「スタート」→「プログラム」→「Oracle - HOME_NAME」→「Configuration and Migration Tools」→「Net Manager」を選択します。

3. 「Oracle Netの構成」を展開し、「ローカル」から「プロファイル」を選択します。

4. 「ネーミング」リストから、「ネットワーク・セキュリティ」を選択します。

   ネットワーク・セキュリティのタブ付きウィンドウが表示されます。

5. 「認証サービス」リストから、RADIUSを選択します。

6. 「チャレンジ・レスポンス」フィールドで、ONと入力してチャレンジ・レスポンスを有効にします。

7. 「デフォルト・キーワード」フィールドで、チャレンジのデフォルト値を受け入れるか、RADIUSサーバーからチャレンジを要求するためのキーワードを入力します。

   キーワード機能は、Oracleから提供されており、一部のRADIUSサーバーでサポートされています(すべてではありません)。この機能は、RADIUSサーバーでサポートされている場合にのみ使用できます。

   キーワードを設定することで、ユーザーはパスワードを使用しなくても識別情報を証明できます。ユーザーがパスワードを入力しない場合、ここで設定するキーワードがRADIUSサーバーに渡され、RADIUSサーバーは運転免許証の番号や生年月日などを要求するチャレンジで応答します。ユーザーがパスワードを入力する場合、RADIUSサーバーの構成に応じて、RADIUSサーバーはチャレンジで応答する場合と応答しない場合があります。

8. 「インタフェース・クラス名」フィールドで、デフォルト値DefaultRadiusInterfaceを受け入れるか、チャレンジ・レスポンス対話を処理するために作成したクラスの名前を入力します。

   デフォルトのRADIUSインタフェース以外を使用する場合は、sqlnet.oraファイルを編集してSQLNET.RADIUS_CLASSPATH=(location)を入力する必要もあります(locationは、jarファイルの完全なパス名です)。これは、デフォルトでは$ORACLE_HOME/network/jlib/netradius.jar:$ORACLE_HOME/JRE/lib/vt.jarです。

9. 「ファイル」メニューから、「ネットワーク構成の保存」を選択します。

   sqlnet.oraファイルは次のエントリで更新されます。

```
SQLNET.RADIUS_CHALLENGE_RESPONSE=([ON | OFF])
SQLNET.RADIUS_CHALLENGE_KEYWORD=(KEYWORD)
SQLNET.RADIUS_AUTHENTICATION_INTERFACE=(name of interface including the package
name delimited by "/" for ".")
```

関連項目:

チャレンジ・レスポンス・ユーザー・インタフェースをカスタマイズする方法の詳細は、「RADIUSを使用した認証デバイスの統合」を参照してください

親トピック: ステップ1C: その他のRADIUS機能の構成

24.4.1.3.3 ステップ1C(3): 代替RADIUSサーバーのパラメータの設定

代替RADIUSサーバーを使用する場合、追加パラメータを設定する必要があります。

- `sqlnet.ora`ファイルで次のパラメータを設定します。

```
SQLNET.RADIUS_ALTERNATE=(hostname or ip address of alternate radius server)
SQLNET.RADIUS_ALTERNATE_PORT=(1812)
SQLNET.RADIUS_ALTERNATE_TIMEOUT=(number of seconds to wait for response)
SQLNET.RADIUS_ALTERNATE_RETRIES=(number of times to re-send to radius server)
```

親トピック: ステップ1C: その他のRADIUS機能の構成

24.4.2 ステップ2: ユーザーの作成とアクセス権の付与

RADIUS認証の完了後、RADIUS構成を担当するOracle Databaseユーザーを作成する必要があります。

1. CDBルートまたはRADIUSが実装されているPDBに接続します。
   たとえば:

```
CONNECT system@pdb_name;
Enter password: password
```

2. CDBルートに接続している場合は共通ユーザーとしてユーザーを作成し、PDBに接続している場合はローカル・ユーザーとしてユーザーを作成します。

```
CREATE USER username IDENTIFIED EXTERNALLY;
GRANT CREATE SESSION TO USER user_name;
```

3. ユーザー`username`をRADIUSサーバーのusersファイルに入力します。

関連項目:

RADIUSサーバーの管理ドキュメント

親トピック: RADIUS認証、認可およびアカウンティングの有効化

24.4.3 ステップ3: 外部RADIUS認可の構成(オプション)

Oracleデータベースに接続する必要があるRADIUSユーザーに対して、Oracleサーバー、OracleクライアントおよびRADIUSサーバーを構成する必要があります。

- ステップ3A: Oracle Server (RADIUSクライアント)の構成

init.oraファイルを編集して、RADIUSクライアントにOracleサーバーを構成できます。

- ステップ3B: Oracleクライアント(ユーザーがログインする場所)の構成
  次に、ユーザーがログインするOracleクライアントを構成する必要があります。
- ステップ3C: RADIUSサーバーの構成
  RADIUSサーバーを構成するには、RADIUSサーバーの属性構成ファイルを変更する必要があります。

親トピック: RADIUS認証、認可およびアカウンティングの有効化

24.4.3.1 ステップ3A: Oracle Server (RADIUSクライアント)の構成

init.oraファイルを編集して、RADIUSクライアントにOracleサーバーを構成できます。

これを行うには、init.oraファイルを変更して、データベースを再起動し、RADIUSチャレンジ・レスポンス・モードを設定する必要があります。

1. OS_ROLESパラメータをinit.oraファイルに追加し、このパラメータを次のようにTRUEに設定します。

   ```
   OS_ROLES=TRUE
   ```

   デフォルトでは、init.oraファイルは、LinuxおよびUNIXシステムの場合はORACLE_HOME/dbsディレクトリ(または同じデータ・ファイルの場所)、Windowsの場合はORACLE_HOME¥databaseディレクトリにあります。

2. init.oraファイルに対する変更がシステムに反映されるように、データベースを再起動します。

   たとえば:

   ```
   SQL> SHUTDOWN
   SQL> STARTUP
   ```

3. ステップ1C(2): チャレンジ・レスポンス・モードの構成に記載されているステップに従って、サーバーのRADIUSチャレンジ・レスポンス・モードをONに設定していない場合は設定します。

4. 外部で識別されるユーザーおよびロールを追加します。

親トピック: ステップ3: 外部RADIUS認可の構成(オプション)

24.4.3.2 ステップ3B: Oracleクライアント(ユーザーがログインする場所)の構成

次に、ユーザーがログインするOracleクライアントを構成する必要があります。

- ステップ1C(2): チャレンジ・レスポンス・モードの構成に記載されているステップに従って、クライアントのRADIUSチャレンジ・レスポンス・モードをONに設定していない場合は設定します。

親トピック: ステップ3: 外部RADIUS認可の構成(オプション)

24.4.3.3 ステップ3C: RADIUSサーバーの構成

RADIUSサーバーを構成するには、RADIUSサーバーの属性構成ファイルを変更する必要があります。

次の属性をRADIUSサーバー属性構成ファイルに追加します。

属性名	コード	型
VENDOR_SPECIFIC	26	整数

属性名	コード	型
ORACLE_ROLE	1	文字列

2. SMIネットワーク管理プライベート・エンタープライズ・コード111が含まれているRADIUSサーバー属性構成ファイルで、Oracleのベンダー IDを割り当てます。

   たとえば、RADIUSサーバー属性構成ファイルに次のように入力します。

   ```
   VALUE VENDOR_SPECIFIC ORACLE 111
   ```

3. 次の構文を使用して、ORACLE_ROLE属性を外部RADIUS認可を使用するユーザーのユーザー・プロファイルに追加します。

   ```
   ORA_databaseSID_rolename[_[A]|[D]]
   ```

   詳細は、次のとおりです。

   - ORAは、このロールをOracle用に使用することを指定します。
   - databaseSIDは、データベースのinit.oraファイル内で構成されているOracleシステム識別子です。

     デフォルトでは、init.oraファイルは、LinuxおよびUNIXシステムの場合はORACLE_HOME/dbsディレクトリ(または同じデータ・ファイルの場所)、Windowsの場合はORACLE_HOME¥databaseディレクトリにあります。
   - rolenameは、データ・ディクショナリで定義されているロール名です。
   - Aは、ユーザーがこのロールの管理者の権限を持つことを示すオプションの文字です。
   - Dは、このロールをデフォルトで有効にすることを示すオプションの文字です。

   Oracleロールにマップされる RADIUSグループが、ORACLE_ROLEの構文に準拠していることを確認します。

   たとえば:

   ```
   USERNAME      USERPASSWD="user_password",
                 SERVICE_TYPE=login_user,
                 VENDOR_SPECIFIC=ORACLE,
                 ORACLE_ROLE=ORA_ora920_sysdba
   ```

   関連項目:

   サーバーの構成の詳細は、RADIUSサーバーの管理ドキュメントを参照してください。

親トピック: ステップ3: 外部RADIUS認可の構成(オプション)

24.4.4 ステップ4: RADIUSアカウンティングの構成

RADIUSアカウンティングは、Oracleデータベース・サーバーへのアクセスに関する情報をログに記録し、RADIUSアカウンティング・サーバー上のファイルに格納します。

この機能は、RADIUSサーバーと認証サーバーの両方でサポートされている場合にのみ使用します。

- ステップ4A: Oracleデータベース・サーバーでのRADIUSアカウンティングの設定
  RADIUSアカウンティングをサーバーで設定するには、Oracle Net Managerを使用します。
- ステップ4B: RADIUSアカウンティング・サーバーの構成

RADIUSアカウンティング・サーバーは、RADIUS認証サーバーと同じホストまたは別のホストにあります。

親トピック: RADIUS認証、認可およびアカウンティングの有効化

24.4.4.1 ステップ4A: OracleデータベースサーバーでのRADIUSアカウンティングの設定

RADIUSアカウンティングをサーバーで設定するには、Oracle Net Managerを使用します。

1. Oracle Net Managerを起動します。
   - (UNIX) `$ORACLE_HOME/bin`から、コマンドラインで次のコマンドを入力します。
     ```
     netmgr
     ```
   - (Windows)「スタート」→「プログラム」→「Oracle - HOME_NAME」→「Configuration and Migration Tools」→「Net Manager」を選択します。
2. 「Oracle Netの構成」を展開し、「ローカル」から「プロファイル」を選択します。
3. 「ネーミング」リストから、「ネットワーク・セキュリティ」を選択します。

   ネットワーク・セキュリティのタブ付きウィンドウが表示されます。
4. 「その他のパラメータ」タブを選択します。
5. 「認証サービス」リストから、RADIUSを選択します。
6. 「アカウンティングの送信」フィールドで、アカウンティングを有効にするにはONと入力し、無効にするにはOFFと入力します。
7. 「ファイル」メニューから、「ネットワーク構成の保存」を選択します。

   `sqlnet.ora`ファイルが次のエントリで更新されます。
   ```
   SQLNET.RADIUS_SEND_ACCOUNTING= ON
   ```

親トピック: ステップ4: RADIUSアカウンティングの構成

24.4.4.2 Step 4B: RADIUSアカウンティング・サーバーの構成

RADIUSアカウンティング・サーバーは、RADIUS認証サーバーと同じホストまたは別のホストにあります。

- RADIUSアカウンティングの構成の詳細は、RADIUSサーバーの管理ドキュメントを参照してください。

親トピック: ステップ4: RADIUSアカウンティングの構成

24.4.5 ステップ5: RADIUSクライアント名のRADIUSサーバー・データベースへの追加

選択するRADIUSサーバーは、RADIUS標準に準拠している必要があります。

Internet Engineering Task Force (IETF) RFC #2138 Remote Authentication Dial In User Service (RADIUS)およびRFC #2139 RADIUS Accountingの規格に準拠するRADIUSサーバーは、すべて使用できます。RADIUSサーバーには様々なものがあるため、固有の相互運用性要件について、使用するRADIUSサーバーのドキュメントで確認してください。

RADIUSクライアント名をLivingston RADIUSサーバーに追加するには:

1. `/etc/raddb/clients`にあるclientsファイルを開きます。

次のテキストと表が表示されます。

```
@ (#) clients 1.1 2/21/96 Copyright 1991 Livingston Enterprises Inc
This file contains a list of clients which are allowed to make authentication
requests and their encryption key. The first field is a valid hostname. The
second field (separated by blanks or tabs) is the encryption key.
Client Name                     Key
```

2. CLIENT NAME列に、Oracleデータベース・サーバーが実行されているホストのホスト名またはIPアドレスを入力します。

   KEY列に、共有シークレットを入力します。

   CLIENT NAME列に入力する値は、クライアントの名前かIPアドレスかにかかわらず、RADIUSサーバーによって異なります。

3. clientsファイルを保存して閉じます。

関連項目:

RADIUSサーバーの管理ドキュメント

親トピック: RADIUS認証、認可およびアカウンティングの有効化

24.4.6 ステップ6: RADIUSとともに使用する認証サーバーの構成

RADIUSクライアント名をRADIUSサーバー・データベースに追加した後、RADIUSを使用するように認証サーバーを構成できます。

- 認証サーバーの構成の詳細は、認証サーバーのドキュメントを参照してください。

親トピック: RADIUS認証、認可およびアカウンティングの有効化

24.4.7 ステップ7: 認証サーバーとともに使用するRADIUSサーバーの構成

RADIUSで使用するように認証サーバーを構成した後、その認証サーバーを使用するようにRADIUSサーバーを構成できます。

- 認証サーバーとともに使用するRADIUSサーバーの構成の詳細は、RADIUSサーバーのドキュメントを参照してください。

親トピック: RADIUS認証、認可およびアカウンティングの有効化

24.4.8 ステップ8: マッピング・ロールの構成

RADIUSサーバーでベンダー・タイプ属性がサポートされている場合は、ロールをRADIUSサーバーに格納して管理できます。

RADIUSを使用したCONNECT要求があると、Oracleデータベース・サーバーはロールをダウンロードします。この機能を使用するには、Oracleデータベース・サーバーとRADIUSサーバーの両方でロールを構成する必要があります。

1. テキスト・エディタを使用して、Oracleデータベース・サーバーの初期化パラメータ・ファイルでOS_ROLESパラメータを設定します。

   デフォルトでは、init.oraファイルは、LinuxおよびUNIXシステムの場合はORACLE_HOME/dbsディレクトリ(または同じデータ・ファイルの場所)、Windowsの場合はORACLE_HOME¥databaseディレクトリにあります。

2. Oracleデータベース・サーバーを停止して再起動します。

たとえば:

```
SHUTDOWN
STARTUP
```

3. RADIUSサーバーがOracleデータベース・サーバー上で管理する各ロールを、値IDENTIFIED EXTERNALLYを使用して作成します。

   RADIUSサーバーでロールを構成するには、次の構文を使用します。

   ```
   ORA_DatabaseName.DatabaseDomainName_RoleName
   ```

   詳細は、次のとおりです。

   - DatabaseNameは、ロールが作成されているOracleデータベース・サーバーの名前です。これは、DB_NAME初期化パラメータの値と同じです。
   - DatabaseDomainNameは、Oracleデータベース・サーバーが属するドメインの名前です。この値は、DB_DOMAIN初期化パラメータの値と同じです。
   - RoleNameは、Oracleデータベース・サーバーで作成されたロールの名前です。

   たとえば:

   ```
   ORA_USERDB.US.EXAMPLE.COM_MANAGER
   ```

4. RADIUSチャレンジ・レスポンス・モードを構成します。

関連トピック

- チャレンジ・レスポンス(非同期)認証モード
- ステップ1C(2): チャレンジ・レスポンス・モードの構成

親トピック: RADIUS認証、認可およびアカウンティングの有効化

24.5 RADIUSを使用したデータベースへのログイン

RADIUSを使用してデータベースにログインするには、同期認証モードまたはチャレンジ・レスポンス・モードのいずれかを使用します。

- SQL*Plusを起動して、次のいずれかの方法でデータベースにログインします。
  - 同期認証モードを使用する場合は、最初にチャレンジ・レスポンス・モードがONになっていないことを確認して、次のコマンドを入力します。

    ```
    CONNECT username@database_alias
    Enter password: password
    ```

  - チャレンジ・レスポンス・モードを使用する場合は、チャレンジ・レスポンス・モードがONに設定されていることを確認して、次のコマンドを入力します。

    ```
    CONNECT /@database_alias
    ```

ノート:

チャレンジ・レスポンス・モードは、ログインのすべてのケースについて構成できます。

親トピック: RADIUS認証の構成

24.6 RSA ACE/Server構成チェックリスト

RSA ACE/Server RADIUSサーバーを使用する場合は、最初の接続を試みる前に、このサーバーのホスト・エージェントとSecurIDトークンを確認してください。

- ノード・シークレットを送信するようにRSA ACE/Serverのホスト・エージェントが設定されていることを確認します。バージョン5.0では、これを行うには「SENT Node secret」ボックスの選択を解除したままにします。RSA ACE/Serverがエージェントへのノード・シークレットの送信に失敗した場合は、ノード検証失敗メッセージがRSA ACE/Serverのログに書き込まれます。
- RSA SecurIDトークンを使用している場合は、トークンがRSA ACE/Serverと同期されていることを確認します。

関連項目:

トラブルシューティングの詳細は、RSA ACE/Serverのドキュメントを参照してください。

親トピック: RADIUS認証の構成

25 厳密認証の使用のカスタマイズ

Oracle Databaseのネイティブ・ネットワーク暗号化および厳密認証のもとで、複数の認証方法を構成できます。

- 厳密認証を使用したデータベースへの接続
  パスワード認証を使用して、厳密認証を使用するように構成されたデータベースに接続できます。
- 厳密認証およびネイティブ・ネットワーク暗号化の無効化
  Oracle Net Managerを使用して、厳密認証およびネイティブ・ネットワーク暗号化を無効にすることができます。
- 複数の認証方式の構成
  多くのネットワークにおいて、単一のセキュリティ・サーバーで複数の認証方式が使用されています。
- 外部認証のためのOracle Databaseの構成
  パラメータを使用して、ネットワーク認証を使用するようにOracle Databaseを構成できます。

親トピック: 厳密認証の管理

25.1 厳密認証を使用したデータベースへの接続

パスワード認証を使用して、厳密認証を使用するように構成されたデータベースに接続できます。

1. Oracleネットワークおよび厳密認証方式が構成されている場合に、ユーザー名とパスワードを使用してOracleデータベース・サーバーに接続するには、外部認証を無効化します。

   Oracleネットワークおよび厳密認証方式が構成されている場合に、ユーザー名とパスワードを使用してOracleデータベース・サーバーに接続する前に、まず「厳密認証およびネイティブ・ネットワーク暗号化の無効化」の手順に従って、外部認証を無効化する必要があります。

2. 外部認証を無効化して、次の形式を使用してデータベースに接続します。

   ```
   % sqlplus username@net_service_name
   Enter password: password
   ```

   たとえば:

   ```
   % sqlplus hr@emp
   Enter password: password
   ```

ノート:

単一データベースに、外部認証ユーザーとパスワード認証ユーザーの両方を含む複数の認証方式を構成できます。

親トピック: 厳密認証の使用のカスタマイズ

25.2 厳密認証およびネイティブ・ネットワーク暗号化の無効化

Oracle Net Managerを使用して、厳密認証およびネイティブ・ネットワーク暗号化を無効にすることができます。

1. Oracle Net Managerを起動します。
   - (UNIX) `$ORACLE_HOME/bin`から、コマンドラインで次のコマンドを入力します。

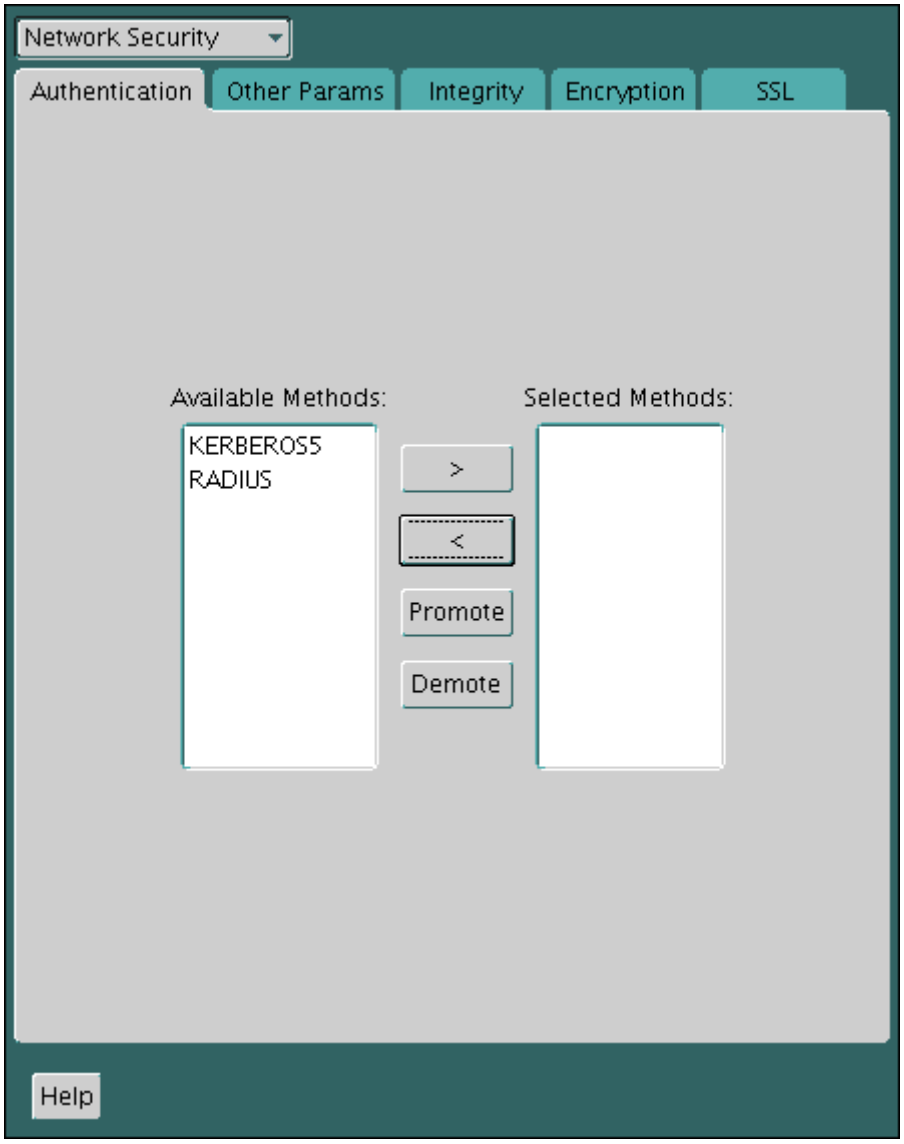
     ```
     netmgr
     ```

- (Windows)「スタート」→「プログラム」→「Oracle - HOME_NAME」→「Configuration and Migration Tools」→「Net Manager」を選択します。

2. 「Oracle Netの構成」を展開し、「ローカル」から「プロファイル」を選択します。
3. 「ネーミング」リストから、「ネットワーク・セキュリティ」を選択します。

   ネットワーク・セキュリティのタブ付きウィンドウが表示されます。

4. 「認証」タブ(デフォルトで選択されています)を選択します。
5. 「選択メソッド」リスト内の認証方式を順番に選択して左矢印[<]をクリックすることによって「使用可能なメソッド」リストにすべて移動します。



6. 「暗号化」タブを選択します。
7. 次を実行します。
   - 「暗号化」メニューから「SERVER」を選択します。
   - 「暗号化タイプ」を「拒否」に設定します。
   - 暗号化シードが使用された場合は、「暗号化シード」フィールドに有効な暗号化シードを入力します。
   - 「選択メソッド」の下の任意のメソッドを「使用可能なメソッド」フィールドに移動します。
8. クライアントのネイティブ・ネットワーク暗号化を無効にするには、「暗号化」メニューからCLIENTを選択して、前述のステップを繰り返します。
9. 「ファイル」メニューから、「ネットワーク構成の保存」を選択します。

sqlnet.oraファイルが次のエントリで更新され、厳密認証およびネイティブ・ネットワーク暗号化が無効であることを示します。

厳密認証:

```
SQLNET.AUTHENTICATION_SERVICES = (NONE)
```

ローカル・データベース・パスワード認証を使用している場合は、クライアントで SQLNET.AUTHENTICATION_SERVICES=(NONE)を設定することもできます。この設定により、クライアントのパフォーマンスが向上します。

ネイティブ・ネットワーク暗号化については、サーバー側とクライアント側で個別に設定できます。次に、サーバーとクライアントの両方でネイティブ・ネットワーク暗号化を無効にする例を示します。

```
SQLNET.ENCRYPTION_SERVER = REJECTED
SQLNET.ENCRYPTION_CLIENT = REJECTED
```

マルチテナント環境では、sqlnet.oraファイルの設定はすべてのプラガブル・データベース(PDB)に適用されることに注意してください。

関連トピック

- 暗号化および整合性のネゴシエーションの値について

親トピック: 厳密認証の使用のカスタマイズ

25.3 複数の認証方式の構成

多くのネットワークにおいて、単一のセキュリティ・サーバーで複数の認証方式が使用されています。

そのため、Oracle Databaseでは、Oracleクライアントが特定の認証方式を使用できるように、および、Oracleデータベース・サーバーが指定された任意の方式を受け入れることができるように、ネットワークを構成できます。

Oracle Net Managerを使用するか、または任意のテキスト・エディタを使用してsqlnet.oraファイルを変更することによって、クライアント・システムとサーバー・システムの両方で複数の認証方式を設定できます。クライアントとサーバーの両方に認証方式を追加するには、Oracle Net Managerを使用します。

1. Oracle Net Managerを起動します。
   - (UNIX) $ORACLE_HOME/binから、コマンドラインで次のコマンドを入力します。

     ```
     netmgr
     ```

   - (Windows)「スタート」→「プログラム」→「Oracle - HOME_NAME」→「Configuration and Migration Tools」→「Net Manager」を選択します。
2. 「Oracle Netの構成」を展開し、「ローカル」から「プロファイル」を選択します。
3. 「ネーミング」リストから、「ネットワーク・セキュリティ」を選択します。

   ネットワーク・セキュリティのタブ付きウィンドウが表示されます。
4. 「認証」タブを選択します。
5. 「使用可能なメソッド」リストに一覧表示されている方式を選択します。
6. 右矢印(>)をクリックして、選択した方式を順番に「選択メソッド」リストに移動します。
7. 選択した方式を目的の利用順に並べます。

   そのためには、「選択メソッド」リストで方式を選択し、「上へ」または「下へ」を選択してリスト内での位置を変更します。

8. 「ファイル」メニューから、「ネットワーク構成の保存」を選択します。

   sqlnet.oraファイルが次のエントリで更新され、選択した認証方式が一覧表示されます。

```
SQLNET.AUTHENTICATION_SERVICES = (KERBEROS5, RADIUS)
```

ノート:

SecurID 機能は RADIUS によって提供されます。RADIUS サポートは RSA ACE/Server に組み込まれています。

関連トピック

- RADIUS認証の構成

親トピック: 厳密認証の使用のカスタマイズ

25.4 外部認証のためのOracle Databaseの構成

パラメータを使用して、ネットワーク認証を使用するようにOracle Databaseを構成できます。

- sqlnet.oraでのSQLNET.AUTHENTICATION_SERVICESパラメータの設定
  SQLNET.AUTHENTICATION_SERVICESパラメータでは、使用する認証方式とバージョンを定義します。
- OS_AUTHENT_PREFIXのNull値への設定
  OS_AUTHENT_PREFIXパラメータでは、Oracle Databaseがサーバーに接続しようとするユーザーの認証に使用する接頭辞を指定します。

親トピック: 厳密認証の使用のカスタマイズ

25.4.1 sqlnet.oraでのSQLNET.AUTHENTICATION_SERVICESパラメータの設定

SQLNET.AUTHENTICATION_SERVICESパラメータでは、使用する認証方式とバージョンを定義します。

すべてのクライアントおよびサーバーで、サポートされている認証方式をそれぞれが使用できるようにするには、sqlnet.oraファイルでSQLNET.AUTHENTICATION_SERVICESパラメータを設定する必要があります。

- 次の構文を使用して、SQLNET.AUTHENTICATION_SERVICESパラメータを設定します。

```
SQLNET.AUTHENTICATION_SERVICES=(oracle_authentication_method)
```

   たとえば、すべてのクライアントとサーバーがKerberos認証を使用する場合は次のようにします。

```
SQLNET.AUTHENTICATION_SERVICES=(KERBEROS5)
```

   デフォルトでは、sqlnet.oraファイルは、ORACLE_HOME/network/adminディレクトリ、またはTNS_ADMIN環境変数によって設定されている場所にあります。TNS_ADMIN変数が正しいsqlnet.oraファイルを指定するように適切に設定されていることを確認します。

   ローカル・データベース・パスワード認証のみを使用している場合は、クライアント・パフォーマンスを向上させるためにSQLNET.AUTHENTICATION_SERVICESを次のように設定します。

```
SQLNET.AUTHENTICATION_SERVICES=(NONE)
```

関連トピック

- SQL*Plusユーザーズ・ガイドおよびリファレンス

親トピック: 外部認証のためのOracle Databaseの構成

25.4.2 OS_AUTHENT_PREFIXのNull値への設定

OS_AUTHENT_PREFIXパラメータでは、Oracle Databaseがサーバーに接続しようとするユーザーの認証に使用する接頭辞を指定します。

認証サービスベースのユーザー名は長くてもかまいませんが、Oracleユーザー名は128バイトまでに制限されています。OS_AUTHENT_PREFIXパラメータをnull値に設定することを強くお薦めします。

- データベース・インスタンスの初期化ファイルで、OS_AUTHENT_PREFIXを次のように設定します。

  ```
  OS_AUTHENT_PREFIX=""
  ```

  次のことに注意してください。

  - OS_AUTHENT_PREFIXのデフォルト値はOPS$ですが、任意の文字列に設定できます。
  - データベースでOS_AUTHENT_PREFIXの値がすでにNULL (" ")以外に設定されている場合は、変更しないでください(変更すると、以前に作成され、外部で識別されたユーザーがOracleサーバーに接続できなくなる可能性があります)。

  OS_AUTHENT_PREFIXをnullに設定した後、次の構文を使用して外部ユーザーを作成できます。

  ```
  CREATE USER os_authent_prefix_username IDENTIFIED EXTERNALLY;
  ```

  たとえば、ユーザーkingを作成するとします。

  ```
  CREATE USER king IDENTIFIED EXTERNALLY;
  ```

  このようにユーザーを作成すると、外部で識別されたユーザーの様々なユーザー名の管理が不要になるという利点があります。これは、サポートされているすべての認証方式に当てはまります。

親トピック: 外部認証のためのOracle Databaseの構成

第VI部 監査を使用したデータベース・アクティビティの管理

第VI部では、監査を使用したデータベース・アクティビティの監視方法について説明します。

- 監査の概要
  特権ユーザーは、他の特権ユーザーを含むすべてのユーザーがデータベースで行った変更を追跡するポリシーを作成できます。
- 監査ポリシーの構成
  統合監査では、カスタム統合監査ポリシー、事前定義の統合監査ポリシーおよびファイングレイン監査がサポートされます。
- 監査証跡の管理
  AUDIT_ADMINロールを付与されているユーザーは、監査証跡の管理、監査証跡のアーカイブおよび監査証跡レコードの削除を実行できます。

26 監査の概要

特権ユーザーは、他の特権ユーザーを含むすべてのユーザーがデータベースで行った変更を追跡するポリシーを作成できます。

ノート:

特に断りのないかぎり、この章では、すべての監査レコードが一元管理される完全な統合監査の使用方法について説明します。

- 監査とは
  監査とは、データベース・アクティビティ(データベース・ユーザーと非データベース・ユーザー両方のアクティビティ)を監視して記録することです。
- 監査を使用する理由
  通常は、監査を使用してユーザー・アクティビティを監視します。
- 監査のベスト・プラクティス
  監査のベスト・プラクティスに関するガイドラインに従ってください。
- 統合監査とは
  統合監査では、統合監査証跡により、次の各種ソースから監査情報が取得されます。
- 統合監査証跡の利点
  統合監査証跡には、次のように多くの利点があります。
- データベースが統合監査に移行したかどうかの確認
  V$OPTION動的ビューは、データベースが統合監査に移行されたかどうかを示します。
- 混合モードの監査
  混合モードの監査は、新しくインストールしたデータベースでのデフォルトの監査です。
- 監査の実行者
  Oracleでは監査を実行するユーザー用にAUDIT_ADMINとAUDIT_VIEWERという2つのロールが用意されています。
- マルチテナント環境での統合監査
  ポリシーのタイプに応じて、個々のPDBに、またはCDBに監査設定を適用できます。
- 分散データベースでの監査
  データベース・インスタンスでは直接接続しているユーザーによって発行された文のみが監査されるため、監査はサイトで自律的に実行されるといえます。

関連トピック

- 監査のガイドライン

親トピック: 監査を使用したデータベース・アクティビティの監視

26.1 監査とは

監査とは、データベース・アクティビティ(データベース・ユーザーと非データベース・ユーザー両方のアクティビティ)を監視して記録することです。

非データベース・ユーザーとは、CLIENT_IDENTIFIER属性を使用してデータベースで認識されるアプリケーション・ユーザーのことです。ユーザーのこの対応を監査する場合は、統合監査ポリシー条件、ファイングレイン監査ポリシー、またはOracle

Database Real Application Securityを使用できます。

このガイドでは、ファイングレイン監査やOracle Database Vaultなどの様々なOracle Databaseコンポーネントの監査証跡を、統合監査を使用して1つの統合監査証跡にまとめるポリシーを作成する方法について説明します。この監査証跡は、UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューで表示できます。(AUDIT_UNIFIED_POLICIESなどの、その他の統合監査証跡ビューを使用できます。)統合監査データ証跡によって、分析を実行する前にまず監査データを1つの場所に集めることなく、1つの操作で監査データ全体の分析レポートを実行できます。Oracle Audit Vaultなどの監査マイニング・ツールは、監査レコードを収集するために複数の場所ではなく1つの場所を参照できます。統合監査証跡は、監査情報が一貫してフォーマットされ、一貫したフィールドが含まれていることを確認します。

または、Oracle Databaseリリース11.2の『Oracle Databaseセキュリティ・ガイド』で説明されている、従来の監査も使用できます。

監査は、実行されたSQL文のタイプなどの個々のアクションに基づいて、またはユーザー名、アプリケーション、時間などを含めることができるセッション・メタデータの組合せに基づいて実行できます。

成功したアクティビティと失敗したアクティビティの両方の監査が可能で、特定のユーザーを監査対象に含めたり、除外したりできます。マルチテナント環境では、プラガブル・データベース(PDB)の個々のアクション、またはマルチテナント・コンテナ・データベース(CDB)全体の個々のアクションを監査できます。データベースで提供される標準のアクティビティの監査に加えて、監査には、Oracle Database Real Application Security、Oracle Recovery Manager、Oracle Data Pump、Oracle Data Mining、Oracle Database Vault、Oracle Label SecurityおよびOracle SQL*Loaderダイレクト・パス・イベントからのアクティビティを含めることができます。

監査はデフォルトで有効です。すべての監査レコードは、同一の形式で統合監査証跡に書き込まれ、UNIFIED_AUDIT_TRAILビューから使用できます。これらのレコードはAUDSYSスキーマにあります。監査レコードは、デフォルトでSYSAUX表領域に格納されます。統合監査証跡用に異なる表領域を構成することをお薦めします。これを行うには、DBMS_AUDIT_MGMT.SET_AUDIT_TRAIL_LOCATIONプロシージャを使用します。Oracle Database Standard EditionおよびExpress Edition (Enterprise Editionは除く)では、統合監査の表領域の関連付けは1回のみであることに注意してください。この関連付けは、統合監査証跡の監査レコードを生成する前に行う必要があります。表領域を関連付けると、Enterprise Editionではパーティション化のみがサポートされるため、表領域を変更できません。

次のいずれかの方法を使用して監査を構成できます。

- 監査設定を1つの統合監査ポリシーにグループ化する。データベースで必要なすべての監査設定を定義する統合監査ポリシーを1つ以上作成できます。これを実行する方法は、統合監査ポリシーおよびAUDIT文を使用したアクティビティの監査に説明されています。
- デフォルトの統合監査ポリシーのいずれかを使用する。Oracle Databaseには、ほとんどの規制機関で必要とされる標準の監査設定を網羅した事前定義の統合監査ポリシーが提供されています。事前定義の統合監査ポリシーを使用したアクティビティの監査を参照してください。
- ファイングレイン監査ポリシーを作成する。アクションの発生時刻などのデータを取得するファイングレイン監査ポリシーを作成できます。ファイングレイン監査を使用した特定のアクティビティの監査を参照してください。

データベースを監査することをお薦めします。監査は、強固な内部制御を規定する有効な方法であるため、サイトでは米国サーベンス・オクスリー法(Sarbanes-Oxley Act)に定義されている法令順守要件を満たすことができます。監査を使用すると、ビジネス操作を監視でき、企業ポリシーから逸脱する可能性があるアクティビティを検出できます。この結果、データベースおよびアプリケーション・ソフトウェアへのアクセスが厳密に制御されるようになり、パッチが予定どおりに適用され、非定型の変更が防止されます。有効な監査ポリシーを作成することで、監査および個人のコンプライアンスに関する監査レコードを生成できます。選択的に監査を行い、ビジネス・コンプライアンスのニーズを満たしていることを確認してください。

親トピック: 監査の概要

26.2 監査を使用する理由

通常は、監査を使用してユーザー・アクティビティを監視します。

監査を使用すると、次の内容が可能になります。

- アクションに対するアカウンタビリティの有効化。特定のスキーマ、表または行に対して実行されるアクション、あるいは特定の内容に影響を与えるアクションなどがあります。
- それぞれのアカウンタビリティに基づいた、ユーザー(または侵入者などの他者)による不適切なアクションの防止。
- 疑わしいアクティビティの調査。たとえば、ユーザーが表からデータを削除しようとした場合、セキュリティ管理者は、そのデータベースへのすべての接続と、そのデータベースにあるすべての表からの行の削除(成功および失敗)をすべて監査できます。
- 認可されていないユーザーのアクションを監査人に通知します。たとえば、認可されていないユーザーがデータを変更または削除を実行できるなど、予期した以上の権限を持っている場合に、ユーザー認可を再評価できます。
- インシデント発生後の調査をサポートします。
- 特定のデータベース・アクティビティに関するデータの監視と収集。たとえば、データベース管理者は、更新された表、実行された論理I/Oの回数、またはピーク時に接続していた同時実行ユーザーの数などに関する統計を収集できます。
- 認可またはアクセス制御の実装に関する問題の検出。たとえば、データは他の方法で保護されているため、監査レコードは生成されないと予測される監査ポリシーを作成できます。しかし、これらのポリシーで監査レコードが生成された場合は、他のセキュリティ制御が正しく実装されていないことがわかります。
- コンプライアンスのための監査要件への対処。次のような法規に、監査に関連する一般的な要件が含まれています。
    - 米国サーベンス・オクスリー法
    - 米国の医療保険の相互運用性と説明責任に関する法律(Health Insurance Portability and Accountability Act、HIPAA)
    - 自己資本の測定と基準に関する国際的統一化: 改訂された枠組(バーゼルII)(International Convergence of Capital Measurement and Capital Standards: a Revised Framework、Basel II)
    - 日本の個人情報保護法
    - 欧州連合のプライバシと電子通信に関する指令(European Union Directive on Privacy and Electronic Communications)

親トピック: 監査の概要

26.3 監査のベスト・プラクティス

ベスト・プラクティスに関するガイドラインに従ってください。

- 原則として、法令順守要件を満たすために必要な量の情報を収集するように監査方針を策定しますが、大きなセキュリティ問題の原因となるアクティビティに焦点を合せてください。たとえば、データベース内のすべての表を監査するのではなく、給与など機密性の高いデータが入った表列を監査するのが実用的です。統合監査とファイングレイン監査の両方により、監査対象の特定のアクティビティに焦点を合せた監査ポリシーの策定に使用できるメカニズムがあります。

- 監査証跡データを定期的にアーカイブして削除します。DBMS_AUDIT_MGMTパッケージを使用して、複数の方法で監査レコードを削除できます。収集された監査レコードを定期的に確認し、サイトの保存ポリシーに基づいて監査レコードの収集および保存体系を確立する必要があります。DBMS_AUDIT_MGMTに加えて、Oracle Data SafeおよびOracle Audit Vault and Database Firewallには、監査証跡データのアーカイブおよび削除を管理するための機能が用意されています。

関連トピック

- 監査のガイドライン
- 監査証跡レコードの削除

親トピック: 監査の概要

26.4 統合監査とは

統合監査では、統合監査証跡により、次の各種ソースから監査情報が取得されます。

統合監査では、次のソースから監査レコードを取得できます。

- 統合監査ポリシーおよびAUDIT設定による監査レコード(SYS監査レコードを含む)
- DBMS_FGA PL/SQLパッケージによるファイングレイン監査レコード
- Oracle Database Real Application Security監査レコード
- Oracle Recovery Manager監査レコード
- Oracle Database Vault監査レコード
- Oracle Label Security監査レコード
- Oracle Data Miningのレコード
- Oracle Data Pump
- Oracle SQL*Loaderダイレクト・ロード

統合監査証跡は、SYSAUX表領域のAUDSYSスキーマの読取り専用の表にあり、この情報をUNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューで同一の形式で使用できるようにし、単一インスタンスおよびOracle Database Real Application Clustersの両方の環境で使用できます。ユーザーSYSに加えて、AUDIT_ADMINおよびAUDIT_VIEWERロールが付与されているユーザーもこれらのビューを問い合せることができます。ユーザーが監査ポリシーの作成ではなく、ビューの問合せのみを必要としている場合は、AUDIT_VIEWERロールを付与します。

データベースが書込み可能な場合、監査レコードは統合監査証跡に書き込まれます。データベースが書込み可能でない場合、監査レコードは$ORACLE_BASE/audit/$ORACLE_SIDディレクトリの新しい形式のオペレーティング・システム・ファイルに書き込まれます。

関連項目:

UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューの詳細は、『Oracle Databaseリファレンス』を参照してください。

親トピック: 監査の概要

26.5 統合監査証跡の利点

統合監査証跡には、次のように多くの利点があります。

たとえば:

- 統合監査の有効化後、以前のリリースで使用されていた初期化パラメータには依存しません。これらの初期化パラメータのリストは、表G-1を参照してください。
- SYS監査証跡のレコードを含む、Oracle Databaseインストールのすべての監査コンポーネントの監査レコードは、1つの場所に1つの形式で配置されるため、様々な場所を参照して各種形式の監査証跡を探す必要はありません。この統合ビューにより、監査者は様々なコンポーネントから監査情報を相互に関連付けることができます。たとえば、INSERT文の実行中にエラーが発生した場合は、標準の監査でエラー番号と実行されたSQLを示すことができます。Oracle Database Vault固有の情報では、このエラーがコマンド・ルール違反またはレルム違反のどちらの原因で発生したかを示すことができます。別のAUDIT_TYPEの監査レコードが2つあります。この統合を適切に設定すると、AUDIT_TYPEがStandard Auditに設定されたSYS監査レコードが表示されます。
- 監査証跡の管理とセキュリティは、監査証跡を単一にすることでも改善されます。
- 監査の全体的なパフォーマンスが大幅に向上します。デフォルトでは、監査レコードはAUDSYSスキーマの内部リレーショナル表に自動的に書き込まれます。
- SYS管理ユーザーだけでなく、サポートされているコンポーネント(この項の最初に記載)を各自で監査できる名前付きの監査ポリシーを作成できます。さらに、条件と除外を各自のポリシーに作成することもできます。
- Oracle Audit Vault and Database Firewall環境を使用している場合は、このデータがすべて1つの場所から取得されるため、統合監査証跡により、監査データの収集がかなり容易になります。

> ノート:
>
> 以前のリリースでは、ユーザーは、追加の権限なしで、各自のスキーマのオブジェクトへの監査構成の追加と削除が許可されていました。この機能は使用できなくなりました。

親トピック: 監査の概要

26.6 データベースが統合監査に移行したかどうかの確認

V$OPTION動的ビューは、データベースが統合監査に移行されたかどうかを示します。

- 次に示すようにUnified Auditingを入力して、V$OPTION動的ビューのVALUE列を問い合せます。

```
SELECT VALUE FROM V$OPTION WHERE PARAMETER = 'Unified Auditing';
PARAMETER          VALUE
-----------------  ----------
Unified Auditing   TRUE
```

この出力には、統合監査が有効であることが表示されます。統合監査が有効でない場合、出力はFALSEになります。

関連トピック

- 統合監査の無効化

親トピック: 監査の概要

26.7 混合モードの監査

混合モードの監査は、新しくインストールしたデータベースでのデフォルトの監査です。

- 混合モードの監査について
  混合モード監査では、従来の監査機能(リリース12cより前のリリースの監査機能)と新しい監査機能(統合監査)の両方を使用できます。
- 統合監査の有効化
  Oracle Databaseのエディションによってはデフォルトで、混合モード監査が使用され、統合監査と従来の監査の両方がサポートされます。
- 有効にした監査のタイプがデータベースの作成でどのように決定されるか
  統合監査では、オペレーティング・システム・ファイルの場所として、`$ORACLE_BASE/audit`ディレクトリが使用されます。
- 混合モードの監査の機能
  混合モードの監査には、いくつかの機能が用意されています。

親トピック: 監査の概要

26.7.1 混合モードの監査について

混合モード監査では、従来の監査機能(リリース12cより前のリリースの監査機能)と新しい監査機能(統合監査)の両方を使用できます。

新しいデータベースを作成すると、デフォルトで混合モード監査が使用されます。

データベースは2つのモード(混合モード監査または完全な統合監査モード)のいずれでも有効にできます。これらのどちらのモードでも統合監査の機能は有効になりますが、両者に違いがあります。混合モードでは、新しい統合監査機能を従来の監査機能と並行して使用できます。完全な統合監査では、統合監査機能のみ使用します。

ノート:

Oracle Database リリース 21 c 以降では、従来の監査は非推奨です。かわりに統合監査を使用することをお薦めします。

表26-1に、これらの2つのモードの機能とその有効化の方法のサマリーを示します。

表26-1 混合モード監査と完全な統合監査の違い

モード	機能	有効化する方法
混合モードの監査	従来の監査と統合監査の両方	任意の統合監査ポリシーを有効にします。データベースを再起動する必要はありません。
完全な統合監査	統合監査のみ	`oracle` バイナリを `uniaud_on` にリンクし、データベースを再起動します。完全な統合監査を有効にする方法は、『Oracle Database アップグレード・ガイド』に記載されています。

混合モードは統合監査を導入するためのもので、その機能や特徴、利点を知ることができます。混合モードで、統合監査を使用するように既存のアプリケーションおよびスクリプトを移行することができます。完全な統合監査を使用することにしたら、統合監査オプションを有効にしてoracleバイナリを再リンクし、統合監査をOracleデータベースで実行する唯一の監査機能として有効にします。混合モードに戻す必要がある場合は戻すことができます。

以前のリリース同様、従来の監査機能は、AUDIT_TRAIL初期化パラメータの影響を受けます。混合モードの監査の場合のみ、このパラメータを対応する従来の監査証跡に設定する必要があります。この従来の監査証跡には、統合監査証跡とともに、監査レコードが移入されます。

データベースを現在のリリースにアップグレードすると、従来の監査が保存され、新規監査レコードが従来の監査証跡に書き込まれます。移行の完了後も、以前のリリースの監査レコードはこれらの監査証跡で使用可能です。企業の保存ポリシーに基づき、DBMS_AUDIT_MGMT PL/SQLプロシージャを使用して、これらの古い監査証跡をアーカイブして削除できます。

関連項目:

- 移行前と移行後の監査環境での機能の可用性の比較は、統合監査の移行による各監査機能への影響を参照してください
- データベースが統合監査に移行したかどうかの確認
- データベースを統合監査に移行する方法と、移行しない場合に使用するドキュメントは、『Oracle Databaseアップグレード・ガイド』を参照してください。

親トピック: 混合モードの監査

26.7.2 統合監査の有効化

Oracle Databaseのエディションによってはデフォルトで、混合モード監査が使用され、統合監査と従来の監査の両方がサポートされます。

完全な統合監査モード(監査パフォーマンスを向上させる)に移行する準備が整った後は、*Oracle Database*アップグレード・ガイドの説明に従って、oracleバイナリをuniaud_onとリンクし、データベースを再起動します。

関連トピック

- データベースが統合監査に移行したかどうかの確認
- Oracle Databaseアップグレード・ガイド

親トピック: 混合モードの監査

26.7.3 有効にした監査のタイプがデータベースの作成でどのように決定されるか

統合監査では、オペレーティング・システム・ファイルの場所として、$ORACLE_BASE/auditディレクトリが使用されます。

新しく作成されたデータベースでは、事前定義のポリシーORA_SECURECONFIGにより、混合モードの監査がデフォルトで有効になります。

統合監査の使用を開始するには、少なくとも1つの統合監査ポリシーを有効にし、その使用を停止するには、すべての統合監査ポリシーを無効にする必要があります。

関連トピック

- セキュア・オプションの事前定義の統合監査ポリシー

親トピック: 混合モードの監査

26.7.4 混合モードの監査の機能

混合モードの監査には、いくつかの機能が用意されています。

これらの機能は次のとおりです。

- 既存の監査初期化パラメータのAUDIT_TRAIL、AUDIT_FILE_DEST、AUDIT_SYS_OPERATIONSおよびAUDIT_SYSLOG_LEVELをすべて使用できるようにします。
- 従来の監査証跡に必須の監査レコードのみを書き込みます。
- 標準の監査レコードを標準の監査構成に基づくようにし、AUDIT_TRAIL初期化パラメータで指定された監査証跡にこれらのレコードを書き込みます。

  ただし。標準の監査証跡レコードは統合監査ポリシーに基づいても生成され、これらの監査レコードのみが統合監査証跡に書き込まれることに注意してください。統合監査ポリシーの結果として生成された標準の監査レコードは、統合監査ポリシーの有効化のセマンティクスに従います。
- 管理ユーザー・セッションにより、SYS監査レコードが生成されます。これらのレコードは、AUDIT_SYS_OPERATIONS初期化パラメータがTRUEに設定されている場合に書き込まれます。このプロセスでは、レコードは従来の監査証跡のみに書き込まれます。ただし、統合監査ポリシーが管理ユーザーで有効な場合は、これらの統合監査レコードも統合監査証跡に書き込まれます。
- 従来の監査証跡に書き込まれる監査レコードの形式は、Oracle Database 11gリリース2と同じままです。
- Oracle Databaseでは、統合監査レコードをAUDSYSスキーマの内部リレーショナル表に即座に書き込みます。
- 監査レコードの書込みのパフォーマンス・コストは、監査レコードを生成して従来の監査証跡と統合監査証跡に書き込むのに必要な総時間と同等です。
- 混合モードの監査は、統合監査モードの機能の一部です。新しい形式の監査ポリシーと監査証跡に問題ない場合は、統合監査モードに移行することをお薦めします。

関連トピック

- AUDSYSスキーマへの統合監査証跡レコードの書込み
- Oracle Databaseアップグレード・ガイド

親トピック: 混合モードの監査

26.8 監査の実行者

Oracleでは監査を実行するユーザー用にAUDIT_ADMINとAUDIT_VIEWERという2つのロールが用意されています。

これらのロールによって提供される権限を次に示します。

- AUDIT_ADMINロール。このロールでは、統合およびファイングレイン監査ポリシーの作成、AUDITおよびNOAUDIT SQL文の使用、監査データの表示、および監査証跡の管理が可能です。このロールは、信頼できるユーザーにのみ付与します。
- AUDIT_VIEWERロール。このロールでは、ユーザーによる監査データの表示と分析が可能です。これは、DBMS_AUDIT_UTIL PL/SQLパッケージに対するEXECUTE権限を提供します。このロールが必要なユーザーの種類は、通常は外部監査者です。

監査ポリシーを変更するか、監査証跡を変更する(古い監査データの削除を含む)には、AUDIT_ADMINロールが付与されている必要があります。監査者は、AUDIT_VIEWERロールが付与された後に監査データを表示できます。

親トピック: 監査の概要

26.9 マルチテナント環境での統合監査

ポリシーのタイプに応じて、各PDBまたはCDBに監査設定を適用できます。

ルートを含め、各PDBには、それ固有の統合監査証跡があります。

- CREATE AUDIT POLICYおよびAUDIT文で作成された統合監査ポリシー: ルートと個々のPDBの両方のポリシーを作成できます。
- SYSLOGに書き込まれた監査レコード: UNIXプラットフォームでは、CDBルートでUNIFIED_AUDIT_COMMON_SYSTEMLOG初期化パラメータを設定して、特定の統合監査証跡列がSYSLOGに書き込まれるようにできます。WindowsとUNIXの両方で、UNIFIED_AUDIT_SYSTEMLOGパラメータをルートとPDBレベルの両方で設定できます。
- ファイングレイン監査ポリシー: ルートではなく、個々のPDBのポリシーのみを作成できます。
- 監査証跡の削除: ルートと個々のPDBの両方に対して、削除操作を実行できます。

関連トピック

- マルチテナント環境での統合監査ポリシーまたはAUDIT設定
- ファイングレイン監査ポリシーの作成
- 監査証跡レコードの削除

親トピック: 監査の概要

26.10 分散データベースでの監査

データベース・インスタンスでは直接接続しているユーザーによって発行された文のみが監査されるため、監査はサイトで自律的に実行されるといえます。

ローカルのOracle Databaseノードでは、リモート・データベースで発生するアクションを監査できません。

親トピック: 監査の概要

27 監査ポリシーの構成

統合監査では、カスタム統合監査ポリシー、事前定義の統合監査ポリシーおよびファイングレイン監査がサポートされます。

- 監査タイプの選択
  一般的なアクティビティ(SQL文のアクションなど)や一般的に使用される監査アクティビティ、ファイングレイン監査のシナリオを監査できます。
- 統合監査ポリシーおよびAUDIT文を使用したアクティビティの監査
  CREATE AUDIT POLICY文とAUDIT文を使用して、統合監査ポリシーを使用できます。
- 事前定義の統合監査ポリシーを使用したアクティビティの監査
  Oracle Databaseには、よく使用されるセキュリティ関連の監査設定を対象とする、事前定義の統合監査ポリシーがあります。
- ファイングレイン監査を使用した特定のアクティビティの監査
  ファイングレイン監査では、非常に詳細なレベルで監査ポリシーを作成できます。
- 監査ポリシーのデータ・ディクショナリ・ビュー
  データ・ディクショナリ・ビューおよび動的ビューを使用して詳細な監査情報を入手できます。

親トピック: 監査を使用したデータベース・アクティビティの監視

27.1 監査タイプの選択

一般的なアクティビティ(SQL文のアクションなど)や一般的に使用される監査アクティビティ、ファイングレイン監査のシナリオを監査できます。

- SQL文、権限および他の一般アクティビティの監査
  SQL文からOracle Database Vaultなどの他のOracle Databaseコンポーネントまで、様々なタイプのオブジェクトを監査できます。
- 一般的に使用されるセキュリティ関連アクティビティの監査
  Oracle Databaseには、デフォルトの統合監査ポリシーが用意されており、これらは一般的に使用されるセキュリティ関連の監査用に選択できます。
- 特定のファイングレイン・アクティビティの監査
  個々の列を監査またはイベント・ハンドラを使用する場合は、ファイングレイン監査を使用します。

親トピック: 監査ポリシーの構成

27.1.1 SQL文、権限および他の一般アクティビティの監査

SQL文やOracle Database VaultなどのOracle Databaseコンポーネントまで、様々なタイプのオブジェクトを監査できます。

また、条件を使用するポリシーも作成できます。ただし、特定の列を監査したり、イベント・ハンドラを使用する場合は、ファイングレイン監査を使用する必要があります。

このタイプの監査を実行する一般的なステップは次のとおりです。

1. ほとんどの場合は、CREATE AUDIT POLICY文を使用して監査ポリシーを作成します。アプリケーション・コンテキスト値を監査する必要がある場合は、AUDIT文を使用します。

   統合監査ポリシーおよびAUDIT文を使用したアクティビティの監査の関連するカテゴリを参照してください。

2. 監査ポリシーを作成する場合は、AUDIT文を使用して有効にし、オプションで、SYSDBA管理権限でログインする管理ユーザー(SYSなど)を含む、1人以上のユーザーに監査設定を適用(または除外)します。

   AUDITでは、アクションの成功または失敗(あるいは両方)時に監査レコードを作成することもできます。

   統合監査ポリシーの有効化およびユーザーとロールへの適用を参照してください。

3. UNIFIED_AUDIT_TRAILビューを問い合せ、生成された監査レコードを検索します。

   追加のビューについては、監査ポリシーのデータ・ディクショナリ・ビューも参照してください。

4. 監査証跡の内容を定期的にアーカイブして削除します。

   「監査証跡レコードの削除」を参照してください。

親トピック: 監査タイプの選択

27.1.2 一般的に使用されるセキュリティ関連アクティビティの監査

Oracle Databaseには、デフォルトの統合監査ポリシーが用意されており、これらは一般的に使用されるセキュリティ関連の監査用に選択できます。

このタイプの監査を実行する一般的なステップは次のとおりです。

1. デフォルトの監査ポリシーについて学習するには、事前定義の統合監査ポリシーを使用したアクティビティの監査を参照してください。
2. AUDIT文を使用してポリシーを有効にし、1人以上のユーザーに監査設定をオプションで適用(または除外)します。

   統合監査ポリシーの有効化およびユーザーとロールへの適用を参照してください。

3. UNIFIED_AUDIT_TRAILビューを問い合せ、生成された監査レコードを検索します。

   追加のビューについては、監査ポリシーのデータ・ディクショナリ・ビューも参照してください。

4. 監査証跡の内容を定期的にアーカイブして削除します。

   「監査証跡レコードの削除」を参照してください。

親トピック: 監査タイプの選択

27.1.3 特定のファイングレイン・アクティビティの監査

個々の列を監査またはイベント・ハンドラを使用する場合は、ファイングレイン監査を使用します。

このタイプの監査では、統合監査ポリシーで使用可能なすべての機能が提供されます。

ファイングレイン監査を実行する一般的なステップは次のとおりです。

1. ファイングレイン監査を使用した特定のアクティビティの監査を参照し、特定のアクティビティの監査の詳細を把握します。
2. DBMS_FGA PL/SQLパッケージを使用して、ファイングレイン監査ポリシーを構成します。DBMS_FGA PL/SQLパッケージを使用したファイングレイン監査ポリシーの管理を参照してください。
3. UNIFIED_AUDIT_TRAILビューを問い合せ、生成された監査レコードを検索します。

   追加のビューについては、監査ポリシーのデータ・ディクショナリ・ビューも参照してください。

4. 監査証跡の内容を定期的にアーカイブして削除します。

   「監査証跡レコードの削除」を参照してください。

27.2 統合監査ポリシーおよびAUDIT文を使用したアクティビティの監査

CREATE AUDIT POLICY文とAUDIT文を使用して、統合監査ポリシーを使用できます。

- 統合監査ポリシーおよびAUDITを使用したアクティビティの監査について
  統合監査ポリシーおよびAUDIT SQL文を使用して、複数のタイプのアクティビティを監査できます。
- カスタム統合監査ポリシーの作成のベスト・プラクティス
  データベースで複数のポリシーを一度に有効にできますが、有効にするポリシーの数を制限することが理想的です。
- 統合監査ポリシーの作成の構文
  統合監査ポリシーを作成するには、CREATE AUDIT POLICY文を使用します。
- ロールの監査
  CREATE AUDIT POLICY文を使用して、データベース・ロールを監査できます。
- システム権限の監査
  CREATE AUDIT POLICY文を使用して、システム権限を監査できます。
- 管理ユーザーの監査
  統合監査ポリシーを作成して、SYSなどの管理ユーザー・アカウントのアクションを取得できます。
- オブジェクト・アクションの監査
  CREATE AUDIT POLICY文を使用して、オブジェクト・アクションを監査できます。
- READ ANY TABLEおよびSELECT ANY TABLE権限の監査
  CREATE AUDIT POLICY文はREAD ANY TABLE権限とSELECT ANY TABLE権限を監査できます。
- 複数層環境におけるSQL文および権限の監査
  統合監査ポリシーを作成して、複数層環境のクライアントのアクティビティを監査できます。
- 統合監査ポリシーの条件の作成
  CREATE AUDIT POLICY文を使用すると、統合監査ポリシーの条件を作成できます。
- アプリケーション・コンテキスト値の監査
  AUDIT文を使用して、アプリケーション・コンテキスト値を監査できます。
- Oracle Database Real Application Securityイベントの監査
  CREATE AUDIT POLICY文を使用して、Oracle Database Real Application Securityイベントを監査できます。
- Oracle Recovery Managerイベントの監査
  CREATE AUDIT POLICY文を使用して、Oracle Recovery Managerイベントを監査できます。
- Oracle Database Vaultイベントの監査
  Oracle Database Vault環境で、CREATE AUDIT POLICY文を使用してDatabase Vaultアクティビティを監査できます。
- Oracle Label Securityイベントの監査
  Oracle Label Security環境で、CREATE AUDIT POLICY文を使用してOracle Label Securityアクティビティを監査できます。
- Oracle Data Miningイベントの監査
  CREATE AUDIT POLICY文を使用して、Oracle Data Miningイベントを監査できます。
- Oracle Data Pumpイベントの監査
  CREATE AUDIT POLICY文を使用して、Oracle Data Pumpを監査できます。
- Oracle SQL*Loaderダイレクト・ロード・パス・イベントの監査

CREATE AUDIT POLICY文を使用して、Oracle SQL*Loaderダイレクト・ロード・パス・イベントを監査できます。

- トップレベルの文のみの監査
  トップレベルのSQLまたはPL/SQL文を監査して、監査レコードのボリュームを制限できます。
- マルチテナント環境での統合監査ポリシーまたはAUDIT設定
  マルチテナント環境では、個々のPDBおよびルートに統合監査ポリシーを作成できます。
- 統合監査ポリシーの変更
  ALTER AUDIT POLICY文を使用して、統合監査ポリシーを変更できます。
- 統合監査ポリシーの有効化およびユーザーとロールへの適用
  AUDIT POLICY文を使用すると、統合監査ポリシーを有効にして、ユーザーとロールに適用できます。
- 統合監査ポリシーの無効化
  NOAUDIT POLICY文を使用して、統合監査ポリシーを無効にすることができます。
- 統合監査ポリシーの削除
  DROP AUDIT POLICY文を使用して、統合監査ポリシーを削除できます。
- 例: 非データベース・ユーザーの監査
  このチュートリアルでは、非データベース・ユーザーのアクションをクライアント識別子を使用して監査する統合監査ポリシーの作成方法を示します。

関連トピック

- SQL文、権限および他の一般アクティビティの監査

親トピック: 監査ポリシーの構成

27.2.1 統合監査ポリシーおよびAUDITを使用したアクティビティの監査について

統合監査ポリシーおよびAUDIT SQL文を使用して、複数のタイプのアクティビティを監査できます。

監査できるアクティビティの種類は、次のとおりです。

- ユーザー・アカウント(SYSDBA管理権限でログインする管理ユーザーを含む)、ロールおよび権限
- 表の削除やプロシージャの実行などのオブジェクト・アクション
- アプリケーション・コンテキスト値
- Oracle Database Real Application Security、Oracle Recovery Manager、Oracle Data Mining、Oracle Data Pump、Oracle SQL*Loaderダイレクト・パス・イベント、Oracle Database VaultおよびOracle Label Securityからのアクティビティ

監査するシステム・アクションを検索するには、AUDITABLE_SYSTEM_ACTIONSシステム表を問い合せます。

これを実行するには、監査する対象に応じて、次を実行します。

- 統合監査ポリシー。統合監査ポリシーとは、データベースでのユーザー動作の特定の部分を監査できる監査設定の名前付きのグループです。ポリシーを作成するには、CREATE AUDIT POLICY文を使用します。ポリシーは、1ユーザーのアクティビティの監査のような単純なものにすることも、条件を使用した複雑な監査ポリシーを作成することもできます。1つのデータベースで同時に複数の監査ポリシーを有効にできます。監査ポリシーには、システム全体の監査オプションとオブジェクト固有の監査オプションの両方を含めることができます。一般的なアクティビティに対する監査の大半(標準監査を含む)に、監査ポリシーを使用する必要があります。
- AUDITおよびNOAUDIT SQL文。AUDITおよびNOAUDIT SQL文では、それぞれ監査ポリシーを有効および無効にできます。AUDIT文では、ポリシーに対して特定のユーザーを含めたり、除外することもできます。AUDITおよび

NOAUDIT文では、アプリケーション・コンテキスト値を監査することもできます。

- Oracle Recovery Managerの場合は、統合監査ポリシーを作成しないでください。UNIFIED_AUDIT_TRAILビューでは、一般的に監査されるRecovery Managerイベントを自動的に取得します。

親トピック: 統合監査ポリシーおよびAUDIT文を使用したアクティビティの監査

27.2.2 カスタム統合監査ポリシーの作成のベスト・プラクティス

データベースで複数のポリシーを一度に有効にできますが、有効にするポリシーの数を制限することが理想的です。

統合監査ポリシーの構文は、データベースで必要なすべての監査設定を網羅する1つのポリシーを作成できるように設計されています。複数の小さいポリシーを作成するのではなく、関連するオプションを1つのポリシーにグループ化することをお薦めします。これにより、ポリシーをより簡単に管理できるようになります。たとえば、事前定義の各監査ポリシーには、1つの統合監査ポリシーに複数の監査設定が含まれています。

ユーザー・セッションで有効な監査ポリシーの数を制限すると、次の利点が得られます。

- 監査ポリシーの詳細をセッションのUGAメモリーにロードすることに伴うログオン・オーバーヘッドを軽減します。有効なポリシーの数が減ると、ポリシー情報のロードにかかる時間が短縮します。
- UGAメモリーにキャッシュするのに必要なポリシーの数が減るため、セッションのUGAメモリー消費量が減ります。
- 関連するイベントの監査レコードを生成するかどうかを決定する内部監査チェック機能が効率的になります。
- LOGON文の統合監査ポリシーを構成した場合は、直接ログインおよびALTER SESSION文とSET CONTAINER文の両方について、監査レコードが生成されます。

関連トピック

- 事前定義の統合監査ポリシーを使用したアクティビティの監査

親トピック: 統合監査ポリシーおよびAUDIT文を使用したアクティビティの監査

27.2.3 統合監査ポリシーの作成の構文

統合監査ポリシーを作成するには、CREATE AUDIT POLICY文を使用します。

統合監査ポリシーを作成すると、Oracle Databaseによって、ポリシーを作成したユーザーのスキーマではなく、SYSスキーマによって所有される最初のクラス・オブジェクトに格納されます。

例27-1に、CREATE AUDIT POLICY文の構文を示します。

例27-1 CREATE AUDIT POLICY文の構文

```
CREATE AUDIT POLICY policy_name
    { {privilege_audit_clause [action_audit_clause ] [role_audit_clause ]}
        | { action_audit_clause  [role_audit_clause ] }
        | { role_audit_clause }
     }
    [WHEN audit_condition EVALUATE PER {STATEMENT|SESSION|INSTANCE}]
    [ONLY TOPLEVEL]
    [CONTAINER = {CURRENT | ALL}];
```

詳細は、次のとおりです。

- privilege_audit_clauseは、権限に関連する監査オプションを記述します。詳細は、システム権限の監査を参照してください。権限の監査オプションを構成するための詳細な構文は、次のとおりです。

```
privilege_audit_clause  :=  PRIVILEGES  privilege1 [, privilege2]
```

- action_audit_clauseおよびstandard_actionsは、オブジェクト・アクションに関連する監査オプションを記述します。オブジェクト・アクションの監査を参照してください。構文は次のとおりです。

```
action_audit_clause := {standard_actions | component_actions}
                                          [, component_actions ]
standard_actions :=
     ACTIONS action1 [ ON {schema.obj_name
                                          | DIRECTORY directory_name
                                          | MINING MODEL schema.obj_name
                                           }
                ]
           [, action2 [ ON {schema.obj_name
                                          | DIRECTORY directory_name
                                          | MINING MODEL schema.obj_name
                   }
                ]
```

- component_actionsでは、Oracle Label Security、Oracle Database Real Application Security、Oracle Database Vault、Oracle Data PumpまたはOracle SQL*Loaderの監査ポリシーを作成できます。統合監査ポリシーおよびAUDIT文を使用したアクティビティの監査の該当する項を参照してください。構文は次のとおりです。

```
component_actions :=
     ACTIONS COMPONENT=[OLS|XS] action1 [,action2 ] |
     ACTIONS COMPONENT=DV DV_action ON DV_object_name |
     ACTIONS COMPONENT=DATAPUMP [ EXPORT | IMPORT | ALL ] |
     ACTIONS COMPONENT=DIRECT_LOAD [ LOAD | ALL ]
```

- role_audit_clauseでは、ロールを監査できます。「ロールの監査」を参照してください。構文は次のとおりです。

```
role_audit_clause := ROLES role1 [, role2]
```

- WHEN audit_condition EVALUATE PERでは、監査ポリシーおよび評価頻度の条件を作成するためのファンクションを指定できます。EVALUATE PER句には、WHEN条件を付ける必要があります。統合監査ポリシーの条件の作成を参照してください。構文は次のとおりです。

```
WHEN 'audit_condition := function operation value_list'
EVALUATE PER {STATEMENT|SESSION|INSTANCE}
```

- ONLY TOPLEVELを使用すると、ユーザーは、この監査ポリシーの一部として構成されているアクションに対して実行されるトップレベルの操作のみを監査できます。「トップレベルの文のみの監査」を参照してください。
- CONTAINERを使用すると、ユーザーは、この監査ポリシーの一部として構成されているアクションに対して実行されるトップレベルの操作のみを監査できます。マルチテナント環境での統合監査ポリシーまたはAUDIT設定を参照してください。

この構文は、ポリシーにリストされているコンポーネントを監査するように設計されています。たとえば、次のポリシーを作成するとします。

```
CREATE AUDIT POLICY table_pol
PRIVILEGES CREATE ANY TABLE, DROP ANY TABLE
ROLES emp_admin, sales_admin;
```

監査証跡では、CREATE ANY TABLEシステム権限またはDROP ANY TABLEシステム権限、ロールemp_adminに直接付与されている任意のシステム権限、またはロールsales_adminに直接付与されている任意のシステム権限を必要とする

SQL文を取得します。(監査されるのは、ロールを介して再帰的に付与されている権限ではなく、直接付与されている権限であることに注意してください。)

ポリシーを作成したら、AUDIT文を使用して有効にする必要があります。オプションで、1人以上のユーザーに対してポリシーの適用または除外を実行したり、監査アクションの成功または失敗(あるいは両方)時に監査レコードが書き込まれるかどうかを指定できます。統合監査ポリシーの有効化およびユーザーとロールへの適用を参照してください。

親トピック: 統合監査ポリシーおよびAUDIT文を使用したアクティビティの監査

27.2.4 ロールの監査

CREATE AUDIT POLICY文を使用して、データベース・ロールを監査できます。

- ロールの監査について
  ロールを監査すると、このロールに直接付与されたすべてのシステム権限がOracle Databaseによって監査されます。
- ロールの統合監査ポリシーの構成
  ロールの使用を取得する統合監査ポリシーを作成するには、CREATE AUDIT POLICY文にROLES句を含めます。
- 例: マルチテナント環境でのDBAロールの監査
  マルチテナント環境で、CREATE AUDIT POLICY文を使用してロールを監査できます。

親トピック: 統合監査ポリシーおよびAUDIT文を使用したアクティビティの監査

27.2.4.1 ロールの監査について

ロールを監査すると、このロールに直接付与されたすべてのシステム権限がOracle Databaseによって監査されます。

ユーザー定義のロールを含むすべてのロールを監査できます。ROLES監査オプションでロールに対して共通の統合監査ポリシーを作成する場合、ロール・リストで共通ロールのみを指定する必要があります。このようなポリシーが有効な場合、Oracle Databaseは、共通ロールに対して共通のシステム権限と共通ロールに直接付与されているシステム権限をすべて監査します。共通ロールに対してローカルに付与されているシステム権限は監査されません。ロールが共通に付与されているかどうかを確認するには、DBA_ROLESデータ・ディクショナリ・ビューを問い合せます。ロールに付与された権限が共通に付与されたかどうかを確認するには、ROLE_SYS_PRIVSビューを問い合せます。

関連トピック

- Oracle Databaseのインストールで事前に定義されているロール

親トピック: ロールの監査

27.2.4.2 ロールの統合監査ポリシーの構成

ロールの使用を取得する統合監査ポリシーを作成するには、CREATE AUDIT POLICY文にROLES句を含めます。

- 次の構文を使用して、ロールを監査する統合監査ポリシーを作成します。

```
CREATE AUDIT POLICY policy_name
 ROLES role1 [, role2];
```

たとえば:

```
CREATE AUDIT POLICY audit_roles_pol
 ROLES IMP_FULL_DATABASE, EXP_FULL_DATABASE;
```

条件を含む場合など、より複雑なロールの統合監査ポリシーを作成できます。ポリシーを作成したら、AUDIT文を使用して有効にする必要があります。

関連トピック

- 統合監査ポリシーの作成の構文

親トピック: ロールの監査

27.2.4.3 例: マルチテナント環境でのDBAロールの監査

マルチテナント環境で、CREATE AUDIT POLICY文を使用してロールを監査できます。

以下の例に、マルチテナント環境で事前定義済の共通ロールDBAを監査する方法を示します。

例27-2 マルチテナント環境でのDBAロールの監査

```
CREATE AUDIT POLICY role_dba_audit_pol
 ROLES DBA
 CONTAINER = ALL;
AUDIT POLICY role_dba_audit_pol;
```

親トピック: ロールの監査

27.2.5 システム権限の監査

CREATE AUDIT POLICY文を使用して、システム権限を監査できます。

- システム権限監査について
  システム権限の監査では、システム権限を正常に使用するアクティビティ(READ ANY TABLEなど)を監査します。
- 監査できるシステム権限
  ほとんどのシステム権限の使用を監査できます。
- 監査できないシステム権限
  いくつかのシステム権限は監査できません。
- システム権限の使用を取得するための統合監査ポリシーの構成
  CREATE AUDIT POLICY文のPRIVILEGES句で、システム権限の使用を監査します。
- 例: ANY権限を持つユーザーの監査
  CREATE AUDIT POLICY文で、ANY権限のユーザーを監査できます。
- 例: 条件を使用するシステム権限の監査
  CREATE AUDIT POLICY文で、システム権限の監査に条件を使用する監査ポリシーを作成できます。
- 監査証跡でのシステム権限の統合監査ポリシーの表示方法
  UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューはシステム権限の監査イベントを表示します。

親トピック: 統合監査ポリシーおよびAUDIT文を使用したアクティビティの監査

27.2.5.1 システム権限監査について

システム権限の監査では、システム権限を正常に使用するアクティビティ(READ ANY TABLEなど)を監査します。

この種の監査では、正常終了するために監査対象の権限を必要とするSQL文が記録されます。

1つの統合監査ポリシーに、権限監査およびアクション監査の両方のオプションを含めることができます。SYSなどの管理ユーザーの権限の使用は監査しないでください。かわりに、オブジェクト・アクションを監査してください。

1つの問合せで複数の監査レコードを生成できることに注意してください。1つの問合せは、問合せでアクセスされるオブジェクトごとに1つずつ生成でき、これらのすべてのオブジェクトで監査が有効になっている場合です。たとえば、ビューを問い合せるときに、ビュー自体で参照される基礎となるオブジェクトごとに複数の監査レコードを生成できます。

ノート:

システム権限、オブジェクト、データベース・イベントなどを監査できます。ただし、データベース権限の使用状況(たとえば、指定のロールに付与されている権限はどれが使用されているか)を検索し、使用されている権限と使用されていない権限のレポートを生成する必要がある場合は、権限キャプチャを作成できます。

関連トピック

- オブジェクト・アクションの監査
- 権限分析の実行による権限使用の特定

親トピック: システム権限の監査

27.2.5.2 監査できるシステム権限

ほとんどのシステム権限の使用を監査できます。

監査可能なシステム権限のリストを検索するには、SYSTEM_PRIVILEGE_MAP表を問い合せます。

たとえば:

```
SELECT NAME FROM SYSTEM_PRIVILEGE_MAP;
NAME
-------------
ALTER ANY CUBE BUILD PROCESS
SELECT ANY CUBE BUILD PROCESS
ALTER ANY MEASURE FOLDER
...
```

アクション監査オプションと同様、権限監査では、データベース・ユーザーに付与されているシステム権限の使用を監査します。SQL文監査と権限監査の両方について類似の監査オプションを設定しても、生成される監査レコードは1つのみです。たとえば、2つのポリシーが存在しており、一方が、特にHR.PROCプロシージャのEXECUTE PROCEDUREを監査し、もう一方が一般的にEXECUTE PROCEDURE (すべてのプロシージャ)を監査する場合は、監査レコードが1つのみ書き込まれます。

権限監査は、アクションが既存の所有者およびオブジェクト権限によってすでに許可されている場合は実行されません。権限監査がトリガーされるのは、権限が不十分な場合、つまりアクションを可能にする権限がシステム権限である場合のみです。たとえば、ユーザーSCOTTにSELECT ANY TABLE権限が付与されており、SELECT ANY TABLEが監査対象であるとします。SCOTTが自分の表(たとえば、SCOTT.EMP)を選択した場合、SELECT ANY TABLE権限は使用されません。自分自身のスキーマ内でSELECT文を実行したため、監査レコードは生成されません。一方、SCOTTが他のスキーマ(たとえばHR.EMPLOYEES表)から選択すると、監査レコードが生成されます。SCOTTは自分自身のスキーマ外にある表を選択したため、SELECT ANY TABLE権限を使用する必要がありました。

親トピック: システム権限の監査

27.2.5.3 監査できないシステム権限

次のシステム権限は監査できません。

これらの権限は、次のとおりです。

- INHERIT ANY PRIVILEGE
- INHERIT PRIVILEGE
- TRANSLATE ANY SQL

- TRANSLATE SQL

親トピック: システム権限の監査

27.2.5.4 システム権限の使用を取得するための統合監査ポリシーの構成

CREATE AUDIT POLICY文のPRIVILEGES句で、システム権限の使用を監査します。

- 次の構文を使用して、権限を監査する統合監査ポリシーを作成します。

```
CREATE AUDIT POLICY policy_name
 PRIVILEGES privilege1 [, privilege2];
```

たとえば:

```
CREATE AUDIT POLICY my_simple_priv_policy
 PRIVILEGES SELECT ANY TABLE, CREATE LIBRARY;
```

条件を含む場合など、より複雑な権限の統合監査ポリシーを作成できます。ポリシーを作成したら、AUDIT文を使用して有効にする必要があります。

関連トピック

- 統合監査ポリシーの作成の構文

親トピック: システム権限の監査

27.2.5.5 例: ANY権限を持つユーザーの監査

CREATE AUDIT POLICY文で、ANY権限のユーザーを監査できます。

例27-3に、ユーザーHR_MGRの複数のANY権限を監査する方法を示します。

例27-3 ANY権限を持つユーザーの監査

```
CREATE AUDIT POLICY hr_mgr_audit_pol
 PRIVILEGES DROP ANY TABLE, DROP ANY CONTEXT, DROP ANY INDEX, DROP ANY LIBRARY;
AUDIT POLICY hr_mgr_audit_pol BY HR_MGR;
```

親トピック: システム権限の監査

27.2.5.6 例: 条件を使用するシステム権限の監査

CREATE AUDIT POLICY文で、システム権限の監査に条件を使用する監査ポリシーを作成できます。

例27-4に、psmithおよびjrawlinsの2人のオペレーティング・システム・ユーザーによって使用される権限を監査する方法を示します。

例27-4 条件を使用するシステム権限の監査

```
CREATE AUDIT POLICY os_users_priv_pol
 PRIVILEGES SELECT ANY TABLE, CREATE LIBRARY
 WHEN 'SYS_CONTEXT (''USERENV'', ''OS_USER'') IN (''psmith'', ''jrawlins'')'
 EVALUATE PER SESSION;
AUDIT POLICY os_users_priv_pol;
```

親トピック: システム権限の監査

27.2.5.7 監査証跡でのシステム権限の統合監査ポリシーの表示方法

UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューはシステム権限の監査イベントを表示します。

次の例では、例27-4で作成した統合監査ポリシーos_users_priv_polに基づいて、オペレーティング・システム・ユーザーpsmithによって使用される権限のリストが表示されます。

```
SELECT SYSTEM_PRIVILEGE_USED FROM UNIFIED_AUDIT_TRAIL
 WHERE OS_USERNAME = 'PSMITH' AND UNIFIED_AUDIT_POLICIES = 'OS_USERS_PRIV_POL';
SYSTEM_PRIVILEGE_USED
---------------------
SELECT ANY TABLE
DROP ANY TABLE
```

ノート:

SELECT ANY TABLE システム権限に対する監査ポリシーを作成した場合、ユーザーが READ オブジェクト権限を実行したか、SELECT オブジェクト権限を実行したかによって監査証跡で取得されるアクションが影響を受けます。

関連トピック

- READ ANY TABLEおよびSELECT ANY TABLE権限の監査

親トピック: システム権限の監査

27.2.6 管理ユーザーの監査

統合監査ポリシーを作成して、SYSなどの管理ユーザー・アカウントのアクションを取得できます。

- 監査可能な管理ユーザー・アカウント
  Oracle Databaseでは、管理権限に関連付けられている管理ユーザー・アカウントが用意されています。
- 管理者アクティビティを取得するための統合監査ポリシーの構成
  CREATE AUDIT POLICY文で、管理ユーザーを監査できます。
- 例: SYSユーザーの監査
  CREATE AUDIT POLICY文で、SYSユーザーを監査できます。

親トピック: 統合監査ポリシーおよびAUDIT文を使用したアクティビティの監査

27.2.6.1 監査可能な管理ユーザー・アカウント

Oracle Databaseでは、管理権限に関連付けられている管理ユーザー・アカウントが用意されています。

表27-1に、デフォルトの管理ユーザー・アカウントと、通常関連付けられている管理権限を示します。

表27-1 管理ユーザーおよび管理権限

管理ユーザー・アカウント	管理権限
SYS	SYSDBA
PUBLIC 脚注 1	SYSOPER
SYSASM	SYSASM

管理ユーザー・アカウント	管理権限
SYSBACKUP	SYSBACKUP
SYSDG	SYSDG
SYSKM	SYSKM

脚注1

PUBLICはユーザーPUBLICを示し、SYSOPER管理権限でログインしている場合に有効なユーザーです。これはPUBLICロールを示しません。

関連トピック

- 強制的に監査されるアクティビティ

親トピック: 管理ユーザーの監査

27.2.6.2 管理者アクティビティを取得するための統合監査ポリシーの構成

CREATE AUDIT POLICY文で、管理ユーザーを監査できます。

- 管理ユーザーを監査する場合は、統合監査ポリシーを作成して、非管理ユーザーに適用する場合と同じように、このポリシーをユーザーに適用します。管理ユーザーによるトップ・レベルの文は、データベースを開くまで強制的に監査されます。

親トピック: 管理ユーザーの監査

27.2.6.3 例: SYSユーザーの監査

CREATE AUDIT POLICY文で、SYSユーザーを監査できます。

例27-5に、ユーザーSYSによるDBMS_FGA PL/SQLパッケージの権限付与を監査する方法を示します。

例27-5 SYSユーザーの監査

```
CREATE AUDIT POLICY dbms_fga_grants
 ACTIONS GRANT
 ON DBMS_FGA;
AUDIT POLICY dbms_fga_grants BY SYS;
```

親トピック: 管理ユーザーの監査

27.2.7 オブジェクト・アクションの監査

CREATE AUDIT POLICY文を使用して、オブジェクト・アクションを監査できます。

- オブジェクト・アクションの監査について
  HR.EMPLOYEES表のUPDATE文など、特定のオブジェクトで実行されるアクションを監査できます。
- 監査できるオブジェクト・アクション
  オブジェクト・アクションの監査では、監査の対象を拡大または限定できます(すべてのユーザー・アクションの監査または一部のユーザー・アクションに限定した監査など)。
- オブジェクト・アクションの統合監査ポリシーの構成
  CREATE AUDIT POLICY文のACTIONS句で、オブジェクト・アクションを取得するポリシーを作成します。

- 例: SYSオブジェクトでのアクションの監査
  CREATE AUDIT POLICY文で、SYSオブジェクトのアクションを監査できます。
- 例: 1つのオブジェクトでの複数のアクションの監査
  CREATE AUDIT POLICY文で、1つのオブジェクト上の複数のアクションを監査できます。
- 例: オブジェクトに対するGRANTおよびREVOKE操作の監査
  CREATE AUDIT POLICY文は、表などのオブジェクトに対するGRANTおよびREVOKE操作を監査できます。
- 例: オブジェクトでのアクションと権限の両方の監査
  CREATE AUDIT POLICY文で、単一のポリシーを使用して1つのオブジェクトでの複数のアクションと権限の両方を監査できます。
- 例: 表でのすべてのアクションの監査
  CREATE AUDIT POLICY文で、表でのすべてのアクションを監査できます。
- 例: データベースでのすべてのアクションの監査
  CREATE AUDIT POLICY文で、データベースでのすべてのアクションを監査できます。
- 監査証跡でのオブジェクト・アクションの統合監査ポリシーの表示方法
  UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューはオブジェクト・アクションの監査イベントを表示します。
- ファンクション、プロシージャ、パッケージおよびトリガーの監査
  ファンクション、プロシージャ、PL/SQLパッケージおよびトリガーを監査できます。
- Oracle Virtual Private Databaseの述語の監査
  統合監査証跡では、Oracle Virtual Private Database (VPD)のポリシーで使用される述語を自動的に取得します。
- Oracle Virtual Private Databaseポリシー関数の監査ポリシー
  監査によって動的VPDポリシーや静的VPDポリシー、コンテキスト依存VPDポリシーが影響を受けることがあります。
- 統合監査とエディション付きオブジェクト
  エディション付きオブジェクトに統合監査ポリシーが付加されている場合、そのオブジェクトが表示されるすべてのエディションにそのポリシーが適用されます。

親トピック: 統合監査ポリシーおよびAUDIT文を使用したアクティビティの監査

27.2.7.1 オブジェクト・アクションの監査について

HR.EMPLOYEES表のUPDATE文など、特定のオブジェクトで実行されるアクションを監査できます。

監査は、オブジェクトに使用されたDDLおよびDML文を含むことができます。1つの統合監査ポリシーに、権限監査およびアクション監査の両方のオプションと、複数のオブジェクトの監査オプションのセットを含めることができます。

