Oracle® Business Intelligence Server 管理ガイド

リリース 10.1.3.2
2007 年 4 月

ORACLE

Oracle Business Intelligence Server 管理ガイド , リリース 10.1.3.2

部品番号：E05032-01

原本名：Oracle Business Intelligence Server Administration Guide, Version 10.1.3.2

原本部品番号：B31770-01

目次

第 10 章： Oracle BI Server のクエリー環境の管理

第 11 章： Oracle BI Server のクエリー・キャッシュ

第 12 章： Oracle BI Server の接続性およびサード・パーティ製ツール

第 13 章： Oracle BI リポジトリの変数の使用

第 14 章： Oracle BI Server のクラスタ化

第 15 章： Oracle BI におけるセキュリティ

第 16 章： Oracle BI Server のデータソースとしての XML の使用

第 17 章： Oracle BI Server SQL リファレンス

付録 A: Oracle BI Server の使用状況トラッキング・データの説明とログ・ファイル・メソッドの使用

付録 B: Oracle BI Server 認証 API

索引

1 このリリースの新機能

Oracle Business Intelligence Enterprise Edition は、以前 Siebel Systems 社が Siebel Business Analytics Platform として販売していたコンポーネントで構成されており、大幅な機能拡張が行われています。

『Oracle Business Intelligence Server 管理ガイド』は、Oracle Business Intelligence Enterprise Edition のドキュメント・セットの一部です。このマニュアルでは、Oracle Business Intelligence Server の設定に関する情報が記載されています。このマニュアルには、新しい記述と、以前は『Siebel Business Analytics Server Administration Guide』というタイトルで公開されていた記述があります。

Oracle BI Infrastructure をインストール、使用またはアップグレードする前に、Oracle Business Intelligence Enterprise Edition のリリース・ノートに目を通すことをお薦めします。Oracle Business Intelligence Enterprise Edition のリリース・ノートは、次の場所にあります。

- Oracle Business Intelligence Enterprise Edition の CD-ROM
- Oracle Technology Network（http://www.oracle.com/technology/documentation/bi_ee.html）（Oracle Technology Network の無料アカウントを登録するには、http://www.oracle.com/technology/about/index.html にアクセスしてください）

『Oracle Business Intelligence Server 管理ガイド』に記述された新機能

12 ページの表 1 に、リリース 10.1.3.2 のソフトウェアをサポートするために、このリリースのドキュメントに記述された変更内容を一覧表示します。この変更内容には、このマニュアル名の変更と多数の製品名の変更も含まれています。

表 1. 『Oracle Business Intelligence Server 管理ガイド』において変更された機能と説明

項	説明
「Administration Tool のメニュー」（19 ページ）	製品名の変更と機能の変更が反映されるようにメニュー項目が更新されました。
「Administration Tool のキーボート・ショートカット」（22 ページ）	新しいショートカットが追加されました。
「Administration Tool のアイコンと記号」（22 ページ）	物理的なキューブ階層タイプ、不明瞭ビューおよび集計オブジェクトを表す新しいアイコンが追加されました。
「Oracle Siebel Marketing アプリケーション用の機能とオプション」（25 ページ）	項が追加されました。
「リポジトリまたはビジネス・モデルの一貫性のチェック」（28 ページ）	新しい Consistency Check Manager の説明に書き換えられました。
「「Options」ダイアログ・ボックスにおける「General」タブの使用」（31 ページ）	Calculation Wizard の説明が更新され、インポート・リポジトリ・オプションが追加されました。また、Time Series Wizard のオプションとリポジトリ・マージの不一致を警告するオプションが削除されました。
「「Options」ダイアログ・ボックスにおける「Multiuser」タブの使用」（34 ページ）	新しいタブに関する情報が追加されました。
「論理レベル・カウントの自動移入」（38 ページ）	項が追加されました。
「マルチディメンショナル・データソースから物理レイヤーを作成するプロセス」（62 ページ）	SAP/BW と新しいバージョンの Microsoft Analysis Services に対するサポートが含まれるように内容が更新されました。
「マルチディメンショナル・データソースからの物理スキーマのインポート」（63 ページ）	SAP/BW に対するサポートが含まれるようにオブジェクトのインポート手順が更新されました。
「物理レイヤーでのデータベース・オブジェクトの手動作成」（65 ページ）	仮想プライベート・データベースをソースとして使用するデータベースの作成に必要なフィールドが追加されました。
「接続プールの作成または変更」（72 ページ）	任意のデータソースに対して接続プールを設定する際に使用する一般的なプロパティが更新されました。

表 1.　『Oracle Business Intelligence Server 管理ガイド』において変更された機能と説明

項	説明
「マルチディメンショナル・データソースに対する接続プールのプロパティの設定」（78 ページ）	接続プールを設定する際に使用する一般的なプロパティが更新されました。
「物理テーブルについて」（85 ページ）	項が更新されました。
「物理別名テーブルについて」（86 ページ）	項が追加されました。
「物理テーブルの一般プロパティの作成および管理」（89 ページ）	マルチディメンショナル・データソースに対する変更が含まれるように内容が更新されました。また、別名テーブルの新しいプロパティが追加されました。
「マルチディメンショナル・データソースのメジャーについて」（92 ページ）	項が追加されました。
「物理テーブルでの列およびキーの作成と管理」（91 ページ）	別名テーブルに関する指示が含まれるように項と手順が更新されました。
「マルチディメンショナル・データソースの物理レイヤーにおける階層の設定」（95 ページ）	新しいプロパティが反映されるように項が更新されました。
「メンバー・カウントの更新」（97 ページ）および「物理キューブ・テーブルのメンバーの表示」（98 ページ）	メンバー・カウントの更新とメンバー・データの表示に関する項が追加されました。
「不明瞭ビューのデプロイ」（107 ページ）	不明瞭ビューのデプロイ、アンデプロイおよび削除を行う方法を説明する項が追加されました。
「ディメンションの作成」（130 ページ）	項が更新されました。
「ディメンション・レベルとキーの作成」（131 ページ）	項が更新され、時間ディメンションと時間順キーの操作方法に関する項が追加されました。
「時間ディメンションにおける時間順キーの選択とソート」（134 ページ）	項が追加されました。
「別のリポジトリからのインポート」（159 ページ）	リポジトリからのインポート・プロセスにおける変更に関する情報が追加されました。
「リポジトリ・メタデータのクエリーと管理」（160 ページ）	クエリーを保存する手順と保存済クエリーを削除する手順が追加されました。また、クエリーを実行する手順とクエリー結果を外部ファイルに保存する手順が更新されました。
「プロジェクトへのメタデータ・サブセットの抽出について」（171 ページ）	プロジェクト抽出機能が動作する仕組みに関する説明が追加されました。
「SA System サブジェクト領域の設定」（184 ページ）	項が改訂され、認可と認証の構成オプションに関する注意事項が追加されました。

表 1.　『Oracle Business Intelligence Server 管理ガイド』において変更された機能と説明

項	説明
依存性トラッキングを使用した Analytics メタデータの抽出に関する項	この項は、「メタデータ・ディクショナリの生成と配置」（192 ページ）に書き換えられました。
Time Series Wizard	「Oracle BI Administration Tool のユーティリティと式ビルダー」（187 ページ）から削除された項です。
Synchronize Aliases	「Oracle BI Administration Tool のユーティリティと式ビルダー」（187 ページ）から削除された項です。このリリースでは、同期は自動的に実行されます。
「メタデータ・ディクショナリの生成と配置」（192 ページ）	項が追加されました。
「Aggregate Persistence Wizard」（194 ページ）	項が追加されました。
「Calculation Wizard」（195 ページ）	項が追加されました。
「時系列の変換関数について」（201 ページ）	式ビルダーの更新に関する項が追加されました。
「IndexCol 変換関数について」（203 ページ）	式ビルダーの更新に関する項が追加されました。
「使用状況トラッキングの管理」（226 ページ）	使用状況トラッキング機能が動作する仕組みに関する説明が追加されました。
「ODBC 手順を使用したキャッシュの消去と保持」（244 ページ）	SAGetSharedRequestKey ODBC プロシージャに関する情報が追加されました。
「SAP/BW データソースのキャッシュの格納と消去」（245 ページ）	項が追加されました。
「Oracle BI Server のクエリーに関する集計の作成」（250 ページ）	項が追加されました。
「セッション変数の概要と作成」（295 ページ）	Oracle Business Intelligence Disconnected Analytics 変数の説明が追加されました。
「リポジトリ変数の作成」（294 ページ）および「セッション変数の作成」（297 ページ）	ユーザー・インタフェースの設計変更が反映されるよう項が再編成および更新されました。また、仮想プライベート・データベースのパラメータに関する情報が追加されました。
「変数を持つ初期化ブロックの使用について」（298 ページ）および「初期化ブロックの作成プロセス」（301 ページ）	新しいユーザー・インタフェースに準拠するようすべての項が改訂されました。
「初期化ブロックを使用したユーザー認証について」（301 ページ）	タイトルが変更され、内容が改訂されました。

表 1.　『Oracle Business Intelligence Server 管理ガイド』において変更された機能と説明

項	説明
「Oracle BI Server のクラスタ化」（309 ページ）	Oracle BI Scheduler に関する情報がこの章に追加されました。
「NQSConfig.INI ファイルにおけるパラメータの設定」（313 ページ）および「NQClusterConfig.INI ファイルにおけるパラメータの設定」（313 ページ）	パラメータの説明が削除されました。パラメータの説明は、『Oracle Business Intelligence Infrastructure インストレーションおよび構成ガイド』に記載されています。
「LDAP 認証の設定」（333 ページ）および「Oracle BI Server のユーザー認証のメンテナンス」（338 ページ）	LDAP 認証に関する項が拡張および再編成されました。
「Oracle BI Server のユーザー認証のメンテナンス」（338 ページ）	『Oracle Business Intelligence Enterprise Edition デプロイメント・ガイド』への相互参照が追加されました。
「FILTER を使用した条件集計の計算」（373 ページ）	項が追加されました。
「集計関数」（377 ページ）	時系列関数に関する項が追加および更新されました。
「IndexCol」（409 ページ）	変換関数が追加されました。
「使用状況トラッキング・データの説明」（417 ページ）	データ列が追加されました。
「Oracle BI Server 認証 API」（419 ページ）	付録が追加されました。

その他の変更

- リポジトリの「Presentation」レイヤー・カタログがサブジェクト領域と識別されることを説明する注意が、このマニュアル全体に記載されています。

2 Oracle BI Administration Tool の基本

この章では、Administration Tool に含まれるユーザー・インタフェース・コンポーネントの概要について説明します。Administration Tool は、Oracle BI 管理者がリポジトリの作成および編集に使用可能な Windows アプリケーションです。

注意： このマニュアルの各表に記載されているユーザー・インタフェース・コンポーネントは、詳細な説明が必要なもののみです。「Show Toolbar」など、名前から自明なものは記載されていません。

この章の内容は次のとおりです。

Administration Tool のユーザー・インタフェース・コンポーネント

この項では、次の各インタフェース・コンポーネントについて説明します。

Administration Tool のメイン・ウィンドウ

Administration Tool のメイン・ウィンドウには、リポジトリにおける次の 3 つの部分がグラフィカルに表示されます。

- **「Physical」レイヤー** :Oracle BI Server のクエリー発行先データソースの物理構造を表します。このレイヤーは、Administration Tool の右ペインに表示されます。
- **「Business Model and Mapping」レイヤー**：リポジトリにおける情報の論理構造を表します。ビジネス・モデルには、論理テーブルに編成された論理列、論理結合およびディメンション階層定義が含まれます。また、このレイヤーには、論理列から「Physical」レイヤーのソース・データへのマッピングも含まれます。このレイヤーは、Administration Tool の中央のペインに表示されます。
- **「Presentation」レイヤー**：リポジトリのプレゼンテーション構造を表します。このレイヤーによって、「Business Model and Mapping」レイヤーとは異なるビューをユーザーに対して実現できます。このレイヤーは、Administration Tool の左ペインに表示されます。

19 ページの図 1 に、前述の箇条書きで説明したように、リポジトリの 3 つのレイヤーを示します。

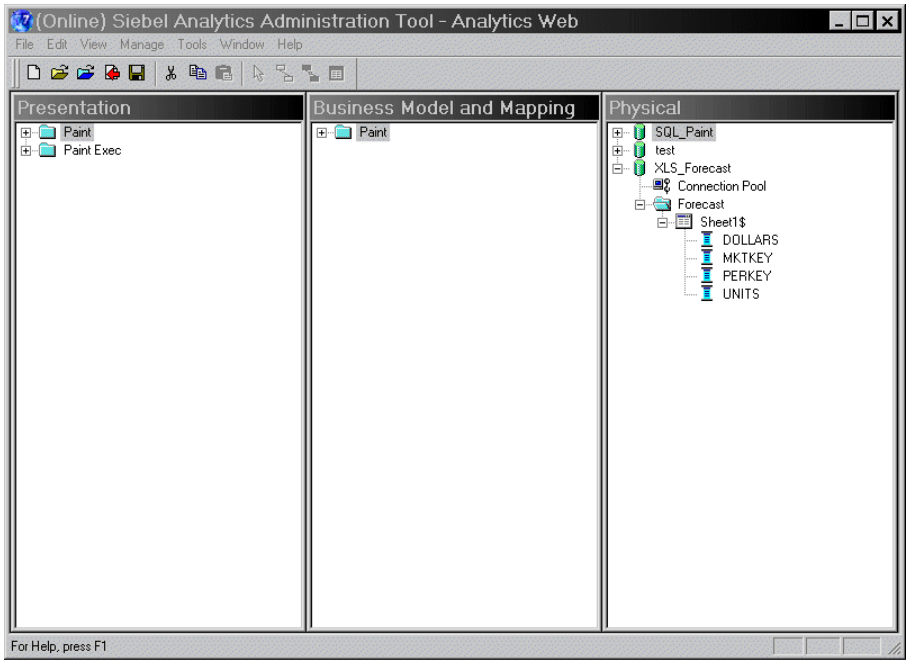


図 1.　Administration Tool のメイン・ウィンドウの例

Administration Tool のメニュー

Administration Tool には、次のメニューがあります。

File

「File」メニューには、リポジトリの各種操作を行うオプションと、リポジトリがオンライン・モードで開いているときにアクティブになるサーバー関連オプションが用意されています。リポジトリが開いていないときは、「File」メニューで使用できるコマンドは少なくなります。

Edit

「Edit」メニューのオプションは、リポジトリ内のオブジェクトに対して基本的な編集操作（切取り、コピー、貼付け、複製および削除）を実行するときに使用できます。また、オブジェクトのプロパティのいくつかを表示および編集できます。

View

「View」メニューのオプションは、特定のメタデータ・ペインのビューを切り替えるとき、結合ダイアグラムにアクセスするとき、およびリポジトリのビューをリフレッシュするときにそれぞれ使用します。

Manage

「Manage」メニューにより、20 ページの表 2 に説明する管理機能にアクセスすることができます。

表 2. 「Manage」メニューの機能

メニュー・オプション	説明
Jobs	このオプションは、リポジトリがオンライン・モードで開いている場合に使用できます。Job Manager は、Oracle Business Intelligence Scheduler に対する管理インタフェースです。BI Scheduler の使用方法の詳細は、『Oracle Business Intelligence Scheduler ガイド』を参照してください。
Sessions	このオプションは、リポジトリがオンライン・モードで開いている場合に使用できます。Session Manager では、リポジトリ変数とセッション変数の現行値を含めて、システムのアクティビティを監視できます。
Cache	このオプションは、リポジトリがオンライン・モードで開いていて、キャッシュが有効である場合に使用できます。Cache Manager では、キャッシュの監視と管理ができます。
Clusters	このオプションは、Oracle BI Cluster Server がインストールされている場合に使用できます。Cluster Manager により、クラスタの運用とアクティビティに関する監視と管理を行います。
Security	Security Manager によりリポジトリのセキュリティ情報を表示し、セキュリティの構成と管理を一元化します。
Joins	Joins Manager により、物理結合と論理結合を操作できます。
Variables	Variables Manager により、変数を作成、編集または削除できます。
Projects	Project Manager により、プロジェクトまたはプロジェクト要素を作成、編集または削除できます。プロジェクト要素には、プレゼンテーション・カタログ（リポジトリのサブジェクト領域）、論理ファクト・テーブル、グループ、ユーザー、変数、初期化ブロックなどがあります。プロジェクトは、マルチユーザーで開発を行うときに使用します。詳細は、「マルチユーザー開発環境の設定（管理者）」（174 ページ）を参照してください。
Marketing	Oracle Siebel Marketing 製品に適用されます。この製品を使用するには、別のライセンスが必要です。Siebel Marketing と Oracle BI とを併用する方法の詳細は、Siebel Marketing アプリケーションの管理者ガイドを参照してください。

Tools

「Tools」メニューのオプションは、すべての行カウントを更新するとき、「Query Repository」ダイアログ・ボックスを開くとき、Administration Tool のオプションを設定するときおよび各種ユーティリティを操作するときにそれぞれ使用できます。

Window

「Window」メニューのオプションは、開いているレイヤー・ウィンドウを重ねて表示したり並べて表示するときや、それらの表示を切り替えるときに使用できます。

注意：「Cascade」と「Tile」のオプションを使用できるのは、「Tools」→「Options」ダイアログ・ボックス→「General」タブの「Tile when resizing」チェック・ボックスの選択が解除されているときのみです。

Help

「Help」メニューから、次の情報を入手できます。

- ヘルプ・トピック：Administration Tool のオンライン・ヘルプ・システムにアクセスします。
- Oracle Web ページ：オラクル社の Web サイトにアクセスします。
- Administration Toolのバージョン情報：Oracle BI Server Administration Toolのバージョン情報を表示します。

Administration Tool のツールバー

頻繁に使用する機能は、ツールバーを使用するとアクセスできます。

ツールバーのオン / オフを切り替えるには

- 「Tools」→「Options」→「Show Toolbar」を選択します。

ツールバーをドッキングするには

- ツールバーの左側にある移動ハンドル上にカーソルを移動してからクリックし、目的の場所までドラッグします。

Administration Tool のキーボート・ショートカット

22 ページの表 3 に、Administration Tool で使用可能なキーボート・ショートカットを示します。これによって頻繁に実行する作業を実行できます。

表 3.　キーボート・ショートカット

メニュー	コマンド	ショートカット
File	New	[Ctrl]+[N]
	「Open」→「Offline」	[Ctrl]+[F]
	「Open」→「Online」	[Ctrl]+[L]
	Save	[Ctrl]+[S]
	Check Global Consistency	[Ctrl]+[K]
Edit	Cut	[Ctrl]+[X]
	Copy	[Ctrl]+[C]
	Paste	[Ctrl]+[V]
	Delete	[Del]
View	Refresh	[F5]
Tools	Show Consistency Checker	[Ctrl]+[E]
	Query Repository	[Ctrl]+[Q]

Administration Tool のアイコンと記号

ほとんどのアイコンでは、そのアイコンのダブルクリックとダイアログ・ボックスのタイトル・バーの読取りを実行すると、そのオブジェクト・タイプを取得できます。22 ページの表 4 に、タイトル・バーに名前が詳述されるアイコンと記号を示します。この表には、他のアイコンと組み合せて使用する記号がいくつかあり、その意味が拡張されます。

表 4.　アイコンと記号

アイコンまたは記号	説明
	ストアド・プロシージャをコールするオブジェクトです。詳細は、「Physical Table」ダイアログ・ボックスの「General」タブにある「Object Type」オプションで指定します。
	ビュー・オブジェクトです。
	デプロイ後の不明瞭ビュー・オブジェクトです。

表 4.　アイコンと記号

アイコンまたは記号	説明
	物理テーブルまたは論理テーブルの主キーです（黄色）。
	Joins Manager の物理テーブルまたは論理テーブルの外部キーです（灰色）。
	論理ディメンション・レベルのキーです（青色）。
	マルチディメンショナル・データソース物理レベルのキーです（緑色）。
	Joins Manager における複合物理結合または複合論理結合です。
	不明な階層タイプです。通常、このアイコンは、キューブのインポート時に階層に割り当てられます。インポート後に、有効なタイプを各階層に割り当てる必要があります。
	「Physical」レイヤーにおけるマルチディメンショナル・データソースの完全均衡階層タイプです。
	「Physical」レイヤーにおけるマルチディメンショナル・データソースの不均衡階層タイプです。
	「Physical」レイヤーにおけるマルチディメンショナル・データソースの不完全均衡階層タイプです。
	「Physical」レイヤーにおけるマルチディメンショナル・データソースのネットワーク階層タイプです。
	「Business Model and Mapping」レイヤーにおけるレベルです。
	「Business Model and Mapping」レイヤーにおけるレベルであり、レベル・キーには別のレベルの列が 1 つ以上含まれます。
	「Physical」レイヤーにおけるマルチディメンショナル・データソースのレベルです。
	物理列または論理列です。
	集計ルールのある論理列です。
	導出された論理列です。
	マルチディメンショナル・データソースの物理キューブ列です。このアイコンは、メジャーでない列を表します。

表 4.　アイコンと記号

アイコンまたは記号	説明
	マルチディメンショナル・データソースの物理キューブ列です。このアイコンは、メジャーである列を表します。
	無効な項目です。たとえば、物理マッピングのない列は無効になる場合があります。
	「Business Model and Mapping」レイヤーにおける縮小ビジネス・モデルで、クエリーを発行できないものです。
	「Business Model and Mapping」レイヤーにおける拡張ビジネス・モデルで、クエリーを発行できないものです。
	属性定義が含まれる項目です。
	他の記号上に重ねて表示され、チェックインされていない新規項目を示します（オンライン・モードでのみ表示される）。たとえば、このアイコンが別名テーブル・アイコン上に表示される場合は、その別名テーブルが新規であることを示します。
	システムの DSN ODBC エントリです。「Import」ダイアログ・ボックスに表示されます。
	メジャー定義です。
	他のアイコン上に重ねて表示され、チェックアウトされていないオブジェクトを示します。たとえば、このアイコンがテーブル・アイコン上にある場合は、そのテーブルがチェックアウトされていることを示します。
	静的なリポジトリ変数です。
	動的なリポジトリ変数です。
	システム・セッション変数です。
	システム以外のセッション変数です。
	初期化ブロックです。
	グループ対応付けです（「Query Repository」ダイアログ・ボックスの結果表示でのみ表示される）。
	レベル間のリレーションシップです（「Query Repository」ダイアログ・ボックスの結果表示でのみ表示される）。

表 4.　アイコンと記号

アイコンまたは記号	説明
	タイプ権限です（「Query Repository」ダイアログ・ボックスの結果表示でのみ表示される）。
	クエリー権限です（「Query Repository」ダイアログ・ボックスの結果表示でのみ表示される）。
	権限パッケージです（「Query Repository」ダイアログ・ボックスの結果表示でのみ表示される）。
	オブジェクト権限です（「Query Repository」ダイアログ・ボックスの結果表示でのみ表示される）。
	他のアイコン上に重ねて表示され、切取りが実行されたオブジェクトを示します。たとえば、このアイコンが別名テーブル記号にある場合は、切り取られた項目が別名テーブルであることを示します。

Administration Tool のオンライン・ヘルプ

ほとんどのウィンドウとダイアログ・ボックスにはヘルプ・ボタンがあります。このボタンにより、作業の実行に役立つ情報が記載されたオンライン・ヘルプ・トピックが表示されます。

Oracle Siebel Marketing アプリケーション用の機能とオプション

Oracle BI Server Administration Tool には、Siebel Marketing を所有している組織で使用するための機能とオプションがいくつかあります。詳細は、Siebel Marketing アプリケーションの管理者ガイドを参照してください。

リポジトリのオンライン・モードとオフライン・モード

リポジトリの編集は、オンライン・モードまたはオフライン・モードのどちらでも実行できます。リポジトリを開いているモードに応じて、実行可能な作業が異なります。

この項の内容は次のとおりです。

- リポジトリをオフライン・モードで開く方法（26 ページ）
- リポジトリをオンライン・モードで開く方法（26 ページ）
- 変更のチェックイン（27 ページ）

リポジトリをオフライン・モードで開く方法

オフライン・モードを使用して、Oracle BI Server にロードされていないリポジトリを表示および変更します。Oracle BI Server にロードされているリポジトリをオフライン・モードで開くと、リポジトリが読取り専用モードで開きます。オフライン・モードでは、一度に 1 つの Administration Tool セッションでのみリポジトリを編集できます。

リポジトリをオフライン・モードで開くには

1. Administration Tool で、「File」→「Open」→「Offline」を選択します。
2. 開く対象となるリポジトリにナビゲートし、「Open」を選択します。
3. 「Open Offline」ダイアログ・ボックスで、有効なユーザー ID とパスワードを入力してから「OK」をクリックします。

   編集可能な状態でリポジトリが開きます。

   注意：サーバーが実行中で、開こうとしているリポジトリがそのサーバーにロードされている場合、リポジトリは読取り専用モードでのみ開きます。ロードされているリポジトリを編集するには、オンライン・モードで開く必要があります。また、リポジトリをオフライン・モードで開いてからサーバーを起動した場合は、リポジトリが他のユーザーにも使用可能になり、リポジトリに対して行った変更はサーバーの再起動時に適用されます。

リポジトリをオンライン・モードで開く方法

オンライン・モードを使用して、Oracle BI Server にロードされているリポジトリを表示および変更します。オンライン・モードでリポジトリを開くには、Oracle BI Server が実行されている必要があります。オンライン・モードでは、オフライン・モードでは実行できない作業を実行できます。オンライン・モードでは、次の作業を実行できます。

- スケジュール済ジョブの管理
- ユーザー・セッションの管理
- クエリー・キャッシュの管理
- クラスタ化されたサーバーの管理
- Oracle BI Server の停止