親トピック: オブジェクト・アクションの監査

27.2.7.2 監査できるオブジェクト・アクション

オブジェクト・アクションの監査では、監査の対象を拡大または限定できます(すべてのユーザー・アクションの監査または一部のユーザー・アクションに限定した監査など)。

表27-2に、オブジェクト・レベルの標準データベース・アクションのオプションを示します。SELECT SQL文に対する監査ポリシーでは、SELECTアクションだけでなくREADアクションも取得されます。

表27-2 オブジェクト・レベルの標準データベース・アクションの監査オプション

オブジェクト	監査できるSQLアクション

オブジェクト	監査できるSQLアクション
表	ALTER、AUDIT、COMMENT、DELETE、FLASHBACK、GRANT、INDEX、INSERT、LOCK、RENAME、SELECT、UPDATE
ビュー	AUDIT、COMMENT、DELETE、FLASHBACK、GRANT、INSERT、LOCK、RENAME、SELECT、UPDATE
順序	ALTER、AUDIT、GRANT、SELECT
プロシージャ(トリガーを含む)	AUDIT, EXECUTE, GRANT
ファンクション	AUDIT, EXECUTE, GRANT
パッケージ	AUDIT, EXECUTE, GRANT
マテリアライズド・ビュー	ALTER、AUDIT、COMMENT、DELETE、INDEX、INSERT、LOCK、SELECT、UPDATE
マイニング・モデル	AUDIT、COMMENT、GRANT、RENAME、SELECT
ディレクトリ	AUDIT, GRANT, READ
ライブラリ	EXECUTE, GRANT
オブジェクト型	ALTER, AUDIT, GRANT
Java スキーマ・オブジェクト (ソース、クラス、リソース)	AUDIT, EXECUTE, GRANT

関連トピック

- ファンクション、プロシージャ、パッケージおよびトリガーの監査
- Oracle Virtual Private Databaseポリシー関数の監査ポリシー

親トピック: オブジェクト・アクションの監査

27.2.7.3 オブジェクト・アクションの統合監査ポリシーの構成

CREATE AUDIT POLICY文のACTIONS句で、オブジェクト・アクションを取得するポリシーを作成します。

- 次の構文を使用して、オブジェクト・アクションを監査する統合監査ポリシーを作成します。

```
CREATE AUDIT POLICY policy_name
 ACTIONS action1 [, action2 ON object1] [, action3 ON object2];
```

たとえば:

```
CREATE AUDIT POLICY my_simple_obj_policy
 ACTIONS SELECT ON OE.ORDERS, UPDATE ON HR.EMPLOYEES;
```

この例に示すように、複数のオブジェクトの複数のアクションを監査できます。

条件を含む場合など、より複雑なオブジェクト・アクションの統合監査ポリシーを作成できます。ポリシーを作成したら、AUDIT文を使用して有効にする必要があります。

関連トピック

- 統合監査ポリシーの作成の構文

親トピック: オブジェクト・アクションの監査

27.2.7.4 例: SYSオブジェクトでのアクションの監査

CREATE AUDIT POLICY文で、SYSオブジェクトのアクションを監査できます。

例27-6では、SYS.USER$システム表でSELECT文を監査する監査ポリシーの作成方法を示します。この監査ポリシーは、SYSおよびSYSTEMを含むすべてのユーザーに適用されます。

例27-6 SYSオブジェクトの監査アクション

```
CREATE AUDIT POLICY select_user_dictionary_table_pol ACTIONS SELECT ON SYS.USER$;
AUDIT POLICY select_user_dictionary_table_pol;
```

親トピック: オブジェクト・アクションの監査

27.2.7.5 例: 1つのオブジェクトでの複数のアクションの監査

CREATE AUDIT POLICY文で、1つのオブジェクト上の複数のアクションを監査できます。

例27-7に、ユーザーjrandolphおよびphawkinsによってapp_libライブラリで実行される複数のSQL文の監査方法を示します。

例27-7 1つのオブジェクトでの複数のアクションの監査

```
CREATE AUDIT POLICY actions_on_hr_emp_pol1
 ACTIONS EXECUTE, GRANT
 ON app_lib;
AUDIT POLICY actions_on_hr_emp_pol1 BY jrandolph, phawkins;
```

親トピック: オブジェクト・アクションの監査

27.2.7.6 例: オブジェクトに対するGRANTおよびREVOKE操作の監査

CREATE AUDIT POLICY文で、表などのオブジェクトに対するGRANTおよびREVOKE操作を監査できます。

オブジェクトに対するGRANT操作の監査を有効にすると、オブジェクトに対するREVOKE操作の監査も自動的に有効になります。

例27-8 GRANT操作とREVOKE操作の監査

```
CREATE AUDIT POLICY grant_revoke_pol
ACTIONS GRANT ON HR.EMPLOYEES;
AUDIT POLICY grant_revoke_pol;
```

このタイプのポリシーのUNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューから選択するには、次のような問合せを実行できます。権限が付与されている権限受領者名は、TARGET_USER列に記録されます。

```
SELECT DBUSERNAME, OBJECT_PRIVILEGES, ACTION_NAME, OBJECT_SCHEMA, OBJECT_NAME,
TARGET_USER
```

```
FROM UNIFIED_AUDIT_TRAIL
WHERE ACTION_NAME IN ('GRANT', 'REVOKE');
```

親トピック: オブジェクト・アクションの監査

27.2.7.7 例: オブジェクトでのアクションと権限の両方の監査

CREATE AUDIT POLICY文で、単一のポリシーを使用して1つのオブジェクトでの複数のアクションと権限の両方を監査できます。

例27-9に、例27-7の変化形(CREATE LIBRARY権限を使用してapp_libライブラリでのすべてのEXECUTE文とGRANT文を監査する)を示します。

例27-9 オブジェクトでのアクションと権限の両方の監査

```
CREATE AUDIT POLICY actions_on_hr_emp_pol2
 PRIVILEGES CREATE LIBRARY
 ACTIONS EXECUTE, GRANT
 ON app_lib;
AUDIT POLICY actions_on_hr_emp_pol2 BY jrandolph, phawkins;
```

ディレクトリ・オブジェクトは監査可能です。たとえば、ORACLE_LOADERアクセス・ドライバで使用されるプリプロセッサ・プログラムを含むディレクトリ・オブジェクトを作成するとします。ディレクトリ・オブジェクト内のこのプログラムを実行するすべてのユーザーを監査できます。

親トピック: オブジェクト・アクションの監査

27.2.7.8 例: 表でのすべてのアクションの監査

CREATE AUDIT POLICY文で、表でのすべてのアクションを監査できます。

ALLキーワードを使用してすべてのアクションを監査できます。すべてのアクションの監査は機密オブジェクトに対してのみ実行することをお薦めします。ALLは、間接的なSELECT操作を取得する場合に便利です。例27-10に、HR.EMPLOYEES表でのすべてのアクション(ユーザーpmulliganによるアクションを除く)を監査する方法を示します。

例27-10 表でのすべてのアクションの監査

```
CREATE AUDIT POLICY all_actions_on_hr_emp_pol
 ACTIONS ALL ON HR.EMPLOYEES;
AUDIT POLICY all_actions_on_hr_emp_pol EXCEPT pmulligan;
```

関連トピック

- 例: データベースでのすべてのアクションの監査

親トピック: オブジェクト・アクションの監査

27.2.7.9 例: データベースでのすべてのアクションの監査

CREATE AUDIT POLICY文で、データベースでのすべてのアクションを監査できます。

この監査ポリシー構成のすべての再帰的アクションであっても、多数の監査レコードが生成されてすぐに監査証跡がいっぱいになることを回避するには、CREATE AUDIT POLICY文にONLY TOPLEVEL句を含めます。ONLY TOPLEVELのかわりに、条件を使用してACTIONS ALLポリシーを作成し、レコードのサブセットのみを取得できます。

ノート:

ACTIONS ALL 監査は注意して使用してください。オンライン・トランザクション処理(OLTP)ワークロードを実行する必要があるユーザーには有効にしないでください。これにより、多数の監査レコードが生成されなくなります。

例27-11に、データベース全体でのすべてのアクションを監査する方法を示します。

例27-11 データベースでのすべてのアクションの監査

```
CREATE AUDIT POLICY all_actions_pol ACTIONS ALL ONLY TOPLEVEL;
AUDIT POLICY all_actions_pol;
```

関連トピック

- 統合監査ポリシーの条件の作成

親トピック: オブジェクト・アクションの監査

27.2.7.10 監査証跡でのオブジェクト・アクションの統合監査ポリシーの表示方法

UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューはオブジェクト・アクションの監査イベントを表示します。

たとえば:

```
SELECT ACTION_NAME, OBJECT_SCHEMA, OBJECT_NAME FROM UNIFIED_AUDIT_TRAIL
WHERE DBUSERNAME = 'SYS';
ACTION_NAME OBJECT_SCHEMA OBJECT_NAME
----------- ------------- ------------
SELECT      HR            EMPLOYEES
```

親トピック: オブジェクト・アクションの監査

27.2.7.11 ファンクション、プロシージャ、パッケージおよびトリガーの監査

ファンクション、プロシージャ、PL/SQLパッケージおよびトリガーを監査できます。

監査可能な領域は次のとおりです。

- スタンドアロン・ファンクション、スタンドアロン・プロシージャおよびPL/SQLパッケージは個別に監査できます。
- PL/SQLパッケージを監査すると、パッケージ内のすべてのファンクションおよびプロシージャが監査されます。
- すべての実行の監査を使用可能にすると、データベース内のすべてのトリガーおよびPL/SQLパッケージ内のすべてのファンクションとプロシージャが監査されます。
- PL/SQLパッケージ内のファンクションまたはプロシージャを個別に監査することはできません。
- PL/SQLストアド・プロシージャまたはストアド・ファンクションでのEXECUTE操作を監査する場合は、監査目的での操作の成否を判断する際に、プロシージャまたはファンクションを検索してその実行を認証する機能のみが監査対象となります。したがって、WHENEVER NOT SUCCESSFUL句を指定すると、無効なオブジェクト・エラー、存在しないオブジェクト・エラー、および認証の失敗が監査されます。プロシージャまたはファンクションの実行時に検出されたエラーは監査されません。WHENEVER SUCCESSFUL句を指定すると、実行時にエラーが検出されたかどうかに関係なく、無効なオブジェクト・エラー、存在しないオブジェクト・エラー、および認証の失敗が監査されます。

親トピック: オブジェクト・アクションの監査

27.2.7.12 Oracle Virtual Private Databaseの述語の監査

統合監査証跡では、Oracle Virtual Private Database (VPD)のポリシーで使用される述語を自動的に取得します。

VPD述語監査情報を取得するために統合監査ポリシーを作成する必要はありません。

このタイプの監査では、DML操作の一部として実行された述語式を識別できるため、DML操作の一部として生じる可能性のある他のアクションを識別するために役立ちます。たとえば、VPD述語を使用してデータベースに対して悪意のある攻撃が行われた場合、統合監査証跡を使用してその攻撃を追跡できます。ユーザーが作成したVPDポリシーの述語に加えて、Oracle Label SecurityおよびOracle Real Application Securityのポリシーの内部述語も取得されます。たとえば、Oracle Label Securityは、OLSポリシーを表に適用する一方で、VPDポリシーを内部的に作成します。Oracle Real Application Securityは、Oracle RASポリシーを有効にする一方で、VPDポリシーを生成します。

統合監査証跡では、この述語情報がUNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューのRLS_INFO列に書き込まれます。ファイングレイン監査ポリシーを使用している場合、これらのビューのRLS_INFO列にVPD述語情報も取得されます。

監査証跡では、オブジェクトに対して複数のVPDポリシーが実施されている場合、述語とその述語が対応するポリシー名を取得できます。監査証跡ではポリシー・スキーマとポリシー名が取得されるため、異なるポリシーから生成された述語を区別することができます。この情報はデフォルトでRLS_INFO列に連結されますが、Oracle Databaseに用意されたDBMS_AUDIT_UTIL PL/SQLパッケージのファンクションを使用すると、結果を読みやすい形式に再フォーマットできます。

次の例は、VPDポリシーの述語を監査する方法を示しています。

1. 次のVPDポリシー関数を作成します。

```
CREATE OR REPLACE FUNCTION auth_orders(
  schema_var IN VARCHAR2,
  table_var  IN VARCHAR2
 )
 RETURN VARCHAR2
 IS
  return_val VARCHAR2 (400);
 BEGIN
  return_val := 'SALES_REP_ID = 159';
  RETURN return_val;
 END auth_orders;
/
```

2. 次のVPDポリシーを作成します。

```
BEGIN
  DBMS_RLS.ADD_POLICY (
    object_schema    => 'oe',
    object_name      => 'orders',
    policy_name      => 'orders_policy',
    function_schema  => 'sec_admin',
    policy_function  => 'auth_orders',
    statement_types  => 'select, insert, update, delete'
   );
 END;
/
```

3. 次の統合監査ポリシーを作成して有効にします。

```
CREATE AUDIT POLICY oe_pol
 ACTIONS SELECT ON OE.ORDERS;
AUDIT POLICY oe_pol;
```

4. ユーザーOEとして接続し、OE.ORDERS表を問い合せます。

```
CONNECT OE
Enter password: password
SELECT COUNT(*) FROM ORDERS;
```

5. AUDIT_ADMINロールを付与されたユーザーとして接続し、UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビュー

を問い合せます。

```
CONNECT sec_admin
Enter password: password
SELECT RLS_INFO FROM UNIFIED_AUDIT_TRAIL;
```

次のような出力が表示されます。

```
((POLICY_TYPE=[3]'VPD'),(POLICY_SCHEMA=[9]'SEC_ADMIN'),(POLICY_NAME=[13]'ORDERS
_POLICY'),(PREDICATE=[16]'SALES_REP_ID=159'));
```

6. これらの詳細を抽出して該当する列に追加するために、DBMS_AUDIT_UTIL PL/SQLパッケージの適切なファンクションを実行します。

   統合監査の場合、DBMS_AUDIT_UTIL.DECODE_RLS_INFO_ATRAIL_UNIファンクションを実行する必要があります。

   たとえば:

```
SELECT DBUSERNAME, ACTION_NAME, OBJECT_NAME, SQL_TEXT,
  RLS_PREDICATE, RLS_POLICY_TYPE, RLS_POLICY_OWNER, RLS_POLICY_NAME
  FROM TABLE (DBMS_AUDIT_UTIL.DECODE_RLS_INFO_ATRAIL_UNI
  (CURSOR (SELECT * FROM UNIFIED_AUDIT_TRAIL)));
```

   再フォーマットされた監査証跡の出力は、次のようになります。

```
DBUSERNAME ACTION_NAME OBJECT_NAME SQL_TEXT
---------- ----------- ----------- --------------------------
RLS_PREDICATE       RLS_POLICY_TYPE RLS_POLICY_OWNER RLS_POLICY_NAME
------------------ ---------------  ---------------- ---------------
OE          SELECT      ORDERS      SELECT COUNT(*) FROM ORDERS
SALES_REP_ID = 159  VPD              SEC_ADMIN        ORDERS_POLICY
```

関連項目:

- Oracle Virtual Private Databaseの詳細は、Oracle Virtual Private Databaseを使用したデータ・アクセスの制御を参照してください
- DBMS_AUDIT_UTIL PL/SQLパッケージの詳細は、Oracle Database PL/SQLパッケージおよびタイプ・リファレンスを参照してください

親トピック: オブジェクト・アクションの監査

27.2.7.13 Oracle Virtual Private Databaseポリシー関数の監査ポリシー

監査によって動的VPDポリシーや静的VPDポリシー、コンテキスト依存VPDポリシーが影響を受けることがあります。

- 動的ポリシー: ポリシー関数が2回(SQL文の解析時に1回と実行時に1回)評価されます。結果として、各評価で2つの監査レコードが生成されます。
- 静的ポリシー: ポリシー関数が1回評価され、SGA内にキャッシュされます。これにより、監査レコードは1つのみ生成されます。
- 状況依存ポリシー: ポリシー関数が文の解析時に1回実行されます。これにより、監査レコードは1つのみ生成されます。

親トピック: オブジェクト・アクションの監査

27.2.7.14 統合監査とエディション付きオブジェクト

エディション付きオブジェクトに統合監査ポリシーが付加されている場合、そのオブジェクトが表示されるすべてのエディションにそのポリシーが適用されます。

エディション付きオブジェクトが実現化されると、そのオブジェクトに付加されている統合監査ポリシーが新しい実際のオブジェクトに新たに付加されます。継承されたエディション付きオブジェクトに新たに統合監査ポリシーを適用すると、そのオブジェクトが実現化されます。

監査対象のオブジェクトが表示されるエディションを調べるには、UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューのOBJECT_NAMEおよびOBJ_EDITION_NAME列を問い合せます。

関連項目:

エディションの詳細は、『Oracle Database開発ガイド』を参照してください。

親トピック: オブジェクト・アクションの監査

27.2.8 READ ANY TABLEおよびSELECT ANY TABLE権限の監査

CREATE AUDIT POLICY文で、READ ANY TABLE権限とSELECT ANY TABLE権限を監査できます。

- READ ANY TABLEおよびSELECT ANY TABLE権限の監査について
  READ ANY TABLEおよびSELECT ANY TABLEシステム権限の使用を取得する統合監査ポリシーを作成できます。
- READオブジェクト権限操作を取得する統合監査ポリシーの作成
  READオブジェクト権限操作を取得する統合監査ポリシーを作成できます。
- 統合監査証跡でのREAD ANY TABLEおよびSELECT ANY TABLEの取得方法
  統合監査証跡では、ユーザーにREAD ANY TABLEまたはSELECT ANY TABLE権限が付与されているかどうかに基づいてSELECT動作が取得されます。

親トピック: 統合監査ポリシーおよびAUDIT文を使用したアクティビティの監査

27.2.8.1 READ ANY TABLEおよびSELECT ANY TABLE権限の監査について

READ ANY TABLEおよびSELECT ANY TABLEシステム権限の使用を取得する統合監査ポリシーを作成できます。

ユーザーが実行しようとしたアクションおよびユーザーに付与されていた権限に基づいて、UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューのSYSTEM_PRIVILEGE_USED列にREAD ANY TABLEシステム権限またはSELECT ANY TABLEシステム権限が記録されます。たとえば、ユーザーにSELECT ANY TABLE権限が付与されていて、表に対する問合せを実行するとします。監査証跡に、ユーザーがSELECT ANY TABLEシステム権限を使用したことが記録されます。ユーザーにREAD ANY TABLEが付与されていて、同じ問合せを実行した場合、READ ANY TABLE権限が記録されます。

親トピック: READ ANY TABLEおよびSELECT ANY TABLE権限の監査

27.2.8.2 READオブジェクト権限操作を取得する統合監査ポリシーの作成

READオブジェクト権限操作を取得する統合監査ポリシーを作成できます。

- すべてのREADオブジェクト操作を取得する統合監査ポリシーを作成する場合、READ文ではなくSELECT文に対するポリシーを作成します。

たとえば:

```
CREATE AUDIT POLICY read_hr_employees
 ACTIONS SELECT ON HR.EMPLOYEES;
```

監査できる他のオブジェクト・アクションの場合と同様に、SELECTオブジェクトの操作に対しても、SELECT文に関するポリシーを作成します。

関連トピック

- オブジェクト・アクションの監査

親トピック: READ ANY TABLEおよびSELECT ANY TABLE権限の監査

27.2.8.3 統合監査証跡でのREAD ANY TABLEおよびSELECT ANY TABLEの取得方法

統合監査証跡では、ユーザーにREAD ANY TABLEまたはSELECT ANY TABLE権限が付与されているかどうかに基づいてSELECT動作が取得されます。

表27-3は、統合監査証跡によるこれらのアクションの取得方法を示しています。

表27-3 READ ANY TABLEおよびSELECT ANY TABLEに対する監査動作

ユーザーが発行した文	ユーザーに付与されている権限	監査されるシステム権限	予期されるUNIFIED_AUDIT_TRAILの動作
SELECT	SELECT ANY TABLE	SELECT ANY TABLE	SYSTEM_PRIVILEGE_USED に挿入されるレコード: SELECT ANY TABLE
SELECT	SELECT ANY TABLE	READ ANY TABLE	レコードなし
SELECT	SELECT ANY TABLE	SELECT ANY TABLE と READ ANY TABLE の両方	SYSTEM_PRIVILEGE_USED に挿入されるレコード: SELECT ANY TABLE
SELECT	SELECT ANY TABLE	SELECT ANY TABLE も READ ANY TABLE も対象外	レコードなし
SELECT	READ ANY TABLE	SELECT ANY TABLE	レコードなし
SELECT	READ ANY TABLE	READ ANY TABLE	SYSTEM_PRIVILEGE_USED に挿入されるレコード: READ ANY TABLE
SELECT	READ ANY TABLE	SELECT ANY TABLE と	SYSTEM_PRIVILEGE_USED に挿入

ユーザーが発行した文	ユーザーに付与されている権限	監査されるシステム権限	予期されるUNIFIED_AUDIT_TRAILの動作
		READ ANY TABLE の両方	されるレコード: READ ANY TABLE
SELECT	READ ANY TABLE	SELECT ANY TABLE も READ ANY TABLE も対象外	レコードなし
SELECT	SELECT ANY TABLE と READ ANY TABLE の両方	SELECT ANY TABLE	READ ANY TABLE がアクセスに使用されたため、レコードなし
SELECT	SELECT ANY TABLE と READ ANY TABLE の両方	READ ANY TABLE	SYSTEM_PRIVILEGE_USED に挿入されるレコード: READ ANY TABLE
SELECT	SELECT ANY TABLE と READ ANY TABLE の両方	SELECT ANY TABLE と READ ANY TABLE の両方	SYSTEM_PRIVILEGE_USED に挿入されるレコード: READ ANY TABLE
SELECT	SELECT ANY TABLE と READ ANY TABLE の両方	SELECT ANY TABLE も READ ANY TABLE も対象外	レコードなし
SELECT	SELECT ANY TABLE も READ ANY TABLE も対象外	SELECT ANY TABLE	レコードなし
SELECT	SELECT ANY TABLE も READ ANY TABLE も対象外	READ ANY TABLE	レコードなし
SELECT	SELECT ANY TABLE も READ ANY TABLE も対	SELECT ANY TABLE と READ ANY TABLE の両方	レコードなし

ユーザーが発行した文	ユーザーに付与されている権限	監査されるシステム権限	予期されるUNIFIED_AUDIT_TRAILの動作
	象外		
SELECT	SELECT ANY TABLE も READ ANY TABLE も対象外	SELECT ANY TABLE も READ ANY TABLE も対象外	レコードなし
SELECT ... FOR UPDATE	SELECT ANY TABLE	SELECT ANY TABLE	SYSTEM_PRIVILEGE_USED に挿入されるレコード: SELECT ANY TABLE
SELECT ... FOR UPDATE	SELECT ANY TABLE	READ ANY TABLE	レコードなし
SELECT ... FOR UPDATE	SELECT ANY TABLE	SELECT ANY TABLE と READ ANY TABLE の両方	SYSTEM_PRIVILEGE_USED に挿入されるレコード: SELECT ANY TABLE
SELECT ... FOR UPDATE	SELECT ANY TABLE	SELECT ANY TABLE も READ ANY TABLE も対象外	レコードなし
SELECT ... FOR UPDATE	READ ANY TABLE	SELECT ANY TABLE	レコードなし
SELECT ... FOR UPDATE	READ ANY TABLE	READ ANY TABLE	レコードなし
SELECT ... FOR UPDATE	READ ANY TABLE	SELECT ANY TABLE と READ ANY TABLE の両方	レコードなし
SELECT ... FOR UPDATE	READ ANY TABLE	SELECT ANY TABLE も READ ANY TABLE も対象外	レコードなし
SELECT ... FOR UPDATE	SELECT ANY TABLE と READ ANY TABLE の両方	SELECT ANY TABLE	SYSTEM_PRIVILEGE_USED に挿入されるレコード: SELECT ANY TABLE

ユーザーが発行した文	ユーザーに付与されている権限	監査されるシステム権限	予期されるUNIFIED_AUDIT_TRAILの動作
SELECT ... FOR UPDATE	SELECT ANY TABLEとREAD ANY TABLEの両方	READ ANY TABLE	READ ANY TABLEがアクセスに使用されたため、レコードなし
SELECT ... FOR UPDATE	SELECT ANY TABLEとREAD ANY TABLEの両方	SELECT ANY TABLEとREAD ANY TABLEの両方	SYSTEM_PRIVILEGE_USEDに挿入されるレコード: SELECT ANY TABLE
SELECT ... FOR UPDATE	SELECT ANY TABLEとREAD ANY TABLEの両方	SELECT ANY TABLEもREAD ANY TABLEも対象外	レコードなし
SELECT ... FOR UPDATE	SELECT ANY TABLEもREAD ANY TABLEも対象外	SELECT ANY TABLE	レコードなし
SELECT ... FOR UPDATE	SELECT ANY TABLEもREAD ANY TABLEも対象外	READ ANY TABLE	レコードなし
SELECT ... FOR UPDATE	SELECT ANY TABLEもREAD ANY TABLEも対象外	SELECT ANY TABLEとREAD ANY TABLEの両方	レコードなし
SELECT ... FOR UPDATE	SELECT ANY TABLEもREAD ANY TABLEも対象外	SELECT ANY TABLEもREAD ANY TABLEも対象外	レコードなし

親トピック: READ ANY TABLEおよびSELECT ANY TABLE権限の監査

27.2.9 複数層環境におけるSQL文および権限の監査

統合監査ポリシーを作成して、複数層環境のクライアントのアクティビティを監査できます。

複数層環境では、クライアントの識別情報をすべての層を通して保持できます。したがって、クライアントにかわって中間層アプリケーションにより行われたアクションを、ポリシーのAUDIT文の中でBY user句を使用することで監査できます。監査は、プロキ

シ・セッションを含むすべてのユーザー・セッションに適用されます。

中間層では、データベース・セッションでユーザーのクライアント識別情報を設定し、中間層アプリケーションでエンド・ユーザーのアクションを監査することもできます。そうすることにより、エンド・ユーザーのクライアント識別情報が監査証跡に記録されます。

次の例は、ユーザーjacksonによって発行されたSELECT TABLE文を監査する方法を示しています。

```
CREATE AUDIT POLICY tab_pol
 PRIVILEGES CREATE ANY TABLE
 ACTIONS CREATE TABLE;
AUDIT tab_pol BY jackson;
```

複数層環境でユーザー・アクティビティを監査できます。監査を開始した後で、UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューを問い合せることで、これらのアクティビティを確認できます。

図27-1に、UNIFIED_AUDIT_TRAILビューのPROXY_SESSIONID、ACTION_NAMEおよびSESSION_ID列を問い合せることで、プロキシ・ユーザーを監査する方法を示します。このシナリオでは、データベース・ユーザーとプロキシ・ユーザーの両方のアカウントがデータベースに認識されます。セッション・プーリングを使用できます。

図27-1 プロキシ・ユーザーの監査

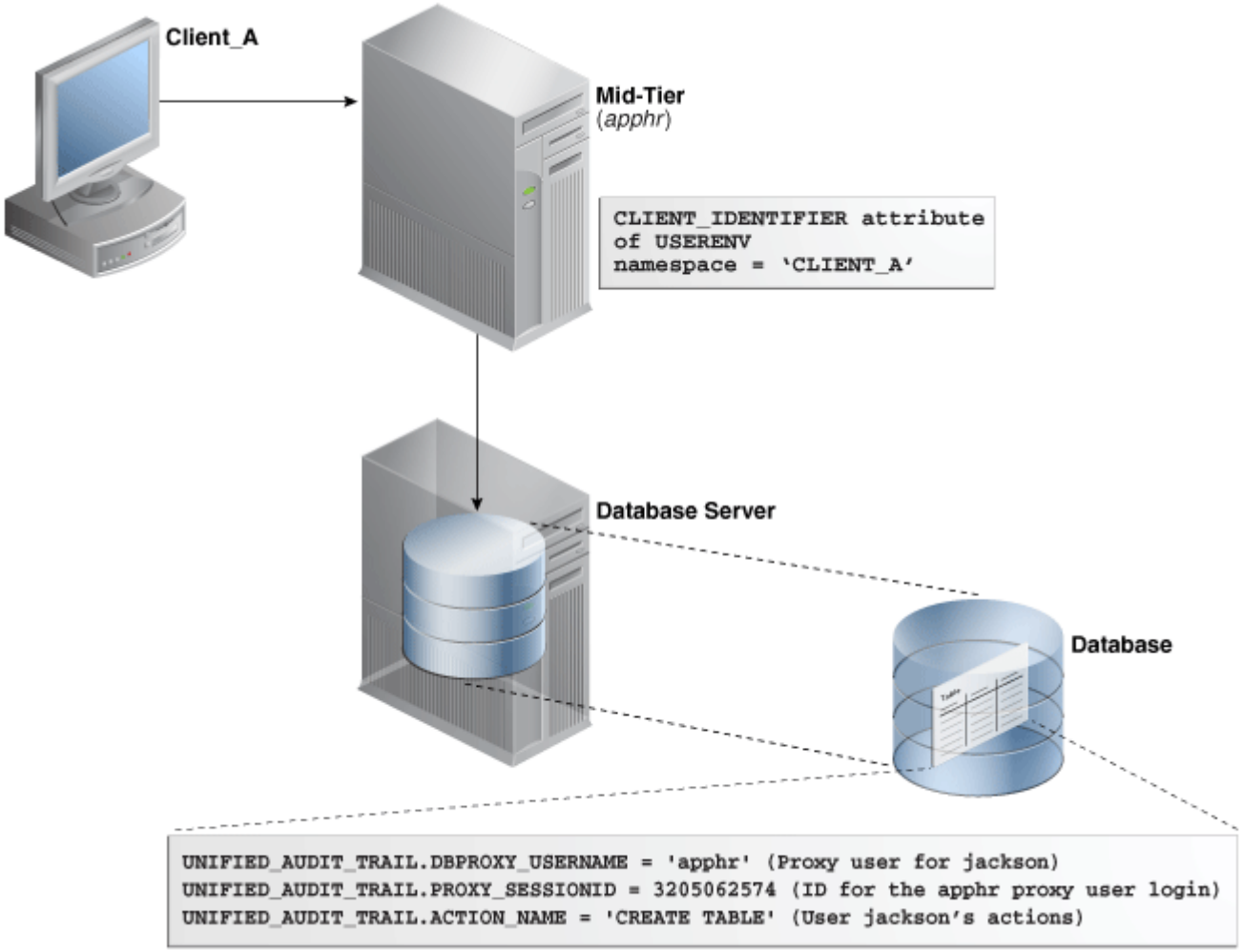
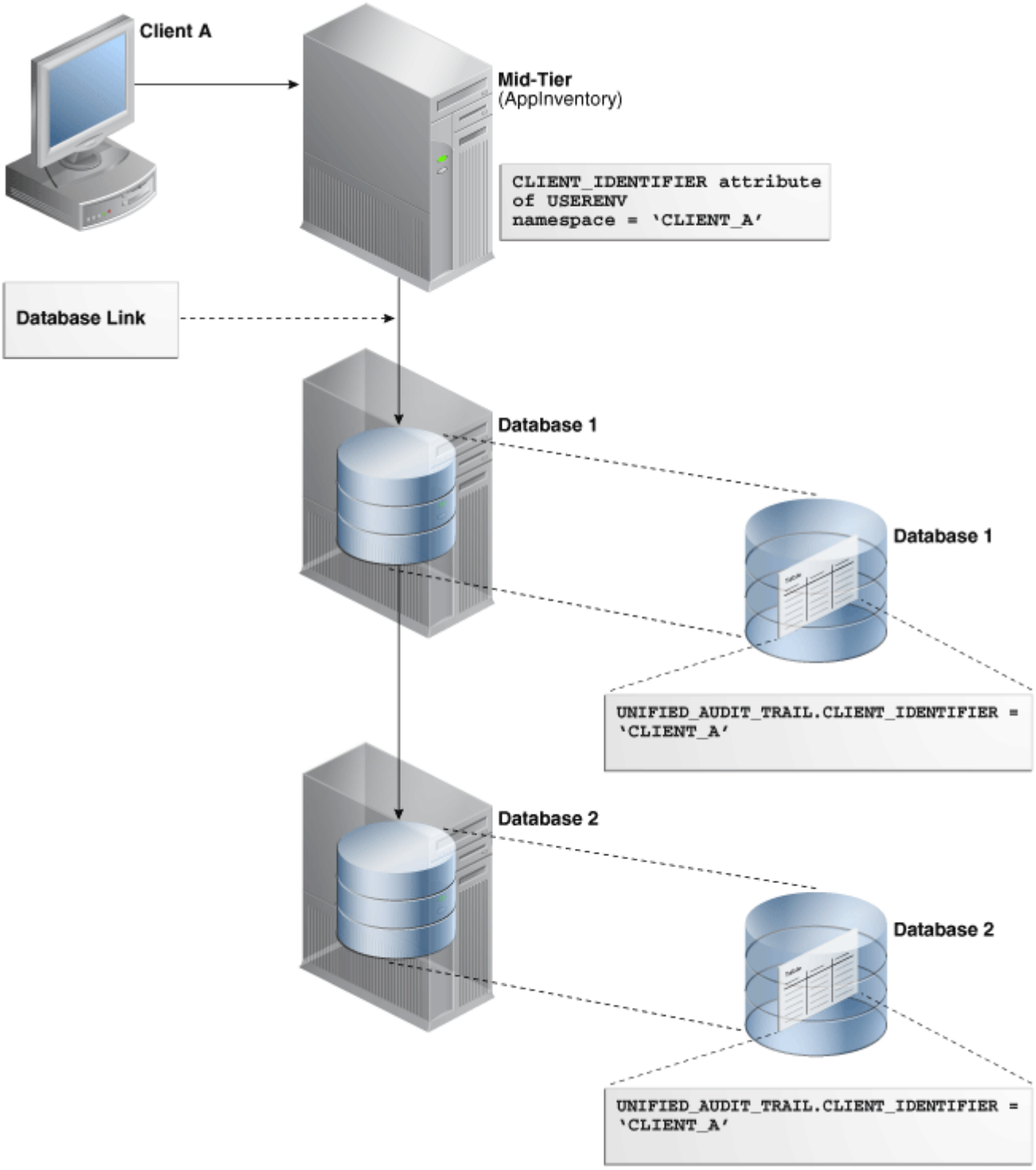


図27-2に、DBA_AUDIT_TRAILデータ・ディクショナリ・ビューのCLIENT_ID列を問い合せることで、複数のデータベース・セッションにわたるクライアント識別子情報を監査する方法を示します。この例では、クライアント識別子はCLIENT_Aに設定されています。図27-1に示すプロキシ・ユーザー/データベース・ユーザーの例と同様に、セッション・プーリングを使用できます。

図27-2 複数のセッションにわたるクライアント識別子情報の監査



関連トピック

- 複数層環境でのユーザー識別情報の保持

親トピック: 統合監査ポリシーおよびAUDIT文を使用したアクティビティの監査

27.2.10 統合監査ポリシーの条件の作成

CREATE AUDIT POLICY文を使用すると、統合監査ポリシーの条件を作成できます。

- 統合監査ポリシーの条件について
  SYS_CONTEXTネームスペースと属性のペアを使用して条件を指定する統合監査ポリシーを作成できます。
- 条件を使用した統合監査ポリシーの構成
  CREATE AUDIT POLICY文のWHEN句で、監査ポリシーの条件を定義します。

- 例: SQL*Plusへのアクセスの監査
  CREATE AUDIT POLICY文で、SQL*Plusへのアクセスを監査できます。
- 例: 特定のホストにはないアクションの監査
  CREATE AUDIT POLICY文で、特定のホストにはないアクションを監査できます。
- 例: システム全体のアクションおよびスキーマ固有のアクションの両方の監査
  CREATE AUDIT POLICY文で、システム全体のアクションおよびスキーマ固有のアクションの両方を監査できます。
- 例: 文の発生ごとの条件の監査
  CREATE AUDIT POLICY文で、条件を監査できます。
- 例: 現在の管理ユーザー・セッションの統合監査セッションID
  SYS_CONTEXTファンクションを使用してセッションIDを確認できます。
- 例: 現在の非管理ユーザー・セッションの統合監査セッションID
  SYS_CONTEXTファンクションを使用して、現在の非管理ユーザー・セッションのセッションIDを確認できます。
- 監査証跡での条件からの監査レコードの表示方法
  統合監査ポリシーからの監査レコードの条件は、監査証跡には表示されません。

親トピック: 統合監査ポリシーおよびAUDIT文を使用したアクティビティの監査

27.2.10.1 統合監査ポリシーについて

SYS_CONTEXTネームスペースと属性のペアを使用して条件を指定する統合監査ポリシーを作成できます。

たとえば、監査条件を満たす可能性のある特定のユーザーや、監査条件を満たすコンピュータ・ホストにこの監査条件を適用します。

監査条件が満たされると、Oracle Databaseによってイベントのレコードが監査されます。条件定義の一部として、監査条件が文の発生、セッションまたはデータベース・インスタンスごとに評価されるかどうかを指定する必要があります。

ノート:

監査条件では、安全なアプリケーション・コンテキストと安全でないアプリケーション・コンテキストの両方を使用できます。

親トピック: 統合監査ポリシーの条件の作成

27.2.10.2 条件を使用した統合監査ポリシーの構成

CREATE AUDIT POLICY文のWHEN句で、監査ポリシーの条件を定義します。

- 次の構文を使用して、条件を使用する統合監査ポリシーを作成します。

```
CREATE AUDIT POLICY policy_name
 action_privilege_role_audit_option
[WHEN function_operation_value_list_1 [[AND | OR]
function_operation_value_list_n]
 EVALUATE PER STATEMENT | SESSION | INSTANCE];
```

詳細は、次のとおりです。

- action_privilege_role_audit_optionは、システム・アクション、オブジェクト・アクション、権限およびロールの監査オプションを示します。
- WHENは、条件を定義します。この項では、次のコンポーネントについて説明します。

- functionは、次のタイプの関数を使用します。

  数値関数: BITAND、CEIL、FLOOR、LN POWERなど

  文字値を戻す文字関数: CONCAT、LOWER、UPPERなど

  数値を戻す文字関数: LENGTH、INSTRなど

  環境および識別子関数: SYS_CONTEXT、UIDなど。SYS_CONTEXTでは、ほとんどの場合、『Oracle Database SQL言語リファレンス』に説明されているUSERENVネームスペースを使用できます。

- operationには、AND、OR、IN、NOT IN、=、<、>、<>の演算子を使用できます。
- value_listは、テストする条件を示します。

function_operation_value_listセットごとに、ANDまたはORで区切られた追加の条件を含めることができます。

WHEN句を記述する場合は、次のガイドラインに従います。

- function operation value設定全体を一重引用符で囲みます。句内では、引用符で囲まれたそれぞれの構成要素を、2つのペアの一重引用符で囲みます。二重引用符は使用しないでください。
- WHEN条件は4000バイトを超えないでください。

- EVALUATE PERは、次のオプションを示します。
  - STATEMENTは、発生する監査可能な関連する各文の条件を評価します。
  - SESSIONは、セッション中に1回のみ条件を評価し、セッションの残りで結果をキャッシュして再利用します。Oracle Databaseでは、ポリシーが最初に使用されるときに条件を評価し、結果をUGAメモリーに後で保存します。
  - INSTANCEは、データベース・インスタンスのライフタイム中に1回のみ条件を評価します。Oracle Databaseは条件を評価した後、インスタンスの残りのライフタイムで結果をキャッシュして再利用します。SESSION評価と同様、評価は最初に必要になるときに実行され、結果は後でUGAメモリーに保存されます。

たとえば:

```
CREATE AUDIT POLICY oe_orders_pol
 ACTIONS UPDATE ON OE.ORDERS
 WHEN 'SYS_CONTEXT(''USERENV'', ''IDENTIFICATION_TYPE'') = ''EXTERNAL'''
 EVALUATE PER STATEMENT;
```

ポリシーを作成したら、AUDIT文を使用して有効にする必要があります。

関連項目:

条件で使用可能な関数の詳細は、『Oracle Database SQL言語リファレンス』を参照してください。

親トピック: 統合監査ポリシーの条件の作成

27.2.10.3 例: SQL*Plusへのアクセスの監査

CREATE AUDIT POLICY文で、SQL*Plusへのアクセスを監査できます。

例27-12に、ロールemp_adminおよびsales_adminが直接付与されているユーザーによる、SQL*Plusを使用するデータベースへのアクセスの監査方法を示します。

例27-12 SQL*Plusへのアクセスの監査

```
CREATE AUDIT POLICY logon_pol
 ACTIONS LOGON
 WHEN 'INSTR(UPPER(SYS_CONTEXT(''USERENV'', ''CLIENT_PROGRAM_NAME'')), ''SQLPLUS'') >
0'
 EVALUATE PER SESSION;
AUDIT POLICY logon_pol BY USERS WITH GRANTED ROLES emp_admin, sales_admin;
```

親トピック: 統合監査ポリシーの条件の作成

27.2.10.4 例: 特定のホストにはないアクションの監査

CREATE AUDIT POLICY文で、特定のホストにはないアクションを監査できます。

例27-13に、OE.ORDERS表に対する2つのアクション(UPDATE文とDELETE文)を監査する(ただし、ホスト名sales_24およびsales_12は監査から除外する)方法を示します。監査はセッション単位で実行され、失敗した場合にのみ監査レコードが書き込まれます。

例27-13 特定のホスト以外でのアクションの監査

```
CREATE AUDIT POLICY oe_table_audit1
 ACTIONS UPDATE ON OE.ORDERS, DELETE ON OE.ORDERS
 WHEN 'SYS_CONTEXT (''USERENV'', ''HOST'') NOT IN (''sales_24'',''sales_12'')'
 EVALUATE PER SESSION;
AUDIT POLICY oe_table_audit1 WHENEVER NOT SUCCESSFUL;
```

親トピック: 統合監査ポリシーの条件の作成

27.2.10.5 例: システム全体のアクションおよびスキーマ固有のアクションの両方の監査

CREATE AUDIT POLICY文で、システム全体のアクションおよびスキーマ固有のアクションの両方を監査できます。

例27-14に、例27-13の変化形(システム全体にわたりUPDATE文を監査する)を示します。DELETE文の監査は、OE.ORDERS表に固有です。

例27-14 システム全体のアクションおよびスキーマ固有のアクションの両方の監査

```
CREATE AUDIT POLICY oe_table_audit2
 ACTIONS UPDATE, DELETE ON OE.ORDERS
 WHEN 'SYS_CONTEXT (''USERENV'', ''HOST'') NOT IN (''sales_24'',''sales_12'')'
 EVALUATE PER SESSION;
AUDIT POLICY oe_table_audit2;
```

親トピック: 統合監査ポリシーの条件の作成

27.2.10.6 例: 文の発生ごとの条件の監査

CREATE AUDIT POLICY文で、条件を監査できます。

例27-15に、OE.ORDERS表に対するDELETE文の発生ごとに条件を監査する(ユーザーjmartinは監査から除外する)方法を示します。

例27-15 文の発生ごとの条件の監査

```
CREATE AUDIT POLICY sales_clerk_pol
 ACTIONS DELETE ON OE.ORDERS
 WHEN 'SYS_CONTEXT(''USERENV'', ''CLIENT_IDENTIFIER'') = ''sales_clerk'''
 EVALUATE PER STATEMENT;
AUDIT POLICY sales_clerk_pol EXCEPT jmartin;
```

親トピック: 統合監査ポリシーの条件の作成

27.2.10.7 例: 現在の管理ユーザー・セッションの統合監査セッションID

SYS_CONTEXT関数を使用してセッションIDを確認できます。

例27-16に、管理ユーザーについて現在のユーザー・セッションの統合監査セッションIDを確認する方法を示します。

例27-16 現在の管理ユーザー・セッションの統合監査セッションID

```
CONNECT SYS AS SYSDBA
Enter password: password
SELECT SYS_CONTEXT('USERENV', 'UNIFIED_AUDIT_SESSIONID') FROM DUAL;
```

次のような出力が表示されます。

```
SYS_CONTEXT('USERENV','UNIFIED_AUDIT_SESSIONID')
--------------------------------------------------------------------------------
2318470183
```

混合モード監査では、USERENVネームスペースのUNIFIED_AUDIT_SESSIONID値はSESSIONIDパラメータで記録される値とは異なります。このため、混合モード監査を使用し、正しい監査セッションIDを確認する場合、SESSIONIDパラメータではなくUSERENV UNIFIED_AUDIT_SESSIONIDパラメータを使用します。完全な統合監査では、SESSIONIDとUNIFIED_AUDIT_SESSIONIDの値は同じです。

親トピック: 統合監査ポリシーの条件の作成

27.2.10.8 例: 現在の非管理ユーザー・セッションの統合監査セッションID

SYS_CONTEXT関数を使用して、現在の非管理ユーザー・セッションのセッションIDを確認できます。

例27-17に、非管理ユーザーについて現在のユーザー・セッションの統合監査セッションIDを確認する方法を示します。

例27-17 現在の非管理ユーザー・セッションの統合監査セッションID

```
CONNECT mblake -- Or, CONNECT mblake@hrpdb for a PDB
Enter password: password
SELECT SYS_CONTEXT('USERENV', 'UNIFIED_AUDIT_SESSIONID') FROM DUAL;
```

次のような出力が表示されます。

```
SYS_CONTEXT('USERENV','UNIFIED_AUDIT_SESSIONID')
--------------------------------------------------------------------------------
2776921346
```

親トピック: 統合監査ポリシーの条件の作成

27.2.10.9 監査証跡での条件からの監査レコードの表示方法

統合監査ポリシーからの監査レコードの条件は、監査証跡には表示されません。

条件がtrueと評価され、レコードが書き込まれると、レコードは監査証跡に表示されます。UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューを問い合せることで、監査証跡をチェックできます。

関連トピック

- 監査ポリシーのデータ・ディクショナリ・ビュー

親トピック: 統合監査ポリシーの条件の作成

27.2.11 アプリケーション・コンテキスト値の監査

AUDIT文を使用して、アプリケーション・コンテキスト値を監査できます。

- アプリケーション・コンテキスト値の監査について
  統合監査証跡でアプリケーション・コンテキスト値を取得できます。
- アプリケーション・コンテキストの監査設定の構成
  CONTEXTキーワードを使用したAUDIT文で、アプリケーション・コンテキスト値の監査を構成します。
- アプリケーション・コンテキストの監査設定の無効化
  NOAUDIT文で、アプリケーション・コンテキストの監査設定を無効にします。
- 例: デフォルト・データベースでのアプリケーション・コンテキスト値の監査
  AUDIT CONTEXT NAMESPACE文で、アプリケーション・コンテキスト値を監査できます。
- 例: Oracle Label Securityのアプリケーション・コンテキスト値の監査
  AUDIT CONTEXT NAMESPACE文で、Oracle Label Securityのアプリケーション・コンテキスト値を監査できます。
- 監査証跡での監査対象のアプリケーション・コンテキストの表示方法
  UNIFIED_AUDIT_POLICIESデータ・ディクショナリ・ビューはアプリケーション・コンテキストの監査イベントを表示します。

親トピック: 統合監査ポリシーおよびAUDIT文を使用したアクティビティの監査

27.2.11.1 アプリケーション・コンテキスト値の監査について

統合監査証跡でアプリケーション・コンテキスト値を取得できます。

この機能では、監査対象の文の実行中にデータベース・アプリケーションによって設定されるアプリケーション・コンテキスト値を取得します。

Oracle Label Securityを監査する場合は、この機能により、データベース監査証跡のセッション・ラベルのアクティビティが取得されます。監査証跡では、指定したコンテキストと属性値のペアで取得されたすべての値が記録されます。

アプリケーション・コンテキストの監査設定または監査ポリシーには、セッションの静的セマンティクがあります。つまり、ユーザーに対して新しいポリシーが有効になると、後続のユーザー・セッションにこのコマンドの結果が表示されます。セッションが確立されると、ポリシーとコンテキストの設定がロードされ、このセッションは後続のAUDIT文の影響を受けません。

マルチテナント環境では、アプリケーション・コンテキストの監査ポリシーは現在のPDBのみに適用されます。

関連項目:

- アプリケーション・コンテキストの詳細は、「アプリケーション・コンテキストを使用したユーザー情報の取得」を参照してください。
- マルチテナント環境での統合監査ポリシーまたはAUDIT設定
- Oracle Label Securityの詳細は、『Oracle Label Security管理者ガイド』を参照してください。

親トピック: アプリケーション・コンテキスト値の監査

27.2.11.2 アプリケーション・コンテキストの監査設定の構成

CONTEXTキーワードを使用したAUDIT文で、アプリケーション・コンテキスト値の監査を構成します。

このタイプの監査では、統合監査ポリシーを作成しません。

- 次の構文を使用して、アプリケーション・コンテキスト値の監査を構成します。

```
AUDIT CONTEXT NAMESPACE context_name1 ATTRIBUTES attribute1 [, attribute2]
 [, CONTEXT NAMESPACE context_name2 ATTRIBUTES attribute1 [, attribute2]]
 [BY user_list];
```

詳細は、次のとおりです。

- `context_name1`: オプションでCONTEXTの名前と属性のペアを1つ追加できます。
- `user_list`は、データベース・ユーザー・アカウントのオプションのリストです。複数の名前はカンマで区切ります。この設定を省略すると、Oracle Databaseによって、すべてのユーザーにアプリケーション・コンテキスト・ポリシーが構成されます。各ユーザーがログインすると、関連するすべてのアプリケーション・コンテキストと属性のリストがユーザー・セッションでキャッシュされます。

たとえば:

```
AUDIT CONTEXT NAMESPACE clientcontext3 ATTRIBUTES module, action,
 CONTEXT NAMESPACE ols_session_labels ATTRIBUTES ols_pol1, ols_pol3
 BY appuser1, appuser2;
```

現在構成されているアプリケーション・コンテキストの監査設定のリストを検索するには、AUDIT_UNIFIED_CONTEXTSデータ・ディクショナリ・ビューを問い合せます。

親トピック: アプリケーション・コンテキスト値の監査

27.2.11.3 アプリケーション・コンテキストの監査設定の無効化

NOAUDIT文で、アプリケーション・コンテキストの監査設定を無効にします。

- アプリケーション・コンテキストの監査設定を無効にするには、NOAUDIT文でネームスペースと属性設定を指定します。属性は任意の順序で入力できます(対応するAUDIT CONTEXT文で使用される順序に一致させる必要はありません)。

たとえば:

```
NOAUDIT CONTEXT NAMESPACE client_context ATTRIBUTES module,
 CONTEXT NAMESPACE ols_session_labels ATTRIBUTES ols_pol1, ols_pol3
 BY USERS WITH GRANTED ROLES emp_admin;
```

現在監査されているアプリケーション・コンテキストを検索するには、AUDIT_UNIFIED_CONTEXTSデータ・ディクショナリ・ビューを問い合せます。

親トピック: アプリケーション・コンテキスト値の監査

27.2.11.4 例: デフォルト・データベースでのアプリケーション・コンテキスト値の監査

AUDIT CONTEXT NAMESPACE文で、アプリケーション・コンテキスト値を監査できます。

例27-18に、ユーザーappuser1による、module属性とaction属性のclientcontextアプリケーション値を監査する方法を示します。

例27-18 デフォルト・データベースでのアプリケーション・コンテキスト値の監査

```
AUDIT CONTEXT NAMESPACE clientcontext ATTRIBUTES module, action
BY appuser1;
```

親トピック: アプリケーション・コンテキスト値の監査

27.2.11.5 例: Oracle Label Securityのアプリケーション・コンテキスト値の監査

AUDIT CONTEXT NAMESPACE文で、Oracle Label Securityのアプリケーション・コンテキスト値を監査できます。

例27-19に、属性ols_pol1およびols_pol2について、Oracle Label Securityのアプリケーション・コンテキストols_session_labelsを監査する方法を示します。

例27-19 Oracle Label Securityのアプリケーション・コンテキスト値の監査

```
AUDIT CONTEXT NAMESPACE ols_session_labels ATTRIBUTES ols_pol1, ols_pol2;
```

親トピック: アプリケーション・コンテキスト値の監査

27.2.11.6 監査証跡での監査対象のアプリケーション・コンテキストの表示方法

UNIFIED_AUDIT_POLICIESデータ・ディクショナリ・ビューはアプリケーション・コンテキストの監査イベントを表示します。

UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューのAPPLICATION_CONTEXTS列には、アプリケーション・コンテキストの監査データが表示されます。アプリケーション・コンテキストは、セミコロンで区切られた値のリストとして表示されます。

たとえば:

```
SELECT APPLICATION_CONTEXTS FROM UNIFIED_AUDIT_TRAIL
 WHERE UNIFIED_AUDIT_POLICIES = 'app_audit_pol';
APPLICATION_CONTEXTS
--------------------------------------------------------
CLIENT_CONTEXT.APPROLE=MANAGER;E2E_CONTEXT.USERNAME=PSMITH
```

親トピック: アプリケーション・コンテキスト値の監査

27.2.12 Oracle Database Real Application Securityイベントの監査

CREATE AUDIT POLICY文を使用して、Oracle Database Real Application Securityイベントを監査できます。

- Oracle Database Real Application Securityイベントの監査について
  Oracle Database Real Application Securityイベントを監査するには、AUDIT_ADMINロールが必要です。
- Oracle Database Real Application Securityの監査可能なイベント
  Oracle Databaseには、CREATE USER、UPDATE USERなど、Real Application Securityの監査可能なイベントが用意されています。
- Oracle Database Real Application Securityのユーザー、権限およびロールの監査イベント
  統合監査証跡では、ユーザー、権限およびロールについてOracle Database Real Application Securityのイベントを取得できます。
- Oracle Database Real Application Securityのセキュリティ・クラスおよびACLの監査イベント
  統合監査証跡では、Oracle Database Real Application Securityのセキュリティ・クラスおよびACLの監査イベントを取得できます。
- Oracle Database Real Application Securityのセッションの監査イベント
  統合監査証跡では、Oracle Database Real Application Securityのセッションの監査イベントを取得できます。
- Oracle Database Real Application SecurityのALLイベント
  統合監査証跡では、Oracle Database Real Application SecurityのALLイベントを取得できます。
- Oracle Database Real Application Securityの統合監査ポリシーの構成
  CREATE AUDIT POLICY文で、Oracle Real Application Securityの統合監査ポリシーを作成できます。
- 例: Real Application Securityのユーザー・アカウントの変更の監査
  CREATE AUDIT POLICY文で、Real Application Securityのユーザー・アカウントの変更を監査できます。

- 例: Real Application Securityの統合監査ポリシーでの条件の使用
  CREATE AUDIT POLICY文で、Real Application Securityの統合監査ポリシーの条件を設定できます。
- 監査証跡でのOracle Database Real Application Securityイベントの表示方法
  DBA_XS_AUDIT_TRAILデータ・ディクショナリ・ビューはOracle Real Application Securityの監査イベントを表示します。

親トピック: 統合監査ポリシーおよびAUDIT文を使用したアクティビティの監査

27.2.12.1 Oracle Database Real Application Securityイベントの監査について

Oracle Database Real Application Securityイベントを監査するには、AUDIT_ADMINロールが必要です。

監査証跡にアクセスするには、Real Application Security固有の列がXS_で始まるUNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューを問い合せます。Oracle Real Application Securityポリシーが有効なときに作成される、内部的に生成されたVPD述語に関する監査情報を確認する場合は、RLS_INFO列を問い合せることができます。

Real Application Security固有のビューは次のとおりです。

- DBA_XS_AUDIT_TRAILでは、監査されたReal Application Securityイベントの詳細情報が提供されます。
- DBA_XS_AUDIT_POLICY_OPTIONSでは、Real Application Securityの統合監査ポリシーに定義された監査オプションを記述します。
- DBA_XS_ENB_AUDIT_POLICIESでは、Real Application Securityの統合監査ポリシーが有効なユーザーのリストを表示します。

関連項目:

- アプリケーション・コンテキスト値の監査

  Oracle Database Real Application Securityの事前定義の監査ポリシー
- RLS_INFO列の出力をフォーマットする方法の詳細は、Oracle Virtual Private Databaseの述語の監査を参照してください
- Oracle Database Real Application Securityの詳細は、『Oracle Database Real Application Security管理者および開発者ガイド』を参照してください。

親トピック: Oracle Database Real Application Securityイベントの監査

27.2.12.2 Oracle Database Real Application Securityの監査可能なイベント

Oracle Databaseには、CREATE USER、UPDATE USERなど、Real Application Securityの監査可能なイベントが用意されています。

監査可能なReal Application Securityイベントのリストを検索するには、次のように、AUDITABLE_SYSTEM_ACTIONSデータ・ディクショナリ・ビューのCOMPONENTおよびNAME列を問い合せます。

```
SELECT NAME FROM AUDITABLE_SYSTEM_ACTIONS WHERE COMPONENT = 'XS';
NAME
-------------
CREATE USER
UPDATE USER
DELETE USER
...
```

関連トピック

- Oracle Database Real Application Securityのユーザー、権限およびロールの監査イベント
- Oracle Database Real Application Securityのセキュリティ・クラスおよびACLの監査イベント
- Oracle Database Real Application Securityのセッションの監査イベント
- Oracle Database Real Application SecurityのALLイベント

親トピック: Oracle Database Real Application Securityイベントの監査

27.2.12.3 Oracle Database Real Application Securityのユーザー、権限およびロールの監査イベント

統合監査証跡では、ユーザー、権限およびロールについてOracle Database Real Application Securityのイベントを取得できます。

表27-4では、これらのイベントについて説明します。

表27-4 Oracle Database Real Application Securityのユーザー、権限およびロールの監査イベント

監査イベント	説明
CREATE USER	XS_PRINCIPAL.CREATE_USER プロシージャを使用して Oracle Database Real Application Security のユーザー・アカウントを作成します。
UPDATE USER	次のプロシージャを使用して Oracle Database Real Application Security のユーザー・アカウントを更新します。 ● XS_PRINCIPAL.SET_EFFECTIVE_DATES ● XS_PRINCIPAL.SET_USER_DEFAULT_ROLES_ALL ● XS_PRINCIPAL.SET_USER_SCHEMA ● XS_PRINCIPAL.SET_GUID ● XS_PRINCIPAL.SET_USER_STATUS ● XS_PRINCIPAL.SET_DESCRIPTION
DELETE USER	XS_PRINCIPAL.DELETE_PRINCIPAL プロシージャを使用して Oracle Database Real Application Security のユーザー・アカウントを削除します。
AUDIT_GRANT_PRIVILEGE	GRANT_SYSTEM_PRIVILEGE 権限を監査します
AUDIT_REVOKE_PRIVILEGE	REVOKE_SYSTEM_PRIVILEGE 権限を監査します
CREATE ROLE	XS_PRINCIPAL.CREATE_ROLE プロシージャを使用して Oracle Database Real Application Security のロールを作成します。

監査イベント	説明
UPDATE ROLE	次のプロシージャを使用して Oracle Database Real Application Security のロールを更新します。 ● XS_PRINCIPAL.SET_DYNAMIC_ROLE_SCOPE ● XS_PRINCIPAL.SET_DYNAMIC_ROLE_DURATION ● XS_PRINCIPAL.SET_EFFECTIVE_DATES ● XS_PRINCIPAL.SET_ROLE_DEFAULT
DELETE ROLE	XS_PRINCIPAL.DELETE_ROLE プロシージャを使用して Oracle Database Real Application Security のロールを削除します
GRANT ROLE	XS_PRINCIPAL.GRANT_ROLES プロシージャを使用して Oracle Database Real Application Security のロールを付与します。
REVOKE ROLE	XS_PRINCIPAL.REVOKE_ROLES プロシージャを使用して Oracle Database Real Application Security のロールを取り消し、付与されたすべてのロールを XS_PRINCIPAL.REVOKE_ALL_GRANTED_ROLES プロシージャを使用して取り消します。
ADD PROXY	XS_PRINCIPAL.ADD_PROXY_USER プロシージャを使用して Oracle Database Real Application Security のプロキシ・ユーザー・アカウントを追加し、XS_PRINCIPAL.ADD_PROXY_TO_SCHEMA プロシージャを使用してデータベース・ユーザーにプロキシを追加します
REMOVE PROXY	XS_PRINCIPAL.REMOVE_PROXY_USER、XS_PRINCIPAL.REMOVE_ALL_PROXY_USERS および XS_PRINCIPAL.REMOVE_PROXY_FROM_SCHEMA PROCEDURES を使用して、Oracle Database Real Application Security のプロキシ・ユーザー・アカウントを削除します。
SET USER PASSWORD	XS_PRINCIPAL.SET_PASSWORD プロシージャを使用して Oracle Database Real Application Security のユーザー・アカウントのパスワードを設定します。
SET USER VERIFIER	XS_PRINCIPAL.SET_VERIFIER プロシージャを使用して Oracle Database Real Application Security のプロキシ・ユーザー・アカウントのベリファイアを設定します。

親トピック: Oracle Database Real Application Securityイベントの監査

27.2.12.4 Oracle Database Real Application Securityのセキュリティ・クラスおよびACLの監査イベント

統合監査証跡では、Oracle Database Real Application Securityのセキュリティ・クラスおよびACLの監査イベントを取得できます。

表27-5では、これらのイベントについて説明します。

表27-5 Oracle Database Real Application Securityのセキュリティ・クラスおよびACLの監査イベント

監査イベント	説明
CREATE SECURITY CLASS	XS_SECURITY_CLASS.CREATE_SECURITY_CLASS プロシージャを使用してセキュリティ・クラスを作成します。
UPDATE SECURITY CLASS	次のプロシージャを使用してセキュリティ・クラスを作成します。 ● XS_SECURITY_CLASS.SET_DEFAULT_ACL ● XS_SECURITY_CLASS.ADD_PARENTS ● XS_SECURITY_CLASS.REMOVE_ALL_PARENTS ● XS_SECURITY_CLASS.REMOVE_PARENTS ● XS_SECURITY_CLASS.ADD_PRIVILEGES ● XS_SECURITY_CLASS.REMOVE_ALL_PRIVILEGES ● XS_SECURITY_CLASS.ADD_IMPLIED_PRIVILEGES ● XS_SECURITY_CLASS.REMOVE_IMPLIED_PRIVILEGES ● XS_SECURITY_CLASS.REMOVE_ALL_IMPLIED_PRIVILEGES ● XS_SECURITY_CLASS.SET_DESCRIPTION
DELETE SECURITY CLASS	XS_SECURITY_CLASS.DELETE_SECURITY_CLASS プロシージャを使用してセキュリティ・クラスを削除します。
CREATE ACL	XS_ACL.CREATE_ACL プロシージャを使用してアクセス制御リスト(ACL)を作成します。
UPDATE ACL	次のプロシージャを使用して ACL を更新します。 ● XS_ACL.APPEND_ACES ● XS_ACL.REMOVE_ALL_ACES ● XS_ACL.SET_SECURITY_CLASS

監査イベント	説明
	• XS_ACL.SET_PARENT_ACL • XS_ACL.ADD_ACL_PARAMETER • XS_ACL.REMOVE_ALL_ACL_PARAMETERS • XS_ACL.REMOVE_ACL_PARAMETER • XS_ACL.SET_DESCRIPTION
DELETE ACL	XS_ACL.DELETE_ACL プロシージャを使用して ACL を削除します。
CREATE DATA SECURITY-	XS_DATA_SECURITY.CREATE_DATA_SECURITY プロシージャを使用してデータ・セキュリティ・ポリシーを作成します。
UPDATE DATA SECURITY	次のプロシージャを使用してデータ・セキュリティ・ポリシーを更新します。 • XS_DATA_SECURITY.CREATE_ACL_PARAMETER • XS_DATA_SECURITY.DELETE_ACL_PARAMETER • XS_DATA_SECURITY.SET_DESCRIPTION
DELETE DATA SECURITY	XS_DATA_SECURITY.DELETE_DATA_SECURITY プロシージャを使用してデータ・セキュリティ・ポリシーを削除します。
ENABLE DATA SECURITY	XS_DATA_SECURITY.ENABLE_OBJECT_POLICY プロシージャを使用して、データベース表またはビューの拡張可能データ・セキュリティを有効にします。
DISABLE DATA SECURITY	XS_DATA_SECURITY.DISABLE_XDS プロシージャを使用して、データベース表またはビューの拡張可能データ・セキュリティを無効にします。

親トピック: Oracle Database Real Application Securityイベントの監査

27.2.12.5 Oracle Database Real Application Securityのセッションの監査イベント

統合監査証跡では、Oracle Database Real Application Securityのセッションの監査イベントを取得できます。

表27-4では、これらのイベントについて説明します。

表27-6 Oracle Database Real Application Securityのセッションの監査イベント

監査イベント	説明
CREATE SESSION	DBMS_XS_SESSIONS.CREATE_SESSION プロシージャを使用してセッションを作成します。

監査イベント	説明
DESTROY SESSION	DBMS_XS_SESSIONS.DESTROY_SESSION プロシージャを使用してセッションを破棄します。
CREATE SESSION NAMESPACE	DBMS_XS_SESSIONS.CREATE_NAMESPACE プロシージャを使用してネームスペースを作成します。
DELETE SESSION NAMESPACE	DBMS_XS_SESSIONS.DELETE_NAMESPACE プロシージャを使用してネームスペースを削除します。
CREATE NAMESPACE ATTRIBUTE	DBMS_XS_SESSIONS.CREATE_ATTRIBUTE プロシージャを使用してネームスペース属性を作成します。
SET NAMESPACE ATTRIBUTE	DBMS_XS_SESSIONS.SET_ATTRIBUTE プロシージャを使用してネームスペース属性を設定します。
GET NAMESPACE ATTRIBUTE	DBMS_XS_SESSIONS.GET_ATTRIBUTE プロシージャを使用してネームスペース属性を取得します。
DELETE NAMESPACE ATTRIBUTE	DBMS_XS_SESSIONS.DELETE_ATTRIBUTE プロシージャを使用してネームスペース属性を削除します。
CREATE NAMESPACE TEMPLATE	XS_NS_TEMPLATE.CREATE_NS_TEMPLATE プロシージャを使用してネームスペース属性を作成します。
UPDATE NAMESPACE TEMPLATE	次のプロシージャを使用してネームスペース属性を更新します。 ● XS_NS_TEMPLATE.SET_HANDLER ● XS_NS_TEMPLATE.ADD_ATTRIBUTES ● XS_NS_TEMPLATE.REMOVE_ALL_ATTRIBUTES ● XS_NS_TEMPLATE.REMOVE_ATTRIBUTES ● XS_NS_TEMPLATE.SET_DESCRIPTION
DELETE NAMESPACE TEMPLATE	XS_NS_TEMPLATE.DELETE_NS_TEMPLATE プロシージャを使用してネームスペースを削除します。
ADD GLOBAL CALLBACK	DBMS_XS_SESSIONS.ADD_GLOBAL_CALLBACK プロシージャを使用してグローバル・コールバックを追加します。

監査イベント	説明
DELETE GLOBAL CALLBACK	DBMS_XS_SESSIONS.DELETE_GLOBAL_CALLBACK プロシージャを使用してグローバル・コールバックを削除します。
ENABLE GLOBAL CALLBACK	DBMS_XS_SESSIONS.ENABLE_GLOBAL_CALLBACK プロシージャを使用してグローバル・コールバックを有効にします。
SET COOKIE	DBMS_XS_SESSIONS.SET_SESSION_COOKIE プロシージャを使用してセッション Cookie を設定します。
SET INACTIVE TIMEOUT	DBMS_XS_SESSIONS.SET_INACTIVITY_TIMEOUT プロシージャを使用して、無効なセッションのタイムアウト時間を設定します。
SWITCH USER	DBMS_XS_SESSIONS.SWITCH_USER プロシージャを使用して、指定したユーザーで新しく初期化されたセキュリティ・コンテキストに現在の軽量ユーザー・セッションのセキュリティ・コンテキストを設定します。
ASSIGN USER	DBMS_XS_SESSIONS.ASSIGN_USER プロシージャを使用して、指定したユーザーに対して動的ロールを 1 つ以上割り当てまたは削除します。
ENABLE ROLE	DBMS_XS_SESSIONS.ENABLE_ROLE プロシージャを使用して、軽量ユーザー・セッションのロールを有効にします。
DISABLE ROLE	DBMS_XS_SESSIONS.DISABLE_ROLE プロシージャを使用して、軽量ユーザー・セッションのロールを無効にします。

親トピック: Oracle Database Real Application Securityイベントの監査

27.2.12.6 Oracle Database Real Application SecurityのALLイベント

統合監査証跡では、Oracle Database Real Application SecurityのALLイベントを取得できます。

表27-7では、これらのイベントについて説明します。

表27-7 Oracle Database Real Application SecurityのALLイベント

監査イベント	説明
ALL	Real Application Security のすべてのアクションを取得します。

親トピック: Oracle Database Real Application Securityイベントの監査

27.2.12.7 Oracle Database Real Application Securityの統合監査ポリシーの構成

CREATE AUDIT POLICY文で、Oracle Real Application Securityの統合監査ポリシーを作成できます。

- 次の構文を使用して、Oracle Database Real Application Securityの統合監査ポリシーを作成します。

```
CREATE AUDIT POLICY policy_name
 ACTIONS COMPONENT=XS component_action1 [, action2];
```

たとえば:

```
CREATE AUDIT POLICY audit_ras_pol
 ACTIONS COMPONENT=XS SWITCH USER, DISABLE ROLE;
```

条件を含む場合など、より複雑なポリシーを作成できます。ポリシーを作成したら、AUDIT文を使用して有効にする必要があります。

関連トピック

- 統合監査ポリシーの作成の構文

親トピック: Oracle Database Real Application Securityイベントの監査

27.2.12.8 例: Real Application Securityのユーザー・アカウントの変更の監査

CREATE AUDIT POLICY文で、Real Application Securityのユーザー・アカウントの変更を監査できます。

例27-20に、ユーザーbhurstによるユーザー切替えとロール無効化の試みを監査する方法を示します。

例27-20 Real Application Securityのユーザー・アカウント変更の監査

```
CREATE AUDIT POLICY ras_users_pol
 ACTIONS COMPONENT=XS SWITCH USER, DISABLE ROLE;
AUDIT POLICY ras_users_pol BY bhurst;
```

親トピック: Oracle Database Real Application Securityイベントの監査

27.2.12.9 例: Real Application Securityの統合監査ポリシーでの条件の使用

CREATE AUDIT POLICY文で、Real Application Securityの統合監査ポリシーの条件を設定できます。

例27-21に、nemosityコンピュータ・ホストからのアクションのみに監査を適用する、Real Application Securityの統合監査ポリシーの作成方法を示します。

例27-21 Real Application Securityの統合監査ポリシーでの条件の使用

```
CREATE AUDIT POLICY ras_acl_pol
 ACTIONS DELETE ON OE.CUSTOMERS
 ACTIONS COMPONENT=XS CREATE ACL, UPDATE ACL, DELETE ACL
 WHEN 'SYS_CONTEXT(''USERENV'', ''HOST'') = ''nemosity'''
 EVALUATE PER INSTANCE;
AUDIT POLICY ras_acl_pol BY pfitch;
```

親トピック: Oracle Database Real Application Securityイベントの監査

27.2.12.10 監査証跡でのOracle Database Real Application Securityイベントの表示方法

DBA_XS_AUDIT_TRAILデータ・ディクショナリ・ビューはOracle Real Application Securityの監査イベントを表示します。

次の例では、Real Application Security固有のビューのDBA_XS_AUDIT_TRAILを問い合せます。

```
SELECT XS_USER_NAME FROM DBA_XS_AUDIT_TRAIL
WHERE XS_ENABLED_ROLE = 'CLERK';
XS_USER_NAME
-------------
USER2
```

親トピック: Oracle Database Real Application Securityイベントの監査

27.2.13 Oracle Recovery Managerイベントの監査

CREATE AUDIT POLICY文を使用して、Oracle Recovery Managerイベントを監査できます。

- Oracle Recovery Managerイベントの監査について
  UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューでは、Oracle Recovery Managerの監査イベントをRMAN_columnに自動的に格納します。
- Oracle Recovery Managerの統合監査証跡イベント
  統合監査証跡では、Oracle Recovery Managerのイベントを取得できます。
- 監査証跡でのOracle Recovery Managerの監査イベントの表示方法
  UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューはOracle Recovery Managerの監査イベントを表示します。

親トピック: 統合監査ポリシーおよびAUDIT文を使用したアクティビティの監査

27.2.13.1 Oracle Recovery Managerイベントの監査について

UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューでは、Oracle Recovery Managerの監査イベントをRMAN_columnに自動的に格納します。

他のOracle Databaseコンポーネントと異なり、Oracle Recovery Managerのイベントには統合監査ポリシーを作成しません。

ただし、UNIFIED_AUDIT_TRAILビューを問い合せて、これらのイベントを確認するには、AUDIT_ADMINまたはAUDIT_VIEWERロールが必要です。SYSBACKUPまたはSYSDBA管理権限がない場合は、V$RMAN_STATUSやV$RMAN_BACKUP_JOB_DETAILSなどのビューを問い合せて、Recovery Managerジョブに関する追加情報を検索できます。

関連項目:

Oracle Databaseバックアップおよびリカバリ・アドバンスト・ユーザーズ・ガイド

親トピック: Oracle Recovery Managerイベントの監査

27.2.13.2 Oracle Recovery Managerの統合監査証跡イベント

統合監査証跡では、Oracle Recovery Managerのイベントを取得します。

表27-8では、これらのイベントについて説明します。

表27-8 UNIFIED_AUDIT_TRAILビューのOracle Recovery Manager列

Recovery Managerの列	説明
RMAN_SESSION_RECID	Recovery Manager のセッション識別子。この列と RMAN_SESSION_STAMP 列を組み合せることで、Recovery Manager ジョブを一意に識別します。Recovery Manager のセッション ID は、Recovery Manager ジョブを識別する制御ファイルの RECID 値です。(Recovery Manager のセッション ID はユー

Recovery Managerの列	説明
	ザー・セッション ID と同じではありません。)
RMAN_SESSION_STAMP	セッションのタイムスタンプ。この列と RMAN_SESSION_RECID 列を組み合せることで、Recovery Manager ジョブを識別します。
RMAN_OPERATION	ジョブによって実行される Recovery Manager 操作。Recovery Manager のセッション内の操作ごとに行が 1 つ追加されます。たとえば、バックアップ・ジョブには、BACKUP が RMAN_OPERATION 値として含まれます。
RMAN_OBJECT_TYPE	Recovery Manager のセッションに含まれるオブジェクトのタイプ。これには、次のいずれかの値が含まれます。Recovery Manager のセッションがこれらの複数を満たさない場合は、プリファレンスが次の順序(リストの上から下)で指定されます。 1. DB FULL (Database Full)は、データベースの全体バックアップを示します。 2. RECVR AREA は、高速リカバリ領域を示します。 3. DB INCR (Database Incremental)は、データベースの増分バックアップを示します。 4. DATAFILE FULL は、データ・ファイルの全体バックアップを示します。 5. DATAFILE INCR は、データ・ファイルの増分バックアップを示します。 6. ARCHIVELOG は、アーカイブ REDO ログ・ファイルを示します。 7. CONTROLFILE は、制御ファイルを示します。 8. SPFILE は、サーバー・パラメータ・ファイルを示します。 9. BACKUPSET は、バックアップ・ファイルを示します。
RMAN_DEVICE_TYPE	Recovery Manager のセッションに関連付けられているデバイス。この列は、DISK、SBT (システム・バックアップ・テープ)または* (アスタリスク)になります。アスタリスクは、複数のデバイスを示します。ほとんどの場合、値は DISK および SBT になります。

親トピック: Oracle Recovery Managerイベントの監査

27.2.13.3 監査証跡でのOracle Recovery Managerの監査イベントの表示方法

UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューはOracle Recovery Managerの監査イベントを表示します。

表27-8に、Oracle Recovery Manager固有の監査データを検索するために問合せ可能な、UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューの列を示します。

たとえば:

```
SELECT RMAN_OPERATION FROM UNIFIED_AUDIT_TRAIL
WHERE RMAN_OBJECT_TYPE = 'DB FULL';
RMAN_OPERATION
---------------
BACKUP
```

親トピック: Oracle Recovery Managerイベントの監査

27.2.14 Oracle Database Vaultイベントの監査

Oracle Database Vault環境で、CREATE AUDIT POLICY文を使用してDatabase Vaultアクティビティを監査できます。

- Oracle Database Vaultイベントの監査について
  すべての統合監査と同様、Oracle Database Vaultのイベントを監査するには、AUDIT_ADMINロールが必要です。
- Oracle Database Vaultの監査者
  Oracle Database Vaultの監査対象ユーザーには、Database Vault管理者、およびアクティビティがDatabase Vaultの規定ポリシーに影響するユーザーが含まれます。
- Oracle Database Vaultの統合監査証跡イベントについて
  Oracle Database Vault環境の監査証跡は、すべての構成変更やDatabase Vaultポリシーの変更の試行を取得します。
- Oracle Database Vaultのレルムの監査イベント
  統合監査証跡によってOracle Database Vaultのレルムのイベントが取得されます。
- Oracle Database Vaultのルール・セットおよびルールの監査イベント
  統合監査証跡では、Oracle Database Vaultのルール・セットおよびルールの監査イベントを取得できます。
- Oracle Database Vaultのコマンド・ルールの監査イベント
  統合監査証跡では、Oracle Database Vaultのコマンド・ルールの監査イベントを取得できます。
- Oracle Database Vaultのファクタの監査イベント
  統合監査証跡では、Oracle Database Vaultのファクタ・イベントを取得できます。
- Oracle Database Vaultのセキュア・アプリケーション・ロールの監査イベント
  統合監査証跡では、Oracle Database Vaultのセキュア・アプリケーション・ロールの監査イベントを取得できます。
- Oracle Database Vault Oracle Label Securityの監査イベント
  統合監査証跡では、Oracle Database Vault Oracle Label Securityの監査イベントを取得できます。
- Oracle Database Vault Oracle Data Pumpの監査イベント
  統合監査証跡では、Oracle Database Vault Oracle Data Pumpの監査イベントを取得できます。
- Oracle Database Vaultの有効および無効な監査イベント
  統合監査証跡では、Oracle Database Vaultの有効および無効な監査イベントを取得できます。
- Oracle Database Vaultの統合監査ポリシーの構成
  CREATE AUDIT POLICY文のACTIONSおよびACTIONS COMPONENT句で、Oracle Database Vaultイベントの統合監査ポリシーを作成できます。
- 例: Oracle Database Vaultのレルムの監査
  CREATE AUDIT POLICY文で、Oracle Database Vaultのレルムを監査できます。

- 例: Oracle Database Vaultのルール・セットの監査
  CREATE AUDIT POLICY文で、Oracle Database Vaultのルール・セットを監査できます。
- 例: 2つのOracle Database Vaultイベントの監査
  CREATE AUDIT POLICY文で、複数のOracle Database Vaultイベントを監査できます。
- 例: Oracle Database Vaultのファクタの監査
  CREATE AUDIT POLICY文で、Oracle Database Vaultのファクタを監査できます。
- 監査証跡でのOracle Database Vaultの監査イベントの表示方法
  UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューはOracle Database Vaultの監査イベントを表示します。

親トピック: 統合監査ポリシーおよびAUDIT文を使用したアクティビティの監査

27.2.14.1 Oracle Database Vaultイベントの監査について

すべての統合監査と同様、Oracle Database Vaultのイベントを監査するには、AUDIT_ADMINロールが必要です。

Oracle Database Vaultの統合監査ポリシーを作成するには、CREATE AUDIT POLICY文のCOMPONENT句をDVに設定し、Rule Set Failureなどのアクションおよびルール・セットの名前などのオブジェクトを指定します。

監査証跡にアクセスする場合は、次のビューを問い合せることができます。

- UNIFIED_AUDIT_TRAIL
- AUDSYS.DV$CONFIGURATION_AUDIT
- AUDSYS.DV$ENFORCEMENT_AUDIT

UNIFIED_AUDIT_TRAILビューでは、Oracle Database Vault固有の列はDV_で始まります。
UNIFIED_AUDIT_TRAILビューを問い合せるには、AUDIT_VIEWERロールが必要です。

これらのビューに加えて、Database Vaultレポートでは、Database Vault固有の統合監査ポリシーの結果も取得します。

関連項目:

- DVSYSおよびLBACSYSスキーマに対するOracle Database Vaultの事前定義の統合監査ポリシー
- Oracle Database Vaultの監査ポリシーの詳細は、『Oracle Database Vault管理者ガイド』を参照してください。

親トピック: Oracle Database Vaultイベントの監査

27.2.14.2 Oracle Database Vaultの監査者

Oracle Database Vaultの監査対象ユーザーには、Database Vault管理者、およびアクティビティがDatabase Vaultの規定ポリシーに影響するユーザーが含まれます。

これらのユーザーは次のとおりです。

- Database Vault管理者。Oracle Database Vaultに対して行われるすべての構成変更は、強制的に監査されます。監査は、レルム、ファクタ、コマンド・ルール、ルール・セット、ルールの作成、変更、削除などのアクティビティを取得します。AUDSYS.DV$CONFIGURATION_AUDITデータ・ディクショナリ・ビューでは、Database Vault管理者によって実行された構成変更が取得されます。
- アクティビティがOracle Database Vaultの規定ポリシーに影響するユーザー。
  AUDSYS.DV$ENFORCEMENT_AUDITデータ・ディクショナリ・ビューでは、規定関連の監査を取得します。

関連項目:

AUDSYS.DV$CONFIGURATION_AUDITおよびAUDSYS.DV$ENFORCEMENT_AUDITデータ・ディクショナリ・ビューの詳細は、Oracle Database Vault管理者ガイドを参照してください。

親トピック: Oracle Database Vaultイベントの監査

27.2.14.3 Oracle Database Vaultの統合監査証跡イベントについて

Oracle Database Vault環境の監査証跡は、すべての構成変更やDatabase Vaultポリシーの変更の試行を取得します。

これは、既存のDatabase Vaultのポリシーに対するユーザーの違反も取得します。

次の種類のOracle Database Vaultイベントを監査できます。

- Oracle Database Vaultのポリシーに対するすべての構成変更または変更の試行。Database Vault管理者による変更および権限のないユーザーによる試行の両方が取得されます。
- 既存のDatabase Vaultのポリシーに対するユーザーの違反。たとえば、ユーザーが作業時間以外に特定のスキーマ表にアクセスできないようにするポリシーを作成すると、監査証跡によって、このアクティビティが取得されます。

親トピック: Oracle Database Vaultイベントの監査

27.2.14.4 Oracle Database Vaultのレルムの監査イベント

統合監査証跡によってOracle Database Vaultのレルムのイベントが取得されます。

表27-9では、これらのイベントについて説明します。

表27-9 Oracle Database Vaultのレルムの監査イベント

監査イベント	説明
CREATE_REALM	DVSYS.DBMS_MACADM.CREATE_REALM プロシージャを使用してレルムを作成します。
UPDATE_REALM	DVSYS.DBMS_MACADM.UPDATE_REALM プロシージャを使用してレルムを更新します。
RENAME_REALM	DVSYS.DBMS_MACADM.RENAME_REALM プロシージャを使用してレルムの名前を変更します。
DELETE_REALM	DVSYS.DBMS_MACADM.DELETE_REALM プロシージャを使用してレルムを削除します。
DELETE_REALM_CASCADE	DVSYS.DBMS_MACADM.DELETE_REALM_CASCADE プロシージャを使用して、レルムおよび関連する Database Vault の構成情報を削除します。
ADD_AUTH_TO_REALM	DVSYS.DBMS_MACADM.ADD_AUTH_TO_REALM プロシージャを

監査イベント	説明
	使用して、レルムに認可を追加します。
DELETE_AUTH_FROM_REALM	DVSYS.DBMS_MACADM.DELETE_AUTH_FROM_REALM プロシージャを使用して、レルムから認可を削除します。
UPDATE_REALM_AUTH	DVSYS.DBMS_MACADM.UPDATE_REALM_AUTHORIZATION プロシージャを使用して、レルム認可を更新します。
ADD_OBJECT_TO_REALM	DVSYS.DBMS_MACADM.ADD_AUTH_TO_REALM プロシージャを使用して、レルム認可にオブジェクトを追加します。
DELETE_OBJECT_FROM_REALM	DVSYS.DBMS_MACADM.DELETE_OBJECT_FROM_REALM プロシージャを使用して、レルム認可からオブジェクトを削除します。

親トピック: Oracle Database Vaultイベントの監査

27.2.14.5 Oracle Database Vaultのルール・セットおよびルールの監査イベント

統合監査証跡では、Oracle Database Vaultのルール・セットおよびルールの監査イベントを取得できます。

表27-10では、これらのイベントについて説明します。

表27-10 Oracle Database Vaultのルール・セットおよびルールの監査イベント

監査イベント	説明
CREATE_RULE_SET	DVSYS.DBMS_MACADM.CREATE_RULE_SET プロシージャを使用してルール・セットを作成します。
UPDATE_RULE_SET	DVSYS.DBMS_MACADM.UPDATE_RULE_SET プロシージャを使用してルール・セットを更新します。
RENAME_RULE_SET	DVSYS.DBMS_MACADM.RENAME_RULE_SET プロシージャを使用して、ルール・セットの名前を変更します。
DELETE_RULE_SET	DVSYS.DBMS_MACADM.DELETE_RULE_SET プロシージャを使用して、ルール・セットを削除します。
ADD_RULE_TO_RULE_SET	DVSYS.DBMS_MACADM.ADD_RULE_TO_RULE_SET プロシージャを使用して、既存のルール・セットにルールを追加します。
DELETE_RULE_FROM_RULE_SET	DVSYS.DBMS_MACADM.DELETE_RULE_FROM_RULE_SET プ

監査イベント	説明
	ロシージャを使用して、既存のルール・セットからルールを削除します。
CREATE_RULE	DVSYS.DBMS_MACADM.CREATE_RULE プロシージャを使用してルールを作成します。
UPDATE_RULE	DVSYS.DBMS_MACADM.UPDATE_RULE プロシージャを使用してルールを更新します。
RENAME_RULE	DVSYS.DBMS_MACADM.RENAME_RULE プロシージャを使用して、ルールの名前を変更します。
DELETE_RULE	DVSYS.DBMS_MACADM.DELETE_RULE プロシージャを使用して、ルールを削除します。
SYNC_RULES	DVSYS.DBMS_MACADM.SYNC_RULES プロシージャを使用して、Oracle Database Vault のルールとアドバンスト・キューイング・ルール・エンジンを同期します。

親トピック: Oracle Database Vaultイベントの監査

27.2.14.6 Oracle Database Vaultのコマンド・ルールの監査イベント

統合監査証跡では、Oracle Database Vaultのコマンド・ルールの監査イベントを取得できます。

表27-11では、これらのイベントについて説明します。

表27-11 Oracle Database Vaultのコマンド・ルールの監査イベント

監査イベント	説明
CREATE_COMMAND_RULE	DVSYS.DBMS_MACADM.CREATE_COMMAND_RULE プロシージャを使用して、コマンド・ルールを作成します。
DELETE_COMMAND_RULE	DVSYS.DBMS_MACADM.DELETE_COMMAND_RULE プロシージャを使用して、コマンド・ルールを削除します。
UPDATE_COMMAND_RULE	DVSYS.DBMS_MACADM.UPDATE_COMMAND_RULE プロシージャを使用して、コマンド・ルールを更新します。