リポジトリをオンライン・モードで開くには

1. Administration Tool で、「File」→「Open」→「Online」を選択します。

   「Open Online Repository」ダイアログ・ボックスが表示され、そこからデータソースを選択します。

   このダイアログ・ボックスに表示されるデータソースは、コンピュータ上におけるすべてのユーザー DSN とシステム DSN で、Oracle BI ODBC ドライバを使用して構成されたものです。

2 データソースを選択して有効なユーザー ID とパスワードを入力してから、「OK」をクリックします。

選択したデータソースに対応するビジネス・モデルを含むリポジトリが開きます。

注意： リポジトリを広範囲に操作する場合（多数のオブジェクトをチェックアウトする場合など）、「Load all objects」オプションを選択します。このオプションでは、選択されたオブジェクトではなく、すべてのオブジェクトがただちにロードされます。初期接続の時間が若干長くなる場合がありますが、ツリーにある項目を開く処理や項目のチェックアウトが高速に実行されます。

変更のチェックイン

オンライン・モードで開いているリポジトリで作業を行う場合は、リポジトリに対する変更をチェックインする前に一貫性チェックの実行を要求されます。

リポジトリに変更を加えてから変更をチェックインせずにリポジトリをクローズしようとすると、必要な操作の選択を要求するダイアログ・ボックスが自動的に開きます。オブジェクトを親の下位から移動してからその親を削除する場合は、削除操作が実行される前に変更のチェックインを要求されます。

「Check in Changes」ダイアログ・ボックスを使用して、次の作業を行います。

- 他のアプリケーションに対して変更をただちに使用可能にする作業を行います。変更のチェックイン後に Oracle BI Server にクエリーを発行するアプリケーションでは、変更がただちに認識されます。サーバーにクエリーを発行中のアプリケーションでは、変更済項目に次回アクセスするときに、変更が認識されます。
- ただちにディスクに書き込むリポジトリ変更を指定する作業を行います。Oracle BI Server が異常終了して停止した場合は、「Check in Changes」ダイアログ・ボックスを使用すると、リポジトリにチェックイン済の変更をリカバリできます。チェックインされているがディスクに保存されていない変更は、サーバーのエラー・リカバリ処理によってリストアされます。この処理は、Oracle BI Server が異常終了して停止した場合に自動的に実行されます。

変更を使用可能にし、ただちにディスクに保存するには

- Administration Tool で、「File」→「Check in Changes」を選択します。

Administration Tool によって無効な変更が検出された場合は、問題の状態を警告する情報メッセージが表示されます。問題に対処して、再度チェックインを実行してください。

注意： オンライン・モードで開いているリポジトリを変更してから Oracle BI Server を停止する場合は、このオプションは使用できません。これは、サーバーの停止時に変更が自動的に保存されるためです。

リポジトリまたはビジネス・モデルの一貫性のチェック

リポジトリに対してクエリーを発行可能にする前に、リポジトリのメタデータでは一貫性のチェックをパスする必要があります。Consistency Check Manager では、一貫性チェック用ルールの有効化と無効化、一貫性のないオブジェクトへのナビゲートと修正、および特定のオブジェクトに対する一貫性チェックの制限を指定できます。

Consistency Check Manager では、メタデータの外部にあるオブジェクトの妥当性は接続を使用してチェックされません。メタデータの一貫性のみがチェックされ、メタデータ外部の物理オブジェクトへのマッピングはチェックされません。データベースに接続できない場合やオブジェクトがデータベースから削除されている場合、Consistency Check Manager では対応するエラーが報告されます。

一貫性チェックから返されるメッセージには、次のタイプがあります。

- **エラー**：このメッセージには、対処が必要なエラーに関する説明文があります。メッセージの情報を参考にして一貫性のない部分を修正してから、一貫性チェックを再度実行します。次に、エラー・メッセージの例を示します。

  ```
  [38082] Type of Hierarchy '"ORT_C41"..."ORT_C41/MDF_BW_QO2"."Product Hierarchy for Material MARA"' in Cube Table '"ORT_C41"..."ORT_C41/MDF_BW_QO2"' needs to be set.
  ```

  注意：無効化するオブジェクトに一貫性がない場合は、それに対してクエリーを発行不可にするかどうかを確認するメッセージが表示されます。

- **警告**：このメッセージは、Oracle BI 管理者の意向によって、エラーになる場合もあれば、エラーにならない場合もあります。たとえば、無効化された結合であることを示す警告メッセージは、Oracle BI 管理者が意図的にその種の結合を無効化していること（循環結合条件が排除されているなど）により出力される場合があります。次に、警告メッセージの例を示します。

  ```
  [39024] Dimension '"Paint"."MarketDim"' has defined inconsistent values in its levels' property 'Number of elements'.
  ```

- **ベスト・プラクティス**：このメッセージは、非一貫性には該当しない条件についての情報を示します。このメッセージは、条件がベスト・プラクティスの推奨事項に違反する場合に表示されます。次に、ベスト・プラクティス・メッセージの例を示します。

  ```
  [89001] The Physical Table '"XLS_Forecast"."Forecast".."Sheet1$"' has no keys defined.
  ```

注意：以前のソフトウェア・リリースからアップグレード後にリポジトリの一貫性をチェックすると、以前の一貫性チェックでは報告されなかったメッセージが出力されることがあります。これは通常、新しいエラーが発生したのではなく、アップグレード前には検出されなかった非一貫性が存在することを示すものです。

Consistency Check Manager の「Messages」タブで、列見出しをクリックすると、メッセージの行をソートできます。また、ステータス・バーには、表示されているすべての行のサマリー情報が表示されます。

Consistency Check Manager の設定

インストール時に、ルールのデフォルト・サブセットがインストールされます。ユーザーは、各ワークステーションで、ルールのデフォルト・サブセットを使用したり、ルールを追加または削除してサブセットを変更できます。デフォルトでは、すべての一貫性チェック・ルールが有効化され、一貫性チェックの実行後にすべてのタイプのメッセージが表示されるよう設定されます。任意の一貫性チェック・ルールを無効にし、メッセージ・リストを変更して特定のメッセージ・タイプを非表示にできます。ただし、1 つ以上のメッセージ・タイプが有効にされている必要があります。

Consistency Check Manager を設定するには

1. Administration Tool で、「Tools」→「Show Consistency Checker」を選択します。
2. Consistency Check Manager の「Messages」タブで、次の手順を実行します。
   a. 非表示にするメッセージ・タイプに対応するチェック・ボックスの選択を解除します。
   b. メッセージで各オブジェクト名を完全修飾名にして表示する場合は、「Show Qualified Name」チェック・ボックスを選択します。
3. 「Options」タブをクリックします。
4. 各メッセージ・タイプを開きます。
5. 有効にされたルールを無効にする場合は、ルールを選択してから「Disable」をクリックします。
6. 無効にされたルールを有効にする場合は、ルールを選択してから「Enable」をクリックします。
7. この後、一貫性チェックを実行しない場合は、「閉じる」をクリックします。

リポジトリの一貫性のチェック

現在、次の方法で一貫性をチェックできます。

- リポジトリのグローバルな一貫性は、「File」メニューおよび Consistency Check Manager（「Check All Objects」）からチェックできます。ルールを Consistency Check Manager で無効にした場合は、それらのルールはチェックされません。

  注意：無効にするオブジェクトに一貫性がない場合は、それに対してクエリーを発行不可にするかどうかを確認するメッセージが表示されます。

- ビジネス・モデルの一貫性は、ビジネス・モデルの右クリック・メニューからチェックできます。
- リポジトリ内の一部またはすべてのオブジェクトの一貫性は、Consistency Check Manager からチェックできます。チェック対象のオブジェクトを限定するには、Consistency Check Manager の「Options」タブで、対象外とするオブジェクトのルールを無効にします。

リポジトリの一貫性をチェックするには

1. Administration Tool の「Tools」メニューから、「Show Consistency Checker」を選択します。

   一貫性チェックを現行セッションで実行した場合は、前回チェックしたときのメッセージが「Messages」タブに表示されます。

2. Consistency Check Manager で「Options」タブをクリックし、ルールが適切に設定されていることを確認します。

3 「Messages」タブをクリックし、「Check All Objects」をクリックします。

一貫性チェックを現行セッションで実行した後でルールを変更した場合は、メッセージが異なる場合があります。

4 リポジトリを編集して非一貫性を修正するには、次の手順を実行します。

a 列にある任意のセルをダブルクリックして、該当するオブジェクトのプロパティ・ダイアログ・ボックスを開きます。

b 該当するオブジェクトのプロパティ・ダイアログ・ボックスで、非一貫性を修正してから「OK」をクリックします。

5 スプレッドシートなどの別のファイルへの貼付けができるようにメッセージをコピーするには、「Copy」をクリックします。

6 一貫性チェックを再度実行する場合は、「Check All Objects」をクリックします。

「Tools」メニューから「Show Consistency Checker」を選択したか「File」メニューから「Check Global Consistency」を選択した場合、「Consistency Check Manager」ダイアログ・ボックスの右上隅にあるリフレッシュ・ボタンをクリックすると、すべてのオブジェクトがチェックされます。

7 終了したら、「閉じる」をクリックします。

リポジトリにおける単一オブジェクトの一貫性をチェックするには

1 Administration Tool で、オブジェクトを右クリックし、「Check Consistency」を選択します。

オブジェクトに一貫性がない場合は、一連のメッセージが表示されます。

2 リポジトリを編集して非一貫性を修正するには、次の手順を実行します。

a 列にある任意のセルをダブルクリックして、該当するオブジェクトのプロパティ・ダイアログ・ボックスを開きます。

b 該当するオブジェクトのプロパティ・ダイアログ・ボックスで、非一貫性を修正してから「OK」をクリックします。

3 スプレッドシートなどの別のファイルへの貼付けができるようにメッセージをコピーするには、「Copy」をクリックします。

4 オブジェクトの一貫性チェックを再度実行する場合は、ダイアログ・ボックスの右上隅にあるリフレッシュ・ボタンをクリックします。

注意：「Check All Objects」をクリックすると、リポジトリ内のすべてのオブジェクトがチェックされます。

プリファレンスの設定

「Options」ダイアログ・ボックスを使用して、Administration Tool の各種プリファレンスを設定できます。

この項の内容は次のとおりです。

「Options」ダイアログ・ボックスにおける「General」タブの使用

「Options」ダイアログ・ボックスの「General」タブを使用して、Administration Tool の一般的なプリファレンスを設定します。

一般的なプリファレンスを設定するには

1. Administration Tool で、「Tools」→「Options」を選択します。
2. 「Options」ダイアログ・ボックスで「General」タブを選択します。
3. 必要なオプションを選択します。

   次の一覧表に、さらに説明が必要なオプションを示します。

オプション	選択時の動作
Tile when resizing	Administration Tool のサイズ変更時に、リポジトリのレイヤー・ペインが自動的に並べて表示されます。このオプションが選択されていると、Administration Tool の「Window」メニューの「Cascade」および「Tile」のオプションは使用できません。
Display qualified names in diagrams	テーブル・ソースの識別が容易になります。
Display original names for alias in diagrams	実際に参照されているテーブルの識別が容易になります。

オプション	選択時の動作
Show Calculation Wizard introduction page	Calculation Wizard の概要ページが表示されます。概要ページには、このページを今後非表示にするためのオプションも含まれます。 Calculation Wizard を使用して、既存の 2 列を比較する新しい計算列を作成したり、メトリックを一括処理（集計処理）して作成しますが、NULL やゼロ除算ロジックなどの既存のエラー・トラップも含まれます。 Calculation Wizard を「Business Model and Mapping」レイヤーで起動するには、数値データ型のディメンション列または論理ファクト列を右クリックし、「Calculation Wizard」オプションを選択します。 **注意：**時間ディメンションとして識別されない論理テーブルでは、時系列計算はできません。詳細は、「ディメンションの作成と管理のプロセス」（130 ページ）の時間ディメンションに関する項を参照してください。
Check out objects automatically	（オンライン・モードのみ）オブジェクトをダブルクリックすると、そのオブジェクトが自動的にチェックアウトされます。 **注意：**このオプションが選択されていない場合は、オブジェクトの編集前に、オブジェクトのチェックアウトを要求されます。
Show row count in physical view	「Physical」レイヤーにおける物理テーブルと物理列の行カウントが表示されます。行カウントは、更新されないかぎり、初期表示されません。行カウントを更新するには、「Tools」→「Update All Row Counts」を選択します。または、「Physical」レイヤーのテーブルまたは列を右クリックして、「Update Row Count」オプションを選択します。 注意：ストアド・プロシージャのコールである項目（「Physical Table」ダイアログの「General」タブにある「Object Type」ドロップダウン・リストから選択）には、行カウントは表示されません。XML、XML サーバーまたはマルチディメンショナル・データベースには、行カウントは使用できません。新規オブジェクトの行カウントは、チェックインするまで更新できません。
Prompt when moving logical columns	論理列の移動に際して、既存の論理テーブル・ソースを無視したり指定することができ、また新規の論理テーブル・ソースを作成することもできます。
Remove unused physical tables after Merge	未使用の物理オブジェクトのリポジトリをクリーンアップするユーティリティを実行します。実行後のリポジトリは、サイズが小さくなる場合があります。
Allow import from repository	このオプションを選択すると、「File」メニューの「Import from repository」オプションが使用可能になります。 **注意：**デフォルトでは、「File」メニューの「Import from repository」オプションは無効で、このオプションは将来サポートされる予定はありません。プロジェクトは、インポート対象のオブジェクトを含むリポジトリで作成してから、リポジトリ・マージを使用して現行リポジトリにインポートすることをお薦めします。詳細は、「プロジェクトへのメタデータ・サブセットの抽出について」（171 ページ）を参照してください。

「Options」ダイアログ・ボックスにおける「Repository」タブの使用

「Repository」タブでは、次のオプションを設定できます。

- **Show tables and dimensions only under display folders**: Oracle BI 管理者は、表示フォルダを作成して、「Physical」レイヤーと「Business Model and Mapping」レイヤーにおいてオブジェクトを整理できます。メタデータ的な意味はありません。表示フォルダの作成後は、選択したオブジェクトが、フォルダにはショートカットとして、データベースまたはビジネス・モデル・ツリーにはオブジェクトとしてそれぞれ表示されます。表示フォルダにはショートカットのみが表示されるように、オブジェクトを非表示にできます。

  表示フォルダの作成方法の詳細は、「「Physical」レイヤーの表示フォルダの設定」（102 ページ）と「「Business Model and Mapping」レイヤーにおける表示フォルダの設定」（140 ページ）を参照してください。

- **Hide level-based measure**: このオプションを選択すると、レベルベースのメジャー（集計ルールが NONE でない列）が、レベルの下にある「Business Model and Mapping」レイヤーにおいて表示されなくなります。

「Repository」タブのオプションを設定するには

1. メニュー・バーから、「Tools」→「Options」を選択します。
2. 「Options」ダイアログ・ボックスで「Repository」タブをクリックします。
3. 「Repository」タブで、目的のオプションを選択してから「OK」をクリックします。

「Options」ダイアログ・ボックスにおける「Sort Objects」タブの使用

「Sort Objects」タブを使用して、Administration Tool にアルファベット順で表示されるリポジトリ・オブジェクトを指定します。

アルファベット順で表示されるリポジトリ・オブジェクトを指定するには

1. Administration Tool で、「Tools」→「Options」を選択します。
2. 「Options」ダイアログ・ボックスで「Sort Objects」タブを選択します。
3. アルファベット順で表示するオブジェクトに対応するチェック・ボックスを選択します。

   たとえば、「Physical」レイヤーに表示されるデータベース・オブジェクトをアルファベット順で表示する場合は、「Database」チェック・ボックスを選択します。

「Options」ダイアログ・ボックスにおける「Cache Manager」タブの使用

「Cache Manager」タブを使用して、Cache Manager に表示する列と、Cache Manager の「Cache」タブにおける列の表示順序を選択します。

Cache Manager に表示する列を選択するには

1. Administration Tool で、「Tools」→「Options」を選択します。
2. 「Options」ダイアログ・ボックスで「Cache Manager」タブを選択します。
3. Cache Manager に表示する列に対応するチェック・ボックスを選択します。

   すでに選択済のチェック・ボックスをクリックすると、その選択が解除されます。チェック・ボックスの選択が解除された項目は表示されません。
4. Cache Manager における列の順序を変更するには、項目を選択してから「Up」と「Down」のボタンを使用してその位置を変更します。

「Options」ダイアログ・ボックスにおける「Multiuser」タブの使用

「Multiuser」タブを使用して、マルチユーザー開発ディレクトリへのパスを指定します。詳細は、「マルチユーザー開発ディレクトリへのポインタの設定」（176 ページ）を参照してください。

「Options」ダイアログ・ボックスにおける「More」タブの使用

「More」タブを使用して、Administration Tool の各種ダイアログ・ボックスにおいてツリーをスクロールするときの速度と、結合ダイアグラムのデフォルト・ウィンドウ・サイズをそれぞれ設定します。

スクロール速度とデフォルト・ウィンドウ・サイズを設定するには

1. Administration Tool で、「Tools」→「Options」を選択します。
2. 「Options」ダイアログ・ボックスで「More」タブを選択します。
3. 「Scrolling Speed」スライダ上で調整つまみを適切な位置に移動して速度を設定します。
4. 「Default diagram zoom」ドロップダウン・リストで、パーセンテージまたは「Best Fit」を選択します。

   デフォルトのウィンドウ・サイズは「Best Fit」です。「Best Fit」オプションを選択した場合は、次のルールが適用されます。

   - オブジェクト数が 5 以下のときは、ズーム・レベルが 50% になります。
   - オブジェクト数が 6 以上のときは、ズーム・レベルが「Zoom to Fit」に自動的に変更されます。

リポジトリ・オブジェクトに対する権限の設定

「Physical」レイヤーの接続プールと「Presentation」レイヤーのオブジェクトに対して、ユーザーとグループの権限を割り当てることができます。また、Security Manager を使用して、各種権限を設定します。セキュリティの管理方法の詳細は、「Oracle BI Security Manager」（324 ページ）を参照してください。

「Permissions」ダイアログ・ボックスには、現在定義されているすべてのユーザーとグループが表示されます。すべてのユーザーとグループを表示するには、「Show all users/groups」チェック・ボックスを選択します。ユーザーとグループごとに、そのチェック・ボックスの選択または選択解除を行って次のオプションを切り替えることで、オブジェクトに対する権限を許可または禁止できます。

- チェック・マークは、権限が付与されていることを示します。
- X マークは、権限が拒否されていることを示します。
- 空のチェック・ボックスは、権限が変更されていないことを示します。

「Permissions」ダイアログ・ボックスには、次のダイアログ・ボックスからアクセスできます。

- 「Connection Pool」の「General」タブ
- 「Presentation Catalog Folder」の「General」タブ

  注意： Administration Tool でプレゼンテーション・カタログという用語は、サブジェクト領域のことを示します。
- 「Presentation Table」の「General」タブ
- 「Presentation Column」の「General」タブ

接続プールに対する権限を追加または編集するには

1. リポジトリを Administration Tool で開いてから物理データベースを開き、接続プールをダブルクリックします。
2. 「Connection Pool」ダイアログ・ボックスで「Permissions」をクリックします。
3. 「Permissions」ダイアログ・ボックスで、権限の変更対象となるユーザーとグループごとに適切なオプションを選択します。
4. 「OK」をクリックします。

「Permissions」ダイアログ・ボックスにおける列のソート

タイプ、ユーザー名およびグループ名は、6 種類の方法でソートできます。「Permissions」ダイアログ・ボックスには、3 つの列があります。最初の列（ソート可能列）には空欄の見出しがあり、ユーザーまたはユーザー・グループのタイプを表すアイコンがあります。2 番目の列（ソート可能列）にはユーザーやグループのオブジェクトの名前があり、3 番目の列（ソート不可能列）には「Read」チェック・ボックスがあります。ソート方法を変更するには、最初の列または 2 番目の列にある見出しをクリックします。

タイプは 3 種類の方法でソートでき、ユーザーとグループの名前は 2 種類の方法でソートできます。したがって、ソート可能な方法は合計で 6 種類（3 × 2 = 6）になります。次に、タイプ列をクリックして実行可能なソート結果を示します。

- 「Everyone」、「Groups」、「Users」の順（タイプ名による昇順）
- 「Users」、「Groups」、「Everyone」の順（タイプ名による降順）
- タイプ列には順序が適用されない（「User/Group」列にあるすべての名前が「User/Group」列による昇順でソートされ、「Type」値は無視される）

次に、「User/Group」列をクリックして実行可能な 2 種類のソート結果を示します。

- タイプ内での昇順
- タイプ内での降順

「Permissions」ダイアログ・ボックスにおける列のソートの例

最初にすべての行をタイプ別で昇順でソートしてから、その各タイプ内をユーザー名やグループ名別で昇順でソートする場合は、次に示す手順でソートを実行します。

1. 「Everyone」タイプが上部に表示されるまで、タイプ列にある空欄の見出しをクリックします。

   タイプ列が昇順になります。

2. 各タイプ内で「User/Group」の名前列が降順になっている場合は、「User/Group」見出しを 1 回クリックします。

   タイプ列は昇順のままで、各タイプ内のユーザー名やグループ名が昇順になります。

3. ユーザー名やグループ名は昇順のままで、タイプのソートを降順に変更するには、タイプの空欄の見出しを 1 回クリックします。
4. タイプ列のソートを無視に変更し、「User/Group」列にある名前別でのみ昇順にソートするには、タイプの見出しをもう 1 回クリックします。
5. タイプ列は無視したままで、「User/Group」列にある名前のソートを降順に変更するには、「User/Group」見出しをクリックします。

リポジトリにおけるオブジェクトの編集、削除および並替え

この項では、オブジェクトの編集、削除および並替えを行う標準的な手順について説明します。リポジトリのレイヤーに関する説明が記載された章では、各オブジェクトに対するこれらの手順は、作業の実行に特に必要でないかぎり繰り返して説明しません。

- オブジェクトを編集するには、オブジェクトをダブルクリックし、表示されるダイアログ・ボックスで必要なフィールドを入力します。ダイアログ・ボックスによっては、「Edit」をクリックすると、適切なダイアログ・ボックスを開くことができるダイアログ・ボックスがあります。
- オブジェクトを削除するには、オブジェクトを選択してから「Delete」をクリックします。
- オブジェクトを並べ替えるには、オブジェクトを目的の場所までドラッグ・アンド・ドロップします。ダイアログ・ボックスによっては、上向きまたは下向きの矢印を使用すると、オブジェクトを目的の場所に移動できるダイアログ・ボックスがあります。

  注意： 並替えは、特定のオブジェクトに対して、特定のダイアログ・ボックスでのみ実行できます。

テーブルと列の行カウントの表示と更新

ユーザーが行カウントをリクエストすると、Administration Tool によって、物理データベースからすべてまたは選択されたテーブルと列にある行の行数（列の場合は個別の値）が取得され、その値がリポジトリに格納されます。この処理に要する時間は、取得する行カウントの数によって異なります。

すべての行カウントを更新すると、行カウントの取得と格納が行われている間、「Updating Row Counts」ウィンドウが表示されます。「取消」をクリックすると、進行中のテーブル（およびその列）が取得された後に取得処理が停止します。行カウントには、取消し操作よりも前に値が取得されたすべてのテーブルと列が含まれます。

大規模なリポジトリに対してすべての行カウントを更新する場合は、完了所要時間が長くなることがあります。そのため、状況によっては、選択したテーブルと列のカウントのみを更新することが望ましい場合があります。

次の項目には、行カウントを使用できません。

- ストアド・プロシージャのオブジェクト・タイプ。
- XML データソースと XML サーバー・データベース。
- マルチディメンショナル・データソース。メンバー・カウントの詳細は、「メンバー・カウントの更新」（97 ページ）を参照してください。
- CountDistinct 関数がサポートされないデータソース（Microsoft Access や Microsoft Excel など）。
- オンライン・モードの場合、セッション変数の :USER と :PASSWORD にユーザー名とパスワードが設定されている接続プールでは「Update Row Count」は機能しません。

  オフライン・モードの場合、「Set values for variables」ダイアログ・ボックスが表示され、セッション変数の :USER と :PASSWORD に値を設定できます。
- オンライン・モードで物理テーブルまたは列をインポートするか手動で作成した場合、新しいオブジェクトはチェックインされるまで、Oracle BI Server によって認識されません。したがって、オブジェクトをチェックインするまで、メニューの「Update Row Count」は使用できません。

「Physical」レイヤーに行カウントを表示するには

1. 「Tools」→「Options」を選択します。
2. 「Options」ダイアログ・ボックスの「General」タブで、「Show row count in physical view」オプションを選択してから「OK」をクリックします。

「Physical」レイヤーにおいて選択した行カウントを更新するには

1. 「Physical」レイヤーで、1 つのテーブルまたは列を右クリックします。

   複数のオブジェクトを選択してから右クリックすることもできます。
2. ショートカット・メニューの「Update Rowcount」を選択します。

「Physical」レイヤーにおいてすべての行カウントを更新するには

1 「Tools」→「Update All Row Counts」を選択します。

リポジトリがオンライン・モードで開いている場合は、「Check Out Objects」ウィンドウが開くことがあります。

2 「はい」をクリックし、オブジェクトをチェックアウトします。

前回更新されてから変更されたすべての行カウントがリフレッシュされます。

論理レベル・カウントの自動移入

「Estimate Levels」により、Oracle BI 管理者は 1 つ以上のディメンション階層のレベル・カウントを自動的に移入できます。レベル・カウントは、クエリー・エンジンによって最適なクエリー計画の決定に利用され、システム全体のパフォーマンスが最適化されます。