親トピック: Oracle Database Vaultイベントの監査

27.2.14.7 Oracle Database Vaultのファクタの監査イベント

統合監査証跡では、Oracle Database Vaultのファクタ・イベントを取得できます。

表27-12では、これらのイベントについて説明します。

表27-12 Oracle Database Vaultのファクタの監査イベント

監査イベント	説明
CREATE_FACTOR_TYPE	DVSYS.DBMS_MACADM.CREATE_FACTOR_TYPE プロシージャを使用して、ファクタ・タイプを作成します。
DELETE_FACTOR_TYPE	DVSYS.DBMS_MACADM.DELETE_FACTOR_TYPE プロシージャを使用して、ファクタ・タイプを削除します。
UPDATE_FACTOR_TYPE	DVSYS.DBMS_MACADM.UPDATE_FACTOR_TYPE プロシージャを使用して、ファクタ・タイプを更新します。
RENAME_FACTOR_TYPE	DVSYS.DBMS_MACADM.RENAME_FACTOR_TYPE プロシージャを使用して、ファクタ・タイプの名前を変更します。
CREATE_FACTOR	DVSYS.DBMS_MACADM.CREATE_FACTOR プロシージャを使用して、ファクタを作成します。
UPDATE_FACTOR	DVSYS.DBMS_MACADM.UPDATE_FACTOR プロシージャを使用して、ファクタを更新します。
DELETE_FACTOR	DVSYS.DBMS_MACADM.DELETE_FACTOR プロシージャを使用して、ファクタを削除します。
RENAME_FACTOR	DVSYS.DBMS_MACADM.RENAME_FACTOR プロシージャを使用して、ファクタの名前を変更します。
ADD_FACTOR_LINK	DVSYS.DBMS_MACADM.ADD_FACTOR_LINK プロシージャを使用して、2 つのファクタの親子関係を指定します。
DELETE_FACTOR_LINK	DVSYS.DBMS_MACADM.DELETE_FACTOR_LINK プロシージャを使用して、2 つのファクタの親子関係を削除します。
ADD_POLICY_FACTOR	DVSYS.DBMS_MACADM.ADD_POLICY_FACTOR プロシージャを使用して、ファクタのラベルをポリシーの Oracle Label Security ラベルに含めることを指定します。
DELETE_POLICY_FACTOR	DBMS_MACADM.DELETE_POLICY_FACTOR プロシージャを使用して、ポリシーの Oracle Label Security ラベルとの関連付けから

監査イベント	説明
	ファクタ・ラベルを削除します。
CREATE_IDENTITY	DVSYS.DBMS_MACADM.CREATE_IDENTITY プロシージャを使用して、ファクタ・アイデンティティを作成します。
UPDATE_IDENTITY	DVSYS.DBMS_MACADM.UPDATE_IDENTITY プロシージャを使用して、ファクタ・アイデンティティを更新します。
CHANGE_IDENTITY_FACTOR	DVSYS.DBMS_MACADM.CHANGE_IDENTITY_FACTOR プロシージャを使用して、アイデンティティを異なるファクタに関連付けます。
CHANGE_IDENTITY_VALUE	DVSYS.DBMS_MACADM.CHANGE_IDENTITY_VALUE プロシージャを使用して、アイデンティティの値を更新します。
DELETE_IDENTITY	DVSYS.DBMS_MACADM.DELETE_IDENTITY プロシージャを使用して、既存のファクタ・アイデンティティを削除します。
CREATE_IDENTITY_MAP	DVSYS.DBMS_MACADM.CREATE_IDENTITY_MAP プロシージャを使用して、ファクタ・アイデンティティ・マップを作成します。
DELETE_IDENTITY_MAP	DVSYS.DBMS_MACADM.DELETE_IDENTITY_MAP プロシージャを使用して、ファクタ・アイデンティティ・マップを削除します。
CREATE_DOMAIN_IDENTITY	DVSYS.DBMS_MACADM.CREATE_DOMAIN_IDENTITY プロシージャを使用して、Oracle Database Real Application Clusters データベース・ノードをドメイン・ファクタ・アイデンティティに追加し、Oracle Label Security ポリシーに従ってラベルを付けます。
DROP_DOMAIN_IDENTITY	DVSYS.DBMS_MACADM.DROP_DOMAIN_IDENTITY プロシージャを使用して、ドメイン・ファクタ・アイデンティティから Oracle RAC ノードを削除します。

親トピック: Oracle Database Vaultイベントの監査

27.2.14.8 Oracle Database Vaultのセキュア・アプリケーション・ロールの監査イベント

統合監査証跡では、Oracle Database Vaultのセキュア・アプリケーション・ロールの監査イベントを取得できます。

表27-13では、これらのイベントについて説明します。

表27-13 Oracle Database Vaultのセキュア・アプリケーション・ロールの監査イベント

監査イベント	説明
CREATE_ROLE	DVSYS.DBMS_MACADM.CREATE_ROLE プロシージャを使用して、Oracle Database Vault のセキュア・アプリケーション・ロールを作成します。
DELETE_ROLE	DVSYS.DBMS_MACADM.DELETE_ROLE プロシージャを使用して、Oracle Database Vault のセキュア・アプリケーション・ロールを削除します。
UPDATE_ROLE	DVSYS.DBMS_MACADM.UPDATE_ROLE プロシージャを使用して、Oracle Database Vault のセキュア・アプリケーション・ロールを更新します。
RENAME_ROLE	DVSYS.DBMS_MACADM.RENAME_ROLE プロシージャを使用して、Oracle Database Vault のセキュア・アプリケーション・ロールの名前を変更します。

親トピック: Oracle Database Vaultイベントの監査

27.2.14.9 Oracle Database Vault Oracle Label Securityの監査イベント

統合監査証跡では、Oracle Database Vault Oracle Label Securityの監査イベントを取得できます。

表27-14では、これらのイベントについて説明します。

表27-14 Oracle Database Vault Oracle Label Securityの監査イベント

監査イベント	説明
CREATE_POLICY_LABEL	DVSYS.DBMS_MACADM.CREATE_POLICY_LABEL プロシージャを使用して、Oracle Label Security のポリシー・ラベルを作成します。
DELETE_POLICY_LABEL	DVSYS.DBMS_MACADM.DELETE_POLICY_LABEL プロシージャを使用して、Oracle Label Security のポリシー・ラベルを削除します。
CREATE_MAC_POLICY	DVSYS.DBMS_MACADM.CREATE_MAC_POLICY を使用して、ファクタのラベルまたは Oracle Label Security セッション・ラベルを算出する際にラベルのマージに使用されるアルゴリズムを指定します。
UPDATE_MAC_POLICY	DVSYS.DBMS_MACADM.UPDATE_MAC_POLICY プロシージャを使用して、Oracle Label Security のラベルのマージ・アルゴリズムを変更します。

監査イベント	説明
DELETE_MAC_POLICY_CASCADE	DVSYS.DBMS_MACADM.DELETE_MAC_POLICY_CASCADE プロシージャを使用して、Oracle Label Security ポリシーに関連する Oracle Database Vault のすべてのオブジェクトを削除します。

親トピック: Oracle Database Vaultイベントの監査

27.2.14.10 Oracle Database Vault Oracle Data Pumpの監査イベント

統合監査証跡では、Oracle Database Vault Oracle Data Pumpの監査イベントを取得できます。

表27-15では、これらのイベントについて説明します。

表27-15 Oracle Database Vault Oracle Data Pumpの監査イベント

監査イベント	説明
AUTHORIZE_DATAPUMP_USER	DVSYS.DBMS_MACADM.AUTHORIZE_DATAPUMP_USER プロシージャを使用して、Oracle Data Pump ユーザーに権限を付与します。
UNAUTHORIZE_DATAPUMP_USER	DVSYS.DBMS_MACADM.UNAUTHORIZE_DATAPUMP_USER プロシージャを使用して、Oracle Data Pump ユーザーを認可から削除します。

親トピック: Oracle Database Vaultイベントの監査

27.2.14.11 Oracle Database Vaultの有効および無効な監査イベント

統合監査証跡では、Oracle Database Vaultの有効および無効な監査イベントを取得できます。

表27-16では、これらのイベントについて説明します。

表27-16 Oracle Database Vaultの有効および無効な監査イベント

イベント	説明
ENABLE_EVENT	DBMS_MACADM.ENABLE_EVENT
DISABLE_EVENT	DBMS_MACADM.DISABLE_EVENT

親トピック: Oracle Database Vaultイベントの監査

27.2.14.12 Oracle Database Vaultの統合監査ポリシーの構成

CREATE AUDIT POLICY文のACTIONSおよびACTIONS COMPONENT句で、Oracle Database Vaultイベントの統合監査ポリシーを作成できます。

- 次の構文を使用して、Oracle Database Vaultの統合監査ポリシーを作成します。

```
CREATE AUDIT POLICY policy_name
 ACTIONS action1 [,action2 ]
```

```
ACTIONS COMPONENT= DV DV_action ON DV_object [,DV_action2 ON DV_object2]
```

詳細は、次のとおりです。

- DV_actionは次のいずれかになります。
  - レルム関連のアクション:

    Realm Violationは、レルム違反を監査します(たとえば、認可されていないユーザーが、レルムで保護されたオブジェクトにアクセスしようとしたとき)。

    Realm Successは、レルムで保護されたオブジェクトに、認可されたユーザーが正常にアクセスしたときに監査します。

    Realm Accessは、Realm ViolationとRealm Successの両方のケースを監査します。つまり、アクセスが成功したか失敗したかに関係なく、レルム・アクセスが試行されるたびに監査します。
  - ルール・セット関連のアクション: Rule Set Failure、Rule Set Success、Rule Set Eval
  - ファクタ関連のアクション: Factor Error、Factor Null、Factor Validate Error、Factor Validate False、Factor Trust Level Null、Factor Trust Level Neg、Factor All
- DV_objectsは次のいずれかになります。
  - Realm_Name
  - Rule_Set_Name
  - Factor_Name

オブジェクトが小文字または大/小文字が混在して作成された場合は、DV_objectsを二重引用符で囲みます。オブジェクトをすべて大文字で作成した場合は、引用符を省略できます。

たとえば、Database Vaultアカウント管理レルムのレルム違反を監査するには、次のようにします。

```
CREATE AUDIT POLICY audit_dv
 ACTIONS CREATE TABLE, SELECT
 ACTIONS COMPONENT=DV Realm Violation ON "Database Vault Account Management";
```

条件を含む場合など、より複雑なポリシーを作成できます。ポリシーを作成したら、AUDIT文を使用して有効にする必要があります。

親トピック: Oracle Database Vaultイベントの監査

27.2.14.13 例: Oracle Database Vaultのレルムの監査

CREATE AUDIT POLICY文で、Oracle Database Vaultのレルムを監査できます。

例27-22に、HRスキーマに対するレルム違反の監査方法を示します。

例27-22 レルム違反の監査

```
CREATE AUDIT POLICY dv_realm_hr
 ACTIONS SELECT, UPDATE, DELETE
 ACTIONS COMPONENT=DV Realm Violation ON "HR Schema Realm";
AUDIT POLICY dv_realm_hr EXCEPT psmith;
```

親トピック: Oracle Database Vaultイベントの監査

27.2.14.14 例: Oracle Database Vaultのルール・セットの監査

CREATE AUDIT POLICY文で、Oracle Database Vaultのルール・セットを監査できます。

例: Oracle Database Vaultのルール・セットの監査に、Can Maintain Accounts/Profileルール・セットの監査方法を示します。DV_ACCTMGRロールがあるため、ユーザー・アカウントおよびユーザー・プロファイルを管理する権限があるユーザーdbv_acctmgrは、この監査ポリシーから除外されます。

例27-23 ルール・セットの監査

```
CREATE AUDIT POLICY dv_rule_set_accts
 ACTIONS CREATE USER, ALTER USER, ALTER PROFILE
 ACTIONS COMPONENT=DV RULE SET FAILURE ON "Can Maintain Accounts/Profile";
AUDIT POLICY dv_rule_set_accts EXCEPT dbv_acctmgr;
```

親トピック: Oracle Database Vaultイベントの監査

27.2.14.15 例: 2つのOracle Database Vaultイベントの監査

CREATE AUDIT POLICY文で、複数のOracle Database Vaultイベントを監査できます。

例27-24に、レルム違反およびルール・セット失敗の監査方法を示します。

例27-24 2つのOracle Database Vaultイベントの監査

```
CREATE AUDIT POLICY audit_dv
 ACTIONS CREATE TABLE, SELECT
 ACTIONS COMPONENT=DV REALM VIOLATION ON "Oracle Enterprise Manager", Rule Set
 Failure ON "Allow Sessions";
AUDIT POLICY audit_dv EXCEPT psmith;
```

親トピック: Oracle Database Vaultイベントの監査

27.2.14.16 例: Oracle Database Vaultのファクタの監査

CREATE AUDIT POLICY文で、Oracle Database Vaultのファクタを監査できます。

例27-25に、1つのファクタで2つのタイプのエラーを監査する方法を示します。

例27-25 Oracle Database Vaultのファクタ設定の監査

```
CREATE AUDIT POLICY audit_dv_factor
 ACTIONS COMPONENT=DV FACTOR ERROR ON "Database_Domain", Factor Validate Error ON
"Client_IP";
AUDIT POLICY audit_dv_factor;
```

親トピック: Oracle Database Vaultイベントの監査

27.2.14.17 監査証跡でのOracle Database Vaultの監査イベントの表示方法

UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューはOracle Database Vaultの監査イベントを表示します。

UNIFIED_AUDIT_TRAILビューのDV_* 列は、Oracle Database Vault固有の監査データを示します。

たとえば:

```
SELECT DV_RULE_SET_NAME FROM UNIFIED_AUDIT_TRAIL
WHERE ACTION_NAME = 'UPDATE';
DV_RULE_SET_NAME
----------------------
Allow System Parameters
```

親トピック: Oracle Database Vaultイベントの監査

27.2.15 Oracle Label Securityイベントの監査

Oracle Label Security環境で、CREATE AUDIT POLICY文を使用してOracle Label Securityアクティビティを監査できます。

- Oracle Label Securityイベントの監査について
  すべての統合監査と同様、Oracle Label Security (OLS)のイベントを監査するには、AUDIT_ADMINロールが必要です。
- Oracle Label Securityの統合監査証跡イベント
  統合監査証跡では、Oracle Label Securityの監査イベントを取得できます。
- Oracle Label Securityの監査可能なユーザー・セッション・ラベル
  ORA_OLS_SESSION_LABELSアプリケーション・コンテキストで、各Oracle Databaseイベントのユーザー・セッション・ラベルの使用状況を取得できます。
- Oracle Label Securityの統合監査ポリシーの構成
  CREATE AUDIT POLICY文のACTIONSおよびACTIONS COMPONENT句を使用して、Oracle Label Securityイベントの監査ポリシーを作成できます。
- 例: Oracle Label Securityのセッション・ラベル属性の監査
  AUDIT CONTEXT NAMESPACE文で、Oracle Label Securityのセッション・ラベル属性を監査できます。
- 例: Oracle Label Securityポリシーからのユーザーの除外
  CREATE AUDIT POLICY文で、ポリシーからユーザーを除外できます。
- 例: Oracle Label Securityのポリシー・アクションの監査
  CREATE AUDIT POLICY文で、Oracle Label Securityのポリシー・アクションを監査できます。
- 例: 監査済のOLSセッション・ラベルの問合せ
  UNIFIED_AUDIT_TRAIL問合せでLBACSYS.ORA_GET_AUDITED_LABELファンクションを使用して、監査済Oracle Label Securityセッション・ラベルを確認できます。
- 監査証跡でのOracle Label Securityの監査イベントの表示方法
  UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューはOracle Label Securityの監査イベントを表示します。

親トピック: 統合監査ポリシーおよびAUDIT文を使用したアクティビティの監査

27.2.15.1 Oracle Label Securityイベントの監査について

すべての統合監査と同様、Oracle Label Security (OLS)のイベントを監査するには、AUDIT_ADMINロールが必要です。

Oracle Label Securityの統合監査ポリシーを作成するには、CREATE AUDIT POLICY文のCOMPONENT句をOLSに設定する必要があります。

ユーザー・セッション情報を監査するには、AUDIT文を使用して、アプリケーション・コンテキスト値を監査します。

監査証跡にアクセスするには、UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューを問い合せます。このビューには、名前がOLS_で始まるOracle Label Security固有の列が含まれます。Oracle Label Securityポリシーを表に適用するときに作成される、内部的に生成されたVPD述語に関する監査情報を特定する場合は、RLS_INFO列を問い合せることができます。

関連項目:

- RLS_INFO列の出力をフォーマットする方法の詳細は、Oracle Virtual Private Databaseの述語の監査を参照してください
- Oracle Label Securityの詳細は、『Oracle Label Security管理者ガイド』を参照してください。

親トピック: Oracle Label Securityイベントの監査

27.2.15.2 Oracle Label Securityの統合監査証跡イベント

統合監査証跡では、Oracle Label Securityの監査イベントを取得できます。

監査可能なOracle Label Securityのイベントのリストを検索するには、次のように、AUDITABLE_SYSTEM_ACTIONSデータ・ディクショナリ・ビューのCOMPONENT列およびNAME列を問い合せます。

たとえば:

```
SELECT NAME FROM AUDITABLE_SYSTEM_ACTIONS WHERE COMPONENT = 'Label Security';
NAME
-------------
CREATE POLICY
ALTER POLICY
DROP POLICY
...
```

表27-17に、Oracle Label Securityの監査イベントを示します。

表27-17 Oracle Label Securityの監査イベント

監査イベント	説明
CREATE POLICY	SA_SYSDBA.CREATE_POLICY プロシージャを使用して、Oracle Label Security のポリシーを作成します。
ALTER POLICY	SA_SYSDBA.ALTER_POLICY プロシージャを使用して、Oracle Label Security のポリシーを変更します。
DROP POLICY	SA_SYSDBA.DROP_POLICY プロシージャを使用して、Oracle Label Security のポリシーを削除します。
APPLY POLICY	SA_POLICY_ADMIN.APPLY_TABLE_POLICY プロシージャを使用して表ポリシーを適用するか、SA_POLICY_ADMIN.APPLY_SCHEMA_POLICY プロシージャを使用してスキーマ・ポリシーを適用します。
REMOVE POLICY	SA_POLICY_ADMIN.REMOVE_TABLE_POLICY プロシージャを使用して表ポリシーを削除するか、SA_POLICY_ADMIN.REMOVE_SCHEMA_POLICY プロシージャを使用してスキーマ・ポリシーを削除します。
SET AUTHORIZATION	Oracle Label Security の権限を含む Oracle Label Security のすべての認可を含み、ユーザーまたは信頼できるストアド・プロシージャのいずれかにラベルを使用します。SET AUTHORIZATION イベントに対応する PL/SQL プロシージャは、SA_USER_ADMIN.SET_USER_LABELS、

監査イベント	説明
	SA_USER_ADMIN.SET_USER_PRIVS および SA_USER_ADMIN.SET_PROG_PRIVS です。
PRIVILEGED ACTION	Oracle Label Security の権限のユーザーを必要とするアクションを含みます。これらのアクションは、ログオン、SA_SESSION.SET_ACCESS_PROFILE の実行、および信頼できるストアド・プロシージャの起動です。
ENABLE POLICY	次のプロシージャを使用して、Oracle Label Security のポリシーを有効にします。 ● SA_SYSDBA.ENABLE_POLICY: ポリシーで保護されている表およびスキーマにアクセス制御を施行します。 ● SA_POLICY_ADMIN.ENABLE_TABLE_POLICY: 指定した表で Oracle Label Security ポリシーを有効にします。 ● SA_POLICY_ADMIN.ENABLE_SCHEMA_POLICY: 指定したスキーマのすべての表で Oracle Label Security ポリシーを有効にします。
DISABLE POLICY	次のプロシージャを使用して、Oracle Label Security のポリシーを無効にします。 ● SA_SYSDBA.DISABLE_POLICY: Oracle Label Security ポリシーの規定を無効にします。 ● SA_POLICY_ADMIN.DISABLE_TABLE_POLICY: 指定した表で Oracle Label Security ポリシーの規定を無効にします。 ● SA_POLICY_ADMIN.DISABLE_SCHEMA_POLICY: 指定したスキーマのすべての表で Oracle Label Security ポリシーの規定を無効にします。
SUBSCRIBE OID	SA_POLICY_ADMIN.POLICY_SUBSCRIBE プロシージャを使用して、Oracle Internet Directory 対応の Oracle Label Security ポリシーをサブスクライブします。
UNSUBSCRIBE OID	SA_POLICY_ADMIN.POLICY_UNSUBSCRIBE プロシージャを使用して、Oracle Internet Directory 対応の Oracle Label Security ポリシーのサブスクライブを取り消します。
CREATE DATA LABEL	SA_LABEL_ADMIN.CREATE_LABEL プロシージャを使用して、Oracle

監査イベント	説明
	Label Security のデータ・ラベルを作成します。CREATE DATA LABEL は、LBACSYS.TO_DATA_LABEL ファンクションにも対応します。
ALTER DATA LABEL	SA_LABEL_ADMIN.ALTER_LABEL プロシージャを使用して、Oracle Label Security のデータ・ラベルを変更します。
DROP DATA LABEL	SA_LABEL_ADMIN.DROP_LABEL プロシージャを使用して、Oracle Label Security のデータ・ラベルを削除します。
CREATE LABEL COMPONENT	次のプロシージャを使用して、Oracle Label Security のコンポーネントを作成します。 ● レベル: SA_COMPONENTS.CREATE_LEVEL ● 区分: SA_COMPONENTS.CREATE_COMPARTMENT ● グループ: SA_COMPONENTS.CREATE_GROUP
ALTER LABEL COMPONENTS	次のプロシージャを使用して、Oracle Label Security のコンポーネントを変更します。 ● レベル: SA_COMPONENTS.ALTER_LEVEL ● 区分: SA_COMPONENTS.ALTER_COMPARTMENT ● グループ: SA_COMPONENTS.ALTER_GROUP および SA_COMPONENTS.ALTER_GROUP_PARENT
DROP LABEL COMPONENTS	次のプロシージャを使用して、Oracle Label Security のコンポーネントを削除します。 ● レベル: SA_COMPONENTS.DROP_LEVEL ● 区分: SA_COMPONENTS.DROP_COMPARTMENT ● グループ: SA_COMPONENTS.DROP_GROUP
ALL	Oracle Label Security のすべてのアクションの監査を有効にします。

親トピック: Oracle Label Securityイベントの監査

27.2.15.3 Oracle Label Securityの監査可能なユーザー・セッション・ラベル

ORA_OLS_SESSION_LABELSアプリケーション・コンテキストで、各Oracle Databaseイベントのユーザー・セッション・ラベルの使用状況を取得できます。

このアプリケーション・コンテキストで使用される属性は、Oracle Label Securityのポリシーを示します。

構文は、「アプリケーション・コンテキストの監査設定の構成」で説明されているアプリケーション・コンテキストの監査に使用される構文と同じです。たとえば:

```
AUDIT CONTEXT NAMESPACE ORA_SESSION_LABELS ATTRIBUTES policy1, policy2;
```

セッション・ラベルの記録はユーザー・セッション固有ではないため、BY user_list句は、Oracle Label Securityのアプリケーション・コンテキストには不要です。

ユーザー・セッション・ラベル情報の監査を無効にするには、NOAUDIT文を使用します。たとえば、ポリシーpolicy1およびpolicy2の監査を停止するには、次の文を入力します。

```
NOAUDIT CONTEXT NAMESPACE ORA_SESSION_LABELS ATTRIBUTES policy1, policy2;
```

親トピック: Oracle Label Securityイベントの監査

27.2.15.4 Oracle Label Securityの統合監査ポリシーの構成

CREATE AUDIT POLICY文のACTIONSおよびACTIONS COMPONENT句を使用して、Oracle Label Securityイベントの監査ポリシーを作成できます。

- 次の構文を使用して、Oracle Label Securityの統合監査ポリシーを作成します。

```
CREATE AUDIT POLICY policy_name
 ACTIONS action1 [,action2 ]
 ACTIONS COMPONENT=OLS component_action1 [, action2];
```

たとえば:

```
CREATE AUDIT POLICY audit_ols
 ACTIONS SELECT ON OE.ORDERS
 ACTIONS COMPONENT=OLS ALL;
```

条件を含む場合など、より複雑なポリシーを作成できます。ポリシーを作成したら、AUDIT文を使用して有効にする必要があります。

関連トピック

- 統合監査ポリシーの作成の構文

親トピック: Oracle Label Securityイベントの監査

27.2.15.5 例: Oracle Label Securityのセッション・ラベル属性の監査

AUDIT CONTEXT NAMESPACE文で、Oracle Label Securityのセッション・ラベル属性を監査できます。

例27-26に、Oracle Label Securityのポリシーusr_pol1およびusr_pol2のORA_OLS_SESSION_LABELSアプリケーション・コンテキスト属性を監査する方法を示します。

例27-26 Oracle Label Securityのセッション・ラベル属性の監査

```
AUDIT CONTEXT NAMESPACE ORA_SESSION_LABELS ATTRIBUTES usr_pol1, usr_pol2;
```

親トピック: Oracle Label Securityイベントの監査

27.2.15.6 例: Oracle Label Securityポリシーからのユーザーの除外

CREATE AUDIT POLICY文で、ポリシーからユーザーを除外できます。

例27-27に、ユーザーols_mgrからのアクションを除外する統合監査ポリシーの作成方法を示します。

例27-27 Oracle Label Securityポリシーからのユーザーの除外

```
CREATE AUDIT POLICY auth_ols_audit_pol
 ACTIONS SELECT ON HR.EMPLOYEES
 ACTIONS COMPONENT=OLS DROP POLICY, DISABLE POLICY;
AUDIT POLICY auth_ols_audit_pol EXCEPT ols_mgr;
```

親トピック: Oracle Label Securityイベントの監査

27.2.15.7 例: Oracle Label Securityのポリシー・アクションの監査

CREATE AUDIT POLICY文で、Oracle Label Securityのポリシー・アクションを監査できます。

例27-28に、DROP POLICY、DISABLE POLICY、UNSUBSCRIBE OIDイベント、およびHR.EMPLOYEES表に対するUPDATE文とDELETE文を監査する方法を示します。次に、ポリシーがHRおよびLBACSYSユーザーに適用され、監査アクションが成功すると、監査レコードが統合監査証跡に書き込まれます。

例27-28 Oracle Label Securityのポリシー・アクションの監査

```
CREATE AUDIT POLICY generic_audit_pol
 ACTIONS UPDATE ON HR.EMPLOYEES, DELETE ON HR.EMPLOYEES
 ACTIONS COMPONENT=OLS DROP POLICY, DISABLE POLICY, UNSUBSCRIBE OID;
AUDIT POLICY generic_audit_pol BY HR, LBACSYS WHENEVER SUCCESSFUL;
```

親トピック: Oracle Label Securityイベントの監査

27.2.15.8 例: 監査済のOLSセッション・ラベルの問合せ

UNIFIED_AUDIT_TRAIL問合せでLBACSYS.ORA_GET_AUDITED_LABEL関数を使用して、監査済Oracle Label Securityセッション・ラベルを確認できます。

例27-29に、UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューの問合せでLBACSYS.ORA_GET_AUDITED_LABEL関数を使用する方法を示します。

例27-29 監査対象のOracle Label Securityセッション・ラベルの問合せ

```
SELECT ENTRY_ID, SESSIONID,
       LBACSYS.ORA_GET_AUDITED_LABEL( APPLICATION_CONTEXTS,'GENERIC_AUDIT_POL1') AS
SESSION_LABEL1,
       LBACSYS.ORA_GET_AUDITED_LABEL( APPLICATION_CONTEXTS,'GENERIC_AUDIT_POL2') AS
SESSION_LABEL2
FROM UNIFIED_AUDIT_TRAIL;
/
ENTRY_ID  SESSIONID  SESSION_LABEL1  SESSION_LABEL2
--------  ---------  --------------  --------------
       1       1023  SECRET          LEVEL_ALPHA
       2       1024  TOP_SECRET      LEVEL_BETA
```

親トピック: Oracle Label Securityイベントの監査

27.2.15.9 監査証跡でのOracle Label Securityの監査イベントの表示方法

UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューはOracle Label Securityの監査イベントを表示します。

UNIFIED_AUDIT_TRAILビューのOLS_*列は、Oracle Label Security固有の監査データを示します。たとえば:

```
SELECT OLS_PRIVILEGES_USED FROM UNIFIED_AUDIT_TRAIL WHERE DBUSERNAME = 'psmith';
OLS_PRIVILEGES_USED
-------------------
READ
WRITEUP
WRITEACROSS
```

監査証跡で取得されるセッション・ラベルは、UNIFIED_AUDIT_TRAILビューのAPPLICATION_CONTEXTS列に格納されます。LBACSYS.ORA_GET_AUDITED_LABELファンクションを使用して、APPLICATION_CONTEXTS列に格納されているセッション・ラベルを取得できます。このファンクションは、UNIFIED_AUDIT_TRAIL.APPLICATION_CONTEXTS列値、およびOracle Label Securityポリシー名を引数として受け入れ、指定したポリシーの列に格納されているセッション・ラベルを返します。

関連項目:

ORA_GET_AUDITED_LABELファンクションの詳細は、『Oracle Label Security管理者ガイド』を参照してください。

親トピック: Oracle Label Securityイベントの監査

27.2.16 Oracle Data Miningイベントの監査

CREATE AUDIT POLICY文を使用して、Oracle Data Miningイベントを監査できます。

- Oracle Data Miningイベントの監査について
  Oracle Data Miningイベントを監査するには、AUDIT_ADMINロールが必要です。
- Oracle Data Miningの統合監査証跡イベント
  統合監査証跡では、Oracle Data Miningの監査イベントを取得できます。
- Oracle Data Miningの統合監査ポリシーの構成
  CREATE AUDIT POLICY文のACTIONSおよびON MINING MODEL句を使用して、Oracle Data Miningイベントの統合監査ポリシーを作成できます。
- 例: ユーザーによる複数のOracle Data Mining操作の監査
  CREATE AUDIT POLICY文で、複数のOracle Data Mining操作を監査できます。
- 例: ユーザーによる失敗したすべてのOracle Data Mining操作の監査
  CREATE AUDIT POLICY文で、ユーザーによる失敗したOracle Data Miningの操作を監査できます。
- 監査証跡でのOracle Data Miningイベントの表示方法
  UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューはOracle Data Miningの監査イベントを表示します。

親トピック: 統合監査ポリシーおよびAUDIT文を使用したアクティビティの監査

27.2.16.1 Oracle Data Miningイベントの監査について

Oracle Data Miningイベントを監査するには、AUDIT_ADMINロールが必要です。

監査証跡にアクセスするには、UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューを問い合せます。

関連項目:

Oracle Data Miningの詳細は、『Oracle Data Mining概要』を参照してください。

親トピック: Oracle Data Miningイベントの監査

27.2.16.2 Oracle Data Miningの統合監査証跡イベント

統合監査証跡では、Oracle Data Miningの監査イベントを取得できます。

表27-18では、これらのイベントについて説明します。

表27-18 Oracle Data Miningの監査イベント

監査イベント	説明
AUDIT	データ・マイニング・モデルの監査レコードを生成します。
COMMENT	データ・マイニング・モデルにコメントを追加します。
GRANT	データ・マイニング・モデルへのアクセス権をユーザーに付与します。
RENAME	データ・マイニング・モデルの名前を変更します。
SELECT	データ・マイニング・モデルを適用またはその署名を表示します。

親トピック: Oracle Data Miningイベントの監査

27.2.16.3 Oracle Data Miningの統合監査ポリシーの構成

CREATE AUDIT POLICY文のACTIONSおよびON MINING MODEL句を使用して、Oracle Data Miningイベントの統合監査ポリシーを作成できます。

- 次の構文を使用して、Oracle Data Miningの統合監査ポリシーを作成します。

```
CREATE AUDIT POLICY policy_name
ACTIONS {operation | ALL}
ON MINING MODEL schema_name.model_name;
```

たとえば:

```
CREATE AUDIT POLICY dm_ops ACTIONS RENAME ON MINING MODEL hr.dm_emp;
```

条件を含む場合など、より複雑なポリシーを作成できます。ポリシーを作成したら、AUDIT文を使用して有効にする必要があります。

関連トピック

- 統合監査ポリシーの作成の構文

親トピック: Oracle Data Miningイベントの監査

27.2.16.4 例: ユーザーによる複数のOracle Data Mining操作の監査

CREATE AUDIT POLICY文で、複数のOracle Data Mining操作を監査できます。

例27-30に、ユーザーpsmithによる複数のOracle Data Mining操作を監査する方法を示します。イベントごとにON MINING MODEL schema_name.model_name句を含み、それぞれカンマで区切ります。この例では、両方のアクションで同じschema_name.model nameを指定しますが、スキーマおよびデータ・モデルごとに異なるschema_name.model_name設定を構文で指定できます。

例27-30 あるユーザーによる複数のOracle Data Mining操作の監査

```
CREATE AUDIT POLICY dm_ops_pol
ACTIONS SELECT ON MINING MODEL dmuser1.nb_model, ALTER ON MINING MODEL
dmuser1.nb_model;
AUDIT POLICY dm_ops_pol BY psmith;
```

親トピック: Oracle Data Miningイベントの監査

27.2.16.5 例: ユーザーによる失敗したすべてのOracle Data Mining操作の監査

CREATE AUDIT POLICY文で、ユーザーによる失敗したOracle Data Miningの操作を監査できます。

例27-31に、ユーザーpsmithによる失敗したすべてのOracle Data Mining操作を監査する方法を示します。

例27-31 あるユーザーによる失敗したすべてのOracle Data Mining操作の監査

```
CREATE AUDIT POLICY dm_all_ops_pol ACTIONS ALL ON MINING MODEL dmuser1.nb_model;
AUDIT POLICY dm_all_ops_pol BY psmith WHENEVER NOT SUCCESSFUL;
```

親トピック: Oracle Data Miningイベントの監査

27.2.16.6 監査証跡でのOracle Data Miningイベントの表示方法

UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューはOracle Data Miningの監査イベントを表示します。

次の例に、UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューを問い合せて、データ・マイニングの監査イベントを確認する方法を示します。

```
SELECT DBUSERNAME, ACTION_NAME, SYSTEM_PRIVILEGE_USED, RETURN_CODE,
OBJECT_SCHEMA, OBJECT_NAME, SQL_TEXT
FROM UNIFIED_AUDIT_TRAIL;
DBUSERNAME ACTION_NAME          SYSTEM_PRIVILEGE_USED     RETURN_CODE
---------- -------------------- ------------------------ -----------
OBJECT_SCHEMA        OBJECT_NAME
-------------------- --------------------
SQL_TEXT
--------------------------------------------------------------------------------
DMUSER1    CREATE MINING MODEL  CREATE MINING MODEL                 0
DMUSER1
BEGIN
  dbms_data_mining.create_model(model_name => 'nb_model',
                mining_function => dbms_data_mining.classification,
                data_table_name => 'dm_data',
                case_id_column_name => 'case_id',
                target_column_name => 'target');
END;

DMUSER1    SELECT MINING MODEL                                     0
DMUSER1              NB_MODEL
select prediction(nb_model using *) from dual

DMUSER2    SELECT MINING MODEL                                 40284
DMUSER1              NB_MODEL
select prediction(dmuser1.nb_model using *) from dual

DMUSER1    ALTER MINING MODEL                                      0
```

```
DMUSER1              NB_MODEL
BEGIN dbms_data_mining.rename_model('nb_model', 'nb_model1'); END;


DMUSER2    ALTER MINING MODEL                                    40284
DMUSER1              NB_MODEL
BEGIN dbms_data_mining.rename_model('dmuser1.nb_model1', 'nb_model'); END;


DMUSER2    ALTER MINING MODEL                                    40284
DMUSER1              NB_MODEL
BEGIN dbms_data_mining.rename_model('dmuser1.nb_model1', 'nb_model'); END;
```

親トピック: Oracle Data Miningイベントの監査

27.2.17 Oracle Data Pumpイベントの監査

CREATE AUDIT POLICY文を使用して、Oracle Data Pumpを監査できます。

- Oracle Data Pumpイベントの監査について
  Oracle Data Pumpの統合監査ポリシーを作成するには、CREATE AUDIT POLICY文のCOMPONENT句をDATAPUMPに設定する必要があります。
- Oracle Data Pumpの統合監査証跡イベント
  統合監査証跡では、Oracle Data Pumpのイベントを取得できます。
- Oracle Data Pumpの統合監査ポリシーの構成
  CREATE AUDIT POLICY文のACTIONS COMPONENT句を使用して、Oracle Data Pumpイベントの統合監査ポリシーを作成できます。
- 例: Oracle Data Pumpのインポート操作の監査
  CREATE AUDIT POLICY文で、Oracle Data Pumpのインポート操作を監査できます。
- 例: Oracle Data Pumpのすべての操作の監査
  CREATE AUDIT POLICY文で、Oracle Data Pumpのすべての操作を監査できます。
- 監査証跡でのOracle Data Pumpの監査イベントの表示方法
  UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューはOracle Data Pumpの監査イベントを表示します。

親トピック: 統合監査ポリシーおよびAUDIT文を使用したアクティビティの監査

27.2.17.1 Oracle Data Pumpイベントの監査について

Oracle Data Pumpの統合監査ポリシーを作成するには、CREATE AUDIT POLICY文のCOMPONENT句をDATAPUMPに設定する必要があります。

Data Pumpのエクスポート(expdp)およびインポート(impdp)操作を監査できます。

すべての統合監査と同様、Oracle Data Pumpのイベントを監査するには、AUDIT_ADMINロールが必要です。

監査証跡にアクセスするには、UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューを問い合せます。このビューのData Pump固有の列はDP_で始まります。

関連項目:

Oracle Data Pumpの詳細は、『Oracle Databaseユーティリティ』を参照してください。

親トピック: Oracle Data Pumpイベントの監査

27.2.17.2 Oracle Data Pumpの統合監査証跡イベント

統合監査証跡によってOracle Data Pumpのイベントを取得できます。

統合監査証跡では、エクスポート(expdp)およびインポート(impdp)の両方の操作に関する情報を取得します。

親トピック: Oracle Data Pumpイベントの監査

27.2.17.3 Oracle Data Pumpの統合監査ポリシーの構成

CREATE AUDIT POLICY文のACTIONS COMPONENT句を使用して、Oracle Data Pumpイベントの統合監査ポリシーを作成できます。

- 次の構文を使用して、Oracle Data Pumpの統合監査ポリシーを作成します。

```
CREATE AUDIT POLICY policy_name
ACTIONS COMPONENT=DATAPUMP { EXPORT | IMPORT | ALL };
```

たとえば:

```
CREATE AUDIT POLICY audit_dp_export_pol
 ACTIONS COMPONENT=DATAPUMP EXPORT;
```

条件を含む場合など、より複雑なポリシーを作成できます。ポリシーを作成したら、AUDIT文を使用して有効にする必要があります。

関連トピック

- 統合監査ポリシーの作成の構文

親トピック: Oracle Data Pumpイベントの監査

27.2.17.4 例: Oracle Data Pumpのインポート操作の監査

CREATE AUDIT POLICY文で、Oracle Data Pumpのインポート操作を監査できます。

例27-32に、Oracle Data Pumpのすべてのインポート操作を監査する方法を示します。

例27-32 Oracle Data Pumpのインポート操作の監査

```
CREATE AUDIT POLICY audit_dp_import_pol
 ACTIONS COMPONENT=DATAPUMP IMPORT;
AUDIT POLICY audit_dp_import_pol;
```

親トピック: Oracle Data Pumpイベントの監査

27.2.17.5 例: Oracle Data Pumpのすべての操作の監査

CREATE AUDIT POLICY文で、Oracle Data Pumpのすべての操作を監査できます。

例27-33に、Oracle Database Pumpのエクスポート操作とインポート操作を両方とも監査する方法を示します。

例27-33 Oracle Data Pumpのすべての操作の監査

```
CREATE AUDIT POLICY audit_dp_all_pol
 ACTIONS COMPONENT=DATAPUMP ALL;
AUDIT POLICY audit_dp_all_pol BY SYSTEM;
```

親トピック: Oracle Data Pumpイベントの監査

27.2.17.6 監査証跡でのOracle Data Pumpの監査イベントの表示方法

UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューはOracle Data Pumpの監査イベントを表示します。

UNIFIED_AUDIT_TRAILビューのDP_*列は、Oracle Data Pump固有の監査データを示します。たとえば:

```
SELECT DP_TEXT_PARAMETERS1, DP_BOOLEAN_PARAMETERS1 FROM UNIFIED_AUDIT_TRAIL
WHERE AUDIT_TYPE = 'DATAPUMP';
DP_TEXT_PARAMETERS1                                DP_BOOLEAN_PARAMETERS1
-------------------------------------------------- ---------------------------------
MASTER TABLE:  "SCOTT"."SYS_EXPORT_TABLE_01",      MASTER_ONLY: FALSE,
JOB_TYPE: EXPORT,                                  DATA_ONLY: FALSE,
METADATA_JOB_MODE: TABLE_EXPORT,                   METADATA_ONLY: FALSE,
JOB VERSION: 19.1.0.0,                             DUMPFILE_PRESENT: TRUE,
ACCESS METHOD: DIRECT_PATH,                        JOB_RESTARTED: FALSE
DATA OPTIONS: 0,
DUMPER DIRECTORY: NULL
REMOTE LINK: NULL,
TABLE EXISTS: NULL,
PARTITION OPTIONS: NONE
```

(この出力は、読みやすくするために変更が加えられています。)

親トピック: Oracle Data Pumpイベントの監査

27.2.18 Oracle SQL*Loaderダイレクト・ロード・パス・イベントの監査

CREATE AUDIT POLICY文を使用して、Oracle SQL*Loaderダイレクト・ロード・パス・イベントを監査できます。

- Oracle SQL*Loaderダイレクト・パス・ロード・イベントの監査について
  Oracle SQL*Loaderダイレクト・パス・イベントを監査するには、AUDIT_ADMINロールが必要です。
- Oracle SQL*Loaderダイレクト・ロード・パスの統合監査証跡イベント
  統合監査証跡によってSQL*Loaderダイレクト・ロード・パスのイベントを取得できます。
- Oracle SQL*Loaderダイレクト・パス・イベントの統合監査証跡ポリシーの構成
  CREATE AUDIT POLICY文のACTIONS COMPONENT句を使用して、Oracle SQL*Loaderダイレクト・パス・イベントの統合監査ポリシーを作成できます。
- 例: Oracle SQL*Loaderダイレクト・パス・ロード操作の監査
  CREATE AUDIT POLICY文で、Oracle SQL*Loaderダイレクト・パス・ロード操作を監査できます。
- 監査証跡でのSQL*Loaderダイレクト・パス・ロードの監査イベントの表示方法
  UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューはSQL*Loaderダイレクト・パス・ロードの監査イベントを表示します。

親トピック: 統合監査ポリシーおよびAUDIT文を使用したアクティビティの監査

27.2.18.1 Oracle SQL*Loaderダイレクト・パス・ロード・イベントの監査について

Oracle SQL*Loaderダイレクト・パス・イベントを監査するには、AUDIT_ADMINロールが必要です。

SQL*Loaderの統合監査ポリシーを作成するには、CREATE AUDIT POLICY文のCOMPONENT句をDIRECT_LOADに設定する必要があります。監査できるのは、従来のパス・ロードなどの他のSQL*Loaderロードではなく、ダイレクト・パス・ロードのみです。

監査証跡にアクセスするには、UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューのDIRECT_PATH_NUM_COLUMNS_LOADED列を問い合せます。

関連項目:

Oracle SQL*Loaderの詳細は、『Oracle Databaseユーティリティ』を参照してください。

親トピック: Oracle SQL*Loaderダイレクト・ロード・パス・イベントの監査

27.2.18.2 Oracle SQL*Loaderダイレクト・ロード・パスの統合監査証跡イベント

統合監査証跡によってSQL*Loaderダイレクト・ロード・パスのイベントを取得できます。

統合監査証跡では、SQL*Loaderで実行されるダイレクト・パス・ロードに関する情報を取得します(SQL*Loaderのコマンドラインまたは`SQL*Loader`の制御ファイルで`direct=true`に設定した場合)。

また、ダイレクト・パスAPIを使用するOracle Call Interface (OCI)プログラムも監査します。

関連項目:

Oracle SQL*Loaderのダイレクト・パス・ロードの詳細は、『Oracle Databaseユーティリティ』を参照してください。

親トピック: Oracle SQL*Loaderダイレクト・ロード・パス・イベントの監査

27.2.18.3 Oracle SQL*Loaderダイレクト・パス・イベントの統合監査証跡ポリシーの構成

`CREATE AUDIT POLICY`文の`ACTIONS COMPONENT`句を使用して、Oracle SQL*Loaderダイレクト・パス・イベントの統合監査ポリシーを作成できます。

- 次の構文を使用して、Oracle SQL*Loaderの統合監査ポリシーを作成します。

```
CREATE AUDIT POLICY policy_name
ACTIONS COMPONENT=DIRECT_LOAD { LOAD };
```

たとえば:

```
CREATE AUDIT POLICY audit_sqlldr_pol
 ACTIONS COMPONENT=DIRECT_LOAD LOAD;
```

条件を含む場合など、より複雑なポリシーを作成できます。ポリシーを作成したら、`AUDIT`文を使用して有効にする必要があります。

関連トピック

- 統合監査ポリシーの作成の構文

親トピック: Oracle SQL*Loaderダイレクト・ロード・パス・イベントの監査

27.2.18.4 例: Oracle SQL*Loaderダイレクト・パス・ロード操作の監査

`CREATE AUDIT POLICY`文で、Oracle SQL*Loaderダイレクト・パス・ロード操作を監査できます。

例27-32に、SQL*Loaderダイレクト・パス・ロード操作を監査する方法を示します。

例27-34 Oracle SQL*Loaderダイレクト・パス・ロード操作の監査

```
CREATE AUDIT POLICY audit_sqlldr_load_pol
 ACTIONS COMPONENT=DIRECT_LOAD LOAD;
AUDIT POLICY audit_sqlldr_load_pol;
```

親トピック: Oracle SQL*Loaderダイレクト・ロード・パス・イベントの監査

27.2.18.5 監査証跡でのSQL*Loaderダイレクト・パス・ロードの監査イベントの表示方法

UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューはSQL*Loaderダイレクト・パス・ロードの監査イベントを表示します。

UNIFIED_AUDIT_TRAILビューのDIRECT_PATH_NUM_COLUMNS_LOADED 列は、SQL*Loaderダイレクト・パス・ロード・メソッドを使用してロードされた列の数を示します。たとえば:

```
SELECT DBUSERNAME, ACTION_NAME, OBJECT_SCHEMA, OBJECT_NAME,
DIRECT_PATH_NUM_COLUMNS_LOADED FROM UNIFIED_AUDIT_TRAIL WHERE AUDIT_TYPE = 'DIRECT
PATH API';
DBUSERNAME  ACTION_NAME OBJECT_SCHEMA OBJECT_NAME  DIRECT_PATH_NUM_COLUMNS_LOADED
----------- ----------- ------------- ------------ ------------------------------
RLAYTON     INSERT       HR            EMPLOYEES    4
```

親トピック: Oracle SQL*Loaderダイレクト・ロード・パス・イベントの監査

27.2.19 トップレベルの文のみの監査

トップレベルのSQLまたはPL/SQL文を監査して、監査レコードのボリュームを制限できます。

- トップレベルのSQL文のみの監査について
  トップレベルの文は、PL/SQLプロシージャ内から実行される文ではなく、ユーザーによって直接実行される文です。
- トップレベルの文のみを取得する統合監査ポリシーの構成
  CREATE AUDIT POLICY文でONLY TOPLEVEL句を指定すると、監査ポリシーの監査構成に応じてエンド・ユーザーが直接発行するSQL文のみを監査できます。
- 例: トップレベルの文の監査
  CREATE AUDIT POLICY文で、任意のユーザーに対する統合監査ポリシーのトップレベルの文の監査レコードを含めるか除外できます。
- 例: トップレベルのSQL文監査の比較
  トップレベルのSQL文監査レコードは、SQLで直接実行されるSQL文またはPL/SQLプロシージャ内から生成できます。
- 統合監査ポリシーのトップレベルのSQL文の取得
  ONLY TOPLEVEL句は、個々の統合監査証跡レコードの出力には影響しません。

親トピック: 統合監査ポリシーおよびAUDIT文を使用したアクティビティの監査

27.2.19.1 トップレベルのSQL文のみの監査について

トップレベルの文は、PL/SQLプロシージャ内から実行される文ではなく、ユーザーによって直接実行される文です。

トップレベルの文を監査できることは、監査レコードのサブセットのみが表示されるように出力をフィルタ処理できることを意味します。ほとんどの場合、指定された監査文に対して1つの監査レコードのみが生成されます。ただし、1つの監査文に対する1つの監査レコードの生成は、他の複数のデータベース表上に構築されたデータベース・ビューおよびその下のビューでSQL問合せを発行するエンド・ユーザーには当てはまりません。問合せでビューにアクセスすると、Oracle Databaseによって、ビューが内部的に展開されて、ビューの構築の基礎となる各オブジェクトにアクセスされます。セキュリティの観点から、統合監査ポリシーがONLY TOPLEVELを追跡している場合でも、Oracle Databaseでは常に、問合せでアクセスされたビューの一部としてアクセスされたオブジェクトごとに1つの監査レコードが生成されます。実際、複数の監査レコードには、UNIFIED_AUDIT_TRAILビューに移入されたものと同じSQL_TEXTの値と同じSTATEMENT_IDの値がありますが、OBJECT_NAMEの値は異なります。

ユーザーSYSを含むすべてのユーザーからトップレベルの文を監査できます。統合監査証跡をトップレベルの文に制限する利点は、特に統合監査ポリシー内の1つの文に対して多数の監査証跡レコードが生成される場合に、監査証跡のサイズが大幅に削減されることです。この機能は、再帰的SQL文を削減するのに役立ちます。これらの監査レコードを制限することで、この機能は有用なデータを提供しないレコードの数も削減します。このシナリオの例として、200,000個を超える個別監査レコードを

生成するDBMS_STATS.GATHER_DATABASE_STATS SQL文の監査などがあります。監査証跡を削減することにより、この機能でデータベース・パフォーマンスが向上し、データベース(および使用中の場合はOracle Audit Vaultリポジトリ)内の領域が節約されます。

親トピック: トップレベルの文のみの監査

27.2.19.2 トップレベルの文のみを取得する統合監査ポリシーの構成

CREATE AUDIT POLICY文でONLY TOPLEVEL句を指定すると、監査ポリシーの監査構成に応じてエンド・ユーザーが直接発行するSQL文のみを監査できます。

ONLY TOPLEVEL句を含むポリシーを検索するには、AUDIT_UNIFIED_POLICIESデータ・ディクショナリ・ビューのAUDIT_ONLY_TOPLEVEL列を問い合せます。
次の構文を使用して、トップレベルのSQL文のみを監査する統合監査ポリシーを作成します。

```
CREATE AUDIT POLICY policy_name
all_existing_options
ONLY TOPLEVEL;
```

たとえば、HR.EMPLOYEES表のSELECT文のトップレベル・インスタンスに監査証跡を制限するには、次のようにします。

```
CREATE AUDIT POLICY actions_on_hr_emp_pol
ACTIONS SELECT ON HR.EMPLOYEES
ONLY TOPLEVEL;
```

親トピック: トップレベルの文のみの監査

27.2.19.3 例: トップレベルの文の監査

CREATE AUDIT POLICY文で、任意のユーザーに対する統合監査証跡にトップレベルの文の監査レコードを含めるか除外できます。

次の例は、ユーザーSYSによって実行されるすべてのトップレベルの文を取得する監査ポリシーを示しています。

例27-35 例: ユーザーSYSによって実行されるトップレベルの文の監査

```
CREATE AUDIT POLICY actions_all_pol ACTIONS ALL
ONLY TOPLEVEL;
AUDIT POLICY actions_all_pol BY SYS;
```

親トピック: トップレベルの文のみの監査

27.2.19.4 例: トップレベルのSQL文監査の比較

トップレベルのSQL文監査レコードは、SQLで直接実行されるSQL文またはPL/SQLプロシージャ内から生成できます。

この例は、PL/SQLプロシージャ内部のビューへのアクセスに対して、PL/SQLプロシージャ外部のビューにアクセスした場合の監査レコードの生成が、どのように異なるかを示しています。この出力は、2つの異なる監査ポリシーから生成される監査レコードのボリュームの違いを示しています。

1. SYSDBA管理権限を持つユーザーSYSとしてデータベース・インスタンスにログインします。

   マルチテナント環境の場合、PDBにログインします。CDB内の使用可能なPDBを確認するには、CDBルート・コンテナにログインし、DBA_PDBSデータ・ディクショナリ・ビューのPDB_NAME列を問い合せます。現在のコンテナを確認するには、show con_nameコマンドを実行します。

2. 次のプロシージャを作成します。

```
CREATE OR REPLACE PROCEDURE proc1 AS
cnt number;
BEGIN
  SELECT COUNT(*) INTO CNT FROM SYS.DBA_USERS WHERE USER_ID=9999;
END;
/
```

3. トップレベルのアクションを取得するには、次の監査ポリシーを作成して有効にします。

```
CREATE AUDIT POLICY toplevel_pol ACTIONS ALL ONLY TOPLEVEL;
AUDIT POLICY toplevel_pol;
```

4. 次の問合せを実行して、監査レコードを生成し、作成したproc1プロシージャの外部にあるSYS.DBA_USERSビューにアクセスします。

```
SELECT /* TOPLEVEL */ COUNT(*) FROM SYS.DBA_USERS WHERE USER_ID=0000;
```

出力は次のようになります。

```
  COUNT(*)
----------
         1
```

5. 先ほど作成したproc1プロシージャを実行して、SYS.DBA_USERSビューに再度アクセスしますが、プロシージャ内からアクセスします。

```
EXEC proc1;
```

6. 次のように、UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューを問い合せます。

```
SELECT ACTION_NAME, OBJECT_SCHEMA,OBJECT_NAME,STATEMENT_ID,ENTRY_ID,
  UNIFIED_AUDIT_POLICIES,SQL_TEXT
  FROM UNIFIED_AUDIT_TRAIL
  ORDER BY EVENT_TIMESTAMP;
```

次のような出力が表示されます。

```
ACTION_NAME          OBJECT_SCHEMA
-------------------- ------------------------------
OBJECT_NAME                    STATEMENT_ID   ENTRY_ID
------------------------------ ------------ ----------
UNIFIED_AUDIT_POLICIES
------------------------------
SQL_TEXT
--------------------------------------------------------------
LOGON
                                          1          1
TOPLEVEL_POL
COMMIT
                                          3          2
TOPLEVEL_POL
COMMIT
                                          4          3
TOPLEVEL_POL
SELECT               SYS
USER$                                     5          4
TOPLEVEL_POL
select /* toplevel */ count(*) from sys.dba_users where user
_id=0000
SELECT               SYS
RESOURCE_GROUP_MAPPING$                   5          5
TOPLEVEL_POL
select /* toplevel */ count(*) from sys.dba_users where user
_id=0000
```

```
SELECT               SYS
TS$                                    5          6
TOPLEVEL_POL
select /* toplevel */ count(*) from sys.dba_users where user
_id=0000
SELECT               SYS
TS$                                    5          7
TOPLEVEL_POL
select /* toplevel */ count(*) from sys.dba_users where user
_id=0000
SELECT               SYS
TS$                                    5          8
TOPLEVEL_POL
select /* toplevel */ count(*) from sys.dba_users where user
_id=0000
SELECT               SYS
PROFNAME$                              5          9
TOPLEVEL_POL
select /* toplevel */ count(*) from sys.dba_users where user
_id=0000
SELECT               SYS
USER_ASTATUS_MAP                       5         10
TOPLEVEL_POL
select /* toplevel */ count(*) from sys.dba_users where user
_id=0000
SELECT               SYS
PROFILE$                               5         11
TOPLEVEL_POL
select /* toplevel */ count(*) from sys.dba_users where user
_id=0000
SELECT               SYS
PROFILE$                               5         12
TOPLEVEL_POL
select /* toplevel */ count(*) from sys.dba_users where user
_id=0000
SELECT               SYS
DBA_USERS                              5         13
TOPLEVEL_POL
select /* toplevel */ count(*) from sys.dba_users where user
_id=0000
EXECUTE              SYS
PROC1                                  7         14
TOPLEVEL_POL
BEGIN proc1; END;
14 rows selected.
```

7. toplevel_pol監査ポリシーを無効にしてから削除します。

```
NOAUDIT POLICY toplevel_pol;
DROP AUDIT POLICY toplevel_pol;
```

8. 新しい監査ポリシーを作成して有効化し、すべてのアクションを取得します。

```
CREATE AUDIT POLICY recursive_pol ACTIONS ALL;
AUDIT POLICY recursive_pol;
```

9. 監査証跡をクリーンアップします。

```
DBMS_AUDIT_MGMT.CLEAN_AUDIT_TRAIL(DBMS_AUDIT_MGMT.AUDIT_TRAIL_UNIFIED,FALSE);
```

10. 次の問合せを実行して、監査レコードを生成し、proc1プロシージャの外部にあるSYS.DBA_USERSビューにアクセスします。

```
SELECT /* TOPLEVEL */ COUNT(*) FROM SYS.DBA_USERS WHERE USER_ID=0000;
```

出力は次のようになります。

```
  COUNT(*)
----------
         1
```

11. proc1プロシージャを実行して、SYS.DBA_USERSに再度アクセスしますが、proc1プロシージャ内からアクセスします。

```
EXEC proc1;
```

12. 次のように、UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューを問い合せます。

```
SELECT ACTION_NAME, OBJECT_SCHEMA,OBJECT_NAME,STATEMENT_ID,ENTRY_ID,
  UNIFIED_AUDIT_POLICIES,SQL_TEXT
  FROM UNIFIED_AUDIT_TRAIL
  ORDER BY EVENT_TIMESTAMP;
```

次のような出力が表示されます。

```
ACTION_NAME          OBJECT_SCHEMA
-------------------- ------------------------------
OBJECT_NAME                    UNIFIED_AUDIT_POLICIES         STATEMENT_ID
------------------------------ ------------------------------ ------------
  ENTRY_ID SQL_TEXT
---------- ----------------------------------------------------------------
LOGON
                               RECURSIVE_POL                             1
         1
ALTER SESSION
                               RECURSIVE_POL                             1
         2 ALTER SESSION SET TIME_ZONE='-07:00'
COMMIT
                               RECURSIVE_POL                             3
         3
COMMIT
                               RECURSIVE_POL                             4
         4
SELECT               SYS
USER$                          RECURSIVE_POL                             5
         5 select /* toplevel */ count(*) from sys.dba_users where user
           _id=0000
SELECT               SYS
RESOURCE_GROUP_MAPPING$        RECURSIVE_POL                             5
         6 select /* toplevel */ count(*) from sys.dba_users where user
           _id=0000
SELECT               SYS
TS$                            RECURSIVE_POL                             5
         7 select /* toplevel */ count(*) from sys.dba_users where user
           _id=0000
SELECT               SYS
TS$                            RECURSIVE_POL                             5
         8 select /* toplevel */ count(*) from sys.dba_users where user
           _id=0000
SELECT               SYS
TS$                            RECURSIVE_POL                             5
         9 select /* toplevel */ count(*) from sys.dba_users where user
           _id=0000
SELECT               SYS
PROFNAME$                      RECURSIVE_POL                             5
        10 select /* toplevel */ count(*) from sys.dba_users where user
           _id=0000
SELECT               SYS
USER_ASTATUS_MAP               RECURSIVE_POL                             5
        11 select /* toplevel */ count(*) from sys.dba_users where user
```

```
           _id=0000
SELECT               SYS
PROFILE$                        RECURSIVE_POL                                5
        12 select /* toplevel */ count(*) from sys.dba_users where user
           _id=0000
SELECT               SYS
PROFILE$                        RECURSIVE_POL                                5
        13 select /* toplevel */ count(*) from sys.dba_users where user
           _id=0000
SELECT               SYS
DBA_USERS                       RECURSIVE_POL                                5
        14 select /* toplevel */ count(*) from sys.dba_users where user
           _id=0000
SELECT               SYS
USER$                           RECURSIVE_POL                                7
        15 SELECT COUNT(*) FROM SYS.DBA_USERS WHERE USER_ID=9999
SELECT               SYS
RESOURCE_GROUP_MAPPING$         RECURSIVE_POL                                7
        16 SELECT COUNT(*) FROM SYS.DBA_USERS WHERE USER_ID=9999
SELECT               SYS
TS$                             RECURSIVE_POL                                7
        17 SELECT COUNT(*) FROM SYS.DBA_USERS WHERE USER_ID=9999
SELECT               SYS
TS$                             RECURSIVE_POL                                7
        18 SELECT COUNT(*) FROM SYS.DBA_USERS WHERE USER_ID=9999
SELECT               SYS
TS$                             RECURSIVE_POL                                7
        19 SELECT COUNT(*) FROM SYS.DBA_USERS WHERE USER_ID=9999
SELECT               SYS
PROFNAME$                       RECURSIVE_POL                                7
        20 SELECT COUNT(*) FROM SYS.DBA_USERS WHERE USER_ID=9999
SELECT               SYS
USER_ASTATUS_MAP                RECURSIVE_POL                                7
        21 SELECT COUNT(*) FROM SYS.DBA_USERS WHERE USER_ID=9999
SELECT               SYS
PROFILE$                        RECURSIVE_POL                                7
        22 SELECT COUNT(*) FROM SYS.DBA_USERS WHERE USER_ID=9999
SELECT               SYS
PROFILE$                        RECURSIVE_POL                                7
        23 SELECT COUNT(*) FROM SYS.DBA_USERS WHERE USER_ID=9999
SELECT               SYS
DBA_USERS                       RECURSIVE_POL                                7
        24 SELECT COUNT(*) FROM SYS.DBA_USERS WHERE USER_ID=9999
EXECUTE              SYS
PROC1                           RECURSIVE_POL                                7
        25 BEGIN proc1; END;
25 rows selected.
```

この問合せの出力では、25個のレコードが生成されます(先ほどは14個)。

13. recursive_polポリシーを無効にして削除します。

```
NOAUDIT POLICY recursive_pol;
DROP AUDIT POLICY recursive_pol;
```

親トピック: トップレベルの文のみの監査

27.2.19.5 統合監査ポリシーでのトップレベルのSQL文の取得方法

ONLY TOPLEVEL句は、個々の統合監査証跡レコードの出力には影響しません。

ONLY TOPLEVELがポリシーに及ぼす影響は、指定の統合監査ポリシーに対して生成されるレコードの数を制限することのみです。

親トピック: トップレベルの文のみの監査

27.2.20 マルチテナント環境での統合監査ポリシーまたはAUDIT設定

マルチテナント環境では、個々のPDBおよびルートに統合監査ポリシーを作成できます。

- ローカル、CDB共通およびアプリケーション共通監査ポリシーについて
  監査ポリシーは、ローカル、CDB共通またはアプリケーション共通のいずれかにすることができます。
- マルチテナント環境での従来の監査
  従来型の監査(統合監査ではない)では、AUDITおよびNOAUDIT文で、マルチテナント環境の文と権限を監査できます。
- ローカル統合監査ポリシーまたは共通統合監査ポリシーの構成
  CONTAINER句の使用はマルチテナント環境限定で、CREATE AUDIT POLICY文で使用します。
- 例: ローカル統合監査ポリシー
  CREATE AUDIT POLICY文で、ルートまたはPDBにローカル統合監査ポリシーを作成できます。
- 例: CDB共通統合監査ポリシー
  CREATE AUDIT POLICY文で、CDB共通統合監査ポリシーを作成できます。
- 例: アプリケーション共通統合監査ポリシー
  アプリケーション・コンテナ共通統合監査ポリシーの場合、アクション・オプションとシステム権限オプションを監査して、共通オブジェクトおよびロールを参照できます。
- 監査証跡でのローカルまたは共通監査ポリシーまたは設定の表示方法
  ルートまたはアクションが発生したPDBから、統合監査ポリシー・ビューを問い合せることができます。

親トピック: 統合監査ポリシーおよびAUDIT文を使用したアクティビティの監査

27.2.20.1 ローカル、CDB共通およびアプリケーション共通監査ポリシーについて

監査ポリシーは、ローカル、CDB共通またはアプリケーション共通のいずれかにすることができます。

これは統合監査ポリシーと、AUDIT SQL文を使用して作成されたポリシーの両方に適用されます。

- ローカル監査ポリシー。このタイプのポリシーは、ルート(CDBまたはアプリケーション)またはPDB (CDBまたはアプリケーション)に存在できます。ルートに存在するローカル監査ポリシーには、ローカル・オブジェクトと共通オブジェクトの両方用のオブジェクト監査オプションを含めることができます。AUDIT_ADMINロールが付与されているローカル・ユーザーおよび共通ユーザーは、両方ともローカル・ポリシーを有効にできます(ローカル・ユーザーはPDBから、共通ユーザーは権限のあるルートまたはPDBから有効にできます)。ローカル監査ポリシーは、ローカル・ユーザーおよびロールと共通ユーザーおよびロールの両方に対して有効にすることができます。

  ローカル監査ポリシーは、アプリケーション・ローカル・オブジェクトおよびアプリケーション・ローカル・ロールのほか、システム・アクション・オプションおよびシステム権限オプションに対して作成できます。ローカル監査ポリシーをすべてのコンテナにわたり共通ユーザーに対して実行することや、共通監査ポリシーをローカル・ユーザーに対して実施することはできません。

- CDB共通監査ポリシー。このタイプのポリシーは、マルチテナント環境のすべてのPDBに使用できます。共通監査ポリシーを作成および保持できるのは、AUDIT_ADMINロールが付与されている共通ユーザーのみです。共通監査ポリシーは、共通ユーザーのみに対して有効にできます。共通監査ポリシーは、ルートにのみ作成する必要があります。このタイプのポリシーには、共通オブジェクトのみのオブジェクト監査オプションを含めることができ、共通ユーザーのみに対して有効にできます。共通監査ポリシーは、共通ユーザーおよびロールに対してのみ有効にできます。

  共通監査ポリシーをすべてのコンテナにわたりローカル・ユーザーに対して実施することはできません。

- アプリケーション共通監査ポリシー。CDB共通監査ポリシーと同様、このタイプのポリシーは、マルチテナント環境のすべてのPDBに使用できます。共通監査ポリシーは、アプリケーション共通オブジェクトおよびアプリケーション共通ロールのほか、システム・アクション・オプションおよびシステム権限オプションに対して作成できます。このタイプのポリシーはアプリケーション・ルート・コンテナでのみ作成できますが、アプリケーション共通ユーザーとCDB共通ユーザーの両方で有効にできます。オブジェクトを監査する場合は、これらのオブジェクトがアプリケーション共通オブジェクトであることを確認してください。DBA_OBJECTSデータ・ディクショナリ・ビューのSHARING列を問い合せることで、オブジェクトがアプリケーション共通オブジェクトであるかどうかを判断できます。

デフォルトでは、CDBとアプリケーションの両方のシナリオにおいて、監査ポリシーは現在のPDBに対してローカルです。

次の表では、異なるマルチテナント環境における監査ポリシーの適用方法について説明します。

表27-19 CDBルート、アプリケーション・ルートおよび個々のPDBへの監査ポリシーの適用方法

監査オプション・タイプ	CDBルート	アプリケーション・ルート	個々のPDB
共通監査文または監査ポリシー	CDB 共通ユーザーに適用されます	CDB 共通ユーザーに適用されます	CDB 共通ユーザーに適用されます
アプリケーション・コンテナの共通監査文または監査ポリシー	適用されません	● CDB 共通ユーザーに適用され、現在のアプリケーション・コンテナに対してのみ有効です ● アプリケーション・コンテナ共通ユーザーに適用されます	● CDB 共通ユーザーに適用され、このアプリケーション・コンテナに対してのみ有効です ● アプリケーション共通ユーザーに適用されます
ローカル監査文または監査ポリシー	ローカル構成は許可されません	ローカル構成は許可されません	● CDB 共通ユーザーに適用されます ● アプリケーション共通ユーザーに適用されます

親トピック: マルチテナント環境での統合監査ポリシーまたはAUDIT設定

27.2.20.2 マルチテナント環境での従来の監査

従来型の監査(統合監査ではない)では、AUDITおよびNOAUDIT文で、マルチテナント環境の文と権限を監査できます。

ローカル監査ポリシーまたは共通監査ポリシーになるように監査ポリシーを構成するには、作成または変更の他のSQL文に対して一般的に行うように、CONTAINER句を含める必要があります。アプリケーション・コンテナを監査する場合、ローカル・ユーザーおよびロールならびに共通ユーザーおよびロールによって実行されたSQL文およびシステム権限を監査できます。監査レコードは、アクションが実行されたコンテナに作成されます。

- AUDITまたはNOAUDIT文を現在のCDBまたはアプリケーションPDBに適用する場合は、このPDBでCONTAINERをCURRENTに設定する必要があります。たとえば:

```
AUDIT DROP ANY TABLE BY SYSTEM BY ACCESS CONTAINER = CURRENT;
```

- AUDITまたはNOAUDIT文をマルチテナント環境全体に適用する場合は、CDBルートでCONTAINERをALLに設定

する必要があります。アプリケーション・ルートの場合は、アプリケーション・ルート内に設定します。たとえば、次のようになります。

```
AUDIT DROP ANY TABLE BY SYSTEM BY ACCESS CONTAINER = ALL;
```

従来の監査オプションがアプリケーション・コンテナでの使用を考慮して設計されているかどうかを確認するには、DBA_OBJ_AUDIT_OPTSおよびDBA_OBJECTSデータ・ディクショナリ・ビューの結合問合せを実行します。具体的には、両方のビューでOWNERおよびOBJECT_NAME列を使用し、DBA_OBJECTSでAPPLICATION列を使用します。

関連項目:

従来のAUDITおよびNOAUDIT SQL文の詳細は、*Oracle Database SQL*言語リファレンスを参照してください

親トピック: マルチテナント環境での統合監査ポリシーまたはAUDIT設定

27.2.20.3 ローカル統合監査ポリシーまたは共通統合監査ポリシーの構成

CONTAINER句の使用はマルチテナント環境限定で、CREATE AUDIT POLICY文で使用します。

CDB環境またはアプリケーション・コンテナ環境でローカルまたは共通(CDBまたはアプリケーション)統合監査ポリシーを作成するには、CREATE AUDIT POLICY文にCONTAINER句を含めます。

- 次の構文を使用して、ローカル統合監査ポリシーまたは共通統合監査ポリシーを作成します。

```
CREATE AUDIT POLICY policy_name
 action1 [,action2 ]
 [CONTAINER = {CURRENT | ALL}];
```

詳細は、次のとおりです。

- CURRENTは、監査ポリシーを現在のPDBにローカルになるように設定します。
- ALLは、監査ポリシーを共通監査ポリシー(マルチテナント環境全体で使用可能にする)にします。

たとえば、共通統合監査ポリシーの場合は次のようになります。

```
CREATE AUDIT POLICY dict_updates
 ACTIONS UPDATE ON SYS.USER$,
  DELETE ON SYS.USER$,
  UPDATE ON SYS.LINK$,
  DELETE ON SYS.LINK$
  CONTAINER = ALL;
```

次のことに注意してください。

- CONTAINER句はCREATE AUDIT POLICY文に設定できますが、ALTER AUDIT POLICYまたはDROP AUDIT POLICYには設定できません。この設定を使用するように既存の統合監査ポリシーの範囲を変更する場合は、ポリシーを削除してから再作成します。
- AUDIT文の場合は、リリース12.x以降の監査機能にまだ移行していないOracleデータベースがある場合など、監査設定のみにCONTAINER句を設定します。統合監査ポリシーを有効にするために使用するAUDIT文には、CONTAINER句を使用できません。
- PDBにいる場合、CONTAINER句に設定できるのはALLではなくCURRENTのみです。PDBにいる場合に設定を省略すると、デフォルトはCONTAINER = CURRENTになります。
- ルートにいる場合は、CONTAINER句を、ポリシーをルートのみに適用する場合はCURRENTに、ポリシーをCDB全体

に適用する場合はALLに設定できます。CONTAINER句を省略すると、デフォルトはCONTAINER = CURRENTになります。

- オブジェクトの場合:
  - 共通監査ポリシーには共通オブジェクトのみ、ローカル監査ポリシーにはローカル・オブジェクトと共通オブジェクトの両方を含めることができます。
  - 関係するオブジェクトがローカルの場合は、CONTAINERをALLに設定することはできません。共通オブジェクトにする必要があります。
- 権限の場合:
  - 関係するユーザー・アカウントがローカル・アカウントと共通アカウントの混在の場合は、CONTAINERをCURRENTに設定(またはCONTAINER句を省略)できます。これにより、現在のPDBのみに適用されるローカル監査構成が作成されます。
  - 関係するユーザーがローカル・ユーザーの場合は、CONTAINERをALLに設定することはできません。共通ユーザーにする必要があります。
  - CONTAINERをALLに設定し、ユーザー・リストを指定しない場合(BY句をAUDIT文に使用)、構成が各PDBのすべての共通ユーザーに適用されます。
- アプリケーション・コンテナの場合、アプリケーションのインストール、アップグレード、パッチ適用およびアンインストールに使用されるアプリケーション・コンテナ・スクリプトから共通統合監査ポリシーを実行できます。これを行うには、次のようにします。
  - アプリケーション・コンテナ・ルートに共通統合監査ポリシーを作成し、このポリシーをCONTAINER = ALLに設定します。または、このポリシーを次のステップで説明するスクリプトに含めることもできます。
  - Oracle Databaseのインストール、アップグレード、パッチ適用またはアンインストールに通常使用するスクリプトのカスタム・バージョンを作成します。
  - このスクリプト内の次の行に、監査するSQL文を含めます。

    ```
    ALTER PLUGGABLE DATABASE APPLICATION BEGIN INSTALL
    List SQL statements here. Separate each statement with a semi-colon.
    ALTER PLUGGABLE DATABASE APPLICATION END INSTALL
    ```

    スクリプトに統合監査ポリシーを含める場合は、CREATE AUDIT POLICYとAUDIT POLICYの両方の文を含めるようにします。

  監査ポリシーを作成して有効にした後、監査ポリシーがデータベースで定義されたものであるか、スクリプトからのものであるかにかかわらず、アプリケーション共通オブジェクトへのすべてのユーザー・アクセスが監査されます。
- アプリケーションのインストール、アップグレード、パッチ適用およびアンインストール操作をアプリケーション・ルートまたはアプリケーションPDBでローカルに監査するには、共通統合監査ポリシーに関する前の手順と同様の手順に従いますが、後からアプリケーションPDBを同期します。たとえば:

```
ALTER PLUGGABLE DATABASE APPLICATION application_name SYNC;
```

関連トピック

- Oracle Multitenant管理者ガイド

親トピック: マルチテナント環境での統合監査ポリシーまたはAUDIT設定

27.2.20.4 例: ローカル統合監査ポリシー

CREATE AUDIT POLICY文で、ルートまたはPDBにローカル統合監査ポリシーを作成できます。

ルートでローカル統合監査ポリシーを作成すると、マルチテナント環境全体ではなく、ルートにのみ適用されます。

以下の例に、共通ユーザーc##sec_adminによってPDBから作成され、共通ユーザーc##hr_adminに適用されているローカル統合監査ポリシーを示します。

例27-36 ローカル統合監査ポリシー

```
CONNECT c##sec_admin@hrpdb
Enter password: password
Connected.
CREATE AUDIT POLICY table_privs
 PRIVILEGES CREATE ANY TABLE, DROP ANY TABLE
 CONTAINER = CURRENT;
AUDIT POLICY table_privs BY c##hr_admin;
```

親トピック: マルチテナント環境での統合監査ポリシーまたはAUDIT設定

27.2.20.5 例: CDB共通統合監査ポリシー

CREATE AUDIT POLICY文で、CDB共通統合監査ポリシーを作成できます。

例27-37に、共通ユーザーc##sec_adminによってルートから作成され、共通ユーザーc##hr_adminに適用されている共通統合監査ポリシーを示します。

例27-37 共通統合監査ポリシー

```
CONNECT c##sec_admin
Enter password: password
Connected.
CREATE AUDIT POLICY admin_pol
 ACTIONS CREATE TABLE, ALTER TABLE, DROP TABLE
 ROLES c##hr_mgr, c##hr_sup
 CONTAINER = ALL;
AUDIT POLICY admin_pol BY c##hr_admin;
```

親トピック: マルチテナント環境での統合監査ポリシーまたはAUDIT設定

27.2.20.6 例: アプリケーション共通統合監査ポリシー

アプリケーション・コンテナ共通統合監査ポリシーの場合、アクション・オプションとシステム権限オプションを監査して、共通オブジェクトおよびロールを参照できます。

アプリケーション共通監査ポリシーの作成はアプリケーション・ルートからに限定されますが、このポリシーをアプリケーション共通ユーザーとCDB共通ユーザーの両方に対して有効にできます。

以下の例に、アプリケーション・コンテナapp_pdbでアプリケーション共通ユーザーSYSTEMを監査するポリシーを作成する方法を示します。この監査ポリシーは、SYSTEM.utils_tab表に対するSELECTアクション、およびコンテナ・データベース(CDBルートを含む)内の任意のPDBに対するDROP TABLEアクションを監査します。このポリシーは、すべてのコンテナでSELECT ANY TABLEシステム権限の使用も監査します。

例27-38 アプリケーション共通統合監査ポリシー

```
CONNECT c##sec_admin@app_pdb
Enter password: password
Connected.
CREATE AUDIT POLICY app_pdb_admin_pol
 ACTIONS SELECT ON hr_app_cdb.utils_tab, DROP TABLE
```

```
 PRIVILEGES SELECT ANY TABLE
 CONTAINER = ALL;
AUDIT POLICY app_pdb_admin_pol by SYSTEM, c##hr_admin;
```

前述の例では、CONTAINERをALLに設定することで、ポリシーは、アプリケーション・ルートおよびアプリケーション・ルートに属するすべてのアプリケーションPDBのすべての関連オブジェクトへのアクセスのみに適用されます。この範囲外にポリシーは適用されません。

親トピック: マルチテナント環境での統合監査ポリシーまたはAUDIT設定

27.2.20.7 監査証跡でのローカルまたは共通監査ポリシーまたは設定の表示方法

ルートまたはアクションが発生したPDBから、統合監査ポリシー・ビューを問い合せることができます。

次のタイプの問合せを実行できます。

- すべてのPDBからのレコードを監査する。監査証跡には、PDBで実行された監査アクションが反映されます。たとえば、PDB1のユーザーlbrownが、共通監査ポリシーまたはローカル監査ポリシーのいずれかによって監査されたアクションを実行すると、監査証跡によってこのアクションが取得されます。UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューのDBID列は、監査アクションが実行され、ポリシーが適用されるPDBを示します。すべてのPDBからの監査レコードを確認する場合は、ルートからCDB_UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューを問い合せる必要があります。
- 共通監査ポリシーからのレコードを監査する。この場所は、共通監査ポリシーが監査レコードになる場所です。アクションが実際に発生した場所に応じて、マルチテナント環境(ルートまたはPDB)の任意の場所で監査レコードを生成できます。たとえば、共通監査ポリシーfga_polは、DBMS_FGA PL/SQLパッケージのEXECUTE権限を監査し、このアクションがPDB1で発生すると、監査レコードはルートではなく、PDB1に生成されます。このため、監査レコードはPDB1に表示できます。

  ポリシー名にWHERE句を使用している場合は(たとえば、WHERE UNIFIED_AUDIT_POLICIES = 'FGA_POL')、ポリシーについて、ルートまたはPDBのいずれかからUNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューを問い合せることができます。

次の例は、共通統合監査ポリシーの結果を検索する方法を示します。

```
CONNECT c##sec_admin
Enter password: password
Connected.
SELECT DBID, ACTION_NAME, OBJECT_SCHEMA, OBJECT_NAME FROM CDB_UNIFIED_AUDIT_TRAIL
WHERE DBUSERNAME = 'c##hr_admin';
46892-1
DBID        ACTION_NAME  OBJECT_SCHEMA  OBJECT_NAME
----------- -----------  -------------  -----------
653916017   UPDATE       HR             EMPLOYEES
653916018   UPDATE       HR             JOB_HISTORY
653916017   UPDATE       HR             JOBS
```

親トピック: マルチテナント環境での統合監査ポリシーまたはAUDIT設定

27.2.21 統合監査ポリシーの変更

ALTER AUDIT POLICY文を使用して、統合監査ポリシーを変更できます。

- 統合監査ポリシーの変更について
  統合監査ポリシーでは、CONTAINER設定を除いたほとんどのプロパティを変更できます。

- 統合監査ポリシーの変更
  ALTER AUDIT POLICY文で、統合監査ポリシーを変更できます。
- 例: 統合監査ポリシーの条件の変更
  ALTER AUDIT POLICY文で、統合監査ポリシーの条件を変更できます。
- 例: 統合監査ポリシーでのOracle Label Securityコンポーネントの変更
  ALTER AUDIT POLICY文で、監査ポリシーのOracle Label Securityコンポーネントを変更できます。
- 例: 統合監査ポリシーのロールの変更
  ALTER AUDIT POLICY文で、統合監査ポリシーのロールを変更できます。
- 例: 統合監査ポリシーからの条件の削除
  ALTER AUDIT POLICY文で、統合監査ポリシーから条件を削除できます。
- 例: 既存の統合監査ポリシーのトップレベルの文の監査の変更
  ALTER AUDIT POLICY文で、統合監査証跡がトップレベルのSQL文のみを取得できるように、既存の統合監査ポリシーを変更できます。

親トピック: 統合監査ポリシーおよびAUDIT文を使用したアクティビティの監査

27.2.21.1 統合監査ポリシーの変更について

統合監査ポリシーでは、CONTAINER設定を除いたほとんどのプロパティを変更できます。

マルチテナント環境では統合監査ポリシーを変更できません。たとえば、共通統合監査ポリシーをローカル統合監査ポリシーにすることはできません。

既存の統合監査ポリシーを検索するには、AUDIT_UNIFIED_POLICIESデータ・ディクショナリ・ビューを問い合せます。有効な統合監査ポリシーのみを検索する場合は、AUDIT_UNIFIED_ENABLED_POLICIESビューを問い合せます。有効な監査ポリシーと無効な監査ポリシーの両方を変更できます。有効な監査ポリシーを変更する場合は、変更後も有効なままです。

オブジェクトの統合監査ポリシーを変更すると、アクティブおよび後続の両方のユーザー・セッションで新しい監査設定が即座に実行されます。システムの監査オプションを変更する場合やポリシーの条件を監査する場合は、現在のユーザー・セッションではなく、新しいユーザー・セッションに対してアクティブになります。

親トピック: 統合監査ポリシーの変更

27.2.21.2 統合監査ポリシーの変更

ALTER AUDIT POLICY文で、統合監査ポリシーを変更できます。

- 統合監査ポリシーを変更するために次の構文を使用し、ALTER AUDIT POLICY文を使用します。

```
ALTER AUDIT POLICY  policy_name
[ADD [privilege_audit_clause][action_audit_clause]
  [role_audit_clause] [ONLY TOPLEVEL] ]
[DROP [privilege_audit_clause][action_audit_clause]
    [role_audit_clause] [ONLY TOPLEVEL]]
[CONDITION {DROP | audit_condition EVALUATE PER {STATEMENT|SESSION|INSTANCE}}]
```

詳細は、次のとおりです。

- ADDでは、次の設定を変更できます。
  - privilege_audit_clauseは、権限に関連する監査オプションを記述します。詳細は、システム権限の監査を参照してください。権限の監査オプションを構成するための詳細な構文は、次のとおりです。

```
ADD privilege_audit_clause  :=  PRIVILEGES  privilege1 [, privilege2]
```

- action_audit_clauseおよびstandard_actionsは、オブジェクト・アクションに関連する監査オプションを記述します。オブジェクト・アクションの監査を参照してください。構文は次のとおりです。

```
ADD action_audit_clause := {standard_actions | component_actions}
                                          [, component_actions ]
standard_actions :=
     ACTIONS action1 [ ON {schema.obj_name
                                          | DIRECTORY directory_name
                                          | MINING MODEL schema.obj_name
                                           }
                 ]
           [, action2 [ ON {schema.obj_name
                                          | DIRECTORY directory_name
                                          | MINING MODEL schema.obj_name
                      }
                 ]
```

- role_audit_clauseでは、ロールのポリシーを追加または削除できます。「ロールの監査」を参照してください。構文は次のとおりです。

```
ADD role_audit_clause := ROLES role1 [, role2]
```

- ONLY TOPLEVELでは、このポリシーの影響を受けるトップレベルのSQL文のみを統合監査ポリシーに含めます。

- DROPでは、ADD句で説明されているのと同じコンポーネントを削除できます。たとえば:

```
DROP role_audit_clause := ROLES role1 [, role2 ONLY TOPLEVEl]
```

- CONDITION {DROP...では、ポリシーの条件を追加または削除できます。既存の条件を変更する場合は、EVALUATE PER句を条件に含める必要があります。統合監査ポリシーの条件の作成を参照してください。構文は次のとおりです。

```
CONDITION 'audit_condition := function operation value_list'
EVALUATE PER {STATEMENT|SESSION|INSTANCE}
```

条件を削除する場合は、条件の定義およびEVALUATE PER句を省略します。たとえば:

```
CONDITION DROP
```

親トピック: 統合監査ポリシーの変更

27.2.21.3 例: 統合監査ポリシーの条件の変更

ALTER AUDIT POLICY文で、統合監査ポリシーの条件を変更できます。

例27-39に、既存の統合監査ポリシーの条件を変更する方法を示します。

例27-39 統合監査ポリシーの条件の変更

```
ALTER AUDIT POLICY orders_unified_audpol
 ADD ACTIONS INSERT ON SCOTT.EMP
CONDITION 'SYS_CONTEXT(''ENTERPRISE'', ''GROUP'') =  ''ACCESS_MANAGER'''
EVALUATE PER SESSION;
```

親トピック: 統合監査ポリシーの変更

27.2.21.4 例: 統合監査ポリシーでのOracle Label Securityコンポーネントの変更

ALTER AUDIT POLICY文で、監査ポリシーのOracle Label Securityコンポーネントを変更できます。

例27-40に、監査ポリシーのOracle Label Securityコンポーネントを変更する方法を示します。

例27-40 統合監査ポリシーでのOracle Label Securityコンポーネントの変更

```
ALTER AUDIT POLICY audit_ols
 ADD ACTIONS SELECT ON HR.EMPLOYEES
 ACTIONS COMPONENT=OLS DROP POLICY, DISABLE POLICY, REMOVE POLICY;
```

親トピック: 統合監査ポリシーの変更

27.2.21.5 例: 統合監査ポリシーのロールの変更

ALTER AUDIT POLICY文で、統合監査ポリシーのロールを変更できます。

例27-41に、共通統合監査ポリシーにロールを追加する方法を示します。

例27-41 統合監査ポリシーのロールの変更

```
CONNECT c##sec_admin
Enter password: password
Connected.
ALTER AUDIT POLICY RoleConnectAudit
 ADD ROLES c##role1, c##role2;
```

親トピック: 統合監査ポリシーの変更

27.2.21.6 例: 統合監査ポリシーからの条件の削除

ALTER AUDIT POLICY文で、統合監査ポリシーから条件を削除できます。

例27-42に、既存の統合監査ポリシーから条件を削除する方法を示します。

例27-42 統合監査ポリシーからの条件の削除

```
ALTER AUDIT POLICY orders_unified_audpol
CONDITION DROP;
```

親トピック: 統合監査ポリシーの変更

27.2.21.7 例: 既存の統合監査ポリシーのトップレベルの文の監査の変更

ALTER AUDIT POLICY文で、統合監査証跡がトップレベルのSQL文のみを取得できるように、既存の統合監査ポリシーを変更できます。

次の例は、orders_unified_audpolポリシーを変更してトップレベルのSQL文のみを取得する方法を示しています。

例27-43 トップレベルの文の監査のための既存の統合監査ポリシーの変更

```
ALTER AUDIT POLICY orders_unified_audpol ADD ONLY TOPLEVEL;
```

同様に、トップレベルのSQL文監査を削除するには、DROP句を使用します。

```
ALTER AUDIT POLICY orders_unified_audpol DROP ONLY TOPLEVEL;
```

親トピック: 統合監査ポリシーの変更

27.2.22 統合監査ポリシーの有効化およびユーザーとロールへの適用

AUDIT POLICY文を使用すると、統合監査ポリシーを有効にして、ユーザーとロールに適用できます。

- 統合監査ポリシーの有効化について

POLICY句を含むAUDIT文で、統合監査ポリシーを有効にして、オブジェクト・レベルのオプションを含むすべてのタイプの監査オプションに適用します。

- 統合監査ポリシーの有効化
  AUDIT POLICY文で、統合監査ポリシーを有効にできます。
- 例: 統合監査ポリシーの有効化
  AUDIT POLICY文で、WHENEVER NOT SUCCESSFULなどの条件を使用して統合監査ポリシーを有効にできます。

親トピック: 統合監査ポリシーおよびAUDIT文を使用したアクティビティの監査

27.2.22.1 統合監査ポリシーの有効化について

POLICY句を含むAUDIT文で、統合監査ポリシーを有効にして、オブジェクト・レベルのオプションを含むすべてのタイプの監査オプションに適用します。

監査ユーザー(またはポリシーに関連付けられたロールを付与されたユーザー)がデータベース・インスタンスにログインするまで、ポリシーは有効となりません。言い方を変えると、監査ユーザーのログイン中にポリシーを作成して有効化すると、ポリシーは監査データを収集できません。監査を開始する前に、ユーザーはログアウトしてから再ログインする必要があります。セッションが監査で設定されると、設定はユーザー・セッションの間継続し、セッションの終了時に終了します。

監査ポリシーは、個々のユーザーまたはロールに対して有効にできます。監査ポリシーをロールに対して有効にすると、そのロールを直接付与されたユーザーのグループに対してポリシーを有効にできます。新規ユーザーに対してロールを直接付与すると、そのユーザーに対してポリシーが自動的に適用されます。ユーザーからロールを取り消すと、そのユーザーにポリシーは適用されなくなります。

監査の結果を確認するには、UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューを問い合せます。既存の統合監査ポリシーのリストを検索するには、AUDIT_UNIFIED_POLICIESデータ・ディクショナリ・ビューを問い合せます。

AUDIT文では、次のオプションの追加設定を指定できます。

- 統合監査ポリシーを1人以上のユーザーまたは1つ以上のロールに適用するかどうか。SYSDBA管理権限(SYSなど)でログインする管理ユーザーを含む、1人以上のユーザーまたは1つ以上のロールにポリシーを適用するには、BY句を使用します。たとえば、ユーザーSYSおよびSYSTEMにポリシーを適用するには、次のようにします。

  たとえば、2人のユーザーにポリシーを適用する場合:

  ```
  AUDIT POLICY role_connect_audit_pol BY SYS, SYSTEM;
  ```

  DBAおよびCDB_DBAロールを直接付与されたユーザーにポリシーを適用するには:

  ```
  AUDIT POLICY admin_audit_pol BY USERS WITH GRANTED ROLES DBA, CDB_DBA;
  ```

- 統合監査ポリシーからユーザーを除外するかどうか。監査ポリシーからユーザーを除外するには、EXCEPT句を含めます。

  たとえば:

  ```
  AUDIT POLICY role_connect_audit_pol EXCEPT rlee, jrandolph;
  ```

- アクティビティが成功または失敗した場合に、監査レコードを作成するかどうか。この監査方法は、監査証跡が減少するため、特定のアクションに重点を置きやすくなります。これにより、データベースの良好なパフォーマンスを維持できます。次のいずれかの句を入力します。

  - WHENEVER SUCCESSFULは、ユーザーのアクティビティの実行が成功した場合のみ監査します。

- WHENEVER NOT SUCCESSFULは、ユーザーのアクティビティの実行が失敗した場合のみ監査します。異常終了したSQL文を監視することで、アクセス違反や不当な処理を行っているユーザーを判別できます。ただし、異常終了したSQL文の原因はそのどちらでもない場合がほとんどです。

たとえば:

```
AUDIT POLICY role_connect_audit_pol WHENEVER NOT SUCCESSFUL;
```

この句を省略すると、失敗および成功した両方のユーザー・アクティビティが監査証跡に書き込まれます。

次のことに注意してください。

- 統合監査ポリシーには、BY、BY USERS WITH GRANTED ROLESまたはEXCEPT句のみを含めることができますが、同じポリシーに複数を含めることはできません。
- 複数のAUDIT文を同じ統合監査ポリシーで実行し、異なるBYユーザーまたは異なるBY USERS WITH GRANTED ROLESロールを指定すると、Oracle Databaseによってこれらのユーザーまたはロールがすべて監査されます。
- 複数のAUDIT文を同じ統合監査ポリシーで実行し、異なるEXCEPTユーザーを指定すると、Oracle Databaseによって、前述のリストのユーザーではなく、最新の例外ユーザー・リストが使用されます。つまり、前のAUDIT POLICY ... EXCEPT文の影響は最新のAUDIT POLICY ... EXCEPT文によってオーバーライドされます。
- EXCEPT句はロールに対して使用できません。これはユーザーのみに適用されます。
- 共通統合監査ポリシーは、共通ユーザーまたはロールに対してのみ有効にできます。
- マルチテナント環境では、共通監査ポリシーはルートからのみ、ローカル監査ポリシーは、ローカル監査ポリシーが適用されるPDBからのみ有効にできます。

親トピック: 統合監査ポリシーの有効化およびユーザーとロールへの適用

27.2.22.2 統合監査ポリシーの有効化

AUDIT POLICY文で、統合監査ポリシーを有効にできます。

- 次の構文を使用して、統合監査ポリシーを有効にします。

```
AUDIT POLICY { policy_auditing }
 [WHENEVER [NOT] SUCCESSFUL]
```

詳細は、次のとおりです。

- policy_auditingは、次のコンポーネントを示します。
  - 統合監査ポリシーの名前。既存のポリシーをすべて検索するには、AUDIT_UNIFIED_POLICIESデータ・ディクショナリ・ビューを問い合せます。現在有効なポリシーを検索するには、AUDIT_UNIFIED_ENABLED_POLICIESを問い合せます。
  - 統合監査ポリシーが適用されるユーザーまたはロール。ポリシーを1人以上のユーザー(ユーザーSYSを含む)に適用するには、BY句を入力します。たとえば:

    ```
    BY psmith, rlee
    ```

    ロールのリストが直接付与された1人以上のユーザーにポリシーを適用するには、BY USERS WITH GRANTED ROLES句を使用します。たとえば:

    ```
    BY USERS WITH GRANTED ROLES HS_ADMIN_ROLE, HS_ADMIN_SELECT_ROLE
    ```

- 統合監査ポリシーから除外するユーザー。ポリシーから1人以上のユーザーを除外するには、EXCEPT句を入力します。たとえば:

  ```
  EXCEPT psmith, rlee
  ```

  必須の監査レコードは、AUDIT POLICY SQL文でUNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューに取得されます。除外されたユーザーを監査レコードで検索するには、UNIFIED_AUDIT_TRAILビューのEXCLUDED_USER列を問い合せることで、除外されたユーザーをリストできます。

同じ文のBY句、BY USERS WITH GRANTED ROLES句およびEXCEPT句で、同じ監査ポリシーを有効にすることはできません。このアクションにより、句が競合して、後続のAUDIT文でエラーがスローされます

- WHENEVER [NOT] SUCCESSFULでは、ユーザーのアクションが成功したか失敗したかに基づいて、ポリシーで監査レコードを生成できます。詳細は、統合監査ポリシーの有効化についてを参照してください。

統合監査ポリシーを有効にして、ポリシーがレコードを生成している場合は、UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューを問い合せることで、監査レコードを検索できます。

親トピック: 統合監査ポリシーの有効化およびユーザーとロールへの適用

27.2.22.3 例: 統合監査ポリシーの有効化

AUDIT POLICY文で、WHENEVER NOT SUCCESSFULなどの条件を使用して統合監査ポリシーを有効にできます。

例27-44に、ユーザーdv_adminによる失敗したアクションのみを統合監査ポリシーで記録できるようにする方法を示します。

例27-44 統合監査ポリシーの有効化

```
AUDIT POLICY dv_admin_pol BY tjones
 WHENEVER NOT SUCCESSFUL;
```

親トピック: 統合監査ポリシーの有効化およびユーザーとロールへの適用

27.2.23 統合監査ポリシーの無効化

NOAUDIT POLICY文を使用して、統合監査ポリシーを無効にすることができます。

- 統合監査ポリシーの無効化について
  NOAUDIT文とPOLICY句を組み合せることにより、統合監査ポリシーを無効にできます。
- 統合監査ポリシーの無効化
  NOAUDIT文で、サポートされている監査オプションを使用して統合監査ポリシーを無効にできます。
- 例: 統合監査ポリシーの無効化
  NOAUDIT POLICY文で、ユーザー名などのフィルタを使用して統合監査ポリシーを無効にします。

親トピック: 統合監査ポリシーおよびAUDIT文を使用したアクティビティの監査

27.2.23.1 統合監査ポリシーの無効化について

NOAUDIT文とPOLICY句を組み合せることにより、統合監査ポリシーを無効にできます。

NOAUDIT文では、BYユーザー・リストまたはBY USERS WITH GRANTED ROLESロール・リストを指定できますが、EXCEPTユーザー・リストは指定できません。統合監査ポリシーを無効にすると、後続のユーザー・セッションに反映されます。

既存の統合監査ポリシーのリストを検索するには、AUDIT_UNIFIED_POLICIESデータ・ディクショナリ・ビューを問い合せます。

マルチテナント環境では、共通監査ポリシーはルートからのみ、ローカル監査ポリシーは、ローカル監査ポリシーが適用されるPDBからのみ無効にできます。

親トピック: 統合監査ポリシーの無効化

27.2.23.2 統合監査ポリシーの無効化

NOAUDIT文で、サポートされている監査オプションを使用して統合監査ポリシーを無効にできます。

- 次の構文を使用して、統合監査ポリシーを無効にします。

```
NOAUDIT POLICY {policy_auditing | existing_audit_options};
```

詳細は、次のとおりです。

- policy_auditingは、ポリシーの名前です。現在有効なポリシーをすべて検索するには、AUDIT_UNIFIED_ENABLED_POLICIESデータ・ディクショナリ・ビューを問い合せます。この指定の一部として、BYまたはBY USERS WITH GRANTED ROLES句をオプションで含めることができますが、EXCEPT句を含めることはできません。詳細は、統合監査ポリシーの有効化についてを参照してください。
- existing_audit_optionsは、次のようなOracle Database 12cリリース1 (12.1)より前のリリースで使用可能だったAUDITオプションを示します。
  - SELECT ANY TABLE, UPDATE ANY TABLE BY SCOTT, HR
  - UPDATE ON SCOTT.EMP

統合ポリシーがすべてのユーザーに適用されている場合は、ポリシー名の指定のみが必要です。たとえば:

```
NOAUDIT POLICY logons_pol;
```

親トピック: 統合監査ポリシーの無効化

27.2.23.3 例: 統合監査ポリシーの無効化

NOAUDIT POLICY文で、ユーザー名などのフィルタを使用して統合監査ポリシーを無効にします。

例27-45に、ユーザーまたはロールに対して統合監査ポリシーを無効にする方法の例を示します。

例27-45 統合監査ポリシーの無効化

```
NOAUDIT POLICY dv_admin_pol BY tjones;
NOAUDIT POLICY dv_admin_pol BY USERS WITH GRANTED ROLES emp_admin;
```

親トピック: 統合監査ポリシーの無効化

27.2.24 統合監査ポリシーの削除

DROP AUDIT POLICY文を使用して、統合監査ポリシーを削除できます。

- 統合監査ポリシーの削除について
  DROP AUDIT POLICY文を使用して、統合監査ポリシーを削除できます。
- 統合監査ポリシーの削除
  統合監査ポリシーを削除するには、最初に無効にし、DROP AUDIT POLICY文を実行して削除します。
- 例: 統合監査ポリシーの無効化および削除
  NOAUDIT POLICY文およびDROP AUDIT POLICY文で、統合監査ポリシーを無効化して削除できます。

親トピック: 統合監査ポリシーおよびAUDIT文を使用したアクティビティの監査

27.2.24.1 統合監査ポリシーの削除について

DROP AUDIT POLICY文を使用して、統合監査ポリシーを削除できます。

統合監査ポリシーがすでにセッションで有効な場合、ポリシーを削除しても、既存のこのセッションには影響しません。この時点まで、統合監査ポリシーの設定は有効のままです。ただし、オブジェクト関連の監査ポリシーの場合、影響は即座に反映されます。

既存の統合監査ポリシーのリストを検索するには、AUDIT_UNIFIED_POLICIESデータ・ディクショナリ・ビューを問い合せます。

監査ポリシーを削除する前に無効化する場合は、有効化に使用した設定と同じ設定を使用して無効化してください。たとえば、logon_polポリシーを次のように有効化したとします。

```
AUDIT POLICY logon_pol BY HR, OE;
```

このポリシーを削除する前に、次のようにNOAUDIT文にHRおよびOEユーザーが含まれている必要があります。

```
NOAUDIT POLICY logon_pol BY HR, OE;
```

マルチテナント環境では、共通監査ポリシーはルートからのみ、ローカル監査ポリシーは、ローカル監査ポリシーが適用されるPDBからのみ削除できます。

親トピック: 統合監査ポリシーの削除

27.2.24.2 統合監査ポリシーの削除

統合監査ポリシーを削除するには、最初に無効にし、DROP AUDIT POLICY文を実行して削除します。

- 次の構文を使用して、統合監査ポリシーを削除します。

  ```
  DROP AUDIT POLICY policy_name;
  ```

マルチテナント環境では、統合監査ポリシーの削除は現在のPDBに適用されます。統合監査ポリシーが共通統合監査ポリシーとして削除された場合は、ローカルのPDBから削除することはできません。

関連トピック

- マルチテナント環境での統合監査ポリシーまたはAUDIT設定

親トピック: 統合監査ポリシーの削除

27.2.24.3 例: 統合監査ポリシーの無効化および削除

NOAUDIT POLICY文およびDROP AUDIT POLICY文で、統合監査ポリシーを無効化して削除できます。

例27-46に、共通統合監査ポリシーを無効化および削除する方法を示します。

例27-46 統合監査ポリシーの無効化および削除

```
CONNECT c##sec_admin
Enter password: password
Connected.
NOAUDIT POLICY dv_admin_pol;
DROP AUDIT POLICY dv_admin_pol
```

親トピック: 統合監査ポリシーの削除

27.2.25 例: 非データベース・ユーザーの監査

このチュートリアルでは、非データベース・ユーザーのアクションをクライアント識別子を使用して監査する統合監査ポリシーの作成方法を示します。

- ステップ1: ユーザー・アカウントの作成とユーザーOEがアクティブであることの確認
  ユーザーを作成し、ユーザーOEがアクティブであることを確認する必要があります。
- ステップ2: 統合監査ポリシーの作成
  ここでは、統合監査ポリシーを作成します。
- ステップ3: ポリシーのテスト
  ポリシーをテストするため、ユーザーOEはOE.ORDERS表から選択します。
- ステップ4: このチュートリアルのコンポーネントの削除
  このチュートリアルのコンポーネントが不要になった場合、それらを削除できます。

親トピック: 統合監査ポリシーおよびAUDIT文を使用したアクティビティの監査

27.2.25.1 ステップ1: ユーザー・アカウントの作成とユーザーOEがアクティブであることの確認

ユーザーを作成し、ユーザーOEがアクティブであることを確認する必要があります。

1. SYSDBA管理権限を持つユーザーSYSとしてログインします。

```
sqlplus sys as sysdba
Enter password: password
```

2. マルチテナント環境で、適切なPDBに接続します。

   たとえば:

```
CONNECT SYS@hrpdb AS SYSDBA
Enter password: password
```

   使用可能なPDBを検索するには、show pdbsコマンドを実行します。現在のPDBを確認するには、show con_nameコマンドを実行します。

3. ファイングレイン監査ポリシーを作成するローカル・ユーザーpolicy_adminを作成します。

```
CREATE USER policy_admin IDENTIFIED BY password;
GRANT CREATE SESSION, AUDIT_ADMIN TO policy_admin;
```

   「パスワードの最低要件」のガイドラインに従って、passwordを安全なパスワードに置き換えます。

4. このポリシーの監査証跡をチェックするローカル・ユーザー・アカウントauditorを作成します。

```
CREATE USER policy_auditor IDENTIFIED BY password;
GRANT CREATE SESSION, AUDIT_VIEWER TO policy_auditor;
```

5. この例ではサンプル・ユーザーOEも使用するため、DBA_USERSデータ・ディクショナリ・ビューを問い合せて、OEがロックされていたり、期限切れになっていないことを確認します。

```
SELECT USERNAME, ACCOUNT_STATUS FROM DBA_USERS WHERE USERNAME = 'OE';
```

   アカウント・ステータスはOPENである必要があります。DBA_USERSビューに、ユーザーOEがロックされて期限切れになっていると表示された場合は、ユーザーSYSTEMでログインし、次の文を入力して、OEアカウントのロックを解除し、新しいパスワードを作成します。

```
ALTER USER OE ACCOUNT UNLOCK IDENTIFIED BY password;
```

「パスワードの最低要件」のガイドラインに従って、passwordを安全なパスワードに置き換えます。セキュリティを向上させるため、以前のリリースのOracle Databaseと同じパスワードをOEアカウントに指定しないでください。

親トピック: 例: 非データベース・ユーザーの監査

27.2.25.2 ステップ2: 統合監査ポリシーの作成

ここでは、統合監査ポリシーを作成します。

1. ユーザーpolicy_adminでSQL*Plusに接続します。

```
CONNECT policy_admin -- Or, CONNECT policy_admin@hrpdb
Enter password: password
```

2. 次のポリシーを作成します。

```
CREATE AUDIT POLICY orders_unified_audpol
  ACTIONS INSERT ON OE.ORDERS, UPDATE ON OE.ORDERS, DELETE ON OE.ORDERS, SELECT
ON OE.ORDERS
  WHEN 'SYS_CONTEXT(''USERENV'', ''CLIENT_IDENTIFIER'') = ''robert'''
    EVALUATE PER STATEMENT;
AUDIT POLICY orders_unified_audpol;
```

この例では、AUDIT_CONDITIONパラメータで非データベース・ユーザーの名前がrobertであると想定しています。ポリシーでは、robertが実行するINSERT、UPDATE、DELETEおよびSELECT文を監視します。ポリシーに入力するユーザーのCLIENT_IDENTITIFER設定は、大文字と小文字を区別し、ここで指定する識別情報に使用される大/小文字のみがポリシーで認識されることに注意してください。つまり、後でユーザー・セッションがRobertまたはROBERTに設定されると、ポリシーの条件は満たされません。

親トピック: 例: 非データベース・ユーザーの監査

27.2.25.3 ステップ3: ポリシーのテスト

ポリシーをテストするため、ユーザーOEはOE.ORDERS表から選択します。

統合監査ポリシーは、監査中のユーザーに対する次のユーザー・セッションで有効となります。したがって、監査記録を取得する前に、ユーザーはポリシーが作成されてからデータベースに接続する必要があります。

1. ユーザーOEで接続し、OE.ORDERS表から選択します。

```
CONNECT OE -- Or, CONNECT OE@hrpdb
Enter password: password
SELECT COUNT(*) FROM ORDERS;
```

次のような出力結果が表示されます。

```
  COUNT(*)
----------
       105
```

2. ユーザーpolicy_auditorで接続し、監査レコードが生成されたかどうかを確認します。

```
CONNECT policy_auditor -- Or, CONNECT policy_auditor@hrpdb
Enter password: password
col dbusername format a10
col client_identifier format a20
col sql_text format a29
SELECT DBUSERNAME, CLIENT_IDENTIFIER, SQL_TEXT FROM UNIFIED_AUDIT_TRAIL
 WHERE SQL_TEXT LIKE '%FROM ORDERS%';
```

次のような出力結果が表示されます。

```
no rows selected
```

3. ユーザーOEで再接続し、クライアント識別子をrobertに設定し、OE.ORDERS表から再選択します。

```
CONNECT OE -- Or, CONNECT OE@hrpdb
Enter password: password
EXEC DBMS_SESSION.SET_IDENTIFIER('robert');
SELECT COUNT(*) FROM ORDERS;
```

次の出力が表示されます。

```
  COUNT(*)
----------
       105
```

4. ユーザーauditorで再接続し、監査証跡を再び確認します。

```
CONNECT policy_auditor -- Or, CONNECT policy_auditor@hrpdb
Enter password: password
SELECT DBUSERNAME, CLIENT_IDENTIFIER, SQL_TEXT FROM UNIFIED_AUDIT_TRAIL
 WHERE SQL_TEXT LIKE '%FROM ORDERS%';
```

今回は、robertが存在し、OE.ORDERS表を問い合せたため、監査証跡ではそのアクションが取得されます。

```
DBUSERNAME CLIENT_IDENTIFIER SQL_TEXT
---------- ----------------- ----------------------------
OE         robert            SELECT COUNT(*) FROM ORDERS;
```

親トピック: 例: 非データベース・ユーザーの監査

27.2.25.4 ステップ4: このチュートリアルのコンポーネントの削除

このチュートリアルのコンポーネントが不要になった場合、それらを削除できます。

1. ユーザーpolicy_adminでSQL*Plusに接続し、orders_unified_audpolポリシーを手動で無効にして削除します。

```
CONNECT policy_admin -- Or, CONNECT policy_admin@hrpdb
Enter password: password
NOAUDIT POLICY orders_unified_audpol;
DROP AUDIT policy orders_unified_audpol;
```

(統合監査ポリシーは、作成したユーザーのスキーマではなく、SYSスキーマに存在します。)

2. ユーザーSYSTEMとしてSQL*Plusに接続します。

```
CONNECT SYSTEM -- Or, CONNECT SYSTEM@hrpdb
Enter password: password
```

3. ユーザーpolicy_adminおよびpolicy_auditorを削除します。

```
DROP USER policy_admin;
DROP USER policy_auditor;
```

4. 他のユーザーがOEを使用しない場合、このアカウントはロックして期限切れにできます。

```
ALTER USER OE PASSWORD EXPIRE ACCOUNT LOCK;
```

親トピック: 例: 非データベース・ユーザーの監査

27.3 事前定義の統合監査ポリシーを使用したアクティビティの監査

Oracle Databaseには、よく使用されるセキュリティ関連の監査設定を対象とする、事前定義の統合監査ポリシーがあります。

- ログオン失敗の事前定義済統合監査ポリシー
  ORA_LOGIN_LOGOUT統合監査ポリシーは、失敗したログオンのみを追跡し、その他のログオンは追跡しません。
- セキュア・オプションの事前定義の統合監査ポリシー
  ORA_SECURECONFIG統合監査ポリシーには、セキュアな構成監査オプションがすべて用意されています。
- Oracle Databaseパラメータ変更の事前定義の統合監査ポリシー
  ORA_DATABASE_PARAMETERポリシーでは、よく使用されるOracle Databaseのパラメータ設定を監査します。
- ユーザー・アカウントおよび権限管理の事前定義の統合監査ポリシー
  ORA_ACCOUNT_MGMTポリシーでは、よく使用されるユーザー・アカウントおよび権限の設定を監査します。
- Center for Internet Securityで推奨される事前定義の統合監査ポリシー
  ORA_CIS_RECOMMENDATIONSポリシーは、Center for Internet Security (CIS)で推奨される監査を実行します。
- Oracle Database Real Application Securityの事前定義の監査ポリシー
  事前定義の統合監査ポリシーをOracle Database Real Application Securityのイベントに使用できます。
- Oracle Database VaultのDVSYSおよびLBACSYSスキーマに対する事前定義統合監査ポリシー
  ORA_DV_AUDPOLの事前定義の統合監査ポリシーは、Oracle Database VaultのDVSYSおよびLBACSYSスキーマ・オブジェクトを監査します。
- Oracle Database Vaultのデフォルトのレルムおよびコマンド・ルールに対する事前定義の統合監査ポリシー
  ORA_DV_AUDPOL2の事前定義の統合監査ポリシーは、Oracle Database Vaultのデフォルトのレルムおよびコマンド・ルールを監査します。

関連トピック

- 一般的に使用されるセキュリティ関連アクティビティの監査

親トピック: 監査ポリシーの構成

27.3.1 ログオン失敗の事前定義の統合監査ポリシー

ORA_LOGIN_LOGOUT統合監査ポリシーは、失敗したログオンのみを追跡し、その他のログオンは追跡しません。

新規データベースの場合、このポリシーは、完全な統合監査モードと混合モードの両方の監査環境で、デフォルトで有効です。このポリシーは、旧バージョンからアップグレードされたデータベースに対して有効になりませんが、以前のリリースから新規データベースを作成し、そのデータベースを最新リリースにアップグレードした場合はこの限りではありません。なお、LOGON文の統合監査ポリシーを構成した場合は、直接ログインおよびALTER SESSION文とSET CONTAINER文の両方について、監査レコードが生成されます。

ノート:

ユーザーSYS のみが、この事前定義のポリシーを変更または削除できます。

次のCREATE AUDIT POLICY文は、ORA_LOGIN_LOGOUT統合監査ポリシー定義を示しています。

```
CREATE AUDIT POLICY ORA_LOGIN_LOGOUT ACTIONS LOGON;
```

ORA_LOGIN_LOGOUT統合監査ポリシーは、次のように有効にします。

```
AUDIT POLICY ORA_LOGIN_LOGOUT WHENEVER NOT SUCCESSFUL;
```

親トピック: 事前定義の統合監査ポリシーを使用したアクティビティの監査

27.3.2 セキュア・オプションの事前定義の統合監査ポリシー

ORA_SECURECONFIG統合監査ポリシーには、セキュアな構成監査オプションがすべて用意されています。

新規データベースの場合、このポリシーは、完全な統合監査モードと混合モードの両方の監査環境で、デフォルトで有効です。このポリシーは、旧バージョンからアップグレードされたデータベースに対して有効になりませんが、以前のリリースから新規データベースを作成し、そのデータベースを最新リリースにアップグレードした場合はこの限りではありません。

ノート:

ユーザーSYS のみが、この事前定義のポリシーを変更または削除できます。

次のCREATE AUDIT POLICY文は、ORA_SECURECONFIG統合監査ポリシー定義を示しています。

```
CREATE AUDIT POLICY ORA_SECURECONFIG
 PRIVILEGES ALTER ANY TABLE, CREATE ANY TABLE, DROP ANY TABLE,
            CREATE ANY PROCEDURE, DROP ANY PROCEDURE, ALTER ANY PROCEDURE,
            GRANT ANY PRIVILEGE, GRANT ANY OBJECT PRIVILEGE, GRANT ANY ROLE,
            AUDIT SYSTEM, CREATE EXTERNAL JOB, CREATE ANY JOB,
            CREATE ANY LIBRARY,
            EXEMPT ACCESS POLICY,
            CREATE USER, DROP USER,
            ALTER DATABASE, ALTER SYSTEM,
            CREATE PUBLIC SYNONYM, DROP PUBLIC SYNONYM,
            CREATE SQL TRANSLATION PROFILE, CREATE ANY SQL TRANSLATION
PROFILE,
            DROP ANY SQL TRANSLATION PROFILE, ALTER ANY SQL TRANSLATION
PROFILE,
            TRANSLATE ANY SQL,
            EXEMPT REDACTION POLICY,
            PURGE DBA_RECYCLEBIN, LOGMINING,
            ADMINISTER KEY MANAGEMENT, BECOME USER
 ACTIONS    ALTER USER, CREATE ROLE, ALTER ROLE, DROP ROLE,
            SET ROLE, CREATE PROFILE, ALTER PROFILE,
            DROP PROFILE, CREATE DATABASE LINK,
            ALTER DATABASE LINK, DROP DATABASE LINK,
            CREATE DIRECTORY, DROP DIRECTORY,
            CREATE PLUGGABLE DATABASE,
            DROP PLUGGABLE DATABASE,
            ALTER PLUGGABLE DATABASE,
            EXECUTE ON DBMS_RLS,
            ALTER DATABASE DICTIONARY;
```

親トピック: 事前定義の統合監査ポリシーを使用したアクティビティの監査

27.3.3 Oracle Databaseパラメータ変更の事前定義の統合監査ポリシー

ORA_DATABASE_PARAMETERポリシーでは、よく使用されるOracle Databaseのパラメータ設定を監査します。

ノート:

ユーザーSYS のみが、この事前定義のポリシーを変更または削除できます。

次のCREATE AUDIT POLICY文は、ORA_DATABASE_PARAMETER統合監査ポリシー定義を示しています。デフォルトでは、このポリシーは有効になっていません。

```
CREATE AUDIT POLICY ORA_DATABASE_PARAMETER
 ACTIONS ALTER DATABASE, ALTER SYSTEM, CREATE SPFILE;
```

親トピック: 事前定義の統合監査ポリシーを使用したアクティビティの監査

27.3.4 ユーザー・アカウントおよび権限管理の事前定義の統合監査ポリシー

ORA_ACCOUNT_MGMTポリシーでは、よく使用されるユーザー・アカウントおよび権限の設定を監査します。

ノート:

ユーザーSYS のみが、この事前定義のポリシーを変更または削除できます。

次のCREATE AUDIT POLICY文は、ORA_ACCOUNT_MGMT統合監査ポリシー定義を示しています。デフォルトでは、このポリシーは有効になっていません。

```
CREATE AUDIT POLICY ORA_ACCOUNT_MGMT
 ACTIONS CREATE USER, ALTER USER, DROP USER, CREATE ROLE, DROP ROLE,
  ALTER ROLE, SET ROLE, GRANT, REVOKE;
```

親トピック: 事前定義の統合監査ポリシーを使用したアクティビティの監査

27.3.5 Center for Internet Securityで推奨される事前定義の統合監査ポリシー

ORA_CIS_RECOMMENDATIONSポリシーは、Center for Internet Security (CIS)で推奨される監査を実行します。

ノート:

ユーザーSYS のみが、この事前定義のポリシーを変更または削除できます。

次のCREATE AUDIT POLICY文は、ORA_CIS_RECOMMENDATIONS統合監査ポリシー定義を示しています。デフォルトでは、このポリシーは有効になっていません。

```
CREATE AUDIT POLICY ORA_CIS_RECOMMENDATIONS
PRIVILEGES SELECT ANY DICTIONARY, ALTER SYSTEM
ACTIONS CREATE USER, ALTER USER, DROP USER,
        CREATE ROLE, DROP ROLE, ALTER ROLE,
        GRANT, REVOKE, CREATE DATABASE LINK,
        ALTER DATABASE LINK, DROP DATABASE LINK,
        CREATE PROFILE, ALTER PROFILE, DROP PROFILE,
        CREATE SYNONYM, DROP SYNONYM,
        CREATE PROCEDURE, DROP PROCEDURE,
        ALTER PROCEDURE, ALTER SYNONYM, CREATE FUNCTION,
        CREATE PACKAGE, CREATE PACKAGE BODY,
        ALTER FUNCTION, ALTER PACKAGE, ALTER SYSTEM,
        ALTER PACKAGE BODY, DROP FUNCTION,
        DROP PACKAGE, DROP PACKAGE BODY,
        CREATE TRIGGER, ALTER TRIGGER,
        DROP TRIGGER;
```

親トピック: 事前定義の統合監査ポリシーを使用したアクティビティの監査

27.3.6 Oracle Database Real Application Securityの事前定義の監査ポリシー

事前定義の統合監査ポリシーをOracle Database Real Application Securityのイベントに使用できます。

- システム管理者操作の事前定義の統合監査ポリシー
  ORA_RAS_POLICY_MGMTの事前定義の統合監査ポリシーでは、アプリケーション・ユーザー、ロールおよびポリシーのOracle Real Application Securityのすべて管理アクションのポリシーを監査します。
- セッション操作の事前定義の統合監査ポリシー
  ORA_RAS_SESSION_MGMTの事前定義の統合監査ポリシーでは、Oracle Real Application Securityのすべてのランタイム・セッション・アクションおよびネームスペース・アクションのポリシーを監査します。

関連トピック

- Oracle Database Real Application Securityイベントの監査

親トピック: 事前定義の統合監査ポリシーを使用したアクティビティの監査

27.3.6.1 システム管理者操作の事前定義の統合監査ポリシー

ORA_RAS_POLICY_MGMTの事前定義の統合監査ポリシーでは、アプリケーション・ユーザー、ロールおよびポリシーのOracle Real Application Securityのすべて管理アクションのポリシーを監査します。

ノート:

ユーザーSYS のみが、この事前定義のポリシーを変更または削除できます。

次のCREATE AUDIT POLICY文は、ORA_RAS_POLICY_MGMT統合監査ポリシー定義を示しています。デフォルトでは、このポリシーは有効になっていません。

```
CREATE AUDIT POLICY ORA_RAS_POLICY_MGMT
 ACTIONS COMPONENT=XS
  CREATE USER, UPDATE USER, DELETE USER,
  CREATE ROLE, UPDATE ROLE, DELETE ROLE, GRANT ROLE, REVOKE ROLE,
  ADD PROXY, REMOVE PROXY,
  SET USER PASSWORD, SET USER VERIFIER, SET USER PROFILE,
  CREATE ROLESET, UPDATE ROLESET, DELETE ROLESET,
  CREATE SECURITY CLASS, UPDATE SECURITY CLASS, DELETE SECURITY CLASS,
  CREATE NAMESPACE TEMPLATE, UPDATE NAMESPACE TEMPLATE, DELETE NAMESPACE TEMPLATE,
  CREATE ACL, UPDATE ACL, DELETE ACL,
  CREATE DATA SECURITY, UPDATE DATA SECURITY, DELETE DATA SECURITY,
  ENABLE DATA SECURITY, DISABLE DATA SECURITY,
  ADD GLOBAL CALLBACK, DELETE GLOBAL CALLBACK, ENABLE GLOBAL CALLBACK;
```

親トピック: Oracle Database Real Application Securityの事前定義の監査ポリシー

27.3.6.2 セッション操作の事前定義の統合監査ポリシー

ORA_RAS_SESSION_MGMTの事前定義の統合監査ポリシーでは、Oracle Real Application Securityのすべてのランタイム・セッション・アクションおよびネームスペース・アクションのポリシーを監査します。

ノート:

ユーザーSYS のみが、この事前定義のポリシーを変更または削除できます。

次のCREATE AUDIT POLICY文は、ORA_RAS_SESSION_MGMT統合監査ポリシー定義を示しています。デフォルトでは、このポリシーは有効になっていません。

```
CREATE AUDIT POLICY ORA_RAS_SESSION_MGMT
 ACTIONS COMPONENT=XS
  CREATE SESSION, DESTROY SESSION,
  ENABLE ROLE, DISABLE ROLE,
  SET COOKIE, SET INACTIVE TIMEOUT,
  SWITCH USER, ASSIGN USER,
  CREATE SESSION NAMESPACE, DELETE SESSION NAMESPACE,
  CREATE NAMESPACE ATTRIBUTE, GET NAMESPACE ATTRIBUTE, SET NAMESPACE ATTRIBUTE,
  DELETE NAMESPACE ATTRIBUTE;
```

親トピック: Oracle Database Real Application Securityの事前定義の監査ポリシー

27.3.7 DVSYSおよびLBACSYSスキーマに対するOracle Database Vaultの事前定義の統合監査ポリシー

事前定義のORA_DV_AUDPOL統合監査ポリシーで、Oracle Database VaultのDVSYSおよびLBACSYSスキーマ・オブジェクトを監査します。

ORA_DV_AUDPOLポリシーは、Oracle Database VaultのDVSYS (DVFを含む)スキーマ・オブジェクトと、Oracle Label SecurityのLBACSYSスキーマ・オブジェクトに対して実行されるすべてのアクションを監査します。DVFスキーマのF$*ファクタ・ファンクションに関するアクションは取得しません。デフォルトでは、このポリシーは有効になっていません。

ノート:

ユーザーSYS のみが、この事前定義のポリシーを変更または削除できます。

このポリシーの完全な定義を表示するには、policy_nameがORA_DV_AUDPOLのAUDIT_UNIFIED_POLICIESデータ・ディクショナリ・ビューを問い合せます。

関連トピック

- Oracle Database Vaultイベントの監査

親トピック: 事前定義の統合監査ポリシーを使用したアクティビティの監査

27.3.8 デフォルト・レルムおよびコマンド・ルールに対するOracle Database Vaultの事前定義の統合監査ポリシー

事前定義のORA_DV_AUDPOL2の統合監査ポリシーで、Oracle Database Vaultのデフォルトのレルムおよびコマンド・ルールが監査されます。

ORA_DV_AUDPOL2ポリシーには、Oracle Database Vaultで提供されるデフォルトのレルムおよびコマンド・ルールの監査設定が定義されています。デフォルトでは、このポリシーは有効になっていません。

ノート:

ユーザーSYS のみが、この事前定義のポリシーを変更または削除できます。

このポリシーの完全な定義を表示するには、policy_nameがORA_DV_AUDPOL2のAUDIT_UNIFIED_POLICIESデータ・ディクショナリ・ビューを問い合せます。

関連トピック

- Oracle Database Vaultイベントの監査

親トピック: 事前定義の統合監査ポリシーを使用したアクティビティの監査

27.4 ファイングレイン監査を使用した特定のアクティビティの監査

ファイングレイン監査では、非常に詳細なレベルで監査ポリシーを作成できます。

- ファイングレイン監査について
  ファイングレイン監査では、ポリシーを作成して、監査が実行される特定の条件を定義できます。
- ファイングレイン監査レコードが格納される場所
  ファイングレイン監査のレコードは、AUDSYSスキーマに格納されます。
- ファイングレイン監査の実行者
  Oracleには、ファイングレイン監査ポリシーを作成したり、ファイングレイン監査ポリシーのデータを表示および分析するために必要な権限のロールが用意されています。
- Oracle VPDポリシーがある表またはビューでのファイングレイン監査
  この監査証跡は、Oracle VPDポリシーに含まれているファイングレイン監査表またはビューからVPD述語を取得します。
- マルチテナント環境でのファイングレイン監査
  ファイングレイン監査ポリシーは、CDBルート、アプリケーション・ルート、CDB PDBおよびアプリケーションPDBで作成できます。
- ファイングレイン監査ポリシーとエディション
  エディション・ベースの再定義のアプリケーションを作成し、アプリケーションが編集ビューで使用する各表を対象とすることができます。
- DBMS_FGA PL/SQLパッケージを使用したファイングレイン監査ポリシーの管理
  DBMS_FGA PL/SQLパッケージで、ファイングレイン監査ポリシーを管理します。
- 例: ファイングレイン監査ポリシーへの電子メール・アラートの追加
  このチュートリアルでは、ユーザーがポリシーに違反したときに電子メールのアラートを生成するファイングレイン監査ポリシーの作成方法を示します。

関連トピック

- 特定のファイングレイン・アクティビティの監査

親トピック: 監査ポリシーの構成

27.4.1 ファイングレイン監査について

ファイングレイン監査では、ポリシーを作成して、監査が実行される特定の条件を定義できます。

ファイングレイン監査を使用して統合監査ポリシーは作成できませんが、データのアクセス時間の監査など、詳細にカスタマイズさ

れた監査設定はファイングレイン監査を使用して作成できます。

これにより、内容に基づいてデータ・アクセスを監視できるようになります。問合せと、INSERT、UPDATE、およびDELETE操作に対して詳細な監査を提供します。ファイングレイン監査を使用すると、次のタイプのアクションを監査できます。

- 午後9時から午前6時の間、または土曜日と日曜日に表にアクセスする場合
- 社内ネットワーク外部のIPアドレスを使用する場合
- 表の列を選択または更新する場合
- 表の列の値を変更する場合

通常、ファイングレイン監査ポリシーは、選択的監査の条件である、表オブジェクトに対する単純なユーザー定義SQL述語に基づいています。フェッチ中に行がポリシーの条件を満たすと、その問合せが監査対象となります。

統合監査ポリシーでは、次のアクションを除き、ファイングレイン監査ポリシーで実行できるほとんどの操作を実行できます。

- 特定の列の監査。給与や社会保障番号など、機密情報が格納されている特定の関連する列を監査できます。
- イベント・ハンドラの使用。たとえば、夜中に変更されないようにする必要がある監査対象の列が更新された場合に、セキュリティ管理者に電子メール・アラートを送信する関数を作成できます。

ファイングレイン監査には、統合監査と比べて次のような利点があります。

- 行値ベースの監査を実行できます。たとえば、更新された値が指定されたしきい値より大きい場合にsalary列の更新を監査できますが、それ以外の場合は監査できません。
- ファイングレイン監査イベント・ハンドラを使用して、管理者または他のユーザーに特定のイベントを事前に通知できます。
- PL/SQLでBULK COLLECTおよびFORALLを使用したバルク・データ処理操作中に、異なるバインド変数に対して同じDML文を繰り返し実行すると、ファイングレイン監査では、繰返し実行される文を適切なバインド変数値で取得できます。

ノート:

- ファイングレイン監査は、コストベースの最適化でのみサポートされています。ルールベースの最適化を使用する問合せでは、行フィルタを適用する前にファイングレイン監査が行われるため、不要な監査イベント・トリガーが発生します。

- フラッシュバック問合せに含まれるオブジェクトで現在有効なポリシーが、指定したフラッシュバック・スナップショット(時間またはシステム変更番号(SCN)に基づく)から戻されたデータに適用されます。
- ファイングレイン監査を使用して直接ロードされるデータを監査する場合(たとえば Oracle Warehouse Builder を使用して DML 文を実行する場合)、Oracle Database は透過的にデータベース・インスタンスで実行されているすべてのダイレクト・ロードを従来型ロードにします。データのダイレクト・ロードを保持する場合は、かわりに統合監査ポリシーの使用を検討してください。

親トピック: ファイングレイン監査を使用した特定のアクティビティの監査

27.4.2 ファイングレイン監査レコードが格納される場所

ファイングレイン監査のレコードは、AUDSYSスキーマに格納されます。

これらの監査レコードは、デフォルトでSYSAUX表領域に格納されます。
DBMS_AUDIT_MGMT.SET_AUDIT_TRAIL_LOCATIONプロシージャを使用して、新しい表領域を指定できます。この表領域は暗号化された表領域にできます。有効な監査ポリシーに対して生成されたレコードを検索するには、UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューを問い合せます。

監査証跡では、SQL文内の表またはビューの参照ごとに監査レコードが取得されます。たとえば、HR.EMPLOYEES表を2回参照するUNION文を実行した場合、文の監査ポリシーによって2つ(HR.EMPLOYEES表へのアクセスごとに1つ)の監査レコードが生成されます。

関連項目:

- 強制的に監査されるアクティビティ
- DBMS_AUDIT_MGMT.SET_AUDIT_TRAIL_LOCATIONプロシージャの詳細は、『Oracle Database PL/SQLパッケージおよびタイプ・リファレンス』を参照してください。
- UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューの詳細は、『Oracle Databaseリファレンス』を参照してください。

親トピック: ファイングレイン監査を使用した特定のアクティビティの監査

27.4.3 ファイングレイン監査の実行者

Oracleには、ファイングレイン監査ポリシーを作成したり、ファイングレイン監査ポリシーのデータを表示および分析するために必要な権限のロールが用意されています。

ファイングレイン監査には、次の権限があります。

- ファイングレイン監査ポリシーを作成するには、AUDIT_ADMINロールまたはDBMS_FGAパッケージに対するEXECUTE権限が付与されている必要があります。
- ファイングレイン監査データを表示および分析するには、AUDIT_VIEWERロールが付与されている必要があります。

PL/SQLパッケージにはAUDIT_ADMINロールがすでに付与されています。すべての権限と同様に、これらのロールは信頼できるユーザーにのみ付与してください。DBA_ROLE_PRIVSデータ・ディクショナリ・ビューを問い合せることで、ユーザーに付与されているロールを確認できます。

親トピック: ファイングレイン監査を使用した特定のアクティビティの監査

27.4.4 Oracle VPDポリシーがある表またはビューでのファイングレイン監査

この監査証跡は、Oracle VPDポリシーに含まれているファイングレイン監査表またはビューからVPD述語を取得します。

この動作は、統合監査証跡で統合監査ポリシーのVPD述語を取得する場合の動作に似ています。

監査証跡では、Oracle Label SecurityおよびOracle Real Application Securityのポリシーの内部述語も取得されます。

VPD述語監査レコードを取得するために特別な監査ポリシーを作成する必要はありません。述語情報は、DBA_FGA_AUDIT_TRAILおよびUNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューのRLS_INFO列に自動的に格納されます。

同じ表またはビューに適用されるVPDポリシーが複数ある場合、これらのポリシーの述部は、デフォルトでRLS_INFO列に連結

されます。各述語がそれ自体の行(対応するVPDポリシー名などの情報で特定)に含まれるように出力を再フォーマットするには、DBMS_AUDIT_UTIL PL/SQL パッケージのファンクションを使用します。

関連項目:

- VPD述語の監査の詳細およびDBMS_AUDIT_UTILパッケージのファンクションを使用して取得済の監査データをフォーマットする方法の例は、Oracle Virtual Private Databaseの述語の監査を参照してください
- DBMS_AUDIT_UTIL PL/SQLパッケージの詳細は、Oracle Database PL/SQLパッケージおよびタイプ・リファレンスを参照してください

親トピック: ファイングレイン監査を使用した特定のアクティビティの監査

27.4.5 マルチテナント環境でのファイングレイン監査

ファイングレイン監査ポリシーは、CDBルート、アプリケーション・ルート、CDB PDBおよびアプリケーションPDBで作成できます。

マルチテナント環境におけるファイングレイン監査ポリシーには、次のような一般的なルールがあります。

- ファイングレイン監査ポリシーは、SYSオブジェクトに対して作成できません。
- ファイングレイン監査ポリシーは(ローカルまたはアプリケーション共通を問わず)、拡張データ・リンク・オブジェクトに対して作成できません。
- CDBルートでファイングレイン監査ポリシーを作成する場合、すべてのPDBにポリシーを適用することはできません。ポリシーはCDBルート内のオブジェクトに適用されます。(つまり、CDBルートに対する共通のファイングレイン監査ポリシーは存在しません。)すべてのPDBで共通オブジェクトのアクセスを監査するようにファイングレイン監査ポリシーを作成する場合は、監査ポリシーを各PDBで明示的に作成し、PDBでアクセス可能にする共通オブジェクトに対してそのポリシーを有効化する必要があります。
- PDBでファイングレイン監査ポリシーを作成する場合、ポリシーはPDB内のオブジェクトにのみ適用されます。
- アプリケーション共通ファイングレイン監査ポリシーは、アプリケーション・ルートに接続し、BEGIN/ENDブロック内にいる場合にのみ作成できます。アプリケーション・ルートに接続し、BEGIN/ENDブロック外でファイングレイン監査ポリシーを作成すると、ファイングレイン監査ポリシーはアプリケーション・ルートに作成されます。
- アプリケーション共通ファイングレイン監査ポリシーは、ローカルPDBオブジェクトに対して作成できません。
- アプリケーション共通ファイングレイン監査ポリシーにハンドラがある場合、このハンドラはアプリケーション共通ユーザーまたはCDB共通ユーザーによって所有されている必要があります。
- アプリケーション・ファイングレイン監査ポリシーは、ローカル(PDB)オブジェクトおよびCDB共通オブジェクトに対して作成できます。ポリシーはそのコンテナに対してローカルであるため、ポリシーが定義されたオブジェクトは、ポリシーが定義された特定のコンテナ内でのみ監査されます。たとえば、ファイングレイン監査ポリシーをhr_pdb PDBで作成する場合、このポリシーを作成する対象のオブジェクトは、hr_pdb PDB内に存在する必要があります。
- ローカル・ファイングレイン監査ポリシーは、アプリケーションPDB内のオブジェクト・リンク・オブジェクトおよび拡張データ・リンク・オブジェクトに対して作成できません。メタデータリンク・オブジェクトは、ファイングレイン監査ポリシーで使用できます。
- アプリケーション・ルート・ローカル・ポリシーは、アプリケーション共通オブジェクトに対して使用できます。
- ファイングレイン監査ポリシーを共通監査ポリシーとしてアプリケーション・ルートで作成する場合、このアプリケーション・ルートに属する各PDBで有効になります。したがって、アプリケーションPDBのアプリケーション共通オブジェクトおよび

CDB共通オブジェクト(アプリケーション共通ファイングレイン監査ポリシーが定義されたもの)は、そのアプリケーションPDB内のファイングレイン監査証跡において監査されます。

- アプリケーションのインストール、アップグレード、パッチ適用またはアンインストール操作用のスクリプトを作成する際、`ALTER PLUGGABLE DATABASE app_name BEGIN INSTALL`および`ALTER PLUGGABLE DATABASE app_name END INSTALL`ブロック内にSQL文を含めて、様々な操作を実行できます。ファイングレイン監査ポリシー文は、これらのブロック内にのみ含めることができます。
- アプリケーション共通ファイングレイン監査ポリシーの有効化、無効化または削除の実行は、アプリケーション・ルートから、およびスクリプト内の`ALTER PLUGGABLE DATABASE app_name BEGIN INSTALL`および`ALTER PLUGGABLE DATABASE app_name END INSTALL`ブロック内からに限定されます。

親トピック: ファイングレイン監査を使用した特定のアクティビティの監査

27.4.6 ファイングレイン監査ポリシーとエディション

エディション・ベースの再定義のアプリケーションを作成し、アプリケーションが編集ビューで使用する各表を対象とすることができます。

これを行う場合、編集ビューに対してこれらの表を保護するファイングレイン監査ポリシーを移動する必要があります。DBA_EDITIONSデータ・ディクショナリ・ビューを問い合せることで、現在構成されているエディションについて情報を確認できます。ファイングレイン監査ポリシーに関する情報を確認するには、DBA_AUDIT_POLICIESを問い合せます。

親トピック: ファイングレイン監査を使用した特定のアクティビティの監査

27.4.7 DBMS_FGA PL/SQLパッケージを使用したファイングレイン監査ポリシーの管理

DBMS_FGA PL/SQLパッケージで、ファイングレイン監査ポリシーを管理します。

- DBMS_FGA PL/SQL PL/SQLパッケージについて
  DBMS_FGA PL/SQLパッケージを使用して複数の文を1つのポリシーにまとめ、その他のファイングレイン監査管理タスクを実行できます。
- DBMS_FGA PL/SQLパッケージとエディション
  エディション環境で使用するためのDBMS_FGAポリシーを作成できます。
- マルチテナント環境でのDBMS_FGA PL/SQLパッケージ
  マルチテナント環境では、DBMS_FGA PL/SQLパッケージは現在のローカルPDBのみに適用されます。
- ファイングレイン監査ポリシーの作成
  `DBMS_FGA.ADD_POLICY`プロシージャで、ファイングレイン監査ポリシーを作成します。
- 例: DBMS_FGA.ADD_POLICYを使用してファイングレイン監査ポリシーを作成する方法
  `DBMS_FGA.ADD_POLICY`プロシージャで、複数の文タイプを使用してファイングレイン監査ポリシーを作成できます。
- ファイングレイン監査ポリシーを使用禁止にする方法
  `DBMS_FGA.DISABLE_POLICY`プロシージャで、ファイングレイン監査ポリシーを無効にします。
- ファイングレイン監査ポリシーを使用可能にする方法
  `DBMS_FGA.ENABLE_POLICY`プロシージャで、ファイングレイン監査ポリシーを有効にします。
- ファイングレイン監査ポリシーの削除
  `DBMS_FGA.DROP_POLICY`プロシージャで、ファイングレイン監査ポリシーを削除します。

親トピック: ファイングレイン監査を使用した特定のアクティビティの監査

27.4.7.1 DBMS_FGA PL/SQL PL/SQLパッケージについて

DBMS_FGA PL/SQLパッケージを使用して複数の文を1つのポリシーにまとめ、その他のファイングレイン監査管理タスクを実行できます。

ただし、列レベルの監査を実行しない場合や、イベント・ハンドラと監査ポリシーを使用しない場合は、統合監査ポリシーおよびAUDIT文を使用したアクティビティの監査で説明されているように、監査ポリシーを作成する必要があります。

DBMS_FGA PL/SQLパッケージを使用すると、SELECT、INSERT、UPDATEおよびDELETE文のすべての組合せを1つのポリシーに追加できます。また、基礎となるアクションのINSERTおよびUPDATEを監査することによって、MERGE文も監査できます。MERGE文を監査するには、INSERTおよびUPDATE文に対するファイングレイン・アクセスを構成します。成功したMERGE操作についてポリシーごとにレコードが1つのみ生成されます。

ファイングレイン監査ポリシーを管理するには、AUDIT_ADMINロールを付与する必要があります。DBMS_FGAパッケージのEXECUTE権限は強制的に監査されることにも注意してください。

監査ポリシーは、監査ポリシーを作成した表にバインドされます。これにより、それぞれのアプリケーションではなく、データベースで一度だけポリシーを変更すればよいため、監査ポリシーの管理が容易です。また、データベースへの接続方法(接続元がアプリケーション、Webインタフェース、SQL*PlusやOracle SQL Developerのいずれであるか)に関係なく、ポリシーに影響を与えるアクションがすべて記録されます。

問合せから戻された行が定義した監査条件と一致すると、ファイングレイン監査証跡に監査エントリが挿入されます。このエントリでは、通常の監査証跡でレポートされるすべての情報が除外されます。つまり、TRUEと評価されたすべてのファイングレイン監査ポリシーに対して、1行の監査情報のみが監査証跡に挿入されます。

関連項目:

DBMS_FGAパッケージの詳細は、『Oracle Database PL/SQLパッケージおよびタイプ・リファレンス』を参照してください。

親トピック: 「DBMS_FGA PL/SQLパッケージを使用したファイングレイン監査ポリシーの管理」

27.4.7.2 DBMS_FGA PL/SQLパッケージとエディション

エディション環境で使用するためのDBMS_FGAポリシーを作成できます。

DBMS_FGAパッケージ・ポリシーを異なる複数のエディションで使用する場合、ポリシーの結果を制御できます。つまり、結果をすべてのエディションで同一にするか、またはポリシーが使用されているエディションに固有にできます。

関連トピック

- エディションがグローバル・アプリケーション・コンテキストのPL/SQLパッケージの結果に与える影響

親トピック: 「DBMS_FGA PL/SQLパッケージを使用したファイングレイン監査ポリシーの管理」

27.4.7.3 マルチテナント環境でのDBMS_FGA PL/SQLパッケージ

マルチテナント環境では、DBMS_FGA PL/SQLパッケージは現在のローカルPDBのみに適用されます。

マルチテナント環境全体に1つのポリシーを作成することはできません。PDB内でオブジェクトにポリシーを指定する必要があります。PDBを確認するには、DBA_PDBSデータ・ディクショナリ・ビューを問い合せます。現在のPDBの名前を見つけるには、show con_nameコマンドを発行します。

親トピック: 「DBMS_FGA PL/SQLパッケージを使用したファイングレイン監査ポリシーの管理」

27.4.7.4 ファイングレイン監査ポリシーの作成

DBMS_FGA.ADD_POLICYプロシージャで、ファイングレイン監査ポリシーを作成します。

- ファイングレイン監査ポリシーの作成について
  DBMS_FGA.ADD_POLICYプロシージャで、指定された述語を監査条件として使用し、監査ポリシーを作成します。
- ファイングレイン監査ポリシーの作成の構文
  DBMS_FGA.ADD_POLICYプロシージャには、複雑な監査のハンドラを使用する機能など、様々な設定が含まれています。
- 特定の列および行の監査
  条件が一致した列(関連列)を監査対象にするなど、監査動作を細かく調節できます。

親トピック: 「DBMS_FGA PL/SQLパッケージを使用したファイングレイン監査ポリシーの管理」

27.4.7.4.1 ファイングレイン監査ポリシーの作成について

DBMS_FGA.ADD_POLICYプロシージャで、指定された述語を監査条件として使用し、監査ポリシーを作成します。

デフォルトでは、ポリシーを作成したユーザーの権限で、ポリシーの述語がOracle Databaseによって実行されます。表オブジェクトまたはビュー・オブジェクトに設定可能なファイングレイン・ポリシーの最大数は256です。Oracle Databaseでは、ポリシーはデータ・ディクショナリ表に格納されますが、SYSスキーマ内に存在しない表またはビューに関するポリシーを作成できます。マルチテナント環境では、ファイングレイン・ポリシーはローカルPDBでのみ作成されます。

ファイングレイン監査ポリシーを作成する実表にマテリアライズド・ビューを作成する場合は、同じ表にマテリアライズド・ビューを作成する前に、実表にファイングレイン監査ポリシーを作成する必要があります。そうしないと、マテリアライズド・ビューに対するリフレッシュ操作はORA-12008: 「マテリアライズド・ビューのリフレッシュ・パスでエラーが発生しました。」エラーで失敗します。

ファイングレイン監査ポリシーを作成後に変更することはできません。ポリシーを変更する必要がある場合は、削除してから再作成します。

ファイングレイン監査ポリシーに関する情報を検索するには、ALL_AUDIT_POLICIES、DBA_AUDIT_POLICIESおよびUSER_AUDIT_POLICIESビューを問い合せます。UNIFIED_AUDIT_TRAILビューには、FGA_POLICY_NAMEという名前の列が含まれ、この列を使用すると、特定のファイングレイン監査ポリシーを使用して生成された行をフィルタできます。

親トピック: ファイングレイン監査ポリシーの作成

27.4.7.4.2 ファイングレイン監査ポリシーの作成の構文

DBMS_FGA.ADD_POLICYプロシージャには、複雑な監査のハンドラを使用する機能など、様々な設定が含まれています。

DBMS_FGA.ADD_POLICYプロシージャの構文は次のとおりです。

```
DBMS_FGA.ADD_POLICY(
   object_schema      IN  VARCHAR2 DEFAULT NULL
   object_name        IN  VARCHAR2,
   policy_name        IN  VARCHAR2,
   audit_condition    IN  VARCHAR2 DEFAULT NULL,
   audit_column       IN  VARCHAR2 DEFAULT NULL
   handler_schema     IN  VARCHAR2 DEFAULT NULL,
   handler_module     IN  VARCHAR2 DEFAULT NULL,
   enable             IN  BOOLEAN DEFAULT TRUE,
   statement_types    IN  VARCHAR2 DEFAULT SELECT,
   audit_trail        IN  BINARY_INTEGER DEFAULT NULL,
   audit_column_opts  IN  BINARY_INTEGER DEFAULT ANY_COLUMNS,
   policy_owner       IN  VARCHAR2 DEFAULT NULL);
```

詳細は、次のとおりです。

- object_schemaには、監査するオブジェクトのスキーマを指定します。(NULLの場合、現行のログオン・ユーザーのスキーマと想定されます。)
- object_nameには、監査するオブジェクトの名前を指定します。
- policy_nameには、作成するポリシーの名前を指定します。この名前は必ず一意にしてください。
- audit_conditionには、行のブール条件を指定します。NULLも指定できます(TRUEとして機能します)。詳細は、特定の列および行の監査を参照してください。監査条件にNULLを指定するか、何も指定しない場合は、そのポリシーが設定された表に対するアクションが行われると、行が戻されるかどうかに関係なく監査レコードが作成されます。

  次のガイドラインに従ってください。

  - ファンクションは、同じ実表で監査可能な文を実行するため、audit_condition設定に含めないでください。たとえば、HR.EMPLOYEES表に対してINSERT文を実行するファンクションを作成するとします。ポリシーのaudit_conditionには、このファンクションが含まれていて、これは(statement_typesにより設定される)INSERT文のポリシーです。このポリシーが使用されると、ファンクションはシステムのメモリーが足りなくなるまで再帰的に実行します。これにより、ORA-1000: 最大オープン・カーソル数を超えました。またはORA-00036: 再帰的SQLレベルの最大値(50)を超えましたのエラーが発生する場合があります。
  - DBMS_FGA.ENABLE_POLICY文またはDBMS_FGA.DISABLE_POLICY文を、ポリシーの条件に含まれるファンクションから発行しないでください。
- audit_columnには、監査対象の1つ以上の列(非表示列も含む)を指定します。NULLに設定するかまたは省略すると、すべての列が監査されます。Oracle Label Securityの非表示列やオブジェクト・タイプ列も対象になります。デフォルトのNULLの場合、アクセスまたは影響を受ける列があれば監査が行われます。
- handler_schema: ポリシーに違反した場合の応答のトリガーにアラートが使用される場合は、イベント・ハンドラが含まれているスキーマの名前を指定します。デフォルトのNULLでは、現行のスキーマが使用されます。例: ファイングレイン監査ポリシーへの電子メール・アラートの追加も参照してください。
- handler_moduleには、イベント・ハンドラの名前を指定します。イベント・ハンドラが含まれるパッケージも対象になります。このファンクションは、問合せの監査条件と一致する最初の行が処理された後でのみ実行されます。

  次のガイドラインに従ってください。

  - 再帰的ファイングレイン監査ハンドラを作成しないでください。たとえば、HR.EMPLOYEES表に対してINSERT文を実行するハンドラを作成するとします。このハンドラに関連付けられるポリシーは、(statement_typesパラメータにより設定される)INSERT文のポリシーです。このポリシーが使用されると、ハンドラはシステムのメモリーが足りなくなるまで再帰的に実行します。これにより、ORA-1000: 最大オープン・カーソル数を超えました。またはORA-00036: 再帰的SQLレベルの最大値(50)を超えましたのエラーが発生する場合があります。
  - DBMS_FGA.ENABLE_POLICY文またはDBMS_FGA.DISABLE_POLICY文をポリシー・ハンドラから発行しないでください。これらの文を発行すると、 ORA-28144: ファイングレイン監査ハンドラの実行に失敗しましたエラーが発生する場合があります。
- enableは、TRUEまたはFALSEを使用してポリシーを使用可能または使用禁止にします。省略した場合、ポリシーは使用可能になります。デフォルトはTRUEです。
- statement_types: 監査対象のSQL文を指定します。INSERT、UPDATE、DELETEまたはSELECTのみです。

MERGE操作を監査する場合は、statement_typesを'INSERT,UPDATE'に設定します。デフォルト値はSELECTです。

- audit_trail: 統合監査に移行している場合は、Oracle Databaseによってこのパラメータは無視され、監査レコードが統合監査証跡に即座に書き込まれます。統合監査に移行している場合は、このパラメータを省略します。

  クレジット・カード情報などの機密データもクリアテキストで記録できることに注意してください。

- audit_column_opts: audit_columnパラメータで複数の列を指定した場合は、すべての列を監査するか特定の列を監査するかをこのパラメータで決定します。詳細は、特定の列および行の監査を参照してください。

- policy_ownerは、ファイングレイン監査ポリシーを所有するユーザーです。ただし、この設定はユーザー指定の引数ではありません。Oracle Data Pumpクライアントは、この設定を内部で使用して、ファイングレイン監査ポリシーを適宜再作成します。

関連項目:

DBMS_FGA.ADD_POLICYの構文の詳細は、『Oracle Database PL/SQLパッケージおよびタイプ・リファレンス』を参照してください。

親トピック: ファイングレイン監査ポリシーの作成

27.4.7.4.3 特定の列および行の監査

条件が一致した列(関連列)を監査対象にするなど、監査動作を細かく調節できます。

これを実行するには、audit_columnパラメータを使用して、機密情報が含まれた列を1つ以上指定します。また、audit_conditionパラメータを使用してブール条件を定義すると、特定の行のデータを監査できます。(ただし、ポリシーで条件の監査のみが必要な場合は、統合監査ポリシーの条件の作成で説明されている監査ポリシー条件の使用を検討してください。)

例27-47では、次の設定によって、部門50 (DEPARTMENT_ID = 50)内のユーザーがSALARY列とCOMMISSION_PCT列にアクセスしようとしたときに監査を実行できるようになりますます。

```
audit_condition    => 'DEPARTMENT_ID = 50',
audit_column       => 'SALARY,COMMISSION_PCT,'
```

この機能は非常に有用です。監査対象を特定の重要なデータ・タイプに限定できるだけでなく、社会保障番号、給与情報、診断書などの機密データを含む列をより強力に保護できます。

audit_columnに複数の列がリストされている場合は、audit_column_optsパラメータを使用すると、文の監査が、audit_columnパラメータで指定されたいずれかの列が問合せで参照されたときに実行されるか、またはすべての列が参照されたときにのみ実行されるかを指定できます。たとえば:

```
audit_column_opts   => DBMS_FGA.ANY_COLUMNS,
audit_column_opts   => DBMS_FGA.ALL_COLUMNS,
```

関連列を指定しない場合、監査はすべての列に適用されます。

関連項目:

DBMS_FGA.ADD_POLICYプロシージャ内のaudit_condition、audit_columnおよびaudit_column_optsパラメータの詳細は、『Oracle Database PL/SQLパッケージおよびタイプ・リファレンス』を参照してください(その項の

ADD_POLICYプロシージャの使用上のノートも参照してください)。

親トピック: ファイングレイン監査ポリシーの作成

27.4.7.5 例: DBMS_FGA.ADD_POLICYを使用してファイングレイン監査ポリシーを作成する方法

DBMS_FGA.ADD_POLICYプロシージャで、複数の文タイプを使用してファイングレイン監査ポリシーを作成できます。

例27-47に、表HR.EMPLOYEESに対する文INSERT、UPDATE、DELETEおよびSELECTを監査する方法を示します。

この例では、audit_column_optsパラメータが必須パラメータではないため省略されていることに注意してください。

例27-47 DBMS_FGA.ADD_POLICYを使用してファイングレイン監査ポリシーを作成する方法

```
BEGIN
  DBMS_FGA.ADD_POLICY(
   object_schema      => 'HR',
   object_name        => 'EMPLOYEES',
   policy_name        => 'chk_hr_employees',
   audit_column       => 'SALARY',
   enable             =>  TRUE,
   statement_types    => 'INSERT, UPDATE, SELECT, DELETE');
END;
/
```

ポリシーを作成した後、DBA_AUDIT_POLICIESビューを問い合せると、新しいポリシーがリストされることを確認できます。

```
SELECT POLICY_NAME FROM DBA_AUDIT_POLICIES;
POLICY_NAME
------------------------------
CHK_HR_EMPLOYEES
```

その後、次のようなSQL文を発行すると、監査イベント・レコードが記録されます。

```
SELECT COUNT(*) FROM HR.EMPLOYEES WHERE COMMISSION_PCT = 20 AND SALARY > 4500;
SELECT SALARY FROM HR.EMPLOYEES WHERE DEPARTMENT_ID = 50;
DELETE FROM HR.EMPLOYEES WHERE SALARY > 1000000;
```

親トピック: 「DBMS_FGA PL/SQLパッケージを使用したファイングレイン監査ポリシーの管理」

27.4.7.6 ファイングレイン監査ポリシーを使用禁止にする方法

DBMS_FGA.DISABLE_POLICYプロシージャで、ファイングレイン監査ポリシーを無効にします。

- 次の構文を使用して、ファイングレイン監査ポリシーを無効にします。

```
DBMS_FGA.DISABLE_POLICY(
   object_schema  VARCHAR2,
   object_name    VARCHAR2,
   policy_name    VARCHAR2);
```

たとえば、例27-47で作成されたファイングレイン監査ポリシーを無効にするには、次のようにします。

```
BEGIN
 DBMS_FGA.DISABLE_POLICY(
  object_schema        => 'HR',
  object_name          => 'EMPLOYEES',
  policy_name          => 'chk_hr_employees');
END;
/
```

関連項目:

DISABLE_POLICYの構文の詳細は、『Oracle Database PL/SQLパッケージおよびタイプ・リファレンス』を参照してください。

親トピック: 「DBMS_FGA PL/SQLパッケージを使用したファイングレイン監査ポリシーの管理」

27.4.7.7 ファイングレイン監査ポリシーを使用可能にする方法

DBMS_FGA.ENABLE_POLICYプロシージャで、ファイングレイン監査ポリシーを有効にします。

- 次の構文を使用して、ファイングレイン監査ポリシーを有効にします。

```
DBMS_FGA.ENABLE_POLICY(
   object_schema   VARCHAR2,
   object_name     VARCHAR2,
   policy_name     VARCHAR2,
   enable          BOOLEAN);
```

たとえば、DBMS_FGA.ENABLE_POLICYプロシージャを使用してchk_hr_empポリシーを再度使用可能にするとします。

```
BEGIN
 DBMS_FGA.ENABLE_POLICY(
  object_schema         => 'HR',
  object_name           => 'EMPLOYEES',
  policy_name           => 'chk_hr_employees',
  enable                => TRUE);
END;
/
```

関連項目:

ENABLE_POLICYの構文の詳細は、『Oracle Database PL/SQLパッケージおよびタイプ・リファレンス』を参照してください。

親トピック: 「DBMS_FGA PL/SQLパッケージを使用したファイングレイン監査ポリシーの管理」

27.4.7.8 ファイングレイン監査ポリシーの削除

DBMS_FGA.DROP_POLICYプロシージャで、ファイングレイン監査ポリシーを削除します。

DBMS_FGA.ADD_POLICYプロシージャのobject_nameパラメータで指定されたオブジェクトを削除したり、監査ポリシーを作成したユーザーを削除した場合、自動的に監査ポリシーが削除されます。

- 次の構文を使用して、ファイングレイン監査ポリシーを削除します。

```
DBMS_FGA.DROP_POLICY(
   object_schema   VARCHAR2,
   object_name     VARCHAR2,
   policy_name     IVARCHAR2);
```

たとえば、DBMS_FGA.DROP_POLICYプロシージャを使用して、ファイングレイン監査ポリシーを手動で削除する方法を示します。

```
BEGIN
 DBMS_FGA.DROP_POLICY(
  object_schema       => 'HR',
  object_name         => 'EMPLOYEES',
  policy_name         => 'chk_hr_employees');
END;
/
```

関連項目:

DROP_POLICYの構文の詳細は、『Oracle Database PL/SQLパッケージおよびタイプ・リファレンス』を参照してください。

親トピック: 「DBMS_FGA PL/SQLパッケージを使用したファイングレイン監査ポリシーの管理」

27.4.8 例: ファイングレイン監査ポリシーへの電子メール・アラートの追加

このチュートリアルでは、ユーザーがポリシーに違反したときに電子メールのアラートを生成するファイングレイン監査ポリシーの作成方法を示します。

- このチュートリアルについて
  このチュートリアルでは、ユーザー(または侵入者)がポリシーに違反したときに実施される電子メールのアラートをファイングレイン監査ポリシーに追加する方法を示します。
- ステップ1: UTL_MAIL PL/SQLパッケージのインストールおよび構成
  UTL_MAIL PL/SQLで、添付、CCおよびBCCなど一般に使用される電子メール機能が組み込まれた電子メールを管理します。
- ステップ2: ユーザー・アカウントの作成
  管理アカウントおよび監査ユーザーを作成する必要があります。
- ステップ3: ネットワーク・サービスに対するアクセス制御リスト・ファイルの構成
  アクセス制御リスト(ACL)ファイルを使用して、外部ネットワーク・サービスへのファイングレイン・アクセスを有効にできます。
- ステップ4: 電子メール・セキュリティ・アラートPL/SQLプロシージャの作成
  電子メール・セキュリティ・アラートPL/SQLプロシージャは、違反について説明するメッセージを生成し、このメッセージを適切なユーザーに送信します。
- ステップ5: ファイングレイン監査ポリシー設定の作成とテスト
  ファイングレイン監査ポリシーは、ポリシーの違反があるとアラートをトリガーします。
- ステップ6: アラートのテスト
  コンポーネントの準備ができたら、アラートをテストします。
- ステップ7: このチュートリアルのコンポーネントの削除
  このチュートリアルのコンポーネントが不要になった場合、それらを削除できます。

親トピック: ファイングレイン監査を使用した特定のアクティビティの監査

27.4.8.1 このチュートリアルについて

このチュートリアルでは、ユーザー(または侵入者)がポリシーに違反したときに実施される電子メールのアラートをファイングレイン監査ポリシーに追加する方法を示します。

ノート:

- このチュートリアルを完了するには、SMTP サーバーのあるデータベースを使用する必要があります。
- マルチテナント環境を使用している場合、このチュートリアルは現在の PDB のみに適用されます。

ファイングレイン監査ポリシーに電子メール・アラートを追加するには、最初にアラートを生成するプロシージャを作成し、次のDBMS_FGA.ADD_POLICYパラメータを使用して、ユーザーがこのポリシーに違反した場合にこのファンクションをコールする必

要があります。

- handler_schema: ハンドラ・イベントが格納されるスキーマ
- handler_module: イベント・ハンドラの名前

アラートは、電子メールまたはポケベルによる通知や、特定のファイルまたは表の更新など、環境に適した形式で生成できます。アラートを作成すると、California Senate Bill 1386などの特定のコンプライアンス規制を満たすことも可能です。この例では、電子メール・アラートを作成します。

この例では、セキュリティ管理者に対して、人事部門の担当者がHR.EMPLOYEES表内の給与情報を選択または変更しようとしていることを通知する電子メール・アラートを作成します。担当者はこの表を変更することを許可されていますが、コンプライアンス規制を満たすために、表内の給与情報に対するすべての選択および変更操作に関するレコードを作成できます。

親トピック: 例: ファイングレイン監査ポリシーへの電子メール・アラートの追加

27.4.8.2 ステップ1: UTL_MAIL PL/SQLパッケージのインストールおよび構成

UTL_MAIL PL/SQLで、添付、CCおよびBCCなど一般に使用される電子メール機能が組み込まれた電子メールを管理します。

このパッケージを使用するには、インストールして構成する必要があります。このコンポーネントは、デフォルトではインストールおよび構成されません。

1. SYSDBA管理権限を持つユーザーSYSとしてログインします。

   ```
   sqlplus sys as sysdba
   Enter password: password
   ```

2. マルチテナント環境で、適切なPDBに接続します。

   たとえば:

   ```
   CONNECT SYS@hrpdb AS SYSDBA
   Enter password: password
   ```

   使用可能なPDBを検索するには、show pdbsコマンドを実行します。現在のPDBを確認するには、show con_nameコマンドを実行します。

3. UTL_MAILパッケージをインストールします。

   ```
   @$ORACLE_HOME/rdbms/admin/utlmail.sql
   @$ORACLE_HOME/rdbms/admin/prvtmail.plb
   ```

   UTL_MAILパッケージにより、電子メールの管理が可能になります。

   現在、UTL_MAIL PL/SQLパッケージではSSLサーバーはサポートされていないことに注意してください。

4. この例が終了した後に元に戻すことができるよう、SMTP_OUT_SERVER初期化パラメータの現行の値を調べてノートにとっておきます。

   たとえば:

   ```
   SHOW PARAMETER SMTP_OUT_SERVER
   ```

   SMTP_OUT_SERVERパラメータがすでに設定されている場合は、次のような出力が表示されます。

   ```
   NAME                 TYPE             VALUE
   -------------------- ---------------- --------------------------------
   SMTP_OUT_SERVER      string           some_imap_server.example.com
   ```

5. 次のALTER SYSTEM文を発行します。

```
ALTER SYSTEM SET SMTP_OUT_SERVER="imap_mail_server.example.com";
```

imap_mail_server.example.comを、電子メール・ツールのアカウント設定にあるSMTPサーバーの名前に置き換えます。これらの設定を引用符で囲んでください。たとえば:

```
ALTER SYSTEM SET SMTP_OUT_SERVER="my_imap_server.example.com";
```

6. SYSOPER権限を使用してSYSとして接続し、データベースを再起動します。

```
CONNECT SYS AS SYSOPER -- Or, CONNECT SYS@hrpdb AS SYSOPER
Enter password: password
SHUTDOWN IMMEDIATE
STARTUP
```

7. SMTP_OUT_SERVERパラメータの設定が正しいことを確認します。

```
CONNECT SYS AS SYSDBA -- Or, CONNECT SYS@hrpdb AS SYSDBA
Enter password: password
SHOW PARAMETER SMTP_OUT_SERVER
```

次のような出力が表示されます。

```
NAME                 TYPE             VALUE
-------------------- ---------------- --------------------------------
SMTP_OUT_SERVER      string           my_imap_server.example.com
```

関連項目:

UTL_MAILパッケージの詳細は、『Oracle Database PL/SQLパッケージおよびタイプ・リファレンス』を参照してください。

親トピック: 例: ファイングレイン監査ポリシーへの電子メール・アラートの追加

27.4.8.3 ステップ2: ユーザー・アカウントの作成

管理アカウントおよび監査ユーザーを作成する必要があります。

1. SYSとしてSYSDBA管理権限で接続していることを確認し、fga_adminユーザー(ファイングレイン監査ポリシーの作成者)を作成します。

たとえば:

```
CONNECT SYS AS SYSDBA -- Or, CONNECT SYS@hrpdb AS SYSDBA
Enter password: password
CREATE USER fga_admin IDENTIFIED BY password;
GRANT CREATE SESSION, CREATE PROCEDURE, AUDIT_ADMIN TO fga_admin;
GRANT EXECUTE ON UTL_TCP TO fga_admin;
GRANT EXECUTE ON UTL_SMTP TO fga_admin;
GRANT EXECUTE ON UTL_MAIL TO fga_admin;
GRANT EXECUTE ON DBMS_NETWORK_ACL_ADMIN TO fga_admin;
```

「パスワードの最低要件」のガイドラインに従って、passwordを安全なパスワードに置き換えます。

UTL_TCP、UTL_SMTP、UTL_MAILおよびDBMS_NETWORK_ACL_ADMIN PL/SQLパッケージは、作成する電子メール・セキュリティ・アラートで使用されます。

2. このポリシーの監査証跡をチェックする監査者ユーザーを作成します。

```
GRANT CREATE SESSION TO fga_auditor IDENTIFIED BY password;
```

```
GRANT AUDIT_VIEWER TO fga_auditor;
```

3. ユーザーSYSTEMで接続します。

```
CONNECT SYSTEM -- Or, CONNECT SYSTEM@hrpdb
Enter password: password
```

4. HRスキーマ・アカウントのロックが解除され、パスワードが付与されていることを確認します。必要に応じて、HRのロックを解除し、このユーザーにパスワードを付与します。

```
SELECT USERNAME, ACCOUNT_STATUS FROM DBA_USERS WHERE USERNAME = 'HR';
```

アカウント・ステータスはOPENである必要があります。DBA_USERSビューに、ユーザーHRがロックされて期限切れになっていると表示された場合は、次の文を入力して、HRアカウントのロックを解除し、新しいパスワードを作成します。

```
ALTER USER HR ACCOUNT UNLOCK IDENTIFIED BY password;
```

「パスワードの最低要件」のガイドラインに従って、安全なパスワードを作成します。セキュリティを向上させるため、以前のリリースのOracle Databaseと同じパスワードをHRアカウントに指定しないでください。

5. 人事部門の担当者であるSusan Mavris(アクションの監査対象)用のユーザー・アカウントを作成し、このユーザーにHR.EMPLOYEES表へのアクセス権を付与します。

```
GRANT CREATE SESSION TO smavris IDENTIFIED BY password;
GRANT SELECT, INSERT, UPDATE, DELETE ON HR.EMPLOYEES TO SMAVRIS;
```

親トピック: 例: ファイングレイン監査ポリシーへの電子メール・アラートの追加

27.4.8.4 ステップ3: ネットワーク・サービス用のアクセス制御リストの構成

アクセス制御リスト(ACL)ファイルを使用して、外部ネットワーク・サービスへのファイングレイン・アクセスを有効にできます。

UTL_MAILなどのPL/SQLネットワーク・ユーティリティ・パッケージを使用するには、このタイプのアクセス制御リスト(ACL)ファイルを構成する必要があります。

1. ユーザーfga_adminでSQL*Plusに接続します。

```
CONNECT fga_admin -- Or, CONNECT fga_admin@hrpdb
Enter password: password
```

2. 次のアクセス制御設定とその権限定義を構成します。

```
BEGIN
 DBMS_NETWORK_ACL_ADMIN.APPEND_HOST_ACE(
  host       => 'SMTP_OUT_SERVER_setting',
  lower_port => 25,
  ace        =>  xs$ace_type(privilege_list => xs$name_list('smtp'),
                             principal_name => 'FGA_ADMIN',
                             principal_type => xs_acl.ptype_db));
END;
/
```

この例では、次のようになります。

- SMTP_OUT_SERVER_setting: 「ステップ1: UTL_MAIL PL/SQLパッケージのインストールおよび構成」でSMTP_OUT_SERVERパラメータに設定したSMTP_OUT_SERVER設定を入力します。この設定は、電子メール・ツールで送信サーバーに指定されている設定と完全に一致させてください。
- lower_port: 電子メール・ツールで送信サーバーに指定されているポート番号を入力します。通常、この

設定は25です。この値をlower_port設定に入力します。(現在、UTL_MAILパッケージではSSLをサポートしていません。電子メール・サーバーがSSLサーバーの場合、その電子メール・サーバーが別のポート番号を使用していても、ポート番号に25を入力します。)

- ace: ここで権限を定義します。

関連項目:

アクセス制御リスト(ACL)ファイルの構成の詳細は、「PL/SQLパッケージおよびタイプでのファイングレイン・アクセスの管理」を参照してください。

親トピック: 例: ファイングレイン監査ポリシーへの電子メール・アラートの追加

27.4.8.5 ステップ4: 電子メール・セキュリティ・アラートPL/SQLプロシージャの作成

電子メール・セキュリティ・アラートPL/SQLプロシージャは、違反について説明するメッセージを生成し、このメッセージを適切なユーザーに送信します。

- ユーザーfga_adminで、次のプロシージャを作成します。

```
CREATE OR REPLACE PROCEDURE email_alert (sch varchar2, tab varchar2, pol
varchar2)
AS
msg varchar2(20000) := 'HR.EMPLOYEES table violation. The time is: ';
BEGIN
  msg := msg||TO_CHAR(SYSDATE, 'Day DD MON, YYYY HH24:MI:SS');
UTL_MAIL.SEND (
    sender      => 'youremail@example.com',
    recipients  => 'recipientemail@example.com',
    subject     => 'Table modification on HR.EMPLOYEES',
    message     => msg);
END email_alert;
/
```

この例では、次のようになります。

- CREATE OR REPLACE PROCEDURE ...AS: 次のステップで監査ポリシーに定義するスキーマ名(sch)、表名(tab)および監査プロシージャ名(pol)を表す署名を指定する必要があります。
- senderおよびrecipients: youremail@example.comを自分の電子メール・アドレス、recipientemail@example.comを通知の受信対象者の電子メール・アドレスに置き換えます。

親トピック: 例: ファイングレイン監査ポリシーへの電子メール・アラートの追加

27.4.8.6 ステップ5: ファイングレイン監査ポリシー設定の作成とテスト

ファイングレイン監査ポリシーは、ポリシーの違反があるとアラートをトリガーします。

1. ユーザーfga_adminで、chk_hr_empポリシーをファイングレイン監査ポリシーとして次のように作成します。

```
BEGIN
 DBMS_FGA.ADD_POLICY (
  object_schema      =>  'HR',
  object_name        =>  'EMPLOYEES',
  policy_name        =>  'CHK_HR_EMP',
  audit_column       =>  'SALARY',
  handler_schema     =>  'FGA_ADMIN',
  handler_module     =>  'EMAIL_ALERT',
  enable             =>   TRUE,
```

```
  statement_types    =>  'SELECT, UPDATE');
END;
/
```

2. データベースに加えた変更をコミットします。

```
COMMIT;
```

3. これまでに作成した設定をテストします。

```
EXEC email_alert ('hr', 'employees', 'chk_hr_emp');
```

SQL*Plusに「PL/SQL procedure successfully completed」というメッセージが表示されます。まもなく、電子メール・サーバーの速度に応じて、電子メール・アラートを受信します。

ORA-24247「アクセス制御リスト(ACL)によりネットワーク・アクセスが拒否されました」エラーの後にORA-06512「string行string」エラーが発生した場合は、アクセス制御リスト・ファイル内の設定を確認してください。

親トピック: 例: ファイングレイン監査ポリシーへの電子メール・アラートの追加

27.4.8.7 ステップ6: アラートのテスト

コンポーネントの準備ができたら、アラートをテストします。

1. ユーザーsmavrisでSQL*Plusに接続し、自分の給与を確認して金額を引き上げます。

```
CONNECT smavris -- Or, CONNECT smavris@hrpdb
Enter password: password
SELECT SALARY FROM HR.EMPLOYEES WHERE LAST_NAME = 'Mavris';
SALARY
-----------
6500
UPDATE HR.EMPLOYEES SET SALARY = 38000 WHERE LAST_NAME = 'Mavris';
```

ここまでの手順を実行すると、電子メール・サーバーの速度に応じて、自分自身(または通知の受信対象者)にTable modification on HR.EMPLOYEESという件名ヘッダーの電子メールが届き、HR.EMPLOYEES表の改ざんについて通知されます。違反者を検索するのに必要なのは、UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューを問い合せるのみです。

2. 次のようにユーザーfga_auditorとして、UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューを問い合せます。

```
CONNECT fga_auditor -- Or, CONNECT fga_auditor@hrpdb
Enter password: password
col dbusername format a20
col sql_text format a66
col audit_type format a17
SELECT DBUSERNAME, SQL_TEXT, AUDIT_TYPE
FROM UNIFIED_AUDIT_TRAIL
WHERE OBJECT_SCHEMA = 'HR' AND OBJECT_NAME = 'EMPLOYEES';
```