リポジトリはオンライン・モードで開き、ビジネス・モデルがクエリーの発行先として使用可能である必要があります。次に、Oracle BI 管理者は、「Business Model and Mapping」レイヤーで次に示す論理レイヤー要素のいずれかを選択してから、「Estimate Levels」コマンドを実行します。

- ビジネス・モデル：クエリーの発行先として使用可能である必要があります。このオブジェクトを選択すると、Administration Tool では、ビジネス・モデルにおいてすべてのオブジェクトのチェックアウトが試行されます。
- ディメンション：Oracle BI 管理者は、ディメンションに対して一貫性チェックを実行し、ディメンションが論理的に問題ないことを確認する必要があります。
- ビジネス・モデルとディメンションの組合せ：Oracle BI 管理者は、複数のディメンションと複数のビジネス・モデルを個別に選択できます。

「Estimate Levels」コマンドを実行すると、次に示す説明に従って、レベル・カウントの一貫性チェックも実行されます。

- レベル・キーの有効性のチェック：レベル内の列には、参照整合性があります。
- 親子関係のチェック：親のレベル・カウントが子のレベル・カウントよりも大きい場合は、エラーが返されます。
- 実行レポート（評価されたすべてのカウントの一覧と、すべてのエラーや一貫性警告が記載）が生成されます。
- クエリーとエラーは、Oracle BI Server の NQQuery.log に記録されます。

注意： これらの情報をログ・ファイルに書き込むには、ログ・レベルを 4 以上に設定します。詳細は、「クエリー・ログの管理」（221 ページ）のログ・レベル設定に関する項を参照してください。

論理レベル・カウントを自動的に移入するには

1 Administration Tool で、リポジトリをオンライン・モードで開きます。

2 1 つ以上のビジネス・モデルとディメンション・オブジェクトを右クリックし、「Estimate Levels」を選択します。

3 「Check Out Objects」ダイアログ・ボックスで、「はい」をクリックして、リストに表示されたオブジェクトをチェックアウトします。

注意：「Estimate Levels」の実行には項目のチェックアウトが必要なため、「いいえ」をクリックした場合は操作が終了します。

「Oracle BI Administration Tool」ダイアログ・ボックスに、ディメンション・レベル・カウントと、すべてのエラー・メッセージや警告メッセージが表示されます。

オブジェクトをチェックインすると、リポジトリのグローバルな一貫性をチェックできます。

「Browse」ダイアログ・ボックスの使用

「Browse」ダイアログ・ボックスには、管理における様々な状況が表示されます。このダイアログ・ボックスを使用して、オブジェクトにナビゲートしてから選択します。

「Browse」ダイアログ・ボックスは、既存のオブジェクトから選択できる一連のダイアログ・ボックスからアクセスできます。

- 「Browse」ダイアログ・ボックスにある左ペインは、次の部分から構成されます。
  - リポジトリの「Presentation」レイヤー、「Business Model and Mapping」レイヤーおよび「Physical」レイヤーにおけるすべてのオブジェクトを階層表示するツリー。
  - レイヤーの選択ができるタブ（左ペインの下部）。「Browse」ダイアログ・ボックスからアクセスしたダイアログ・ボックスで操作可能なオブジェクトを含むレイヤーのタブのみが表示されます。
- 「Browse」ダイアログ・ボックスにある右ペインは、次の部分から構成されます。
  - 「Query」により、名前とタイプを指定して、リポジトリ内のオブジェクトにクエリーを発行できます。「Name」フィールドではアスタリスク（*）がワイルドカード文字として処理されるため、クエリーには部分一致を使用できます。
  - 「Show Qualified Names」チェック・ボックスにより、オブジェクトが所属する親を識別できます。
  - 「View」により、選択したオブジェクトのプロパティを読取り専用モードで表示できます。

「Browse」ダイアログ・ボックスでオブジェクトにクエリーを発行するには

1 「Type」ドロップダウン・リストから目的のオブジェクト・タイプを選択します。

2 「Name」フィールドに、オブジェクトの名前を入力するか、名前の一部とワイルドカード文字（*）を入力します。次の例を参照してください。
  - 名前が Q の文字で始まる論理テーブルを検索するには、「Type」ドロップダウン・リストから「Logical Tables」を選択し、「Name」フィールドに Q* と入力します。
  - 名前が dim の文字列で終わる論理テーブルを検索するには、「Name」フィールドに *dim と入力します。

3 「Query」ボタンをクリックします。

該当するオブジェクトが、クエリー結果リストに表示されます。

「Browse」ダイアログ・ボックスでオブジェクトを選択するには

■ 目的のオブジェクトを選択してから「Select」をクリックします。

「Browse」ダイアログ・ボックスが閉じ、オブジェクトが元のダイアログ・ボックスに表示されます。

クエリー結果リストのオブジェクトをツリー制御リストに同期化するには

1 「Query」リストでオブジェクトを選択します。

2 「Synchronize Contents」アイコンをクリックします。

3 Oracle BI リポジトリの計画と作成

この章の内容は次のとおりです。

Oracle BI リポジトリを計画し設定するためのロードマップ

ロードマップに関する項は、このマニュアルのいくつかの章において記載されています。Oracle BI リポジトリの計画と設定を行うために参照する項は次のとおりです。

- Oracle BI リポジトリの計画と設計のプロセス（42 ページ）
- 新しい Oracle BI リポジトリ・ファイルの作成（56 ページ）
- 第 4 章「Oracle BI リポジトリの「Physical」レイヤーの作成と管理」
- 第 5 章「Oracle BI リポジトリの「Business Model and Mapping」レイヤーの作成と管理」
- 第 6 章「Oracle BI リポジトリの「Presentation」レイヤーの作成とメンテナンス」
- 第 7 章「Oracle BI リポジトリ・ファイルの設定完了と管理」

Oracle BI リポジトリの計画と設計のプロセス

この項の内容は次のとおりです。

- リポジトリの計画と設計について（42 ページ）
- ビジネス・モデルの計画（43 ページ）
- ビジネス・モデルのデータベース・コンテンツの識別（48 ページ）
- リポジトリ設計のガイドライン（52 ページ）

リポジトリの計画と設計について

この項は、「Oracle BI リポジトリの計画と設計のプロセス」（42 ページ）の一部です。

リポジトリを作成したり、Oracle Business Intelligence に同梱されているリポジトリを変更する前に、ビジネス・モデルを計画してリポジトリを設計する必要があります。

Oracle BI リポジトリには、次のようなレイヤーがあります。

- リポジトリの「Physical」レイヤー：最初にこのレイヤー（スキーマ）を物理データベースのテーブルから作成します。

  警告：リポジトリの物理レイヤーを作成する前に、ビジネス要件、ビジネス・データ・モデルおよびユーザーを徹底的に分析して把握する必要があります。それによって、物理レイヤーを正しく設定できるようになります。詳細は、「ビジネス・モデルの計画」（43 ページ）を参照してください。

- リポジトリの「Business Model and Mapping」レイヤー：「Physical」レイヤーを設定した後、「Physical」レイヤーを「Business Model and Mapping」レイヤーにドラッグ・アンド・ドロップすると、「Business Model and Mapping」レイヤーを作成することができます。その際、「Physical」レイヤーで確立されていたマッピングは保持されます。

- リポジトリの「Presentation」レイヤー：「Business Model and Mapping」レイヤーを設定した後、「Business Model and Mapping」レイヤーを「Presentation」レイヤーにドラッグ・アンド・ドロップすると、「Presentation」レイヤーを作成することができます。エンド・ユーザーがアプリケーション（Oracle Siebel Marketing など）を介してアクセスするオブジェクトは、このレイヤーから表示されます。

カスタマイズでは、「Physical」レイヤー、「Business Model and Mapping」レイヤーおよび「Presentation」レイヤーを変更する作業が関連しています。これはデリケートな作業であるため、十分に注意して行う必要があります。Oracle BI リポジトリの設計とカスタマイズを行う場合は、この章に記載された情報を活用することをお薦めします。

ビジネス・モデルの計画

この項は、「Oracle BI リポジトリの計画と設計のプロセス」（42 ページ）の一部です。

ビジネス・モデルの計画は、意思決定支援のために使用可能なデータ・モデルを作成する最初の手順です。この項の計画ガイドラインに従うと、リポジトリの作成を開始できます。この項の内容は次のとおりです。

ビジネス・モデルの分析

最初の作業は、ビジネス・モデルを徹底的に把握することです。物理レイヤーのコンテンツを決める前に、作成するビジネス・モデルを把握する必要があります。

意思決定支援環境におけるデータ・モデリングの目的は、ビジネス・アナリストがビジネス構造を把握する作業に対応する形で、ビジネス情報を表現するモデルを設計することにあります。完成されたモデルでは、アナリストがビジネスについて質問する場合のように直感的にクエリーを作成することができ、クエリーのプロセスが自然なプロセスとなります。このモデルは、ビジネス・アナリストが容易に把握でき、しかも質問の主旨に対して的確に答えるモデルである必要があります。

ビジネス・モデルを分析するには、次の質問に自答していき、ビジネスをいくつかの構成要素に分解する必要があります。

- アナリストはどのようなビジネスの質問に対して回答しますか。
- ビジネスの業績を把握するには、どのようなメジャーが必要ですか。
- ビジネスはどのようなディメンションを軸に運営されていますか。
- 各ディメンションには階層的な要素はありますか。また、各階層はどのような親子関係によって定義されていますか。

このような質問に自答していくと、ビジネス・モデルの要素を識別して定義できるようになります。次の項では、ビジネスの構成要素を識別する方法について説明します。

ビジネス・モデルのコンテンツの識別

ビジネス・モデルを作成するには、データを論理的にビジネス・モデルにマッピングする必要があります。Oracle BI Server では、このような用途にディメンション・モデルを使用できます。この項では、代表的なディメンション・モデルの構成要素と種類について説明します。

ビジネスは、関連するディメンション条件によって分析され、ビジネス・モデルは、関連するディメンションから作成されます。そのため、ディメンション・モデルは、Oracle BI Server で使用する有効なビジネス・モデルの基盤となります。すべてのディメンション・モデルはスター・スキーマ上に構築されます。つまり、このモデルでは、様々なディメンション属性の観点から測定可能なファクトをモデリングすることができます。

ビジネス・モデルを分析した後、ビジネス・モデルに使用する論理テーブルと階層を識別する必要があります。これらのオブジェクトの詳細は、「ビジネス・モデルのコンテンツの識別」（43 ページ）を参照してください。この項の内容は次のとおりです。

ファクト・テーブルの識別

ファクト・テーブルは、メジャーを格納するテーブルです。メジャーは、論理ファクト・テーブルで定義する必要があります。メジャーまたはファクトとは、一般的に金額や販売数量のように計算対象となるデータを意味し、ディメンションによって指定できるものです。たとえば、特定の市場において特定の期間における特定製品の合計金額を求める場合があります。

メジャーにはすべて、SUM、AVG、MIN または MAX などの固有の集計ルールがあります。企業がメジャーの値を比較する場合は、比較を行うための計算式が必要になります。また、集計ルールは特定のディメンションに固有にすることもできます。Oracle BI Server では、ディメンションに固有で複雑な集計ルールを定義することができます。

Oracle BI Server では、リポジトリの「Business Model and Mapping」レイヤーにおけるテーブルに、そのテーブルを指している多対 1（N:1）の論理結合のみが存在する場合、通常はそのテーブルがファクト・テーブルとして認識されます。

たとえば、2 つのテーブルがあるとします。Sales Transactions がファクト・テーブルであり、Stores がディメンション・テーブルであるとします。ある 1 つの店舗で多数の販売トランザクションが発生し、別の店舗でトランザクションが発生しなかった場合（改装のため閉店中の場合）、それぞれ 1:n と 1:0 のリレーションシップが発生します。また、トランザクションがそれらの店舗で発生しなくても、会議では 0:n（0 対多）のリレーションシップが発生します。

注意：ファクト・テーブルは、0、1 対 N（0、1 対多関係）結合の最後の部分（N）に相当します。

45 ページの図 2 に、ファクト・テーブルに対して多対 1 の結合がある Business Model Diagram を示します。この図では、すべての結合にカラスの足跡のような記号があり、この記号が Fact-Pipeline テーブルを指しています（多い側を示している）。テーブルから外向きを指している結合はありません。この例のようなビジネス・モデルを表示するには、リポジトリを Administration Tool で開いてからファクト・テーブルを右クリックし、「Business Model Diagram」→「Whole Diagram」を選択します。

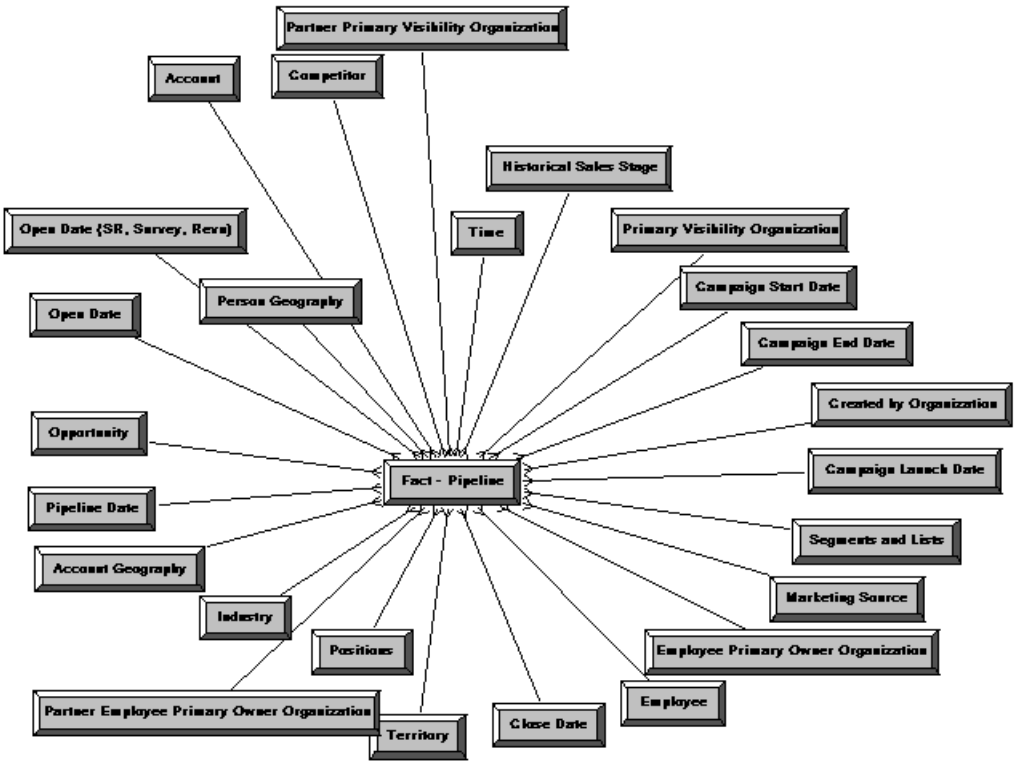


図 2.　ファクト・テーブルの結合図

注意：ブリッジ・テーブルは、この結合ルールの例外です。詳細は、「ブリッジ・テーブルの識別」（46 ページ）を参照してください。

ディメンション・テーブルの識別

企業が業績を測定する場合、明確に確立されたディメンション（期間別、製品別、市場別など）の観点からファクトを使用します。すべてのディメンションには、名称の付いた一連の属性があります。ディメンション・テーブルには、事業体を説明する属性が格納されます。たとえば、顧客名、地域、住所、国などの属性があります。また、ディメンション・テーブルには、各メンバーを識別する主キーも含まれます。

ディメンション・テーブルの属性により、数値データに条件が設定され、その条件によって、サービス・リクエストを分類することができます。たとえば、このディメンションに格納される属性には、サービス・リクエストの所有者、領域、アカウント、優先順位などがあります。

ディメンションとその属性を識別する最高の方法は、そのデータを使用する社内のアナリストから情報を集めることです。アナリストが使用して理解している用語を収集することが重要です。

ブリッジ・テーブルの識別

ディメンション・テーブルとファクト・テーブルとの間における多対多のリレーションシップをモデル化する必要がある場合、ファクト・テーブルとディメンション・テーブルとの間においてブリッジ・テーブルを作成することができます。ブリッジ・テーブルには、そのディメンションに対応する複数のレコードを格納します。Administration Tool で、「Logical Table」ダイアログ・ボックスの「General」タブをクリックすると、テーブルがブリッジ・テーブルであることを指定するチェック・ボックスが表示されます。

ブリッジ・テーブルでは、多対多のデータ・リレーションシップを解決することができます。たとえば、Employees テーブル（従業員用）と Jobs テーブル（職種用）があるとします。社内の従業員が、事務、緊急サポート要員、フロア・リーダーなどの複数の職種を兼任する場合があります。また、ある特定の職種を複数の従業員が担当し、社内に 10 人のフロア・リーダーがいる場合もあります。つまり、従業員と職種との間には、多対多のリレーションシップが存在します。しかし、1 つのテーブルにはすべての従業員が格納され、もう 1 つのテーブルには社内のすべての職種が格納されます。

この多対多のリレーションシップを記録して解決するには、Assignment というブリッジ（中間）テーブルを作成します。これにより、1 人の従業員に多数の担当業務を割り当てることができ、1 つの職種で多数の担当者が存在することができるようになります。この Assignment テーブルには、従業員と職種との組合せが記録されます。そして、主キーは従業員と職種との組合せになり、割当てという一意のレコードが作成されます。つまり、従業員対割当ては 1:n のリレーションシップになり、職種対割当ても 1:n のリレーションシップになります。Assignment テーブルが、Jobs テーブルと Employees テーブルとの間における橋渡し的なテーブルとして機能することで、従業員と職種の間における多対多のリレーションシップを解決することができます。その結果、たとえば、Mike と Gavin はどちらも緊急サポート要員とフロア・リーダーを兼任できるようになります。

ディメンション階層の識別

ユーザーのビジネスに必要な階層を把握することは、メタデータを用意するうえで重要です。このメタデータによって、Oracle BI Server は、特定のリクエストに対して、すでに計算されている集計で回答できるかどうかを判断できます。たとえば、月レベルをロールアップしたものが年レベルであり、月レベルの集計テーブルが存在する場合、月レベルのすべてのデータを年レベルに加算すると、そのテーブルで年レベルでの質問に回答することができます。

階層とは、ディメンション内において特定の属性間に存在する一連の親子関係です。階層の属性はレベルとも呼ばれ、子から親にロールアップします。たとえば、月は年にロールアップできます。このようなロールアップが、階層の要素において存在することにより、自然なビジネス・リレーションシップが形成されます。

ディメンションとは、ビジネスを定義する属性のカテゴリです。一般的なディメンションには、期間、製品、市場、顧客、サプライヤ、販促条件、原材料、製造工場、輸送手段、メディアのタイプ、時間などがあります。1 つのディメンションの中には、多数の属性が含まれる場合があります。たとえば、期間ディメンションには、日、週、月、四半期、年などの属性が含まれる場合があります。ディメンションにどのような属性が含まれるかは、ビジネスの分析方法によって異なります。

ディメンションの階層は、複数の属性間で 1 対多関係を表します。47 ページの図 3 に、時間ディメンションとその階層のサンプルを示します。

図 3.　階層のサンプル

このディメンションのサンプルでは、日は週に集計またはロールアップできます。月は四半期にロールアップでき、また四半期は年にロールアップできます。1 つの属性が別の属性にロールアップする場合、それは 1 対多関係を表します。

これらの階層定義は、ビジネス・モデルに固有のものである必要があります。つまり、あるモデルでは週が年にロールアップし、別のモデルでは週が年にロールアップしないよう設定する場合があります。たとえば、週が年にロールアップするモデルでは、すべての週を確実に 1 年に対応させることができますが、1 つの週が 2 つの年度にまたがるカレンダでは、1 年に対応させることができません。

月と年の組合せが 1 四半期にロールアップすると、階層によっては複数の要素でロールアップが必要な場合があります。Oracle BI Server では、どのように複雑であっても、特定のビジネスの階層を定義して、ビジネスの定義に従った分析を行うことができます。

できるだけ自然な階層を多数識別する必要があります。ビジネスの質問と同様に、明確な階層もあれば、明確でない階層もあり、特定のビジネスを担当するエンド・ユーザーのみ把握している階層もあります。したがって、Administration Tool を使用して階層を正しく定義するには、階層を十分に把握している必要があります。

スター・モデルとスノーフレーク・モデルについて

スター・モデルとスノーフレーク・モデルは、ディメンション・ルールの 1 対多関係に基づいています。スター・スキーマには、論理ディメンション・テーブルと論理ファクト・テーブルとの間において、1 対多関係があります。スノーフレーク・スキーマには、同じタイプのリレーションシップがありますが、ディメンションの複数の要素間において 1 対多関係もあります。例として、48 ページの図 4 と 48 ページの図 5 では、Sales Facts と Facts はファクト・テーブルを示し、Markets、Periods、Products、Account Hierarchy、Account Region Hierarchy および Account はディメンション・テーブルを示します。ファクト・テーブルでもブリッジ・テーブルでもない論理テーブルはディメンション・テーブルです。詳細は、「ブリッジ・テーブルの識別」（46 ページ）を参照してください。

48 ページの図 4 に、スター・スキーマを示します。

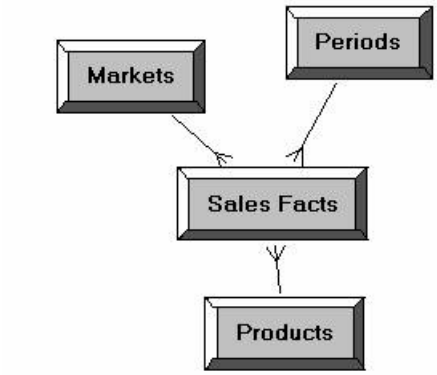


図 4.　スター・スキーマの図

注意：スノーフレーク・スキーマの使用は最小限に抑えることをお薦めします。

スノーフレーク・スキーマでは、サーバーは、論理テーブル・ソース内にある外部結合に対して、他の外部結合とは異なる処理を行います。論理テーブル・ソース内において、結合は必ず実行されます。結合が複数の論理テーブル間にあるときは、必要なときにのみ実行されます。スノーフレーク・スキーマの詳細は、「物理スキーマのタイプについて」（49 ページ）を参照してください。48 ページの図 5 に、論理スノーフレーク・スキーマを示します。

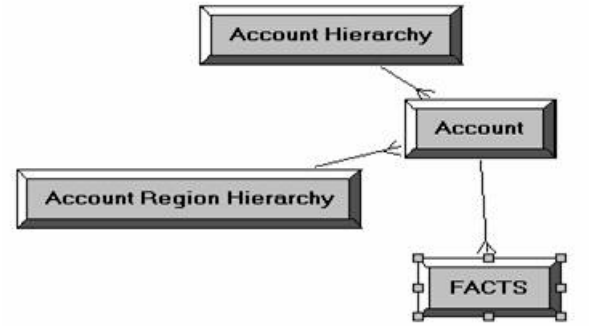


図 5.　スノーフレーク・スキーマの図

ビジネス・モデルのデータベース・コンテンツの識別

この項は、「Oracle BI リポジトリの計画と設計のプロセス」（42 ページ）の一部です。

Oracle BI Server は、Oracle BI リポジトリを、基礎となる物理データベースにマッピングするためのインタフェースを備えています。これによって、基礎となるデータベースの物理的な設計を行うこともできます。また、データベースがすでに存在している場合は、そのコンテンツを操作する必要があります。いずれの場合も、物理データベースの構造とコンテンツを把握している必要があります。

この項の内容は次のとおりです。

物理スキーマのタイプについて

物理スキーマ（モデル）には、E-R スキーマとディメンション・スキーマの 2 種類があります。E-R スキーマは、データ格納における冗長性を最小限に抑え、データの更新を最適化できるように設計されています。ディメンション・スキーマは、理解しやすさを重視し、クエリーのパフォーマンスを最適化できるように設計されています。

- **E-R スキーマ**：E-R スキーマは、完全に正規化された標準的なリレーショナル・スキーマであり、多くのオンライン・トランザクション処理（OLTP）システムで使用されています。複数のテーブル間のリレーションシップは、複数のデータ間のリレーションシップを表します。必ずしもビジネスのリレーションシップを表しているわけではありません。

  通常、E-R スキーマには多数のテーブルがあり、その数は数百個や数千個に及ぶ場合があります。多数のテーブルが存在するのは、データの冗長性をなくして更新のパフォーマンスを向上させる目的のために、データが細かく分解（データベースの用語では正規化）されているためです。E-R スキーマは、OLTP データベースにとっては非常に効率的です。E-R データベースに対してクエリーを行う場合は通常、結合を事前に決定しておき、最適化することができます。E-R データベースは通常、どこでなにをたずねたらよいかを認識しているアプリケーションによってクエリーが行われます。また、これらのアプリケーションから 1 回のクエリーで取得する情報も通常、顧客レコード、注文レコードまたは出荷レコードなどの小さい単位の情報になります。

  E-R スキーマは通常、履歴分析を行うクエリーには向いていません。その理由は、パフォーマンスが低く、SQL で質問を作成するのが難しいという 2 つの大きな問題があるためです。

  - E-R 履歴分析クエリーが実行された場合にパフォーマンスが低下する問題は解消されません。なぜなら、データベースが過去のデータにさかのぼる必要があり、処理に時間がかかる複雑なプロセスになるからです。また、データベース管理システムのコストベース・オプティマイザは、そのような複雑なクエリーに対処できるように設計されておらず、作成されるクエリー計画はパフォーマンスが低くなります。
  - データに詳しいデータベース・アナリスト（DBA）であれば、ビジネスの質問に理論的に回答できる SQL クエリーを E-R スキーマに対して作成できる場合がありますが、そのような詳しいデータ知識は一般的に、エンドユーザーのビジネス・アナリストの担当範囲外にあります。SQL を適切に作成できても、その結果をユーザーに返すまでに非常に長いレスポンス時間がかかる場合が多いです。