次のような出力が表示されます。

```
DBUSERNAME  SQL_TEXT
AUDIT_TYPE
----------  ------------------------------------------------------------------ -
---------------
SMAVRIS     UPDATE HR.EMPLOYEES SET SALARY = 38000 WHERE LAST_NAME = 'Mavris'
FineGrainedAudit
```

監査証跡では、Susan Mavrisが実行し、HR.EMPLOYEES表のSALARY列に影響を与えたSQL文が取得されます。彼女が実行した最初の文(現在の給与に関して尋ねた文)は、監査ポリシーの影響を受けなかったため、記録されませんでした。これは、Oracle Databaseでは、監査関数は自律型トランザクションとして実行され、handler_module設定のアクションのみをコミットし、ユーザー・トランザクションはコミットしないためです。この関数はユーザーのSQLトランザクションには影響を与えません。

親トピック: 例: ファイングレイン監査ポリシーへの電子メール・アラートの追加

27.4.8.8 ステップ7: このチュートリアルのコンポーネントの削除

このチュートリアルのコンポーネントが不要になった場合、それらを削除できます。

1. SYSTEM権限を持つユーザーでSQL*Plusに接続し、ユーザーfga_admin (fga_adminスキーマ内のオブジェクトを含む)、fga_auditorおよびsmavrisを削除します。

```
CONNECT SYSTEM -- Or, CONNECT SYSTEM@hrpdb
Enter password: password
DROP USER fga_admin CASCADE;
DROP USER fga_auditor;
DROP USER smavris;
```

2. ユーザーHRで接続し、Susan Mavrisの引き上げた給与を元に戻します。

```
CONNECT HR -- Or, CONNECT HR@hrpdb
Enter password: password
UPDATE HR.EMPLOYEES SET SALARY = 6500 WHERE LAST_NAME = 'Mavris';
```

3. 他のユーザーがHRを使用しない場合、このアカウントはロックして期限切れにできます。

```
ALTER USER HR PASSWORD EXPIRE ACCOUNT LOCK;
```

4. 次のALTER SYSTEM文を発行して、ステップ1: UTL_MAIL PL/SQLパッケージのインストールおよび構成のステップ5からSMTP_OUT_SERVERパラメータを前の値にリストアします。

```
ALTER SYSTEM SET SMTP_OUT_SERVER="previous_value";
```

この設定は引用符で囲みます。たとえば:

```
ALTER SYSTEM SET SMTP_OUT_SERVER="some_imap_server.example.com"
```

5. データベース・インスタンスを再起動します。

親トピック: 例: ファイングレイン監査ポリシーへの電子メール・アラートの追加

27.5 監査ポリシーのデータ・ディクショナリ・ビュー

データ・ディクショナリ・ビューおよび動的ビューを使用して詳細な監査情報を入手できます。

表27-20に、これらのビューを示します。

ヒント:

監査ポリシーに関するエラー情報を検索するには、トレース・ファイルを確認します。USER_DUMP_DEST 初期化パラメータは、トレース・ファイルの位置を示します。

表27-20 監査アクティビティに関する情報を表示するビュー

ビュー	説明
ALL_AUDIT_POLICIES	すべてのファイングレイン監査ポリシーに関する情報が表示されます。
ALL_DEF_AUDIT_OPTS	オブジェクトの作成時に適用されるデフォルトのオブジェクト監査オプションがリストされます。
AUDIT_UNIFIED_CONTEXTS	監査証跡で取得されるように構成されているアプリケーション・コンテキスト値が表示されます。
AUDIT_UNIFIED_ENABLED_POLICIES	データベースで有効なすべての統合監査ポリシーが表示されます。
AUDIT_UNIFIED_POLICIES	データベースで作成されたすべての統合監査ポリシーが表示されます。
AUDIT_UNIFIED_POLICY_COMMENTS	COMMENT SQL 文を使用して統合監査ポリシーに説明が入力された場合に、各統合監査ポリシーの説明が表示されます。
AUDITABLE_SYSTEM_ACTIONS	監査可能なシステム・アクション番号がアクション名にマップされます。
CDB_UNIFIED_AUDIT_TRAIL	UNIFIED_AUDIT_TRAIL ビューと同様、監査レコードを表示しますが、マルチテナント環境のすべての PDB からの監査レコードを表示します。このビューは、CDB ルートのみで使用可能で、そこから問い合せる必要があります。
DBA_AUDIT_POLICIES	ファイングレイン監査ポリシーに関する情報が表示されます。
DBA_SA_AUDIT_OPTIONS	ユーザーによって実行される監査対象の Oracle Label Security イベントが表示され、ユーザーのアクションが成功したか失敗したかが示されます。
DBA_XS_AUDIT_TRAIL	Oracle Database Real Application Security に関連する監査証跡の情報が表示されます。
DV$CONFIGURATION_AUDIT	Oracle Database Vault 管理者による構成変更が表示されます。
DV$ENFORCEMENT_AUDIT	Oracle Database Vault ポリシーによって影響を受けるユーザー・アクティビティが表示されます。
SYSTEM_PRIVILEGE_MAP (表)	権限(監査オプション)型コードが表示されます。この表を使用して、権限(監査オプション)の型番号を型名にマップできます。

ビュー	説明
USER_AUDIT_POLICIES	現行ユーザーによって所有される表およびビューのすべてのファイングレイン監査ポリシーに関する情報が表示されます。
UNIFIED_AUDIT_TRAIL	すべての監査レコードが表示されます。
V$OPTION	統合監査の PARAMETER 列を問い合せて、統合監査が有効かどうかを確認できます。
V$XML_AUDIT_TRAIL	XML 形式のファイルに書き込まれた標準監査、ファイングレイン監査、SYS 監査および必須監査のレコードが表示されます。

関連トピック

- Oracle Databaseリファレンス

親トピック: 監査ポリシーの構成

28 監査証跡の管理

AUDIT_ADMINロールを付与されているユーザーは、監査証跡の管理、監査証跡のアーカイブおよび監査証跡レコードの削除を実行できます。

- 統合監査証跡の管理
  監査はデフォルトで有効ですが、監査レコードがディスクに書き込まれるタイミングを制御できます。
- 監査証跡のアーカイブ
  従来のオペレーティング・システム、統合データベースおよび従来のデータベースの監査証跡をアーカイブできます。
- 監査証跡レコードの削除
  DBMS_AUDIT_MGMT PL/SQLパッケージを使用して、自動削除ジョブをスケジューリングして、監査レコードを手動で削除し、他の監査証跡操作を実行できます。
- 監査証跡管理のデータ・ディクショナリ・ビュー
  Oracle Databaseには、監査証跡の管理設定に関する情報を表示するデータ・ディクショナリ・ビューが用意されています。

親トピック: 監査を使用したデータベース・アクティビティの監視

28.1 統合監査証跡の管理

監査はデフォルトで有効ですが、監査レコードがディスクに書き込まれるタイミングを制御できます。

- 監査レコードが作成されるときと場所
  監査は常に有効になっています。Oracle Databaseは、監査対象のSQL文の実行フェーズ中または実行フェーズ後に監査レコードを生成します。
- 強制的に監査されるアクティビティ
  セキュリティ・センシティブな特定のデータベース・アクティビティは常に監査され、このような監査構成は無効化できません。
- カーソルが監査に与える影響
  カーソル内で監査対象の操作が実行されるたびに、Oracle Databaseでは監査証跡に監査レコードが1つ挿入されます。
- AUDSYSスキーマへの統合監査証跡レコードの書込み
  Oracle Databaseは、監査レコードをAUDSYSスキーマの内部リレーショナル表に自動的に書き込みます。
- SYSLOGまたはWindowsイベントビューアへの統合監査証跡レコードの書込み
  初期化パラメータを設定することによって、SYSLOGまたはWindowsイベントビューアに統合監査証跡レコードを書き込むことができます。
- 監査レコードがオペレーティング・システムに書き込まれる場合
  データベース表が統合監査レコードを受け入れられない場合、これらのレコードはオペレーティング・システムの過剰監査ファイル(.bin形式)に書き込まれます。
- 統合監査証跡へのオペレーティング・システムの監査レコードの移動
  スピルオーバー監査ファイルに書き込まれた監査レコードは、統合監査証跡データベース表に移動できます。
- Oracle Data Pumpを使用した統合監査証跡のエクスポートおよびインポート
  Oracle Database Pumpのエクスポートおよびインポート用ダンプ・ファイルに統合監査証跡を含めることができます。
- 統合監査の無効化
  統合監査を無効にすることができます。

関連トピック

- 監査証跡レコードの削除

親トピック: 監査証跡の管理

28.1.1 監査レコードが作成されるときと場所

監査は常に有効になっています。Oracle Databaseは、監査対象のSQL文の実行フェーズ中または実行フェーズ後に監査レコードを生成します。

PL/SQLプログラム・ユニット内のSQL文は、プログラム・ユニットの実行時に必要に応じて個別に監査されます。

統合監査証跡の読取りパフォーマンスを高めるために、統合監査レコードはディスクのAUDSYSスキーマの内部リレーショナル表に即座に書き込まれます。以前のリリースでは、統合監査レコードはSecureFile LOBに書き込まれていました。Oracle Database 12cリリース1 (12.1)の統合監査に移行済の場合は、SecureFile LOBからこの内部表に統合監査レコードを手動で転送できます。使用中のデータベースのバージョンがパーティション化された表をサポートしている場合、この内部表はパーティション化された表です。この場合は、DBMS_AUDIT_MGMT.ALTER_PARTITION_INTERVALプロシージャを使用して表のパーティション間隔を変更できます。この表のパーティション化されたバージョンは、デフォルトのパーティション間隔が1か月のパーティション・キーとしてEVENT_TIMESTAMPタイムスタンプに基づいています。データベース・バージョンがパーティショニングをサポートしていない場合、内部表は標準のもので、パーティション化されていない表です。

監査証跡レコードの生成と挿入は、ユーザー・トランザクションのコミットからは独立して実行されます。つまり、ユーザー・トランザクションがロールバックされても、監査証跡レコードはコミットされたままになります。

データベース・ユーザーがデータベースに接続した時点で有効になる統合監査ポリシーの文監査オプションと権限監査オプションは、そのセッションの持続期間中は有効です。セッションがすでにアクティブになっている場合、文または権限の統合監査のオプションを設定または変更しても、そのセッション中は有効になりません。修正した文監査オプションまたは権限監査オプションは、カレント・セッションを終了し、新しいセッションを作成した時点で有効になります。

一方、オブジェクト監査オプションについて変更した内容は、カレント・セッションでただちに有効になります。

デフォルトでは、監査証跡レコードはSYSAUX表領域のAUDSYSスキーマに書き込まれます。DBMS_AUDIT_MGMT.SET_AUDIT_TRAIL_LOCATIONプロシージャを使用して、暗号化されている表領域を含め、異なる表領域を指定できます。

関連トピック

- AUDSYSスキーマへの統合監査証跡レコードの書込み
- Oracle Databaseアップグレード・ガイド
- Oracle Database PL/SQLパッケージ・プロシージャおよびタイプ・リファレンス

親トピック: 統合監査証跡の管理

28.1.2 強制的に監査されるアクティビティ

セキュリティ・センシティブな特定のデータベース・アクティビティは常に監査され、このような監査構成は無効化できません。

UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューでは、SYSDBA、SYSBACKUP、SYSKMなどの管理ユーザーからのアクティビティを取得します。統合監査証跡を監査する必要はありません。統合監査証跡は、AUDSYSスキーマ内の読取り専用表にあります。したがって、統合監査証跡ビューに対するDMLは許可されません。AUDSYSスキーマからの基礎となるディクショナリ表に対するDMLおよびDDL操作も許可されません。

SYSTEM_PRIVILEGE_USED列は、アクティビティに使用された管理権限のタイプを示します。

監査ポリシーの変更など、次の監査関連のアクティビティは強制的に監査されます。

- CREATE AUDIT POLICY
- ALTER AUDIT POLICY
- DROP AUDIT POLICY
- AUDIT
- NOAUDIT
- DBMS_FGA PL/SQLパッケージのEXECUTE
- DBMS_AUDIT_MGMT PL/SQLパッケージのEXECUTE
- AUDSYS監査証跡表でのALTER TABLEの試行(この表は監査できません)
- 管理ユーザーSYS、SYSDBA、SYSOPER、SYSASM、SYSBACKUP、SYSDGおよびSYSKMによるトップ・レベルの文(データベースがオープンするまで)。データベースがオープンされると、Oracle Databaseはシステムの監査構成(AUDIT文のBY句などを使用して適用された構成だけでなく、AUDIT文にBY句がない場合や、EXCEPT句が使用され、これらのユーザーが除外されなかった場合に、すべてのユーザーに対して適用された構成)を使用して、これらのユーザーを監査します。
- SYS.AUD$およびSYS.FGA_LOG$ディクショナリ表に対してユーザーが発行したすべてのDML文
- 統合監査内部表のデータまたはメタデータの変更。この表に対するSELECT文はデフォルトで、または強制的に監査されます。
- Oracle Database Vaultに対して行われるすべての構成変更

. オプティマイザ・ディクショナリ表内の機密列へのアクセス。DBMS_STATSパッケージによるこれらの表列への内部アクセスでは、必須の監査レコードは生成されないということに注意してください。オプティマイザ・ディクショナリ表を次に示します。

オプティマイザ・ディクショナリ表	列
SYS.HIST_HEAD$	minimum、maximum、lowval、hival
SYS.HISTGRM$	endpoint、epvalue_raw
SYS.WRI$_OPTSTAT_HISTHEAD_HISTORY	minimum、maximum、lowval、hival
SYS.WRI$_OPSTAT_HISTGRM_HISTORY	endpoint、epvalue_raw

関連トピック

- 管理ユーザーの監査

親トピック: 統合監査証跡の管理

28.1.3 カーソルが監査に与える影響

カーソル内で監査対象の操作が実行されるたびに、Oracle Databaseでは監査証跡に監査レコードが1つ挿入されます。

カーソルを再利用させるイベントは、次のとおりです。

- カーソルを再利用のためにオープン状態にしておく、Oracle Formsなどのアプリケーション

- 新しいバインド変数を使用したカーソルの継続実行
- 1つのカーソルを再利用するためにPL/SQLエンジンにより文が最適化される場合に、PL/SQLループ内で実行される文

監査は、カーソルが共有されているかどうかの影響を受けません。それぞれのユーザーは、カーソルの最初の実行時に自分の監査証跡レコードを作成します。

親トピック: 統合監査証跡の管理

28.1.4 AUDSYSスキーマへの統合監査証跡レコードの書込み

Oracle Databaseは、監査レコードをAUDSYSスキーマの内部リレーショナル表に自動的に書き込みます。

Oracle Database 12cリリース1 (12.1)では、監査レコードをメモリー(キュー書込みモード)にキューイングし、AUDSYSスキーマ監査表に定期的に書き込むオプションがありました。ただし、Oracle Database 12cリリース2 (12.2)以降では、即時書込みモードおよびキュー書込みモードは非推奨です。制御するパラメータ(`UNIFIED_AUDIT_SGA_QUEUE_SIZE`、`DBMS_AUDIT_MGMT.AUDIT_TRAIL_IMMEDIATE_WRITE`および`DBMS_AUDIT_MGMT.AUDIT_TRAIL_QUEUED_WRITE`)は、まだ表示可能ですが、機能はありません。

監査レコードがAUDSYSスキーマのリレーショナル表に常に書き込まれる新しい機能によって、インスタンスがクラッシュした場合や`SHUTDOWN ABORT`操作時に監査レコードが失われるリスクが回避されます。新しい機能によって、監査証跡とデータベース全体のパフォーマンスも向上します。

Oracle Database 12cリリース1 (12.1)からアップグレードし、そのリリースの統合監査に移行した場合は、`DBMS_AUDIT_MGMT.TRANSFER_UNIFIED_AUDIT_RECORDS`プロシージャを使用して、以前のリリースで生成された監査レコードをAUDSYS監査内部表に転送することをお薦めします。アップグレード後の統合監査レコードの転送の詳細は、Oracle Databaseアップグレード・ガイドを参照してください。

関連トピック

- Oracle Databaseアップグレード・ガイド

親トピック: 統合監査証跡の管理

28.1.5 SYSLOGまたはWindowsイベントビューアへの統合監査証跡レコードの書込み

初期化パラメータを設定することによって、SYSLOGまたはWindowsイベントビューアに統合監査証跡レコードを書き込むことができます。

- SYSLOGまたはWindowsイベントビューアへの統合監査証跡レコードの書込みについて
  この機能を使用して、一部の主要な統合監査フィールドをSYSLOGまたはWindowsイベントビューアにコピーできます。
- SYSLOGおよびWindowsイベントビューアでの統合監査証跡の取得の有効化
  統合監査証跡レコードのサブセットをUNIX SYSLOGまたはWindowsイベントビューアに書き込むことができます。

親トピック: 統合監査証跡の管理

28.1.5.1 SYSLOGまたはWindowsイベントビューアへの統合監査証跡レコードの書込みについて

この機能を使用すると、一部の主要な統合監査フィールドをSYSLOGまたはWindowsイベントビューアにコピーできます。

従来の監査とは異なり、`UNIFIED_AUDIT_TRAIL`データ・ディクショナリ・ビューの統合監査レコードのキー・フィールドのみが

SYSLOGにコピーされます。統合監査環境のSYSLOGレコードによって、操作の整合性が証明されます。

この機能は、UNIXシステムおよびMicrosoft Windowsシステムの両方で構成できます。Windowsシステムでは、これを有効または無効のいずれかにします。有効な場合は、レコードがWindowsイベントビューアに書き込まれます。

UNIXシステムでは、SYSLOGに対する統合監査証跡レコードの取得を微調整して、SYSLOGレコードが送信される機能を指定したり、レコードの重大度レベル(たとえば、デバッグ関連メッセージを取得する場合はDEBUG)を指定したりできます。

表28-1は、SYSLOGおよびWindowsイベントビューアに書き込まれる統合監査レコード・フィールドに指定される名前を、UNIFIED_AUDIT_TRAILビューの対応する列名にマップしたものです。

表28-1 SYSLOGおよびWindowsイベントビューアの監査レコードのフィールド名

フィールド名	UNIFIED_AUDIT_TRAILの列名	列タイプ	列の説明
TYPE	AUDIT_TYPE	NUMBER	監査レコードのタイプ
DBID	DBID	NUMBER	データベース識別子
SESID	SESSION_ID	NUMBER	セッション識別子
CLIENTID	CLIENT_IDENTIFIER	VARCHAR2	セッションでのクライアント識別子
ENTRYID	ENTRY_ID	NUMBER	システムの各監査レコードの識別子
STMTID	STATEMENT_ID	NUMBER	システムで実行されている各文の識別子
DBUSER	DB_USERNAME	VARCHAR2	セッション・ユーザー
CURUSER	CURRENT_USER	VARCHAR2	監査対象イベントの有効なユーザー
ACTION	ACTION	NUMBER	監査対象イベントのアクション・コード
RETCODE	RETURN_CODE	NUMBER	監査対象イベントのリターン・コード
SCHEMA	OBJECT_SCHEMA	VARCHAR2	オブジェクトのスキーマ名。
OBJNAME	OBJECT_NAME	VARCHAR2	オブジェクト名
PDB_GUID	NULL (このフィールドのUNIFIED_AUDIT_TRAIL に列はありません)	VARCHAR2	統合監査レコードが生成されるコンテナの GUID

親トピック: SYSLOGまたはWindowsイベントビューアへの統合監査証跡レコードの書き込み

28.1.5.2 SYSLOGおよびWindowsイベントビューアでの統合監査証跡の取得の有効化

統合監査証跡レコードのサブセットをUNIX SYSLOGまたはWindowsイベントビューアに書き込むことができます。

1. init.ora初期化ファイルを探します。これはデフォルトで$ORACLE_HOME/dbsディレクトリにあります。
2. init.oraファイルを編集して、UNIFIED_AUDIT_SYSTEMLOGパラメータを含めます。
   CDBルートまたはPDBのどちらかでUNIFIED_AUDIT_SYSTEMLOGを設定できます。
   Oracle Real Application Clusters (Oracle RAC)環境では、各Oracle RACインスタンス上のUNIFIED_AUDIT_SYSTEMLOGを同じ値に設定します。
   - Windowsでは、UNIFIED_AUDIT_SYSTEMLOGをTRUEまたはFALSEに設定します。TRUEの場合はSYSLOG値をWindowsイベントビューアに書き込み、FALSEの場合はパラメータを無効にします。
     Windowsの場合、デフォルトはFALSEです。たとえば:

     ```
     UNIFIED_AUDIT_SYSTEMLOG = TRUE
     ```

   - UNIXシステムの場合、次の構文を使用します。

     ```
     UNIFIED_AUDIT_SYSTEMLOG = 'facility_clause.priority_clause'
     ```

     UNIXシステムでは、UNIFIED_AUDIT_SYSTEMLOGのデフォルト設定はありません。

     詳細は、次のとおりです。

     - facility_clauseは、監査証跡レコードを書き込む機能を指定します。有効な選択肢はUSERおよびLOCALです。LOCALと入力した場合、必要に応じて0-7を追加し、SYSLOGレコードのローカル・カスタム機能を指定します。
     - priority_clauseは、レコードを分類する警告のタイプを指定します。有効な選択肢は、NOTICE、INFO、DEBUG、WARNING、ERR、CRIT、ALERT、およびEMERGです。

   たとえば:

   ```
   UNIFIED_AUDIT_SYSTEMLOG = 'LOCAL7.EMERG'
   ```

3. UNIXプラットフォームで統合監査レコードをSYSLOGに書き込むには、ルート内のinit.oraファイルにおいて、UNIFIED_AUDIT_COMMON_SYSTEMLOGパラメータをTRUEまたはFALSEに設定します。

   UNIFIED_AUDIT_COMMON_SYSTEMLOGをTRUEに設定すると、共通統合監査ポリシーからSYSLOGに統合監査レコードの事前定義済の列が書き込まれます。FALSEに設定すると、これらの列のSYSLOGへの書込みを無効にします。

   このパラメータは、プラガブル・データベース(PDB)では設定できません。Windowsでは UNIFIED_AUDIT_COMMON_SYSTEMLOGに相当するパラメータはありません。

4. 監査ファイルの宛先をSYSLOG構成ファイル/etc/syslog.confに追加します。
   たとえば、UNIFIED_AUDIT_SYSTEMLOGをLOCAL7.EMERGに設定したとすると、次のように入力します。

   ```
   local7.emerg /var/log/audit.log
   ```

   この設定では、すべての緊急メッセージが/var/log/audit.logファイルに記録されます。

5. SYSLOGログ出力を再起動します。

   ```
   $/etc/rc.d/init.d/syslog restart
   ```

   これで、すべての監査レコードがsyslogデーモンを介してファイル/var/log/audit.logに取得されます。

6. データベース・インスタンスに再びログインします。
7. データベースを再起動します。

   たとえば:

```
SHUTDOWN IMMEDIATE
STARTUP
```

   PDBでUNIFIED_AUDIT_SYSTEMLOGを設定した場合は、PDBをクローズして再オープンします。

```
ALTER PLUGGABLE DATABASE pdb_name CLOSE IMMEDIATE;
ALTER PLUGGABLE DATABASE pdb_name OPEN;
```

関連トピック

- Oracle Databaseリファレンス

親トピック: SYSLOGまたはWindowsイベントビューアへの統合監査証跡レコードの書き込み

28.1.6 監査レコードがオペレーティング・システムに書き込まれる場合

データベース表が統合監査レコードを受け入れられない場合、これらのレコードはオペレーティング・システムの過剰監査ファイル(.bin形式)に書き込まれます。

統合監査の.bin過剰ファイルのデフォルトの場所は次のとおりです。

- プラガブル・データベース(PDB)の場合: $ORACLE_BASE/audit/$ORACLE_SID/PDB_GUID
- 非統合データベースの場合、またはCDBルートの場合: $ORACLE_BASE/audit/$ORACLE_SID/

データベース表への書込み機能は、監査表領域がオフライン、表領域が読取り専用、表領域がいっぱい、データベースが読取り専用などの場合、失敗する可能性があります。統合監査レコードは、OSディスク領域がいっぱいになるまで、OSの過剰ファイルに書き込まれます。この時点で、OSに監査レコードの空き領域がない場合、ユーザー監査可能なトランザクションは「ORA-02002　監査証跡への書込み中にエラーが発生しました。」エラーで失敗します。この問題を回避するために、監査証跡を定期的に削除することをお薦めします。

関連トピック

- 監査証跡レコードの削除

親トピック: 統合監査証跡の管理

28.1.7 統合監査証跡へのオペレーティング・システムの監査レコードの移動

スピルオーバー監査ファイルに書き込まれた監査レコードは、統合監査証跡データベース表に移動できます。

データベースが書込み可能でない場合(データベースのマウント中など)、データベースがクローズされている場合、または読取り専用の場合は、Oracle Databaseによって、監査レコードがこれらの外部ファイルに書き込まれます。これらの外部ファイルのデフォルトの場所は、$ORACLE_BASE/audit/$ORACLE_SIDディレクトリです。

DBMS_AUDIT_MGMT.LOAD_UNIFIED_AUDIT_FILESプロシージャを実行して、データベースにファイルをロードできます。多くのオペレーティング・システム監査レコードを外部ファイルに移動している場合は、パフォーマンスが影響を受ける場合があるので注意してください。

データベースが書込み可能な場合に、これらのファイルの監査レコードをAUDSYSスキーマ監査表に移動するには:

1. AUDIT_ADMINロールを割り当てられているユーザーとして、データベース・インスタンスにログインします。

たとえば:

```
CONNECT aud_admin
Enter password: password
Connected.
```

マルチテナント環境では、AUDIT_ADMINロールでCDBルートにログインします。現在のリリースまたはOracle Databaseにアップグレードする前に、アップグレード・プロセス中にオペレーティング・システムの過剰ファイルが失われないようにするために、CDBルートからDBMS_AUDIT_MGMT.LOAD_UNIFIED_AUDIT_FILESプロシージャを実行する必要があります。

たとえば:

```
CONNECT c##aud_admin
Enter password: password
Connected.
```

2. データベースがオープンで、書込み可能であることを確認します。

   非CDBアーキテクチャの場合、データベースが開いていて書込み可能かどうかを確認するには、V$DATABASEビューを問い合せます。

   たとえば、CDB環境では次のようになります。

```
SELECT NAME, OPEN_MODE FROM V$DATABASE;
NAME            OPEN_MODE
--------------- ----------
HRPDB           READ WRITE
```

   マルチテナント環境では、show pdbsコマンドを実行して、現在のインスタンスに関連付けられているPDBに関する情報を検索できます。

3. DBMS_AUDIT_MGMT.LOAD_UNIFIED_AUDIT_FILESプロシージャを実行します。

```
EXEC DBMS_AUDIT_MGMT.LOAD_UNIFIED_AUDIT_FILES;
```

4. マルチテナント環境で個々のPDB監査レコードをロードする場合は、各PDBにログインしてDBMS_AUDIT_MGMT.LOAD_UNIFIED_AUDIT_FILESプロシージャを再実行します。

監査レコードはAUDSYSスキーマ監査表に即座にロードされ、$ORACLE_BASE/audit/$ORACLE_SIDディレクトリから削除されます。

親トピック: 統合監査証跡の管理

28.1.8 Oracle Data Pumpを使用した統合監査証跡のエクスポートおよびインポート

Oracle Database Pumpのエクスポートおよびインポート用ダンプ・ファイルに統合監査証跡を含めることができます。

統合監査証跡は、Oracle Data Pumpを使用した、データベース全体または部分的なデータベースのいずれかのエクスポートおよびインポート操作に自動的に含められます。たとえば、部分的データベース・エクスポートの操作で統合監査証跡の表のみをエクスポートする必要がある場合は、次のexpdpコマンドを入力できます。

1. SQL*Plusで、スピルオーバー監査ファイルに書き込まれたすべてのオペレーティング・システム監査レコードを統合監査証跡の表に移動します。これにより、すべてのレコードがエクスポートされます。
2. オペレーティング・システムのプロンプトから、次のコマンドを実行します。

```
expdp system
full=y
```

```
directory=aud_dp_dir
logfile=audexp_log.log
dumpfile=audexp_dump.dmp
version=18.02.00.02.00
INCLUDE=AUDIT_TRAILS
Password: password
```

次に、エクスポート・ダンプ・ファイルを読み込んで、エクスポートされたすべてのコンテンツをインポートできます。この操作では、統合監査証跡表のみがインポートされます。

```
impdp system
full=y
directory=aud_dp_dir
dumpfile=audexp_dump.dmp
logfile=audimp_log.log
Password: password
```

この操作を実行するために特別な構成を行う必要はありません。ただし、エクスポート操作を実行する場合はEXP_FULL_DATABASEロールを持っている必要があり、インポート操作を実行する場合はIMP_FULL_DATABASEロールを持っている必要があります。

関連トピック

- 統合監査証跡へのオペレーティング・システムの監査レコードの移動

親トピック: 統合監査証跡の管理

28.1.9 統合監査の無効化

統合監査を無効にすることができます。

1. 現在有効となっている統合監査ポリシーをすべて無効化します。
   このステップは、このプロシージャが完了したときに、データベースが混合モード監査に入ることを防ぎます。
   a. AUDIT_ADMINロールを割り当てられているユーザーとして、データベース・インスタンスにログインします。
   b. 有効な統合監査ポリシーを検索するには、AUDIT_UNIFIED_ENABLED_POLICIESデータ・ディクショナリ・ビューのPOLICY_NAME列およびENABLED_OPT列を問い合せます。
   c. NOAUDIT POLICY文を実行して有効になっている各ポリシーを無効にします。
   たとえば、ユーザーpsmithに適用されたポリシーを無効にするには、次のようにします。

   ```
   NOAUDIT POLICY audit_pol BY psmith;
   ```

2. SYSOPER権限を持つユーザーSYSとして接続します。

   ```
   CONNECT sys as sysoper
   Enter password: password
   ```

   マルチテナント環境の場合は、このコマンドによりルートに接続されます。

3. データベースを停止します。

   たとえば:

   ```
   SHUTDOWN IMMEDIATE
   ```

   マルチテナント環境の場合は、このコマンドによりCDB内のすべてのPDBが終了します。

4. Windowsシステムでは、OracleServiceSIDプロセスを停止します。

   ```
   net stop OracleServiceSID
   ```

5. プラットフォームに応じて次のようにします。
   a. UNIXシステム: 次のコマンドを実行します。

```
cd $ORACLE_HOME/rdbms/lib
make -f ins_rdbms.mk uniaud_off ioracle
```

   b. Windowsシステム: `%ORACLE_HOME%/bin/orauniaud19.dll`ファイルの名前を`%ORACLE_HOME%/bin/orauniaud19.dll.dbl`に変更します。

   マルチテナント環境の場合は、これらのアクションによりCDB内のすべてのPDBの統合監査が無効化されます。

6. Windowsシステムでは、データベースを再起動する前に、`OracleServiceSID`プロセスを再起動します。

```
net start OracleServiceSID
```

7. SQL*Plusで、データベースを再起動します。

```
STARTUP
```

   マルチテナント環境の場合は、このコマンドによりCDB内のすべてのPDBが再起動します。

関連トピック

- 混合モードの監査について
- 統合監査ポリシーの無効化

親トピック: 統合監査証跡の管理

28.2 監査証跡のアーカイブ

従来のオペレーティング・システム、統合データベースおよび従来のデータベースの監査証跡をアーカイブできます。

- 従来のオペレーティング・システム監査証跡のアーカイブ
  Oracle Databaseをアップグレードした後に、従来のオペレーティング・システム監査ファイルのアーカイブを作成できます。
- 統合監査証跡および従来のデータベースの監査証跡のアーカイブ
  監査証跡は、大きくなりすぎないように定期的にアーカイブし、削除する必要があります。

親トピック: 監査証跡の管理

28.2.1 従来のオペレーティング・システム監査証跡のアーカイブ

Oracle Databaseをアップグレードした後に、従来のオペレーティング・システム監査ファイルのアーカイブを作成できます。

アップグレードしたデータベースから従来のオペレーティング・システムの監査証跡をアーカイブするには、プラットフォーム固有のオペレーティング・システム・ツールを使用して、従来のオペレーティング・システム監査ファイルのアーカイブを作成します。

- 従来のオペレーティング・システム監査ファイルをアーカイブするには、次の方法を使用します。
  - Oracle Audit Vault and Database Firewallを使用します。Oracle Databaseとは別にOracle Audit Vault and Database Firewallをインストールします。
  - テープまたはディスクにバックアップを作成します。監査ファイルの圧縮ファイルを作成し、それをテープまたはディスクに格納できます。詳細は、使用しているオペレーティング・システムのマニュアルを参照してください。

その後、監査証跡管理を容易にするために、従来のオペレーティング・システム監査レコードをパージ(削除)する必要があります。

関連トピック

- 統合監査証跡へのオペレーティング・システムの監査レコードの移動
- 監査証跡レコードの削除

親トピック: 監査証跡のアーカイブ

28.2.2 統合監査証跡および従来のデータベースの監査証跡のアーカイブ

監査証跡は、大きくなりすぎないように定期的にアーカイブし、パージする必要があります。

アーカイブしてパージすると、データベース監査証跡のパージが容易になります。

統合および従来のデータベースの監査証跡のアーカイブを作成するには、Oracle Audit Vault and Database Firewallを使用します。Oracle Databaseとは別にOracle Audit Vault and Database Firewallをインストールします。

アーカイブが完了したら、データベース監査証跡の内容を削除できます。

- 統合監査レコード、従来の標準監査レコードおよび従来のファイングレイン監査レコードをアーカイブするには、関連するレコードを通常のデータベース表にコピーします。

  たとえば:

```
INSERT INTO table SELECT ... FROM UNIFIED_AUDIT_TRAIL ...;
INSERT INTO table SELECT ... FROM SYS.AUD$ ...;
INSERT INTO table SELECT ... FROM SYS.FGA_LOG$ ...;
```

関連トピック

- 監査証跡レコードの削除

親トピック: 監査証跡のアーカイブ

28.3 監査証跡レコードの削除

DBMS_AUDIT_MGMT PL/SQLパッケージを使用して、自動削除ジョブをスケジューリングして、監査レコードを手動で削除し、他の監査証跡操作を実行できます。

- 監査証跡レコードの削除について
  様々な方法を使用して、監査証跡レコードを削除できます。
- 監査証跡の削除方法の選択
  スケジュールに基づいて定期的に、または日時を指定して削除を実行できます。
- 監査証跡の自動削除ジョブのスケジューリング
  自動削除ジョブをスケジューリングするには、オンラインREDOログとアーカイブREDOログのサイズのチューニングなどについて事前に計画する必要があります。
- 監査証跡の手動削除
  DBMS_AUDIT_MGMT.CLEAN_AUDIT_TRAILプロシージャを使用して、監査証跡を手動で削除できます。
- 他の監査証跡削除操作
  監査証跡削除には、監査証跡削除ジョブの有効化と無効化や、監査証跡削除ジョブのデフォルト間隔の設定なども含まれます。
- 例: 統合監査証跡の削除操作の直接コール
  カスタマイズされたアーカイブ・プロシージャを作成して、統合監査証跡の削除操作を直接コールできます。

関連トピック

- 統合監査証跡の管理

親トピック: 監査証跡の管理

28.3.1 監査証跡レコードの削除について

様々な方法を使用して、監査証跡レコードをパージできます。

監査証跡レコードは、定期的にアーカイブしてから削除するようにしてください。監査証跡レコードのサブセットを削除したり、指定した時間間隔で実行する削除ジョブを作成したりできます。Oracle Databaseでは、アーカイブ・タイムスタンプより前に作成された監査証跡レコードが削除されるか、すべての監査証跡レコードが削除されます。読取り/書込みデータベースおよび読取り専用データベースの両方で監査証跡レコードを削除できます。

削除プロセスでは、統合監査証跡だけでなく、前のリリースのOracle Databaseの監査証跡も考慮されます。たとえば、オペレーティング・システムまたはXMLの監査レコードが含まれるアップグレード済のデータベースを移行した場合は、この項のプロシージャを使用して、それらをアーカイブして削除できます。

監査証跡の削除タスクを実行するには、DBMS_AUDIT_MGMT PL/SQLパッケージを使用します。DBMS_AUDIT_MGMTパッケージを使用するには、AUDIT_ADMINロールが必要です。Oracle Databaseでは、DBMS_AUDIT_MGMT PL/SQLパッケージ・プロシージャのすべての実行を強制的に監査します。

Oracle Audit Vault and Database Firewallをインストールしている場合は、監査証跡パージ・プロセスは、このマニュアルで説明している手順とは異なります。たとえば、Oracle Audit Vaultでは、監査証跡が自動的にアーカイブされます。

ノート:

Oracle Database では、監査証跡からのレコードの削除はすべて例外なく監査されます。

関連トピック

- Oracle Database PL/SQLパッケージ・プロシージャおよびタイプ・リファレンス

親トピック: 監査証跡レコードの削除

28.3.2 監査証跡の削除方法の選択

スケジュールに基づいて、または日時を指定して削除を実行できます。

- スケジュールに基づいた監査証跡の定期的な削除
  すべての監査レコード、または指定したタイムスタンプより前に作成された監査レコードをスケジュールに基づいて定期的に削除できます。
- 指定時間での監査証跡の手動削除
  削除スケジュールを作成するのではなく、1回の手動操作で監査レコードを即座に削除できます。

親トピック: 監査証跡レコードの削除

28.3.2.1 スケジュールに基づいた監査証跡の定期的なパージ

すべての監査レコード、または指定したタイムスタンプより前に作成された監査レコードをスケジュールに基づいて定期的にパージできます。

たとえば、土曜日の午前2時に毎回パージを実行するようにスケジューリングできます。

1. 必要に応じて、監査表の削除プロセス中に追加生成されるレコードに対応するために、オンラインREDOログとアーカイブREDOログのサイズをチューニングします。
2. タイムスタンプおよびアーカイブ方針を計画します。
3. オプションで、監査レコードのアーカイブ・タイムスタンプを設定します。
4. 削除ジョブを作成してスケジューリングします。

関連トピック

- 監査証跡の自動削除ジョブのスケジューリング

親トピック: 監査証跡の削除方法の選択

28.3.2.2 指定時間での監査証跡の手動パージ

削除スケジュールを作成するのではなく、1回の手動操作で監査レコードを即座に削除できます。

1. 必要に応じて、監査表の削除プロセス中に追加生成されるレコードに対応するために、オンラインREDOログとアーカイブREDOログのサイズをチューニングします。
2. タイムスタンプおよびアーカイブ方針を計画します。
3. オプションで、監査レコードのアーカイブ・タイムスタンプを設定します。
4. 削除操作を実行します。

関連トピック

- 監査証跡の手動削除

親トピック: 監査証跡の削除方法の選択

28.3.3 監査証跡の自動削除ジョブのスケジューリング

自動削除ジョブをスケジューリングするには、オンラインREDOログとアーカイブREDOログのサイズのチューニングなどについて事前に計画する必要があります。

- 自動削除ジョブのスケジューリングについて
  監査証跡全体を削除することも、タイムスタンプより前に作成された一部の監査証跡のみを削除することもできます。
- ステップ1: オンラインREDOログとアーカイブREDOログのサイズのチューニング(必要に応じて)
  削除プロセスでは、REDOログが追加生成される場合があります。
- ステップ2: タイムスタンプおよびアーカイブ方針の計画
  監査レコードをアーカイブするには、これらのレコードのタイムスタンプを記録する必要があります。
- ステップ3: 監査レコードのアーカイブ・タイムスタンプの設定(必要に応じて)
  すべての監査証跡を削除する場合は、このステップを省略できます。
- ステップ4: 削除ジョブの作成とスケジューリング
  DBMS_AUDIT_MGMT PL/SQLパッケージを使用して、削除ジョブを作成およびスケジューリングできます。

親トピック: 監査証跡レコードの削除

28.3.3.1 自動削除ジョブのスケジューリングについて

監査証跡全体を削除することも、タイムスタンプより前に作成された一部の監査証跡のみを削除することもできます。

タイムスタンプより前に作成された個々の監査レコードを削除できます。

監査証跡(特に、大きなもの)を削除する場合は、完了するまでに時間がかかることがあります。削除ジョブは、データベースの負荷が低い時間帯に実行するようにスケジューリングするのが賢明です。

競合しないかぎり、異なる監査証跡タイプに対する複数の削除ジョブを作成できます。たとえば、標準監査証跡表の削除ジョブを作成し、その後でファイングレイン監査証跡表の削除ジョブを作成できます。ただし、DBMS_AUDIT_MGMT.AUDIT_TRAIL_DB_STDまたはDBMS_AUDIT_MGMT.AUDIT_TRAIL_ALLプロパティを使用して、両方のタイプまたはすべてのタイプをまとめて処理する削除ジョブを作成することはできません。また、DBMS_SCHEDULER PL/SQLパッケージで作成されるジョブは読取り専用データベースで実行されないことに注意してください。DBMS_AUDIT_MGMTで作成される自動削除ジョブではDBMS_SCHEDULERパッケージを使用してタスクをスケジューリングします。したがって、これらのジョブは読取り専用モードで開いているデータベースまたはPDBで実行できません。

親トピック: 監査証跡の自動削除ジョブのスケジューリング

28.3.3.2 ステップ1: オンラインREDOログとアーカイブREDOログのサイズのチューニング(必要に応じて)

削除プロセスでは、REDOログが追加生成される場合があります。

- 必要に応じて、監査表の削除プロセス中に追加生成されるレコードに対応するために、オンラインREDOログとアーカイブREDOログのサイズをチューニングします。

統合監査環境では、混合モードの監査環境と同じ数のREDOログは削除プロセスで生成されないため、統合監査に移行している場合は、このステップを省略できます。

関連項目:

ログ・ファイルのチューニングの詳細は、『Oracle Database管理者ガイド』を参照してください。

親トピック: 監査証跡の自動削除ジョブのスケジューリング

28.3.3.3 ステップ2: タイムスタンプおよびアーカイブ方針の計画

監査レコードをアーカイブするには、これらのレコードのタイムスタンプを記録する必要があります。

- タイムスタンプの日付を見つけるために、DBA_AUDIT_MGMT_LAST_ARCH_TSデータ・ディクショナリ・ビューを問い合せます。

その後、削除を実行すると、このアーカイブ・タイムスタンプの日付より前に作成された監査証跡レコードのみが削除されます。

レコードのタイムスタンプを設定したら、アーカイブを開始できます。

関連トピック

- ステップ3: 監査レコードのアーカイブ・タイムスタンプの設定(必要に応じて)
- 監査証跡のアーカイブ

親トピック: 監査証跡の自動削除ジョブのスケジューリング

28.3.3.4 ステップ3: 監査レコードのアーカイブ・タイムスタンプの設定(必要に応じて)

すべての監査証跡を削除する場合は、このステップを省略できます。

最後に監査レコードがアーカイブされた日時のタイムスタンプを設定できます。アーカイブ・タイムスタンプを設定すると、削除インフラストラクチャのクリーン・アップ・ポイントが指定されます。読取り専用データベースにタイムスタンプを設定する場合、

DBMS_AUDIT.MGMT.GET_LAST_ARCHIVE_TIMESTAMPファンクションを使用して、それが実行されたインスタンス用に構成された最後のアーカイブ・タイムスタンプを確認します。読取り/書込みデータベースの場合、DBA_AUDIT_MGMT_LAST_ARCH_TSデータ・ディクショナリ・ビューを問い合せます。

統合監査証跡の最終アーカイブ・タイムスタンプを調べるには、DBA_AUDIT_MGMT_LAST_ARCH_TSデータ・ディクショナリ・ビューを問い合せます。タイムスタンプを設定した後でDBMS_AUDIT_MGMT.CLEAN_AUDIT_TRAIL PL/SQLプロシージャを実行すると、そのタイムスタンプより前の時間を示すすべての監査レコードが監査証跡から削除されます。アーカイブ・タイムスタンプ設定をクリアする場合は、アーカイブ・タイムスタンプ設定のクリアを参照してください。

Oracle Database Real Application Clustersを使用している場合は、ネットワーク・タイム・プロトコル(NTP)を使用して、Oracle Databaseインスタンスをインストールしている各コンピュータ上の時間を同期化してください。たとえば、1つのOracle RACインスタンス・ノードの時間を午前11:00:00に設定し、次のOracle RACインスタンス・ノードの時間を11:00:05に設定するとします。その結果、2つのノードの時間に矛盾が生じます。ネットワーク・タイム・プロトコル(NTP)を使用して、これらのOracle RACインスタンス・ノードの時間を同期化できます。

削除ジョブのタイムスタンプを設定するには:

1. AUDIT_ADMINロールを割り当てられているユーザーとして、データベース・インスタンスにログインします。

   マルチテナント環境で、削除ジョブをスケジューリングするルートまたはPDBにログインします。ほとんどの場合、個々のPDBで削除ジョブをスケジューリングできます。

   たとえば、hrpdbというPDBにログインするには、次のようにします。

   ```
   CONNECT aud_admin@hrpdb
   Enter password: password
   Connected.
   ```

2. DBMS_AUDIT_MGMT.SET_LAST_ARCHIVE_TIMESTAMP PL/SQLプロシージャを実行して、タイムスタンプを設定します。

   たとえば:

   ```
   BEGIN
     DBMS_AUDIT_MGMT.SET_LAST_ARCHIVE_TIMESTAMP(
      AUDIT_TRAIL_TYPE     =>  DBMS_AUDIT_MGMT.AUDIT_TRAIL_UNIFIED,
      LAST_ARCHIVE_TIME    =>  '12-OCT-2013 06:30:00.00',
      RAC_INSTANCE_NUMBER  =>  1,
      CONTAINER            => DBMS_AUDIT_MGMT.CONTAINER_CURRENT);
   END;
   /
   ```

   この例では、次のようになります。

   - AUDIT_TRAIL_TYPEは、監査証跡タイプを指定します。
     DBMS_AUDIT_MGMT.AUDIT_TRAIL_UNIFIEDは、これを統合監査証跡に設定します。

     前のリリースの従来の監査データがまだ存在するアップグレード済のデータベースの場合:

     - DBMS_AUDIT_MGMT.AUDIT_TRAIL_AUD_STDは、従来の標準監査証跡表AUD$に使用されます。(この設定は読取り専用データベースには適用されません。)
     - DBMS_AUDIT_MGMT.AUDIT_TRAIL_FGA_STDは、従来のファイングレイン監査証跡表FGA_LOG$に使用されます。(この設定は読取り専用データベースには適用されません。)
     - DBMS_AUDIT_MGMT.AUDIT_TRAIL_OSは、拡張子.audが付けられた従来のオペレーティング・システム監査証跡ファイルに使用されます。(この設定はWindowsイベント・ログ・エントリには適

用されません。)

- DBMS_AUDIT_MGMT.AUDIT_TRAIL_XMLは、従来のXMLオペレーティング・システム監査証跡ファイルに使用されます。

AUDSYS.AUD$UNIFIED表またはオペレーティング・システムの過剰ファイルからレコードをアーカイブする場合:

- DBMS_AUDIT_MGMT.AUDIT_TRAIL_UNIFIED_TABLEは、AUDSYS.AUD$UNIFIED表からレコードをアーカイブします。
- DBMS_AUDIT_MGMT.AUDIT_TRAIL_UNIFIED_FILESは、各データベース(プライマリまたはスタンバイ)のオペレーティング・システムの過剰ファイルからレコードをアーカイブします。

- LAST_ARCHIVE_TIMEは、タイムスタンプを指定します(AUDIT_TRAIL_UNIFIED、AUDIT_TRAIL_AUD_STDおよびAUDIT_TRAIL_FGA_STDの場合は、YYYY-MM-DD HH:MI:SS.FF UTC (協定世界時)形式で指定し、AUDIT_TRAIL_OSおよびAUDIT_TRAIL_XMLの場合は、ローカル・タイム・ゾーンで指定します)。この値には、将来のシステム日付またはタイムスタンプ(SYSDATE + 1や将来の日付など)を入力しないでください。
- RAC_INSTANCE_NUMBERは、Oracle RACインストールのインスタンス番号を指定します。この設定では、シングル・インスタンス・データベースは関係がありません。監査証跡タイプとしてDBMS_AUDIT_MGMT.AUDIT_TRAIL_AUD_STDまたはDBMS_AUDIT_MGMT.AUDIT_TRAIL_FGA_STDを指定した場合は、RAC_INSTANCE_NUMBER引数を省略できます。これは、Oracle RACインストールの場合でも、存在するAUD$またはFGA_LOG$表は1つのみであるためです。デフォルトは、NULLです。現在のインスタンスのインスタンス番号を調べるには、SQL*PlusでSHOW PARAMETER INSTANCE_NUMBERコマンドを発行します。
- CONTAINERは、タイムスタンプをマルチテナント環境に適用します。DBMS_AUDIT_MGMT.CONTAINER_CURRENTは現在のPDBを指定し、DBMS_AUDIT_MGMT.CONTAINER_ALLはマルチテナント環境のすべてのPDBに適用されます。

  CONTAINERをDBMS_MGMT.CONTAINER_ALLに設定できるのはルートからのみで、DBMS_MGMT.CONTAINER_CURRENTに設定できるのはPDBからのみです。

通常、タイムスタンプを設定したら、DBMS_AUDIT_MGMT.CLEAN_AUDIT_TRAIL PL/SQLプロシージャを使用して、タイムスタンプの日付より前に作成された監査レコードを削除できます。

親トピック: 監査証跡の自動削除ジョブのスケジューリング

28.3.3.5 ステップ4: 削除ジョブの作成とスケジューリング

DBMS_AUDIT_MGMT PL/SQLパッケージを使用して、削除ジョブを作成およびスケジューリングできます。

- DBMS_AUDIT_MGMT.CREATE_PURGE_JOB PL/SQLプロシージャを実行して、削除ジョブを作成してスケジューリングします。

  たとえば:

```
CONNECT aud_admin@hrpdb
Enter password: password
Connected.
BEGIN
  DBMS_AUDIT_MGMT.CREATE_PURGE_JOB (
   AUDIT_TRAIL_TYPE            => DBMS_AUDIT_MGMT.AUDIT_TRAIL_UNIFIED,
```

```
   AUDIT_TRAIL_PURGE_INTERVAL  => 12,
   AUDIT_TRAIL_PURGE_NAME      => 'Audit_Trail_PJ',
   USE_LAST_ARCH_TIMESTAMP     => TRUE,
   CONTAINER                   => DBMS_AUDIT_MGMT.CONTAINER_CURRENT);
END;
/
```

この例では、次のようになります。

- AUDIT_TRAIL_TYPE: 監査証跡タイプを指定します。DBMS_AUDIT_MGMT.AUDIT_TRAIL_UNIFIEDは、これを統合監査証跡に設定します。

  前のリリースの監査データがまだ存在するアップグレード済のデータベースの場合:

  - DBMS_AUDIT_MGMT.AUDIT_TRAIL_AUD_STDは、標準監査証跡表AUD$に使用されます。(この設定は読取り専用データベースには適用されません。)
  - DBMS_AUDIT_MGMT.AUDIT_TRAIL_FGA_STDは、ファイングレイン監査証跡表FGA_LOG$に使用されます。(この設定は読取り専用データベースには適用されません。)
  - DBMS_AUDIT_MGMT.AUDIT_TRAIL_DB_STDは、標準監査証跡表とファイングレイン監査証跡表の両方に使用されます。(この設定は読取り専用データベースには適用されません。)
  - DBMS_AUDIT_MGMT.AUDIT_TRAIL_OSは、拡張子.audが付けられたオペレーティング・システム監査証跡ファイルに使用されます。(この設定はWindowsイベント・ログ・エントリには適用されません。)
  - DBMS_AUDIT_MGMT.AUDIT_TRAIL_XMLは、XMLオペレーティング・システム監査証跡ファイルに使用されます。
  - DBMS_AUDIT_MGMT.AUDIT_TRAIL_FILESは、オペレーティング・システム監査証跡ファイルとXML監査証跡ファイルの両方に使用されます。
  - DBMS_AUDIT_MGMT.AUDIT_TRAIL_ALLは、すべての監査証跡レコード、つまりデータベース監査証跡タイプとオペレーティング・システム監査証跡タイプの両方に使用されます。(この設定は読取り専用データベースには適用されません。)

  AUDSYS.AUD$UNIFIED表またはオペレーティング・システムの過剰ファイルからレコードをパージする場合:

  - DBMS_AUDIT_MGMT.AUDIT_TRAIL_UNIFIED_TABLEは、AUDSYS.AUD$UNIFIED表からレコードをパージします。
  - DBMS_AUDIT_MGMT.AUDIT_TRAIL_UNIFIED_FILESは、各データベース(プライマリまたはスタンバイ)のオペレーティング・システムの過剰ファイルからレコードをパージします。

- AUDIT_TRAIL_PURGE_INTERVALは、この削除ジョブを実行する間隔を時間単位で指定します。DBMS_AUDIT_MGMT.CREATE_PURGE_JOBプロシージャを実行すると、計時が開始されます(この例では、このプロシージャを実行してから12時間後)。後でこの値を更新する場合は、DBMS_AUDIT_MGMT.SET_PURGE_JOB_INTERVALプロシージャを実行します。
- USE_LAST_ARCH_TIMESTAMPは、次の設定のいずれかを受け入れます。
  - TRUEは、最終アーカイブ・タイムスタンプより前に作成された監査レコードを削除します。最後に記録されたタイムスタンプを確認するには、読取り/書込みデータベースの場合はDBA_AUDIT_MGMT_LAST_ARCH_TSデータ・ディクショナリ・ビューのLAST_ARCHIVE_TS列を問い合せ、読取り専用データベースの場合はDBMS_AUDIT_MGMT.GET_LAST_ARCHIVE_TIMESTAMPファンクションを使用します。デフォルト値は

TRUEです。USE_LAST_ARCH_TIMESTAMPはTRUEに設定することをお薦めします。

- FALSEは、最終アーカイブ・タイムスタンプを考慮せずに、すべての監査レコードを削除します。削除してはならない監査レコードを誤って削除しないように、この設定を使用する際には注意が必要です。

- CONTAINERは、マルチテナント環境で削除ジョブを作成する場所を定義するのに使用されます。CONTAINERをDBMS_AUDIT_MGMT.CONTAINER_CURRENTに設定すると、現在のPDBからのみ使用および表示可能になり、管理されます。DBMS_AUDIT_MGMT.CONTAINER_ALL設定では、ジョブがルートに作成されます。定義済のジョブ・スケジュールに従って実行されるグローバル・ジョブとしてジョブを定義します。ジョブが起動されると、マルチテナント環境のすべてのPDBの監査証跡がクリーンアップされます。ジョブをルートに作成した場合は、ルートからのみ表示可能です。したがって、有効化、無効化および削除はルートからのみ可能です。

親トピック: 監査証跡の自動削除ジョブのスケジューリング

28.3.4 監査証跡の手動削除

DBMS_AUDIT_MGMT.CLEAN_AUDIT_TRAILプロシージャを使用して、監査証跡を手動でパージできます。

- 監査証跡の手動削除について
  監査証跡は、削除ジョブをスケジューリングしなくても、手動で即座に削除できます。
- DBMS_AUDIT_MGMT.CLEAN_AUDIT_TRAILを使用した監査証跡の手動削除
  準備ステップの完了後、DBMS_AUDIT_MGMT.CLEAN_AUDIT_TRAILプロシージャを使用して、監査証跡を手動で削除できます。

親トピック: 監査証跡レコードの削除

28.3.4.1 監査証跡の手動削除について

監査証跡は、削除ジョブをスケジューリングしなくても、手動で即座に削除できます。

削除ジョブと同様に、アーカイブ・タイムスタンプの日付より前に作成された監査証跡レコード、または監査証跡内のすべてのレコードを削除できます。このプロシージャを実行した場合、現行の監査ディレクトリのみがクリーン・アップされます。

前のリリースの監査証跡がまだ存在しているアップグレード済のデータベースの場合は、DBMS_AUDIT_MGMT.CLEAN_AUDIT_TRAIL PL/SQLプロシージャに関する次の点に注意してください。

- DBMS_AUDIT_MGMTパッケージではWindowsイベントビューアのクリーン・アップはサポートされていないため、Microsoft WindowsでAUDIT_TRAIL_TYPEプロパティをDBMS_AUDIT_MGMT.AUDIT_TRAIL_OSに設定しても効果はありません。Windows上のオペレーティング・システム監査レコードはWindowsイベントビューアに書き込まれるためです。DBMS_AUDIT_MGMTパッケージでは、このタイプのクリーン・アップ操作はサポートされていません。

- UNIXプラットフォームで、AUDIT_SYSLOG_LEVEL初期化パラメータを設定している場合は、Oracle Databaseによってオペレーティング・システムのログ・ファイルがsyslogファイルに書き込まれます。(syslogファイルを使用するよう構成すると、メッセージがsyslogデーモン・プロセスに送信されることに注意してください。syslogデーモン・プロセスは、syslogファイルへのコミットされた書込みを示す確認応答をOracle Databaseに返しません。)AUDIT_TRAIL_TYPEプロパティをDBMS_AUDIT_MGMT.AUDIT_TRAIL_OSに設定すると、監査ディレクトリ内の.audファイルのみが削除されます(このディレクトリは、AUDIT_FILE_DEST初期化パラメータで指定します)。

親トピック: 監査証跡の手動削除

28.3.4.2 DBMS_AUDIT_MGMT.CLEAN_AUDIT_TRAILを使用した監査証跡の手動パージ

準備ステップの完了後、DBMS_AUDIT_MGMT.CLEAN_AUDIT_TRAILプロシージャを使用して、監査証跡を手動で削除

できます。

1. 監査証跡がオペレーティング・システムのログ・ファイル(syslog)に書き込まれるようにAUDIT_SYSLOG_LEVEL初期化パラメータを設定した場合は、次のことを確認します。
   - 監査証跡ファイルが現在書込み中ではないことを確認します。
   - 監査証跡ファイルに関連付けられているセッションIDがPMONプロセスに所有されていないことを確認します。

   これらの条件のどちらかに当てはまる場合は、監査証跡を削除できません。
2. 監査証跡の自動削除ジョブのスケジューリングで説明した次のステップを実行します。
   - ステップ1: オンラインREDOログとアーカイブREDOログのサイズのチューニング(必要に応じて)
   - ステップ2: タイムスタンプおよびアーカイブ方針の計画
   - ステップ3: 監査レコードのアーカイブ・タイムスタンプの設定(必要に応じて)
3. マルチテナント環境を使用している場合は、削除ジョブを作成したデータベースに接続します。

   削除ジョブをルートに作成した場合は、ルートにログインする必要があります。削除ジョブを特定のPDBに作成した場合は、そのPDBにログインします。

   たとえば:

```
CONNECT aud_admin@hrpdb
Enter password: password
Connected.
```

4. DBMS_AUDIT_MGMT.CLEAN_AUDIT_TRAIL PL/SQLプロシージャを実行して、監査証跡レコードを削除します。

   たとえば:

```
BEGIN
  DBMS_AUDIT_MGMT.CLEAN_AUDIT_TRAIL(
   AUDIT_TRAIL_TYPE            =>  DBMS_AUDIT_MGMT.AUDIT_TRAIL_UNIFIED,
   USE_LAST_ARCH_TIMESTAMP     =>  TRUE,
   CONTAINER                   =>  DBMS_AUDIT_MGMT.CONTAINER_CURRENT );
END;
/
```

   この例では、次のようになります。

   - AUDIT_TRAIL_TYPE: 監査証跡タイプを指定します。
     DBMS_AUDIT_MGMT.AUDIT_TRAIL_UNIFIEDは、これを統合監査証跡に設定します。

     前のリリースの監査データがまだ存在するアップグレード済のデータベースの場合:
     - DBMS_AUDIT_MGMT.AUDIT_TRAIL_AUD_STD: 標準監査証跡表AUD$です。(この設定は読取り専用データベースには適用されません。)
     - DBMS_AUDIT_MGMT.AUDIT_TRAIL_FGA_STD: ファイングレイン監査証跡表FGA_LOG$です。(この設定は読取り専用データベースには適用されません。)
     - DBMS_AUDIT_MGMT.AUDIT_TRAIL_DB_STD: 標準監査証跡表とファイングレイン監査証跡表の両方です。(この設定は読取り専用データベースには適用されません)
     - DBMS_AUDIT_MGMT.AUDIT_TRAIL_OS: 拡張子.audが付けられたオペレーティング・システム監査証跡ファイル。(この設定はWindowsイベント・ログ・エントリには適用されません。)

- DBMS_AUDIT_MGMT.AUDIT_TRAIL_XML: XMLオペレーティング・システム監査証跡ファイルです。
- DBMS_AUDIT_MGMT.AUDIT_TRAIL_FILES: オペレーティング・システム監査証跡ファイルとXML監査証跡ファイルの両方です。
- DBMS_AUDIT_MGMT.AUDIT_TRAIL_ALL: すべての監査証跡レコード、つまりデータベース監査証跡タイプとオペレーティング・システム監査証跡タイプの両方です。(この設定は読取り専用データベースには適用されません。)

AUDSYS.AUD$UNIFIED表またはオペレーティング・システムの過剰ファイルからレコードをパージする場合:

- DBMS_AUDIT_MGMT.AUDIT_TRAIL_UNIFIED_TABLEは、AUDSYS.AUD$UNIFIED表からレコードをパージします。
- DBMS_AUDIT_MGMT.AUDIT_TRAIL_UNIFIED_FILESは、各データベース(プライマリまたはスタンバイ)のオペレーティング・システムの過剰ファイルからレコードをパージします。

- USE_LAST_ARCH_TIMESTAMP: 次の設定のいずれかを入力します。
  - TRUE: 最終アーカイブ・タイムスタンプより前に作成された監査レコードを削除します。アーカイブ・タイムスタンプを設定する方法は、ステップ3: 監査レコードのアーカイブ・タイムスタンプの設定(必要に応じて)を参照してください。デフォルト(推奨)値はTRUEです。USE_LAST_ARCH_TIMESTAMPはTRUEに設定することをお薦めします。
  - FALSE: 最終アーカイブ・タイムスタンプを考慮せずに、すべての監査レコードを削除します。削除してはならない監査レコードを誤って削除しないように、この設定を使用する際には注意が必要です。
- CONTAINER: マルチテナント環境にクレンジングを適用します。
  DBMS_AUDIT_MGMT.CONTAINER_CURRENTは、ローカルのPDBを指定し、
  DBMS_AUDIT_MGMT.CONTAINER_ALLは、すべてのデータベースに適用されます。

親トピック: 監査証跡の手動削除

28.3.5 他の監査証跡削除操作

監査証跡削除には、監査証跡削除ジョブの有効化と無効化や、監査証跡削除ジョブのデフォルト間隔の設定なども含まれます。

- 監査証跡の削除ジョブを使用可能または使用禁止にする方法
  DBMS_AUDIT_MGMT.SET_PURGE_JOB_STATUSプロシージャで、監査証跡削除ジョブを有効または無効にします。
- 指定した削除ジョブに対するデフォルトの監査証跡削除ジョブの間隔の設定
  次の削除ジョブ操作が実行されるまでのデフォルトの削除操作間隔を時間単位で設定できます。
- 監査証跡の削除ジョブの削除
  既存の監査証跡削除ジョブを削除できます。
- アーカイブ・タイムスタンプ設定の消去
  DBMS_AUDIT_MGMT.CLEAR_LAST_ARCHIVE_TIMESTAMPプロシージャで、アーカイブ・タイムスタンプの設定をクリアできます。

親トピック: 監査証跡レコードの削除

28.3.5.1 監査証跡の削除ジョブを使用可能または使用禁止にする方法

DBMS_AUDIT_MGMT.SET_PURGE_JOB_STATUSプロシージャで、監査証跡削除ジョブを有効または無効にします。

マルチテナント環境では、DBMS_AUDIT_MGMT.SET_PURGE_JOB_STATUSプロシージャの実行場所は、削除ジョブの場所(DBMS_MGMT.CREATE_PURGE_JOBプロシージャのCONTAINERパラメータによって決定)によって異なります。CONTAINERをCONTAINER_ALLに設定した場合(削除ジョブをルートに作成する場合)は、DBMS_AUDIT_MGMT.SET_PURGE_JOB_STATUSプロシージャをルートから実行する必要があります。CONTAINERをCONTAINER_CURRENTに設定した場合は、DBMS_AUDIT_MGMT.SET_PURGE_JOB_STATUSプロシージャが作成されたPDBから、このプロシージャを実行する必要があります。

- 監査証跡パージ・ジョブを有効または無効にするには、DBMS_AUDIT_MGMT.SET_PURGE_JOB_STATUSPL/SQLプロシージャを使用します。

  たとえば、削除ジョブをhrpdb PDBに作成したとします。

```
CONNECT aud_admin@hrpdb
Enter password: password
Connected.
BEGIN
 DBMS_AUDIT_MGMT.SET_PURGE_JOB_STATUS(
  AUDIT_TRAIL_PURGE_NAME       => 'Audit_Trail_PJ',
  AUDIT_TRAIL_STATUS_VALUE     => DBMS_AUDIT_MGMT.PURGE_JOB_ENABLE);
END;
/
```

この例では、次のようになります。

- AUDIT_TRAIL_PURGE_NAMEは、Audit_Trail_PJという削除ジョブを指定します。既存の削除ジョブを調べるには、DBA_AUDIT_MGMT_CLEANUP_JOBSデータ・ディクショナリ・ビューのJOB_NAMEおよびJOB_STATUS列を問い合せます。
- AUDIT_TRAIL_STATUS_VALUEは、次のプロパティのいずれかを受け入れます。
  - DBMS_AUDIT_MGMT.PURGE_JOB_ENABLEは、指定した削除ジョブを使用可能にします。
  - DBMS_AUDIT_MGMT.PURGE_JOB_DISABLEは、指定した削除ジョブを使用禁止にします。

親トピック: 他の監査証跡削除操作

28.3.5.2 指定した削除ジョブに対するデフォルトの監査証跡削除ジョブの間隔の設定

次の削除ジョブ操作が実行されるまでのデフォルトの削除操作間隔を時間単位で設定できます。

DBMS_AUDIT_MGMT.CREATE_PURGE_JOBプロシージャで使用される間隔設定が、この設定よりも優先されます。

- 特定のパージ・ジョブの監査証跡パージ・ジョブのデフォルト間隔を設定するには、DBMS_AUDIT_MGMT.SET_PURGE_JOB_INTERVALプロシージャを実行します。

  たとえば、削除ジョブをhrpdb PDBに作成したとします。

```
CONNECT aud_admin@hrpdb
Enter password: password
Connected.
BEGIN
 DBMS_AUDIT_MGMT.SET_PURGE_JOB_INTERVAL(
  AUDIT_TRAIL_PURGE_NAME        => 'Audit_Trail_PJ',
  AUDIT_TRAIL_INTERVAL_VALUE    => 24);
END;
/
```

この例では、次のようになります。

- AUDIT_TRAIL_PURGE_NAMEは、監査証跡の削除ジョブの名前を指定します。既存の削除ジョブのリストを調べるには、DBA_AUDIT_MGMT_CLEANUP_JOBSデータ・ディクショナリ・ビューのJOB_NAMEおよびJOB_STATUS列を問い合せます。
- AUDIT_TRAIL_INTERVAL_VALUEは、DBMS_AUDIT_MGMT.CREATE_PURGE_JOBプロシージャで設定したデフォルトの間隔(時間単位)を更新します。1から999の値を入力します。削除ジョブを実行すると、計時が開始されます。

マルチテナント環境では、DBMS_AUDIT_MGMT.SET_PURGE_JOB_INTERVALプロシージャの実行場所は、削除ジョブの場所(DBMS_MGMT.CREATE_PURGE_JOBプロシージャのCONTAINERパラメータによって決定)によって異なります。CONTAINERをCONTAINER_ALLに設定した場合、削除ジョブはルートに存在するため、DBMS_AUDIT_MGMT.SET_PURGE_JOB_STATUSプロシージャをルートから実行する必要があります。CONTAINERをCONTAINER_CURRENTに設定した場合は、DBMS_AUDIT_MGMT.SET_PURGE_JOB_INTERVALプロシージャが作成されたPDBから、このプロシージャを実行する必要があります。

親トピック: 他の監査証跡削除操作

28.3.5.3 監査証跡の削除ジョブの削除

既存の監査証跡パージ・ジョブを削除できます。

既存の削除ジョブを調べるには、DBA_AUDIT_MGMT_CLEANUP_JOBSデータ・ディクショナリ・ビューのJOB_NAMEおよびJOB_STATUS列を問い合せます。

- 監査証跡パージ・ジョブを削除するには、DBMS_AUDIT_MGMT.DROP_PURGE_JOB PL/SQLプロシージャを使用します。

たとえば、削除ジョブをhrpdb PDBに作成したとします。

```
CONNECT aud_admin@hrpdb
Enter password: password
Connected.
BEGIN
 DBMS_AUDIT_MGMT.DROP_PURGE_JOB(
  AUDIT_TRAIL_PURGE_NAME  => 'Audit_Trail_PJ');
END;
/
```

マルチテナント環境では、DBMS_AUDIT_MGMT.DROP_PURGE_JOBプロシージャの実行場所は、削除ジョブの場所(DBMS_MGMT.CREATE_PURGE_JOBプロシージャのCONTAINERパラメータによって決定)によって異なります。CONTAINERをCONTAINER_ALLに設定した場合、削除ジョブはルートに存在するため、DBMS_AUDIT_MGMT.SET_PURGE_JOB_STATUSプロシージャをルートから実行する必要があります。CONTAINERをCONTAINER_CURRENTに設定した場合は、DBMS_AUDIT_MGMT.DROP_PURGE_JOB_INTERVALプロシージャが作成されたPDBから、このプロシージャを実行する必要があります。

親トピック: 他の監査証跡削除操作

28.3.5.4 アーカイブ・タイムスタンプ設定のクリア

DBMS_AUDIT_MGMT.CLEAR_LAST_ARCHIVE_TIMESTAMPプロシージャで、アーカイブ・タイムスタンプの設定をクリアできます。

監査証跡ログのクリーン・アップの履歴を検索するには、OBJECT_NAMEがDBMS_AUDIT_MGMT、OBJECT_SCHEMAが

SYSで、SQL_TEXTがLIKE %DBMS_AUDIT_MGMT.CLEAN_AUDIT_TRAIL%に設定されていることを条件として使用し、UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューを問い合せます。

- アーカイブ・タイムスタンプ設定を消去するには、DBMS_AUDIT_MGMT.CLEAR_LAST_ARCHIVE_TIMESTAMP PL/SQLプロシージャを使用して、監査証跡タイプと、マルチテナント環境の場合はコンテナ・タイプを指定します。

  たとえば、削除ジョブをhrpdb PDBに作成したとします。

```
CONNECT aud_admin@hrpdb
Enter password: password
Connected.
BEGIN
  DBMS_AUDIT_MGMT.CLEAR_LAST_ARCHIVE_TIMESTAMP(
   AUDIT_TRAIL_TYPE     =>  DBMS_AUDIT_MGMT.AUDIT_TRAIL_UNIFIED,
   CONTAINER            =>  DBMS_AUDIT_MGMT.CONTAINER_CURRENT);
END;
/
```

この例では、次のようになります。

- 統合監査証跡の場合は、AUDIT_TRAIL_TYPEが設定されます。AUDIT_TRAIL_TYPEプロパティをDBMS_AUDIT_MGMT.AUDIT_TRAIL_OSまたはDBMS_AUDIT_MGMT.AUDIT_TRAIL_XMLに設定している場合、RAC_INSTANCE_NUMBERを0に設定することはできません。AUDIT_TRAIL_TYPEをDBMS_AUDIT_MGMT.AUDIT_TRAIL_UNIFIEDに設定した場合は、RAC_INSTANCE_NUMBER設定を省略できます。

  AUDSYS.AUD$UNIFIED表からアーカイブ・タイムスタンプをクリアするには、DBMS_AUDIT_MGMT.AUDIT_TRAIL_UNIFIED_TABLEを設定します。各データベース(プライマリまたはスタンバイ)のオペレーティング・システムの過剰ファイルからアーカイブ・タイムスタンプをクリアするには、DBMS_AUDIT_MGMT.AUDIT_TRAIL_UNIFIED_FILESを設定します。

- CONTAINERは、タイムスタンプをマルチテナント環境に適用します。DBMS_AUDIT_MGMT.CONTAINER_CURRENTは、ローカルのPDBを指定し、DBMS_AUDIT_MGMT.CONTAINER_ALLは、すべてのデータベースに適用されます。

親トピック: 他の監査証跡削除操作

28.3.6 例: 統合監査証跡の削除操作の直接コール

カスタマイズされたアーカイブ・プロシージャを作成して、統合監査証跡の削除操作を直接コールできます。

例28-1の疑似コードはデータベース監査証跡削除操作を作成し、この操作は、ユーザーが統合監査証跡に対してDBMS_ADUIT.CLEAN_AUDIT_TRAILプロシージャを起動することでコールします。

この削除操作では、ループを使用することで、前回アーカイブされたタイムスタンプより前に作成されたレコードを削除します。ループは監査レコードをアーカイブし、どの監査レコードがアーカイブされたかを計算してSetCleanUpAuditTrailコールを使用して最終アーカイブ・タイムスタンプを設定し、それからCLEAN_AUDIT_TRAILプロシージャをコールします。この例では、重要なステップはboldで示しています。

例28-1 データベース監査証跡の削除操作の直接コール

```
-- 1. Set the last archive timestamp:
PROCEDURE SetCleanUpAuditTrail()
 BEGIN
  CALL FindLastArchivedTimestamp(AUD$);
  DBMS_AUDIT_MGMT.SET_LAST_ARCHIVE_TIMESTAMP(
   AUDIT_TRAIL_TYPE          => DBMS_AUDIT_MGMT.AUDIT_TRAIL_UNIFIED,
```

```
   LAST_ARCHIVE_TIME         => '23-AUG-2013 12:00:00',
   CONTAINER                 => DBMS_AUDIT_MGMT.CONTAINER_CURRENT);
 END;
/
-- 2. Run a customized archive procedure to purge the audit trail records:
BEGIN
  CALL MakeAuditSettings();
  LOOP (/* How long to loop*/)
    -- Invoke function for audit record archival
    CALL DoUnifiedAuditRecordArchival();

    CALL SetCleanUpAuditTrail();
    IF(/* Clean up is needed immediately */)
      DBMS_AUDIT_MGMT.CLEAN_AUDIT_TRAIL(
       AUDIT_TRAIL_TYPE        => DBMS_AUDIT_MGMT.AUDIT_TRAIL_UNIFIED,
       USE_LAST_ARCH_TIMESTAMP => TRUE,
       CONTAINER               => DBMS_AUDIT_MGMT.CONTAINER_CURRENT );
    END IF
  END LOOP /*LOOP*/
END; /* PROCEDURE */
/
```

親トピック: 監査証跡レコードの削除

28.4 監査証跡管理のデータ・ディクショナリ・ビュー

Oracle Databaseには、監査証跡の管理設定に関する情報を表示するデータ・ディクショナリ・ビューが用意されています。

表28-2に、これらのビューを示します。

表28-2 監査証跡の管理設定に関する情報を表示するビュー

ビュー	説明
DBA_AUDIT_MGMT_CLEAN_EVENTS	従来(統合以外)の監査証跡の削除イベントの履歴が表示されます。定期的に、AUDIT_ADMIN ロールが付与されているユーザーで接続し、このビューが大きくなりすぎないように、内容を削除する必要があります。たとえば: DELETE FROM DBA_AUDIT_MGMT_CLEAN_EVENTS; このビューは、読取り/書込みデータベースにのみ適用されます。読取り専用データベースの場合、削除イベントの履歴はアラート・ログにあります。 統合監査の場合、削除されたイベントの履歴を検索するには、OBJECT_NAME が DBMS_AUDIT_MGMT、OBJECT_SCHEMA が SYS で、SQL_TEXT が LIKE %DBMS_AUDIT_MGMT.CLEAN_AUDIT_TRAIL%に設定されていることを条件として使用し、UNIFIED_AUDIT_TRAIL データ・ディクショナリ・ビューを問い合せます。
DBA_AUDIT_MGMT_CLEANUP_JOBS	現在構成されている監査証跡の削除ジョブが表示されます。

ビュー	説明
DBA_AUDIT_MGMT_CONFIG_PARAMS	現在構成されている監査証跡プロパティが表示されます。これらのプロパティは、DBMS_AUDIT_MGMT PL/SQL パッケージで使用されます。
DBA_AUDIT_MGMT_LAST_ARCH_TS	監査証跡の削除用に設定された最後のアーカイブ・タイムスタンプが表示されます。

関連トピック

- Oracle Databaseリファレンス

親トピック: 監査証跡の管理

付録

第VII部は、リファレンス用の一連の付録で構成されています。

- Oracle Databaseの安全性の維持
  Oracleには、ユーザー・アカウント、権限、ロール、パスワードおよびデータの保護に関するアドバイスなど、データベースの安全性を維持するためのガイドラインがあります。
- データ暗号化および整合性パラメータ
  `sqlnet.ora`ファイルには、データ暗号化および整合性パラメータがあります。
- Kerberos、TLSおよびRADIUS認証パラメータ
  `sqlnet.ora`ファイルおよびデータベース初期化ファイルには、Kerberos、RADIUSまたはTLS認証パラメータが用意されています。
- RADIUSを使用した認証デバイスの統合
  RADIUSチャレンジ・レスポンス・ユーザー・インタフェースは、RADIUS構成での認証をさらに強化します。
- Oracle Database FIPS 140-2の設定
  Oracleでは、米国連邦情報処理標準(FIPS)の標準である140-2がサポートされます。
- 公開キー・インフラストラクチャ(PKI)要素の管理
  `orapki`コマンドライン・ユーティリティとsqlnet.oraのパラメータを使用して、公開キー・インフラストラクチャ(PKI)要素を管理できます。
- 統合監査の移行による各監査機能への影響
  統合監査の移行前に、Oracle Database 12cリリース1 (12.1)以前の監査機能の大部分を使用できます。

A Oracle Databaseの安全性の維持

Oracleには、ユーザー・アカウント、権限、ロール、パスワードおよびデータの保護に関するアドバイスなど、データベースの安全性を維持するためのガイドラインがあります。

- Oracle Databaseセキュリティ・ガイドラインについて
  情報のセキュリティとプライバシ、および企業の資産とデータの保護は、どのようなビジネスにおいても非常に重要です。
- セキュリティ・パッチのダウンロードと脆弱性についてのOracleへの連絡
  セキュリティ・パッチが入手可能になったら、必ずすぐに適用する必要があります。問題が発生した場合、脆弱性についてOracleに連絡してください。
- ユーザー・アカウントと権限の保護に関するガイドライン
  Oracleには、ユーザー・アカウントと権限を保護するためのガイドラインがあります。
- ロールの保護に関するガイドライン
  Oracleには、ロール管理用のガイドラインがあります。
- パスワードの保護に関するガイドライン
  オラクル社では、様々な状況でのパスワードの保護についてガイドラインを提供しています。
- データの保護に関するガイドライン
  Oracleには、システムでデータを保護するためのガイドラインがあります。
- ORACLE_LOADERアクセス・ドライバの保護に関するガイドライン
  Oracleには、ORACLE_LOADERアクセス・ドライバを保護するためのガイドラインがあります。
- データベースのインストールと構成の保護に関するガイドライン
  Oracleには、データベースのインストールと構成を保護するためのガイドラインがあります。
- ネットワークの保護に関するガイドライン
  ネットワーク通信のセキュリティは、クライアント、リスナー、および完全な保護を確保するためのネットワーク・ガイドラインを使用することで改善されます。
- 外部プロシージャの保護に関するガイドライン
  ENFORCE_CREDENTIAL環境変数では、extprocプロセスがどのようにユーザー資格証明およびコールアウト関数を認証するかを制御します。
- 監査に関するガイドライン
  Oracleには、監査のためのガイドラインがあります。
- CONNECTロール変更への対処
  CONNECTロールはOracle Databaseリリース7で導入され、データベース・ロールに新しい堅牢なサポートが追加されました。

親トピック: 付録

A.1 Oracle Databaseセキュリティ・ガイドラインについて

情報のセキュリティとプライバシ、および企業の資産とデータの保護は、どのようなビジネスにおいても非常に重要です。

Oracle Databaseは、厳重なデータ保護、監査、スケーラブルなセキュリティ、安全なホスティング、データ交換などの最先端のセキュリティ機能を提供することによって、情報セキュリティに対するニーズに幅広く対応します。

Oracle Databaseは、セキュリティ機能において業界をリードしています。Oracle Databaseが提供するセキュリティ機能をすべてのビジネス環境で最大限に活用するには、データベース自体の保護が万全であることが必須条件です。

セキュリティ・ガイドラインは、運用データベースのデプロイメントに関する業界標準として推奨されるセキュリティ・プラクティスに準拠し、それを推奨することによって、Oracle Databaseを安全に構成する方法を示しています。この項で説明するガイドラインの多くは、米国サーベンス・オクスリー法(Sarbanes-Oxley Act)で規定されているような一般的な規制要件に対応しています。法令順守、個人を特定できる情報の保護、および内部の脅威に対するOracle Databaseの対処方法の詳細は、次の場所を参照してください。

http://www.oracle.com/technetwork/topics/security/whatsnew/index.html

親トピック: Oracle Databaseの安全性の維持

A.2 セキュリティ・パッチのダウンロードと脆弱性についてのOracleへの連絡

セキュリティ・パッチが入手可能になったら、必ずすぐに適用する必要があります。問題が発生した場合、脆弱性についてOracleに連絡してください。

- セキュリティ・パッチと回避ソリューションのダウンロード
  セキュリティ・パッチは、Oracle Databaseを稼働しているオペレーティング・システム、Oracle Database自体、およびインストール済のOracle Databaseのオプションとコンポーネントすべてに適用します。
- Oracle Databaseの脆弱性に関するOracleのセキュリティ窓口への連絡
  Oracle Databaseの脆弱性に関してOracleのセキュリティ窓口に連絡できます。

親トピック: Oracle Databaseの安全性の維持

A.2.1 セキュリティ・パッチと回避ソリューションのダウンロード

セキュリティ・パッチは、Oracle Databaseを稼働しているオペレーティング・システム、Oracle Database自体、およびインストール済のOracle Databaseのオプションとコンポーネントすべてに適用します。

- セキュリティ・パッチおよび回避ソリューションをダウンロードするには:
  - セキュリティ・パッチについては、Oracle Technology Networkのセキュリティ・サイトで、Oracleによってリリースされたセキュリティ・アラートに関する詳細を定期的に確認してください:
    http://www.oracle.com/technetwork/topics/security/alerts-086861.html
  - Oracleワールドワイド・サポート・サービスのサイト、My Oracle Supportで、セキュリティに関するパッチの入手可能性や予定などを確認してください:

    https://support.oracle.com

親トピック: セキュリティ・パッチのダウンロードと脆弱性についてのOracleへの連絡

A.2.2 Oracle Databaseの脆弱性に関するOracleのセキュリティ窓口への連絡

Oracle Databaseの脆弱性に関してOracleのセキュリティ窓口に連絡できます。

- Oracleのセキュリティ窓口への連絡には、次のいずれかの方法を使用します。
  - OracleユーザーまたはOracleパートナであるユーザーがOracle製品の潜在的なセキュリティ上の脆弱性を発見した場合は、My Oracle Supportを使用してサービス・リクエストを発行してください。
  - 製品のバージョンやプラットフォームも含めた問題の詳細を、スクリプトと例を添えて、電子メールで`secalert_us@oracle.com`に送信してください。オラクル社にお問合せの際は、弊社の暗号キーを使用して、電子メールを暗号化することをお薦めします。

A.3 ユーザー・アカウントと権限の保護に関するガイドライン

Oracleには、ユーザー・アカウントと権限を保護するためのガイドラインがあります。

1. デフォルト(事前定義)のユーザー・アカウントをロックし、期限切れにする。

   Oracle Databaseをインストールすると、いくつかのデフォルトのデータベース・ユーザー・アカウントがインストールされます。Database Configuration Assistantは、データベースが正常にインストールされると、デフォルトのデータベース・ユーザー・アカウントの大半を自動的にロックし、期限切れにします。

   Database Configuration Assistantを利用せずに手動でOracle Databaseをインストールした場合は、データベース・サーバーが正常にインストールされたときに、デフォルトのデータベース・ユーザーは一切ロックされません。あるいは、以前のリリースのOracle Databaseからアップグレードした場合、以前のリリースからのデフォルト・アカウントが設定されていることがあります。これらのデータベース・ユーザーをデフォルト状態のままにしておくと、それらのユーザー・アカウントを悪用してデータが不正にアクセスされたり、データベース操作が妨害される可能性があります。

   デフォルトのデータベース・ユーザー・アカウントはすべてロックし、期限切れにする必要があります。Oracle Databaseには、この操作を実行するためのSQL文が用意されています。たとえば:

   ```
   ALTER USER ANONYMOUS PASSWORD EXPIRE ACCOUNT LOCK;
   ```

   初期インストールの後に、製品およびコンポーネントを追加インストールすると、デフォルトのデータベース・アカウントがさらに作成されます。Database Configuration Assistantは、追加作成されたデータベースのすべてのユーザー・アカウントを自動的にロックし、期限切れにします。定期的にアクセスする必要があるアカウントのロックのみを解除し、解除した各アカウントに強固で有効なパスワードを割り当てます。この操作を実行するためのSQLおよびパスワード管理が用意されています。

   なんらかの理由で、OPEN以外の状態にあるデフォルトのデータベース・ユーザー・アカウントが必要な場合、データベース管理者(DBA)は、そのアカウントのロックを解除し、安全性の高い新しいパスワードを設定してアクティブにする必要があります。

   なんらかの理由で、OPEN以外の状態にあるデフォルトのデータベース・ユーザー・アカウントが必要な場合、データベース管理者(DBA)は、そのアカウントのロックを解除し、安全性の高い新しいパスワードを設定してアクティブにできます。

   Oracle Enterprise Managerアカウントの保護

   Oracle Enterprise Managerをインストールする場合は、Oracle Enterprise Managerを集中管理用に構成しないかぎり、SYSMANとDBSNMPアカウントはOPENとなります。この場合、SYSMANアカウント(存在する場合)はロックされます。

   Oracle Enterprise Managerをインストールしない場合は、SYSとSYSTEMアカウントのみがOPENとなります。Database Configuration Assistantは、他のすべてのアカウント(SYSMANとDBSNMPを含む)をロックし、期限切れにします。