- **ディメンション・スキーマ**：ディメンション・スキーマは、ビジネス・モデルに従い、正規化されていないスキーマです。ディメンション・スキーマには、ディメンション・テーブルとファクト・テーブルが含まれます。ディメンション・テーブルには、ビジネスの属性が含まれ、ファクト・テーブルには、いくつかのファクトが格納された個別のレコードと、各ディメンション・テーブルに対する外部キーが含まれます。

  ディメンション・スキーマは、ビジネス分析に使用され、意思決定支援に関しては、E-R スキーマとは異なる次の 2 つの大きな利点があります。

  - 良好なクエリー・パフォーマンス

- 理解しやすさ

ディメンション・スキーマは、個別のレコードを更新する場合は、E-R スキーマほど効率的ではありませんが、複数のディメンションにわたってビジネスを分析するクエリーには効率的です。

ディメンション・スキーマには、次の 2 種類があります。

- **スター・スキーマ**：スター・スキーマは、1 つのファクト・テーブルが複数のディメンション・テーブルと外部キーのリレーションシップを持ったディメンション・スキーマです。
  - ディメンション・テーブルは、ビジネス・モデルが反映され、サンプルの Products ディメンションに示した Product、Size、Color のような名前の属性の列があります。また、ディメンション・テーブルには、テーブル内の各行を一意に識別するキー列が 1 つ以上あります。
  - ファクト・テーブルにはマルチパート主キーがあり、通常ディメンション・テーブルに対する外部キー参照から構成されます。また、ファクト・テーブルにはすべてのメジャーまたはファクトが含まれ、ビジネスの業績を測定するために使用されます。格納ディテール・レベルにおいて最も小さいレベルは、ファクト・テーブルの粒度です。それより大きいレベルの集計情報は、ディテール・レベルのレコードから計算されるか事前に計算され、別の集計ファクト・テーブルに格納されて、複数のスター・スキーマが作成されます。集計ファクト・テーブルの詳細は、「ファクト・テーブルの識別」（44 ページ）を参照してください。
- **スノーフレーク・スキーマ**：スノーフレーク・スキーマは、1 つ以上のディメンションが、ある程度正規化されたディメンション・スキーマです。

  注意：スノーフレーク・スキーマの使用は最小限に抑えることをお薦めします。

  スノーフレーク・スキーマと E-R スキーマの各正規化のタイプにおける違いは、スノーフレークの正規化がビジネス階層の属性に基づいている点にあります。ディメンションからスノーフレーク・スキーマとして構成された複数のテーブルでは、ディメンション階層が反映された親子関係があります。スノーフレーク・スキーマでは、複数の論理テーブルが 1 つの論理テーブルとみなされます。

  50 ページの図 6 に、スノーフレーク・スキーマの例を示します。ここでは、Product Line と Products がスノーフレークのブランチになっています。

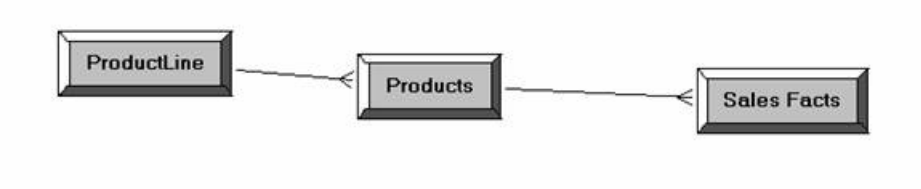


図 6.　スノーフレーク・スキーマの図

Products ブランチには、1 つのディメンション階層を作成する必要があります。この階層に作成する必要がある最小単位のレベルは次のとおりです。

- 総計レベル
- ファクト・テーブルに結合されないディメンションのディテール・レベル（この例では、ProductLine）

- ファクト・テーブルに結合されるディメンションのディテール・レベル

51 ページの図 7 に、作成する必要がある階層を示します。

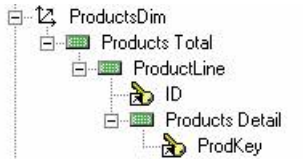


図 7. 「Business Model and Mapping」レイヤーにおける「Products」ブランチの階層

主キーと外部キー間のリレーションシップについて

物理データベースの構造とコンテンツを十分に把握するには、主キーと外部キーのリレーションシップの概念を理解することが重要です。

主キーと外部キーのリレーションシップでは、リレーショナル・データベースにおける 2 つのテーブルの間で 1 対多関係が定義されます。外部キーとは、あるテーブルにおいて別のテーブルにある主キー列を参照する 1 つ以上の列のことです。主キーは、一意の値によってテーブルにおいて 1 つの行を識別する列（または一連の列）として定義されます。

たとえば、51 ページの図 8 のようなテーブルがあるとします。上部にあるテーブルは Sales というファクト・テーブルで、下側にあるテーブルは Date というディメンション・テーブルです。Sales ファクト・テーブルには、各販売トランザクションに対応する行が 1 行あり、Date ディメンション・テーブルには、データベースでカバーする日付に対応する行が 1 行あります。

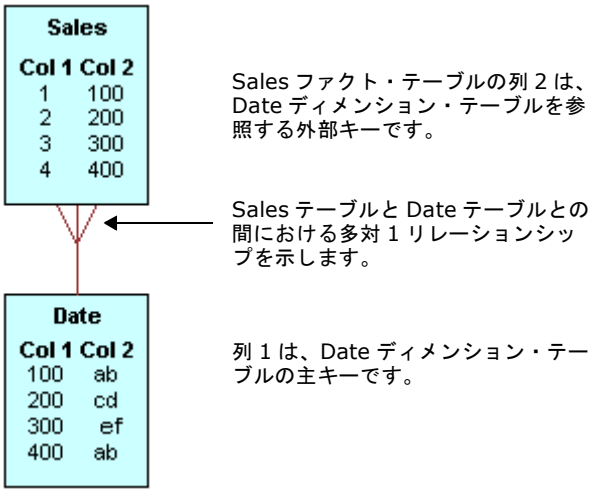


図 8. 主キーと外部キーのサンプル

主キーと外部キーのこのリレーションシップによって、Sales テーブルと Date テーブルを結合して、Date テーブルの他の属性と Sales テーブルのレコードを組み合せることができます。たとえば、アナリストが製品販売時点における合計販売高、曜日、製品名および店舗を問い合せた場合、Sales テーブルと様々なディメンション・テーブルとの間における主キーと外部キーのリレーションシップを介して、販売高、日付、製品および店舗のテーブルを結合し、必要な情報を編集することができます。

インポートするデータベース・テーブルの構造の識別

Administration Tool では、論理テーブルを、データベースの基礎となる物理テーブルにマッピングするためのインタフェースを備えています。テーブルをマッピングする前に、ビジネス・モデルに関連付ける物理データベースのコンテンツを識別する必要があります。この作業を適切に行うには、次のように物理データベースのコンテンツを識別する必要があります。

- 各テーブルのコンテンツを識別します。
- 各テーブルのディテール・レベルを識別します。
- 各集計テーブルのテーブル定義を識別します。これにより、集計ナビゲーションを設定できます。Oracle BI Server には、次のディテールが必要です。
  - テーブルをグループ化する列（集計レベル）
  - 集計のタイプ（SUM、AVG、MIN、MAX または COUNT）

  集計ナビゲーションをリポジトリで設定する方法の詳細は、「Oracle BI リポジトリでの集計ナビゲーション用のフラグメンテーション・コンテンツの設定」（205 ページ）を参照してください。
- 各列のコンテンツを識別します。
- 各メジャーが計算される方法を識別します。
- データベースで定義された結合を識別します。

前述のようなデータに関する情報を集めるには、データベースの実装時に作成されるデータ要素について説明しているドキュメントを参照するか、データベースごとに DBA から情報を収集する必要が発生する場合があります。また、すべてのデータ要素について完全に把握するには、データを利用する社内ユーザー、データの所有者、またはデータを作成するアプリケーションのアプリケーション開発者にそれぞれ問い合せることも必要になる場合があります。

リポジトリ設計のガイドライン

この項は、「Oracle BI リポジトリの計画と設計のプロセス」（42 ページ）の一部です。

ビジネス・モデルのニーズを分析し、ビジネスに必要なデータベースのコンテンツを識別すると、リポジトリの設計に着手することができます。この項では、効率的なリポジトリを実装するための、リポジトリの設計方法のベスト・プラクティスについて説明します。

通常は、データベースをインポートしてテストするまで、パフォーマンスのチューニング変更を行う必要はありません。この作業は、リポジトリの設定が完了する最後の手順で行います。これらの最後の手順の詳細は、「Oracle BI リポジトリ・ファイルの設定完了と管理」（155 ページ）を参照してください。

この項の内容は次のとおりです。

リポジトリを操作する場合の一般的なヒント

Oracle BI Server によってメタデータがリポジトリに格納されます。Oracle BI 管理者は、Administration Tool ソフトウェアのグラフィカル・ユーザー・インタフェース（GUI）を使用して、リポジトリを作成して管理します。Oracle BI リポジトリは、3 つのレイヤーで構成されます。各レイヤーは、Administration Tool のユーザー・インタフェースにおいて個別のペインに表示され、ツリー構造になります（Windows Explorer と同様）。これらのレイヤーは、エンド・ユーザーには表示されません。

ほとんどのウィンドウとダイアログ・ボックスには、作業に役立つオンライン・ヘルプが付いています。ヘルプ・トピックにアクセスするには、ダイアログ・ボックスのヘルプ・ボタンをクリックするか、メニューから「Help」を選択します。

Oracle BI リポジトリ構造を設計する場合は、次の方法をお薦めします。

- リポジトリ編集作業の大半をオフライン・モードで実行すると、結果において一貫性が保たれます。これにより、環境によっては開発作業時間が短縮する場合があります。

  オンライン・モードで作業する場合は、一定の作業を開始する前と完了した後は必ず、オンライン・リポジトリのバックアップを保存します。必要に応じて、「File」メニューの「Copy As」を使用して、変更内容が含まれたオフライン・コピーを作成します。

- 設計に独立性を持たせる場合は、ビジネス・モデルごとに個別にデータベースをインポートします。たとえば、アプリケーションごとにデータベースを 1 つ使用する場合があります。複数のテーブルが複数のデータベースに存在する場合は、異なるビジネス・モデルごとにそれらのテーブルをカスタマイズできます。これにより、異なるデータベース間で結合を行う必要がなくなります。

- データベースをインポートした後は、そのデータベースをテストしてから、別のデータベースをインポートします。パフォーマンスのチューニングを行う前に、メタデータから適切なレコード・セットが生成されることを確認します。

- Administration Tool の物理ダイアグラムを使用して、ソースと結合を確認します。

「Physical」レイヤー（スキーマ）を設計する場合のヒント

「Physical」レイヤーには、物理データソースに関する情報が含まれています。スキーマを「Physical」レイヤーに作成する最も一般的な方法は、データベースや他のデータソースからメタデータをインポートする方法です。メタデータをインポートした場合、プロパティの多くは、インポート処理中に収集された情報に基づいて自動的に構成されます。また、物理データソースのメタデータに存在しなかった結合リレーションシップなど、物理データソースの他の属性を定義することもできます。

「Physical」レイヤーには、データベースや XML ドキュメントなどのデータソースを 1 つ以上配置することができます。サポートされるデータベースの詳細は、『Oracle Business Intelligence Suite Enterprise Edition システム要件およびサポートされるプラットフォーム』を参照してください。

データソースごとに、対応する接続プールが 1 つ以上あります。接続プールには、データソースとの接続に使用されるデータソース名（DSN）、最大接続数、タイムアウト情報およびそれ以外の接続に関連した管理情報が含まれています。詳細は、「接続プールの設定」（69 ページ）と第 4 章「Oracle BI リポジトリの「Physical」レイヤーの作成と管理」を参照してください。

「Physical」レイヤーを設計する場合は、次のヒントを参考にしてください。

- データ・ウェアハウスのメタデータをリポジトリの「Physical」レイヤーにインポートする前に、データ・ウェアハウス内の外部結合をすべて削除します。この場合、Extract Transform Load（ETL）検索を使用して検索します。外部結合を削除すると、レコード・セットの一貫性が維持されてビジネス・モデルが単純になり、パフォーマンスが向上します。
- 物理結合をインポートすることによって発生する問題に対処するには、物理データを外部キーなしでインポートしてから、必要に応じてキーを作成します。
- 今すぐに使わないが削除しないテーブルを、「Physical」レイヤーにインポートする場合があります。「Business Model and Mapping」レイヤーですぐに使用するテーブルを識別する方法として、ビジネス・モデルのレイヤーにマッピングする前に、物理テーブルに別名を割り当てる方法があります。

  注意：別名を割り当てたテーブルの名前を表示するには、「Tools」→「Options」→「General」をクリックし、「Display original names for alias in diagrams」を選択します。
- ネーミング規則を「Physical」レイヤーに適用し、次の例に示すように、論理コンポーネントの後に物理テーブルの名前を付けます。

  Created By Employee (W_PERSON_D)

  注意：「Physical」レイヤーのオブジェクトの名前は、物理的な SQL 文に含まれます。修飾された「Physical」レイヤー名を使用すると、デバッグが容易になります。
- 不明瞭ビュー（SELECT 文で構成される物理レイヤー・テーブル）は、他の方法がない場合にのみ使用する必要があります。理想的には、物理テーブルを作成するか、マテリアライズド・ビューを作成する必要があります。従来のデータベース・ビューは、不明瞭ビューと同じであるため、必要ありません。詳細は、「不明瞭ビューのデプロイ」（107 ページ）を参照してください。

「Business Model and Mapping」レイヤーを設計する場合のヒント

「Business Model and Mapping」レイヤーでは、ビジネス・モデルにより情報が構成されます。すべてのビジネス・モデルには、論理テーブルが含まれます。論理テーブルは、論理列で構成されます。複数の論理テーブルの間には、相互に論理結合を持つリレーションシップがあります。複数の論理列間におけるリレーションシップは、ビジネス・モデルの階層における定義に従った階層構造にできます。論理テーブルは、「Physical」レイヤーのソース・データにマッピングされます。マッピングには、複雑な変換や計算式を含めることができます。

「Business Model and Mapping」レイヤーは、各物理ソースの意味とコンテンツをビジネス・モデル用語で定義します。Oracle BI Server ではこれらの定義を使用して、各データ・リクエストに適切なソースを割り当てます。

物理オブジェクトの名前、対応するビジネス・モデル・オブジェクトの名前およびプロパティとは個別に変更することができます。基礎となる物理データベースを変更しても、ビジネス・モデルのテーブルと物理テーブルとの間におけるマッピングを変更しても、リポジトリの表示やクエリーを実行するエンド・ユーザー・アプリケーションのビューは変更されない場合があります。

各ビジネス・モデルで定義された論理スキーマには、複数の論理テーブルが含まれる必要があります。すべての論理テーブルの間には、リレーションシップを定義する必要があります。ビジネス・モデル・スキーマの詳細は、「Oracle BI リポジトリの計画と設計のプロセス」（42 ページ）を参照してください。「Business Model and Mapping」レイヤーの設定方法の詳細は、第 5 章「Oracle BI リポジトリの「Business Model and Mapping」レイヤーの作成と管理」を参照してください。

「Business Model and Mapping」レイヤーを設計する場合は、次のヒントを参考にしてください。

- ビジネス・モデルを作成する場合は、できるだけ論理ディメンション・テーブルとファクト・テーブルとの間において 1 対多の複合結合を使用します。ビジネス・モデルは、理想的には、各ファクト・テーブルが複数のディメンションに直接リンクされる単純なスター・スキーマに近づける必要があります。

  注意：スノーフレーク・モデルを使用すると、柔軟性の高い設計を行うことができる場合があります（外部結合を使用できるなど）。ただし、多数の列がプレゼンテーション・レイヤーに作成される場合があります。スノーフレーク・スキーマの使用は最小限に抑えることをお薦めします。

- 外部結合は、レポート作成 SQL では使用しないでください。外部結合は、ETL の様々な方法によって排除できます。外部結合を排除すると、別のテーブルをレポート SQL から削除できるだけでなく、パフォーマンスも向上します。

- すべての類似ディメンション属性は、1 つの論理ディメンション・テーブルにまとめます。必要に応じて、別名を「Physical」レイヤーのテーブルで使用して、他のディメンション・テーブルのデータを、主要なディメンション・ソースに入れます。これらの作業を ETL プロセスで行うと、パフォーマンスが最適化されます。

- 各論理ディメンション・テーブルには、対応したディメンション階層が必要です。集計コンテンツを使用して、すべてのファクト・ソースが、その階層において適切なレベルにリンクされることを確認します。集計コンテンツは、「Logical Table Source」プロパティ・ウィンドウの「Content」タブで設定します。

- 集計ソースは、論理テーブルに別のソースとして作成する必要があります。ファクト集計の場合は、「Logical Table Source」プロパティ・ウィンドウの「Content」タブを使用して、適切な論理レベルを各ディメンションに割り当てます。

- 「Business Model」レイヤーではできるだけテーブルの別名を使用して、次のように余分な結合を削除することをお薦めします。

  - 別名を使用して、複数のディメンション間におけるすべての物理結合（複数のディメンション間の循環結合）を削除します。

  - 物理テーブルの別名を作成して、物理モデル内の論理テーブル・ソースにおけるすべての循環結合（ディメンション内の循環結合）を削除します。

- 「Business Model and Mapping」レイヤーにおいて要素の名前を変更すると、自動的に別名が作成されます。

- 集計ナビゲーションに関する問題を防止するには、ディメンション階層の各論理レベルで、「Number of elements at this level」というフィールドに適切な値が設定されていることを確認します。ファクト・ソースは、選択されたフィールドの組合せと、リンク先ディメンションにおけるレベルの組合せに基づいて選択されます。これらの値を調整すると、Oracle Business Intelligence によって選択されたファクト・ソースを変更することができます。

- テーブル・ソースが使用されなくても、論理テーブル・ソースの外部結合は常にクエリーに含まれます。可能な場合は、外部結合がない 1 つの論理テーブル・ソースと、外部結合がある別の論理テーブル・ソースを作成します。外部結合のある論理テーブル・ソースは、外部結合のない論理テーブル・ソースの後に配置すると、必要な場合にのみ使用されるようになります。

「Presentation」レイヤーを設計する場合のヒント

「Presentation」レイヤーでは、ビジネス・モデルのユーザー・ビューを設定します。「Presentation」レイヤーにおけるフォルダと列の名前は、ローカライズされた言語に翻訳されて表示されます。「Presentation」レイヤーは、ユーザー権限の設定対象となるレイヤーです。このレイヤーでは、次の操作を実行できます。

- 「Business Model and Mapping」レイヤーに存在する列よりも少ない列を表示できます。たとえば、キー列は、ビジネス的な意味がないため、除外することができます。
- 「Business Model and Mapping」レイヤーにおけるテーブル構造と異なる構造を使用して列を編成できます。
- 「Business Model and Mapping」レイヤーにおける列名と異なる列名を表示できます。
- 各カタログ、テーブルおよび列に対するユーザーのアクセスを許可または拒否する権限を設定できます。
- 論理キーを ODBC ベースのクエリーとレポート・ツールにエクスポートできます。

「Presentation」レイヤーの設定方法の詳細は、第 6 章「Oracle BI リポジトリの「Presentation」レイヤーの作成とメンテナンス」を参照してください。

「Presentation」レイヤーを設計する場合は、次のヒントを参考にしてください。

- **列の別名**：「Presentation」レイヤーの列には別名を使用しないことをお薦めします。

  注意：「Presentation」レイヤーにおいて要素の名前を変更すると、自動的に別名が作成されます。これにより、レポートが元の要素を参照できるようになります。

- **単一テーブル・モデル**：できるだけ単純にするために、単一のテーブルで構成されるサブジェクト領域を作成できます。単一テーブル・モデルを作成するには、最初に論理ディメンション・モデルを作成してから、Administration Tool の「Presentation」レイヤーで単一のテーブルのスキーマとして表示します。論理ディメンション・モデルでは、Oracle BI Server が適切な物理テーブルにナビゲートするために必要なメタデータを設定する必要があります。「Presentation」レイヤーの詳細は、「Oracle BI リポジトリの「Presentation」レイヤーの作成とメンテナンス」（147 ページ）を参照してください。

新しい Oracle BI リポジトリ・ファイルの作成

計画と設計作業が完了すると、リポジトリを作成できます。リポジトリを作成する最初の手順は、リポジトリ・ファイルを作成することです。リポジトリを保存するたびに、ダイアログ・ボックスが表示され、グローバルな一貫性をチェックするかどうかの確認が行われます。オプションには次のものがあります。

- **はい**：グローバルな一貫性をチェックしてから、リポジトリ・ファイルを保存します。
- **いいえ**：グローバルな一貫性をチェックせずに、リポジトリ・ファイルを保存します。
- **取消**：グローバルな一貫性をチェックせず、リポジトリ・ファイルを保存しません。

詳細は、「リポジトリまたはビジネス・モデルの一貫性のチェック」（28 ページ）を参照してください。

注意：オフラインで編集する際には、リポジトリを随時保存してください。ビジネス・モデルに一貫性がない場合でも、リポジトリはオフライン・モードで保存できます。

新しいリポジトリ・ファイルを作成するには

1. Administration Tool のメニューから「File」→「New」を選択するか、ツールバーの「New」ボタンをクリックします。
2. リポジトリの名前を入力します。

   ファイルの拡張子を明示的に指定しない場合は、RPD ファイルの拡張子が自動的に付加されます。すべてのリポジトリのデフォルトの場所は、Oracle BI のソフトウェア・インストール・フォルダにある Repository サブディレクトリ（Oracle BI¥server¥Repository）です。

   新しいリポジトリは空です。リポジトリ設定プロセスにおける残りの手順では、「Oracle BI リポジトリを計画し設定するためのロードマップ」（42 ページ）に記載されているように、リポジトリの設定と移入を行う手順について説明します。

リポジトリ・ファイルの作成後

リポジトリ・ファイルを Administration Tool で作成すると、テーブルを「Physical」レイヤーにインポートすることができます。

注意：一般的には、「Physical」レイヤー、「Business Model and Mapping」レイヤー、「Presentation」レイヤーの順序で作成します。ただし、リポジトリを作成するどの段階でも、レイヤーを個別に作成することができます。セキュリティの設定は、リポジトリのテストを開始するときに行うことができます。

詳細は、「Oracle BI リポジトリの「Physical」レイヤーの作成と管理」（59 ページ）を参照してください。

4 Oracle BI リポジトリの「Physical」レイヤーの作成と管理

この章は、リポジトリを計画および設定するためのロードマップの一部です。詳細は、「Oracle BI リポジトリの計画と作成」（41 ページ）を参照してください。

リポジトリの物理レイヤーを作成する前に、リポジトリ・ファイルを計画、設計および作成する必要があります。詳細は、「Oracle BI リポジトリの計画と設計のプロセス」（42 ページ）を参照してください。

Administration Tool の「Physical」レイヤーでは、Oracle BI Server がクエリーを発行するデータソース、およびデータソースへの複数のクエリーの処理に使用する物理データベースと他のデータソース間の関係を定義します（使用しているライセンスのオプションにより、データベースと接続プールを物理レイヤーに追加できるかどうかが決まります）。

物理レイヤーを作成する最初の手順は、スキーマを作成することです。サポートされているデータソース・タイプの物理スキーマをインポートできます。既存のデータソースから、スキーマまたはその一部をインポートできます。物理レイヤーは手動で作成することもできますが、これは面倒でエラーが発生しやすい作業です。

警告：物理スキーマをインポートすることを強くお薦めします。

この章は、Administration Tool を使用してリポジトリの物理レイヤーを作成する際に役立つ次の項で構成されています。

リレーショナル・データソースから物理レイヤーを作成するプロセス

物理スキーマをインポートすると、物理レイヤーの構造がインポートされるので、時間と手間を節約できます。サポートされているデータソースのスキーマをインポートできます。スキーマをインポートしない場合は、テーブル、主キー、外部キーなど、生成するクエリーに対するオブジェクトを作成する必要があります。これらのソースにあるデータは、Oracle BI の Interactive Dashboard に表示できます。

物理スキーマをインポートする場合は、次のヒントを参考にしてください。

- ほとんどのデータベースでは、スキーマをインポートすると、テーブル、主キーおよび外部キーがデフォルトでインポートされます。

  注意：数多くのテーブルをインポートするとプロセスの所要時間が長くなるため、データベースから外部キーをインポートしないことをお薦めします。

- 物理テーブルをインポートするときには、作成するビジネス・モデルで使用する可能性の高いデータが含まれたテーブルのみをインポートするようにしてください。フィルタ（テーブル・マスク）を使用して、インポート・リストに表示するテーブルの数を制限できます。これにより、インポートするテーブルを容易に検索および選択できるようになります。
- また、データベース・ビュー、別名、シノニムおよびシステム・テーブルもインポートできます。こうしたオブジェクトは、Oracle BI Server でそれらのオブジェクトに対するクエリーを生成する場合にかぎりインポートしてください。
- 無関係のテーブルなどのオブジェクトを数多くインポートすると、過度に複雑になり、リポジトリのサイズが増大します。

注意：データ・ウェアハウス（OLAP）オブジェクトは、Oracle アナリティック・ワークスペース（AW）に格納され、そこを通じてアクセスできます。これらのオブジェクトを抽出する方法の詳細は、Web サイト（oracle.com）で、10*g* リリース 2 に関する Oracle 関連ドキュメントを参照してください。

スキーマをインポートして物理レイヤーを作成するには、ここに示す順番で次の手順を実行してください。

- リレーショナル・データソースからの物理スキーマのインポート（60 ページ）
- データベース・オブジェクトの設定（64 ページ）
- 接続プールの設定（69 ページ）
- 物理テーブルについて（85 ページ）
- 物理テーブルの作成および設定（86 ページ）

リレーショナル・データソースからの物理スキーマのインポート

この項は、「リレーショナル・データソースから物理レイヤーを作成するプロセス」（60 ページ）の一部です。

サポートされているデータソース・タイプの物理スキーマをインポートできます。インポートを使用して物理スキーマを作成する場合は、次のいずれかのインポート・オプションおよび適切な接続タイプを選択する必要があります。

- **From a database**: オフライン・モードとオンライン・モードで使用できます。このオプションは、すべてのデータベース接続が自分のマシンで設定されている場合に使用します。「Import」オプションで、次の接続タイプを使用できます。
  - ほとんどの場合、物理スキーマのインポートは ODBC 接続タイプを使用して実行されます。
  - スキーマをインポートするためのネイティブのデータベース・ゲートウェイは、接続タイプが DB2（DB2 CLI または DB2 CLI Unicode を使用）と XML の場合のみサポートされます。Oracle BI Server の XML ゲートウェイを使用した XML データのインポートの詳細は、「Oracle BI Server XML Gateway の例」（348 ページ）を参照してください。
  - テーブル・マスクを使用して、「Import」ダイアログ・ボックスに表示するテーブルのリストを制限（フィルタリング）できます。具体的なテーブルを 1 つ以上インポートする場合は、テーブル・マスクを使用してテーブルを検索できます。
- **Through the Oracle BI Server**: オンライン・モードで使用できます。このオプションは、Oracle BI Server の接続を使用してスキーマをインポートする場合に使用します。このオプションでは、Oracle BI の管理者が、Oracle BI Server マシンのデータソース名（DSN）を使用して物理スキーマ情報をインポートできます。接続ソフトウェアと複製 DSN は、Oracle BI の管理者のマシンおよび Oracle BI Server マシンに常駐させておく必要はありません。「Import」オプションで、次の接続タイプを使用できます。
  - 使用可能な接続タイプは、ODBC、DB2 CLI、DB2 CLI（Unicode）および XML です。
  - UNIX プラットフォームで実行している場合、Oracle BI Server では ODBC 接続タイプを使用してスキーマをインポートすることはできません。
- **From a repository**: オフライン・モードで使用できます。デフォルトでは、「File」メニューの「Import from repository」オプションは無効で、このオプションは将来サポートされる予定はありません。この方法でメタデータをインポートする必要がある場合は、「「Options」ダイアログ・ボックスにおける「General」タブの使用」（31 ページ）を参照してください。

  注意：プロジェクトは、インポート対象のオブジェクトを含むリポジトリで作成してから、リポジトリ・マージを使用して現行リポジトリにインポートすることをお薦めします。詳細は、「プロジェクトへのメタデータ・サブセットの抽出について」（171 ページ）を参照してください。
- **From XMLA**: マルチディメンショナル・データソースのインポートに使用します。詳細は、「マルチディメンショナル・データソースからの物理スキーマのインポート」（63 ページ）を参照してください。

ODBC 接続タイプから物理スキーマをインポートするには

1. Administration Tool で「File」→「Import」を選択し、次のオプションから物理スキーマのソース・タイプを選択します。
   - from Database
   - through Server
   - from Repository
2. 「Select Data Source」ダイアログ・ボックスで次の手順を実行します。
   a. 「Connection Type」ドロップダウン・リストから接続タイプを選択します。

   注意：日本語のテーブル名や列名など国際的な文字を含むスキーマをインポートする場合は、ODBC 3.5 または DB2 CLI Unicode を使用することをお薦めします。

b DSN リストで、スキーマのインポート元であるデータソースを選択します。

Oracle BI Server を通じてインポートすると、DSN エントリはローカルのマシンではなく Oracle BI Server に置かれます。

c カスタマ・リレーションシップ・マネジメント（CRM）テーブルからインポートするには、「Read from CRM metadata tables」チェック・ボックスを選択します。

d 「OK」をクリックします。

3 ログオン画面が表示されたら、データソースに対して有効なユーザー ID とパスワードを入力してから、「OK」をクリックします。

Oracle BI の管理者が、データソースのユーザー名とパスワードを用意する必要があります。Oracle BI の管理者のマシンでインポートを実行している場合、そのマシンの DSN 構成からユーザー名が取得されます。

4 「Import」ダイアログ・ボックスで、インポートするオブジェクトのタイプに対応したチェック・ボックスを選択します。「Table」、「Keys」、「Foreign Keys」などがこれに該当します。

一部のオブジェクトのチェック・ボックスは、自動的に選択されています。デフォルトで選択されるものは、データソースによって異なります。

5 テーブル・マスクを使用するには、F% などのテーブル・マスクを入力し、最上位のデータベース・フォルダをクリックします。

フィルタ条件に合致するテーブルが表示されます。

6 インポートするテーブルおよび列を選択します。

注意：列を特定しないでテーブルのみを選択すると、そのテーブルの列がすべてインポートされます。

7 インポートするオブジェクトをすべて選択したら、「Import」をクリックするか、「Physical」レイヤーにドラッグ・アンド・ドロップします。

マルチディメンショナル・データソースから物理レイヤーを作成するプロセス

この項では、Administration Tool を使用してマルチディメンショナル・データソースをリポジトリの物理レイヤーに追加する方法について説明します。マルチディメンショナル・データソースを使用する機能により、Microsoft Analysis Services や SAP/BW（SAP/Business Warehouse）などのソースに Oracle BI Server を接続して、データを抽出できます。これらのソースにあるデータは、Oracle BI の Interactive Dashboard に表示できます。

Oracle BI Server は、XML for Analysis（XMLA）規格のプロトコルを使用して、マルチディメンショナル・ソースに接続します。このとき、接続先のデータソースで Web サービスのインタフェース機能がすべて使用可能であることが必要です。この規格では、Oracle BI Server が接続先データとクエリー・データに接続するための様々なプロトコルを定めています。

マルチディメンショナル・ソースからデータをインポートすると、Oracle BI リポジトリにデータソースのメタデータが作成されます。マルチディメンショナル・データソースとリレーショナル・データソースの設定の大きな違いは物理レイヤーにあります。ビジネス・モデルとプレゼンテーション・レイヤーの設定についてはほぼ同じです。

マルチディメンショナル・データソースの物理レイヤーを作成するには、ここに示す順番で次の手順を実行してください。

マルチディメンショナル・データソースからの物理スキーマのインポート

この項は、「マルチディメンショナル・データソースから物理レイヤーを作成するプロセス」（62 ページ）の一部です。

警告：マルチディメンショナル・データソースから物理スキーマを手動で作成するのは、面倒でエラーが発生しやすい作業です。このため、物理スキーマをインポートすることを強くお薦めします。

Oracle BI Server では、XMLA 規格を使用して、マルチディメンショナル・データソースから Oracle BI リポジトリにデータをインポートします。インポート・プロセスの際、マルチディメンショナル・データソースの各キューブが、1 つの物理キューブ・テーブルとして作成されます。Oracle BI Server では、メトリック、ディメンション、階層などのキューブがインポートされます。キューブをインポートした後で、物理キューブの列に正しい集計ルールが割り当てられていることと、デフォルトのメンバー・タイプである ALL が階層に正しくインポートされていることを確認する必要があります。詳細は、「物理キューブ・テーブルへの階層の追加」（95 ページ）を参照してください。

データをインポートする前に、次の点を検討する必要があります。

- 現在 XMLA 互換のデータソースとしてサポートされているのは、Microsoft Analysis Services と SAP/BW のみです。
- Oracle BI Server では、Oracle BI でサポートしているディメンションと階層のみをインポートします。したがって、不規則な階層や親子階層が含まれているキューブはインポートされません。また、メジャー階層もインポートされず、サポートされていません。
- 階層と列は、ビジネス・モデルで使用しない場合には、物理レイヤーから削除しておくことをお薦めします。これにより、不要なオブジェクトを Administration Tool でメンテナンスする必要がなくなり、パフォーマンスが向上する可能性があります。

マルチディメンショナル・データソースから物理スキーマをインポートするには

1. データソース管理者から、URL 接続文字列と、データソースのユーザー名およびパスワードを取得します。
2. Administration Tool で、「File」→「Import from XMLA」を選択します。
3. 「Import From XMLA」ダイアログ・ボックスで、次のフィールドに値を入力します。
   a. 「URL」フィールドに、XMLA プロバイダとなる Web サービスの URL を入力します。

      データソース管理者から取得した URL 接続文字列を使用します。

b ユーザー名とパスワードのフィールドに、データソースで有効なユーザー名とパスワードを入力します。

データソース管理者から取得したユーザー名とパスワードを使用します。

4 「OK」をクリックします。

Administration Tool が接続先ソースに接続し、使用可能なデータ・カタログ（データベース）のリストが取得されます。

5 「Browse」ダイアログ・ボックスで、データソースとカタログ（データベース）を開き、必要に応じて、インポートするカタログ（データベース）またはキューブを選択します。

「Import」ボタンは、ダウンロード可能なオブジェクトを選択するまで使用できません。

6 インポートするオブジェクトをすべて選択したら、「Import」ボタンをクリックします。

インポートできないオブジェクトがあった場合は、「Oracle BI Administration Tool」ダイアログ・ボックスに、警告メッセージのリストが表示されます。次に例を示します。

```
Hierarchy "[Measures]" from data source "SDCHS40I023", catalog
"XMLA_council_interop", cube "BeverageSales" is of measure type and not imported.
```

7 「Oracle BI Administration Tool」ダイアログ・ボックスでは、次の操作を実行できます。

- 具体的な用語を検索するには、「Find」をクリックし、さらに「Find Again」をクリックします。
- ウィンドウの内容をコピーして別のファイルにメッセージを貼り付けるには、「Copy」をクリックします。

8 インポートが完了したら、各ダイアログ・ボックスで「OK」をクリックしてから「閉じる」をクリックします。

データベース・オブジェクトの設定

この項は、「リレーショナル・データソースから物理レイヤーを作成するプロセス」（60 ページ）および「マルチディメンショナル・データソースから物理レイヤーを作成するプロセス」（62 ページ）の一部です。

スキーマをインポートすると、そのスキーマのデータベース・オブジェクトが自動的に作成されますが、データベースのプロパティはユーザーが設定する必要があります。

サポートされるデータベースの詳細は、『Oracle Business Intelligence Suite Enterprise Edition システム要件およびサポートされるプラットフォーム』を参照してください。

物理レイヤーでデータベースを作成または編集するには、次の作業を実行します。

- 物理レイヤーでのデータベース型について（65 ページ）
- 物理レイヤーでのデータベース・オブジェクトの手動作成（65 ページ）
- データベースでサポートされている SQL 機能の指定（68 ページ）

物理レイヤーでのデータベース型について

物理スキーマを「Physical」レイヤーにインポートすると、通常はデータベース型が自動的に割り当てられます。データベース型の自動割当てに関する追加情報を次に示します。

- **リレーショナル・データソース**：インポート・プロセスの際、ODBC ドライバによっては Oracle BI Server にデータベース型が割り当てられることがあります。ただし、サーバーでデータベース型が判断できない場合、近似の ODBC タイプがデータベース・オブジェクトに割り当てられます。この ODBC タイプは、データベース型のドロップダウン・メニューに含まれる最も近いエントリに置き換えられます。
- **マルチディメンショナル・データソース**：現在 XMLA 互換のデータソースとしてサポートされているのは、Microsoft Analysis Services と SAP/BW のみです。マルチディメンショナル・データソースをインポートするときには、適切なデータベース型とバージョンを選択する必要があります。

物理レイヤーでのデータベース・オブジェクトの手動作成

データベース・オブジェクトを手動で作成した場合は、接続プール、テーブル、列など、すべてのデータベース要素を手動で設定する必要があります。

注意：マルチディメンショナル・データソースでは、リポジトリの物理レイヤーに物理スキーマを作成した場合、データソース内の同じカタログ（データベース）に属するキューブまたはキューブ・セットごとに、物理レイヤーにデータベースを 1 つ作成する必要があります。1 つの物理データベースには複数のキューブを含めることができます。ただし、それらのキューブはすべて、データソース内の同じカタログに含まれている必要があります。「Connection Pool」ダイアログ・ボックスでカタログを指定します。

警告：物理スキーマをインポートすることを強くお薦めします。

データベース・オブジェクトを作成するには

1. Administration Tool の「Physical」レイヤーで右クリックして、「New Database」を選択します。

   右クリックするときにオブジェクトが選択されていないことを確認してください。

2. 66 ページの表 5 を参照し、「Database」ダイアログ・ボックスの「General」タブで、フィールドに値を入力します。

表 5. 「Database」ダイアログ・ボックスの「General」タブのフィールド

フィールド	説明
Allow direct database requests by default（チェック・ボックス）	選択すると、すべてのユーザーが物理クエリーを実行できるようになります。Oracle BI Server では、ユーザーが入力した未処理の物理 SQL が、基礎となるデータベースに直接送られます。返された結果セットを Oracle BI Presentation Services でレンダリングしてからグラフ化し、ダッシュボードにレンダリングして Oracle BI のリクエストとして扱えます。 すべてではないが大半のユーザーが物理クエリーを実行できるようにするには、このオプションを選択してから、特定のユーザーやグループに対してクエリーを制限します。クエリーの制限の詳細は、「クエリーの実行権限の管理」（339 ページ）を参照してください。 **警告：**正しく構成しないと、機密性の高いデータが予想外のユーザーに見られてしまう可能性があります。詳細は、「デフォルトで直接データベース・リクエストを許可する場合の推奨事項」（67 ページ）を参照してください。 物理 SQL の実行の詳細は、『Oracle Business Intelligence Answers, Delivers, and Interactive Dashboards ユーザーズ・ガイド』を参照してください。
Allow populate queries by default（チェック・ボックス）	選択すると、すべてのユーザーが POPULATE SQL を実行できるようになります。すべてではないが大半のユーザーが POPULATE SQL を実行できるようにするには、このオプションを選択してから、特定のユーザーやグループに対してクエリーを制限します。クエリーの制限の詳細は、「クエリーの実行権限の管理」（339 ページ）を参照してください。
Data Source Definition CRM Metadata Tables	マルチディメンショナル・データソースでは使用できません。選択すると、インポート・ユーティリティによって、Oracle Siebel CRM 固有のテーブルからテーブル定義が検索されます。Siebel CRM のテーブルをインポートすると Siebel メタデータ・ディクショナリが読み取られ、物理テーブルと列が定義されます（データベースのデータ・ディクショナリは読み取られません）。これは、ソースが従来の Siebel Systems の場合のみです。
Data Source Definition Database	使用しているデータベースのデータベース型です。 「Features」タブを使用して、指定したデータベース型でサポートされている SQL 機能を検証する方法の詳細は、「データベースでサポートされている SQL 機能の指定」（68 ページ）を参照してください。

表 5.　「Database」ダイアログ・ボックスの「General」タブのフィールド

フィールド	説明
Data Source Definition Virtual Private Database	物理データベース・ソースを仮想プライベート・データベース（VPD）として識別します。VPD を使用すると、返されるデータ結果はユーザー認証資格証明によって異なります。したがって、これらのソースの識別が重要になります。これらのデータ結果は、キャッシュとともに使用されるクエリー結果セットの妥当性に影響します。詳細は、「Oracle BI Server のクエリー・キャッシュ」（237 ページ）を参照してください。 **注意：**このチェック・ボックスを選択した場合は、「Session Variable」ダイアログ・ボックスの「Security Sensitive」チェック・ボックスも選択する必要があります。詳細は、「セッション変数の作成」（297 ページ）を参照してください。
Persist Connection Pool	永続接続プールを使用するには、まず、一時テーブルを設定する必要があります。詳細は、「永続接続プールのプロパティの設定」（83 ページ）を参照してください。

デフォルトで直接データベース・リクエストを許可する場合の推奨事項

このプロパティを設定すると、すべてのユーザーが物理クエリーを実行できるようになります。正しく構成しないと、機密性の高いデータが予想外のユーザーに見られてしまう可能性があります。このデータベースのプロパティを設定する場合は、次の推奨ガイドラインに従ってください。

- Oracle BI Server では、Oracle BI Server、Oracle BI Presentation Services または Oracle BI Scheduler Server を実行しているマシンからのみ接続リクエストを受け入れるように構成してください。この制限は、Oracle BI Presentation Services の IP アドレスを使用して TCP/IP レベルで設定する必要があります。これにより、Oracle BI Presentation Services の IP アドレスからの TCP/IP 接続のみが許可されます。
- ユーザーがこのマシンにリモートでログインして nQCmd.exe（SQL スクリプト実行ファイル）を実行するのを防ぐには、次の方法によって、Oracle BI Presentation Services をインストールしたマシンにアクセスできないようにする必要があります。
  - TELNET
  - リモート・シェル
  - リモート・デスクトップ
  - 電話会議用のソフトウェア（Windows NetMeeting など）

  必要に応じて、Administrator 権限を持つユーザーを例外にできます。
- Administrator 権限を持つユーザーにかぎり、次の作業を許可する必要があります。
  - Oracle BI Server と Oracle BI Presentation Services のマシンに TELNET し、nQCmd.exe の実行などの作業を実行してキャッシュがシードされるようにします。詳細は、「SQL スクリプト・ファイルの実行による集計の作成と削除」（256 ページ）を参照してください。
- Answers の拡張 SQL ページにアクセスして、レポートを作成します。詳細は、『Oracle Business Intelligence Presentation Services 管理ガイド』を参照してください。

- Oracle BI Presentation Services でグループまたはユーザーに基づく権限を設定し、直接データベース・リクエストを編集（Oracle BI Presentation Services の管理者によるアクセスが許可されるように事前構成されている場合）または実行（誰からのアクセスも許可しないように事前構成されている場合）する機能へのアクセスを制御します。詳細は、『Oracle Business Intelligence Presentation Services 管理ガイド』を参照してください。

データベースでサポートされている SQL 機能の指定

スキーマをインポートするか、「Database」ダイアログ・ボックスの「General」タブでデータベース型を指定すると、「Feature」テーブルには、データベース型に適したデフォルト値が自動的に移入されます。これらは、このデータソースとともに Oracle BI Server が使用する SQL 機能です。

データベースに対するデフォルトのクエリー機能を調整できます。たとえば、左側外部結合クエリーをサポートしているデータソースにおいて、Oracle BI Server がそのようなクエリーを特定のデータベースに送信することを禁止する場合は、「Feature」テーブルでデフォルトの設定を変更できます。

警告：「Feature」テーブルの変更は慎重に行ってください。データベースでサポートされていない SQL 機能を有効化した場合、クエリーでエラーが返され、予期しない結果となることがあります。サポートされている SQL 機能を無効にした場合、サーバーからデータベースに対して発行される SQL の効率が低下します。

データベースでサポートされている SQL 機能を指定するには

1. Administration Tool の「Physical」レイヤーで、データベースをダブルクリックします。
2. 「Database」ダイアログ・ボックスで「Features」タブをクリックします。
3. 「Features」タブで各 SQL 機能のプロパティを指定する方法の詳細は、68 ページの表 6 を参照してください。

表 6.　SQL 機能のタブの説明

フィールドまたはボタン	説明
Ask DBMS	「Feature」テーブルのないデータベースをインストールおよびクエリーしている場合にかぎり使用するボタン。「Feature」テーブルのエントリでこのデータベースをクエリーできます。詳細は、「「Ask DBMS」を使用した「Feature」テーブルのエントリの変更」（69 ページ）を参照してください。
Default	デフォルトの SQL 機能を特定するチェック・ボックス。データソースのデータベース型でサポートされているデフォルトの SQL 機能が、自動的に選択されています。
Find	リストで機能を検索する際に使用する文字列を入力するためのボタン。
Find Again	「Find」をクリックした後で使用可能になるボタン。同一文字列に対して複数の検索を実行できます。

表 6.　SQL 機能のタブの説明

フィールドまたはボタン	説明
Revert to Defaults	デフォルト値に戻すボタン。
Value	追加の SQL 機能を指定できるチェック・ボックス。選択するとクエリー・タイプが有効になり、選択解除すると無効になります。デフォルトの SQL 機能は無効にしないことを強くお薦めします。

「Ask DBMS」を使用した「Feature」テーブルのエントリの変更

「Ask DBMS」ボタンは、「Feature」テーブルのないデータベースをインストールおよびクエリーしている場合にかぎり使用してください。

注意： XML データソースまたはマルチディメンショナル・データソースを使用している場合、「Ask DBMS」ボタンは使用できません。

「Database」ダイアログ・ボックスの「Features」タブにある「Ask DBMS」ボタンを使用すると、サポートしている SQL 機能でデータベースをクエリーできます。クエリー結果に基づいて、「Feature」テーブルに表示されたエントリを変更できます。

警告：「Ask DBMS」の使用は慎重に行ってください。機能クエリーの結果は、データソースで実際にサポートされている SQL 機能を正確に反映したものとはかぎりません。この機能は、Oracle テクニカル・サポートの助言があった場合にかぎり使用してください。

接続プールの設定

この項は、「リレーショナル・データソースから物理レイヤーを作成するプロセス」（60 ページ）および「マルチディメンショナル・データソースから物理レイヤーを作成するプロセス」（62 ページ）の一部です。

接続プールは、「Physical」レイヤーのオブジェクトであり、データソースへのアクセスを記述するものです。ここには、Oracle BI Server とそのデータソースの接続に関する情報が含まれています。

Administration Tool の「Physical」レイヤーには、各データベースに 1 つ以上の接続が含まれています。データソースのスキーマをインポートして物理レイヤーを作成すると、接続プールが自動的に作成されます。1 つのデータベースに対して複数の接続プールを構成できます。接続プールを使用すると、複数の同時データソース・リクエスト（クエリー）で 1 つのデータベース接続を共有できるため、データベース接続のオーバーヘッドを軽減できます。

注意： 初期化ブロックには専用の接続プールを作成することをお薦めします。詳細は、「接続プールの作成または変更」（72 ページ）を参照してください。

各接続プールに対し、同時接続を許可する最大数を指定する必要があります。Oracle BI Server では、この制限に達すると、他の接続がすべて別の接続プールにルーティングされます。他の接続プールが存在しない場合は、接続が可能になるまで接続リクエストが待機状態になります。

同時接続の許容数を増やすと、接続プールがアクセスしている基礎データベースの負荷が高くなる可能性があります。テストを行い、データベース管理者に問い合せて、接続プールで指定した接続数をデータソースで処理できるかどうかを確認してください。また、接続数に基づくチャージ・バック・システムがデータソースに備わっている場合には、同時接続数を制限してチャージ・バックのコストを抑えることもできます。

Oracle BI Server では、データベース・リソースに関連する潜在的な負荷とコストに加えて、サーバー起動時にも各接続に共有メモリーが割り当てられています。このため、接続数が増加し、Oracle BI Server のメモリー使用量が多くなります。

70 ページの表 7 では、事前構成された接続プールについて、手短に説明します。

表 7.　事前構成された接続プール

接続名	説明
BBB Data Warehouse	Oracle Business Analytics Warehouse で会社全体のデータを表示できます。 Oracle Siebel 業務系アプリケーション（Oracle BI の顧客のみ）。
BBB OLTP	Oracle Siebel トランザクション・データベースで会社全体のデータを表示できます。 Oracle Siebel 業務系アプリケーション（Oracle BI の顧客のみ）。
BBB XLS Data	DSN は接続プール名と同じです。データは、 C:¥Data¥Applications¥BBB¥BBB に格納されます。 Oracle Siebel 業務系アプリケーション（Oracle BI の顧客のみ）。
ERM OLTP	Oracle Siebel Workforce Analytics を Oracle Siebel トランザクション・データベースに接続します。 （Workforce Oracle BI の顧客のみ）
Externalized Metadata Strings	Oracle Siebel 業務系データベース・アプリケーションに接続して、メタデータ文字列の翻訳をロードします。 Financial Services の顧客、および英語以外の言語で Oracle BI をデプロイしているすべての顧客が対象です。 **注意：**この接続プールは、Oracle Siebel トランザクション・データベースと同じように構成されています。
Forecasting Oracle Business Analytics Warehouse	Oracle Business Analytics Warehouse に接続します。 （Forecasting Oracle BI の顧客のみ）
Forecasting Siebel OLTP	Oracle Siebel トランザクション・データベースに接続します。 （リアルタイムの Forecasting Oracle BI の顧客のみ）
Incentive Compensation Siebel OLTP	Oracle Siebel トランザクション・データベースにデータベース接続します。 （Incentive Compensation Oracle BI の顧客のみ）
Pharma Data Warehouse	Pharmaceutical データ・ウェアハウスに接続します。 （製薬業界に特化した顧客のみ）