2. ユーザーが、SQL文でNOLOGGING句を使用できないようにする。

   いくつかのSQL文で、ユーザーはオプションとしてNOLOGGING句を指定できます。このオプションを指定すると、データベース操作がオンラインREDOログ・ファイルに記録されません。ユーザーがこの句を指定しても、REDOはオンラインREDOログ・ファイルに書き込まれます。ただし、このレコードにはデータが関連付けられていません。このため、NOLOGGINGを使用すると、不正コードが入力されて、監査証跡に記録されずに実行される可能性があります。

3. 「最低限の権限」原則を実践する。

次のガイドラインに従うことをお薦めします。

a. 必要な権限のみを付与する。

データベース・ユーザーまたはロールに、必要以上の権限を付与しないでください。(可能な場合、ユーザーではなくロールに権限を付与してください。)つまり、最小権限の原則とは、ユーザーに、効率的かつ簡潔に作業を行うために実際に必要な権限のみを与えることです。

この原則を実装するには、次の項目について可能なかぎり制限します。

- データベース・ユーザーに付与するSYSTEMおよびOBJECT 権限の数。
- SYS権限によってデータベースに接続できるユーザーの数。
- DROP ANY TABLE権限などのANY権限を付与する対象のユーザー数。たとえば、通常、DBA権限のないユーザーにCREATE ANY TABLE権限を付与する必要はありません。
- データベース・オブジェクトを(TRUNCATE TABLE、DELETE TABLE、DROP TABLE文などで)作成、変更または削除を実行できるユーザーの数。

b. CREATE ANY EDITIONおよびDROP ANY EDITION権限の付与を制限する。

オブジェクトの追加バージョンを維持するために、エディションではデータベース内のリソースおよびディスク領域の使用量を増やすことができます。CREATE ANY EDITIONおよびDROP ANY EDITION権限は、アップグレードの実行を担当する信頼できるユーザーにのみ付与します。

c. ユーザーに付与したSELECTオブジェクト権限およびSELECT ANY TABLEシステム権限を再評価する。

ユーザーが表、ビュー、マテリアライズド・ビューおよびシノニムの問合せのみを行えるように制限する場合、READオブジェクト権限を付与し、信頼できるユーザーにのみREAD ANY TABLEシステム権限を付与します。問合せ操作の実行以外に、ユーザーが排他モードでの表のロックまたはSELECT ... FOR UPDATE文を実行できるようにする場合、ユーザーにSELECTオブジェクト権限を付与し、信頼できるユーザーにのみSELECT ANY TABLEシステム権限を付与します。

d. CREATE ANY JOB、BECOME USER、EXP_FULL_DATABASEおよびIMP_FULL_DATABASE権限を制限する。CREATE DIRECTORYおよびCREATE ANY DIRECTORY権限の付与も制限してください。

これらは強力なセキュリティ関連権限です。これらの権限は、必要とするユーザーにのみ付与してください。

e. Oracle Data PumpおよびDBMS_WORKLOAD_CAPTUREとDBMS_WORKLOAD_REPLAYパッケージのユーザーに対してはBECOME USER権限を制限します。

BECOME USER権限は次のサブシステムに対してのみ使用します。

- Oracle Data Pump Importユーティリティimpdpおよびimpで、第三者が直接実行できない操作(オブジェクト権限を付与するようなオブジェクトのロード)を実行する場合に別のユーザーIDを利用できます。Oracle Database Vault環境では、BECOME USERの権限付与に影響する複数レベルの必須認可がDatabase Vaultに設定されています。
- DBMS_WORKLOAD_CAPTUREおよびDBMS_WORKLOAD_REPLAYのPL/SQLパッケージ。これらのパッケージを使用する必要があるユーザーにはこの権限を付与する必要があります。

これらのいずれかのサブシステム(たとえば、PL/SQLコードの静的参照)を呼び出す際にAUTHID

CURRENT_USER句を使用する場合は、CURRENT_USERにBECOME USER権限が直接付与されているか、ロールを介して付与されていることを確認してください。

f. ライブラリ関連の権限を信頼できるユーザーのみに制限する。

CREATE LIBRARY、CREATE ANY LIBRARY、ALTER ANY LIBRARY、およびEXECUTE ANY LIBRARY権限と、EXECUTE ON library_nameの付与によって、ユーザーに大きい力が与えられます。ライブラリへのPL/SQLインタフェースを作成する場合は、PL/SQLインタフェースにEXECUTE権限を付与するのみにしてください。基礎となるライブラリにEXECUTEを付与しないでください。ライブラリへのPL/SQLインタフェースを作成するためには、ライブラリに対するEXECUTE権限が必要です。しかしユーザーは、自分自身のスキーマで作成するライブラリに対して暗黙的にこの権限を持っています。EXECUTE ON library_nameの明示的付与が必要になることはほとんどありません。これらの権限の明示的付与は、信頼できるユーザーに対してのみ行ってください。決してPUBLICロールに対して付与しないでください。

g. 制限シノニム関連の権限を信頼できるユーザーのみに制限する。

CREATE PUBLIC SYNONYMおよびDROP PUBLIC SYNONYMシステム権限によって、これらのユーザーに大きい力が与えられます。信頼できないかぎり、これらの権限をユーザーに付与しないでください。

h. SYSスキーマが所有するオブジェクトに、管理者以外のユーザーがアクセスできないようにする。

ユーザーが表の行やSYSスキーマのスキーマ・オブジェクトを変更できないようにしてください。変更されると、データ整合性が損われる危険があります。DROP TABLE、TRUNCATE TABLE、DELETE、INSERTなどの文、または類似したオブジェクト変更文のSYSオブジェクトに対する使用を、強い権限を持つ管理ユーザーのみに制限してください。

i. ランタイム機能の権限を制限する。

多くのOracle Database製品は、Oracle Java Virtual Machine(OJVM)などのランタイム機能を使用しています。データベース・ランタイム機能に対してすべての権限を割り当てないでください。かわりに、データベース外部のファイルやパッケージを実行する機能については、特定の権限を明示的なドキュメント・ルート・ファイル・パスに設定してください。

個別ファイルが指定された無防備なランタイム・コールの例:

```
call dbms_java.grant_permission('wsmith',
'SYS:java.io.FilePermission','<<ALL FILES>>','read');
```

かわりにディレクトリ・パスが指定された、適切な(安全性の高い)ランタイム・コールの例:

```
call dbms_java.grant_permission('wsmith',
'SYS:java.io.FilePermission','<<actual directory path>>','read');
```

4. 次のビューまたはロールからアクセス権を取り消す。

a. SYSとDBAアカウント以外のすべてのユーザーからSYS.USER_HISTORY$表を取り消します。

b. 通常のアプリケーション・アカウントからRESOURCEロールを取り消します。

c. 通常のアプリケーション・アカウントからCONNECTロールを取り消します。

d. DBAロールが不要なユーザーから、このロールを取り消します。

5. ロールに対してのみ権限を付与する。

個々のユーザーではなくロールに権限を付与すると、権限の管理および追跡が容易になります。

6. プロキシ・アカウント(プロキシ認可の場合)の権限をCREATE SESSIONのみに制限する。
7. セキュア・アプリケーション・ロールを使用して、アプリケーション・コードによって有効化するロールを保護する。

   セキュア・アプリケーション・ロールでは、ユーザーがアプリケーションにログインできるかどうかを判断する一連の条件を、PL/SQLパッケージ内で定義できます。ユーザーはセキュア・アプリケーション・ロールではパスワードを使用する必要がありません。

   ロールがアプリケーション内で使用可能または使用禁止になるのを防ぐ別の方法は、ロールのパスワードを使用することです。この方法では、ユーザーが、データベースにSQL(アプリケーションではなく)で直接アクセスして、ロールに関連付けられている権限を使用することを防ぎます。ただし、別の一連のパスワードを管理する必要がないように、セキュア・アプリケーション・ロールの使用をお薦めします。
8. 権限キャプチャを作成して、過度に付与されている権限を検索する。権限分析は、ユーザーおよびアプリケーションが使用する権限を取得して、これらを簡単に分析できる形式で提示します。ここで、必要に応じて不要な権限を取り消すことができます。
9. 次の権限が、それらを必要としているユーザーとロールのみに付与されていることを監視する。

   デフォルトでは、次の権限がOracle Databaseによって監査されます。

   a. ALTER SYSTEM
   b. AUDIT SYSTEM
   c. CREATE EXTERNAL JOB

   次の権限も監査することをお薦めします。

   d. ALL PRIVILEGES(BECOME USER、CREATE LIBRARY、CREATE PROCEDUREなどの権限を含む)
   e. DBMS_BACKUP_RESTOREパッケージ
   f. DBMS_SYS_SQLに対するEXECUTE権限
   g. SELECT ANY TABLE
   h. PERFSTAT.STATS$SQLTEXTに対するSELECT権限
   i. PERFSTAT.STATS$SQL_SUMMARYに対するSELECT権限
   j. SYS.SOURCE$に対するSELECT権限
   k. WITH ADMIN句付きの権限
   l. WITH GRANT句付きの権限
   m. CREATEキーワード付きの権限
10. 次のデータ・ディクショナリ・ビューを使用して、データベースへのユーザー・アクセスに関する情報を検索する。

    a. DBA_*
    b. DBA_ROLES
    c. DBA_SYS_PRIVS
    d. DBA_ROLE_PRIVS
    e. DBA_TAB_PRIVS
    f. DBA_AUDIT_TRAIL(標準監査が使用可能な場合)

g. DBA_FGA_AUDIT_TRAIL(ファイングレイン監査が使用可能な場合)

関連トピック

- Oracle Database Vaultの管理者ガイド
- 権限分析の実行による権限使用の特定

親トピック: Oracle Databaseの安全性の維持

A.4 ロールの保護に関するガイドライン

Oracleには、ロール管理用のガイドラインがあります。

1. ロールは、そのロールの権限すべてを必要とするユーザーにのみ付与する。

   ロール(権限のグループ)は、ユーザーに許可を素早く簡単に付与する場合に便利です。Oracleで定義されているロールを使用することもできますが、必要な権限のみを含む独自のロールを作成すると、より継続的な制御が可能になります。Oracle Database定義ロールの権限は変更または削除される場合があります。たとえばCONNECTロールの場合、現在持っている権限はCREATE SESSION権限のみです。以前は他に8個の権限がありました。

   定義したロールには担当業務を反映する権限のみが含まれていることを確認してください。アプリケーション・ユーザーが既存のロールに組み込まれている権限のすべては必要としていない場合は、適切な権限のみを含む異なるロールのセットを適用してください。または、さらに権限が制限されているロールを作成して割り当てます。

   たとえば、ユーザーSCOTTはよく知られているアカウントで、侵入されやすい可能性があるため、このユーザーの権限を厳密に制限する必要があります。CREATE DBLINK権限ではあるデータベースから別のデータベースへのアクセスが許可されるため、SCOTTに対するこの権限を削除します。次に、このユーザーのロール全体を削除します。これは、ロールによって付与される権限は、個別に削除できないためです。必要な権限のみを含む独自のロールを再作成し、新しいロールをユーザーに付与します。同様に、セキュリティを高めるために、CREATE DBLINK権限を必要としないすべてのユーザーからこの権限を削除します。

2. アプリケーション開発者にはユーザー・ロールを付与しない。

   ストアド・プログラム構文内のスキーマ・オブジェクトにアクセスする権限は直接付与される必要があるため、ロールはアプリケーション開発者の使用を目的としていません。実行者権限プロシージャを除くストアド・プロシージャ内ではロールが有効でないことに注意してください。詳細は、PL/SQLブロックでのロールの機能を参照してください。

3. Oracle Databaseのインストールごとに固有のロールを作成して割り当てる。

   これにより、組織でロールおよび権限を詳細に制御できます。また、現在はCREATE SESSION権限のみとなっているCONNECTなどのOracle Database定義のロールを、Oracle Databaseで変更または削除した場合の調整が不要となります。以前は他にも8つの権限がありました。

4. エンタープライズ・ユーザーに対してグローバル・ロールを作成する。

   グローバル・ロールは、Oracle Internet Directoryなどのエンタープライズ・ディレクトリ・サービスで管理されます。グローバル・ロールの詳細は次の各項を参照してください。

   - ユーザーのグローバル認証とグローバル認可
   - エンタープライズ・ディレクトリ・サービスによるグローバル・ロールの認可
   - Oracle Databaseエンタープライズ・ユーザー・セキュリティ管理者ガイド

親トピック: Oracle Databaseの安全性の維持

A.5 パスワードの保護に関するガイドライン

オラクル社では、様々な状況でのパスワードの保護についてガイドラインを提供しています。

ユーザー・アカウントを作成すると、そのユーザーに対してデフォルトのパスワード・ポリシーが割り当てられます。このパスワード・ポリシーには、パスワードの作成方法(最小文字数や期限切れの時期など)についてのルールが定義されています。パスワードは、パスワード・ポリシーを使用することで強化できます。パスワードを保護するための別の方法については、「パスワード保護の構成」も参照してください。

パスワードをさらに強化するには、次のガイドラインに従います。

1. パスワードを慎重に選択する。

   パスワードの最低要件については、「パスワードの最低要件」を参照してください。パスワードを作成または変更する際には、次の追加のガイドラインに従います。

   - パスワードは、長さを12から30バイトまでにし、英字と数字の両方を含むようにします。
   - パスワードには、数値、大文字および小文字を少なくとも1文字ずつ含めます。
   - パスワードには、大/小文字と特殊文字を混在させて使用します。(「パスワードのセキュリティへの脅威からの12Cパスワード・バージョンによる保護」を参照してください。)
   - パスワードにはマルチバイト・キャラクタを含めることができますが、共通ユーザーまたは共通ロールのパスワードに含めることはできません。
   - パスワードの文字にはデータベース文字セットを使用します。このセットには、アンダースコア(_)、ドル記号($)および番号記号(#)の各文字があります。
   - 次のパスワードは二重引用符で囲む必要があります。
     - マルチバイト・キャラクタを含むパスワード
     - 数字または特殊文字で始まり、アルファベットを含むパスワード。たとえば:

       ```
       "123abc"
       "#abc"
       "123dc$"
       ```

     - アルファベット、数字および特殊文字以外の文字を含むパスワード。たとえば:

       ```
       "abc>"
       "abc@",
       " "
       ```

   - 次のパスワードは二重引用符で囲む必要はありません。
     - アルファベット(aからz、AからZ)で始まり、数字(0から9)または特殊文字($、#、_)を含むパスワード。たとえば:

       ```
       abc123
       ab23a
       ab$#_
       ```

     - 数値のみを含むパスワード
     - アルファベット(aからz、AからZ)のみを含むパスワード。

- パスワードには二重引用符を使用しないでください。
- 意味のある単語のみで構成されるパスワードを使用しないでください。

2. 短い覚えやすいパスワードをより長く複雑なパスワードにするには、覚えやすい文の単語の先頭文字を使用してパスワードを作成します。

   たとえば、"I usually work until 6:00 almost every day of the week"であれば、`Iuwu6aedotw`となります。

3. 十分複雑なパスワードを使用するようにする。

   Oracle Databaseには、パスワードの複雑度が十分であるかどうかをチェックするためのパスワード複雑度検証ルーチン(PL/SQLスクリプト`utlpwdmg.sql`)が用意されています。理想的には、`utlpwdmg.sql`スクリプトを編集して、パスワードの安全性を高めます。パスワードのチェックに使用できるサンプル・ルーチンの概要については、「パスワードの複雑度検証について」も参照してください。

4. 非マルチテナント環境またはPDBでは、パスワードにマルチバイト文字を使用する必要がある場合は、認証が正しく機能するように、データベース文字セットがマルチバイト文字セットとして構成されていることを確認してください。

   マルチバイト・キャラクタはシングルバイト・キャラクタよりも多くのバイトを使用するため1バイト当たりの不規則性がより低くなりやすいということに注意してください。パスワードにおいては、現在は最大長が30バイトに決められているため、不規則性が高まるよう、マルチバイト・キャラクタを使用している場合でもシングルバイト・キャラクタをいくつか含めることをお薦めします。

   共通ユーザーおよび共通ロールのパスワードにはマルチバイト・パスワードは使用できません。

5. ユーザー・プロファイルまたはデフォルト・プロファイルに対してパスワード複雑度ファンクションを関連付けます。

   `CREATE PROFILE`および`ALTER PROFILE`文の`PASSWORD_VERIFY_FUNCTION`句は、パスワードの複雑度検証関数をユーザー・プロファイルまたはデフォルト・プロファイルと関連付けます。パスワード複雑度ファンクションでは、ユーザーがそのサイトに固有のガイドラインを使用して強力なパスワードを作成しているかどうかを確認します。また、パスワードの複雑度検証関数を設定するには、ユーザーが(`ALTER USER`システム権限を使用せずに)自分のパスワードを変更して、古いパスワードと新しいパスワードの両方を指定する必要があります。独自のパスワード複雑度ファンクションを作成したり、Oracle Databaseが提供するパスワード複雑度ファンクションを使用できます。

   詳細は、「パスワードの複雑度の管理」を参照してください。

6. デフォルトのユーザー・パスワードを変更する。

   Oracle Databaseは、事前定義のデフォルト・ユーザー・アカウントのセットとともにインストールされます。セキュリティは、デフォルトのデータベース・ユーザー・アカウントがインストール後もデフォルト・パスワードを使用している場合に最も崩壊しやすくなります。ユーザー・アカウント`SCOTT`はよく知られているアカウントで侵入されやすい可能性があるため、これが特に当てはまります。Oracle Databaseでは、デフォルトのアカウントはロックされ、パスワードは期限切れの状態でインストールされますが、以前のリリースからのアップグレードの場合は、デフォルトのパスワードを使用しているアカウントが存在している場合があります。

   デフォルトのパスワードが設定されているユーザー・アカウントを検索するには、`DBA_USERS_WITH_DEFPWD`データ・ディクショナリ・ビューを問い合せます。詳細は、「デフォルト・パスワードが設定されているユーザー・アカウントの検索」を参照してください。

7. 管理ユーザーのデフォルト・パスワードを変更する。

   `SYS`、`SYSTEM`、`SYSMAN`および`DBSNMP`の管理アカウントには、同じパスワードまたは異なるパスワードを使用でき

ます。それぞれに対して異なるパスワードを使用することをお薦めします。つまり、すべてのOracle環境(本番またはテスト)において、これらの管理アカウントに強固で安全性の高い、個別のパスワードを割り当てます。Database Configuration Assistantを使用して新規データベースを作成する場合は、SYSおよびSYSTEMアカウントにパスワードを入力して、デフォルトのパスワードCHANGE_ON_INSTALLおよびMANAGERを使用できないようにします

同様に、本番環境では、SYSMANおよびDBSNMPも含めて、管理アカウントにはデフォルトのパスワードを使用しないでください。

8. パスワード管理を徹底する。

   基本的なパスワード管理ルール(パスワードの長さ、履歴および複雑度など)をすべてのユーザー・パスワードに適用します。Oracle Databaseには、デフォルト・プロファイルで使用可能なパスワード・ポリシーがあります。この項のガイドライン1には、これらのパスワード・ポリシーがリストされています。

   ユーザー・アカウントの情報を検索するには、DBA_USERSビューを問い合せます。DBA_USERSビューのPASSWORD列は、パスワードがグローバル、外部またはNULLのいずれであるかを示します。DBA_USERSビューには、ユーザー・アカウントの状態、アカウントがロックされているかどうか、パスワードのバージョンなどの有用な情報が表示されます。

   また、可能な場合は、Oracle厳密認証を、ネットワーク認証サービス(Kerberosなど)、トークン・カード、スマートカードまたはX.509証明書と一緒に使用することもお薦めします。これらのサービスを使用すると、ユーザーの厳密な認証が可能になるため、Oracle Databaseを不正なアクセスから保護できます。

9. Oracle表にはユーザー・パスワードをクリアテキストで格納しない。

   セキュリティを強化するために、Oracle表には、ユーザー・パスワードをクリアテキスト(判読可能なテキスト)で格納しないでください。この問題は、安全性の高い外部パスワード・ストアを使用してOracleウォレット内のパスワードを暗号化することで解決できます。(Oracleウォレットは、認証および署名用資格証明を格納する安全性の高いソフトウェア・コンテナです。)詳細は、「パスワード資格証明用の安全性の高い外部パスワード・ストアの管理」を参照してください。

   ユーザー・アカウントのパスワードを作成または変更すると、Oracle Databaseでは、そのパスワードの暗号化ハッシュまたはダイジェストが自動的に作成されます。DBA_USERSビューを問い合せてユーザー・アカウントの情報を検索する場合、PASSWORD列のデータはユーザー・パスワードがグローバル、外部またはNULLのいずれかであるかを示します。DBA_USERSビューにはPASSWORD_VERSIONSという列もあり、この列には、そのユーザーのパスワードのために存在する暗号化ハッシュのタイプ(11Gまたは12C)が一覧表示されます。これらの略称が対応している暗号化アルゴリズムについては、Oracle DatabaseリファレンスのDBA_USERSに関する項を参照してください。

10. ユーザーがXDB認証もHTTPダイジェスト認証も使用しない場合は、HTTPベリファイアを無効にします。

    HTTPベリファイアは、XDB認証およびHTTPダイジェスト認証にのみ使用されます。ユーザーがXDB認証もHTTPダイジェスト認証も使用しない場合は、ユーザーのベリファイア・リストからHTTPベリファイアを安全に削除できます。ユーザーのHTTPベリファイアを削除するには、次の文を実行します。

```
ALTER USER username DIGEST DISABLE;
```

親トピック: Oracle Databaseの安全性の維持

A.6 データの保護に関するガイドライン

Oracleには、システムでデータを保護するためのガイドラインがあります。

1. オペレーティング・システムのアクセスを制限する。

次のガイドラインに従ってください。

- オペレーティング・システムのユーザー数を制限します。
- Oracle Databaseが稼働しているホスト・コンピュータ上のオペレーティング・システム・アカウントの権限(管理、ルート権限またはデータベース管理)を、ユーザーによる業務の遂行に必要な最小限の権限に制限してください。
- Oracle Databaseホーム(インストール)ディレクトリやその内容について、デフォルトのファイルやディレクトリのアクセス権を変更できる権限を制限します。オラクル社から特に指示がないかぎり、権限を持つオペレーティング・システム・ユーザーとOracle所有者は、これらのアクセス権を変更しないでください。
- シンボリック・リンクを制限します。データベースへのパスやファイルを指定する場合は、そのファイルやパスのどの部分も、信頼のおけないユーザーによる変更が不可能であることを確認します。ファイル、およびパスのすべての構成要素は、データベース管理者、またはrootなどの信頼できるアカウントが所有する必要があります。

  この推奨事項は、ログ・ファイル、トレース・ファイル、外部表、BFILEデータ型など、あらゆるタイプのファイルに適用されます。

2. 機密性の高いデータと、データベース・ファイルが格納された全バックアップ・メディアを暗号化する。

   一般的な法令順守要件に従って、クレジット・カード番号とパスワードなどの機密性の高いデータを暗号化する必要があります。データベースから機密性の高いデータを削除した場合、暗号化されたデータはデータ・ブロック、オペレーティング・システム・ファイルまたはディスク上のセクターに残りません。

   多くの場合、透過的データ暗号化を使用して機密性の高いデータを暗号化する必要があります。詳細は、Oracle Database Advanced Securityガイドを参照してください。また、データを暗号化すべきでない場合は、暗号化で解決しないセキュリティの問題を参照してください。

3. LinuxおよびUNIXシステムでのOracle Automatic Storage Management(Oracle ASM)環境の場合、Oracle ASM File Access Controlを使用してOracle ASMディスク・グループへのアクセスを制御します。

   様々なオペレーティング・システム・ユーザーおよびグループをOracle Databaseインストールで使用する場合は、Oracle ASM File Access Controlを構成して、Oracle ASMディスク・グループ内のファイルへのアクセスを、認可されたユーザーのみに制限できます。たとえば、データベース管理者がアクセスできるのは、データベース管理者が管理するデータベースのデータ・ファイルのみになります。この管理者は、他のデータベースに属している(使用される)データ・ファイルは参照することも上書きすることもできなくなります。

   Oracle ASM File Access Controlのディスク・グループ管理の詳細は、『Oracle Automatic Storage Management管理者ガイド』を参照してください。複数のソフトウェア所有者に必要な様々な権限の詳細は、『Oracle Automatic Storage Management管理者ガイド』も参照してください。

親トピック: Oracle Databaseの安全性の維持

A.7 ORACLE_LOADERアクセス・ドライバの保護に関するガイドライン

Oracleには、ORACLE_LOADERアクセス・ドライバを保護するためのガイドラインがあります。

1. アクセス・ドライバ・プリプロセッサを格納する、別個のオペレーティング・システム・ディレクトリを作成します。Oracle Databaseの各ユーザーが異なるプリプロセッサを実行する場合、オペレーティング・システム・マネージャは複数のディレクトリを作成する必要がある場合があります。特定のユーザーに対して、別のプリプロセッサへのアクセスを許可しながら、あるプリプロセッサの使用を禁止するには、そのプリプロセッサを別個のディレクトリに配置します。すべてのユーザーに同

等のアクセス権を付与する必要がある場合は、プリプロセッサをまとめて1つのディレクトリに配置できます。これらのオペレーティング・システム・ディレクトリを作成した後、SQL*Plusで各ディレクトリのディレクトリ・オブジェクトを作成できます。

2. オペレーティング・システム・ユーザーORACLEに、アクセス・ドライバ・プリプロセッサを実行するための適切なオペレーティング・システム権限を付与します。また、プリプロセッサ・プログラムを、プリプロセッサ・プログラムの管理担当ユーザー以外のオペレーティング・システム・ユーザーによるWRITEアクセスから保護します。
3. ディレクトリ・オブジェクト内のプリプロセッサ・プログラムを実行する各ユーザーに、EXECUTE権限を付与します。このユーザーには、ディレクトリ・オブジェクトに対するWRITE権限を付与しないでください。ユーザーにディレクトリ・オブジェクトに対するEXECUTE権限とWRITE権限の両方を付与することはできません。
4. プリプロセッサを含むディレクトリ・オブジェクトを管理するユーザーに、慎重にWRITE権限を付与します。これにより、データベース・ユーザーが誤ってまたは意図的にプリプロセッサ・プログラムを上書きするのを防ぐことができます。
5. 外部表に必要なすべてのデータ・ファイルに対して、別個のオペレーティング・システム・ディレクトリおよびディレクトリ・オブジェクトを作成します。これらのディレクトリおよびディレクトリ・オブジェクトは、アクセス・ディレクトリ・プリプロセッサが使用するディレクトリおよびディレクトリ・オブジェクトとは別個であることが必要です。

   オペレーティング・システム・マネージャとともに作業して、このディレクトリへのアクセス権が適切なオペレーティング・システム・ユーザーのみに付与されていることを確認します。ORACLEオペレーティング・システム・ユーザーに、データベース・ユーザーに付与されたREAD権限を持つディレクトリ・オブジェクトを含むすべてのディレクトリへのREADアクセス権を付与します。同様に、ORACLEオペレーティング・システム・ユーザーに、データベース・ユーザーに付与されたWRITE権限を持つすべてのディレクトリへのWRITEアクセス権を付与します。
6. アクセス・ドライバが生成するあらゆるファイルに対して、別々のオペレーティング・システム・ディレクトリおよびディレクトリ・オブジェクトを作成します。これには、ログ・ファイル、不良ファイル、および廃棄ファイルが含まれます。オペレーティング・システム・マネージャとともに、このディレクトリとディレクトリ・オブジェクトが、ガイドライン5で説明されているように適切な保護を受けていることを確認してください。データ・ファイル内の問題を解決するとき、データベース・ユーザーはこれらのファイルへのアクセスが必要になる場合があるため、このユーザーがこれらのファイルを読み取る方法をオペレーティング・システム・マネージャとともに決める必要があります。
7. CREATE ANY DIRECTORY権限とDROP ANY DIRECTORY権限を慎重に付与します。これらの権限を付与されたユーザーおよびDBAロールを付与されたユーザーは、すべてのディレクトリ・オブジェクトへの完全なアクセス権を持ちます。
8. DROP ANY DIRECTORY権限の監査を検討します。権限の監査の詳細は、システム権限の監査を参照してください。
9. ディレクトリ・オブジェクトの監査を検討します。詳細は、オブジェクト・アクションの監査を参照してください。

関連項目:

ORACLE_DATAPUMPアクセス・ドライバの詳細は、Oracle Databaseユーティリティを参照してください

親トピック: Oracle Databaseの安全性の維持

A.8 データベースのインストールと構成の保護に関するガイドライン

Oracleには、データベースのインストールと構成を保護するためのガイドラインがあります。

安全性を高めるために、Oracle Databaseのデフォルト構成が変更されました。この項の推奨事項には、この新規のデフォル

ト構成が追加されています。

1. UNIXシステムの場合は、Oracle Databaseのインストールを開始する前に、Oracle所有者アカウントのumask値が022であることを確認する。

   LinuxおよびUNIXシステム上のOracle Databaseを管理する方法の詳細は、『Oracle Database管理者リファレンス for Linux and UNIX-Based Operating Systems』を参照してください。

2. 必要なもののみをインストールする。

   オプションと製品: Oracle DatabaseのCDパックには、データベース・サーバーだけでなく、製品およびオプションが含まれています。必要な場合にかぎり、製品およびオプションを追加してインストールします。カスタム・インストール機能を使用して必要のない製品のインストールを防止するか、もしくは通常のインストールを実行してから不要な製品およびオプションを削除してください。使用していない製品およびオプションは、維持する必要はありません。製品およびオプションは、必要に応じて簡単にインストールできます。

   サンプル・スキーマ: Oracle Databaseには、例を示すための共通のプラットフォームを提供するサンプル・スキーマが用意されています。このサンプル・スキーマは、データベースを本番環境で使用する場合はインストールしないでください。サンプル・スキーマをテスト・データベースにインストールした場合は、本番に移行する前に、そのサンプル・スキーマのアカウントを削除または再びロックしてください。サンプル・スキーマの詳細は、『Oracle Databaseサンプル・スキーマ』を参照してください。

3. インストール時に、パスワードの入力を求めるプロンプトが表示された場合は、安全性の高いパスワードを作成する。

   パスワードの保護に関するガイドラインのガイドライン1、6および7に従います。

4. インストール直後に、デフォルトのユーザー・アカウントをロックし、期限切れにする。

   ユーザー・アカウントと権限の保護に関するガイドラインのガイドライン1を参照してください。

親トピック: Oracle Databaseの安全性の維持

A.9 ネットワークの保護に関するガイドライン

ネットワーク通信のセキュリティは、クライアント、リスナー、および完全な保護を確保するためのネットワーク・ガイドラインを使用することで改善されます。

- クライアント接続のセキュリティ
  クライアントを厳密に認証し、接続に対する暗号化を構成して、厳密認証を使用すると、クライアント接続が強化されます。
- ネットワーク接続のセキュリティ
  不適切なアクセスまたは変更からネットワークとその通信を保護することは、ネットワーク・セキュリティの最重要点です。
- Transport Layer Security接続のセキュリティ
  Oracleには、Transport Layer Security (TLS)を保護するためのガイドラインがあります。

親トピック: Oracle Databaseの安全性の維持

A.9.1 クライアント接続のセキュリティ

クライアントを厳密に認証し、接続に対する暗号化を構成して、厳密認証を使用すると、クライアント接続が強化されます。

クライアント・コンピュータの認証には問題が多いため、通常は、かわりにユーザー認証が実施されます。このアプローチは、クライアント・システムで、偽造されたIPアドレス、ハッキングされたオペレーティング・システムまたはアプリケーション、および偽造または

盗用されたクライアント・システムIDが使用される問題を回避します。

また、次のガイドラインによって、クライアント接続のセキュリティが向上します。

1. 効果的なアクセス制御と厳密なクライアント認証を規定する。

   デフォルトでは、Oracleで許可されるオペレーティング・システム認証ログインは、保護された接続のみを介したログインであるため、Oracle Netおよび共有サーバー構成を使用したログインは含まれません。このデフォルトの制限によって、ネットワーク接続を介したユーザーが阻止されます。

   初期化パラメータREMOTE_OS_AUTHENT をTRUEに設定すると、データベースはセキュアでない接続を介して受信したクライアントのオペレーティング・システムのユーザー名を受け入れ、このユーザー名をアカウント・アクセスに使用します。PCなどのクライアントは、オペレーティング・システムの認証を正しく実行していない場合があるため、この機能を使用するとセキュリティが非常に低下します。

   デフォルトの設定REMOTE_OS_AUTHENT = FALSEを使用すると、安全性の高い構成となり、Oracleデータベースに接続するクライアントがサーバーベースで適切に認証されます。REMOTE_OS_AUTHENTは、Oracle Databaseリリース11g(11.1)では非推奨となっており、下位互換性のためにのみ保持されている点に注意してください。

   したがって、REMOTE_OS_AUTHENT初期化パラメータのデフォルト設定FALSEは変更しないでください。

   このパラメータをFALSEに設定しても、ユーザーがリモートから接続できなくなるわけではありません。クライアントがすでに認証されていたとしても、データベースにより標準の認証プロセスが適用されるだけです。

   REMOTE_OS_AUTHENTパラメータは、Oracle Database 11g リリース1(11.1)では非推奨となっており、下位互換性のためにのみ保持されている点に注意してください。

2. 暗号化を使用するように接続を構成する。

   Oracleネイティブ・ネットワーク暗号化を使用すると、傍受が困難となります。

3. 強力な認証を設定する。

   Kerberosおよび公開キー・インフラストラクチャ(PKI)の使用方法の詳細は、Kerberos認証の構成を参照してください。

4. Oracle Data Guard環境で、ADG_ACCOUNT_INFO_TRACKING初期化パラメータを設定します。

   ADG_ACCOUNT_INFO_TRACKINGパラメータは、Oracle Active Data Guardスタンバイ・データベースでのログイン試行を制御します。これにより、Oracle Databaseの本番環境とすべてのActive Data Guardスタンバイ・データベースにおいてログイン侵入攻撃に対するセキュリティが強化されます。次のいずれかの設定を使用します。

   - LOCAL(デフォルト)では、既存の動作(スタンバイ・データベースのインメモリー・ビューでユーザー・アカウント情報のローカル・コピーを保持)が適用されます。この設定では、データベース単位でローカルのログイン失敗のみが追跡されます。ログイン失敗の最大回数に達すると、ログインが拒否されます。
   - GLOBALでは、すべてのData Guardプライマリ・データベースおよびスタンバイ・データベースにおいてユーザー・アカウント情報の単一グローバル・コピーを維持することで、ログインのセキュリティが強化されます。Data Guard環境のすべてのデータベースでのログイン失敗回数が、最大回数の対象としてカウントされます。この数に達すると、どこでもログインがアクセス拒否されます。

   ADG_ACCOUNT_INFO_TRACKINGパラメータについてさらに学習するには、Oracle Databaseリファレンスを参照してください。

親トピック: ネットワークの保護に関するガイドライン

A.9.2 ネットワーク接続のセキュリティ

不適切なアクセスまたは変更からネットワークとその通信を保護することは、ネットワーク・セキュリティの最重要点です。

データが移動するすべてのパスを検討し、各パスおよびノードに対する脅威を評価してください。次に、脅威およびセキュリティが侵害された場合の結果を抑制または排除するステップを実行します。また、監視および監査を実施し、脅威レベルの増加または侵入試行を検出します。

ネットワーク接続の管理には、Oracle Net Managerを使用できます。Net Managerの詳細は、『*Oracle Database Net Services*管理者ガイド』を参照してください。

次の手続きでネットワーク・セキュリティを改善します。

1. リスナーの管理にTransport Layer Security (TLS)を使用する。

   TLSは、証明書(および必要な場合は暗号化)を使用して、安全性の高い認証、認可およびメッセージ機能をサポートし、ユーザーあるいはアプリケーションおよびサーバーが送受信するメッセージを保護できます。

2. リスナーのパスワードおよびサーバー上のlistener.oraファイルへの書込み権限は、管理者が保持するように規定することで、オンライン管理を回避する。

   a. この行を`listener.ora`ファイルに追加するか、またはこの行を変更してください。

   ```
   ADMIN_RESTRICTIONS_LISTENER=ON
   ```

   b. `RELOAD`を使用して構成を再ロードします。

   c. 次のように、アドレス・リストでTCPSプロトコルを最初のエントリにすることで、リスナーの管理にTLSを使用します。

   ```
   LISTENER=
     (DESCRIPTION=
       (ADDRESS_LIST=
         (ADDRESS=
           (PROTOCOL=tcps)
           (HOST = sales.us.example.com)
           (PORT = 8281)))
   ```

   リスナーをリモート管理するために、クライアント・コンピュータの`listener.ora`ファイルにリスナーを定義します。たとえば、リスナーUSER281にリモートでアクセスするには、次の構成を使用します。

   ```
   user281 =
     (DESCRIPTION =
       (ADDRESS =
         (PROTOCOL = tcps)
         (HOST = sales.us.example.com)
         (PORT = 8281))
       )
     )
   ```

3. リスナーのパスワードを設定しない。

   `listener.ora`ファイルにパスワードが設定されていないことを確認します。ローカルのオペレーティング・システム認証によって、リスナー管理が保護されます。パスワードが設定されていない場合、リモートのリスナー管理は使用禁止になります。このため、リスナーのパスワードの総当り攻撃が防止されます。

   リスナーのパスワードは、このリリースでは非推奨となりました。次回のOracle Databaseリリースではサポートされなくなります。

4. ホスト・コンピュータに、複数のNetwork Interface Controller(NIC)カードに関連付けられている複数のIPアドレスがある場合は、特定のIPアドレスに対してリスナーを構成する。

   これによって、リスナーはすべてのIPアドレスをリスニングできます。また、リスナーが特定のIPアドレスをリスニングするように制限できます。このタイプのコンピュータでは、リスナーですべてのIPアドレスをリスニングするのではなく、特定のIPアドレスを指定することをお薦めします。リスナーを特定のIPアドレスに限定することで、侵入者が、リスナー・プロセスからTCPエンド・ポイントを盗むのを防止できます。

5. リスナーの権限を制限して、データベースまたはOracleサーバーのアドレス空間にあるファイルを読取り/書込みできないようにする。

   この制限によって、リスナー(またはエージェントが実行するプロシージャ)によって起動された外部プロシージャ・エージェントは、読取りまたは書込み操作の実行機能を継承しないようになります。この個別のリスナー・プロセスの所有者には、Oracle Databaseをインストールした所有者またはOracle Databaseインスタンスを実行する所有者(デフォルトの所有者である0RACLEなど)を指定しないでください。

   リスナーの外部プロシージャの構成の詳細は、『*Oracle Database Net Services*管理者ガイド』を参照してください。

6. 暗号化を使用して、転送中のデータを保護する。

   厳密認証は、ネットワーク・データの暗号化の保護に役立ちます。

7. ファイアウォールを利用する。

   ファイアウォールを適切に配置および構成することで、データベースへの外部からのアクセスを防止できます。

   a. データベース・サーバーはファイアウォールの内側に配置してください。Oracle Databaseのネットワーク・インフラストラクチャであるOracle Net Services (旧称SQL*Net)は、様々なベンダーの各種ファイアウォールに対するサポートを提供しています。プロキシ対応のファイアウォールでは、Network Associates社のGauntletとAxent社のRaptorをサポートしています。パケット・フィルタ型のファイアウォールではCisco社のPIX Firewall、ステートフル・インスペクション型のファイアウォール(より高機能のパケット・フィルタ型ファイアウォール)ではCheckPoint社のFirewall-1をサポートしています。

   b. ファイアウォールは、保護するネットワークの外側に配置されている必要があります。

   c. 安全性が確認されているプロトコル、アプリケーションまたはクライアント/サーバーのソースのみを受け入れるようにファイアウォールを構成します。

   d. Net8やOracle Connection Managerなどの製品を使用して、データベースへの単一のネットワーク接続を介した複数のクライアント・ネットワーク・セッションの多重化を管理します。これにより送信元、送信先およびホスト名でフィルタ処理できます。この製品を使用すると、物理的に保護された端末または既知のIPアドレスを備えたアプリケーションWebサーバーからの接続のみを受け入れるようにできます。(IPアドレスは偽造可能なため、IPアドレスのみでフィルタ処理した認証では不十分です。)

8. Oracleリスナーの不正な管理を防止する。

   リスナーの詳細は、『*Oracle Database Net Services*管理者ガイド』を参照してください。

9. ネットワークIPアドレスをチェックする。

   Oracle Netの有効なノードの確認セキュリティ機能を利用すると、指定のIPアドレスを持つネットワーク・クライアントからOracleサーバー・プロセスへのアクセスを許可または拒否できます。この機能を使用するには、`sqlnet.ora`構成ファイルの各パラメータを次のように設定します。

```
tcp.validnode_checking = YES
```

```
tcp.excluded_nodes = {list of IP addresses}
tcp.invited_nodes = {list of IP addresses}
```

tcp.validnode_checkingパラメータによって、この機能が有効になります。tcp.excluded_nodesおよびtcp.invited_nodesパラメータは、Oracleリスナーに接続しようとする特定のクライアントのIPアドレスを拒否および使用可能にします。これによって、サービス拒否攻撃を防止できます。

10. ネットワーク・トラフィックを暗号化する。

可能な場合は、Oracleネイティブ・ネットワーク・データ暗号化を使用して、クライアント、データベースおよびアプリケーション・サーバー間のネットワーク・トラフィックを暗号化します。

11. ホスト・オペレーティング・システム(Oracle Databaseがインストールされているシステム)を保護する。

オペレーティング・システムの不要なサービスをすべて使用禁止にして、ホスト・オペレーティング・システムを保護します。UNIXおよびWindowsには様々なオペレーティング・システム・サービスが組み込まれていますが、それらの大部分は、一般的なデプロイメントには不要です。これらのサービスには、FTP、TFTP、TELNETなどがあります。使用を禁止している各サービスのUDPポートとTCPポートは、両方とも必ず閉じてください。一方のポートを使用禁止にしていても、他方のポートが使用可能であると、オペレーティング・システムの安全性が低下します。

12. データベース・リンクの通信プロトコルを構成する。

データベース・リンクの通信で使用するプロトコルを指定するには、OUTBOUND_DBLINK_PROTOCOLS初期化パラメータを次の設定のいずれかに設定します。

a. ALL (デフォルト)は、データベース・リンクですべてのネット・プロトコルを使用できます。

b. comma-separated_list_of_protocolsには、TPC、TCPSまたはIPCを設定できます。たとえば、1つのプロトコルの場合は、次のようになります。

```
ALTER SYSTEM SET OUTBOUND_DBLINK_PROTOCOLS=TCPS;
```

複数のプロトコルの場合は、次のようになります。

```
ALTER SYSTEM SET OUTBOUND_DBLINK_PROTOCOLS=TCP,TCPS,IPC;
```

c. NONEは、データベース・リンクの通信を無効にします。

13. 必要な場合、グローバル・データベース・リンクのLDAPルックアップを無効にする。

ALLOW_GLOBAL_DBLINKS初期化パラメータを設定して、グローバル・データベース・リンクのLDAPルックアップを有効または無効にします。次の設定があります。

a. ONは、グローバル・データベース・リンクのLDAPルックアップを有効にします。

b. OFF (デフォルト)は、グローバル・データベース・リンクのLDAPルックアップを無効にします。

関連トピック

- Oracle Database Net Services管理者ガイド
- Transport Layer Security認証の構成
- Oracle Database Net Services管理者ガイド
- 厳密認証の概要
- Oracle Database Net Services管理者ガイド
- Oracle Databaseのネイティブ・ネットワーク暗号化とデータ整合性の構成

A.9.3 Transport Layer Security接続のセキュリティ

Oracleには、Transport Layer Security (TLS)を保護するためのガイドラインがあります。

Transport Layer Security (TLS)は保護された通信のためのインターネット標準プロトコルで、データの整合性と暗号化のメカニズムを提供します。これらのメカニズムは、証明書(および必要な場合は暗号化)を使用して、安全性の高い認証、認可およびメッセージ機能をサポートし、ユーザーあるいはアプリケーションおよびサーバーが送受信するメッセージを保護できます。適切なセキュリティの実施により、保護が最大限に発揮され、セキュリティを脅かすギャップや露見が最小限に抑えられます。

1. 構成ファイル(クライアントおよびリスナー用など)では、インストール時に構成された正しいポートをTLSに対して使用する。

   HTTPSは任意のポートで実行できますが、標準的には、すべてのHTTPS準拠のブラウザがデフォルトで確認できるポート443を指定します。たとえば、次のようにポートをURLに指定することもできます。

   ```
   https://secure.example.com:4445/
   ```

   ファイアウォールを使用している場合は、保護された(TLS)通信のために同じポートを使用する必要もあります。

2. tnsnames.oraファイル(通常はクライアント上またはLDAPディレクトリ内)のADDRESSパラメータに、PROTOCOLとしてTCPSが指定されていることを確認する。

   同一の仕様が、(通常は`$ORACLE_HOME/network/admin`ディレクトリ内の)`listener.ora`ファイルに指定されている必要もあります。

3. 各通信の両サイドでTLSモードが一致していることを確認する。たとえば、データベース(一方)とユーザーまたはアプリケーション(もう一方)が同じTLSモードである必要があります。

   クライアントまたはサーバー認証(一方向)、クライアントおよびサーバー認証(双方向)、または認証なしのモードを指定できます。

4. サーバーがクライアントの暗号スイートをサポートしていること、および証明書のキーのアルゴリズムが使用されていることを確認する。

5. サーバーとクライアントの両方でDN一致を使用可能にし、接続時にサーバーがクライアントに対してその識別情報を偽造するのを防ぐ。

   この設定では、サーバーの識別情報のグローバル・データベース名をサーバー証明書のDNと照合することで、その情報が正しいことが確認されます。

   DN一致は、`tnsnames.ora`ファイルで使用可能にできます。たとえば:

   ```
   set:SSL_SERVER_CERT_DN="cn=finance,cn=OracleContext,c=us,o=example"
   ```

   この設定を使用可能にしない場合、クライアント・アプリケーションはサーバー証明書をチェックできず、サーバーによる識別情報の偽造が可能となります。

6. server.keyファイル内部のRSA秘密キーから暗号化を削除しない。このファイルを読み取り、解析するためにパスフレーズを入力する必要があります。

ノート:

TLS 非対応のサーバーではパスフレーズは不要です。

サーバーが十分に保護されていると判断した場合は、元のファイルを保持しながらRSA秘密キーから暗号化を削除できます。これによって、パスフレーズが不要になるため、システム・ブート・スクリプトでデータベース・サーバーを起動できるようになります。理想的には、権限をルート・ユーザーのみに制限し、Webサーバーを`root`として起動し、別のユーザーとしてログインします。そうしないと、このキーを取得しただれもがネット上でルート・ユーザーになりすましたり、サーバーに送信されたデータを復号化する可能性があります。

関連トピック

- Transport Layer Security認証の構成
- 『Oracle Database Net Servicesリファレンス・ガイド』

親トピック: ネットワークの保護に関するガイドライン

A.10 外部プロシージャの保護に関するガイドライン

`ENFORCE_CREDENTIAL`環境変数では、`extproc`プロセスがどのようにユーザー資格証明およびコールアウト関数を認証するかを制御します。

この変数は`extproc.ora`ファイルに指定できます。この変数を変更する前に、外部ライブラリの処理用にサイトのセキュリティ要件を確認します。セキュリティを最大にするために、`ENFORCE_CREDENTIAL`変数を`TRUE`に設定します。デフォルト設定は`FALSE`です。

関連項目:

外部プロシージャの保護

親トピック: Oracle Databaseの安全性の維持

A.11 監査に関するガイドライン

Oracleには、監査のためのガイドラインがあります。

- 監査情報の管理の容易性
  監査は比較的低コストですが、監査するイベントの数はできるだけ制限してください。
- 通常のデータベース・アクティビティの監査
  Oracleには、特定のデータベース・アクティビティに関する履歴情報を収集する必要がある場合のガイドラインがあります。
- 疑わしいデータベース・アクティビティの監査
  Oracleには、不審なデータベース・アクティビティの監視を監査する場合のガイドラインがあります。
- 機密データの監査
  機密オブジェクトに対して統合監査ポリシーを作成する場合は、`ACTIONS ALL`句を含めることをお薦めします。
- 監査の推奨設定
  Oracleには、ほとんどのサイトに適用される推奨の監査設定が含まれた事前定義ポリシーがあります。
- UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューの問合せのためのベスト・プラクティス
  `UNIFIED_AUDIT_TRAIL`データ・ディクショナリ・ビューの問合せから最良の結果を得るには、次のガイドラインに従う必要があります。

親トピック: Oracle Databaseの安全性の維持

A.11.1 監査情報の管理の容易性

監査は比較的低コストですが、監査するイベントの数はできるだけ制限してください。

この制限によって、監査対象の文を実行したときのパフォーマンスへの影響が最小限に抑えられ、監査証跡のサイズが最小限になるため、分析と理解が容易になります。

監査方針を企画する際は、次のガイドラインに従ってください。

1. 監査の目的を評価する。

   監査の目的を明確にしておくと、適切な監査方針を企画でき、不要な監査を回避できます。

   たとえば、不審なデータベース・アクティビティの調査のために監査すると仮定します。この情報のみでは不明確です。どのデータベース・アクティビティが疑わしい、または注意を要するといった具体的な情報が必要です。そのために、たとえば、データベース内の表から無許可にデータが削除されていないかを監査するというように、監査方針を絞り込みます。このような目的を設定すれば、監査の対象となるアクションの種類や、疑わしいアクティビティによって影響を受けるオブジェクトの種類を限定できます。

2. 監査について十分理解する。

   目標とする情報の取得に必要な最小限の文、ユーザーまたはオブジェクトを監査します。これによって、不要な監査情報のために重要な情報の識別が困難になることや、SYSTEM表領域内の貴重な領域が無駄に使用されることがなくなります。収集が必要なセキュリティ情報の量と、その情報を格納して処理する能力とのバランスを保つ必要があります。

   たとえば、データベース・アクティビティに関する情報を収集するために監査する場合は、追跡するアクティビティの種類を正確に判断した上で、必要な情報を収集するために必要な期間内で、目的のアクティビティのみを監査します。別の例として、各セッションの論理I/O情報のみを収集する場合は、オブジェクトを監査しないでください。

3. 監査方針を実施する前に法務部門と打ち合せる。

   組織の法務部門に監査方針を検討するように依頼する必要があります。監査では組織の他のユーザーが監視されるため、サイトのコンプライアンス・ポリシーと会社の方針に適切に従っていることを確認する必要があります。

親トピック: 監査のガイドライン

A.11.2 通常のデータベース・アクティビティの監査

Oracleには、特定のデータベース・アクティビティに関する履歴情報を収集する必要がある場合のガイドラインがあります。

1. 関連のあるアクションのみを監査する。

   少なくとも、ユーザー・アクセス、システム権限の使用およびデータベース・スキーマ構造の変更を監査します。役に立たない監査レコードのために重要な情報が識別できない事態を避け、監査証跡管理の量を削減するために、目的のデータベース・アクティビティのみを監査してください。監査対象が多すぎるとデータベースのパフォーマンスに影響する可能性があることにも注意してください。

   たとえば、データベースのすべての表に対する変更を監査すると、監査証跡レコードが多くなりすぎてデータベースのパフォーマンスが低下する可能性があります。ただし、Human Resources(人事管理)表内の給与のように、重要な表に対する変更を監査することは有益です。

   ファイングレイン監査を使用することで、特定のアクションを監査できます。ファイングレイン監査については、ファイングレイ

ン監査を使用した特定のアクティビティの監査を参照してください。

2. 監査レコードをアーカイブし、監査証跡を削除する。

   必要な情報を収集した後は、目的の監査レコードをアーカイブし、この情報の監査証跡を削除します。次の項を参照してください。

   - 監査証跡のアーカイブ
   - 監査証跡レコードの削除

3. 自社のプライバシに関する考慮事項を検討する。

   プライバシに関する法規によって、追加のビジネス・プライバシ・ポリシーが必要となる場合があります。プライバシに関するほとんどの法規では、個人を特定できる情報(PII)へのアクセスをビジネスで監視する必要があり、このような監視は監査によって実施されます。ビジネス・レベルのプライバシ・ポリシーでは、技術的、法的および企業ポリシーの問題など、データ・アクセスおよびユーザー・アカウンタビリティに関するすべての事項を処理する必要があります。

4. その他の監査情報をOracle Databaseのログ・ファイルでチェックする。

   Oracle Databaseによって生成されたログ・ファイルには、データベースの監査時に使用できる有益な情報が含まれています。たとえば、Oracle Databaseではアラート・ファイルが作成され、STARTUP操作、SHUTDOWN操作およびデータベースにデータ・ファイルを追加するなどの構造上の変更が記録されます。

   たとえば、コミットまたはロールバックされたトランザクションを監査する場合は、REDOログ・ファイルを使用できます。

5. 監査証跡および再帰的SQL文のサイズを減らすには、トップレベルの文のみを監査します。

   作成する統合監査ポリシーによって非常に多数のレコードが生成されることが懸念される場合は、`CREATE AUDIT POLICY`文に`ONLY TOPLEVEL`句を含めます。たとえば、`DBMS_STATS.GATHER_DATABASE_STATS` SQL文の監査では、何千もの監査レコードが生成されます。ユーザー`SYS`を含むすべてのユーザーからトップレベルの文を監査できます。

親トピック: 監査のガイドライン

A.11.3 疑わしいデータベース・アクティビティの監査

Oracleには、不審なデータベース・アクティビティの監視を監査する場合のガイドラインがあります。

1. 最初に一般的な監査を行い、特定の対象を監査する。

   疑わしいデータベース・アクティビティの監査を開始するとき、通常は対象のユーザーまたはスキーマ・オブジェクトの特定に使用できる情報がありません。したがって、まず、一般的な監査、つまり、統合監査ポリシーを使用した監査を行います。「監査ポリシーの構成」に、SQL文、スキーマ、オブジェクト、権限などを監査する方法が説明されています。

   準備的な監査情報の記録と分析を終了した後は、監査ポリシーを変更して特定のアクションと権限を監査します。ポリシーに条件を追加して、不要な監査レコードを除外できます。`AUDIT POLICY`文で`EXCEPT`句を使用して、監査する必要のない特定のユーザーを除外することもできます。統合監査ポリシーの詳細は、統合監査ポリシーおよびAUDIT文を使用したアクティビティの監査を参照してください。

   ファイングレイン監査を使用した特定のアクティビティの監査で説明するように、ファイングレイン監査を使用して特定のアクションを監査できます。

   この処理は、疑わしいデータベース・アクティビティの原因について結論が出せるだけの十分な裏付けが収集できるまで継続してください。

2. 一般的な疑わしいアクティビティを監査する。

   一般的な疑わしいアクティビティは、次のとおりです。

   - 通常以外の時間中にデータベースにアクセスするユーザー
   - 複数回失敗したユーザー・ログイン試行
   - 存在しないユーザーによるログイン試行

   また、SQLテキストなど、SQL問合せで使用すると、クレジット・カード番号などの機密データが監査証跡の列に表示される可能性があることに注意してください。アカウントを共有しているユーザー、または同じIPアドレスからログインしている複数のユーザーを監視する必要もあります。この種のアクティビティは、UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューを問い合せることで検索できます。非常にきめ細かい方法では、ファイングレイン監査ポリシーを作成します。

親トピック: 監査のガイドライン

A.11.4 機密データの監査

機密オブジェクトに対して統合監査ポリシーを作成する場合は、ACTIONS ALL句を含めることをお薦めします。

この句を含めると、これらの機密オブジェクトへの直接アクセスと間接アクセスの両方に対する監査レコードが確実に生成されます。ACTIONS ALLは、機密オブジェクトの監査にのみ使用してください。

関連トピック

- 例: 表でのすべてのアクションの監査

親トピック: 監査のガイドライン

A.11.5 監査の推奨設定

Oracleには、ほとんどのサイトに適用される推奨の監査設定が含まれた事前定義ポリシーがあります。

たとえば:

- ORA_SECURECONFIGは、Oracle Databaseリリース11gの同じデフォルト監査設定を監査します。これは、ALTER ANY TABLE、GRANT ANY PRIVILEGE、CREATE USERなどの複数の権限の使用を追跡します。追跡されるアクションには、ALTER USER、CREATE ROLE、LOGON、および一般に実行されるその他のアクティビティが含まれます。データベースがOracle Database 12cで生成された場合にかぎり、ポリシーはデフォルトで有効になります。
- ORA_DATABASE_PARAMETERは、一般的に使用されるOracle Databaseパラメータ設定のALTER DATABASE、ALTER SYSTEMおよびCREATE SPFILEを監査します。デフォルトでは、このポリシーは有効になっていません。
- ORA_ACCOUNT_MGMTは、一般的に使用されるユーザー・アカウントおよび権限の設定のCREATE USER、ALTER USER、DROP USER、CREATE ROLE、DROP ROLE、ALTER ROLE、SET ROLE、GRANTおよびREVOKEを監査します。デフォルトでは、このポリシーは有効になっていません。

関連項目:

これらの事前定義の監査ポリシーとその他の監査ポリシーの詳細は、事前定義の統合監査ポリシーを使用したアクティビティの監査を参照してください

親トピック: 監査のガイドライン

A.11.6 UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューのためのベスト・プラクティス

UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューの問合せから最良の結果を得るには、次のガイドラインに従う必要があります。

1. 統合監査内部表の統計が最新であることを確認してください。

   UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューを問い合せる前に、AUDSYSスキーマのAUD$UNIFIED表でDBMS_STATS.GATHER_TABLE_STATSプロシージャを実行して、統合監査表の統計が更新されるようにします。

2. オペレーティング・システムの過剰ファイルに書き込まれた統合監査レコードをロードします。

   これは、明示的に実行するか、DBMS_AUDIT_MGMT.LOAD_UNIFIED_AUDIT_FILESプロシージャを使用してOracle Schedulerジョブを構成することによって実行できます。

3. 統合監査証跡内のレコード数が非常に大きい数(100万など)に達したら、適切なアーカイブおよびパージ・メカニズムを開始します。

   統合監査証跡をアーカイブおよび削除すると、そうしない場合は量が増大して読取りパフォーマンスの問題が発生する可能性があるデータの量が減少します。標準の削除ポリシーを構成することをお薦めします。作成する削除ポリシーは、システムで生成される監査レコードの率によって異なります。監査レコード生成率が高い場合は、頻繁な削除が必要です。

4. 統合監査証跡をカスタム表領域に移動します。

   カスタム表領域を使用すると、監査データをより適切に管理でき、SYSAUX表領域の他のオブジェクトへの影響が軽減されます。デフォルトでは、統合監査証跡レコードはSYSAUX表領域に書き込まれます。別の表領域を使用するには、DBMS_AUDIT_MGMT.SET_AUDIT_TRAIL_LOCATIONプロシージャを実行します。

5. UNIFIED_AUDIT_TRAILデータ・ディクショナリ・ビューを問い合せる場合は、EVENT_TIMESTAMP_UTC列をWHERE句に含めます。

   EVENT_TIMESTAMP_UTC列は、UTCタイムゾーン内の監査イベントのタイムスタンプを記録します。この列を問合せに含めるとパーティション・プルーニングが達成されるため、UNIFIED_AUDIT_TRAILビューの読取りパフォーマンスの向上に役立ちます。

関連トピック

- 統合監査証跡へのオペレーティング・システムの監査レコードの移動
- 監査証跡のアーカイブ
- 監査証跡レコードの削除

親トピック: 監査のガイドライン

A.12 CONNECTロール変更への対処

CONNECTロールはOracle Databaseリリース7で導入され、データベース・ロールに新しい堅牢なサポートが追加されました。

- CONNECTロールが変更された理由

CONNECTロールは、サンプル・コード、アプリケーション、ドキュメントおよび技術論文に使用されています。

- CONNECTロール変更がアプリケーションに与える影響
  CONNECTロールの変更は、データベース・アップグレード、アカウント・プロビジョニングおよび新しいデータベースを使用したアプリケーションのインストールにおいて確認できます。
- CONNECTロール変更がユーザーに与える影響
  CONNECTロールへの変更による影響は、一般ユーザー、アプリケーション開発者およびクライアント/サーバー・アプリケーションでそれぞれ異なります。
- CONNECTロール変更に対処する方法
  CONNECTロールの変更の影響に対処するには、次の3つの方法をお薦めします。

親トピック: Oracle Databaseの安全性の維持

A.12.1 CONNECTロールが変更された理由

CONNECTロールは、サンプル・コード、アプリケーション、ドキュメントおよび技術論文に使用されています。

Oracle Database 10gリリース2 (10.2)では、CONNECTロールは変更されました。Oracle Database 10.2より前のリリースから現行のリリースにアップグレードする場合は、最新のリリースでのCONNECTロールの変更に注意してください。

CONNECTロールは、当初、特殊な権限を伴って設定されていました。その権限とは、ALTER SESSION、CREATE CLUSTER、CREATE DATABASE LINK、CREATE SEQUENCE、CREATE SESSION、CREATE SYNONYM、CREATE TABLE、CREATE VIEWです。

Oracle Database 10g リリース2以降のCONNECTロールには、CREATE SESSION権限のみが含まれ、他の権限はすべて削除されています。Oracle Database 12cリリース1以降、CONNECTロールにCREATE SESSIONおよびSET CONTAINER権限が含まれました。

CONNECTロールは、Oracle Databaseでの新規アカウントのプロビジョニングで頻繁に使用されましたが、データベースへの接続に前述の権限がすべて必要なわけではありません。この変更によって、優れたセキュリティ手続きを簡単に規定できるようになります。

各ユーザーにはそれぞれのタスクに必要な権限のみを付与します。これが「最低限の権限」原則と呼ばれる考え方です。最低限の権限により権限を制限することでリスクが軽減されるため、必要なことはこれまでどおり簡単に実行できると同時に、不適切な行為を不注意に、あるいは意図的に実行される可能性が低くなります。

親トピック: CONNECTロール変更への対処

A.12.2 CONNECTロール変更がアプリケーションに与える影響

CONNECTロールの変更は、データベース・アップグレード、アカウント・プロビジョニングおよび新しいデータベースを使用したアプリケーションのインストールにおいて確認できます。

- CONNECTロール変更がデータベース・アップグレードに与える影響
  CONNECTロールがデータベース・アップグレードに与える影響に注意してください。
- CONNECTロール変更がアカウント・プロビジョニングに与える影響
  CONNECTロールがアカウント・プロビジョニングに与える影響に注意してください。
- CONNECTロール変更が新規のデータベースを使用するアプリケーションに与える影響
  CONNECTロールが新規のデータベースを使用するアプリケーションに与える影響に注意してください。

親トピック: CONNECTロール変更への対処

A.12.2.1 CONNECTロール変更がデータベース・アップグレードに与える影響

CONNECTロールがデータベース・アップグレードに与える影響に注意してください。

既存のOracleデータベースをOracle Database 10gリリース2 (10.2)にアップグレードすると、CONNECTロールの権限はCREATE SESSION権限のみになるよう自動的に変更されます。

ほとんどのアプリケーションは、アプリケーション・オブジェクトがすでに存在するため、この影響を受けません。新たに表、ビュー、順序、シノニム、クラスタまたはデータベース・リンクを作成する必要はありません。

表、ビュー、順序、シノニム、クラスタまたはデータベース・リンクを作成したり、ALTER SESSIONコマンドを動的に使用するアプリケーションは、権限が不十分なために失敗する場合があります。

親トピック: CONNECTロール変更がアプリケーションに与える影響

A.12.2.2 CONNECTロール変更がアカウント・プロビジョニングに与える影響

CONNECTロールがアカウント・プロビジョニングに与える影響に注意してください。

アプリケーションまたはDBAがアカウント・プロビジョニング・プロセスの一部としてCONNECTロールを付与する場合は、CREATE SESSION権限のみが含まれます。追加の権限は、直接または別のロールを介して付与する必要があります。

この問題は、カスタマイズされた新しいデータベース・ロールを作成することで対処できます。

関連項目:

CONNECTロール変更に対処する方法

親トピック: CONNECTロール変更がアプリケーションに与える影響

A.12.2.3 CONNECTロール変更が新規のデータベースを使用するアプリケーションに与える影響

CONNECTロールが新規のデータベースを使用するアプリケーションに与える影響に注意してください。

Oracle Database 10g リリース2(10.2)のユーティリティ(DBCA)、またはDBCAから生成されたデータベース作成テンプレートを使用して作成された新しいデータベースでは、CREATE SESSION権限のみを伴ったCONNECTロールが定義されます。

新しいデータベースを使用するアプリケーションのインストールは、そのアプリケーションに使用されているデータベース・スキーマの権限がCONNECTロールのみを介して付与されている場合、失敗する可能性があります。

親トピック: CONNECTロール変更がアプリケーションに与える影響

A.12.3 CONNECTロール変更がユーザーに与える影響

CONNECTロールへの変更による影響は、一般ユーザー、アプリケーション開発者およびクライアント/サーバー・アプリケーションでそれぞれ異なります。

- CONNECTロール変更が一般ユーザーに与える影響
  CONNECTロールが一般ユーザーに与える影響に注意してください。
- CONNECTロール変更がアプリケーション開発者に与える影響
  CONNECTロールがアプリケーション開発者に与える影響に注意してください。
- CONNECTロール変更がクライアント・サーバー・アプリケーションに与える影響
  CONNECTロールがクライアント・サーバー・アプリケーションに与える影響に注意してください。

親トピック: CONNECTロール変更への対処

A.12.3.1 CONNECTロール変更が一般ユーザーに与える影響

CONNECTロールが一般ユーザーに与える影響に注意してください。

新しいCONNECTロールは、CREATE SESSION権限のみを提供します。アプリケーションを使用するためにデータベースに接続するユーザーの場合、CONNECTロールにはまだCREATE SESSION権限があるため影響はありません。

ただし、CONNECTロールのみでプロビジョニングされた特定のユーザーの場合は、適切な権限がないことになります。表、ビュー、順序、シノニム、クラスタまたはデータベース・リンクを作成したり、ALTER SESSIONコマンドを使用するユーザーです。これらのユーザーに必要な権限は、CONNECT ロールでは提供されなくなりました。必要な追加権限を認可するには、データベース管理者が該当する権限用の追加ロールを作成および適用するか、その権限を必要としているユーザーに直接付与する必要があります。

ALTER SESSION権限は、イベントを設定するために必要です。ALTER SESSION権限が必要なデータベース・ユーザーはほとんどいません。

ALTER SESSION権限は、他のALTER SESSIONコマンドには必要ありません。

親トピック: CONNECTロール変更がユーザーに与える影響

A.12.3.2 CONNECTロール変更がアプリケーション開発者に与える影響

CONNECTロールがアプリケーション開発者に与える影響に注意してください。

CONNECTロールのみでプロビジョニングされたアプリケーション開発者には、表、ビュー、順序、シノニム、クラスタまたはデータベース・リンクを作成したり、ALTER SESSION文を使用するための適切な権限はありません。

該当する権限用の追加ロールを作成および適用するか、その権限を必要としているアプリケーション開発者に直接付与する必要があります。

親トピック: CONNECTロール変更がユーザーに与える影響

A.12.3.3 CONNECTロール変更がクライアント・サーバー・アプリケーションに与える影響

CONNECTロールがクライアント・サーバー・アプリケーションに与える影響に注意してください。

専用のユーザー・アカウントを使用するほとんどのクライアント/サーバー・アプリケーションは、この変更の影響は受けません。

ただし、アカウント・プロビジョニングまたはランタイム操作時に、動的SQLを使用してユーザー・スキーマ内にプライベート・シノニムまたは一時表を作成するアプリケーションは影響を受けます。この場合は、その動作に適したシステム権限を取得するための追加ロールや権限付与が必要です。

親トピック: CONNECTロール変更がユーザーに与える影響

A.12.4 CONNECTロール変更に対処する方法

CONNECTロールの変更の影響に対処するには、次の3つの方法をお薦めします。

- 新しいデータベース・ロールの作成
  CONNECTロールから削除された権限は、新しいデータベース・ロールを作成することで管理できます。
- CONNECT権限のリストア
  rstrconn.sqlスクリプトはCONNECT権限をリストアします。
- CONNECT権限受領者を表示するデータ・ディクショナリ・ビュー
  DBA_CONNECT_ROLE_GRANTEESデータ・ディクショナリ・ビューを使用すると、古いCONNECTロールを継続して使用する管理者は、どのユーザーがそのロールを持っているかを確認できます。

- 最低限の権限の分析調査
  Oracleパートナおよびアプリケーション・プロバイダは、より安全性の高い製品をOracleの顧客に供給できるように最低限の権限の分析を行う必要があります。

親トピック: CONNECTロール変更への対処

A.12.4.1 新しいデータベース・ロールの作成

CONNECTロールから削除された権限は、新しいデータベース・ロールを作成することで管理できます。

1. アップグレードされたOracleデータベースに接続して、新しいデータベース・ロールを作成します。

   次の例では、my_app_developerというロールを使用しています。

```
CREATE ROLE my_app_developer;
GRANT CREATE TABLE, CREATE VIEW, CREATE SEQUENCE, CREATE SYNONYM, CREATE
CLUSTER, CREATE DATABASE LINK, ALTER SESSION TO my_app_developer;
```

2. CONNECTロールを持っているユーザーまたはデータベース・ロールを判別し、そのユーザーまたはロールに新しいロールを付与します。

```
SELECT USER$.NAME, ADMIN_OPTION, DEFAULT_ROLE
 FROM USER$, SYSAUTH$, DBA_ROLE_PRIVS
 WHERE PRIVILEGE# =
 (SELECT USER# FROM USER$ WHERE NAME = 'CONNECT')
 AND USER$.USER# = GRANTEE#
 AND GRANTEE = USER$.NAME
 AND GRANTED_ROLE = 'CONNECT';
NAME                           ADMIN_OPTI DEF
------------------------------ ---------- ---
R1                             YES        YES
R2                             NO         YES
GRANT my_app_developer TO R1 WITH ADMIN OPTION;
GRANT my_app_developer TO R2;
```

3. 権限分析ポリシーを作成することでユーザーが必要とする権限を決定します。

   収集した情報を分析して、よりきめ細かい追加のデータベース・ロール作成に使用できます。特定のユーザーについて使用されていない権限は取り消すことができます。

   たとえば:

```
BEGIN
 DBMS_PRIVILEGE_CAPTURE.CREATE_CAPTURE(
  name          => 'my_app_dev_role_pol',
  description   => 'Captures my_app_developer role use',
  type          => DBMS_PRIVILEGE_CAPTURE.G_ROLE,
  roles         => role_name_list('my_app_developer');
END;
/
EXEC DBMS_PRIVILEGE_CAPTURE.ENABLE_CAPTURE ('my_app_dev_role_pol');
```

4. 一定期間後に、権限分析ポリシーを無効化して、レポートを生成します。

```
EXEC DBMS_PRIVILEGE_CAPTURE.DISABLE_CAPTURE ('my_app_dev_role_pol');
EXEC DBMS_PRIVILEGE_CAPTURE.GENERATE_RESULT ('my_app_dev_role_pol');
```

5. レポートを生成した後、権限分析データ・ディクショナリ・ビューを問い合せます。

   たとえば:

```
SELECT USERNAME, SYS_PRIV, OBJECT_OWNER, OBJECT_NAME FROM DBA_USED_PRIVS;
```

関連トピック

- 権限分析の実行による権限使用の特定

親トピック: CONNECTロール変更に対処する方法

A.12.4.2 CONNECT権限のリストア

rstrconn.sqlスクリプトはCONNECT権限をリストアします。

データベースのアップグレードまたは新しいデータベースの作成後は、このスクリプトを使用して、Oracle Database 10gリリース2 (10.2)でCONNECTロールから削除された権限を付与できます。この方法を使用した場合、使用されていない権限は、必要としていないユーザーから取り消してください。

CONNECT権限をリストアするには:

1. $ORACLE_HOME/rdbms/adminディレクトリにあるrstrconn.sqlスクリプトを実行します。

   ```
   @$ORACLE_HOME/rdbm_admin/rstrconn.sql
   ```

2. 使用される権限を監視します。

   たとえば:

   ```
   CREATE AUDIT POLICY connect_priv_pol
     PRIVILEGES AUDIT CREATE TABLE, CREATE SEQUENCE, CREATE SYNONYM, CREATE
   DATABASE LINK, CREATE CLUSTER, CREATE VIEW, ALTER SESSION;
   AUDIT POLICY connect_priv_pol BY psmith;
   ```

3. データベース権限の使用状況を定期的に監視します。

   たとえば:

   ```
   SELECT USERID, NAME FROM AUD$, SYSTEM_PRIVILEGE_MAP WHERE - PRIV$USED =
   PRIVILEGE;
   USERID                         NAME
   ------------------------------ ----------------
   ACME                           CREATE TABLE
   ACME                           CREATE SEQUENCE
   ACME                           CREATE TABLE
   ACME                           ALTER SESSION
   APPS                           CREATE TABLE
   APPS                           CREATE TABLE
   APPS                           CREATE TABLE
   APPS                           CREATE TABLE
   8 rows selected.
   ```

親トピック: CONNECTロール変更に対処する方法

A.12.4.3 CONNECT権限受領者を表示するデータ・ディクショナリ・ビュー

DBA_CONNECT_ROLE_GRANTEESデータ・ディクショナリ・ビューを使用すると、古いCONNECTロールを継続して使用する管理者は、どのユーザーがそのロールを持っているかを確認できます。

表A-1に、DBA_CONNECT_ROLE_GRANTEESビューの列を示します。

表A-1 DBA_CONNECT_ROLE_GRANTEESの列と内容

列	データ型	NULL	説明

列	データ型	NULL	説明
GRANTEE	VARCHAR2(128)	NULL	CONNECT ロールを付与されたユーザー。
PATH_OF_CONNECT_ROLE_GRANT	VARCHAR2(4000)	NULL	ユーザーへの CONNECT の付与に使用されたロール(またはネストされたロール)。
ADMIN_OPT	VARCHAR2(3)	NULL	ユーザーの CONNECT に ADMIN オプションがある場合は YES、ない場合は NO。

親トピック: CONNECTロール変更に対処する方法

A.12.4.4 最低限の権限の分析調査

Oracleパートナおよびアプリケーション・プロバイダは、より安全性の高い製品をOracleの顧客に供給できるように最低限の権限の分析を行う必要があります。

「最低限の権限」原則は、所定の機能を実行するために必要な最低限のセットに権限を制限することでリスクを軽減します。

分析によって同一の権限セットが必要とされたユーザーのクラスごとに、その権限のみを持ったロールを作成します。他の権限はすべて該当するユーザーから取り消し、作成したロールを割り当てます。必要な権限が変化したときは、追加の権限を直接またはこれらの新しいロールを介して付与するか、新たに必要となった権限に応じた新しいロールを作成できます。この方法を使用すると、不適切な権限が制限され、不注意または意図的な被害が軽減されます。

データベース・ユーザーによる権限の使用を示す権限分析ポリシーを作成できます。ポリシーはこの情報を取得し、データ・ディクショナリ・ビューで使用できるようにします。これらのレポートに基づいて、だれにデータへのアクセス権限を持たせるかを決定できます。

関連トピック

- 権限分析の実行による権限使用の特定

親トピック: CONNECTロール変更に対処する方法

B データ暗号化および整合性パラメータ

sqlnet.oraファイルには、データ暗号化および整合性パラメータがあります。

- データ暗号化および整合性のsqlnet.oraの使用について
  データの暗号化および整合性を構成するためのガイドラインとして、デフォルト・パラメータ設定を使用できます。
- サンプルsqlnet.oraファイル
  sqlnet.ora構成ファイルのサンプルは、似た特性を持つ一連のクライアントと似た特性を持つ一連のサーバーに基づいています。
- データ暗号化および整合性パラメータ
  Oracleには、sqlnet.oraファイルで設定できるデータおよび整合性のパラメータがあります。

親トピック: 付録

B.1 データ暗号化と整合性のためのsqlnet.oraの使用について

データの暗号化および整合性を構成するためのガイドラインとして、デフォルト・パラメータ設定を使用できます。

このsqlnet.oraファイルは、Oracle Databaseのネイティブ・ネットワーク暗号化とデータ整合性の構成およびTransport Layer Security認証の構成で説明したネットワーク構成を実行すると生成されます。暗号化およびデータ整合性パラメータも用意されています。

親トピック: データ暗号化および整合性パラメータ

B.2 サンプルsqlnet.oraファイル

sqlnet.ora構成ファイルのサンプルは、似た特性を持つ一連のクライアントと似た特性を持つ一連のサーバーに基づいています。

このファイルには、Oracle Databaseの暗号化とデータ整合性のパラメータの例が含まれています。

デフォルトでは、sqlnet.oraファイルは、ORACLE_HOME/network/adminディレクトリ、またはTNS_ADMIN環境変数によって設定されている場所にあります。TNS_ADMIN変数が正しいsqlnet.oraファイルを指定するように適切に設定されていることを確認します。TNS_ADMIN変数の詳細および設定例は、『SQL*Plusユーザーズ・ガイドおよびリファレンス』を参照してください。

トレース・ファイルの設定

```
#Trace file setup
trace_level_server=16
trace_level_client=16
trace_directory_server=/orant/network/trace
trace_directory_client=/orant/network/trace
trace_file_client=cli
trace_file_server=srv
trace_unique_client=true
```

Oracle Databaseのネイティブ・ネットワーク暗号化

```
sqlnet.encryption_server=accepted
sqlnet.encryption_client=requested
sqlnet.encryption_types_server=(RC4_40)
sqlnet.encryption_types_client=(RC4_40)
```

ノート:

RC4_40 アルゴリズムは、このリリースでは非推奨です。より強力なアルゴリズムを使用するように Oracle Database 環境を移行するには、My Oracle Support ノート 2118136.2 で説明されているパッチをダウンロードしてインストールします。

Oracle Databaseのネットワーク・データ整合性

```
#ASO Checksum
sqlnet.crypto_checksum_server=requested
sqlnet.crypto_checksum_client=requested
sqlnet.crypto_checksum_types_server = (SHA256)
sqlnet.crypto_checksum_types_client = (SHA256)
```

Transport Layer Security

```
#SSL
WALLET_LOCATION = (SOURCE=
                    (METHOD = FILE)
                       (METHOD_DATA =
                          DIRECTORY=/wallet)
SSL_CIPHER_SUITES=(SSL_DH_anon_WITH_RC4_128_MD5)
SSL_VERSION= 3
SSL_CLIENT_AUTHENTICATION=FALSE
```

Common

```
#Common
automatic_ipc = off
sqlnet.authentication_services = (beq)
names.directory_path = (TNSNAMES)
```

Kerberos

```
#Kerberos
sqlnet.authentication_services = (beq, kerberos5)
sqlnet.authentication_kerberos5_service = oracle
sqlnet.kerberos5_conf= /krb5/krb.conf
sqlnet.kerberos5_keytab= /krb5/v5srvtab
sqlnet.kerberos5_realms= /krb5/krb.realm
sqlnet.kerberos5_cc_name = /krb5/krb5.cc
sqlnet.kerberos5_clockskew=900
sqlnet.kerberos5_conf_mit=false
```

RADIUS

```
#Radius
sqlnet.authentication_services = (beq, RADIUS )
sqlnet.radius_authentication_timeout = (10)
sqlnet.radius_authentication_retries = (2)
sqlnet.radius_authentication_port = (1645)
sqlnet.radius_send_accounting = OFF
sqlnet.radius_secret = /orant/network/admin/radius.key
sqlnet.radius_authentication = radius.us.example.com
sqlnet.radius_challenge_response = OFF
sqlnet.radius_challenge_keyword = challenge
sqlnet.radius_challenge_interface =
oracle/net/radius/DefaultRadiusInterface
sqlnet.radius_classpath = /jre1.1/
```

親トピック: データ暗号化および整合性パラメータ

B.3 データ暗号化および整合性パラメータ

Oracleには、sqlnet.oraファイルで設定できるデータおよび整合性のパラメータがあります。

- データ暗号化および整合性パラメータについて
  データ暗号化および整合性パラメータは、使用する暗号化アルゴリズムのタイプを制御します。
- SQLNET.ENCRYPTION_SERVER
  SQLNET.ENCRYPTION_SERVERパラメータでは、クライアントまたはクライアントとして機能しているサーバーがこのサーバーに接続する際の暗号化動作を指定します。
- SQLNET.ENCRYPTION_CLIENT
  SQLNET.ENCRYPTION_CLIENTパラメータでは、このクライアントまたはクライアントとして機能しているサーバーがサーバーに接続する際の暗号化動作を指定します。
- SQLNET.CRYPTO_CHECKSUM_SERVER
  SQLNET.CRYPTO_CHECKSUM_SERVERパラメータでは、クライアントまたはクライアントとして機能している別のサーバーがこのサーバーに接続する際のデータ整合性動作を指定します。
- SQLNET.CRYPTO_CHECKSUM_CLIENT
  SQLNET.CRYPTO_CHECKSUM_CLIENTパラメータでは、このクライアントまたはクライアントとして機能しているサーバーがサーバーに接続する際のデータ整合性動作を指定します。
- SQLNET.ENCRYPTION_TYPES_SERVER
  SQLNET.ENCRYPTION_TYPES_SERVERパラメータでは、このサーバーで使用する暗号化アルゴリズムをアルゴリズムの使用順に指定します。
- SQLNET.ENCRYPTION_TYPES_CLIENT
  SQLNET.ENCRYPTION_TYPES_CLIENTパラメータでは、このクライアントまたはクライアントとして機能しているサーバーで使用する暗号化アルゴリズムを指定します。
- SQLNET.CRYPTO_CHECKSUM_TYPES_SERVER
  SQLNET.CRYPTO_CHECKSUM_TYPES_SERVERパラメータでは、このサーバーまたは別のサーバーのクライアントで使用するデータ整合性アルゴリズムをアルゴリズムの使用順に指定します。
- SQLNET.CRYPTO_CHECKSUM_TYPES_CLIENT
  SQLNET.CRYPTO_CHECKSUM_TYPES_CLIENTパラメータでは、このクライアントまたはクライアントとして機能しているサーバーで使用するデータ整合性アルゴリズムのリストを指定します。

親トピック: データ暗号化および整合性パラメータ

B.3.1 データ暗号化および整合性パラメータについて

データ暗号化および整合性パラメータは、使用する暗号化アルゴリズムのタイプを制御します。

サーバー暗号化、クライアント暗号化、サーバー・チェックサムまたはクライアント・チェックサムの値を指定しない場合、対応する構成パラメータはsqlnet.oraファイルに含まれません。ただし、デフォルトはACCEPTEDです。

データ暗号化と整合性の両方のアルゴリズムでは、サーバーは、そのサーバーのsqlnet.oraファイル内のアルゴリズムのうち、クライアントのsqlnet.oraファイル、またはクライアントのsqlnet.oraファイルにアルゴリズムがリストされていない場合はクライアントのインストール済リストにリストされているアルゴリズムに、最初に一致するものを選択します。サーバーのsqlnet.oraファイルにエントリがない場合、サーバーはそのインストール済リストを順に検索して、クライアント側(クライアントのsqlnet.oraファイルまたはクライアントのインストール済リスト)の項目と照合します。一致するアルゴリズムが見つからず、接続の一方でアルゴ

リズムのタイプ(データ暗号化または整合性)がREQUIREDである場合、接続は失敗します。 それ以外の場合、接続はアルゴリズムのタイプinactiveで成功します。

データ暗号化と整合性のアルゴリズムは、互いに独立して選択されます。表B-1に示すように、暗号化は整合性なしでアクティブにでき、整合性は暗号化なしでアクティブにできます。

表B-1 アルゴリズムのタイプの選択

暗号化の選択	整合性の選択
はい	いいえ
はい	はい
いいえ	はい
いいえ	いいえ

関連トピック

- Oracle Databaseのネイティブ・ネットワーク暗号化とデータ整合性の構成
- 暗号化および整合性のアクティブ化について

親トピック: データ暗号化および整合性パラメータ

B.3.2 SQLNET.ENCRYPTION_SERVER

SQLNET.ENCRYPTION_SERVERパラメータでは、クライアントまたはクライアントとして機能しているサーバーがこのサーバーに接続する際の暗号化動作を指定します。

サーバーの動作は、接続の相手側のSQLNET.ENCRYPTION_CLIENTの設定に部分的に依存します。

表B-2では、SQLNET.ENCRYPTION_SERVERパラメータの属性について説明します。

表B-2 SQLNET.ENCRYPTION_SERVERパラメータの属性

属性	説明
構文	SQLNET.ENCRYPTION_SERVER = valid_value
有効な値	ACCEPTED、REJECTED、REQUESTED、REQUIRED
デフォルト設定	ACCEPTED

関連項目:

SQLNET.ENCRYPTION_SERVERパラメータの詳細は、『Oracle Database Net Servicesリファレンス』を参照してください。

親トピック: データ暗号化および整合性パラメータ

B.3.3 SQLNET.ENCRYPTION_CLIENT

SQLNET.ENCRYPTION_CLIENTパラメータでは、このクライアントまたはクライアントとして機能しているサーバーがサーバーに接続する際の暗号化動作を指定します。

クライアントの動作は、接続の相手側のSQLNET.ENCRYPTION_SERVERの設定値に部分的に依存します。

表B-3では、SQLNET.ENCRYPTION_CLIENTパラメータの属性について説明します。

表B-3 SQLNET.ENCRYPTION_CLIENTパラメータの属性

属性	説明
構文	SQLNET.ENCRYPTION_CLIENT = valid_value
有効な値	ACCEPTED、REJECTED、REQUESTED、REQUIRED
デフォルト設定	ACCEPTED

関連項目:

SQLNET.ENCRYPTION_CLIENTパラメータの詳細は、『Oracle Database Net Servicesリファレンス』を参照してください。

親トピック: データ暗号化および整合性パラメータ

B.3.4 SQLNET.CRYPTO_CHECKSUM_SERVER

SQLNET.CRYPTO_CHECKSUM_SERVERパラメータでは、クライアントまたはクライアントとして機能している別のサーバーがこのサーバーに接続する際のデータ整合性動作を指定します。

動作は、接続の相手側のSQLNET.CRYPTO_CHECKSUM_CLIENTの設定に部分的に依存します。

表B-4では、SQLNET.CRYPTO_CHECKSUM_SERVERパラメータの属性について説明します。

表B-4 SQLNET.CRYPTO_CHECKSUM_SERVERパラメータの属性

属性	説明
構文	SQLNET.CRYPTO_CHECKSUM_SERVER = valid_value
有効な値	ACCEPTED、REJECTED、REQUESTED、REQUIRED
デフォルト設定	ACCEPTED

関連項目:

SQLNET.CRYPTO_CHECKSUM_SERVERパラメータの詳細は、『Oracle Database Net Servicesリファレンス』を参照してください。

親トピック: データ暗号化および整合性パラメータ

B.3.5 SQLNET.CRYPTO_CHECKSUM_CLIENT

SQLNET.CRYPTO_CHECKSUM_CLIENTパラメータでは、このクライアントまたはクライアントとして機能しているサーバーがサーバーに接続する際のデータ整合性動作を指定します。