表 7.　事前構成された接続プール

接続名	説明
Real-time OLTP	Oracle Siebel トランザクション・データベースに接続します。 リアルタイムの Oracle BI（すべての顧客）。
SIA Data Warehouse	Oracle Siebel Industry Applications データ・ウェアハウスに接続します。 （Oracle Siebel Industry Applications の顧客のみ）
Oracle BI Usage	Usage Tracking Writer 接続プールでは、Oracle BI Server の使用状況統計を格納しているデータベースに接続します（すべての顧客に対してオプション対応）。 **注意：** これを使用するには、使用状況統計がデータベースにロードされるように Oracle BI Scheduler Server を設定する必要があります。
Oracle Business Intelligence Warehouse	Oracle Business Analytics Warehouse にデータベース接続します。 （Oracle BI アプリケーションの顧客のみ）
Siebel OLTP	**注意：** Oracle Siebel トランザクション・データベースへの接続プールは 2 つあります。両方を適切に構成する必要があります。 OLTP DbAuth では、Oracle Siebel トランザクション・データベースに接続して認証および認可を行います。ユーザー名とパスワードは USER および PASSWORD として事前構成されます。データベース・ログオンによってユーザーを認証する場合は、そのままにしておく必要があります。 （すべての顧客）
UQ Siebel OLTP	Oracle Siebel トランザクション・データベースに接続します。Oracle Siebel Universal（Queuing の顧客のみ）。
Usage Accelerator Datawarehouse	Oracle Business Analytics Warehouse にデータベース接続します。 （Oracle BI アプリケーションの顧客のみ）

この項の内容は次のとおりです。

接続プールの作成または変更

接続プールを作成する前に、データベース・オブジェクトを作成する必要があります。データベース・オブジェクトと接続プールは通常、物理スキーマをインポートすると自動的に作成されます。Administration Tool の「Physical」レイヤーで、接続プールを作成または変更します。

警告：Oracle への接続には OCI を使用することを強くお薦めします。ODBC は、Oracle からインポートする場合にのみ使用してください。

この項の内容は次のとおりです。

初期化ブロックの接続プールについて

初期化ブロックには専用の接続プールを作成することをお薦めします。この接続プールは、クエリーには使用しないでください。

さらに、初期化ブロックのタイプに応じて接続プールを切り離すことをお薦めします。こうしておくと、認証およびログイン固有の初期化ブロックで、ログイン・プロセスの速度が低下することはなくなります。次のタイプの初期化ブロックでは、接続プールを分ける必要があります。

- 認証およびログイン固有のすべての初期化ブロック（言語、外部文字列、グループの割当てなど）。
- セッション変数を設定するすべての初期化ブロック。
- リポジトリ変数を設定するすべての初期化ブロック。これらの初期化ブロックは、常にシステム管理者のユーザー・ログインを使用して実行する必要があります。

  これらの初期化ブロックの数、スケジュール済のリフレッシュ率、および実行スケジュールに注意してください。一般的に、このシナリオがリソースに影響を与えることはごくまれです。たとえば、リフレッシュ率を分単位で設定した場合、同時にリフレッシュする初期化ブロックが 15 を超える場合、およびこの両方の場合、主要ユーザーがタイム・フレームにアクセスしている間に、リソースへの影響が発生する可能性があります。使用可能なリソースへの負担を避けるには、Oracle BI のデフォルトの管理者へのクエリー・ロギングを無効にしておくこともできます。

初期化ブロックは、Oracle BI Server の変数の最大数を各ブロックに割り当てることができるように設計してください。たとえば、変数が 5 つある場合、5 つの変数すべてを含む初期化ブロックを 1 つ構成すると、効率を高め、リソースへの負担を少なくできます。使用する初期化ブロックが 1 つの場合、その初期化文字列を使用して、1 つのコールによってバックエンドのテーブルで値を解決できます。5 つの初期化ブロック、つまり各変数に対して初期化ブロックを 1 つずつ作成した場合、バックエンド・テーブルの割当てに 5 つのコールが必要となります。

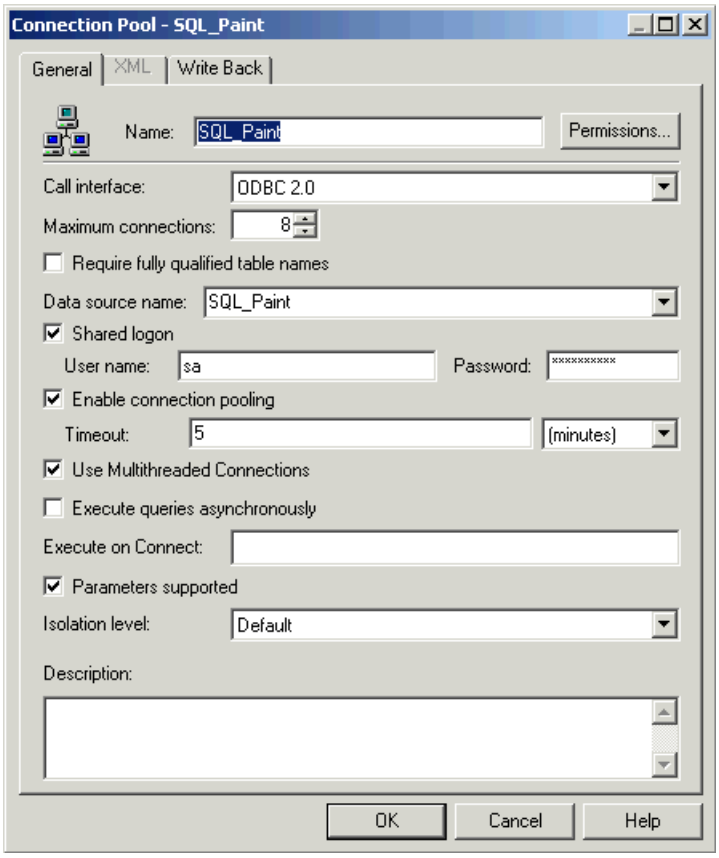
接続プールの一般プロパティの設定

「General」タブで値を入力するには、この項を参照してください。

接続プールの一般プロパティを設定するには

1 Administration Tool の「Physical」レイヤーでデータベースを右クリックして、「New Object」→「Connection Pool」を選択するか、既存の接続プールをダブルクリックします。

「Connection Pool」ダイアログ・ボックスの「General」タブは、次のようになっています。



2 「Connection Pool」ダイアログ・ボックスで「General」タブをクリックし、74 ページの表 8 の記述に従ってフィールドに値を入力します。

マルチディメンショナル・データソースに固有のプロパティを、79 ページの表 9 に示します。

表 8. 接続プールの一般プロパティ

フィールドまたはボタン	説明
Call interface	データソースへのアクセスに使用するアプリケーション・プログラム・インタフェース（API）です。ネイティブ API を使用してアクセスするデータベース、ODBC を使用してアクセスするデータベース、両方の方法で動作するデータベースがあります。コール・インタフェースが XML の場合は「XML」タブを使用できますが、XML データソースに適用されないオプションは使用できません。
Data source name	ドロップダウン・リストに、システムで構成されたユーザーおよびシステム DSN が表示されます。接続するデータベースにアクセスできるように構成するデータソース名です。データソース名には、データソースで有効なログオン情報を含める必要があります。この情報が無効な場合、DSN で指定したデータベース・ログオンが失敗します。
Enable connection pooling	将来のクエリー・リクエストに備えて、指定した時間にわたって、1 つのデータベース接続を開いた状態にしておくことができます。接続プーリングを使用すると、クエリーのたびに新しい接続を開閉するオーバーヘッドを節約できます。このオプションを選択しない場合、データベースにクエリーが送られるたびに新しい接続が開きます。
Execute on Connect	データベースへの接続が確立するたびに、Oracle BI の管理者が、実行するコマンドを指定できるようになります。データベースで許可される任意のコマンドを指定できます。たとえば、引用識別子をオンにする場合にも使用できます。メインフレーム環境では、DB2 に接続したときのセカンダリ認証 ID を設定して、セキュリティを強制的に終了し、RACF などのメインフレームのセキュリティ・パッケージに戻す場合に使用できます。これによって、メインフレーム環境のセキュリティを 1 箇所でメンテナンスできます。
Execute queries asynchronously	データソースで非同期クエリーをサポートするかどうかを指定します。

表 8.　接続プールの一般プロパティ

フィールドまたはボタン	説明
Isolation level	この値では、ODBC ゲートウェイと DB2 ゲートウェイで、バックエンド・データベースの各接続に対するトランザクション分離レベルを設定します。分離レベルを設定すると、接続により発行されるすべての文に対するデフォルトのトランザクション・ロッキング動作を制御できます。一度に設定できるオプションは 1 つのみです。その接続に対する設定は、明示的に変更されるまで保持されます。 次に、オプションを示します。 **Committed Read:** データの読取り時に内容を保証しない読取りとならないように、共有ロックを指定します。ただし、データはトランザクションの終了前に変更できます。これによって、リピータブル・リードや擬似データが発生しなくなります。 **Dirty Read:** 内容を保証しない読取りを実装します（分離レベル 0 のロッキング）。このオプションを設定すると、コミットされていないデータや内容を保証しないデータを読み取ったり、データの値を変更したり、トランザクションの終了前にデータセットの行を表示または非表示にすることができます。これは、最も制限の緩やかな分離レベルです。 **Repeatable Read:** クエリーで使用しているデータをすべてロックし、他のユーザーがデータを更新できないようにします。ただし、別のユーザーが新しい擬似行をデータセットに挿入できます。その場合、その行は現在のトランザクションの以降の読取りに含まれます。 **Serializable:** データセットの一定範囲をロックし、トランザクションが完了するまで、他のユーザーが行の更新やデータセットへの挿入をできないようにします。これは、4 つの中で最も制限の厳しい分離レベルです。同時実行性が低下するため、このオプションは必要な場合にのみ使用してください。
Maximum connections	この接続プールで許可される最大接続数です。デフォルトは 10 です。この値は、データベースの作成とモデル、データベースを実行しているハードウェア・ボックスの構成、およびアクセスを必要とする同時ユーザー数によって判断する必要があります。 **注意：** Intelligence Dashboard のページでデプロイする場合には、同時ユーザー数をダッシュボードのリクエスト数で乗算した値の 10 ～ 20% 程度で検討してください。この数値は、使用状況に応じて調整できます。リポジトリのすべての接続の総数は 800 接続未満にしてください。ある初期化ブロック専用の接続プールで必要な最大接続数は、初期化ブロックの実行中に同時にログオンしているユーザー数から試算することもできます。
Name	接続プールの名前です。名前を入力しない場合は、Administration Tool によって名前が生成されます。マルチディメンショナル・データソースおよび XML データソースの場合は事前入力されています。

表 8.　接続プールの一般プロパティ

フィールドまたはボタン	説明
Parameters supported	データベース機能テーブルがパラメータをサポートしている場合、接続プールのパラメータをサポートするプロパティのチェック・ボックスが選択解除されていると、Oracle BI Server がフィルタ（または計算）をデータベースにプッシュすることを可能にする特殊コードが、パラメータを使用して実行されます。Oracle BI Server では、SQLPrepare の付加的なコールをデータベースに送信することによって、ゲートウェイとアダプタのレイヤー内のパラメータ・サポートをシミュレーションするという方法でこれを実行します。
Permissions	接続プールにアクセスするための権限を、個々のユーザーまたはグループに割り当てます。また、権限のあるユーザー・グループを設定して、独自の接続プールを持たせることもできます。
Require fully qualified table names	データベースで必要であれば、このチェック・ボックスを選択します。 このオプションを選択すると、接続プールから送信されたリクエストがすべて、完全修飾名を使用して、基礎となるデータベースをクエリーします。完全修飾名は、リポジトリの物理オブジェクト名に基づいています。物理レイヤーのメタデータのインポート元と同じテーブルをクエリーした場合、問題なくこのオプションを選択できます。ある物理データベースから、データベース名とスキーマ名の異なる別のデータベースにリポジトリを移行した場合、移行後のデータベースでは完全修飾名は無効になります。この場合、このオプションを選択しないと、新しいデータベース・オブジェクトに対してクエリーが継承されます。 データソースによっては、完全修飾名を使用すると、目的のデータベースの目的のテーブルに確実にクエリーを送ることができるので安全です。たとえば、RDBMS がマスター・データベースの概念をサポートしていれば、*foo* という名前のテーブルに対するクエリーは、最初にマスター・データベース、次に指定したデータベースで目的のテーブルを検索します。foo という名前のテーブルがマスター・データベースに存在すると、そのテーブルがクエリーされ、指定したデータベースで *foo* という名前のテーブルはクエリーされません。
Shared logon	「Shared logon」チェック・ボックスを選択するのは、接続プールを使用したクエリーを実行するユーザーがすべて、同じユーザー名およびパスワードを使用して基礎となるデータベースにアクセスできるようにする場合です。 このオプションを選択すると、データベースに対して接続プールを使用するデータベース接続はすべて、接続プールで指定したユーザー名およびパスワードを使用します。これは、そのユーザーが、DSN（またはユーザー構成）でデータベースのユーザー名およびパスワードを指定した場合でも同様です。 このオプションを選択しないと、接続プールを経由する接続は、DSN またはユーザー・プロファイルで指定したデータベースのユーザー ID およびパスワードを使用します。

表 8.　接続プールの一般プロパティ

フィールドまたはボタン	説明
Timeout (Minutes)	リクエスト完了後にデータソースへの接続を維持する時間を、分単位で指定します。その時間内であれば、新しいリクエストは、新しい接続を開かずにその接続を使用します（最大接続数に指定した数値に達するまで）。この時間は、接続リクエストが完了するたびにリセットされます。 タイムアウトを 0 に設定すると、接続プーリングが無効になります。つまり、リクエストが完了するとすぐに、データソースへの接続が切断されます。新しい接続は、別の接続プールを使用するか新しい接続を開きます。
Use Multithreaded Connections	このチェック・ボックスを選択すると、Oracle BI Server は、アイドル状態の物理クエリー（スレッド）を終了します。選択を解除すると、1 つのスレッドが 1 つのデータベース接続に関連付けられます（スレッド数 = 最大接続数）。スレッドは、アイドル状態であってもメモリーを消費します。 NQSConfig.ini の Server セクションにある DB_GATEWAY_THREAD_RANGE パラメータは、Oracle BI Server がアイドル状態のスレッドを終了すると確立されます。下限は、Oracle BI Server が操作を実行するまで開かれていたスレッドの数です。開かれているスレッド数が下限を上回ると、Oracle BI Server はアイドル状態のスレッドを終了します。たとえば、DB_GATEWAY_THREAD_RANGE が 40 ～ 200 に設定され、75 のスレッドが開いている場合、Oracle BI Server はアイドル状態のスレッドを終了します。

マルチディメンショナル・データソースに対する接続プールのプロパティの設定

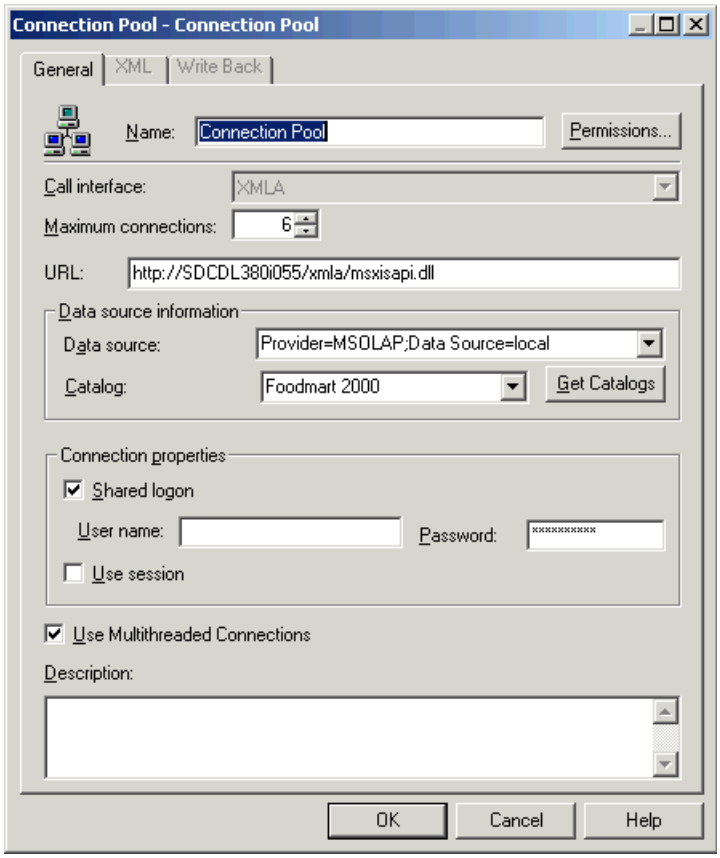
外部のマルチディメンショナル・データソースをインポートすると、接続プールが自動的に物理レイヤーに設定されます。接続プールは、この項で説明する手順に従って手動で追加できます。

74 ページの表 8 では、複数のデータソースの接続プールで共有される一般的なプロパティの一部について説明します。79 ページの表 9 では、このダイアログ・ボックスに表示される、マルチディメンショナル・データソースに固有のプロパティについて説明します。

マルチディメンショナル・データソースへの接続プールを設定するには

1 Administration Tool の「Physical」レイヤーでデータベースを右クリックして、「New Object」→「Connection Pool」を選択するか、既存の接続プールをダブルクリックします。

マルチディメンショナル・データソースの場合、「Connection Pool」ダイアログ・ボックスの「General」タブは、次のようになっています。



2 「Connection Pool」ダイアログ・ボックスの「General」タブで、74 ページの表 8 および 79 ページの表 9 の記述に従ってフィールドに値を入力します。

注意： 79 ページの表 9 では、マルチディメンショナル・データソースに固有の接続プールのプロパティおよび内容について説明します。

表 9. マルチディメンショナル・データソースに対する接続プールの一般プロパティ

プロパティ	説明
Data Source Information: Catalog	自分のデータソースからデータをインポートした場合に使用できるカタログのリストです。キューブ・テーブルは、接続プールで使用するカタログに対応しています。
Data Source Information: Data Source	マルチディメンショナル・データソースへの接続に使用するベンダー固有の情報です。仕様が異なる場合があるため、設定手順については、マルチディメンショナル・データソースの管理者に相談してください。たとえば、XML for Analysis SDK の v1.0 を使用している場合、この値は `Provider-MSOLAP;Data Source-local` となります。v1.1 の場合は `Local Analysis Server` となります。
Shared logon	「Shared logon」チェック・ボックスを選択するのは、接続プールを使用したクエリーを実行するユーザーがすべて、同じユーザー名およびパスワードを使用して基礎となるデータベースにアクセスできるようにする場合です。 このオプションを選択すると、データベースに対して接続プールを使用するデータベース接続はすべて、接続プールで指定したユーザー名およびパスワードを使用します。これは、そのユーザーが、DSN（またはユーザー構成）でデータベースのユーザー名およびパスワードを指定した場合でも同様です。 このオプションを選択しないと、接続プールを経由する接続は、DSN またはユーザー・プロファイルで指定したデータベースのユーザー ID およびパスワードを使用します。
URL	XMLA プロバイダに接続する URL です。キューブをホストするマシンの XMLA 仮想ディレクトリをポイントします。この仮想ディレクトリは、msxisapi.dll（Microsoft XML for Analysis SDK インストールの一部）に関連付ける必要があります。たとえば、URL は次のようになります。 http://SDCDL360i101/xmla/msxisap.dll
Use session	クエリーが共通セッションを経由するかどうかを制御します。このオプションを有効にする必要があるかどうかについては、マルチディメンショナル・データソースの管理者に相談してください。デフォルトではオフ（選択解除）になっています。

XML データソースに対する接続プールの追加プロパティの設定

「Connection Pool」ダイアログ・ボックスの「XML」タブを使用して、XML データソースの追加プロパティを設定します。

警告：「Connection Pool」ダイアログ・ボックスの「XML」タブには、「Physical Table」ダイアログ・ボックスの「XML」タブと同じ機能があります。ただし、「Physical Table」ダイアログ・ボックスの「XML」タブでプロパティを設定すると、「Connection Pool」ダイアログ・ボックスで対応する設定より優先されます。

XML データソースへの接続プールのプロパティを設定するには

1. XML データベースを右クリックして、「New Object」→「Connection Pool」を選択します。
2. 「Connection Pool」ダイアログ・ボックスで「XML」タブをクリックします。
3. 80 ページの表 10 を参照し、フィールドに値を入力します。

表 10. XML 接続プールのプロパティ

プロパティ	説明
Connection method Search script	XML Server データソースに使用します。
Connection properties Maximum connections	デフォルトは 10 です。
Connection properties URL loading time out	XML データソースに使用します。クエリーのタイムアウト間隔です。デフォルトは 15 分です。 データソースにアクセスする URL を指定した場合は、URL ロードのタイムアウトを次のように設定します。 ■ ドロップダウン・リストから値を選択する（「Infinite」、「Days」、「Hours」、「Minutes」または「Seconds」） ■ 間隔の数字部分として数字全体を指定する
Connection properties URL refresh interval	XML データソースに使用します。リフレッシュ間隔は、データベース・テーブルのキャッシュ保持設定に類似しています。URL のリフレッシュ間隔とは、キャッシュにある結果を使用しないで、XML データソースに対して再度クエリーが直接発行されるようになるまでの時間間隔です。デフォルトの設定は「Infinite」で、XML データソースがリフレッシュされることはありません。 データソースにアクセスする URL を指定した場合は、URL のリフレッシュ間隔を次のように設定します。 ■ ドロップダウン・リストから値を選択する（「Infinite」、「Days」、「Hours」、「Minutes」または「Seconds」） ■ 間隔の数字部分として数字全体を指定する

表 10. XML 接続プールのプロパティ

プロパティ	説明
Query input supplements Header file/Trailer file	XML Server データソースに使用します。
Query output format	XML データソースの XML のみを選択します。 XML Server データソースでは他の選択も可能です。
XPath expression	XPath 式は、1 行に収まる単純な XSLT 変換ルールです。XSLT がサポートされていることを考えると重要なことではありませんが、利便性を考えてサポートしています。たとえば、*/BOOK[not(PRICE>'200')] のようなエントリになります。 ■ XML Server データソースの場合、「Physical Table」オブジェクトの「XML」タブで XPath 式を指定することはできません。
XSLT file	XSLT ファイルには、XSLT 規格に従って記述されたフォーマット・ルールが含まれています。ここでは、XML ファイルの変換方法が定義されています。現在実装されている XML ゲートウェイでは、すべての XML ファイルをサポートしているわけではなく、第 2 レベルの要素が繰り返される、テーブルに似たフォーマットのみがサポートされています。XML ゲートウェイの汎用性を高めるため、Oracle BI Server がサポートしているフォームに XML ファイルを事前に変換する XSLT ファイルを顧客が指定できます。接続プールで XSLT ファイルを指定すると、その指定が接続プールのすべての XML 物理テーブルに適用されます。 ■ XML データソースの場合、「Physical Table」オブジェクトの「XML」タブで、テーブル単位で XSLT ファイルを指定できます。これは、接続プールの指定より優先されます。 ■ XML Server データソースの場合、「Physical Table」オブジェクトの「XML」タブで XSLT ファイルを指定することはできません。

クエリー出力フォーマットの設定を指定するには

1 （オプション）XSLT ファイルの場合、「XSLT file」フィールドに XSLT ファイルのパスと名前を入力するか、「Browse」ボタンを使用します。

2 （オプション）XPath 式の場合、「XPath Expression」フィールドに XPath 式（//XML など）を入力するか、「Browse」ボタンを使用します。

ライトバック・プロパティの設定

「Connection Pool」ダイアログ・ボックスの「Write Back」タブで値を入力するには、この項を参照してください。

接続プールのライトバック・プロパティを設定するには

1. Administration Tool の「Physical」レイヤーでデータベースを右クリックして、「New Object」→「Connection Pool」を選択するか、既存の接続プールをダブルクリックします。
2. 「Connection Pool」ダイアログ・ボックスで「Write Back」タブをクリックします。
3. 82 ページの表 11 を参照し、「Write Back」タブのフィールドに値を入力します。

表 11. 「Write Back」タブのフィールドの説明

フィールド	説明
Bulk Insert Buffer Size (KB)	データベース・テーブルにデータを挿入するときの、1 回当たりのバイト数の制限に使用します。
Bulk Insert Transaction Boundary	データベース・テーブルに挿入するバッチのサイズを制御します。
Temporary table Database Name	一時テーブルを作成するデータベースです。IBM OS/390 では CREATE TABLE 文の一部となるデータベース名の修飾子が必要であるため、このプロパティは IBM OS/390 にのみ適用されます。空白のままにした場合、OS/390 では、ターゲット・データベースが、ユーザーに Create Table 権限が割り当てられないシステム・データベースにデフォルト設定されます。
Temporary table Owner	たとえば、owner.tablename というテーブルを作成する場合に SQL 文で一時テーブルの修飾名として使用されるテーブルの所有者名です。空白のままにした場合、書込み可能な接続プールで指定したユーザー名がテーブルの修飾名として使用され、「General」タブの「Shared Logon」フィールドも設定されます。
Temporary table Prefix	Oracle BI Server で一時テーブルを作成したときの、一時テーブル名の最初の 2 文字となります。デフォルトは TT です。
Temporary table Tablespace Name	一時テーブルを作成する表領域です。OS/390 では CREATE TABLE 文の一部となる一時テーブル名の修飾子が必要であるため、このプロパティは OS/390 にのみ適用されます。空白のままにした場合、OS/390 では、ターゲット・データベースが、ユーザーに Create Table 権限が割り当てられないシステム・データベースにデフォルト設定されます。

表 11. 「Write Back」タブのフィールドの説明

フィールド	説明
Unicode Database Type	このチェック・ボックスは、Unicode データベースで明示的な Unicode データ型（NCHAR など）の列で操作するときに選択します。これによって、バインドが正しく設定され、データが正しく挿入されます。データベース・ベンダーごとに、文字データ型が異なり、Unicode サポートのレベルも異なります。このチェック・ボックスをいつ設定するか判断するには、次の一般的なガイドラインに従ってください。 ■ CHAR データ型が Unicode をサポートしていて、それとは別に NCHAR データ型がないデータベースでは、このチェック・ボックスは選択しないでください。 ■ NCHAR データ型が使用可能なデータベースでは、このチェック・ボックスを選択することをお薦めします。 ■ CHAR データ型と NCHAR データ型が Unicode をサポートするように構成されているデータベースでは、このチェック・ボックスの選択は任意です。 **注意：** Unicode および非 Unicode のデータ型は、1 つの非 Unicode データベースに共存させることはできません。たとえば、1 つの非 Unicode データベース環境に CHAR と NCHAR のデータ型を混在させることはできません。

永続接続プールのプロパティの設定

永続接続プールは、特定のタイプのクエリーに使用されるデータベースのプロパティです（通常は、Marketing のクエリーのサポートに使用されます）。クエリーの関数には、データベースでサポートされていないものもあるため、一部のクエリーでは、すべての論理クエリーをトランザクション・データベースに送信できるわけではありません。データベースに物理テーブルを一時的に構築し、Oracle BI Server のクエリーをリライトして、一時的な新しい物理テーブルを参照するという方法で解決できることがあります。

永続接続プールを使用できるのは、次の場合です。

- **ストアド・プロシージャに移入するとき**：論理 SQL 結果セットを管理対象テーブルにリライトするときに使用します。通常は、Oracle Siebel Marketing Server がセグメント化のキャッシュ結果セットを書き込む場合に使用します。
- **一般化されたサブクエリーを実行するとき**：非関数のサブクエリーを一時テーブルに格納し、そのテーブルに対する元のサブクエリーをリライトします。Oracle BI Server とデータベース間のデータ移動が少なくなり、IN リストの値が無制限にサポートされるので、結果的にパフォーマンスが向上することがあります。

注意： この場合、論理 SQL を発行しているユーザーは、ターゲット・データベースに対する Populate 権限が付与されている必要があります。

永続接続プールの機能では、ライトバック可能な接続プールを指定して、このタイプのクエリーを処理します。1 つのデータベースでは、1 つの接続プールを永続接続プールとして割り当てることができます。この機能が有効な場合、接続プールで指定したユーザー名には、データベースに DDL（データ定義言語）と DML（データ操作言語）を作成する権限が必要です。

バッファ・サイズとトランザクション境界の使用例

結果セットの各行のサイズが 1KB でバッファ・サイズが 20KB の場合、配列の最大サイズは 20KB になります。120 行では 6 バッチとなり、各バッチのサイズは 20 行に制限されます。

「Transaction boundary」フィールドを 3 に設定すると、サーバーは 2 回コミットします。サーバーが 1 回目にコミットするのは、第 60 行（3 * 20）の後です。サーバーが 2 回目にコミットするのは、第 120 行の後です。サーバーのコミットが失敗すると、サーバーは現在のトランザクションのみをロールバックします。たとえば、コミットが 2 回あり、最初のコミットは成功したが 2 回目のコミットは失敗した場合、サーバーは 2 回目のコミットのみをロールバックします。配列に基づいて挿入を正しく実行するには、10 より大きなトランザクション境界を設定しないで、バッファ・サイズを約 32KB に設定することをお薦めします。

永続接続プールを割り当てるには

1. 「Physical」レイヤーで、データベースのアイコンをダブルクリックします。
2. 「Database」ダイアログ・ボックスで「General」タブをクリックします。
3. 「Persist Connection Pool」領域で「Set」をクリックします。

   接続プールが 1 つしかない場合は、「Persist Connection Pool」フィールドに接続プールが表示されます。
4. 接続プールが複数ある場合は、「Browse」ダイアログ・ボックスで適切な接続プールを選択してから「OK」をクリックします。

   選択した接続プールの名前が「Persist Connection Pool」フィールドに表示されます。
5. （オプション）ライトバック・プロパティを設定するには、「Connection Pools」タブをクリックします。
6. 接続プールのリストで、目的の接続プールをダブルクリックします。
7. 「Connection Pool」ダイアログ・ボックスで「Write Back」タブをクリックします。
8. 82 ページの表 11 を参照し、フィールドに値を入力します。
9. 「OK」を 2 回クリックして、永続接続プールを保存します。

永続接続プールを削除するには

1. 「Physical」レイヤーで、データベースのアイコンをダブルクリックします。
2. 「Database」ダイアログ・ボックスで「General」タブをクリックします。
3. 「Persist Connection Pool」領域で「Clear」をクリックします。

   「Persist Connection Pool」フィールドで、データベース名が `not assigned` に置き換えられます。
4. 「OK」をクリックします。

物理テーブルについて

この項は、「リレーショナル・データソースから物理レイヤーを作成するプロセス」（60 ページ）および「マルチディメンショナル・データソースから物理レイヤーを作成するプロセス」（62 ページ）の一部です。

物理テーブルとは、Administration Tool の「Physical」レイヤーに含まれるオブジェクトで、物理データベースのテーブルに対応します。物理テーブルは通常、データベースまたは別のデータソースからインポートされます。Oracle BI Server が SQL リクエストでテーブルにアクセスするために必要なメタデータが用意されます。

物理テーブルのインポートに加え、「Physical Table」ダイアログ・ボックスの「Table Type」フィールドに指定されている値を使用して、「Physical」レイヤーの仮想物理テーブルを作成することもできます。仮想物理テーブルは、ストアド・プロシージャまたは SELECT 文です。仮想テーブルを作成すると、Oracle BI Server および基礎となるデータベースに対し、一部の拡張クエリー・リクエストを実行するための適切なメタデータが用意される場合があります。

物理テーブルのテーブル・タイプ

「Physical Table」ダイアログ・ボックスの「General」タブにある「Table Type」ドロップダウン・リストでは、物理テーブルのオブジェクト・タイプを指定できます。85 ページの表 12 では、使用可能なオブジェクト・タイプについて説明します。

表 12. 物理テーブルのテーブル・タイプの説明

テーブル・タイプ	説明
Physical Table	物理テーブルのオブジェクトが物理テーブルを表すことを指定します。
Stored Proc	物理テーブルのオブジェクトがストアド・プロシージャであることを指定します。このオプションを選択した場合は、テキスト・ボックスにストアド・プロシージャを入力します。このテーブルに対するリクエストが、ストアド・プロシージャをコールします。 ストアド・プロシージャがデータベース固有の場合は、「Use database specific SQL」チェック・ボックスを選択します。実行時にストアド・プロシージャが定義されている場合は、そのストアド・プロシージャが実行されます。それ以外の場合はデフォルトの構成が実行されます。 **注意：** Oracle データベースを使用するストアド・プロシージャは、結果セットを返しません。詳細は、「Oracle データベースでのストアド・プロシージャの使用」（86 ページ）を参照してください。 ストアド・プロシージャおよび別名テーブルの詳細は、「物理別名テーブルについて」（86 ページ）を参照してください。

表 12. 物理テーブルのテーブル・タイプの説明

テーブル・タイプ	説明
Select	物理テーブルのオブジェクトが SELECT 文であることを指定します。このオプションを選択した場合は、テキスト・フィールドに SELECT 文を入力します。次に、テーブル列を手動で作成する必要があります。列名は、SELECT 文で指定した列名と一致させてください。列の別名は、集計や CASE 文などの拡張 SQL 機能に必要です。 このテーブルに対するリクエストが、SELECT 文を実行します。 SELECT 文がデータベースに固有の場合は、「Use database specific SQL」チェック・ボックスを選択します。実行時に SELECT 文が定義されている場合は、その SELECT 文が実行されます。それ以外の場合はデフォルトの構成が実行されます。

Oracle データベースでのストアド・プロシージャの使用

Oracle 内のストアド・プロシージャは、結果セットを返しません。したがって、Oracle BI 内から開始することはできません。プロシージャを Oracle ファンクションとしてリライトし、それを Administration Tool の初期化ブロックの SELECT 文で使用し、「Session Variables」ダイアログ・ボックスで Oracle BI セッションの適切な変数に関連付ける必要があります。

このファンクションは GET_ROLES ファンクションを使用し、ユーザー ID をパラメータとして取得して、セミコロンで区切ったグループ名のリストを返します。

初期化 SQL 文字列の例を次に示します。ここでは、USER、GROUP、DISPLAYNAME の各変数に関連付けられた GET_ROLES ファンクションを使用しています。

```
select user_id, get_roles(user_id), first_name || ' ' || last_name
   from csx_security_table
   where user_id = ':USER' and password = ':PASSWORD'
```

物理テーブルの作成および設定

この項は、「リレーショナル・データソースから物理レイヤーを作成するプロセス」（60 ページ）および「マルチディメンショナル・データソースから物理レイヤーを作成するプロセス」（62 ページ）の一部です。

すべてのデータソースに対して、一般的なプロパティ、列、主キーおよび外部キーを定義できます。

物理別名テーブルについて

別名テーブル（別名）は、タイプが Alias の物理テーブルです。これは、論理テーブル・ソースを参照しています。また論理テーブル・ソースのすべての列定義と一部のプロパティを継承しています。論理テーブル・ソースでは、物理レイヤー、物理テーブル、ストアド・プロシージャおよび SELECT 文に論理オブジェクトをマッピングする方法を示します。別名テーブルは、これらの論理テーブルのソース・タイプを参照しています。詳細は、「Oracle BI リポジトリの「Business Model and Mapping」レイヤーの作成と管理」（113 ページ）を参照してください。

別名テーブルは、物理レイヤーを設計するうえで重要です。別名テーブルを作成する主な理由を次に示します。

- 物理レイヤーで既存のテーブルを再利用するため（何度もインポートする必要がない）。
- 複数の別名テーブルを設定し、テーブルごとに別々のキー、名前および結合を設定するため。
- ビジネス・モデル・レイヤーに、高度なスターやスノーフレークを設計するため。別名テーブルは、ER スキーマをディメンショナル・スキーマに変換するプロセスで重要な役割を果たします。詳細は、「ビジネス・モデルのデータベース・コンテンツの識別」（48 ページ）を参照してください。

別名テーブルでは、「Physical Table」ダイアログ・ボックスで優先フラグを設定することにより、ソース・テーブルとは異なるキャッシュ・プロパティを持たせることができます。別名テーブルでは、列の追加、削除または変更はできません。列は自動的に同期化されるので、手動の処理は不要です。

注意：同期化によって、ソース・テーブルとそれに関連する別名テーブルでは、列定義が同じになります。たとえば、ソース・テーブルから列を削除すると、その列は別名テーブルからも自動的に削除されます。

すべての別名テーブルを先に削除しないかぎり、ソース・テーブルは削除できません。別名テーブルのソース・テーブルは、新しいソース・テーブルが現在のソース・テーブルのスーパーセットである場合には変更できます。ただし、ソース・テーブルを変更したことによって、使用中の列が削除された場合、リポジトリの整合性が失われる可能性があります。変更しようとすると、問題が起こる可能性があることを知らせる警告メッセージが表示され、操作を取り消すことができます。

注意：整合性チェックを実行すると、孤立している別名が特定されます。

オンライン・モードで物理テーブルまたは列を編集するときには、別名テーブルおよび列をすべてチェックアウトする必要があります。オンラインでのチェックアウトの動作は、次の規則に従います。

- ソース・テーブルまたは列をチェックアウトすると、対応する別名テーブルおよび列もすべてチェックアウトされます。
- 別名テーブルまたは列をチェックアウトすると、対応するソース・テーブルおよび列もチェックアウトされます。
- チェックアウト・オプションは、オンラインのリポジトリ（読取り専用でない場合）、すべてのソース・テーブルと列、すべての別名テーブルと列で使用できます。

別名テーブルは、対応するソース・テーブルの一部のプロパティを継承します。プロキシを設定しているプロパティは、常にソース・テーブルと同じ値であり、変更できません。ソース・テーブルでそのプロパティの値を変更すると、対応する別名テーブルでも同じ変更が行われます。

プロキシが設定されるプロパティを次に示します。

- IsCacheable（継承されたプロパティは上書き可能）
- CacheExpiry（継承されたプロパティは上書き可能）
- Row Count
- Last Updated
- Table Type
- External Db Specifications

プロキシが設定されないプロパティを次に示します。

- Name

- Description
- Display Folder Containers
- Foreign Keys
- Columns（テーブルが列を共有することは絶対にありません。別名とソースは、互いを別名とするまったく異なる列を持ちます）
- Table Keys
- Complex Joins
- Source Connection Pool
- Polling Frequency
- すべての XML 属性

マルチディメンショナル・データソースに対する物理テーブルの作成および設定について

マルチディメンショナル・データソースの各キューブは、物理テーブルの一種である物理キューブ・テーブルとして設定されます。物理キューブの列とキー（オプション）、外部キー（オプション）など、テーブルの機能がすべて含まれています。また階層やレベルなど、キューブ固有のメタデータもあります。

「Physical」レイヤーでは、物理キューブ・テーブルは通常のテーブルのように表示されますが、アイコンが異なります。アイコンの詳細は、「Administration Tool のアイコンと記号」（22 ページ）を参照してください。

物理スキーマをインポートすると、Oracle BI Server では、キューブが、そのメトリック、階層およびレベルとともにインポートされます。階層のアイコンを展開すると、その階層のレベルが表示されます。「Physical Cube Table」ダイアログ・ボックスの「Hierarchies」タブには、キューブのディメンショナル階層が一覧表示されます。

データベースの各マルチディメンショナル・カタログには、複数の物理キューブを含めることができます。これらのキューブを 1 つ以上、BI リポジトリにインポートできます。キューブ・テーブルは手動で作成できます。ただし、キューブ・テーブルとそのコンポーネントはインポートすることをお薦めします。

注意：キューブを手動で作成した場合、各キューブは 1 つずつの階層として構築し、構築したらそのたびに、別のキューブの構築前にテストすることを強くお薦めします。たとえば、時間の階層とメジャーを作成してテストします。問題がなければ、地理階層を作成してそれをテストします。これによって各キューブが正しく設定され、設定エラーを特定しやすくなります。

物理テーブルまたは物理キューブ・テーブルと必要なプロパティを作成するには、次の作業を実行します。

- 物理テーブルの一般プロパティの作成および管理（89 ページ）
- 物理テーブルでの列およびキーの作成と管理（91 ページ）
- マルチディメンショナル・データソースの物理レイヤーにおける階層の設定（95 ページ）
- XML データソースの物理テーブルのプロパティ設定（100 ページ）

物理テーブルの一般プロパティの作成および管理

「Physical Table」ダイアログ・ボックスの「General」タブを使用して、Administration Tool の「Physical」レイヤーで物理テーブルを作成または編集します。

この項の内容は次のとおりです。

物理テーブルの作成または編集

この項では、テーブルの一般プロパティを作成または編集する方法について説明します。物理キューブ・テーブルと別名テーブルも対象です。

物理テーブルの作成、またはテーブルおよび別名テーブルの一般プロパティを編集するには

1. Administration Tool の「Physical」レイヤーで、次の手順を実行します。
   - 物理テーブルを作成するには、物理データベースを右クリックして、「New Object」→「Physical Table」を選択します。
   - マルチディメンショナル・データソースの物理キューブ・テーブルを作成するには、物理データベースを右クリックして、「New Object」→「Physical Cube Table」を選択します。

     注意：キューブ・テーブルは、手動で作成しないでインポートすることを強くお薦めします。
   - 別名テーブルを作成するには、物理テーブルを右クリックして、「New Object」→「Alias」を選択します。

     注意：また、不明瞭ビューとストアド・プロシージャの別名も作成できます。
   - 既存の物理テーブルを編集するには、物理テーブルのアイコンをダブルクリックします。
2. 89 ページの表 13 を参照し、選択した「Physical Table」ダイアログ・ボックスで、フィールドに値を入力します。

表 13. リレーショナル・データソースおよび XML データソースの物理テーブルの一般プロパティ

プロパティ	説明
Name	Oracle BI の管理者が、新しいテーブルに名前を割り当てます。
Cacheable	Oracle BI Server のクエリー・キャッシュにテーブルを含めるには、このチェック・ボックスを選択します。このチェック・ボックスを選択すると、「Cache persistence time」の設定がアクティブになります。これは、OLTP データソースなど、頻繁に更新されるデータソースに便利です。通常、ほとんどのテーブルではこのオプションを選択してください。

表 13. リレーショナル・データソースおよび XML データソースの物理テーブルの一般プロパティ

プロパティ	説明
Cache never expires	このオプションを選択すると、キャッシュ・エントリが期限切れになることはなくなります。ユーザーが実行する大量のクエリーにとってテーブルが重要である場合、このオプションは便利です。たとえば、大半のクエリーにアカウント・オブジェクトへの参照が含まれている場合には、無期限にキャッシュされるようにしておくと、実際のパフォーマンスが損なわれることはなく、むしろ向上します。 **警告：** これは、一部のオブジェクトでのみ使用します。このオプションを設定したオブジェクトが多すぎると、管理対象のキャッシュが大きくなり、キャッシュからオブジェクトが抜け落ちて効率が低下する可能性があります。
Cache persistence time	テーブルのエントリをクエリー・キャッシュに保持する期間です。デフォルト値は「Infinite」です。つまり、キャッシュ・エントリが自動的に期限切れになることはありません。ただし、これは、エントリが常にキャッシュに残っているという意味ではありません。手動消去、LRU（最低使用頻度）の置換、メタデータの変更、キャッシュのポーリング・テーブルの使用などの無効化手法を実行すると、キャッシュからエントリが削除されます。 永続時間が異なる複数の物理テーブルをクエリーで参照している場合、そのクエリーのエントリは、クエリーで参照しているテーブルのうち、最短に設定された永続時間が経過すると、キャッシュから削除されます。これで、以降のクエリーでは、期限の切れたキャッシュ・エントリからキャッシュ・ヒットを取得することはなくなります。 デフォルトを分または秒に変更する場合は、左側のフィールドに、数値を省略せずに入力してください。 詳細は、「イベント・ポーリング・テーブルに発生した問題のトラブルシューティング」（262 ページ）を参照してください。
External name	マルチディメンショナル・データソースの物理キューブ・テーブルに適用されます。「Table Type」に「Physical Table」を選択した場合は、外部データソース名が表示されます。
Override Source Table Caching Properties	別名テーブルで使用できるチェック・ボックスです。選択すると、キャッシュ可能なプロパティが使用可能になり、適切なオプションを選択または選択解除できます。
Source Table	別名テーブルに適用されます。「Select」ボタンを使用すると、別名テーブルの作成元となる物理テーブルを選択できます。
Table Type	「Physical Table」の値：「Physical Table」、「Stored Proc」（ストアド・プロシージャ）、「Select」のいずれか。 「Physical Cube Table」の値：「Physical Table」または「Select」。

表 13. リレーショナル・データソースおよび XML データソースの物理テーブルの一般プロパティ

プロパティ	説明
Use Dynamic Name	「Table Type」に「Physical Table」を選択したときに、非マルチディメンショナル・データソース・テーブルで使用できるチェック・ボックスです。選択すると、セッション変数を選択できるダイアログ・ボックスが開きます。
Use Database Specific SQL Default Initialization String	非マルチディメンショナル・データソース・テーブル（別名テーブルではない）では、「Table Type」に「Stored Proc」または「Select」を選択した場合に表示されます。マルチディメンショナル・データソース・テーブルでは、「Table Type」に「Select」を選択した場合に表示されます。 このチェック・ボックスを選択すると、データベースを指定して SQL を入力できます。

物理テーブルの削除

物理テーブルを削除すると、依存オブジェクトがすべて削除されます。列、キー、外部キーなどがこれに該当します。物理キューブ・テーブルを削除すると、階層も削除されます。

注意：その物理テーブルに別名がある場合、削除は失敗します。

「Physical」レイヤーから物理テーブルを削除するには

1. Administration Tool の「Physical」レイヤーで、削除するテーブルを検索します。
2. 目的のテーブルを右クリックし、「Delete」を選択します。

物理テーブルまたは列でのデータ表示

オブジェクトを右クリックして「View Data」を選択すると、物理テーブルまたは個別の物理列にデータを表示できます。オンライン編集モードでは、変更をチェックインしておかないと「View Data」が使用できません。

物理キューブ・テーブルまたは列では「View Data」は使用できません。詳細は、「物理キューブ・テーブルのメンバーの表示」（98 ページ）を参照してください。

警告：「View Data」は、接続プールのユーザー名を :USER、パスワードを :PASSWORD に設定した場合は使用できません。オフライン・モードの場合は「Set values for variables」ダイアログ・ボックスが表示され、表示プロセスの一部として :USER および :PASSWORD に値を設定できます。

物理テーブルでの列およびキーの作成と管理

Administration Tool の「Physical」レイヤーの各テーブルには、1 つ以上の物理列があります。

「Physical Table」ダイアログ・ボックスの「Columns」、「Keys」、「Foreign Keys」の各タブを使用すると、テーブルに関連付けられた既存の列、キーおよび外部キーの表示と編集、またそれらの新規作成ができます。

次に、タブに表示されるボタンについて説明します。

- **New:** タブに対応するダイアログ・ボックスを開きます。

- **Edit:** オブジェクトを選択して「Edit」をクリックすると、タブに対応するダイアログ・ボックスが開きます。ここでオブジェクトのプロパティを編集できます。
- **Delete:** 選択したオブジェクトを削除します。

この項の内容は次のとおりです。

マルチディメンショナル・データソースのメジャーについて

物理キューブ列の集計ルールは、メジャーが正しくなるように慎重に選択する必要があります。正しく設定すると、パフォーマンスが向上する可能性があります。

集計ルールの検証と割当てを正しく行うには、次のガイドラインを参照してください。

- 集計ルールは、キューブをインポートしてから検証します。通常、集計ルールは、キューブをインポートしたときに正しく割り当てられます。ただし、メジャーが計算されたものである場合、集計ルールは「None」として報告されます。したがって、キューブをインポートした後ですべてのメジャーの集計ルールを調べ、集計ルールが正しく割り当てられていることを確認する必要があります。

  値が「None」の集計ルールが割り当てられているすべてのメジャーにおいて、集計ルールの値が正しいかどうかをマルチディメンショナル・データソース管理者に確認してください。集計ルールの変更が必要な場合は、「Physical Cube Column」ダイアログ・ボックスで変更します。

- メジャーを手動で作成したときに集計ルールを設定します。集計ルールは、マルチディメンショナル・データソースの定義と一致するように設定します。

外部集計メジャーについて

マルチディメンショナル・データソースには、非常に複雑な、マルチレベル・ベースのメジャーが含まれているキューブもあります。Oracle BI の管理者が Aggr_External という集計ルールを割り当てた場合、BI Server では、内部集計メカニズムを経由せずに事前集計されたメジャーを使用します。これらのメジャーには、インポート時に「None」という集計値が割り当てられます。

事前集計されたメジャーを操作する際のガイドラインを次に示します。

- 外部集計は、このような複雑な計算をサポートするマルチディメンショナル・データソース（MS Analysis Services や SAP/BW など）にのみ適用されます。
- 外部集計は、標準的なデータソース（リレーショナル）のメジャーに割り当てないでください。サポートされたメジャーを使用しており、リレーショナル・データベースへのマッピングが可能な場合には、そのメジャーは複雑ではないため、外部集計は不要です。
- 標準データソース（リレーショナル）の複雑ではないメジャーと、マルチディメンショナル・データソースの複雑なメジャーは混在させないでください。
- Oracle BI Server で集計する場合は、標準データソース（リレーショナル）の複雑ではないメジャーと、マルチディメンショナル・データソースの複雑ではないメジャーは混在させてもかまいません。

物理テーブルでの列の作成および編集

列をインポートした場合、列のプロパティが自動的に設定されます。次に、「Physical」レイヤーにインポートした列の NULL 値可能な値およびデータ型の値について説明します。

- **Nullable:** 「Physical Column」ダイアログ・ボックスの「Nullable」オプションでは、その列で NULL 値が許可されているかどうかを示します。基礎となるテーブルに NULL 値を含めることができる場合は、このオプションを選択する必要があります。これによって、NULL 値をユーザーに返すことができます。これは一部の関数または外部結合で使用されます。一般的には、物理列では、NULL 値不可の値を NULL 値可能な値に変更しておくと安全です。
- **Data type:** 「Data type」リストでは、その列のデータ型を示します。データ型の値を変更するときは注意が必要です。基礎となるデータソースで不正な値を設定すると、予期しない結果となることがあります。データ型の不一致がある場合は、リポジトリで修正するか、一致しなかったデータ型で列を再インポートします。

  列を再インポートした場合には、再マッピングした列を参照する論理列ソースも再度マッピングする必要があります。ビジネス・モデルにおける論理列のデータ型は、その物理列ソースのデータ型と一致させる必要があります。Oracle BI Server では、論理列にあるこれらのデータ型をクライアント・アプリケーションに渡します。

注意： 別段の指定がある場合を除き、物理キューブ列と他の物理列では、特性と動作が同じです。

別名テーブルに関連する列がある列の作成および編集について

別名テーブルに対応する列を持つ物理ソース・テーブルの列を作成および編集すると、次の結果になります。

- ソース列の作成。物理ソース・テーブルに列を作成すると、対応するすべての別名テーブルに同じ列が作成されます。手順を次に示します。
- ソース列の削除。物理ソース・テーブルから列を削除すると、対応するすべての別名テーブルから同じ列が削除されます。
- ソース列の変更。物理ソース・テーブルにある列を変更すると、対応するすべての別名テーブルにある同じ列が変更されます。

物理列を作成または変更するには

1. Administration Tool の「Physical」レイヤーで、次の手順を実行します。
   - 物理列を作成するには、物理テーブルを右クリックして、ショートカット・メニューから「New Object」→「Physical Column」を選択します。
   - マルチディメンショナル・データソースの物理キューブ列を作成するには、物理キューブ・テーブルを右クリックして、「New Object」→「Physical Cube Column」を選択します。
   - 既存の物理列を編集するには、物理列のアイコンをダブルクリックします。
2. 「Physical Column」ダイアログ・ボックスで、物理列の名前を入力します。