動作は、接続の相手側のSQLNET.CRYPTO_CHECKSUM_SERVERの設定に部分的に依存します。

表B-5では、SQLNET.CRYPTO_CHECKSUM_CLIENTパラメータの属性について説明します。

表B-5 SQLNET.CRYPTO_CHECKSUM_CLIENTパラメータの属性

属性	説明
構文	SQLNET.CRYPTO_CHECKSUM_CLIENT = valid_value
有効な値	ACCEPTED、REJECTED、REQUESTED、REQUIRED
デフォルト設定	ACCEPTED

親トピック: データ暗号化および整合性パラメータ

B.3.6 SQLNET.ENCRYPTION_TYPES_SERVER

SQLNET.ENCRYPTION_TYPES_SERVERパラメータでは、このサーバーで使用する暗号化アルゴリズムをアルゴリズムの使用順に指定します。

このリストは、相互に使用可能なアルゴリズムを接続のクライアント側とネゴシエートする際に使用されます。各アルゴリズムは、一致するものが見つかるまで、使用可能なクライアント・アルゴリズムのタイプのリストに対してチェックされます。インストールされていないアルゴリズムをこの側で指定した場合、接続はエラー・メッセージORA-12650: 共通の暗号化またはデータ整合性アルゴリズムがありません。で終了します。

表B-6では、SQLNET.ENCRYPTION_TYPES_SERVERパラメータの属性について説明します。

表B-6 SQLNET.ENCRYPTION_TYPES_SERVERパラメータの属性

属性	説明
構文	SQLNET.ENCRYPTION_TYPES_SERVER = (valid_encryption_algorithm [,valid_encryption_algorithm])
有効な値	• AES256: AES (256 ビット・キー・サイズ) • AES192: AES (192 ビット・キー・サイズ) • AES128: AES (128 ビット・キー・サイズ)

属性	説明
デフォルト設定	ローカルの sqlnet.ora ファイルでアルゴリズムが定義されていない場合、インストールされているすべてのアルゴリズムが前述の順でネゴシエーションに使用されます。
使用上のノート	複数の暗号化アルゴリズムを指定できます。単一値またはアルゴリズム名のリストを指定できます。たとえば、次の暗号化パラメータはいずれも使用できます。 SQLNET.ENCRYPTION_TYPES_SERVER=(AES256) SQLNET.ENCRYPTION_TYPES_SERVER=(AES256,AES192,AES128)

関連項目:

SQLNET.ENCRYPTION_TYPES_SERVERパラメータの詳細は、『Oracle Database Net Servicesリファレンス』を参照してください。

親トピック: データ暗号化および整合性パラメータ

B.3.7 SQLNET.ENCRYPTION_TYPES_CLIENT

SQLNET.ENCRYPTION_TYPES_CLIENTパラメータでは、このクライアントまたはクライアントとして機能しているサーバーで使用する暗号化アルゴリズムを指定します。

このリストは、相互に使用可能なアルゴリズムを接続の相手側とネゴシエートする際に使用されます。インストールされていないアルゴリズムをこの側で指定した場合、接続はORA-12650: 共通の暗号化またはデータ整合性アルゴリズムがありません。のメッセージで終了します。

表B-7では、SQLNET.ENCRYPTION_TYPES_CLIENTパラメータの属性について説明します。

表B-7 SQLNET.ENCRYPTION_TYPES_CLIENTパラメータの属性

属性	説明
構文	SQLNET.ENCRYPTION_TYPES_CLIENT = (valid_encryption_algorithm [,valid_encryption_algorithm])
有効な値	● AES256: AES (256 ビット・キー・サイズ) ● AES192: AES (192 ビット・キー・サイズ) ● AES128: AES (128 ビット・キー・サイズ)
デフォルト設定	ローカルの sqlnet.ora ファイルでアルゴリズムが定義されていない場合、インストールされているすべてのアルゴリズムがネゴシエーションに使用されます。

属性	説明
使用上のノート	各暗号化アルゴリズムをカンマで区切ることで、複数の暗号化アルゴリズムを指定できます。たとえば: `SQLNET.ENCRYPTION_TYPES_CLIENT=(AES256,AES192,AES128)`

関連項目:

`SQLNET.ENCRYPTION_TYPES_CLIENT`パラメータの詳細は、『Oracle Database Net Servicesリファレンス』を参照してください。

親トピック: データ暗号化および整合性パラメータ

B.3.8 SQLNET.CRYPTO_CHECKSUM_TYPES_SERVER

`SQLNET.CRYPTO_CHECKSUM_TYPES_SERVER`パラメータでは、このサーバーまたは別のサーバーのクライアントで使用するデータ整合性アルゴリズムをアルゴリズムの使用順に指定します。

このリストは、相互に使用可能なアルゴリズムを接続の相手側とネゴシエートする際に使用されます。各アルゴリズムは、一致するものが見つかるまで、使用可能なクライアント・アルゴリズムのタイプのリストに対してチェックされます。インストールされていないアルゴリズムをこの側で指定した場合、接続は`ORA-12650: 共通の暗号化またはデータ整合性アルゴリズムがありません。`のエラー・メッセージで終了します。

表B-8では、`SQLNET.CRYPTO_CHECKSUM_TYPES_SERVER`パラメータの属性について説明します。

表B-8 SQLNET.CRYPTO_CHECKSUM_TYPES_SERVERパラメータの属性

属性	説明
構文	`SQLNET.CRYPTO_CHECKSUM_TYPES_SERVER = (valid_crypto_checksum_algorithm [,valid_crypto_checksum_algorithm])`
有効な値	● `SHA512`: SHA-2、512 ビットのハッシュを生成します。 ● `SHA384`: SHA-2、384 ビットのハッシュを生成します。 ● `SHA256`: SHA-2、256 ビットのハッシュを生成します。これがデフォルト値です。 ● `SHA-1`: Secure Hash Algorithm ● `MD5`: Message Digest 5 ノート: MD5 は、このリリースでは非推奨です。より強力なアルゴリズムを使用するようにOracle Database 環境を移行するには、My Oracle Support ノート

属性	説明
	2118136.2 で説明されているパッチをダウンロードしてインストールします。
デフォルト設定	使用可能なすべてのアルゴリズム

関連項目:

SQLNET.CRYPTO_CHECKSUM_TYPES_SERVERパラメータの詳細は、『Oracle Database Net Servicesリファレンス』を参照してください。

親トピック: データ暗号化および整合性パラメータ

B.3.9 SQLNET.CRYPTO_CHECKSUM_TYPES_CLIENT

SQLNET.CRYPTO_CHECKSUM_TYPES_CLIENTパラメータでは、このクライアントまたはクライアントとして機能しているサーバーで使用するデータ整合性アルゴリズムのリストを指定します。

このリストは、相互に使用可能なアルゴリズムを接続の相手側とネゴシエートする際に使用されます。インストールされていないアルゴリズムをこの側で指定した場合、接続はORA-12650: 共通の暗号化またはデータ整合性アルゴリズムがありません。のエラー・メッセージで終了します。

表B-9では、SQLNET.CRYPTO_CHECKSUM_TYPES_CLIENTパラメータの属性について説明します。

表B-9 SQLNET.CRYPTO_CHECKSUM_TYPES_CLIENTパラメータの属性

属性	説明
構文	SQLNET.CRYPTO_CHECKSUM_TYPES_CLIENT = (valid_crypto_checksum_algorithm [,valid_crypto_checksum_algorithm])
有効な値	● SHA512: SHA-2、512 ビットのハッシュを生成します。 ● SHA384: SHA-2、384 ビットのハッシュを生成します。 ● SHA256: SHA-2、256 ビットのハッシュを生成します。これがデフォルト値です。 ● SHA-1: Secure Hash Algorithm ● MD5: Message Digest 5 ノート: MD5 は、このリリースでは非推奨です。より強力なアルゴリズムを使用するようにOracle Database 環境を移行するには、My Oracle Support ノート

属性	説明
	2118136.2 で説明されているパッチをダウンロードしてインストールします。
デフォルト設定	ローカルの sqlnet.ora ファイルでアルゴリズムが定義されていない場合、インストールされているすべてのアルゴリズムが SHA256 で始まるネゴシエーションに使用されます。

関連項目:

SQLNET.CRYPTO_CHECKSUM_TYPES_CLIENTパラメータの詳細は、『Oracle Database Net Servicesリファレンス』を参照してください。

親トピック: データ暗号化および整合性パラメータ

C Kerberos、TLSおよびRADIUS認証パラメータ

sqlnet.oraファイルおよびデータベース初期化ファイルには、Kerberos、RADIUSまたはTLS認証パラメータが用意されています。

- Kerberos認証を使用するクライアントとサーバーのパラメータ
  Oracle Databaseには、Kerberos認証を使用するためのクライアントとサーバーのパラメータが用意されています。
- Transport Layer Securityを使用するクライアントとサーバーのパラメータ
  Oracleには、Transport Layer Security認証を制御するパラメータが用意されています。
- RADIUS認証を使用するクライアントとサーバーのパラメータ
  Oracleには、RADIUS認証用のパラメータが用意されています。

親トピック: 付録

C.1 Kerberos認証を使用するクライアントとサーバーのパラメータ

Oracle Databaseには、Kerberos認証を使用するためのクライアントとサーバーのパラメータが用意されています。

表C-1に、Kerberosを使用するクライアントとサーバーの構成ファイルに挿入するパラメータを示します。

表C-1 Kerberos認証パラメータ

ファイル名	構成パラメータ
sqlnet.ora	SQLNET.AUTHENTICATION_SERVICES=(KERBEROS5): クライアントとサーバーの両方に設定します。 SQLNET.AUTHENTICATION_KERBEROS5_SERVICE=oracle : クライアントとサーバーの両方に設定します。 SQLNET.KERBEROS5_CC_NAME=/usr/tmp/DCE-CC : 通常、サーバーでは必要ありません。クライアントが Microsoft Windows 上にありドメインの一部である場合は、インメモリー・チケット・キャッシュの使用と、このパラメータを OSMSFT://または MSLSA:に設定することを検討できます。 SQLNET.KERBEROS5_CLOCKSKEW=1200 : クライアントとサーバーの両方に設定します。 SQLNET.KERBEROS5_CONF=/krb5/krb.conf : クライアントとサーバーの両方に設定します。(通常、クライアントでのこのパスは、サーバーでのパスとは異なります。) SQLNET.KERBEROS5_CONF_MIT=(TRUE) : クライアントとサーバーの両方で、これを TRUE に設定します。 SQLNET.KERBEROS5_REALMS=/krb5/krb.realms : この設定は、通常、クライアントまたはサーバーでは必要ありません。 SQLNET.KERBEROS5_KEYTAB=/krb5/v5srvtab: このパラメータはサーバーにのみ設

ファイル名	構成パラメータ
	定し、クライアントには設定しません。 SQLNET.FALLBACK_AUTHENTICATION=FALSE: クライアントとサーバーの両方に設定します。
初期化パラメータ・ファイル	OS_AUTHENT_PREFIX="": このパラメータは、サーバーにのみ設定し、クライアントには設定しません。

関連トピック

- ステップ6C: sqlnet.oraパラメータの設定(オプション)

親トピック: Kerberos、TLSおよびRADIUS認証パラメータ

C.2 Transport Layer Securityを使用するクライアントとサーバーのパラメータ

Oracleには、Transport Layer Security認証を制御するパラメータが用意されています。

- Transport Layer Securityのパラメータの構成方法
  Transport Layer Security (TLS)のパラメータの構成方法は2つあります。
- クライアントとサーバーのTransport Layer Security認証パラメータ
  Oracleには、静的と動的の両方のTransport Layer Security (TLS)認証パラメータが用意されています。
- Transport Layer Securityの暗号スイート・パラメータ
  Transport Layer Security (TLS)に対して暗号スイート・パラメータを構成できます。
- サポートされているTransport Layer Security暗号スイート
  Oracle Databaseでは、Transport Layer Security (TLS)に対して数多くの暗号スイートをサポートしています。
- Transport Layer Securityバージョン・パラメータ
  使用するTLSのバージョンを構成するために、様々なTransport Layer Security (TLS)パラメータを設定できます。
- Transport Layer Securityクライアント認証パラメータ
  クライアントに対してSecure Sockes Layer (TLS)の静的および動的パラメータを構成できます。
- Transport Layer Security X.509サーバー照合パラメータ
  SSL_SERVER_DN_MATCHおよびSSL_SERVER_CERT_DNパラメータで、クライアントが接続するサーバーの識別情報を検証します。
- Oracleウォレット・ロケーション
  セキュリティ資格証明をプロセス領域にロードするためにOracleウォレットにアクセスする必要があるアプリケーションに対して、ウォレット・ロケーション・パラメータを指定する必要があります。

親トピック: Kerberos、TLSおよびRADIUS認証パラメータ

C.2.1 Transport Layer Securityのパラメータの構成方法

Transport Layer Security (TLS)のパラメータの構成方法は2つあります。

- 静的: sqlnet.oraファイルに存在するパラメータの名前。SSL_CIPHER_SUITESやSSL_VERSIONなどのパラ

メータは、listener.oraファイルを使用して構成することもできます。

- 動的: Oracle Netアドレスのセキュリティ・サブセクションで使用されるパラメータの名前。

親トピック: Transport Layer Securityを使用するクライアントとサーバーのパラメータ

C.2.2 クライアントとサーバーのTransport Layer Security認証パラメータ

Oracleには、静的と動的の両方のTransport Layer Security (TLS)認証パラメータが用意されています。

表C-2では、サーバーでTLSを構成するための静的パラメータと動的パラメータについて説明します。

表C-2 クライアントとサーバーのTLS認証パラメータ

属性	説明
パラメータ名(静的)	SQLNET.AUTHENTICATION_SERVICES
パラメータ名(動的)	AUTHENTICATION
パラメータ・タイプ	文字列 LIST
パラメータ・クラス	静的
設定できる値	使用可能な認証サービスのリストに TCPS を追加します。
デフォルト値	デフォルト値なし。
説明	ユーザーが使用する認証サービスを制御します。 ノート: 動的バージョンでは、1 つのタイプの設定のみがサポートされます。
既存/新規パラメータ	既存
構文(静的)	SQLNET.AUTHENTICATION_SERVICES = (TCPS, selected_method_1, selected_method_2)
例(静的)	SQLNET.AUTHENTICATION_SERVICES = (TCPS, radius)
構文(動的)	AUTHENTICATION = string
例(動的)	AUTHENTICATION = (TCPS)

親トピック: Transport Layer Securityを使用するクライアントとサーバーのパラメータ

C.2.3 Transport Layer Securityの暗号スイート・パラメータ

Transport Layer Security (TLS)に対して暗号スイート・パラメータを構成できます。

表C-3では、暗号スイートを構成するための静的パラメータと動的パラメータについて説明します。

表C-3 Transport Layer Securityの暗号スイート・パラメータ

属性	説明
パラメータ名(静的)	SSL_CIPHER_SUITES
パラメータ名(動的)	SSL_CIPHER_SUITES
パラメータ・タイプ	文字列 LIST
パラメータ・クラス	静的
設定できる値	既知の TLS 暗号スイート
デフォルト値	デフォルトなし
説明	TLS で使用する暗号化とデータ整合性の組合せを制御します。
既存/新規パラメータ	既存
構文(静的)	SSL_CIPHER_SUITES=(SSL_cipher_suite1[, SSL_cipher_suite2, ... SSL_cipher_suiteN])
例(静的)	SSL_CIPHER_SUITES=(SSL_DH_DSS_WITH_DES_CBC_SHA)
構文(動的)	SSL_CIPHER_SUITES=(SSL_cipher_suite1 [, SSL_cipher_suite2, ...SSL_cipher_suiteN])
例(動的)	SSL_CIPHER_SUITES=(SSL_DH_DSS_WITH_DES_CBC_SHA)

親トピック: Transport Layer Securityを使用するクライアントとサーバーのパラメータ

C.2.4 サポートされているTransport Layer Security暗号スイート

Oracle Databaseでは、Transport Layer Security (TLS)に対して数多くの暗号スイートをサポートしています。

暗号スイートを次に示します。

- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA

- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
- TLS_RSA_WITH_AES_128_CBC_SHA256
- TLS_RSA_WITH_AES_128_GCM_SHA256
- TLS_RSA_WITH_AES_128_CBC_SHA
- TLS_RSA_WITH_AES_256_CBC_SHA
- TLS_RSA_WITH_AES_256_CBC_SHA256

関連トピック

- TLS暗号スイートの認証、暗号化、整合性およびTLSバージョン

親トピック: Transport Layer Securityを使用するクライアントとサーバーのパラメータ

C.2.5 Transport Layer Securityバージョン・パラメータ

使用するTLSのバージョンを構成するために、様々なTransport Layer Security (TLS)パラメータを設定できます。

表C-4では、使用するTLSのバージョンを構成するための静的および動的なSSL_VERSIONパラメータについて説明します。

表C-4 Transport Layer Securityバージョン・パラメータ

属性	説明
パラメータ名(静的)	SSL_VERSION
パラメータ名(動的)	SSL_VERSION
パラメータ・タイプ	文字列
パラメータ・クラス	静的
設定できる値	TLS の有効なバージョン。値は次のとおりです。 undetermined | 1.0 | 1.1 | 1.2 | 3.0 あるバージョンまたは別のバージョンを指定する場合は、"or"を使用します。次の値を使用できます。 1.0 or 3.0 | 1.2 or 3.0 | 1.1 or 1.0 | 1.2 or 1.0 | 1.2 or 1.1 | 1.1 or 1.0 or 3.0 | 1.2 or 1.0 or 3.0 | 1.2 or 1.1 or 1.0 | 1.2 or 1.1 or 3.0 |1.2 or 1.1 or 1.0 or 3
デフォルト値	1.2、1.1 または 1.0 特定の値(1.2 など)を使用する場合は、明示的に設定する必要がありま

属性	説明
	す。
説明	TLS 接続のバージョンを強制します。
既存/新規パラメータ	新規
構文(静的)	`SSL_VERSION=version`
例(静的)	`SSL_VERSION=1.1`
構文(動的)	`SSL_VERSION=version`
例(動的)	`SSL_VERSION=1.1 or 1.2`

ノート:

`ADD_SSLv3_IMPLICITLY` 初期化パラメータは、`SSL_VERSION` パラメータには影響しません。

親トピック: Transport Layer Securityを使用するクライアントとサーバーのパラメータ

C.2.6 Transport Layer Securityクライアント認証パラメータ

クライアントに対してSecure Sockes Layer (TLS)の静的および動的パラメータを構成できます。

表C-5では、`SSL_CLIENT_AUTHENTICATION`パラメータについて説明します。

表C-5 Transport Layer Securityクライアント認証パラメータ

属性	説明
パラメータ名(静的)	`SSL_CLIENT_AUTHENTICATION`
パラメータ名(動的)	`SSL_CLIENT_AUTHENTICATION`
パラメータ・タイプ	ブール
パラメータ・クラス	静的
設定できる値	`TRUE` または `FALSE`
デフォルト値	`TRUE`

属性	説明
説明	クライアント(サーバーに加えて)を TLS を使用して認証するかどうかを制御します。
既存/新規パラメータ	新規
構文(静的)	`SSL_CLIENT_AUTHENTICATION={TRUE | FALSE}`
例(静的)	`SSL_CLIENT_AUTHENTICATION=FALSE`
構文(動的)	`SSL_CLIENT_AUTHENTICATION={TRUE | FALSE}`
例(動的)	`SSL_CLIENT_AUTHENTICATION=FALSE`

親トピック: Transport Layer Securityを使用するクライアントとサーバーのパラメータ

C.2.7 Transport Layer Security X.509サーバー照合パラメータ

SSL_SERVER_DN_MATCHおよびSSL_SERVER_CERT_DNパラメータで、クライアントが接続するサーバーの識別情報を検証します。

- SSL_SERVER_DN_MATCH
  SSL_SERVER_DN_MATCHパラメータでは、サーバーの識別名(DN)がサービスの名前と一致することを強制します。
- SSL_SERVER_CERT_DN
  SSL_SERVER_CERT_DNでは、サーバーの識別名(DN)を指定します。

親トピック: Transport Layer Securityを使用するクライアントとサーバーのパラメータ

C.2.7.1 SSL_SERVER_DN_MATCH

SSL_SERVER_DN_MATCHパラメータでは、サーバーの識別名(DN)がサービスの名前と一致することを強制します。

表C-6では、SSL_SERVER_DN_MATCHパラメータについて説明します。

表C-6 SSL_SERVER_DN_MATCHパラメータ

属性	説明
パラメータ名	SSL_SERVER_DN_MATCH
格納場所	`sqlnet.ora`
目的	このパラメータを使用して、サーバーの識別名(DN)とそのサービス名の一致を強制します。一致確認を強制する場合は、TLS によって証明書がサーバーからのものであることが保証されます。一致確認を強制しない場合、TLS によるチェックは実行されるものの、一致しているかどうかに関係なく接続は許可されます。 一致

属性	説明
	を強制しないと、サーバーの識別情報の偽装が可能になります。
値	yes|on|true。一致を強制します。DN とサービス名が一致した場合、接続は正しく行われ、それ以外の場合、接続は失敗します。 no|off|false。一致を強制しません。DN がサービス名と一致していない場合、接続は成功しますが、エラーが sqlnet.log ファイルに記録されます。
デフォルト	Oracle8i 以降: false。TLS クライアントは(常に)サーバーDN をチェックします。サービス名と一致していない場合、接続は成功しますが、エラーが sqlnet.log ファイルに記録されます。
使用上のノート	さらに、tnsnames.ora のパラメータ SSL_SERVER_CERT_DN もサーバー DN の一致を有効にするように構成します。

親トピック: Transport Layer Security X.509サーバー照合パラメータ

C.2.7.2 SSL_SERVER_CERT_DN

SSL_SERVER_CERT_DNでは、サーバーの識別名(DN)を指定します。

表C-7では、SSL_SERVER_CERT_DNパラメータについて説明します。

表C-7 SSL_SERVER_CERT_DNパラメータ

属性	説明
パラメータ名	SSL_SERVER_CERT_DN
格納場所	tnsnames.ora。接続先のサーバーごとにクライアントに格納するか、接続先のサーバーごとに LDAP ディレクトリに格納し、一元的に更新できます。
目的	このパラメータでは、サーバーの識別名(DN)を指定します。クライアントは、サーバーの DN とそのサービス名が確実に一致するように、この情報を使用して、各サーバーに予定している DN のリストを取得します。
値	サーバーの識別名(DN)と等しい値を指定します。
デフォルト	なし
使用上のノート	さらに、sqlnet.ora のパラメータ SSL_SERVER_DN_MATCH もサーバーDN の一致を有効にするように構成します。

属性	説明
例	`dbalias=(description=address_list=(address=(protocol=tcps)(host=hostname)(port=portnum)))(connect_data=(sid=Finance))(security=(SSL_SERVER_CERT_DN="CN=Finance,CN=OracleContext,C=US,O=Acme"))`

親トピック: Transport Layer Security X.509サーバー照合パラメータ

C.2.8 Oracleウォレット・ロケーション

セキュリティ資格証明をプロセス領域にロードするためにOracleウォレットにアクセスする必要があるアプリケーションに対して、ウォレット・ロケーション・パラメータを指定する必要があります。

表C-8に、ウォレット・ロケーションを指定する必要がある構成ファイルを示します。

- `sqlnet.ora`
- `listener.ora`

表C-8 ウォレット・ロケーション・パラメータ

静的構成	動的構成
`WALLET_LOCATION = (SOURCE= (METHOD=File) (METHOD_DATA= (DIRECTORY=your_wallet_dir) ) )`	`MY_WALLET_DIRECTORY = your_wallet_dir`

デフォルトのウォレット・ロケーションはORACLE_HOMEディレクトリです。

親トピック: Transport Layer Securityを使用するクライアントとサーバーのパラメータ

C.3 RADIUS認証を使用するクライアントとサーバーのパラメータ

Oracleには、RADIUS認証用のパラメータが用意されています。

- sqlnet.oraファイルのパラメータ
  `sqlnet.ora`ファイルにRADIUS固有のパラメータを含めることができます。
- 最小限のRADIUSパラメータ
  少なくとも、`SQLNET.AUTHENTICATION_SERVICES`パラメータと`SQLNET.RADIUS.AUTHENTICATION`パラメータを使用する必要があります。
- RADIUSの初期化ファイル・パラメータ
  RADIUSの場合、`OS_AUTHENT_PREFIX`初期化パラメータを設定します。

親トピック: Kerberos、TLSおよびRADIUS認証パラメータ

C.3.1 sqlnet.oraファイルのパラメータ

`sqlnet.ora`ファイルにRADIUS固有のパラメータを含めることができます。

- SQLNET.AUTHENTICATION_SERVICES
  `SQLNET.AUTHENTICATION_SERVICES`パラメータでは、RADIUSアダプタを使用するようにクライアントまたは

サーバーを構成します。

- SQLNET.RADIUS_ALTERNATE
  SQLNET.RADIUS_ALTERNATEパラメータでは、プライマリ・サーバーをフォルト・トレランスに使用できない場合に使用する代替RADIUSサーバーの場所を設定します。
- SQLNET.RADIUS_ALTERNATE_PORT
  SQLNET.RADIUS_ALTERNATE_PORTパラメータでは、代替RADIUSサーバーのリスニング・ポートを設定します。
- SQLNET.RADIUS_ALTERNATE_TIMEOUT
  SQLNET.RADIUS_ALTERNATE_TIMEOUTパラメータでは、代替RADIUSサーバーで応答を待機する時間を設定します。
- SQLNET.RADIUS_ALTERNATE_RETRIES
  SQLNET.RADIUS_ALTERNATE_RETRIESパラメータでは、代替RADIUSサーバーでメッセージを再送信する回数を設定します。
- SQLNET.RADIUS_AUTHENTICATION
  SQLNET.RADIUS_AUTHENTICATIONパラメータでは、プライマリRADIUSサーバーの場所(ホスト名またはドット区切りの10進形式)を設定します。
- SQLNET.RADIUS_AUTHENTICATION_INTERFACE
  SQLNET.RADIUS_AUTHENTICATION_INTERFACEパラメータでは、RADIUSがチャレンジ・レスポンス(非同期)モードの場合に、GUIを含むJavaクラスの名前を設定します。
- SQLNET.RADIUS_AUTHENTICATION_PORT
  SQLNET.RADIUS_AUTHENTICATION_PORTパラメータでは、プライマリRADIUSサーバーのリスニング・ポートを設定します。
- SQLNET.RADIUS_AUTHENTICATION_TIMEOUT
  SQLNET.RADIUS_AUTHENTICATION_TIMEOUTパラメータでは、応答を待機する時間を設定します。
- SQLNET.RADIUS_AUTHENTICATION_RETRIES
  SQLNET.RADIUS_AUTHENTICATION_RETRIESパラメータでは、認証情報を再送信する回数を設定します。
- SQLNET.RADIUS_CHALLENGE_RESPONSE
  SQLNET.RADIUS_CHALLENGE_RESPONSEパラメータでは、チャレンジ・レスポンス(非同期)モードのオンとオフを切り替えます。
- SQLNET.RADIUS_CHALLENGE_KEYWORD
  SQLNET.RADIUS_CHALLENGE_KEYWORDパラメータでは、RADIUSサーバーからのチャレンジを要求するためのキーワードを設定します。
- SQLNET.RADIUS_CLASSPATH
  SQLNET.RADIUS_CLASSPATHパラメータでは、JavaクラスおよびJDK Javaライブラリのパスを指定します。
- SQLNET.RADIUS_SECRET
  SQLNET.RADIUS_SECRETパラメータでは、RADIUS秘密キーのファイル名と場所を指定します。
- SQLNET.RADIUS_SEND_ACCOUNTING
  SQLNET.RADIUS_SEND_ACCOUNTINGパラメータでは、アカウンティングのオンとオフを切り替えます。

親トピック: RADIUS認証を使用するクライアントとサーバーのパラメータ

C.3.1.1 SQLNET.AUTHENTICATION_SERVICES

SQLNET.AUTHENTICATION_SERVICESパラメータでは、RADIUSアダプタを使用するようにクライアントまたはサーバーを構成します。

表C-9では、SQLNET.AUTHENTICATION_SERVICESパラメータの属性について説明します。

表C-9 SQLNET.AUTHENTICATION_SERVICESパラメータの属性

属性	説明
構文	`SQLNET.AUTHENTICATION_SERVICES=(radius)`
デフォルト設定	なし

親トピック: sqlnet.oraファイルのパラメータ

C.3.1.2 SQLNET.RADIUS_ALTERNATE

`SQLNET.RADIUS_ALTERNATE`パラメータでは、プライマリ・サーバーをフォルト・トレランスに使用できない場合に使用する代替RADIUSサーバーの場所を設定します。

表C-10では、`SQLNET.RADIUS_ALTERNATE`パラメータの属性について説明します。

表C-10 SQLNET.RADIUS_ALTERNATEパラメータの属性

属性	説明
構文	`SQLNET.RADIUS_ALTERNATE=alternate_RADIUS_server_hostname_or_IP_address`
デフォルト設定	`off`

親トピック: sqlnet.oraファイルのパラメータ

C.3.1.3 SQLNET.RADIUS_ALTERNATE_PORT

`SQLNET.RADIUS_ALTERNATE_PORT`パラメータでは、代替RADIUSサーバーのリスニング・ポートを設定します。

表C-11では、`SQLNET.RADIUS_ALTERNATE_PORT`パラメータの属性について説明します。

表C-11 SQLNET.RADIUS_ALTERNATE_PORTパラメータの属性

属性	説明
構文	`SQLNET.RADIUS_ALTERNATE_PORT=alternate_RADIUS_server_listening_port_number`
デフォルト設定	`1645`

親トピック: sqlnet.oraファイルのパラメータ

C.3.1.4 SQLNET.RADIUS_ALTERNATE_TIMEOUT

`SQLNET.RADIUS_ALTERNATE_TIMEOUT`パラメータでは、代替RADIUSサーバーで応答を待機する時間を設定します。

表C-12では、`SQLNET.RADIUS_ALTERNATE_TIMEOUT`パラメータの属性について説明します。

表C-12 SQLNET.RADIUS_ALTERNATE_TIMEOUTパラメータの属性

属性	説明
構文	SQLNET.RADIUS_ALTERNATE_TIMEOUT=time_in_seconds
デフォルト設定	5

親トピック: sqlnet.oraファイルのパラメータ

C.3.1.5 SQLNET.RADIUS_ALTERNATE_RETRIES

SQLNET.RADIUS_ALTERNATE_RETRIESパラメータでは、代替RADIUSサーバーでメッセージを再送信する回数を設定します。

表C-13では、SQLNET.RADIUS_ALTERNATE_RETRIESパラメータの属性について説明します。

表C-13 SQLNET.RADIUS_ALTERNATE_RETRIESパラメータの属性

属性	説明
構文	SQLNET.RADIUS_ALTERNATE_RETRIES=n_times_to_resend
デフォルト設定	3

親トピック: sqlnet.oraファイルのパラメータ

C.3.1.6 SQLNET.RADIUS_AUTHENTICATION

SQLNET.RADIUS_AUTHENTICATIONパラメータでは、プライマリRADIUSサーバーの場所(ホスト名またはドット区切りの10進形式)を設定します。

RADIUSサーバーがOracleサーバーとは異なるコンピュータ上にある場合は、そのコンピュータのホスト名またはIPアドレスを指定する必要があります。

表C-14では、SQLNET.RADIUS_AUTHENTICATIONパラメータの属性について説明します。

表C-14 SQLNET.RADIUS_AUTHENTICATIONパラメータの属性

属性	説明
構文	SQLNET.RADIUS_AUTHENTICATION=RADIUS_server_IP_address
デフォルト設定	localhost

親トピック: sqlnet.oraファイルのパラメータ

C.3.1.7 SQLNET.RADIUS_AUTHENTICATION_INTERFACE

SQLNET.RADIUS_AUTHENTICATION_INTERFACEパラメータでは、RADIUSがチャレンジ・レスポンス(非同期)モードの場合に、GUIを含むJavaクラスの名前を設定します。

表C-15では、SQLNET.RADIUS_AUTHENTICATION_INTERFACEパラメータの属性について説明します。

表C-15 SQLNET.RADIUS_AUTHENTICATION_INTERFACEパラメータの属性

属性	説明
構文	`SQLNET.RADIUS_AUTHENTICATION_INTERFACE=Java_class_name`
デフォルト設定	`DefaultRadiusInterface (oracle/net/radius/DefaultRadiusInterface)`

親トピック: sqlnet.oraファイルのパラメータ

C.3.1.8 SQLNET.RADIUS_AUTHENTICATION_PORT

SQLNET.RADIUS_AUTHENTICATION_PORTパラメータでは、プライマリRADIUSサーバーのリスニング・ポートを設定します。

表C-16では、SQLNET.RADIUS_AUTHENTICATION_PORTパラメータの属性について説明します。

表C-16 SQLNET.RADIUS_AUTHENTICATION_PORTパラメータの属性

属性	説明
構文	`SQLNET.RADIUS_AUTHENTICATION_PORT=port_number`
デフォルト設定	1645

親トピック: sqlnet.oraファイルのパラメータ

C.3.1.9 SQLNET.RADIUS_AUTHENTICATION_TIMEOUT

SQLNET.RADIUS_AUTHENTICATION_TIMEOUTパラメータでは、応答を待機する時間を設定します。

表C-17では、SQLNET.RADIUS_AUTHENTICATION_TIMEOUTパラメータの属性について説明します。

表C-17 SQLNET.RADIUS_AUTHENTICATION_TIMEOUTパラメータの属性

属性	説明
構文	`SQLNET.RADIUS_AUTHENTICATION_TIMEOUT=time_in_seconds`
デフォルト設定	5

親トピック: sqlnet.oraファイルのパラメータ

C.3.1.10 SQLNET.RADIUS_AUTHENTICATION_RETRIES

SQLNET.RADIUS_AUTHENTICATION_RETRIESパラメータでは、認証情報を再送信する回数を設定します。

表C-18では、SQLNET.RADIUS_AUTHENTICATION_RETRIESパラメータの属性について説明します。

表C-18 SQLNET.RADIUS_AUTHENTICATION_RETRIESパラメータの属性

属性	説明
構文	SQLNET.RADIUS_AUTHENTICATION_RETRIES=n_times_to_resend
デフォルト設定	3

親トピック: sqlnet.oraファイルのパラメータ

C.3.1.11 SQLNET.RADIUS_CHALLENGE_RESPONSE

SQLNET.RADIUS_CHALLENGE_RESPONSEパラメータでは、チャレンジ・レスポンス(非同期)モードのオンとオフを切り替えます。

表C-19では、SQLNET.RADIUS_CHALLENGE_RESPONSEパラメータの属性について説明します。

表C-19 SQLNET.RADIUS_CHALLENGE_RESPONSEパラメータの属性

属性	説明
構文	SQLNET.RADIUS_CHALLENGE_RESPONSE=on
デフォルト設定	off

親トピック: sqlnet.oraファイルのパラメータ

C.3.1.12 SQLNET.RADIUS_CHALLENGE_KEYWORD

SQLNET.RADIUS_CHALLENGE_KEYWORDパラメータでは、RADIUSサーバーからのチャレンジを要求するためのキーワードを設定します。

ユーザーはクライアントでパスワードを入力しません。

表C-20では、SQLNET.RADIUS_CHALLENGE_KEYWORDパラメータの属性について説明します。

表C-20 SQLNET.RADIUS_CHALLENGE_KEYWORDパラメータの属性

属性	説明
構文	SQLNET.RADIUS_CHALLENGE_KEYWORD=keyword
デフォルト設定	challenge

親トピック: sqlnet.oraファイルのパラメータ

C.3.1.13 SQLNET.RADIUS_CLASSPATH

SQLNET.RADIUS_CLASSPATHパラメータでは、JavaクラスおよびJDK Javaライブラリのパスを指定します。

チャレンジ・レスポンス認証モードを使用する場合、最初にパスワード、続いて追加情報(トークン・カードから取得する動的パスワードなど)を要求するJavaベースのグラフィカル・インタフェースがユーザーに表示されます。

sqlnet.oraファイルにSQLNET.RADIUS_CLASSPATHパラメータを追加して、そのグラフィカル・インタフェースのJavaクラス

を設定し、JDK Javaライブラリへのパスを設定します。

表C-21では、SQLNET.RADIUS_CLASSPATHパラメータの属性について説明します。

表C-21 SQLNET.RADIUS_CLASSPATHパラメータの属性

属性	説明
構文	SQLNET.RADIUS_CLASSPATH=path_to_GUI_Java_classes
デフォルト設定	$ORACLE_HOME/jlib/netradius.jar:$ORACLE_HOME/JRE/lib/sparc/native_threads

親トピック: sqlnet.oraファイルのパラメータ

C.3.1.14 SQLNET.RADIUS_SECRET

SQLNET.RADIUS_SECRETパラメータでは、RADIUS秘密キーのファイル名と場所を指定します。

表C-22では、SQLNET.RADIUS_SECRETパラメータの属性について説明します。

表C-22 SQLNET.RADIUS_SECRETパラメータの属性

属性	説明
構文	SQLNET.RADIUS_SECRET=path_to_RADIUS_secret_key
デフォルト設定	$ORACLE_HOME/network/security/radius.key

親トピック: sqlnet.oraファイルのパラメータ

C.3.1.15 SQLNET.RADIUS_SEND_ACCOUNTING

SQLNET.RADIUS_SEND_ACCOUNTINGパラメータでは、アカウンティングのオンとオフを切り替えます。

アカウンティングを有効にした場合、パケットは、1を加えたリスニング・ポートでアクティブなRADIUSサーバーに送信されます。デフォルトでは、パケットはポート1646に送信されます。この機能は、RADIUSサーバーでアカウンティングがサポートされ、ユーザーがシステムにログオンする回数を追跡する場合にのみオンにする必要があります。

表C-23では、SQLNET.RADIUS_SEND_ACCOUNTINGパラメータの属性について説明します。

表C-23 SQLNET.RADIUS_SEND_ACCOUNTINGパラメータの属性

属性	説明
構文	SQLNET.RADIUS_SEND_ACCOUNTING=on
デフォルト設定	off

親トピック: sqlnet.oraファイルのパラメータ

C.3.2 最小限のRADIUSパラメータ

少なくとも、SQLNET.AUTHENTICATION_SERVICESパラメータとSQLNET.RADIUS.AUTHENTICATIONパラメータを使用する必要があります。

次の設定を使用します。

```
sqlnet.authentication_services = (radius)
sqlnet.radius.authentication   = IP-address-of-RADIUS-server
```

親トピック: RADIUS認証を使用するクライアントとサーバーのパラメータ

C.3.3 RADIUSの初期化ファイル・パラメータ

RADIUSの場合、OS_AUTHENT_PREFIX初期化パラメータを設定します。

たとえば:

```
OS_AUTHENT_PREFIX=""
```

親トピック: RADIUS認証を使用するクライアントとサーバーのパラメータ

D RADIUSを使用した認証デバイスの統合

RADIUSチャレンジ・レスポンス・ユーザー・インタフェースは、RADIUS構成での認証をさらに強化します。

- RADIUSチャレンジ・レスポンス・ユーザー・インタフェースについて
  サード・パーティ認証ベンダーを使用して、RADIUSチャレンジ・レスポンス・ユーザー・インタフェースを特定のデバイスに合うようにカスタマイズできます。
- RADIUSチャレンジ・レスポンス・ユーザー・インタフェースのカスタマイズ
  `OracleRadiusInterface`インタフェースは、独自のクラスを作成することでカスタマイズできます。
- 例: OracleRadiusInterfaceインタフェースの使用
  OracleRadiusInterfaceインタフェースを使用して、ユーザー名とパスワードを取得できます。

親トピック: 付録

D.1 RADIUSチャレンジ・レスポンス・ユーザー・インタフェースについて

サード・パーティ認証ベンダーを使用して、RADIUSチャレンジ・レスポンス・ユーザー・インタフェースを特定のデバイスに合うようにカスタマイズできます。

RADIUS規格をサポートする任意の認証デバイスを、Oracleユーザーが認証されるように設定できます。認証デバイスでチャレンジ・レスポンス・モードを使用する場合、グラフィカル・インタフェースでエンド・ユーザーはまずパスワードの入力、次に追加情報の入力を求められます(たとえば、ユーザーがトークン・カードから取得する動的パスワードがあります)。このインタフェースはJavaベースで、最適なプラットフォーム独立性を提供します。

認証デバイスのサード・パーティ・ベンダーは、その独自のデバイスに合せてこのグラフィカル・ユーザー・インタフェースをカスタマイズする必要があります。たとえば、スマートカード・ベンダーは、チャレンジをスマートカード・リーダーに発行するようにOracleクライアントをカスタマイズします。スマートカードはチャレンジを受け取ると、ユーザーにPINなどの追加情報の入力を求めることで応答します。

関連トピック

- RADIUS認証の構成

親トピック: RADIUSを使用した認証デバイスの統合

D.2 RADIUSチャレンジ・レスポンス・ユーザー・インタフェースのカスタマイズ

`OracleRadiusInterface`インタフェースは、独自のクラスを作成することでカスタマイズできます。

1. `sqlnet.ora`ファイルを開きます。

   デフォルトでは、`sqlnet.ora`ファイルは、`ORACLE_HOME/network/admin`ディレクトリ、または`TNS_ADMIN`環境変数によって設定されている場所にあります。`TNS_ADMIN`変数が正しい`sqlnet.ora`ファイルを指定するように適切に設定されていることを確認します。

2. `SQLNET.RADIUS_AUTHENTICATION_INTERFACE`パラメータを探し、そこにリストされているクラスの名前(`DefaultRadiusInterface`)を作成した新しいクラスの名前に置き換えます。

   この変更を`sqlnet.ora`ファイルで行うと、クラスは認証プロセスを処理するためにOracleクライアントにロードされます。

3. sqlnet.oraファイルを保存して終了します

サード・パーティは、ORACLE.NET.RADIUSパッケージにあるOracleRadiusInterfaceインタフェースを実装する必要があります。

関連項目:

TNS_ADMIN変数の設定例の詳細は、『SQL*Plusユーザーズ・ガイドおよびリファレンス』を参照してください

親トピック: RADIUSを使用した認証デバイスの統合

D.3 例: OracleRadiusInterfaceインタフェースの使用

OracleRadiusInterfaceインタフェースを使用して、ユーザー名とパスワードを取得できます。

例D-1に、OracleRadiusInterfaceインタフェースの使用方法を示します。

例D-1 OracleRadiusInterfaceインタフェースの使用

```
public interface OracleRadiusInterface {
  public void radiusRequest();
  public void radiusChallenge(String challenge);
  public String getUserName();
  public String getPassword();
}
```

詳細は、次のとおりです。

- radiusRequestは、エンド・ユーザーにユーザー名とパスワードの入力を求めます(これらは後でgetUserNameとgetPasswordを使用して取得されます)。
- getUserNameは、ユーザーが入力したユーザー名を抽出します。空の文字列が戻された場合、ユーザーが操作をキャンセルするとみなされます。その後、ユーザーは認証に失敗したことを示すメッセージを受信します。
- getPasswordは、ユーザーが入力したパスワードを抽出します。getUserNameが有効な文字列を戻し、getPasswordが空の文字列を戻した場合、データベースによってチャレンジ・キーワードがパスワードとして使用されます。ユーザーが有効なパスワードを入力した場合、チャレンジがRADIUSサーバーから戻される場合と戻されない場合があります。
- radiusChallengeは、ユーザーがサーバーのチャレンジに応答できるように、RADIUSサーバーから送信された要求を表示します。
- getResponseは、ユーザーが入力した応答を抽出します。有効な応答が戻された場合、その情報が新しいAccess-RequestパケットのUser-Password属性に移入されます。空の文字列が戻された場合、操作は対応する値を戻すことによって両側から中断されます。

親トピック: RADIUSを使用した認証デバイスの統合

E Oracle Database FIPS 140-2の設定

Oracleでは、米国連邦情報処理標準(FIPS)の標準である140-2がサポートされます。

- Oracle Database FIPS 140-2の設定について
  連邦情報処理標準(FIPS)とは、米国商務省国立標準技術研究所(NIST)によって開発された連邦政府のコンピュータ・システムの標準およびガイドラインです。
- 透過的データ暗号化およびDBMS_CRYPTO用のFIPS 140-2の構成
  DBFIPS_140初期化パラメータで、FIPSモードを構成します。
- Transport Layer Securityに対するFIPS 140-2の構成
  SSLFIPS_140パラメータで、Transport Layer Security (TLS)のFIPSモードを構成します。
- ネイティブ・ネットワーク暗号化のためのFIPS 140-2の構成
  サーバーとクライアントの両方に対して、sqlnet.oraファイルでパラメータを設定することで、ネイティブ・ネットワーク暗号化用にFIPS 140-2を構成できます。
- FIPS 140-2のインストール後のチェック
  FIPS 140-2設定の構成後に、オペレーティング・システムで権限を確認する必要があります。
- FIPS 140-2接続の検証
  トレース・ファイルおよびその他の方法を使用して、FIPS 140-2接続を検証できます。

親トピック: 付録

E.1 Oracle Database FIPS 140-2の設定について

連邦情報処理標準(FIPS)とは、米国商務省国立標準技術研究所(NIST)によって開発された連邦政府のコンピュータ・システムの標準およびガイドラインです。

FIPSは、連邦情報セキュリティ・マネジメント法(FISMA)に従って開発されました。FIPSは連邦政府で使用するために開発されましたが、多くの民間団体が自主的にこれらの基準を使用しています。

FIPS 140-2には、暗号化モジュールによって満たされるセキュリティ要件が規定されており、幅広い潜在的なアプリケーションと環境をカバーするための、4つの拡大する定性的なレベルが示されています。セキュリティ・レベル1は、FIPS 140-2アルゴリズム、キー・サイズ、整合性チェックおよび規制によって課せられるその他の要件に準拠します。FIPS 140-2のセキュリティ・レベル1では、本番グレードの設備の要件を超える、モジュールの物理的なセキュリティ・メカニズムは必要ありません。そのため、このレベルでは、指定したオペレーティング環境で実行されている汎用コンピュータでソフトウェア暗号化機能を実行できます。

FIPS 140-2設定がOracle Database用に構成されている場合、データベースはFIPS 140-2レベル1の検証済の暗号ライブラリを使用して、保存されているデータとネットワークで転送中のデータを保護します。Oracle Databaseでは、ネイティブ・ネットワーク暗号化、列および表領域の透過的データ暗号化(TDE) (Oracle SecureFilesを含む)、Transport Layer Security (TLS)、およびDBMS_CRYPTO PL/SQLパッケージに、これらの暗号ライブラリを使用します。

Oracle Databaseでは現在、FIPS 140-2レベル1の検証済の暗号ライブラリとして、以前はRSA BSAFEと呼ばれていたDell BSAFEが使用されています。現在のFIPS認定のステータスの情報は、米国商務省国立標準技術研究所のコンピュータ・セキュリティ・リソース・センター(CSRC)のWebサイトの次のアドレスで確認できます。

http://csrc.nist.gov/groups/STM/cmvp/validation.html

検証済の暗号化モジュールでベンダー「RSA」およびモジュール名「BSAFE」を検索して、FIPS固有の情報を確認できます。

Oracle Database FIPS設定では、Oracleデータベースのみに対してFIPS承認済アルゴリズムが使用されます。FIPSモード

で実行されているOracle Databaseで使用されるサード・パーティ・ベンダーのソフトウェアでは、これらのFIPS承認済アルゴリズムのみを使用する必要があります。そうしないと、ベンダーのソフトウェアに障害が発生します。

親トピック: Oracle Database FIPS 140-2の設定

E.2 透過的データ暗号化およびDBMS_CRYPTO用のFIPS 140-2の構成

DBFIPS_140初期化パラメータで、FIPSモードを構成します。

1. 透過的データ暗号化およびDBMS_CRYPTO PL/SQLパッケージ・プログラム・ユニットをFIPSモードで実行するよう構成するには、DBFIPS_140初期化パラメータをTRUEに設定します。
   このパラメータの効果は、プラットフォームによって異なります。
2. データベースを再起動します。

表E-1に、DBFIPS_140パラメータの各プラットフォームへの作用について説明します。

表E-1 DBFIPS_140初期化パラメータのプラットフォームへの作用

プラットフォーム	DBFIPS_140をTRUEまたはFALSEに設定した場合の効果
Intel x86_64 上の Linux または Windows	● TRUE: TDE および DBMS_CRYPTO プログラム・ユニットは、RSA BSAFE Crypto-C Micro Edition (CCME) 4.1.5 を使用する Micro Edition Suite (MES) 4.6 FIPS モードを使用します。 ● FALSE: TDE および DBMS_CRYPTO プログラム・ユニットは、Intel Performance Primitives (IPP)を使用します。
SPARC T シリーズまたは Intel x86_64 上の Solaris 11.1+	● TRUE: TDE および DBMS_CRYPTO プログラム・ユニットは、RSA BSAFE Crypto-C Micro Edition (CCME) 4.1.5 を使用する Micro Edition Suite (MES) 4.6 FIPS モードを使用します。 ● FALSE: TDE および DBMS_CRYPTO プログラム・ユニットは、Solaris Cryptographic Framework (SCF)/UCrypto (FIPS 140 に対して個別に検証される)を使用します。
他のオペレーティング・システムまたはハードウェア	● TRUE: TDE および DBMS_CRYPTO プログラム・ユニットは、RSA BSAFE Crypto-C Micro Edition (CCME) 4.1.5 を使用する MES 4.6 FIPS モードを使用します。 ● FALSE: TDE および DBMS_CRYPTO プログラム・ユニットは MES 4.6 非 FIPS モードを使用します。

DBFIPS_140をTRUEに設定し、基礎となるライブラリをFIPSモードで使用すると、各プロセスで初めてライブラリをロードする際、一定のオーバーヘッドが生じることに注意してください。これは、署名の検証およびライブラリに対する自己テストの実行によるものです。ライブラリのロード後は、パフォーマンスへの影響はありません。

関連トピック

- Oracle Databaseリファレンス

親トピック: Oracle Database FIPS 140-2の設定

E.3 Transport Layer Securityに対するFIPS 140-2の構成

SSLFIPS_140パラメータで、Transport Layer Security (TLS)のFIPSモードを構成します。

- Transport Layer Security用のSSLFIPS_140およびSSLFIPS_LIBパラメータの構成
  TLS用にFIPS 140-2を構成するには、SSLFIPS_140パラメータを設定する必要があります。Oracle Instant Clientを使用している場合は、SSLFIPS_LIBパラメータも設定する必要があります。
- FIPS 140-2用に承認されているTLS暗号スイート
  暗号スイートは、ネットワーク・ノード間でメッセージを交換する認証、暗号化およびデータ整合性アルゴリズムのセットです。

親トピック: Oracle Database FIPS 140-2の設定

E.3.1 Transport Layer Security用のSSLFIPS_140およびSSLFIPS_LIBパラメータの構成

TLS用にFIPS 140-2を構成するには、SSLFIPS_140パラメータを設定する必要があります。Oracle Instant Clientを使用している場合は、SSLFIPS_LIBパラメータも設定する必要があります。

SSLFIPS_140パラメータで、Transport Layer Security (TLS)アダプタをFIPSモードで実行するように構成します。SSLFIPS_LIBで、FIPSライブラリの場所を設定します。

1. fips.oraファイルが$ORACLE_HOME/ldap/adminディレクトリにあるか、またはFIPS_HOME環境変数で指定された場所にあることを確認します。
2. fips.oraファイルで、SSLFIPS_140およびSSLFIPS_LIBパラメータを設定します。
   - SSLFIPS_140をTRUEに設定し、TLSアダプタをFIPSモードで実行できるようにします。たとえば:

     ```
     SSLFIPS_140=TRUE
     ```

     このパラメータは、デフォルトではFALSEです。
   - Oracle Instant Clientを使用している場合は、SSLFIPS_LIBをFIPSライブラリの場所に設定します。たとえば:

     ```
     SSLFIPS_LIB=$ORACLE_HOME/lib
     ```
3. 任意のデータベース・サーバーまたはクライアントの任意のOracle Databaseホームでこの手順を繰り返します。

SSLFIPS_140をTRUEに設定した場合、Transport Layer Security暗号操作はFIPSモードの埋込みRSA/Micro Edition Suite (MES)ライブラリで行われます。ハードウェア・アクセラレーションが使用可能でホスト・ハードウェアおよびソフトウェアで適切に構成されている場合、これらの暗号操作はCPUによって高速化されます。

SSLFIPS_140をFALSEに設定した場合、Transport Layer Security暗号操作は、非FIPSモードの埋込みRSA/Micro Edition Suite (MES)ライブラリで行われ、TRUEに設定した場合と同様に、可能な場合は操作が高速化されます。

ノート:

Oracle Database 10g リリース 2 (10.2)で使用された SQLNET.SSLFIPS_140 パラメータは、SSLFIPS_140 パラメータに置換されます。このパラメータは、sqlnet.ora ファイルではなく fips.ora ファイルで設定する必要があります。

親トピック: Transport Layer Securityに対するFIPS 140-2の構成

E.3.2 FIPS 140-2用に承認されているTLS暗号スイート

暗号スイートは、ネットワーク・ノード間でメッセージを交換する認証、暗号化およびデータ整合性アルゴリズムのセットです。

たとえば、TLSハンドシェイク時に、メッセージを送受信するときに使用する暗号スイートを確認するために2つのノード間でネゴシエーションが行われます。

特定の暗号スイートの構成

Oracle DatabaseのTLS暗号スイートは、FIPS認定の暗号スイートに自動的に設定されます。特定の暗号スイートを構成する必要がある場合は、sqlnet.oraまたはlistener.oraファイルでSSL_CIPHER_SUITESパラメータを設定します。

```
SSL_CIPHER_SUITES=(SSL_cipher_suite1[,SSL_cipher_suite2[,..]])
```

このパラメータは、Oracle Net Managerを使用してサーバーおよびクライアントで設定することもできます。

特定の暗号スイートを指定しなかった場合、Oracle Databaseでは、データベース・サーバーとクライアントの両方に共通の、最も強力な暗号スイートが使用されます。選択される暗号スイートの優先順位は、次の優先暗号リストおよび低優先暗号リストに示されている順となります。3DES暗号スイートには脆弱性があるため、Oracle Databaseでは、それらは自動的には選択されません。それらは明示的に構成する必要があります。

優先暗号スイート

トランスポート層セキュリティ(TLS)バージョン1.2を使用している場合は、次の暗号スイートがFIPS検証用に承認されています。

- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256

トランスポート層セキュリティ(TLS)バージョン1、1.1または1.2を使用している場合は、次の暗号スイートがFIPS検証用に承認されています。

- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA

3DESベース暗号スイート

3DESベース暗号スイートは設計に脆弱性があるため、Oracleではそれらは推奨されていません。Oracle Databaseリリース21c以上では、次の3DESベース暗号スイートがサポートされています。ただし、それらはデフォルトでは有効になっておらず、sqlnet.oraまたはlistener.oraファイル内のSSL_CIPHER_SUITESパラメータを使用して明示的に構成する必要があります。

- TLS_ECDHE_ECDSA_WITH_3DES_EDE_CBC_SHA
- TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA
- TLS_RSA_WITH_3DES_EDE_CBC_SHA

関連トピック

- ステップ1C: サーバーでのTransport Layer Security暗号スイートの設定(オプション)
- ステップ2D: クライアントのTransport Layer Security暗号スイートの設定(オプション)

親トピック: Transport Layer Securityに対するFIPS 140-2の構成

E.4 ネイティブ・ネットワーク暗号化のためのFIPS 140-2の構成

サーバーとクライアントの両方に対して、sqlnet.oraファイルでパラメータを設定することで、ネイティブ・ネットワーク暗号化用にFIPS 140-2を構成できます。

- ネイティブ・ネットワーク暗号化のためのFIPS 140-2の構成について
  ネイティブ・ネットワーク暗号化のためのFIPS 140-2の構成は、Transport Layer Security (TLS)の構成に似ています。
- ネイティブ・ネットワーク暗号化のためのFIPS_140パラメータの構成
  ネイティブ・ネットワーク暗号化用にFIPS 140-2を構成するには、sqlnet.oraファイルでFIPS_140パラメータを設定する必要があります。

親トピック: Oracle Database FIPS 140-2の設定

E.4.1 ネイティブ・ネットワーク暗号化のためのFIPS 140-2の構成について

ネイティブ・ネットワーク暗号化のためのFIPS 140-2の構成は、Transport Layer Security (TLS)の構成に似ています。

ネットワーク・ネイティブ暗号化の場合は、fips.oraのSSL_FIPS140設定ではなく、sqlnet.ora構成ファイルでSSL_FIPS140を設定してFIPSモードを有効にします。

ネイティブ・ネットワーク暗号化向けにFIPSライブラリでサポートされているアルゴリズムは、次のとおりです。

- 暗号化: AES128、AES192およびAES256
- チェックサム: SHA1、SHA256、SHA384およびSHA512

親トピック: ネイティブ・ネットワーク暗号化のためのFIPS 140-2の構成

E.4.2 ネイティブ・ネットワーク暗号化のためのFIPS_140パラメータの構成

ネイティブ・ネットワーク暗号化用にFIPS 140-2を構成するには、sqlnet.oraファイルでFIPS_140パラメータを設定する必要があります。

FIPS_140パラメータで、FIPSモードで実行するようにネイティブ・ネットワーク暗号化アダプタを構成します。

1. データベース・クライアントまたはデータベース・サーバーによって使用されるsqlnet.oraファイルを探します。
2. sqlnet.oraファイルに次の行を追加します。

```
SQLNET.FIPS_140=TRUE
```

3. 任意のデータベース・サーバーまたはクライアントの任意のOracle Databaseホームでこの手順を繰り返します。

FIPS_140がTRUEに設定されている場合、ネイティブ・ネットワーク暗号化の暗号操作は、FIPSモードの埋込みBSAFE Micro Edition Suite (MES)ライブラリで行われます。ハードウェア・アクセラレーションが使用可能でホスト・ハードウェアおよび

ソフトウェアで適切に構成されている場合、これらの暗号操作はCPUによって高速化されます。

親トピック: ネイティブ・ネットワーク暗号化のためのFIPS 140-2の構成

E.5 FIPS 140-2のインストール後のチェック

FIPS 140-2設定の構成後に、オペレーティング・システムで次の権限を確認する必要があります。

その権限を次に示します。

- システムのセキュリティ・ポリシーに従って、権限のないユーザーがOracle Cryptographic Librariesを実行できないようにするには、すべてのOracle実行可能ファイルに対して実行権限を設定する必要があります。
- ユーザーが誤ってまたは故意にOracle Cryptographic Librariesのファイルを読み取ったり変更しないようにするには、すべてのOracle実行可能ファイルに対して読取りおよび書込み権限を設定する必要があります。

FIPS 140-2 Level 2要件に準拠するには、権限のないユーザーが、オペレーティング・システムで使用しているOracle Cryptographic Librariesのプロセスおよびメモリーの読取り、変更または実行を行えないようにする手順をセキュリティ・ポリシーに含めます。

親トピック: Oracle Database FIPS 140-2の設定

E.6 FIPS 140-2接続の検証

トレース・ファイルおよびその他の方法を使用して、FIPS 140-2接続を検証できます。

- Transport Layer Securityに対するFIPS 140-2接続の検証
  トレース・ファイルを使用して、Transport Layer Security (TLS)に対するFIPS 140-2接続を確認できます。
- ネットワーク・ネイティブ暗号化に対するFIPS 140-2接続の検証
  トレース・ファイルを使用して、ネットワーク・ネイティブ暗号化に対するFIPS 140-2接続を確認できます。
- 透過的データ暗号化およびDBMS_CRYPTOに対するFIPS 140-2接続の検証
  SQL*Plusを使用して、FIPSモードが有効になっているかどうかを確認できます。

親トピック: Oracle Database FIPS 140-2の設定

E.6.1 Transport Layer Securityに対するFIPS 140-2接続の検証

トレース・ファイルを使用して、Transport Layer Security (TLS)に対するFIPS 140-2接続を確認できます。

1. トレースを有効にするには、`sqlnet.ora`に次の行を追加します。

   ```
   trace_directory_server=trace_directory
   trace_file_server=trace_file
   trace_level_server=trace_level
   ```

   たとえば:

   ```
   trace_directory=/private/oracle/owm
   trace_file_server=fips_trace.trc
   trace_level_server=16
   ```

   トレース・レベル16は、FIPSの自己テストの結果を確認するために必要な最低限のトレース・レベルです。

2. `Provider Type: FIPS140`を検索してトレース・ファイルを確認します。

親トピック: FIPS 140-2接続の検証

E.6.2 ネットワーク・ネイティブ暗号化に対するFIPS 140-2接続の検証

トレース・ファイルを使用して、ネットワーク・ネイティブ暗号化に対するFIPS 140-2接続を確認できます。

- トレースを有効にするには、sqlnet.oraに次の行を追加します。

```
trace_directory_server=trace_directory
trace_file_server=trace_file
trace_level_server=trace_level
```

たとえば:

```
trace_directory=/private/oracle/owm
trace_file_server=fips_trace.trc
trace_level_server=16
```

トレース・レベル16は、FIPSの自己テストの結果を確認するために必要な最低限のトレース・レベルです。

親トピック: FIPS 140-2接続の検証

E.6.3 透過的データ暗号化およびDBMS_CRYPTOに対するFIPS 140-2接続の検証

SQL*Plusを使用して、FIPSモードが有効になっているかどうかを確認できます。

1. SQL*Plusを使用して、データベース・インスタンスに接続します。
2. 次のSHOW PARAMETERコマンドを実行します。

```
SHOW PARAMETER DBFIPS_140
```

次のような出力が表示されます。

```
NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------------------------
DBFIPS_140                           boolean     TRUE
```

親トピック: FIPS 140-2接続の検証

F 公開キー・インフラストラクチャ(PKI)要素の管理

orapkiコマンドライン・ユーティリティとsqlnet.oraのパラメータを使用して、公開キー・インフラストラクチャ(PKI)要素を管理できます。

- orapkiユーティリティの使用
  orapkiユーティリティで、公開キー・インフラストラクチャ(PKI)要素(ウォレットや証明書失効リストなど)をコマンドラインから管理します。
- orapkiユーティリティの構文
  orapkiユーティリティの構文で、Oracleウォレット、証明書失効リストまたはPKIデジタル証明書を指定します。
- テスト用の署名付き証明書の作成
  orapkiユーティリティは、テスト用の署名付き証明書を作成する便利で手軽な方法を提供します。
- 証明書の表示
  証明書を作成した後は、orapkiユーティリティを使用してその証明書を表示できます。
- Oracleウォレットへのユーザー指定証明書または信頼できる証明書のインポート
  Oracleウォレットにユーザー指定証明書または信頼できる証明書を追加できます。
- MD5およびSHA-1証明書の使用の制御
  sqlnet.oraファイルを使用して、MD5およびSHA-1署名付き証明書を受け入れるかどうかを制御できます。
- orapkiユーティリティを使用したOracleウォレットの管理
  orapkiユーティリティで、ウォレットを作成、表示、変更できます。証明書と証明書リクエストを追加およびエクスポートできます。
- orapkiユーティリティを使用した証明書失効リスト(CRL)の管理
  証明書失効リスト(CRL)は、orapkiユーティリティを使用して管理する必要があります。
- orapkiの使用方法
  orapkiコマンドの例には、ウォレットおよびユーザー証明書の作成、自己署名証明書を含むウォレットおよび証明書のエクスポートがあります。
- orapkiユーティリティ・コマンドのサマリー
  orapkiコマンドは、ウォレット、証明書失効リスト(CRL)および証明書の様々な管理タスクを実行します。

親トピック: 付録

F.1 orapkiユーティリティの使用

orapkiユーティリティで、公開キー・インフラストラクチャ(PKI)要素(ウォレットや証明書失効リストなど)をコマンドラインから管理します。

これにより、スクリプトを使用してこれらのタスクを自動化できます。PKI要素の管理をスクリプトに組み込む方法を提供することにより、PKIを管理するための数多くのルーチン・タスクを自動化できるようになります。

orapkiコマンドライン・ユーティリティを使用して次のタスクを実行できます。

- テスト用の署名付き証明書の作成および表示
- Oracleウォレットの管理(透過的データ暗号化キーストアを除く):
  - Oracleウォレットの作成および表示
  - 証明書リクエストの追加および削除

- 証明書の追加および削除
- 信頼できる証明書の追加および削除

- 証明書失効リスト(CRL)の管理:
  - 証明書検証用ハッシュ値によるCRLの名前変更
  - Oracle Internet DirectoryでのCRLのアップロード、一覧表示、表示および削除

ノート:

PKI 暗号化と透過的データ暗号化を組み合せた使用は非推奨です。透過的データ暗号化を構成するには、`ADMINISTER KEY MANAGEMENT` SQL 文を使用します。詳細は、『Oracle Database Advanced Security ガイド』を参照してください。

親トピック: 公開キー・インフラストラクチャ(PKI)要素の管理

F.2 orapkiユーティリティの構文

`orapki`ユーティリティの構文で、Oracleウォレット、証明書失効リストまたはPKIデジタル証明書を指定します。

`orapki`コマンドライン・ユーティリティの構文を次に示します。

```
orapki module command -parameter value
```

ここで、`module`を`wallet` (Oracleウォレット)、`crl` (証明書失効リスト)または`cert` (PKIデジタル証明書)にできます。使用可能なコマンドは、使用する`module`によって異なります。

たとえば、`wallet`を使用している場合、`add`コマンドを使用して証明書またはキーをウォレットに追加できます。次の例では、`/private/lhale/cert.txt`にあるユーザー証明書が、`$ORACLE_HOME/wallet/ewallet.p12`にあるウォレットに追加されます。

```
orapki wallet add -wallet $ORACLE_HOME/wallet/ewallet.p12 -user_cert -cert
/private/lhale/cert.txt
```

親トピック: 公開キー・インフラストラクチャ(PKI)要素の管理

F.3 テスト用の署名付き証明書の作成

`orapki`ユーティリティは、テスト用の署名付き証明書を作成する便利で手軽な方法を提供します。

- テスト用の署名付き証明書を作成するには、次のコマンドを使用します。

  ```
  orapki cert create [-wallet wallet_location] -request
  certificate_request_location -cert certificate_location -validity
  number_of_days [-summary]
  ```

このコマンドにより、証明書リクエストから署名付き証明書が作成されます。`-wallet`パラメータは、証明書リクエストへの署名に使用されるユーザー証明書と秘密キーを含むウォレットを指定します。`-validity`パラメータは、現在の日付から数えてこの証明書が有効である日数を指定します。証明書と証明書リクエストの指定は、このコマンドでは必須です。

親トピック: 公開キー・インフラストラクチャ(PKI)要素の管理

F.4 証明書の表示

証明書を作成した後は、orapkiユーティリティを使用してその証明書を表示できます。

- 証明書を表示するには、次のコマンドを使用します。

```
orapki cert display -cert certificate_location [-summary | -complete]
```

このコマンドを使用すると、orapkiを使用して作成したテスト証明書を表示できます。-summaryまたは-completeのいずれかを選択できます。これによりコマンドが表示するデータの詳細さが決まります。-summaryを選択すると、証明書とその有効期限が表示されます。-completeを選択すると、シリアル番号、公開キーなどの追加の証明書情報が表示されます。

親トピック: 公開キー・インフラストラクチャ(PKI)要素の管理

F.5 Oracleウォレットへのユーザー指定証明書または信頼できる証明書のインポート

ユーザー指定証明書または信頼できる証明書をOracleウォレットに追加できます。

- Oracleウォレットに信頼できる証明書を追加するには、-trusted_certパラメータを指定したorapki wallet addを使用します。

```
orapki wallet add -wallet wallet_location [-pwd wallet_password] -trusted_cert
-cert root_and/or_intermediate_certificate_file
```

- Oracleウォレットにユーザー作成証明書を追加するには、-user_certパラメータを指定したorapki wallet addを使用します。

```
orapki wallet add -wallet wallet_location [-pwd wallet_password] -user_cert -
cert user_certificate_file
```

親トピック: 公開キー・インフラストラクチャ(PKI)要素の管理

F.6 MD5およびSHA-1証明書の使用の制御

sqlnet.oraファイルを使用して、MD5およびSHA-1署名付き証明書を受け入れるかどうかを制御できます。

MD5およびSHA-1署名付き証明書を受け入れるかどうかを制御するには、sqlnet.oraファイルを編集して、その使用を有効または無効にします。

ノート:

MD5 は、このリリースでは非推奨です。より強力なアルゴリズムを使用するように Oracle Database 環境を移行するには、My Oracle Support ノート 2118136.2 で説明されているパッチをダウンロードしてインストールします。

1. Oracleデータベースが存在するサーバーにログインします。
2. sqlnet.oraファイルを編集します。

   デフォルトでは、sqlnet.oraファイルは、$ORACLE_HOME/dbsディレクトリ、またはTNS_ADMIN環境変数によって設定されている場所にあります。
3. 次のパラメータを設定します。

- ACCEPT_MD5_CERTSはMD5証明書の使用を制御します。デフォルトはFALSEです。このパラメータは、ORACLE_SSL_ALLOW_MD5_CERT_SIGNATURES環境変数を置き換えます。
- ACCEPT_SHA1_CERTSはSHA-1証明書の使用を制御します。デフォルトはTRUEです。

親トピック: 公開キー・インフラストラクチャ(PKI)要素の管理

F.7 orapkiユーティリティを使用したOracleウォレットの管理

orapkiユーティリティで、ウォレットを作成、表示、変更できます。証明書と証明書リクエストを追加およびエクスポートできます。

- orapkiを使用したウォレットの管理について
  Oracleウォレットの作成および管理に使用されるorapkiコマンドライン・ユーティリティ構文を理解する必要があります。
- orapkiを使用したウォレットの作成、表示および変更
  orapkiを使用して、Oracleウォレットでの様々な管理アクティビティを実行できます。
- orapkiを使用した証明書と証明書リクエストのOracleウォレットへの追加
  orapkiユーティリティを使用して、様々な証明書関連タスクを実行できます。
- orapkiを使用した証明書と証明書リクエストのOracleウォレットからのエクスポート
  orapkiユーティリティを使用して、Oracleウォレットから証明書および証明書リクエストをエクスポートできます。

親トピック: 公開キー・インフラストラクチャ(PKI)要素の管理

F.7.1 orapkiを使用したウォレットの管理について

Oracleウォレットの作成および管理に使用されるorapkiコマンドライン・ユーティリティ構文を理解する必要があります。

orapkiユーティリティのwalletモジュール・コマンドをスクリプトで使用して、ウォレット作成プロセスを自動化できます。たとえば、PKCS#12ウォレットおよび自動ログイン・ウォレットを作成できます。PKCS#12ウォレットに関連付けられている自動ログイン・ウォレット、またはウォレットが作成されたコンピュータおよびウォレット作成ユーザーに対してローカルな自動ログイン・ウォレットを作成できます。ウォレットを表示したり、ウォレットのパスワードを変更したり、AES256アルゴリズムを使用するようにウォレットを変換できます。

ノート:

-wallet パラメータは、すべての wallet モジュール・コマンドで必須です。

親トピック: orapkiユーティリティを使用したOracleウォレットの管理

F.7.2 orapkiを使用したウォレットの作成、表示および変更

orapkiを使用して、Oracleウォレットでの様々な管理アクティビティを実行できます。

- PKCS#12ウォレットの作成
  orapkiユーティリティを使用して、PKCS#12のOracleウォレットを作成できます。
- 自動ログイン・ウォレットの作成
  orapkiユーティリティを使用して、自動ログイン・ウォレットを作成できます。
- PKCS#12ウォレットに関連付けられた自動ログイン・ウォレットの作成
  PKCS#12ウォレットに関連付けられた自動ログイン・ウォレットを作成できます。
- コンピュータとウォレット作成ユーザーにローカルな自動ログイン・ウォレットの作成

orapkiユーティリティを使用して、ウォレットを作成したユーザーのコンピュータにローカルな自動ログイン・ウォレットを作成できます。

- ウォレットの表示
  orapkiユーティリティを使用して、ウォレットを表示できます。
- ウォレットのパスワードの変更
  orapkiユーティリティを使用して、ウォレットのパスワードを変更できます。
- AES256アルゴリズムの使用を目的としたOracleウォレットの変換
  デフォルトでは、ADMINISTER KEY MANAGEMENT文またはALTER SYSTEM文を使用したOracleウォレットは、3DESで暗号化されます。

親トピック: orapkiユーティリティを使用したOracleウォレットの管理

F.7.2.1 PKCS#12ウォレットの作成

orapkiユーティリティを使用して、PKCS#12のOracleウォレットを作成できます。

- Oracle PKCS#12ウォレット(ewallet.p12)を作成するには、orapki wallet createコマンドを使用します。

```
orapki wallet create -wallet wallet_location [-pwd password]
```

コマンドラインでパスワードを指定していない場合、このコマンドではウォレットのパスワードの入力と再入力を求められます。-walletで指定された場所にウォレットが作成されます。

ノート:

セキュリティ上の理由から、コマンドラインにパスワードを指定しないことをお薦めします。パスワードの入力は、求められた場合にのみ行ってください。

親トピック: orapkiを使用したウォレットの作成、表示および変更

F.7.2.2 自動ログイン・ウォレットの作成

orapkiユーティリティを使用して、自動ログイン・ウォレットを作成できます。

- ウォレットを開く際にパスワードを必要としない自動ログイン・ウォレット(cwallet.sso)を作成するには、orapki wallet createコマンドを使用します。

```
orapki wallet create -wallet wallet_location -auto_login_only
```

パスワードを使用しないでウォレットを変更または削除できます。ファイル・システム権限によって、このような自動ログイン・ウォレットに必要なセキュリティが提供されます。

ローカルの自動ログイン・ウォレットを別のコンピュータに移動することはできません。それらは作成されたホストで使用する必要があります。

ローカルの自動ログイン・ウォレットを開く際にパスワードが不要な場合でも、ウォレットを変更または削除するには、関連付けられたPKCS#12ウォレットのパスワードを入力する必要があります。PKCS#12ウォレットを更新した場合、関連付けられた自動ログイン・ウォレットも更新されます。

親トピック: orapkiを使用したウォレットの作成、表示および変更

F.7.2.3 PKCS#12ウォレットに関連付けられた自動ログイン・ウォレットの作成

PKCS#12ウォレットに関連付けられた自動ログイン・ウォレットを作成できます。

自動ログイン・ウォレットを開くのにパスワードは必要ありません。

ただし、ウォレットを変更または削除するには、関連付けられたPKCS#12ウォレットのパスワードを入力する必要があります。PKCS#12ウォレットを更新した場合、関連付けられた自動ログイン・ウォレットも更新されます。

- PKCS#12ウォレット(`ewallet.p12`)に関連付けられた自動ログイン・ウォレット(`cwallet.sso`)を作成するには、`orapki wallet create`コマンドを使用します。

```
orapki wallet create -wallet wallet_location -auto_login [-pwd wallet_password]
```

このコマンドでは、自動ログインが有効なウォレット(`cwallet.sso`)が作成され、PKCS#12ウォレット(`ewallet.p12`)に関連付けられます。コマンドラインでパスワードを指定していない場合、このコマンドではPKCS#12ウォレットのパスワードの入力を求められます。

`wallet_location`がPKCS#12ウォレットにすでに含まれている場合は、その自動ログインが有効になります。既存のPKCS#12ウォレットの自動ログインを有効にするには、そのパスワードを入力する必要があります。

`wallet_location`がPKCS#12ウォレットに含まれていない場合は、新しいPKCS#12ウォレットが作成されます。新しいPKCS#12ウォレットのパスワードを指定する必要があります。

PKCS#12ウォレットの自動ログイン機能をオフにする場合は、Oracle Wallet Managerを使用します。

関連項目:

詳細は、Oracle Databaseエンタープライズ・ユーザー・セキュリティ管理者ガイドを参照してください

親トピック: orapkiを使用したウォレットの作成、表示および変更

F.7.2.4 コンピュータとウォレット作成ユーザーにローカルな自動ログイン・ウォレットの作成

`orapki`ユーティリティを使用して、ウォレットを作成したユーザーのコンピュータにローカルな自動ログイン・ウォレットを作成できます。

- 自動ログイン・ウォレットが作成されるコンピュータとそのウォレットを作成したユーザーの両方にローカルな自動ログイン・ウォレットを作成するには、次のコマンドを使用します。

```
orapki wallet create -wallet wallet_location -auto_login_local [-pwd
wallet_password]
```

このコマンドにより、自動ログイン・ウォレット(`cwallet.sso`)が作成されます。これは、PKCS#12ウォレット(`ewallet.p12`)に関連付けられます。コマンドラインでパスワードを指定していない場合、このコマンドではPKCS#12ウォレットのパスワードの入力を求められます。

親トピック: orapkiを使用したウォレットの作成、表示および変更

F.7.2.5 ウォレットの表示

`orapki`ユーティリティを使用して、ウォレットを表示できます。

- Oracleウォレットを表示するには、`orapki wallet display`コマンドを使用します。

```
orapki wallet display -wallet wallet_location
```

このコマンドを使用すると、ウォレットに含まれている証明書リクエスト、ユーザー証明書および信頼できる証明書が表示されます(これらは拡張子.p12のバイナリPKCS12ファイルであることが必要です)。他のファイルでは失敗します。

親トピック: orapkiを使用したウォレットの作成、表示および変更

F.7.2.6 ウォレットのパスワードの変更

orapkiユーティリティを使用して、ウォレットのパスワードを変更できます。

- ウォレットのパスワードを変更するには、orapki wallet change_pwdコマンドを使用します。

```
orapki wallet change_pwd -wallet wallet_location [-oldpwd wallet_password ] [-
newpwd wallet_password]
```

このコマンドでは、現在のウォレットのパスワードが新しいパスワードに変更されます。コマンドラインでパスワードを指定していない場合、このコマンドでは新旧のパスワードを求められます。

ノート:

セキュリティ上の理由から、コマンドラインでパスワード・オプションを指定しないことをお薦めします。パスワードの入力は、求められたときに行ってください。

親トピック: orapkiを使用したウォレットの作成、表示および変更

F.7.2.7 AES256アルゴリズムの使用を目的としたOracleウォレットの変換

デフォルトでは、ADMINISTER KEY MANAGEMENT文またはALTER SYSTEM文を使用したOracleウォレットは、3DESで暗号化されます。

FIPS 140-2準拠の場合は、ウォレット暗号化アルゴリズム3DESをAES256に変更してください。orapki convertコマンドを使用して、3DESアルゴリズムよりも強力なAES256アルゴリズムを使用するようにウォレットを変換できます。
ADMINISTER KEY MANAGEMENT文やALTER SYSTEM文ではなくorapkiを使用してウォレットを作成した場合、そのウォレットはデフォルトでAES256アルゴリズムを使用します。

ノート:

このリリースでは、3DES112 および 3DES168 アルゴリズムは非推奨です。より強力なアルゴリズムを使用するように Oracle Database 環境を移行するには、My Oracle Support ノート 2118136.2 で説明されているパッチをダウンロードしてインストールします。

- ウォレット・アルゴリズムを3DESからAES256に変更するには、orapki wallet convertコマンドを使用します。

```
orapki wallet convert -wallet wallet_location [-pwd wallet_password] [-
compat_v12]
```

compat_v12設定は、3DESからAES256への変換を行います。

親トピック: orapkiを使用したウォレットの作成、表示および変更

F.7.3 orapkiを使用した証明書と証明書リクエストのOracleウォレットへの追加

orapkiユーティリティを使用して、様々な証明書関連タスクを実行できます。

- 証明書リクエストのOracleウォレットへの追加
  orapkiユーティリティを使用して、証明書および証明書リクエストをOracleウォレットに追加できます。
- 信頼できる証明書のOracleウォレットへの追加
  orapkiユーティリティを使用して、信頼できる証明書をOracleウォレットに追加できます。
- ルート証明書のOracleウォレットへの追加
  orapkiユーティリティを使用して、ルート証明書をOracleウォレットに追加できます。
- ユーザー証明書のOracleウォレットへの追加
  orapkiユーティリティを使用して、ユーザー証明書をOracleウォレットに追加できます。
- PKCS#11ウォレットを使用したハードウェア・デバイス上の資格証明の検証
  PKCS#11ウォレットを使用して、ハードウェア・デバイス上の資格証明を検証できます。
- PKCS#11情報のOracleウォレットへの追加
  他のOracleウォレットと同様に、PKCS#11情報を含むウォレットを使用できます。

親トピック: orapkiユーティリティを使用したOracleウォレットの管理

F.7.3.1 証明書リクエストのOracleウォレットへの追加

orapkiユーティリティを使用して、証明書および証明書リクエストをOracleウォレットに追加できます。

- Oracleウォレットに証明書リクエストを追加するには、orapki wallet addコマンドを使用します。

```
orapki wallet add -wallet wallet_location -dn user_dn -keySize 512|1024|2048
```

このコマンドを使用すると、指定した識別名(user_dn)を持つユーザーのウォレットに証明書リクエストが追加されます。このリクエストでは、リクエストされた証明書のキー・サイズ(512、1024または2048ビット)も指定します。リクエストに署名するには、エクスポート・オプションを使用してそのリクエストをエクスポートします。

関連トピック

- orapkiを使用した証明書と証明書リクエストのOracleウォレットからのエクスポート

親トピック: orapkiを使用した証明書と証明書リクエストのOracleウォレットへの追加

F.7.3.2 信頼できる証明書のOracleウォレットへの追加

orapkiユーティリティを使用して、信頼できる証明書をOracleウォレットに追加できます。

- Oracleウォレットに信頼できる証明書を追加するには、orapki wallet addコマンドを使用します。

```
orapki wallet add -wallet wallet_location -trusted_cert -cert
certificate_location
```

このコマンドを使用すると、指定した場所(-cert certificate_location)にある信頼できる証明書がウォレットに追加されます。ユーザー証明書を追加する前に、ユーザー証明書の証明連鎖にあるすべての信頼できる証明書を追加する必要があります。そうしないと、ユーザー証明書を追加するコマンドは失敗します。

親トピック: orapkiを使用した証明書と証明書リクエストのOracleウォレットへの追加

F.7.3.3 ルート証明書のOracleウォレットへの追加

orapkiユーティリティを使用して、ルート証明書をOracleウォレットに追加できます。

- Oracleウォレットにルート証明書を追加するには、`orapki wallet add`コマンドを使用します。

```
orapki wallet add -wallet wallet_location -dn certificate_dn -keySize
512|1024|2048 -self_signed -validity number_of_days
```

このコマンドを使用すると、新しい自己署名(ルート)証明書が作成され、ウォレットに追加されます。`-validity`パラメータ(必須)は、現在の日付から数えてこの証明書が有効である日数を指定します。このルート証明書のキー・サイズ(`-keySize`)は、512、1024または2048ビットに指定できます。

親トピック: orapkiを使用した証明書と証明書リクエストのOracleウォレットへの追加

F.7.3.4 ユーザー証明書のOracleウォレットへの追加

`orapki`ユーティリティを使用して、ユーザー証明書をOracleウォレットに追加できます。

- Oracleウォレットにユーザー証明書を追加するには、`orapki wallet add`コマンドを使用します。

```
orapki wallet add -wallet wallet_location -user_cert -cert certificate_location
```

このコマンドでは、`-cert`パラメータで指定された場所にあるユーザー証明書が、`wallet_location`にあるOracleウォレットに追加されます。ユーザー証明書をウォレットに追加する前に、証明連鎖を構成するすべての信頼できる証明書を追加する必要があります。ユーザー証明書を追加する前に、すべての信頼できる証明書がウォレットに追加されていない場合、ユーザー証明書の追加は失敗します。

ノート:

セキュリティ上の理由から、コマンドラインにパスワードを指定しないことをお薦めします。パスワードの入力は、求められたときに行ってください。

親トピック: orapkiを使用した証明書と証明書リクエストのOracleウォレットへの追加

F.7.3.5 PKCS#11ウォレットを使用したハードウェア・デバイス上の資格証明の検証

PKCS#11ウォレットを使用して、ハードウェア・デバイス上の資格証明を検証できます。

- 資格証明の詳細を検証するには、`orapki wallet p11_verify`コマンドを使用します。

```
orapki wallet p11_verify -wallet wallet_location [-pwd wallet_password]
```

親トピック: orapkiを使用した証明書と証明書リクエストのOracleウォレットへの追加

F.7.3.6 PKCS#11情報のOracleウォレットへの追加

他のOracleウォレットと同様に、PKCS#11情報を含むウォレットを使用できます。

秘密キーはハードウェア・デバイスに格納されます。暗号化処理もデバイスで実行されます。

- ウォレットにPKCS#11情報を追加するには、`orapki wallet p11_add`コマンドを使用します。

```
orapki wallet p11_add -wallet wallet_location -p11_lib pkcs11Lib
[-p11_tokenlabel tokenLabel] [-p11_tokenpw tokenPassphrase]
[-p11_certlabel certLabel] [-pwd wallet_password]
```

詳細は、次のとおりです。

- `wallet`では、ウォレット・ロケーションを指定します。

- p11_libでは、PKCS#11ライブラリへのパスを指定します。これにはライブラリのファイル名が含まれます。
- p11_tokenlabelでは、デバイスで使用されるトークンまたはスマートカードを指定します。これはデバイスに複数のトークンがある場合に使用します。トークンのラベルは、ベンダー・ツールを使用して設定します。
- p11_tokenpwでは、トークンへのアクセスに使用されるパスワードを指定します。トークンのパスワードは、ベンダー・ツールを使用して設定します。
- p11_certlabelは、トークン上の証明書ラベルを指定するために使用します。これはトークンに複数の証明書がある場合に使用します。証明書ラベルはベンダー・ツールを使用して設定します。
- pwdは、ウォレットのパスワードを指定するために使用します。

親トピック: orapkiを使用した証明書と証明書リクエストのOracleウォレットへの追加

F.7.4 orapkiを使用した証明書と証明書リクエストのOracleウォレットからのエクスポート

orapkiユーティリティを使用して、Oracleウォレットから証明書および証明書リクエストをエクスポートできます。

- Oracleウォレットから証明書をエクスポートするには、orapki wallet exportコマンドを使用します。

```
orapki wallet export -wallet wallet_location -dn certificate_dn -cert
certificate_filename
```

このコマンドを使用すると、サブジェクトの識別名(-dn)を持つ証明書が、ウォレットから-certで指定されたファイルにエクスポートされます。

Oracleウォレットから証明書リクエストをエクスポートするには、次のコマンドを使用します。

```
orapki wallet export -wallet wallet_location -dn certificate_request_dn -request
certificate_request_filename
```

このコマンドを使用すると、サブジェクトの識別名(-dn)を持つ証明書リクエストが、ウォレットから-requestで指定されたファイルにエクスポートされます。

親トピック: orapkiユーティリティを使用したOracleウォレットの管理

F.8 orapkiユーティリティを使用した証明書失効リスト(CRL)の管理

証明書失効リスト(CRL)は、orapkiユーティリティを使用して管理する必要があります。

このユーティリティは、CRL発行者名のハッシュ値を作成して、システム内でCRLの場所を特定します。orapkiを使用しないと、Oracleサーバーは、CRLを探してPKIデジタル証明書を検証することができません。

関連トピック

- 証明書失効リストの管理

親トピック: 公開キー・インフラストラクチャ(PKI)要素の管理

F.9 orapkiの使用方法

orapkiコマンドの例には、ウォレットおよびユーザー証明書の作成、自己署名証明書を含むウォレットおよび証明書のエクスポートがあります。

- 例: 自己署名証明書を含むウォレットおよび証明書のエクスポート
  orapki wallet addコマンドで自己署名証明書を含むウォレットを作成し、orapki wallet exportで証明書をエクスポートできます。
- 例: ウォレットおよびユーザー証明書の作成
  orapkiユーティリティで、ウォレットおよびユーザー証明書を作成できます。

親トピック: 公開キー・インフラストラクチャ(PKI)要素の管理

F.9.1 例: 自己署名証明書を含むウォレットおよび証明書のエクスポート

orapki wallet addコマンドで自己署名証明書を含むウォレットを作成し、orapki wallet exportで証明書をエクスポートできます。

例F-1では、自己署名証明書を含むウォレットを作成し、そのウォレットを表示して証明書をファイルにエクスポートするステップを示します。

例F-1 自己署名証明書を含むウォレットの作成と証明書のエクスポート

1. ウォレットを作成します。

   たとえば:

   ```
   orapki wallet create -wallet /private/user/orapki_use/root
   ```

   ウォレットは、/private/user/orapki_use/rootに作成されます。

2. 自己署名証明書をウォレットに追加します。

   ```
   orapki wallet add -wallet /private/user/orapki_use/root -dn
   'CN=root_test,C=US' -keysize 2048 -self_signed -validity 3650
   ```

   この結果、3650日の有効期限を持つ自己署名された証明書が作成されます。サブジェクトの識別名はCN=root_test,C=USです。証明書のキー・サイズは2048ビットです。

3. ウォレットを表示します。

   ```
   orapki wallet display -wallet /private/user/orapki_use/root
   ```

   これはウォレットに含まれる証明書を表示するために使用します。

4. 証明書をエクスポートします。

   ```
   orapki wallet export -wallet /private/user/orapki_use/root -dn
   'CN=root_test,C=US' -cert /private/user/orapki_use/root/b64certificate.txt
   ```

   これにより、自己署名証明書がファイルb64certificate.txtにエクスポートされます。使用される識別名はステップ2と同じであることに注意してください。

親トピック: orapkiの使用方法

F.9.2 例: ウォレットおよびユーザー証明書の作成

orapkiユーティリティで、ウォレットおよびユーザー証明書を作成できます。

例F-2では、ユーザー証明書の作成に関連する様々な作業について説明します。

次に、ウォレットの作成、証明書リクエストの作成、証明書リクエストのエクスポート、テスト用のリクエストからの署名付き証明書

の作成、証明書の表示、ウォレットへの信頼できる証明書の追加、およびウォレットへのユーザー証明書の追加のステップを示します。

例F-2 ウォレットおよびユーザー証明書の作成

1. 自動ログインが有効なウォレットを作成します。

   次に例を示します。

   ```
   orapki wallet create -wallet /private/user/orapki_use/server -auto_login
   ```

   これにより、自動ログインを有効にしたウォレットが/private/user/orapki_use/serverに作成されます。

2. 証明書リクエストをウォレットに追加します。

   ```
   orapki wallet add -wallet /private/user/orapki_use/server/ewallet.p12 -dn
   'CN=server_test,C=US' -keysize 2048
   ```

   これにより、作成されたウォレット(ewallet.p12)に証明書リクエストが追加されます。サブジェクトの識別名はCN=server_test,C=USです。指定されたキー・サイズは2048ビットです。

3. 証明書リクエストをファイルにエクスポートします。

   ```
   orapki wallet export -wallet /private/user/orapki_use/server -dn
   'CN=server_test,C=US' -request /private/user/orapki_use/server/creq.txt
   ```

   これにより、指定されたファイル(この場合はcreq.txt)に証明書リクエストがエクスポートされます。

4. テスト用のリクエストからの署名付き証明書を作成します。

   ```
   orapki cert create -wallet /private/user/orapki_use/root -request
   /private/user/orapki_use/server/creq.txt -cert
   /private/user/orapki_use/server/cert.txt -validity 3650
   ```

   これにより、3650日の有効期限を持つ証明書cert.txtが作成されます。証明書は前のステップで生成された証明書リクエストから作成されます。

5. 証明書を表示します。

   ```
   orapki cert display -cert /private/user/orapki_use/server/cert.txt -complete
   ```

   これにより、前のステップで生成された証明書が表示されます。-completeオプションでは、シリアル番号や公開キーなどの追加的な証明書情報を表示できます。

6. 信頼できる証明書をウォレットに追加します。

   ```
   orapki wallet add -wallet /private/user/orapki_use/server/ewallet.p12 -
   trusted_cert -cert /private/user/orapki_use/root/b64certificate.txt
   ```

   これにより、信頼できる証明書b64certificate.txtがewallet.p12ウォレットに追加されます。ユーザー証明書を追加する前に、ユーザー証明書の証明連鎖内のすべての信頼できる証明書を追加する必要があります。

7. ユーザー証明書をウォレットに追加します。

   ```
   orapki wallet add -wallet /private/user/orapki_use/server/ewallet.p12 -
   user_cert -cert /private/user/orapki_use/server/cert.txt
   ```

   このコマンドはユーザー証明書cert.txtをewallet.p12ウォレットに追加します。

親トピック: orapkiの使用方法

F.10 orapkiユーティリティ・コマンドのサマリー

orapkiコマンドは、ウォレット、証明書失効リスト(CRL)および証明書の様々な管理タスクを実行します。

- orapki cert create
  orapki cert createコマンドは、テスト用の署名付き証明書を作成します。
- orapki cert display
  orapki cert displayコマンドは、特定の証明書の詳細を表示します。
- orapki crl deleteコマンド
  orapki crl deleteコマンドは、Oracle Internet Directoryから証明書失効リスト(CRL)を削除します。
- orapki crl display
  orapki crl displayコマンドは、Oracle Internet Directoryに格納されている、指定した証明書失効リスト(CRL)を表示します。
- orapki crl hash
  orapki crl hashコマンドは、証明書失効リスト(CRL)発行者のハッシュ値を生成して、証明書検証用にCRLファイル・システムの場所を特定します。
- orapki crl list
  orapki crl listコマンドは、Oracle Internet Directoryに格納されている証明書失効リスト(CRL)のリストを表示します。
- orapki crl upload
  orapki crl uploadコマンドは、証明書失効リスト(CRL)をOracle Internet DirectoryのCRLサブツリーにアップロードします。
- orapki wallet add
  orapki wallet addコマンドは、証明書リクエストおよび証明書をOracleウォレットに追加します。
- orapki wallet convert
  orapki wallet convertコマンドは、Oracleウォレットの暗号化アルゴリズムを3DESからAES256に変更します。
- orapki wallet create
  orapki wallet createコマンドは、Oracleウォレットの作成またはOracleウォレットの自動ログインの有効化を行います。
- orapki wallet display
  orapki wallet displayコマンドは、Oracleウォレット内の証明書リクエスト、ユーザー証明書および信頼できる証明書を表示します。
- orapki wallet export
  orapki wallet exportコマンドは、証明書リクエストおよび証明書をOracleウォレットからエクスポートします。

親トピック: 公開キー・インフラストラクチャ(PKI)要素の管理

F.10.1 orapki cert create

orapki cert createコマンドは、テスト用の署名付き証明書を作成します。

構文

```
orapki cert create [-wallet wallet_location] -request certificate_request_location -
cert certificate_location -validity number_of_days [-summary]
```

- walletは、証明書リクエストへの署名に使用されるユーザー証明書と秘密キーを含むウォレットを指定します。

- `request` (必須)は、作成する証明書の証明書リクエストの場所を指定します。
- `cert` (必須)は、ツールによって新しい署名付き証明書が配置されるディレクトリの場所を指定します。
- `validity` (必須)は、現在の日付から数えてこの証明書が有効である日数を指定します。

親トピック: orapkiユーティリティ・コマンドのサマリー

F.10.2 orapki cert display

`orapki cert display`コマンドは、特定の証明書の詳細を表示します。

構文

```
orapki cert display -cert certificate_location [-summary|-complete]
```

- `cert`は、表示する証明書の場所を指定します。
- `-summary`パラメータまたは`-complete`パラメータのいずれかを使用して、次の情報を表示できます。
  - `summary`を使用すると、証明書およびその有効期限が表示されます。
  - `complete`を使用すると、シリアル番号、公開キーなどの追加の証明書情報が表示されます。

親トピック: orapkiユーティリティ・コマンドのサマリー

F.10.3 orapki crl deleteコマンド

`orapki crl delete`コマンドは、Oracle Internet Directoryから証明書失効リスト(CRL)を削除します。

`orapki`を使用してディレクトリからCRLを削除するユーザーは、`CRLAdmins` (`cn=CRLAdmins,cn=groups,%s_OracleContextDN%`)ディレクトリ・グループのメンバーである必要があります。

前提条件

なし

構文

```
orapki crl delete -issuer issuer_name -ldap hostname:ssl_port -user username [-wallet
wallet_location] [-summary]
```

- `issuer`は、CRLを発行した認証局(CA)の名前を指定します。
- `ldap`は、CRLを削除するディレクトリのホスト名とSSLポートを指定します。このポートは、認証を使用しないディレクトリのSSLポートであることが必要です。

  このポートの詳細は、Oracle Internet DirectoryへのCRLのアップロードも参照してください。
- `user`は、ディレクトリのCRLサブツリーからCRLを削除する権限のあるディレクトリ・ユーザーのユーザー名を指定します。
- `wallet` (オプション)は、CRLを発行した認証局(CA)の証明書を含むウォレットの場所を指定します。これを使用すると、CRLをディレクトリから削除する前に、ツールによってCAの証明書に対するCRLの有効性が検証されます。
- `summary`はオプションです。削除されたCRL LDAPエントリが表示されます。

親トピック: orapkiユーティリティ・コマンドのサマリー

F.10.4 orapki crl display

`orapki crl display`コマンドは、Oracle Internet Directoryに格納されている、指定した証明書失効リスト(CRL)を表示します。

構文

```
orapki crl display -crl crl_location [-wallet wallet_location] [-summary|-complete]
```

- `crl`パラメータは、ディレクトリ内のCRLの場所を指定します。`orapki crl list`コマンドを使用すると表示されるリストからCRLの位置を貼り付けると便利です。orapki crl listを参照してください。
- `wallet` (オプション)は、CRLを発行した認証局(CA)の証明書を含むウォレットの場所を指定します。これを使用すると、CRLを表示する前に、ツールによってCAの証明書に対するCRLの有効性が確認されます。
- `summary`および`complete`は、次の情報を表示します。
  - `summary`を選択すると、CRL発行者名およびCRLの有効期間を含むリストが表示されます。
  - `complete`を選択すると、そのCRLに含まれるすべての失効した証明書のリストが表示されます。このオプションを選択すると、CRLのサイズによっては、表示に時間がかかる場合があることに注意してください。

親トピック: orapkiユーティリティ・コマンドのサマリー

F.10.5 orapki crl hash

`orapki crl hash`コマンドは、証明書失効リスト(CRL)発行者のハッシュ値を生成して、証明書検証用にCRLファイル・システムの場所を特定します。

構文

```
orapki crl hash -crl crl_filename|URL [-wallet wallet_location] [-symlink|-copy]
crl_directory [-summary]
```

- `crl`は、CRLまたはCRLがあるURLを含むファイル名を指定します。
- `wallet` (オプション)は、CRLを発行した認証局(CA)の証明書を含むウォレットの場所を指定します。これを使用すると、CRLをディレクトリにアップロードする前に、ツールによってCAの証明書に対するCRLの有効性が確認されます。
- オペレーティング・システムに応じて、`-symlink`パラメータまたは`-copy`パラメータのいずれかを使用します。
  - (UNIX) `symlink`は、`crl_directory`の場所にCRLへのシンボリック・リンクを作成します。
  - (Windows) `copy`は、`crl_directory`の場所にCRLのコピーを作成します。
- `summary` (オプション)は、CRL発行者の名前を表示します。

親トピック: orapkiユーティリティ・コマンドのサマリー

F.10.6 orapki crl list

`orapki crl list`コマンドは、Oracle Internet Directoryに格納されている証明書失効リスト(CRL)のリストを表示します。

構文

特定のCRLを検出してローカル・ファイル・システムで表示またはダウンロードする際に、このリストを参照すると便利です。

```
orapki crl list -ldap hostname:ssl_port
```

ldapは、CRLリストの作成対象のディレクトリ・サーバーのホスト名およびSSLポートを指定します。このポートは、認証を使用しないディレクトリのSSLポートであることが必要です。

関連項目:

このポートの詳細は、Oracle Internet DirectoryへのCRLのアップロードを参照してください

親トピック: orapkiユーティリティ・コマンドのサマリー

F.10.7 orapki crl upload

orapki crl uploadコマンドは、証明書失効リスト(CRL)をOracle Internet DirectoryのCRLサブツリーにアップロードします。

CRLをディレクトリにアップロードするには、ディレクトリ管理グループCRLAdmins (cn=CRLAdmins,cn=groups,%s_OracleContextDN%)のメンバーである必要があることに注意してください。

構文

```
orapki crl upload -crl crl_location -ldap hostname:ssl_port -user username [-wallet wallet_location] [-summary]
```

- crlは、ディレクトリの場所またはディレクトリにアップロードするCRLが配置されているURLを指定します。
- ldapは、CRLのアップロード先のディレクトリのホスト名およびSSLポートを指定します。このポートは、認証を使用しないディレクトリのSSLポートであることが必要です。

  このポートの詳細は、Oracle Internet DirectoryへのCRLのアップロードも参照してください。
- userは、ディレクトリのCRLサブツリーにCRLを追加する権限のあるディレクトリ・ユーザーのユーザー名を指定します。
- walletは、CRLを発行した認証局(CA)の証明書を含むウォレットの場所を指定します。これはオプションのパラメータです。これを使用すると、CRLをディレクトリにアップロードする前に、ツールによってCAの証明書に対するCRLの有効性が確認されます。
- summaryはオプションです。これを使用すると、CRL発行者名、およびCRLがディレクトリに格納されているLDAPエントリが表示されます。

親トピック: orapkiユーティリティ・コマンドのサマリー

F.10.8 orapki wallet add

orapki wallet addコマンドは、証明書リクエストおよび証明書をOracleウォレットに追加します。

構文

証明書リクエストを追加するには:

```
orapki wallet add -wallet wallet_location -dn user_dn -keySize 512|1024|2048
```

- walletは、証明書リクエストの追加先のウォレットの場所を指定します。
- dnは、証明書の所有者の識別名を指定します。
- keysizeは、証明書のキー・サイズを指定します。

- リクエストに署名するには、エクスポート・オプションを使用してそのリクエストをエクスポートします。orapki wallet exportを参照してください

信頼できる証明書を追加するには:

```
orapki wallet add -wallet wallet_location -trusted_cert -cert certificate_location
```

- trusted_certは、-certで指定された場所にある信頼できる証明書をウォレットに追加します。

ルート証明書を追加するには:

```
orapki wallet add -wallet wallet_location -dn certificate_dn -keySize 512|1024|2048 -
self_signed -validity number_of_days
```

- self_signedは、ルート証明書を追加します。
- validityは必須です。これを使用して、現在の日付から数えてこのルート証明書が有効である日数を指定します。

ユーザー証明書を追加するには:

```
orapki wallet add -wallet wallet_location -user_cert -cert certificate_location
```

- user_certは、-certパラメータで指定された場所にあるユーザー証明書をウォレットに追加します。ユーザー証明書をウォレットに追加する前に、証明連鎖を構成するすべての信頼できる証明書を追加する必要があります。ユーザー証明書を追加する前に、すべての信頼できる証明書がウォレットに追加されていない場合、ユーザー証明書の追加は失敗します。

親トピック: orapkiユーティリティ・コマンドのサマリー

F.10.9 orapki wallet convert

orapki wallet convertコマンドは、Oracleウォレットの暗号化アルゴリズムを3DESからAES256に変更します。

構文

```
orapki wallet convert -wallet wallet_location [-pwd wallet_password] [-compat_v12]
```

- walletには、新しいウォレットの場所または自動ログインを有効にするウォレットの場所を指定します。
- pwdはウォレット・パスワードです。
- compat_v12は、3DESからAES256への変換を行います。

親トピック: orapkiユーティリティ・コマンドのサマリー

F.10.10 orapki wallet create

orapki wallet createコマンドは、Oracleウォレットの作成またはOracleウォレットの自動ログインの有効化を行います。

構文

```
orapki wallet create -wallet wallet_location [-auto_login|-auto_login_local]
```

- walletには、新しいウォレットの場所または自動ログインを有効にするウォレットの場所を指定します。
- auto_loginにより、自動ログイン・ウォレットが作成されるか、または、-walletオプションで指定されたウォレットの自動ログインが有効になります。

  自動ログイン・ウォレットの詳細は、『Oracle Databaseエンタープライズ・ユーザー・セキュリティ管理者ガイド』を参照

してください。

- auto_login_localにより、ローカル自動ログイン・ウォレットが作成されるか、または、-walletオプションで指定されたウォレットのローカル自動ログインが有効になります。

親トピック: orapkiユーティリティ・コマンドのサマリー

F.10.11 orapki wallet display

orapki wallet displayコマンドは、Oracleウォレット内の証明書リクエスト、ユーザー証明書および信頼できる証明書を表示します。

構文

```
orapki wallet display -wallet wallet_location
```

- walletは、開くウォレットの場所を指定します(そのウォレットが現在の作業ディレクトリにない場合)。

親トピック: orapkiユーティリティ・コマンドのサマリー

F.10.12 orapki wallet export

orapki wallet exportコマンドは、証明書リクエストおよび証明書をOracleウォレットからエクスポートします。

構文

証明書をOracleウォレットからエクスポートするには:

```
orapki wallet export -wallet wallet_location -dn certificate_dn -cert
certificate_filename
```

- walletは、証明書のエクスポート元のウォレットの場所を指定します。
- dnは、証明書の識別名を指定します。
- certは、エクスポートされる証明書を含むファイルの名前を指定します。

証明書リクエストをOracleウォレットからエクスポートするには:

```
orapki wallet export -wallet wallet_location -dn certificate_request_dn -request
certificate_request_filename
```

- requestは、エクスポートされる証明書リクエストを含むファイルの名前を指定します。

親トピック: orapkiユーティリティ・コマンドのサマリー

G 統合監査の移行による各監査機能への影響

統合監査の移行前に、Oracle Database 12cリリース1 (12.1)以前の監査機能の大部分を使用できます。

表G-1では、移行によるOracle Database 12cより前の監査機能の変更内容について説明します。

表G-1 移行前後での統合監査機能の可用性

機能	移行前の環境での可用性	移行後の環境での可用性
一般的な監査機能	-	-
オペレーティング・システム監査証跡	はい	いいえ
XML ファイル監査証跡	はい	いいえ
ネットワーク監査	はい	いいえ
ユーザーが監査および監査を独自のスキーマ・オブジェクトから削除できるかどうか	はい	いいえ
監査の管理アクションの必須の監査	いいえ	はい
ロールの監査	-	-
AUDIT_ADMIN	○(ただし、自身のオブジェクトを監査するユーザー、または ALTER SYSTEM 権限がすでにあり、監査初期化パラメータを変更するユーザーには不要)	はい
AUDIT_VIEWER	はい	はい
システム表	-	-
SYS.AUD$	はい	○(ただし、統合前の監査レコードのみを含む)
SYS.FGA_LOG$	はい	○(ただし、統合前の監査レコードのみを含む)
初期化パラメータ	-	-

機能	移行前の環境での可用性	移行後の環境での可用性
AUDIT_TRAIL	はい	○(ただし、影響はない)
AUDIT_FILE_DEST	はい	○(ただし、影響はない)
AUDIT_SYS_OPERATIONS	はい	○(ただし、影響はない)
AUDIT_SYSLOG_LEVEL	はい	○(ただし、影響はない)
UNIFIED_AUDIT_SGA_QUEUE_SIZE	はい(ただし、このパラメータは非推奨であり、現在は下位互換性のために維持されている)。	はい(ただし、このパラメータは非推奨であり、現在は下位互換性のために維持されている)。
データ・ディクショナリ・ビュー 脚注 1	-	-
ALL_AUDIT_POLICIES	はい	○(ただし、DBMS_FGA PL/SQL パッケージを使用してファイングレイン監査ポリシーが作成される場合のみ)
DBA_AUDIT_POLICIES	はい	○(ただし、DBMS_FGA PL/SQL パッケージを使用してファイングレイン監査ポリシーが作成される場合のみ)
DBA_AUDIT_POLICY_COLUMNS	はい	○(ただし、DBMS_FGA PL/SQL パッケージを使用してファイングレイン監査ポリシーが作成される場合のみ)
DBA_COMMON_AUDIT_TRAIL	はい	○(ただし、統合前の監査レコードのみを含む)
DBA_AUDIT_EXISTS	はい	はい
DBA_AUDIT_OBJECT	はい	はい
DBA_AUDIT_POLICIES	はい	○(ただし、DBMS_FGA PL/SQL パッケージを使用してファイングレイン監査ポリシーが作成される場合のみ)
DBA_AUDIT_POLICY_COLUMNS	はい	○(ただし、DBMS_FGA PL/SQL パッケージを使用してファイングレイン監査

機能	移行前の環境での可用性	移行後の環境での可用性
		ポリシーが作成される場合のみ)
DBA_AUDIT_SESSION	はい	○(ただし、統合前の監査レコードのみを含む)
DBA_AUDIT_STATEMENT	はい	○(ただし、統合前の監査レコードのみを含む)
DBA_AUDIT_TRAIL	はい	○(ただし、統合前の監査レコードのみを含む)RLS_INFO 列は、監査済の Oracle VPD 述語を取得します。
DBA_FGA_AUDIT_TRAIL	はい	○(ただし、統合前の監査レコードのみを含む)RLS_INFO 列は、監査済の Oracle VPD 述語を取得します。
DBA_OBJ_AUDIT_OPTS	はい	はい
DBA_PRIV_AUDIT_OPTS	はい	はい
DBA_STMT_AUDIT_OPTS	はい	はい
UNIFIED_AUDIT_TRAIL	○(ただし、監査レコードは収集しない)	○(監査レコードを収集する)
USER_AUDIT_OBJECT	はい	はい
USER_AUDIT_POLICY_COLUMN	はい	○(ただし、DBMS_FGA PL/SQL パッケージを使用してファイングレイン監査ポリシーが作成される場合のみ)
USER_AUDIT_POLICIES	はい	○(ただし、DBMS_FGA PL/SQL パッケージを使用してファイングレイン監査ポリシーが作成される場合のみ)
USER_AUDIT_SESSION	はい	はい
USER_AUDIT_STATEMENT	はい	はい

機能	移行前の環境での可用性	移行後の環境での可用性
USER_AUDIT_TRAIL	はい	○(ただし、統合前の監査レコードのみを含む)
USER_OBJ_AUDIT_OPTS	はい	はい
V$XML_AUDIT_TRAIL	はい	○(ただし、統合前の監査レコードのみを含む)RLS_INFO 列は、監査済の Oracle VPD 述語を取得します。
CREATE AUDIT POLICY、ALTER AUDIT POLICY および DROP AUDIT POLICY 文	文は使用できますが、監査ポリシーは古い監査証跡に書き込まれません。ポリシーが有効な場合、監査レコードは統合監査証跡に書き込まれます。	はい(ただし、監査レコードを統合監査証跡のみに書き込む)
AUDIT 文および NOAUDIT 文	-	-
AUDIT	○(マルチテナント環境で使用可能)	○(ただし、監査ポリシーを有効化、アプリケーション・コンテキスト監査設定を作成、成功または失敗(あるいは両方)時に監査レコードを作成、およびマルチテナント環境で使用できるように拡張)
NOAUDIT	○(マルチテナント環境で使用可能)	○(ただし、監査ポリシーを無効化、アプリケーション・コンテキスト監査設定を無効化、およびマルチテナント環境で使用できるように変更)
DBMS_FGA.ADD_POLICY プロシージャ・パラメータ	-	-
audit_trail	○(以前のリリースで使用)	○(ただし、すべてのレコードが統合監査証跡に書き込まれるため、統合監査が有効な場合は、このパラメータを省略可能)
DBMS_AUDIT_MGMT パッケージの AUDIT_TRAIL_TYPE プロパティ・オプション	-	-

機能	移行前の環境での可用性	移行後の環境での可用性
DBMS_AUDIT_MGMT.AUDIT_TRAIL_AUD_STD	はい	○(ただし、統合前の監査レコードのみ)
DBMS_AUDIT_MGMT.AUDIT_TRAIL_FGA_STD	はい	○(ただし、統合前の監査レコードのみ)
DBMS_AUDIT_MGMT.AUDIT_TRAIL_DB_STD	はい	○(ただし、統合前の監査レコードのみ)
DBMS_AUDIT_MGMT.AUDIT_TRAIL_OS	はい	○(ただし、統合前の監査レコードのみ)
DBMS_AUDIT_MGMT.AUDIT_TRAIL_XML	はい	○(ただし、統合前の監査レコードのみ)
DBMS_AUDIT_MGMT.AUDIT_TRAIL_FILES	はい	○(ただし、統合前の監査レコードのみ)
DBMS_AUDIT_MGMT.AUDIT_TRAIL_ALL	はい	○(ただし、統合前の監査レコードのみ)
Oracle Database Vault 機能	-	-
DVSYS.AUDIT_TRAIL$システム表	はい	名前をDVSYS.OLD_AUDIT_TRAIL$に変更し、古い監査レコードを保持しています。前のDVSYS.AUDIT_TRAIL$表はDVSYS.AUDIT_TRAIL$という名前のビューに変更されています。新しい監査レコードは追加されません。
Oracle Label Security 機能	-	-
SA_AUDIT_ADMIN PL/SQL パッケージ	はい	いいえ

脚注1

これらのデータ・ディクショナリ・ビューには、SYS.AUD$システム表およびSYS.FGA_LOG$システム表に存在する監査レコードの

監査データが引き続き表示されます。統合監査証跡レコードは、統合監査証跡固有のビューにのみ表示されます。先頭がUSER_でないビューを問い合せるには、AUDIT_ADMINまたはAUDIT_VIEWERロールを付与する必要があります。

親トピック: 付録

用語集

アクセス制御

特定のクライアントまたはクライアント・グループに対して特定のデータへのアクセス権を付与または制限するシステム機能。

親トピック: 用語集

アクセス制御リスト(ACL)

ユーザーが定義するアクセス・ディレクティブのグループ。ディレクティブは、特定のクライアント、クライアント・グループまたはその両方に対して特定のデータへのアクセス・レベルを付与する。

親トピック: 用語集

Advanced Encryption Standard

Advanced Encryption Standard (AES)は、米国商務省国立標準技術研究所によってDESに代わるものとして承認された新しい暗号アルゴリズムである。(DESはこのリリースでは非推奨になりました。より強力なアルゴリズムを使用するようにOracle Database環境を移行するには、My Oracle Supportノート2118136.2で説明されているパッチをダウンロードしてインストールします。) AES標準は、Federal Information Processing Standards Publication 197にあります。AESアルゴリズムは、128、192および256ビットの長さの暗号キーを使用して、128ビットのデータ・ブロックを処理できる対称型ブロック暗号です。

親トピック: 用語集

AES

「Advanced Encryption Standard」を参照。

親トピック: 用語集

アプリケーション・コンテキスト

名前と値のペア。このペアによって、アプリケーションは、ユーザーに関するセッション情報(ユーザーIDや他のユーザー固有の情報など)にアクセスして、その情報をデータベースに安全に引き渡すことができます。

「グローバル・アプリケーション・コンテキスト」も参照してください。

親トピック: 用語集

属性

LDAPディレクトリ内のエントリの性質を説明する情報項目。エントリは属性のセットから構成され、それぞれの属性がオブジェクト・クラスに属する。さらに、各属性にはタイプと値があり、タイプは属性の情報の種類を説明し、値には実際のデータが格納されています。

親トピック: 用語集

アプリケーション・ロール

アプリケーション・ユーザーに付与されるデータベース・ロール。アプリケーション内に埋め込まれているパスワードによって保護されています。

「セキュア・アプリケーション・ロール」も参照してください。

親トピック: 用語集

認証

コンピュータ・システムのユーザー、デバイスまたはその他のエンティティの識別情報を検証するプロセスであり、システムのリソースへのアクセス権を付与するための前提条件となることが多い。認証されたメッセージの受信者は、そのメッセージの発信元(送信者)を確認できる。認証は、第三者が送信者を装っている可能性を排除するものとみなされる。

親トピック: 用語集

認証方式

分散環境のユーザー、クライアントまたはサーバーの識別情報を検証するセキュリティ方式。ネットワーク認証方式では、ユーザーにシングル・サインオン(SSO)の利点も提供できる。次の認証方式がサポートされています:

- Kerberos
- RADIUS
- Transport Layer Security (TLS)
- Windowsネイティブ認証

親トピック: 用語集

認可

オブジェクトまたはオブジェクトのセットにアクセスするためにユーザー、プログラムまたはプロセスに付与される権限。Oracleでは、認可はロール・メカニズムを通じて行われる。1つのユーザーまたはユーザー・グループに対して、1つのロールまたは一連のロールを付与できる。また、ロールに他のロールを付与することもできる。認証されたエンティティが利用できる権限のセット。

親トピック: 用語集

自動ログイン・ウォレット

アクセス時に資格証明を発行しない、サービスへのパスワードベースのアクセスです。自動ログイン・アクセスは、自動ログイン機能がウォレットに対して無効になるまで有効です。ファイル・システム権限では、ウォレットの自動ログインに必要なセキュリティが提供されます。ウォレットに対して自動ログインが有効になっている場合は、そのウォレットを作成したオペレーティング・システム・ユーザーのみ、そのウォレットを使用できます。これらはシングル・サイン・オン機能を提供するため、SSOウォレットと呼ばれることもあります。

親トピック: 用語集

CDB

マルチテナント・コンテナ・データベース。1つのルートと0(ゼロ)個以上のプラガブル・データベース(PDB)を含むOracle Databaseインストール。すべてのOracle Databaseは、CDBか非CDBのいずれかです。

親トピック: 用語集

ベース

LDAP準拠ディレクトリでのサブツリー検索のルート。

親トピック: 用語集

CA

「認証局」を参照してください。

親トピック: 用語集

証明書

識別情報を公開キーに安全にバインドするITU x.509 v3標準データ構造。

証明書は、エンティティの公開キーが、信頼されている機関(認証局)によって署名されたときに作成されます。この証明書は、そのエンティティの情報が正しいこと、および公開キーがそのエンティティに属していることを保証します。

証明書には、エンティティの名前、識別情報および公開キーが含まれます。シリアル番号、有効期限、および証明書に付随する権利、使用および権限も含まれることがあります。最後に、その証明書を発行した認証局に関する情報が含まれます。

親トピック: 用語集

認証局

他のエンティティ(ユーザー、データベース、管理者、クライアント、サーバー)が本物であることを証明する、信頼できる第三者。ユーザーを証明するとき、認証局は最初にそのユーザーが証明書失効リスト(CRL)に掲載されていないことを確認してからそのユーザーのアイデンティティを検証し、証明書を付与し、認証局の秘密キーを使用してその証明書に署名します。認証局には自身の証明書と公開キーがあり、公開されています。サーバーおよびクライアントは、これらを使用して認証局の署名を検証します。認証局は、証明書サービスを提供する外部の会社の場合や、企業のMIS部門のような内部の組織の場合があります。

親トピック: 用語集

証明連鎖

エンド・ユーザーまたはサブスクライバの証明書とその認証局の証明書を含む順序付きの証明書リスト。

親トピック: 用語集

証明書リクエスト

認証リクエスト情報、署名アルゴリズム識別子および認証リクエスト情報に対するデジタル署名の3つの部分から構成される証明書リクエスト。認証リクエスト情報は、対象の識別名、公開キーおよびオプションの属性セットから構成される。属性では、対

象の識別情報に関する郵便の宛先などの追加情報、または対象エンティティが後で証明書失効を要求するためのチャレンジ・パスワードを提供できる。「PKCS #10」を参照してください。

親トピック: 用語集

証明書失効リスト(CRL)

(CRL)失効した証明書 のリストを含む署名付きデータ構造。CRLの信頼性と整合性は、CRLに付加されているデジタル署名によって提供される。通常、CRLの署名者は、発行された証明書に署名したエンティティと同じである。

親トピック: 用語集

チェックサム

メッセージ・パケットに含まれているデータに基づいてメッセージ・パケットの値を計算し、その値をデータとともに渡して、データが書き換えられていないことを認証するメカニズム。データの受信者は暗号チェックサムを再計算して、それをデータとともに送られた暗号チェックサムと比較します。これらが一致している場合は、データが送信中に書き換えられなかったことの確率的な証明になります。

親トピック: 用語集

クリアテキスト

暗号化されていない平文。

親トピック: 用語集

暗号ブロック連鎖(CBC)

暗号化メソッドの1つ。先行するすべてのブロックに従って暗号ブロックの暗号化を行い、ブロック再生攻撃からデータを保護します。無許可の復号化が段階的に困難になるように設計されています。Oracle Databaseでは、外部暗号ブロック連鎖が使用されています。これは、内部暗号ブロックよりも安全性が高く、実質的なパフォーマンスの低下を伴わないためです。

親トピック: 用語集

CIDR

IPアドレスに使用する標準の表記法。CIDR表記法では、IPv6サブネットは、サブネット接頭辞およびビットで示した接頭辞のサイズ(小数)がスラッシュ文字(/)で区切られて示されます。たとえば、`fe80:0000:0217:f2ff::/64`は、アドレス`fe80:0000:0217:f2ff:0000:0000:0000:0000`から`fe80:0000:0217:f2ff:ffff:ffff:ffff:ffff`までのサブネットを示します。CIDR表記法には、IPv4アドレスのサポートが含まれます。たとえば、`192.0.2.1/24`は、アドレス`192.0.2.1`から`192.0.2.255`までのサブネットを示します。

親トピック: 用語集

暗号スイート

ネットワーク・ノード間でメッセージを交換するために使用される認証、暗号化およびデータ整合性のアルゴリズムのセット。たとえば、TLSハンドシェイク中に2つのノードがネゴシエーションして、メッセージの送受信中に使用する暗号スイートを確認する。

親トピック: 用語集

暗号スイート名

暗号スイートは、特定のセッションで接続により使用される暗号保護の種類を示す。

親トピック: 用語集

暗号文

暗号化されたメッセージ・テキスト。

親トピック: 用語集

クラスレス・ドメイン間ルーティング

「CIDR」を参照してください。

親トピック: 用語集

クライアント

サービスを利用する側。クライアントはユーザーの場合や、データベース・リンク中にユーザーとして機能するプロセス(プロキシともいう)の場合がある。

親トピック: 用語集

共通権限付与

共通ユーザーが、他の共通ユーザーまたは共通ロールに付与する権限。共通権限付与は、システム権限またはオブジェクト権限のいずれかで、CDBのすべてのPDBに適用されます。

「ローカル権限付与」も参照。

親トピック: 用語集

共通ロール

CDB内のすべてのコンテナに存在するロール。

親トピック: 用語集

共通ユーザー

CDBで、既存および将来のすべてのPDBに同じIDで存在するデータベース・ユーザー。

親トピック: 用語集

機密保護

暗号化の機能。機密保護によって、メッセージを参照(暗号文を復号化する)できるのはメッセージの本来の受信者のみであることが保証される。

親トピック: 用語集

接続記述子

特別にフォーマットされた、ネットワーク接続のための宛先の記述。接続記述子には、接続先のサービスおよびネットワーク・ルート情報が含まれます。接続先サービスは、Oracle9i またはOracle8i データベースのサービス名か、Oracleデータベース・リリース8.0のOracle システム識別子(SID)を使用して指定されます。ネットワーク・ルートは、少なくとも、ネットワーク・アドレスを使用してリスナーの場所を示します。「接続識別子」を参照してください。

親トピック: 用語集

接続識別子

接続記述子を解決する名前、ネット・サービス名またはサービス名。次のように、ユーザーは、接続するサービスに対して、接続文字列内の接続識別子とともにユーザー名とパスワードを渡して、接続要求を実行します。

たとえば:

```
CONNECT username@connect_identifier
Enter password: password
```

親トピック: 用語集

接続文字列

ユーザー名、パスワード、ネット・サービス名など、ユーザーが接続先のサービスに渡す情報。たとえば:

```
CONNECT username@net_service_name
Enter password: password
```

親トピック: 用語集

コンテナ

CDBで、ルートまたはPDB。

親トピック: 用語集

コンテナ・データ・オブジェクト

CDBでは、複数のコンテナと場合によってはCDB全体に関連するデータを、そのようなオブジェクトで特定の共通ユーザーに表示されるデータを1つ以上のコンテナに制限するメカニズムとともに格納する表またはビューです。コンテナ・データ・オブジェクトの例は、Oracle提供のビューで、その名前はV$およびCDB_で始まります。

親トピック: 用語集

資格証明

データベースにアクセスするために使用するユーザー名、パスワードまたは証明書。

親トピック: 用語集

CRL

「証明書失効リスト(CRL)」を参照。

親トピック: 用語集

CRL配布ポイント

(CRL DP) X.509バージョン3証明書標準で指定されるオプションの拡張子であり、証明書の失効情報が格納される区分CRLの位置を示します。通常、この拡張子の値はURLの形式です。CRL DPによって、1つの認証局ドメイン内の失効情報を複数のCRLにポストできます。CRL DPによって、失効情報はより管理しやすい部分に細分化され、CRLが膨大に増加するのが回避されるため、パフォーマンスが向上します。たとえば、CRL DPを証明書に指定し、その証明書の失効情報をダウンロードできる、Webサーバー上のファイルを指すようにできます。

親トピック: 用語集

CRL DP

「CRL配布ポイント」を参照。

親トピック: 用語集

暗号化

データのエンコーディングおよびデコーディング処理であり、メッセージを保全する。

親トピック: 用語集

データ・ディクショナリ

データベースに関する情報を提供する読取り専用の表のセット。

親トピック: 用語集

Data Encryption Standard (DES)

米国連邦情報処理標準の古い暗号化アルゴリズムであり、Advanced Encryption Standard (AES)に置き換えられました。このリリースでは、DES、DES40、3DES112および3DES168アルゴリズムは非推奨です。より強力なアルゴリズムを使用するようにOracle Database環境を移行するには、My Oracle Supportノート2118136.2で説明されているパッチをダウンロードしてインストールします。

親トピック: 用語集

データベース管理者

(1)Oracleサーバーまたはデータベース・アプリケーションを操作および管理する個人。(2)DBA権限を付与され、データベース管理機能を実行できるOracleユーザー名。通常、これら2つを同時に意味します。多くのサイトでは複数のDBAが配置されます。

親トピック: 用語集

データベース別名

「ネット・サービス名」を参照。

親トピック: 用語集

データベース・インストール管理者

データベース作成者とも呼ばれる。この管理者は、新規データベースの作成を担当する。この作業には、Database Configuration Assistantを使用したディレクトリへの各データベースの登録が含まれる。この管理者は、データベース・サービス・オブジェクトおよび属性に対する作成および変更のアクセス権を持つ。この管理者は、デフォルトのドメインも変更できる。

親トピック: 用語集

データベース・リンク

ローカル・データベースまたはネットワーク定義に格納されるネットワーク・オブジェクトであり、リモート・データベース、そのデータベースへの通信パス、およびオプションでユーザー名とパスワードを識別します。定義された後、データベース・リンクはリモート・データベースへのアクセスに使用されます。

あるデータベースから別のデータベースへのパブリックまたはプライベート・データベース・リンクは、DBAまたはユーザーによってローカル・データベースに作成される。

グローバル・データベース・リンクは、Oracle Namesで各データベースからネットワーク内の他のすべてのデータベースに自動的に作成される。グローバル・データベース・リンクはネットワーク定義に格納される。

親トピック: 用語集

データベース・パスワード・バージョン

ユーザーのデータベース・パスワードから導出された取消しできない値。パスワード・ベリファイアとも呼ばれます。この値は、データベースに対するパスワード認証時に、接続ユーザーの識別情報が正しいことを確認するために使用されます。

親トピック: 用語集

データベース・セキュリティ管理者

データベース・エンタープライズ・ユーザー・セキュリティの最上位レベルの管理者。この管理者は、すべてのエンタープライズ・ドメインに対する権限を持ち、次のことに責任を担っている。

- Oracle `DBSecurityAdmins`および`OracleDBCreators`グループの管理

新規エンタープライズ・ドメインの作成。

- エンタープライズ内のあるドメインから別のドメインへのデータベースの移動

親トピック: 用語集

復号化

暗号化されたメッセージ(暗号文)の内容を元どおり読める形式(平文)に変換する処理です。

親トピック: 用語集

定義者権限プロシージャ

現行ユーザーではなく、所有者の権限で実行されるプロシージャ(プログラム・ユニット)。定義者権限サブプログラムは、これらのサブプログラムが格納されるスキーマにバインドされます。

たとえば、ユーザー`blake`とユーザー`scott`のそれぞれが`dept`という名前の表を、それぞれのユーザー・スキーマの中に持っています。ユーザー`blake`が`dept`表を更新するためにユーザー`scott`が所有している定義者権限プロシージャをコールすると、このプロシージャは`dept`表を`scott`スキーマで更新します。これは、プロシージャはプロシージャを所有している(定義した)ユーザー(つまり`scott`)の権限で実行するためです。

「実行者権限プロシージャ」も参照してください。

親トピック: 用語集

サービス拒否(DoS)攻撃

Webサイトをアクセス不能または使用不能にする攻撃。サービス拒否攻撃は様々な手法で行われますが、よく利用される攻撃手法としては、サイトをクラッシュさせるもの、サイトへの接続を拒否するもの、または低速化によりサイトを使用不能にするものがあります。DoS攻撃には、次の2つの形式があります。

- 基本サービス拒否攻撃(1台のみまたは数台のコンピュータが必要)
- 分散型DoS攻撃(多数のコンピュータの実行が必要)

親トピック: 用語集

DES

「Data Encryption Standard(DES)」を参照してください。

親トピック: 用語集

辞書攻撃

パスワードに対する一般的な攻撃。攻撃者は、多数の一般的なパスワードのリストを作成し、それらを暗号化します。次に、攻撃者は暗号化されたパスワードを含むファイルを盗み、暗号化した一般的なパスワードのリストと比較します。暗号化したパスワードの値(ベリファイアと呼ばれる)のいずれかが一致した場合、攻撃者は対応するパスワードを盗むことができます。辞書攻撃は、暗号化の前にパスワードにsaltを使用することで回避できます。「salt」を参照してください。

親トピック: 用語集

Diffie-Hellmanキー交換アルゴリズム

これは、安全でないチャネルを通じて通信する2つのパーティに、それらのパーティのみが知っているランダムな数字を合意させる方法です。Diffie-Hellmanキー交換アルゴリズムの実行中は、当事者は非保護チャネルで情報を交換しますが、攻撃者がネットワーク通信を分析し、当事者の間で取り決めた乱数を計算によって推定するのはほぼ不可能です。Oracle Databaseでは、セッション・キーの生成にDiffie-Hellmanキー交換アルゴリズムが使用されています。

親トピック: 用語集

デジタル署名

デジタル署名は、公開キー・アルゴリズムを使用して送信者の秘密キーで送信者のメッセージに署名するときに作成される。このデジタル署名によって、文書が信頼できるものであること、別のエンティティで偽造されていないこと、変更されていないこと、送信者によって拒否されないことが保証される。

親トピック: 用語集

ディレクトリ情報ツリー(DIT)

LDAPディレクトリ内のエントリのDNから構成される階層ツリー型構造。「識別名(DN)」を参照。

親トピック: 用語集

ディレクトリ・ネーミング

データベース・サービス、ネット・サービス名またはネット・サービス別名を中央ディレクトリ・サーバーに格納されている接続記述子に変換するネーミング・メソッド。A

親トピック: 用語集

ディレクトリ・ネーミング・コンテキスト

ディレクトリ・サーバー内で意味を持つサブツリー。通常は、組織サブツリーの最上位です。あるディレクトリでは、固定のこのようなコンテキストが1個のみ許可されますが、他のディレクトリでは、ディレクトリ管理者によって0個から多数までのコンテキストを構成できます。

親トピック: 用語集

識別名(DN)

ディレクトリ・エントリの一意の名前。親エントリからディレクトリ情報ツリーのルート・エントリまでのすべての個々の名前から構成されます。「ディレクトリ情報ツリー(DIT)」を参照してください。

親トピック: 用語集

ドメイン

ドメイン・ネーム・システム(DNS)ネームスペース内の任意のツリーまたはサブツリー。一般にドメインは、ホスト名が共通の接尾辞(ドメイン名)を共有するコンピュータのグループを表す。

親トピック: 用語集

ドメイン・ネーム・システム(DNS)

ドメインの階層に編成された、コンピュータおよびネットワーク・サービスのネーミングのためのシステム。DNSは、ユーザーにわかりやすい名前でコンピュータの位置を特定するためにTCP/IPネットワークで使用される。DNSは、このわかりやすい名前を、コンピュータが理解できるIPアドレスに変換する。

Oracle Net Servicesでは、DNSはTCP/IPアドレスのホスト名をIPアドレスに変換する。

親トピック: 用語集

直接付与されたロール

間接的に付与されたロールとは対照的に、ユーザーに直接付与されたロール。

親トピック: 用語集

暗号化テキスト

暗号化アルゴリズムを使用して暗号化されたテキスト。暗号化プロセスの出力ストリーム。最初に復号化の対象とならないかぎり、そのままでは読取りまたは解読できません。暗号文とも呼ばれます。暗号化テキストは、最終的には平文になります。

親トピック: 用語集

暗号化

メッセージを宛先の受信者以外の第三者が判読できない書式に変換すること。

親トピック: 用語集

エンタープライズ・ドメイン

データベースとエンタープライズ・ロールのグループで構成されたディレクトリ構造。1つのデータベースが同時に複数のエンタープライズ・ドメイン内に存在することはありません。エンタープライズ・ドメインは、共通ディレクトリ・データベースを共有するコンピュータの集合であるWindows 2000ドメインとは異なる。

親トピック: 用語集

エンタープライズ・ドメイン管理者

新規エンタープライズ・ドメイン管理者を追加する権限を含め、特定のエンタープライズ・ドメインの管理を認可されたユーザー。

親トピック: 用語集

エンタープライズ・ロール

エンタープライズ・ユーザーに割り当てられるアクセス権限。エンタープライズ・ドメイン内の1つ以上のデータベースに対するOracleロールベースの認可のセット。エンタープライズ・ロールは、ディレクトリに格納され、1つ以上のグローバル・ロールが含まれています。

親トピック: 用語集

エンタープライズ・ユーザー

ディレクトリで定義および管理されるユーザー。各エンタープライズ・ユーザーは、企業内で固有の識別情報を保持します。

親トピック: 用語集

エントリ

ディレクトリの基本要素であり、ディレクトリ・ユーザーに関係のあるオブジェクトに関する情報が含まれている。

親トピック: 用語集

外部認証

KerberosやRADIUSなどのサード・パーティ認証サービスによるユーザー識別情報の検証。

親トピック: 用語集

米国連邦情報処理標準(FIPS)

暗号化モジュールのセキュリティ要件を定義する米国連邦政府の標準であり、コンピュータおよびテレコミュニケーション・システム内の非機密情報を保護するセキュリティ・システムで使用されます。米国商務省国立標準技術研究所(NIST)によって発行されます。

親トピック: 用語集

FIPS

「米国連邦情報処理標準(FIPS)」を参照してください。

親トピック: 用語集

強制クリーン・アップ

すべての監査レコードをデータベースから強制的にクリーン・アップ(つまり、削除)するための機能。この処理を行うには、`DBMS_AUDIT_MGMT.CLEAN_AUDIT_TRAIL`プロシージャの`USE_LAST_ARCH_TIMESTAMP`引数を`FALSE`に設定します。

「削除ジョブ」も参照してください。

親トピック: 用語集

フォレスト

相互に信頼する1つ以上のActive Directoryツリーのグループ。フォレスト内のすべてのツリーは、共通のスキーマ、構成およびグローバル・カタログを共有する。フォレストに複数のツリーが含まれる場合、ツリーは連続したネームスペースを形成しない。特定フォレスト内のすべてのツリーは、推移的な双方向の信頼関係を介して相互に信頼する。

親トピック: 用語集

転送可能なチケット認可チケット

プロキシに転送できる特殊なKerberosチケット。プロキシ認証用に、プロキシはクライアントにかわって追加Kerberosチケットを取得することが許可されます。

「Kerberosチケット」も参照してください。

親トピック: 用語集

グローバル・ロール

ディレクトリで管理されるロールだが、その権限は単一のデータベースに格納されている。グローバル・ロールは、次の構文を使用してデータベースに作成される。

```
CREATE ROLE role_name IDENTIFIED GLOBALLY;
```

親トピック: 用語集

グローバル・アプリケーション・コンテキスト

アプリケーション・コンテキスト値を複数のデータベース・セッションにわたってアクセス可能にする名前と値のペア。

「アプリケーション・コンテキスト」も参照してください。

親トピック: 用語集

グリッド・コンピューティング

多くのサーバーと記憶域を調整して単一の大容量コンピュータとして動作させるコンピューティング・アーキテクチャ。Oracle Grid Computingは、柔軟性のあるオンデマンド・コンピューティング・リソースをすべてのエンタープライズ・コンピューティング・ニーズに対して作成します。Oracle Databaseグリッド・コンピューティング・インフラストラクチャで稼働しているアプリケーションは、フェイルオーバー、ソフトウェア・プロビジョニングおよび管理のための共通インフラストラクチャ・サービスを利用できます。Oracle Grid Computingは、リソースの需要を分析し、それに応じて供給を調整します。

親トピック: 用語集

HTTP

Hypertext Transfer Protocol: World Wide Web上でのファイル(テキスト、グラフィック・イメージ、サウンド、ビデオおよびその他のマルチメディア・ファイル)の交換に関するルール・セット。TCP/IPプロトコル・スイート(インターネット上での情報交換の基礎)に対して、HTTPはアプリケーション・プロトコルである。

親トピック: 用語集

HTTPS

標準のHTTPアプリケーション・レイヤーのサブレイヤーとしてTransport Layer Security (TLS)を使用したプロトコル。

親トピック: 用語集

間接的に付与されたロール

ユーザーにすでに付与されている別のロールを通じてこのユーザーに付与されるロールのことです。その後、role2ロールとrole3ロールをrole1ロールに付与します。これで、role2とrole3の2つのロールは、role1に含まれることになります。つまり、psmithには、直接付与されたrole1に加え、role2ロールとrole3ロールが間接的に付与されたことになります。psmithに対して、直接付与されたロールrole1を使用可能にすると、間接的に付与されたロールrole2およびrole3も同様に使用可能になります。

親トピック: 用語集

識別情報

エンティティの公開キーとその他の公開情報の組合せ。公開情報には、電子メール・アドレスなどのユーザー識別データを含めることができる。宣言どおりのエンティティとして証明されているユーザー。

親トピック: 用語集

アイデンティティ管理

オンライン、すなわちデジタルなエンティティの作成、管理および使用。アイデンティティ管理には、デジタル識別情報の作成(デジタル識別情報のプロビジョニング)から、メンテナンス(電子リソースへのアクセスに関する組織ポリシーの施行)、さらに最終的な終了までのライフ・サイクル全体の安全な管理が含まれる。

親トピック: 用語集

アイデンティティ管理レルム

Oracle Internet Directoryのサブツリーであり、Oracleコンテキストだけでなく、それぞれアクセス制御リストで保護されているユーザーおよびグループの追加サブツリーも含む。

親トピック: 用語集

初期チケット

Kerberos認証では、初期チケットまたはチケット認可チケット(TGT)によって、ユーザーが追加のサービス・チケットを要求する権利を持つものとして識別される。初期チケットがないと、他のチケットは取得できない。初期チケットは、`okinit`プログラムを実行し、パスワードを指定することで取得される。

親トピック: 用語集

インスタンス

稼働中のすべてのOracleデータベースは、Oracleインスタンスに関連付けられている。データベースが(コンピュータのタイプに関係なく)データベース・サーバー上で起動すると、Oracleによってシステム・グローバル領域(SGA)というメモリー領域が割り当てられ、Oracleプロセスが起動する。このSGAとOracleプロセスの組合せをインスタンスと呼ぶ。インスタンスのメモリーおよびプロセスは、関連付けられているデータベースのデータを効率的に管理し、1つ以上のデータベース・ユーザーを処理する。

親トピック: 用語集

整合性

受信したメッセージの内容が、送信された元のメッセージの内容から変更されていないという保証。

親トピック: 用語集

実行者権限プロシージャ

現行ユーザー、つまりプロシージャを起動するユーザーの権限で実行されるプロシージャ(プログラム・ユニット)。これらのプロシー

ジャは、特定のスキーマにバインドされません。このプロシージャは様々なユーザーが実行でき、これによって、複数のユーザーが、集中化したアプリケーション・ロジックを使用してそれぞれのデータを管理できます。実行者権限プロシージャは、プロシージャ・コードの宣言セクションにあるAUTHID句を使用して作成されます。

たとえば、ユーザーblakeとユーザーscottのそれぞれがdeptという名前の表を、それぞれのユーザー・スキーマの中に持っています。ユーザーblakeがdept表を更新するためにユーザーscottが所有している実行者権限プロシージャをコールすると、このプロシージャはdept表をblakeスキーマで更新します。これは、プロシージャはプロシージャを起動したユーザー(つまりblake)の権限で実行するためです。

「定義者権限プロシージャ」も参照してください。

親トピック: 用語集

Javaコード不明瞭化

Javaコード不明瞭化は、Javaプログラムをリバース・エンジニアリングから保護するために使用される。特別なプログラム(obfuscator)を使用して、コードに見つかったJavaシンボルをスクランブルする。プロセスは、元のプログラム構造をそのまま保持し、意図した動作を隠すためにクラス、メソッドおよび変数の名前を変更する一方でプログラムが正常に稼働するようにする。不明瞭化されていないJavaコードをデコンパイルして読み取ることは可能だが、不明瞭化されたJavaコードのデコンパイルは、米国政府の輸出規制を満たすのに十分なほど困難になっている。

親トピック: 用語集

Java Database Connectivity (JDBC)

Javaプログラムからリレーショナル・データベースに接続するための業界標準のJavaインタフェース。Sun Microsystemsが定義した。

親トピック: 用語集

JDBC

「Java Database Connectivity (JDBC)」を参照。

親トピック: 用語集

KDC

Key Distribution Center (KDC)を参照。

親トピック: 用語集

Kerberos

Massachusetts Institute of TechnologyのAthenaプロジェクトで開発されたネットワーク認証サービスであり、分散環境でのセキュリティを強化します。Kerberosは信頼できるサード・パーティ認証システムであり、共有秘密鍵に基づき、サード・パーティがセキュアであることを前提とします。シングル・サインオン機能とデータベース・リンク認証(MIT Kerberosのみ)があり、パスワードを一元的に保管してPCのセキュリティを強化します。

親トピック: 用語集

Kerberosチケット

特定サービスに関するクライアントの識別情報を検証する一時的な電子資格証明。サービス・チケットとも呼ばれます。

親トピック: 用語集

Key Distribution Center (KDC)

Kerberos認証では、KDCはユーザー・プリンシパルのリストを管理し、ユーザーの初期チケットに関して`kinit`(`okinit`はOracleバージョン)プログラムを通じてアクセスされます。KDCおよびチケット認可サービスが同じエンティティに結合されることが多いですが、その場合もKDCと呼ばれます。チケット認可サービスは、サービス・プリンシパルのリストを管理し、このようなサービスを提供するサーバーに対してユーザーの認証が必要な場合にアクセスされます。KDCは、セキュア・ホストで実行する必要のある信頼できるサード・パーティです。チケット認可チケットおよびサービス・チケットを作成します。

「Kerberosチケット」も参照してください。

親トピック: 用語集

キーのペア

公開キーおよびそれに関連付けられている秘密キー。「公開キーと秘密キーのペア」を参照。

親トピック: 用語集

keytabファイル

1つ以上のサービス・キーを含むKerberosキー表ファイル。キー表ファイルは、ユーザーが各自のパスワードを使用する場合と同様に、ホストまたはサービスで使用されます。

親トピック: 用語集

kinstance

Kerberos認証されたサービスのインスタンス化または場所。これは、任意の文字列ですが、通常はサービスのホスト・コンピュータ名を指定します。

親トピック: 用語集

kservice

Kerberosサービス・オブジェクトの任意の名前。

親トピック: 用語集

最終アーカイブ・タイムスタンプ

監査レコードが最後にアーカイブされた時間を示すタイムスタンプ。データベース監査証跡の場合、このタイムスタンプは、最後にアーカイブされた監査レコードを示します。オペレーティング・システム監査ファイルの場合、このタイムスタンプは、アーカイブされた監査ファイルの最終変更タイムスタンプ・プロパティを示します。このタイムスタンプを設定するには、`DBMS_AUDIT_MGMT.SET_LAST_ARCHIVE_TIMESTAMP` PL/SQLプロシージャを使用します。

「削除ジョブ」も参照してください。

親トピック: 用語集

LDAP

「Lightweight Directory Access Protocol (LDAP)」を参照。

親トピック: 用語集

ldap.oraファイル

次のディレクトリ・サーバー・アクセス情報を含むファイル。Oracle Net Configuration Assistantによって作成される。

- ディレクトリ・サーバーのタイプ
- ディレクトリ・サーバーの場所
- クライアントまたはサーバーで使用するデフォルトのアイデンティティ管理レルムまたはOracleコンテキスト(ポートを含む)

親トピック: 用語集

Lightweight Directory Access Protocol (LDAP)

標準の拡張可能ディレクトリ・アクセス・プロトコル。LDAPクライアントおよびサーバーが通信に使用する共通言語です。Oracle Internet Directoryなど、業界標準のディレクトリ製品をサポートする設計規則のフレームワーク。

親トピック: 用語集

リスナー

サーバー上のプロセスであり、着信クライアント接続リクエストをリスニングし、サーバーへのトラフィックを管理する。

クライアントがサーバーとのネットワーク・セッションをリクエストするたびに、リスナーが実際のリクエストを受信する。クライアント情報がリスナー情報と一致する場合、リスナーはサーバーへの接続を付与する。

親トピック: 用語集

listener.oraファイル

次の内容を識別するリスナーの構成ファイル。

- リスナー名
- 接続リクエストを受け入れるプロトコル・アドレス
- リスニングするサービス

通常、`listener.ora`ファイルは、UNIXプラットフォームでは`$ORACLE_HOME/network/admin`に、Windowsでは`ORACLE_BASE¥ORACLE_HOME¥network¥admin`にあります。

親トピック: 用語集

軽量ユーザー・セッション

ユーザーがログインしているアプリケーションに関連した情報のみが含まれるユーザー・セッションです。軽量ユーザー・セッションには、このセッションのデータベース・リソース(トランザクションやカーソルなど)は入っていません。そのため「軽量」とみなされます。軽量ユーザー・セッションが消費するシステム・リソースは、従来のデータベース・セッションよりはるかに少ない量です。軽量ユーザー・セッションが消費するサーバー・リソースは非常に少ないものになるため、軽量ユーザー・セッションを各エンド・ユーザー専用にして、アプリケーションにより必要とみなされるかぎり持続させることができます。

親トピック: 用語集

ローカル権限付与

権限が付与されたPDBにのみ適用できる権限。

「共通権限付与」も参照。

親トピック: 用語集

ローカル・ロール

CDBで、非CDBのみに存在する非CDBのロールと同様に、単一のPDBのみに存在するロール。共通ロールと異なり、ローカル・ロールには、そのロールが存在するコンテナで適用されるロールおよび権限のみを含めることができます。

親トピック: 用語集

ローカル・ユーザー

CDBで、共通ユーザーではないユーザー。

親トピック: 用語集

必須監査

デフォルトで監査されるアクティビティ。管理ユーザーSYS、SYSDBA、SYSOPER、SYSASM、SYSBACKUP、SYSDGおよびSYSKMによる、統合監査証跡ポリシー(`ALTER AUDIT POLICY`など)および最上位レベルの文への変更などがあります。詳細は、「強制的に監査されるアクティビティ」を参照してください。

親トピック: 用語集

MD5

メッセージ・ダイジェスト5。与えられたデータから128ビットの暗号メッセージ・ダイジェスト値を生成することにより、データの整合性を保証するアルゴリズム。データの単一ビット値のみ変更された場合、MD5はデータ変更のチェックサムを計算します。MD5で元のデータと同じ結果が生成されるようにデータを偽造することは、計算上不可能と考えられます。

MD5は、このリリースでは非推奨です。より強力なアルゴリズムを使用するようにOracle Database環境を移行するには、My Oracle Supportノート2118136.2で説明されているパッチをダウンロードしてインストールします。

親トピック: 用語集

メッセージ認証コード

データ認証コード(DAC)とも呼ばれる。秘密キーを追加したチェックサム。キーを持つ人のみが暗号チェックサムを検証できる。

親トピック: 用語集

メッセージ・ダイジェスト

「チェックサム」を参照してください。

親トピック: 用語集

CDB

「CDB」を参照してください。

親トピック: 用語集

ネームスペース

Oracle Databaseのセキュリティにおけるアプリケーション・コンテキストの名前。CREATE CONTEXT文でこの名前を作成します。

親トピック: 用語集

ネーミング・メソッド

クライアント・アプリケーションがデータベース・サービスへの接続を試みるときに、接続識別子を接続記述子に変換するために使用する解決メソッド。

親トピック: 用語集

米国商務省国立標準技術研究所(NIST)

米国商務省の機関であり、コンピュータおよびテレコミュニケーション・システム内の暗号ベース・セキュリティ・システムの設計、取得および実装に関連するセキュリティ標準の開発を担う。連邦機関、または連邦の機能を遂行するために連邦政府にかわって情報を処理する連邦機関の請負業者やその他の組織によって運営されている。

親トピック: 用語集

ネット・サービス別名

ディレクトリ・サーバーのディレクトリ・ネーミング・オブジェクトの代替名。ディレクトリ・サーバーには、定義されたネット・サービス名またはデータベース・サービスのネット・サービス別名が格納される。ネット・サービス別名のエントリには、接続記述子情報はない。かわりに、別名の対象のオブジェクトの場所のみ参照する。クライアントがネット・サービス別名のディレクトリ検索をリクエストすると、ディレクトリは、エントリがネット・サービス別名であることを判断し、そのネット・サービス別名が実際に参照しているエントリであるかのように検索を実行する。

親トピック: 用語集

ネット・サービス名

接続記述子に解決されるサービスの単純名。ユーザーは、接続するサービスに対する接続文字列内に、ネット・サービス名とともにユーザー名およびパスワードを渡すことで接続リクエストを開始します。

```
CONNECT username@net_service_name
Enter password: password
```

必要に応じて、ネット・サービス名は次のような様々な場所に格納できます。

- 各クライアントのローカル構成ファイル、`tnsnames.ora`
- ディレクトリ・サーバー
- NISなどの外部ネーミング・サービス

親トピック: 用語集

ネットワーク認証サービス

分散環境で、クライアントからサーバー、サーバー間およびユーザーからクライアントとサーバーの両方を認証する手段。ネットワーク認証サービスは、ユーザーに関する情報、ユーザーがアクセスする様々なサーバー上のサービスに関する情報、およびネットワーク上のクライアントとサーバーに関する情報を格納するためのリポジトリである。認証サーバーは、物理的に異なるコンピュータにすることも、システム内の別のサーバー上の共同の場所にある設備にすることもできる。可用性を保証するために、一部の認証サービスを複製してシングル・ポイント障害を回避できる。

親トピック: 用語集

ネットワーク・リスナー

1つ以上のプロトコルで1つ以上のデータベースの接続リクエストをリスニングするサーバー上のリスナー。「リスナー」を参照してください。

親トピック: 用語集

NIST

「米国商務省国立標準技術研究所(NIST)」を参照。

親トピック: 用語集

非CDB

CDBでないOracle Database。

親トピック: 用語集

否認防止

メッセージの発信元、配信、送信または伝送の明白な証明。

親トピック: 用語集

不明瞭化

情報を判読不能な形式にスクランブルするプロセス。スクランブルに使用されているアルゴリズムが不明な場合、スクランブル解除が非常に困難になる。

親トピック: 用語集

不明瞭化プログラム

Javaソース・コードの不明瞭化に使用される特殊なプログラム。「不明瞭化」を参照してください。

親トピック: 用語集

オブジェクト・クラス

名前を持った属性のグループ。エントリに属性を割り当てるには、これらの属性を保持するオブジェクト・クラスをそのエントリに割り当てる。同じオブジェクト・クラスに関連付けられているすべてのオブジェクトは、同じ属性を共有する。

親トピック: 用語集

Oracleコンテキスト

1. LDAP準拠のインターネット・ディレクトリ内にあるcn=OracleContextというエントリ。このディレクトリには、Oracle Net Servicesディレクトリ・ネーミングおよびチェックサムのセキュリティのエントリなど、Oracleソフトウェア関連のすべての情報が格納されている。

1つのディレクトリに1つ以上のOracleコンテキストを設定できる。Oracleコンテキストは、通常はアイデンティティ管理レルムにある。

親トピック: 用語集

Oracle Virtual Private Database

行および列レベルでデータベース・アクセスを制御するセキュリティ・ポリシーを作成できる一連の機能。基本的には、Oracle Virtual Private Databaseのセキュリティ・ポリシーが適用された表、ビューまたはシノニムに対して発行されるSQL文に、動的なWHERE句が追加されます。

親トピック: 用語集

Oracle Net Services

OracleサーバーまたはDesigner/2000などのOracleのツール製品を実行する2台以上のコンピュータがサード・パーティ・ネットワークを通じてデータを交換できるようにするOracle製品。Oracle Net Servicesは、分散処理および分散データベース機能をサポートする。Oracle Net Servicesは、通信プロトコルに依存しないためオープン・システムであり、ユーザーは多くのネットワーク環境へのインタフェースとしてOracle Netを使用できる。

親トピック: 用語集

Oracle PKI証明書使用

証明書 でサポートされるOracleアプリケーションのタイプを定義します。

親トピック: 用語集

パスワードでアクセス可能なドメインのリスト

パスワード認証ユーザーからの接続を受け入れるために構成されたエンタープライズ・ドメインのグループ。

親トピック: 用語集

PCMCIAカード

Personal Computer Memory Card International Association (PCMCIA)標準に準拠する小さなクレジット・カード・サイズのコンピューティング・デバイス。これらのデバイスはPCカードとも呼ばれ、メモリー、モデムまたはハードウェア・セキュリティ・モジュールの追加に使用される。PCMCIAカードは、ハードウェア・セキュリティ・モジュールが公開キーと秘密キーのペアの秘密キー・コンポーネントを安全に格納する際に使用され、暗号操作も実行するものもあります。

親トピック: 用語集

PDB

CDBの一部である、個別のデータベース。

「ルート」も参照してください。

親トピック: 用語集

ピア識別情報

SSL接続セッションは、特定のクライアントと特定のサーバー間のセッションである。ピアの識別情報は、セッションのセットアップの一部として設定される場合がある。接続先は、X.509証明連鎖によって識別されます。

親トピック: 用語集

PEM

インターネット上で安全な電子メールを提供するためにInternet Architecture Boardによって採用されたInternet Privacy-Enhanced Mailプロトコル標準。PEMプロトコルは、暗号化、認証、メッセージ整合性およびキー管理を提供する。PEMは、データ暗号化キーを暗号化するための対称型スキームと公開キー・スキームの両方を含む様々なキー管理アプローチと互換性を持つよう意図された包括的な標準である。PEMの仕様は、4つのInternet Engineering Task Force (IETF)ドキュメント、RFC 1421、1422、1423および1424に記載されている。

親トピック: 用語集

PKCS #10

認証リクエストの構文を記述するRSA Security社のPublic-Key Cryptography Standards (PKCS)仕様。認証リクエストは、識別名、公開キーおよびオプションの属性セットから構成され、証明書をリクエストするエンティティによって一括して署名

される。このマニュアルでは、認証リクエストを証明書リクエストと呼ぶ。「証明書リクエスト」を参照してください。

親トピック: 用語集

PKCS #11

暗号情報を保持し、暗号操作を実行するデバイスに対する、Cryptokiと呼ばれるアプリケーション・プログラミング・インタフェース(API)を定義するRSA Security社のPublic-Key Cryptography Standards (PKCS)仕様。「PCMCIAカード」を参照してください。

親トピック: 用語集

PKCS #12

通常はウォレットと呼ばれる形式で個人認証資格証明を格納および転送するための転送構文を記述するRSA Security社のPublic-Key Cryptography Standards (PKCS)仕様。

親トピック: 用語集

PKI

「公開キー・インフラストラクチャ(PKI)」を参照。

親トピック: 用語集

平文

暗号化されていないメッセージ・テキスト。

親トピック: 用語集

プラガブル・データベース

「PDB」を参照してください。

親トピック: 用語集

プリンシパル

Kerberos資格証明のセットが割り当てられているクライアントまたはサーバーを一意に識別する文字列。通常、これには`kservice/kinstance@REALM`という3つの部分が含まれます。ユーザーの場合、`kservice`はユーザー名です。「kservice」、「kinstance」および「レルム」も参照してください。

親トピック: 用語集

秘密キー

公開キー暗号化では、このキーが秘密キーです。主に復号化に使用されますが、デジタル署名とともに暗号化にも使用されます。「公開キーと秘密キーのペア」を参照。

親トピック: 用語集

プロキシ認証

ファイアウォールなどの中間層を伴う環境で一般に使用されるプロセス。エンド・ユーザーは中間層に対して認証を行い、中間層はエンド・ユーザーのプロキシとして、ユーザーのかわりにディレクトリに対して認証を実施します。中間層は、プロキシ・ユーザーとしてディレクトリにログインします。プロキシ・ユーザーは識別情報を切り替えることができ、ディレクトリにログインするとエンド・ユーザーの識別情報に切り替わります。プロキシ・ユーザーは、特定のエンド・ユーザーに適した認可を使用して、そのエンド・ユーザーにかわって操作を実行できます。

親トピック: 用語集

公開キー

公開キー暗号化では、このキーがすべてに対して公開されます。主に暗号化に使用されますが、署名の検証にも使用できます。「公開キーと秘密キーのペア」を参照。

親トピック: 用語集

公開キーと秘密キーのペア

暗号化および復号化に使用される2つの数値のセットであり、一方を秘密キーと呼び、もう一方を公開キーと呼ぶ。通常、公開キーは広範に使用可能であるが、秘密キーはそれぞれの所有者が保持する。数学的な関連性はあるが、一般には公開キーから秘密キーを導出するのは計算上不可能とみなされている。公開キーと秘密キーは、公開キー暗号化アルゴリズムまたは公開キー暗号方式とも呼ばれる非対称型暗号化アルゴリズムでのみ使用される。キーのペアの公開キーまたは秘密キーのいずれかで暗号化されたデータは、キーのペアのうち関連付けられているキーで復号化できる。ただし、公開キーで暗号化されたデータを同じ公開キーで復号化することはできず、秘密キーで暗号化されたデータを同じ秘密キーで復号化することはできない。

親トピック: 用語集

公開キー・インフラストラクチャ(PKI)

公開キー暗号化の原理を利用した情報セキュリティ・テクノロジ。公開キー暗号化には、共有の公開キーと秘密キーのペアを使用した情報の暗号化および復号化が含まれる。パブリック・ネットワーク内にセキュアでプライベートな通信を提供する。

親トピック: 用語集

PUBLICロール

すべてのデータベース・アカウントが自動的に保有する特殊なロール。デフォルトでは割り当てられている権限がありませんが、多くのJavaオブジェクトに対する付与があります。PUBLICロールは削除できません。また、ユーザー・アカウントは常にこのロールを前提とするため、このロールの手動の付与や取消しは意味がありません。PUBLICロールはすべてのデータベース・ユーザー・アカウントが前提とするため、DBA_ROLESおよびSESSION_ROLESデータ・ディクショナリ・ビューには表示されません。

親トピック: 用語集

削除ジョブ

DBMS_AUDIT_MGMT.CREATE_PURGE_JOBプロシージャにより作成されたデータベース・ジョブで、監査証跡の削除を管理します。データベース管理者が削除ジョブをスケジューリングおよび使用可能/使用禁止にします。削除ジョブはアクティブにな

ると、監査レコードをデータベース監査表から削除したり、Oracle Databaseオペレーティング・システム監査ファイルを削除します。

「強制クリーン・アップ」、「最終アーカイブ・タイムスタンプ」も参照してください。

親トピック: 用語集

RADIUS

Remote Authentication Dial-In User Service(RADIUS)は、リモート・アクセス・サーバーが中央サーバーと通信してダイアルイン・ユーザーを認証し、リクエストされたシステムまたはサービスへのアクセスを認可できるようにするクライアント/サーバー・プロトコルおよびソフトウェアです。

親トピック: 用語集

レルム

1. アイデンティティ管理レルムの省略形。 2.Kerberosオブジェクト。1つのKey Distribution Center/Ticket Granting Service (KDC/TGS)の下で動作するクライアントとサーバーのセット。同じ名前を共有する異なるレルム内のサービス(kserviceを参照)は一意である。

親トピック: 用語集

レルムOracleコンテキスト

Oracle Internet Directoryのアイデンティティ管理レルムの一部であるOracleコンテキスト。

親トピック: 用語集

レジストリ

コンピュータの構成情報を格納するWindowsリポジトリ。

親トピック: 用語集

リモート・コンピュータ

ローカル・コンピュータ以外のネットワーク上にあるコンピュータ。

親トピック: 用語集

ロール

関連する権限のグループに名前を付けたもの。ユーザーや他のロールに付与します。

「間接的に付与されたロール」も参照してください。

親トピック: 用語集

root

マルチテナント環境で、すべてのPDBが属している、オラクル社が提供およびユーザーが作成したスキーマのコレクション。コンテ

ナ・データベースでは、ルートは1つのみです。各PDBは、このルートの子であるとみなされます。ルートは、データ・ディクショナリに各PDBの存在を示すエントリを持っています。

「コンテナ」、「CDB」、「PDB」も参照。

親トピック: 用語集

ルート・キー証明書

「信頼できる証明書」を参照してください。

親トピック: 用語集

salt

暗号化技術において、暗号化されたデータのセキュリティを強化する方法。データが暗号化される前に追加されるランダムな文字列で、攻撃者が暗号文のパターンを既知の暗号文サンプルに一致させてデータを盗むことを困難にします。saltは通常、辞書攻撃(悪意のあるハッカー(攻撃者)がパスワードを盗むために使用する方法)を防ぐために、暗号化される前のパスワードにも追加されます。暗号化されたsalt処理済の値により、暗号化されたパスワードのハッシュ値(ベリファイアとも呼ばれます)と、一般のパスワード・ハッシュ値の辞書リストとの照合が困難になります。

親トピック: 用語集

スキーマ

1. データベース・スキーマ: 表、ビュー、クラスタ、プロシージャ、パッケージ、属性、オブジェクト・クラスなどのオブジェクトと、それらに対応する一致ルールの名前付きコレクションであり、特定のユーザーに関連付けられています。 2.LDAPディレクトリ・スキーマ: 属性、オブジェクト・クラス、およびそれらに対応する一致ルールのコレクション。

親トピック: 用語集

スキーマ・マッピング

「ユーザー・スキーマ・マッピング」を参照してください。

親トピック: 用語集

セキュア・アプリケーション・ロール

アプリケーション・ユーザーに付与されるデータベース・ロール。ただし、実行者権限ストアド・プロシージャを使用して保護され、ロールのパスワードをデータベース表から取得します。セキュア・アプリケーション・ロールのパスワードは、アプリケーション内に埋め込まれていません。

「アプリケーション・ロール」も参照してください。

親トピック: 用語集

Secure Hash Algorithm (SHA)

指定されたデータから160ビットの暗号メッセージ・ダイジェスト値を生成することにより、データの整合性を保証するアルゴリズム。データのわずか1ビットが変更された場合でも、データのSecure Hash Algorithmチェックサムが変更される。Secure Hash

Algorithmで元のデータと同じ結果が生成されるように指定されたデータ・セットを偽造することは、計算上不可能と考えられる。

264ビット未満の長さのメッセージを受け取り、160ビットのメッセージ・ダイジェストを生成するアルゴリズムである。このアルゴリズムはMD5に比べて少し低速だが、メッセージ・ダイジェストが長いほど、総当たり攻撃と侵入攻撃に対する安全度が増す。

親トピック: 用語集

Secure Sockets Layer (SSL)

ネットワーク接続を保護するためにNetscape社が開発した業界標準プロトコル。SSLは、公開キー・インフラストラクチャ(PKI)を使用した認証、暗号化、およびデータの整合性を提供する。

Transport Layer Security (TLS)プロトコルは、SSLプロトコルの後継です。

親トピック: 用語集

業務分離

アクティビティを、それを実行する必要があるユーザーのみに制限すること。たとえば、SYSDBA管理権限は一般ユーザーには付与しないようにします。この権限は管理ユーザーにのみ付与します。業務分離は、多くのコンプライアンス・ポリシーで必要です。適切なユーザーへの権限付与に関するガイドラインは、「ユーザー・アカウントと権限の保護に関するガイドライン」を参照してください。

親トピック: 用語集

サーバー

サービスのプロバイダ。

親トピック: 用語集

サービス

1. Oracleデータベース・サーバーなど、クライアントによって使用されるネットワーク・リソース。

2. Windowsレジストリにインストールされ、Windowsによって管理される実行可能プロセス。サービスが作成され、開始された後は、コンピュータにログオンしているユーザーがいない場合でも実行できる。

親トピック: 用語集

サービス名

Kerberosベースの認証では、サービス・プリンシパルのkservice部分がサービス名です。

親トピック: 用語集

サービス・プリンシパル

「プリンシパル」を参照してください。

親トピック: 用語集

サービス・キー表

Kerberos認証では、サービス・キー表はkinstanceに存在するサービス・プリンシパルのリストです。KerberosをOracleで使用するには、その前にこの情報をKerberosから抽出し、Oracleサーバー・コンピュータにコピーする必要がある。

親トピック: 用語集

サービス・チケット

サービス・チケットは、事前定義された期間、特定のサービスまたはサーバーに対してクライアントを認証するために使用する信頼できる情報です。初期チケットを使用してKDCから取得されます。「Kerberosチケット」も参照してください。

親トピック: 用語集

セッション・キー

少なくとも二者(通常はクライアントとサーバー)によって共有されるキーであり、単一の通信セッション中のデータ暗号化に使用されます。セッション・キーは通常、ネットワーク・トラフィックを暗号化するために使用されます。クライアントとサーバーはセッションの開始時にセッション・キーをネゴシエーションすることができ、そのキーはそのセッションの関係者間のすべてのネットワーク・トラフィックを暗号化するために使用されます。クライアントとサーバーが新しいセッションで再び通信する場合は、新しいセッション・キーをネゴシエーションします。

親トピック: 用語集

セッション・レイヤー

プレゼンテーション・レイヤーのエンティティが必要とするサービスを提供するネットワーク・レイヤーであり、エンティティで対話の編成と同期およびデータ交換の管理を行えるようにする。このレイヤーは、クライアントとサーバー間でネットワーク・セッションを確立、管理および終了する。セッション・レイヤーの例には、ネットワーク・セッションがある。

親トピック: 用語集

SHA

「Secure Hash Algorithm (SHA)」を参照してください。

親トピック: 用語集

共有スキーマ

複数のエンタープライズ・ユーザーが使用できるデータベースまたはアプリケーション・スキーマ。Oracle Databaseでは、データベース上の同じ共有スキーマへの複数のエンタープライズ・ユーザーのマッピングがサポートされます。これにより、管理者はそれぞれのデータベースでユーザーごとにアカウントを作成する必要がなくなります。管理者は、ユーザーを1つの場所、つまり、エンタープライズ・ディレクトリに作成して、そのユーザーを共有スキーマにマップできます。この共有スキーマには他のエンタープライズ・ユーザーもマップできます。ユーザー/スキーマの分割とも呼ばれます。

親トピック: 用語集

単一キー・ペア・ウォレット

単一のユーザー証明書とその関連する秘密キーが含まれるPKCS #12 形式のウォレット。公開キーは証明書に埋め込まれている。

親トピック: 用語集

単一パスワード認証

単一パスワードを使用して複数のデータベースでユーザーを認証する機能。Oracle Databaseの実装では、パスワードはLDAP準拠ディレクトリに格納され、暗号化やアクセス制御リストで保護されます。

親トピック: 用語集

シングル・サインオン(SSO)

ユーザーが1度認証を受けると、その後の他のデータベースまたはアプリケーションへの接続に、厳密な認証が透過的に実施される機能。シングル・サインオンでは、ユーザーは1回の接続中に入力した単一のパスワードで複数のアカウントおよびアプリケーションにアクセスできます。単一のパスワードによる単一の認証です。Oracle Databaseは、KerberosおよびSSLベースのシングル・サインオンをサポートしています。

親トピック: 用語集

スマートカード

ユーザー名やパスワードなどの情報を格納するため、また認証交換に関連する計算を実行するための集積回路が埋め込まれた(クレジット・カードに似た)プラスチックのカード。スマートカードは、クライアントまたはサーバーでハードウェア・デバイスによって読み取られる。

スマートカードは、1回かぎりのパスワードとして使用できる乱数を生成できる。この場合、スマートカードは、サーバー上のサービスと同期するため、サーバーはスマートカードによって同じパスワードが生成されると想定する。

親トピック: 用語集

Sniffer

ネットワークからのプライベート・データ・トラフィックを不正にリスニングまたは取得するために使用されるデバイス。

親トピック: 用語集

SSO

「シングル・サインオン(SSO)」を参照。

親トピック: 用語集

システム・グローバル領域(SGA)

Oracleインスタンスのデータおよび制御情報を含む共有メモリー構造のグループ。

親トピック: 用語集

システム識別子(SID)

Oracleインスタンスの一意の名前。Oracleデータベース間を切り替えるには、ユーザーが目的のSIDを指定する必要がある。SIDは、tnsnames.oraファイル内の接続記述子のCONNECT DATA部分と、listener.oraファイル内のネットワーク・リスナーの定義に含まれる。

親トピック: 用語集

第三者攻撃

第三者によるメッセージの不正傍受という特徴を持つセキュリティ攻撃。第三者は、メッセージを復号化して再暗号化し(元のメッセージを変更する場合と変更しない場合があります)、元の宛先である受信者に転送します。これらの処理はすべて、正当な送受信者が気付かないうちに行われます。このタイプのセキュリティ攻撃は、認証が行われていない場合のみ機能する。以前は介在者攻撃と呼ばれていました。

親トピック: 用語集

チケット

所有者の識別に役立つ情報。「初期チケット」および「サービス・チケット」を参照。

親トピック: 用語集

tnsnames.ora

接続記述子が含まれているファイル。各接続記述子はネット・サービス名にマップされます。すべてのクライアントまたは各クライアントで使用するために、このファイルを集中して維持することも、ローカルで維持することもできます。このファイルは通常、プラットフォームに応じて次の場所にあります。

- (UNIXの場合)ORACLE_HOME/network/admin
- (Windowsの場合)ORACLE_BASE¥ORACLE_HOME¥network¥admin

親トピック: 用語集

トークン・カード

ユーザーが容易に認証サービスを利用できるように、数種類のメカニズムを提供するデバイス。一部のトークン・カードは、認証サービスと同期されている1回かぎりのパスワードを提供する。サーバーは認証サービスとやりとりすることにより、トークン・カードが提供するパスワードをいつでも検証できる。チャレンジ・レスポンス・ベースで動作するトークン・カードもある。その場合、サーバーはユーザーがトークン・カードに入力するチャレンジ(番号)を提供する。そして、トークン・カードは別の番号(チャレンジから暗号的に導出)を提供し、それをユーザーがサーバーに提供する。

親トピック: 用語集

トランスポート・レイヤー

データ・フロー制御とエラー・リカバリ方式を通じてエンドツーエンドの信頼性を維持するネットワーキング・レイヤー。Oracle Net Servicesは、トランスポート・レイヤーにOracleプロトコル・サポートを使用します。

親トピック: 用語集

Transport Layer Security (TLS)

ネットワーク接続を保護するための業界標準プロトコル。TLSプロトコルはSSLプロトコルの後継です。公開キー・インフラストラクチャ(PKI)を使用した認証、暗号化およびデータの整合性を提供します。TLSプロトコルは、Internet Engineering Task Force (IETF)によって開発されています。

親トピック: 用語集

信頼できる証明書

一定の信頼度を有すると認定されたサード・パーティの識別情報で、ルート・キー証明書とも呼ばれることがある。信頼できる証明書は、エンティティが本人(本物)であるという識別情報を検証する際に使用される。通常は、信頼する認証局を信頼できる証明書と呼ぶ。複数レベルの信頼できる証明書がある場合、証明連鎖で下位レベルにある信頼できる証明書では、それより上位レベルの証明書すべての再検証を必要としない。

親トピック: 用語集

信頼できる認証局

「認証局」を参照してください。

親トピック: 用語集

トラスト・ポイント

「信頼できる証明書」を参照してください。

親トピック: 用語集

ユーザー名

データベース内のオブジェクトに接続およびアクセスするための名前。

親トピック: 用語集

ユーザー・スキーマ・マッピング

ユーザーが存在するディレクトリ内のベースおよびユーザーがマップされるデータベース・スキーマの名前という値のペアを含むLDAPディレクトリ・エントリ。マッピングで参照されるユーザーは、データベースへの接続時に指定されたスキーマに接続されます。ユーザー・スキーマ・マッピング・エントリは、1つのデータベースにのみ適用できるか、ドメイン内のすべてのデータベースに適用できる。「共有スキーマ」を参照してください。

親トピック: 用語集

ユーザー/スキーマの分割

「共有スキーマ」を参照してください。

親トピック: 用語集

ユーザー検索ベース

LDAPディレクトリ内のユーザーが存在するノード。

親トピック: 用語集

ビュー

1つ以上の表(またはその他のビュー)の選択的な表示で、構造とデータの両方を示す。

親トピック: 用語集

ウォレット

個々のエンティティのセキュリティ資格証明を格納したり、管理するために使用されるデータ構造。

親トピック: 用語集

Windowsシステム固有の認証

Windowsサーバーおよびそのサーバー上で稼働しているデータベースへのクライアントのシングル・ログイン・アクセスを可能にする認証方式。

親トピック: 用語集

X.509

デジタル証明 の業界標準仕様。

親トピック: 用語集

索引

記号

数字

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X