   XML データソースの場合、このフィールドには、XML 文書の列（属性）の非修飾名が格納および表示されます。
3. 「Type」フィールドに、物理列のデータ型を選択します。

4 必要に応じて、データ型の長さを指定します。

マルチディメンショナル・データソースで VARCHAR を選択した場合は、「Length」フィールドに値を入力する必要があります。

5 列に NULL 値が許可されている場合は、「Nullable」オプションを選択します。

6 「External Name」フィールドに外部名を入力します。

- これは、同じ名前（STATE など）を複数の階層で使用している場合に必要です。
- XML 文書の場合はオプションです。「External Name」フィールドには、列（属性）の完全修飾名が格納および表示されます。

7 （マルチディメンショナル・データソースの場合）物理キューブ列がメジャーのとき、「Aggregation rule」ドロップダウン・リストで、適切な値を選択します。

注意：新しい物理キューブ列が、デフォルトではメジャーとして作成されます。これを変更するには、「マルチディメンショナル・データソースの物理レイヤーにおける階層の設定」（95 ページ）を参照してください。

8 「OK」をクリックします。

物理テーブルの主キーの指定

「Physical Key」ダイアログ・ボックスを使用して、物理テーブルの主キーを定義する列（複数も可）を指定します。

物理テーブルの主キーを指定するには

1 Administration Tool の「Physical」レイヤーで物理テーブルを右クリックして、「Properties」を選択します。

2 「Physical Table」ダイアログ・ボックスで「Keys」タブをクリックします。

3 「Keys」タブで「New」をクリックします。

4 「Physical Key」ダイアログ・ボックスで、キーの名前を入力します。

5 物理テーブルの主キーを定義する列のチェック・ボックスを選択します。

6 （オプション）「Physical Key」ダイアログ・ボックスで、キーの説明を入力して「OK」をクリックします。

すべてのデータソースでの物理列の削除

すべてのデータソースで、同じ方法で物理列を削除できます。結果としては、次のようなものがあります。

- **マルチディメンショナル・データソース**：あるレベルからプロパティまたはキーの列を削除すると、関連付けも削除され、その列は親キューブ・テーブルの下のメジャーに変更されます。
- **別名テーブル**：別名テーブルの作成元となった物理ソース・テーブルから列を削除すると、対応するすべての別名テーブルから同じ列が削除されます。

「Physical」レイヤーから物理列を削除するには

1 Administration Tool の「Physical」レイヤーで、削除する列を検索します。

2 目的の列を右クリックし、「Delete」を選択します。

マルチディメンショナル・データソースの物理レイヤーにおける階層の設定

「Physical」レイヤーで階層を設定する際のガイドラインを次に示します。

- 不規則な階層または親子関係を持つ階層はインポートされません。物理レイヤーでは、階層タイプを変更することにより、不均等な階層を設定できます。
- メジャーからプロパティまたはレベル・キーに列を変更するには、階層を設定して、その階層とキューブ列を関連付ける必要があります。あるレベルからプロパティまたはレベル・キーの列を削除すると、その列は、親キューブ・テーブルの下のメジャーに戻ります。

  警告：「Business Model and Mapping」レイヤーに、一致する階層を作成する必要があります。この処理をしておかないと、クエリーは機能しているように見えても正しい結果が返ってこない場合があります。

「Physical」レイヤーで階層を作成およびメンテナンスするには、次の作業を実行します。

- 物理キューブ・テーブルへの階層の追加（95 ページ）
- 階層レベルの確認（97 ページ）
- メンバー・カウントの更新（97 ページ）
- 物理キューブ・テーブルのメンバーの表示（98 ページ）
- 既存の階層でのキューブ列の追加または削除（98 ページ）
- 物理キューブ・テーブルからの階層の削除（99 ページ）
- 物理キューブ列と階層レベルの関連付け（99 ページ）

物理キューブ・テーブルへの階層の追加

ほとんどの階層は物理レイヤーにインポートされます。インポートされていない階層に関連付けられた列はインポートされません。インポートされていない列にユーザーがアクセスする必要がある場合は、まずそれらの列を物理レイヤーに追加してから、階層のレベルに関連付けます。この項では、階層を追加する手順について説明します。

階層の各レベルにはレベル・キーがあります。階層のレベルに関連付けられる（追加される）最初のキューブ列がレベル・キーです。これは、キューブのデータソース定義と一致している必要があります。データソース・キューブ・テーブルは、1 つの列をレベル・キーとして設定できないため、Oracle BI の物理レイヤー・テーブルではレベル・キーとして別の列を設定します。最初に選択した列のアイコンは、階層のレベルと関連付けられた後でキーのアイコンに変更されます。

階層に追加する列を選択するときは、最上位レベルから階層の順序に従って選択することをお薦めします。複数の列を選択し、それらを同時に階層に取り込んだ場合、選択した列のグループの順序がそのまま保持されます。階層に列を追加した後、「Browse」ダイアログ・ボックスで列の順序を変更できます。

クエリーが階層のメンバーを明示的に参照していない場合は、デフォルトのメンバーが使用されます。したがって、あらゆる階層をデフォルトのメンバー（通常は ALL メンバー）に関連付けておく必要があります。「Hierarchy」ダイアログ・ボックスに含まれているチェック・ボックス（「Default member type ALL」）は、デフォルトとして ALL メンバーを指定するときに使用します。このチェック・ボックスの選択に関するガイドラインを、次にいくつか示します。

- キューブをインポートした場合、「Default member type ALL」チェック・ボックスが自動的に選択されます。インポート時に ALL メンバーが指定されます。
- 階層を手動で作成した場合、このチェック・ボックスは自動的には選択されません。チェック・ボックスを選択する前に、マルチディメンショナル・データソースの管理者に依頼して、ALL 以外のメンバーがデフォルトとして定義されていないか確認してください。たとえば、Year レベルで、1997 がデフォルトのメンバーとして指定されている可能性があります。この場合、「Default member type ALL」チェック・ボックスを選択しないでください。

物理キューブ・テーブルに階層を追加するには

1. Administration Tool の「Physical」レイヤーで、階層を追加するテーブルをダブルクリックします。
2. 「Physical Cube Table」ダイアログ・ボックスで「Hierarchies」タブをクリックし、「Add」をクリックします。
3. 97 ページの表 14 を参照し、「Hierarchy」ダイアログ・ボックスで、フィールドに値を入力します。
4. レベルを作成するには、次の手順を実行します。
   a. 「Hierarchy」ダイアログ・ボックスで「Add」をクリックします。
   b. 97 ページの表 14 を参照し、「Physical Level 」ダイアログ・ボックスで、フィールドに値を入力します。

   注意：階層のレベルはトップ・ダウンで追加してください（順序は後で変更できます）。正しい階層順序を使用しておくと、クエリーで正確な情報を返すことができ、エラーを回避できます。
5. レベルに 1 つ以上の列を追加するには、「Physical Level」ダイアログ・ボックスで「Add」をクリックします。

   注意：物理レベルへの列の追加は、レベル・オブジェクトの物理列をドラッグ・アンド・ドロップして行うこともできます。最初に追加する列がキーになります。それ以降の列はプロパティになります。
6. 「Browse」ダイアログ・ボックスで次の手順を実行します。
   a. 「Name」リストで、階層に追加する列を検索します。
   b. まずキー列を選択してから「Select をクリックします。
   c. 「Physical Level」ダイアログ・ボックスで「OK」をクリックします。
7. 列をさらに追加するには、96 ページの手順 5 から 96 ページの手順 6 を繰り返します。

   注意：キーボードの [Ctrl] キーを押しながら各列をクリックし、「Select」をクリックすると、複数の列を追加できます。

8 列の追加が終了したら、「Hierarchy」ダイアログ・ボックスで「OK」をクリックします。

表 14. 物理キューブ・テーブルの階層およびレベルのプロパティ

プロパティ	説明
Default member type ALL	ALL メンバーをデフォルトとして指定するときに使用するチェック・ボックスです。ALL 以外のメンバーをデフォルトとして選択する場合は使用しないでください。
Dimension Name	（ディメンションの一意の名前）階層が属するディメンションです。
External Name	オブジェクトの完全修飾名です。
Level Number	階層レベルの順序を特定します。このプロパティは、レベルの順序を変更するときに使用します。
Time Dimension	年、日、四半期など、ディメンションを一定期間として指定するチェック・ボックスです。
Type	階層のタイプ：「Fully Balanced」、「Unbalanced」、「Ragged Balanced」または「Network」。

階層レベルの確認

複数のレベルを含む階層を設定した後で、階層レベルの順序を確認することを強くお薦めします。

階層レベルを確認するには

1 Administration Tool の「Physical」レイヤーで、確認するテーブルをダブルクリックします。

2 「Physical Cube Table」ダイアログ・ボックスで「Hierarchies」タブをクリックします。

3 「Hierarchies」タブで階層を選択して、「Edit」をクリックします。

4 「Hierarchy」ダイアログ・ボックスで、レベルが正しいことを確認します。

「Hierarchy」ダイアログ・ボックスには、選択した階層に対して定義されたレベルがすべて一覧表示されます。階層で最上位のレベルが、リストの最初（最上位）の項目です。

5 階層レベルを並べ替える場合は、レベルを選択し、「Up」または「Down」をクリックしてレベルの順序を変更します。

レベルは複数あり、有効となるボタンのレベルを選択する必要があります。

6 正しいレベルに並べ替えたら、「OK」をクリックします。

7 「Physical Cube Table」ダイアログ・ボックスで「OK」をクリックします。

メンバー・カウントの更新

メンバー・カウントを更新するには、オンライン・モードでリポジトリを開く必要があります。

カウントを更新する必要があるかどうかを判断するには、階層名またはレベル名までマウスを移動します。カウントを更新する必要があること、または最後に更新された日時を知らせるメッセージが表示されます。

メンバー・カウントを更新すると、選択した階層から、現在のメンバー・カウントが返されます。メンバー・カウントの更新が正常に終了した後は、階層名またはレベル名までマウスを移動すると、更新後のメンバー・カウントがメッセージに表示されます。メッセージは次の構文で表示されます。

```
<hierarchy name> (<x> members, last updated <time stamp>)
```

メンバー・カウントを更新するには

1. Administration Tool の「Physical」レイヤーで、階層またはレベルにカーソルを移動します。

   カウントを更新する必要がある場合は、メッセージが表示されます。

2. 1 つ以上の階層およびレベルを右クリックします。
3. メニューの「Update Member Count」を選択します。

   更新が正常に終了すると、更新されたメッセージが表示されます。

物理キューブ・テーブルのメンバーの表示

メンバーを表示するには、リポジトリをオンライン・モードで開く必要があります。これは、Analysis Services および SAP/BW のデータソースの物理キューブ・テーブルに対して使用できます。

リポジトリの物理レイヤーに、階層またはレベルのメンバーを表示できます。メンバーが階層内のレベル別に一覧表示されるため、サーバーの XMLA 接続が正しく設定されたかどうかを判断するのに役立ちます。データを返すまでの時間を短縮する場合は「Starting from」オプションで開始ポイントを指定し、返されるデータのサイズを縮小する場合は「Show」オプションで返す行の数を指定します。

メンバーを表示するには

1. Administration Tool の「Physical」レイヤーで、階層またはレベルを右クリックします。
2. 「View Members」を選択します。

   ウィンドウが開き、階層のメンバー数とレベルのリストが表示されます。返されたデータをすべて表示するには、ウィンドウおよび列を拡大してください。

3. 「Query」をクリックすると、結果が表示されます。
4. 終了したら、「閉じる」をクリックします。

既存の階層でのキューブ列の追加または削除

階層の設定後、場合によっては列を追加または削除する必要があります。階層が誤って作成され、やりなおしたい場合は削除も可能です。

階層からキューブ列を削除した場合、その列は階層からは削除されますが、キューブ・テーブルには残るため、他のレベルに追加するために使用することができます。

既存の階層でキューブ列を追加または削除するには

1. Administration Tool の「Physical」レイヤーで、変更するテーブルをダブルクリックします。

2 「Physical Cube Table」ダイアログ・ボックスで「Hierarchies」タブをクリックします。
3 変更する階層を選択してから「Edit」をクリックします。
4 「Hierarchy」ダイアログ・ボックスでレベルを選択し、「Edit」をクリックします。
5 「Physical Level」ダイアログ・ボックスで、次のいずれかの手順を実行します。
   a 列を追加するには「Add」をクリックします。
      ❑ 「Browse」ダイアログ・ボックスの「Name」リストで、追加する列を選択します。
      ❑ 「Select」をクリックします。
   b 列を削除するには、削除する列を選択して「Remove」をクリックします。
   c 階層レベルの順序を変更するには、変更するレベルを選択して、「Up」または「Down」をクリックします。
   d 「OK」をクリックします。
6 「Hierarchy」ダイアログ・ボックスで「OK」をクリックします。
7 「Physical Cube Table」ダイアログ・ボックスで「OK」をクリックします。

物理キューブ・テーブルからの階層の削除

階層が誤って作成され、やりなおしたい場合や、使用されていないオブジェクトを削除したい場合は削除も可能です。たとえば、物理的なマルチディメンショナル・スキーマ全体をインポートして、ビジネス・モデルで一部のみを使用する場合があります。

注意：「Physical」レイヤーで階層を削除すると、その階層およびその階層の一部である列が削除されます。

物理キューブ・テーブルから階層を削除するには

1 Administration Tool の「Physical」レイヤーで、変更するテーブルをダブルクリックします。
2 「Physical Cube Table」ダイアログ・ボックスで「Hierarchies」タブをクリックします。
3 削除する階層を選択してから「Remove」をクリックします。

物理キューブ列と階層レベルの関連付け

物理レイヤーで属性を使用すると、特定の階層レベルにのみ存在する列を表すことができます。たとえば、Population が Geography 階層の State レベルに関連付けられている属性だとします。このとき、Population をクエリーすると、この階層の State レベルのデータが暗黙的に要求されます。

1 つのレベルには 0 以上の属性を関連付けることができます。レベルに最初に関連付けられた物理キューブ列がレベル・キーになります。後続の列をレベルに関連付けると、その列はレベル・キーではなく属性になります。

物理キューブ列と階層の関連付けの例

State というレベルがあり、このレベルに Population という列を関連付ける場合を考えます。

■ 階層と State レベルを作成します。

- Population の物理キューブ列を作成します。
- 「Physical Cube Table」ダイアログ・ボックスの「Hierarchies」タブで、State レベルを選択して「Edit」をクリックします。
- 「Hierarchy」ダイアログ・ボックスで「Add」をクリックします。
- 「Physical Level」ダイアログ・ボックスで「Add」をクリックします。
- 「Browse」ダイアログ・ボックスで Population 列を選択し、「Select」をクリックします。

  メジャーのアイコンがプロパティのアイコンに変わります。

XML データソースの物理テーブルのプロパティ設定

「XML」タブを使用して、XML データソースのプロパティを設定または編集します。「Physical Table」ダイアログ・ボックスの「XML」タブには、「Connection Pool」ダイアログ・ボックスの「XML」タブと同じ機能があります。ただし、「Physical Table」ダイアログ・ボックスでプロパティを設定すると、「Connection Pool」ダイアログ・ボックスで対応する設定より優先されます。詳細は、「XML データソースに対する接続プールの追加プロパティの設定」（80 ページ）を参照してください。

「Physical」レイヤーのフォルダの作成

この項の内容は次のとおりです。

「Physical」レイヤーのカタログおよびスキーマの作成

複数のスキーマのグループ化には、カタログという手段を使用することもできます。カタログには、データベース・オブジェクトのスキーマ（メタデータ）がすべて含まれています。スキーマには、特定のユーザーまたはアプリケーションのメタデータ情報のみが含まれています。データベースを構成した後で物理レイヤーをモデル化します。

データベースには、カタログとスキーマのいずれかを含めることはできますが、両方を含めることはできません。データベースに 1 つ以上のスキーマが含まれていると、カタログを作成できません。データベースに 1 つ以上のカタログが含まれていると、スキーマを作成できません。

注意：カタログ・オブジェクトまたはスキーマ・オブジェクトを作成する前に、データベース・オブジェクトを作成する必要があります。

カタログの作成

Oracle BI の管理者は、大きなリポジトリの「Physical」レイヤーに、1 つ以上の物理スキーマを含むカタログを作成できます。

カタログを作成するには

1. 「Physical」レイヤーでデータベース・オブジェクトを右クリックし、「New Object」→「Physical Catalog」を選択します。
2. 「Physical Catalog」ダイアログ・ボックスで、カタログの名前を入力します。
3. カタログの説明を入力して「OK」をクリックします。

スキーマの作成

スキーマ・オブジェクトには、物理スキーマのテーブルおよび列が含まれています。Administration Tool の「Physical」レイヤーでは、スキーマ・オブジェクトはオプションです。

スキーマを作成するには

1. 「Physical」レイヤーでデータベース・オブジェクトを右クリックし、「New Object」→「Physical Schema」を選択します。
2. 「Physical Schema」ダイアログ・ボックスで名前を入力します。
3. スキーマの説明を入力して「OK」をクリックします。

カタログまたはスキーマの名前を指定する変数の使用

変数を使用して、カタログまたはスキーマのオブジェクトの名前を指定できます。たとえば、複数のクライアントのデータがあり、各クライアントのデータを別々のカタログに含めるようにデータベースを構成したとします。ここで、Client などの名前が付いたセッション変数を初期化すると、ユーザーが Oracle BI Server にサイン・オンしたとき、そのカタログ・オブジェクトの名前を動的に設定する際にこの変数を使用できます。

注意：「Dynamic Name」タブは、1 つ以上の変数が定義されていないとアクティブになりません。

使用するセッション変数を「Dynamic Name」タブで指定するには

1. 「Dynamic Name」タブの「Name」列で、使用するセッション変数の名前をクリックします。この変数の初期値（存在する場合）が「Default Initializer」列に表示されます。
2. 強調表示されている変数を選択するには、「Select」をクリックします。

   動的な名前のフィールドに変数名が表示され、「Select」ボタンが「Clear」ボタンに切り替わります。

セッション変数の割当てを「Dynamic Name タブで削除するには

- 「Clear」ボタンをクリックすると、変数に対する動的な名前の割当てが削除されます。

  動的な名前のフィールドに「Not Assigned」という値が表示され、「Clear」ボタンが「Select」ボタンに切り替わります。

列のエントリを「Dynamic Name」タブでソートするには

- 関連付けた列見出し（「Name」または「Default Initializer」）をクリックすることにより、列のエントリをソートできます。列見出しをクリックすると、列のタイプに応じて、その列のエントリ順序が昇順または降順に切り替わります。

  動的な名前が割り当てられていないと、「Select」ボタンの左側にある動的な名前のフィールドには「Not Assigned」と表示されます。動的な名前が割り当てられていると、「Select」ボタンが「Clear」ボタンに切り替わり、動的な名前のフィールドには変数名が表示されます。

「Physical」レイヤーの表示フォルダの設定

Oracle BI の管理者は、表示フォルダを作成して「Physical」レイヤーでテーブル・オブジェクトを整理することができます。メタデータ的な意味はありません。表示フォルダの作成後は、選択したテーブルが、フォルダにはショートカットとして、「Physical」レイヤー・ツリーにはオブジェクトとしてそれぞれ表示されます。表示フォルダにはショートカットのみが表示されるように、オブジェクトを非表示にできます。これらのオブジェクトを非表示にする方法の詳細は、「「Options」ダイアログ・ボックスにおける「Repository」タブの使用」（33 ページ）を参照してください。

注意：オブジェクトを表示フォルダで削除しても、削除されるのはそのオブジェクトのショートカットのみです。

物理表示フォルダを設定するには

1. 「Physical」レイヤーでデータベース・オブジェクトを右クリックし、「New Object」→「Physical Display Folder」を選択します。
2. 「Physical Display Folder」ダイアログ・ボックスの「Tables」タブで、フォルダの名前を入力します。
3. テーブルを表示フォルダに追加するには、次の手順を実行します。
   a. 「Add」をクリックします。
   b. 「Browse」ダイアログ・ボックスで、フォルダに追加するファクト・テーブルまたは物理テーブルを選択してから、「Select」をクリックします。
4. 「OK」をクリックします。

物理結合について

有効な物理結合はすべて、Administration Tool の「Physical」レイヤーで構成する必要があります。

注意：結合は、マルチディメンショナル・データソースには作成しません。

物理スキーマにキーをインポートすると、主キーと外部キーの結合が自動的に定義されます。これ以外の各データベース内の結合、またはデータベース間の結合は、物理レイヤーのテーブル間の関係を表現するように明示的に定義する必要があります。

注意：インポートしたキーと外部キーの結合をメタデータで使用する必要はありません。参照整合性制約を強制するように定義された結合は、クエリーで指定されている結合では不正となることがあります。たとえば、多目的の検索テーブルと複数の他のテーブル間の結合は、Oracle BI Server で発行される SQL では、不要または不正な循環結合になることがあります。

マルチデータベースの結合

マルチデータベース結合は、別のメタデータ・データベース・オブジェクトの下のテーブルと結合している 1 つのメタデータ・データベース・オブジェクトの下のテーブルとして定義します。複数のデータベースのデータを結合するには、マルチデータベース結合を指定する必要があります。「Physical Table Diagram」ウィンドウを編集して、マルチデータベース結合を指定します。結合は、データベース型に関係なく、どのデータベースのテーブル間でも実現でき、Oracle BI Server 内で実行されます。Oracle BI Server にはマルチデータベース結合のパフォーマンスを最適化するための方針がいくつかありますが、マルチデータベース結合は、同一データベース内のテーブル間の結合と比べ、速度が大幅に低下します。マルチデータベース結合はできるだけ避けることをお薦めします。Physical Table Diagram の詳細は、「Physical Diagram での物理結合の定義」（105 ページ）を参照してください。

フラグメント化されたデータ

フラグメント化されたデータとは、複数のテーブルに分割されている、シングル・ドメインのデータです。たとえば、姓が A から M で始まる顧客のデータを 1 つのテーブル、N から Z で始まる顧客を別のテーブルにまとめている売上データをデータベースに格納しているとします。フラグメント化されたテーブルを使用する場合、各フラグメントとそれが関連するすべてのテーブルとの間の結合条件をすべて定義する必要があります。103 ページの図 9 では、フラグメント化された売上テーブルとフラグメント化された顧客テーブルの物理結合を示します。どちらのテーブルも同じ方法でフラグメント化されています（A から M と N から Z）。

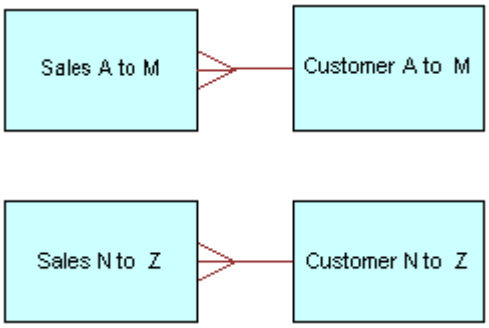


図 9.　フラグメント化されたテーブルの例

フラグメント化されたファクト・テーブルとフラグメント化されたディメンショナル・テーブルがあり、フラグメントが複数の値にわたる場合があります。この場合は、104 ページの図 10 に示すように、有効な結合をすべて定義する必要があります。

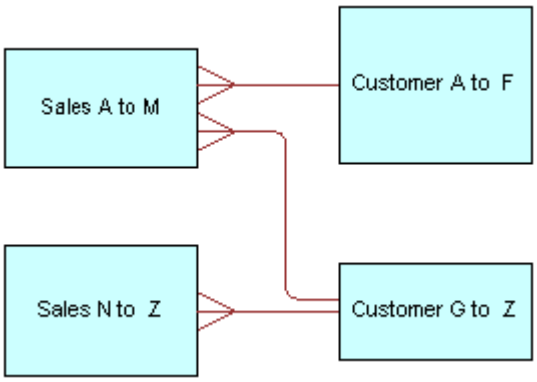


図 10. フラグメント化されたテーブルの結合の例

ヒント： 不要な結合条件の追加は避けてください（たとえば、103 ページの図 9 の「Sales A to M」と「Customer N to Z」）。余分な結合条件がパフォーマンス低下の原因となることがあります。

主キーと外部キーのリレーションシップ

主キーと外部キーのリレーションシップでは、2 つのテーブルの間で 1 対多関係が定義されます。外部キーとは、あるテーブルにおいて別のテーブルにある主キー列を参照する 1 つ以上の列のことです。主キーは、一意の値によってテーブルにおいて 1 つの行を識別する列または一連の列として定義されます。主キーと外部キーは、Physical Table Diagram で指定することも、「Physical Table」ダイアログ・ボックスの「Keys」タブか「Foreign Keys」タブを使用して指定することもできます。「Physical Diagram での物理結合の定義」（105 ページ）および「物理テーブルでの列およびキーの作成と管理」（91 ページ）も参照してください。

複合結合

リポジトリの物理レイヤーでは、複合結合とは、非外部キーと主キーの列の結合です。物理レイヤーで複合結合を作成する場合は、式を指定することや、結合を作成する対象となる列を指定することができます。ビジネス・モデル・レイヤーで複合結合を作成する場合は、式を指定しません。

物理外部キーおよび結合の定義

物理外部キー、複合結合および論理結合を作成するには、Joins Manager、Physical Table Diagram または Logical Table Diagram を使用できます。

注意： 結合は、マルチディメンショナル・データ ソースには作成しません。

物理結合を定義するには、次の項を参照してください。

- Joins Manager による物理外部キーまたは複合結合の定義（105 ページ）
- Physical Diagram での物理結合の定義（105 ページ）

Joins Manager による物理外部キーまたは複合結合の定義

Joins Manager を使用すると、結合関係を確認することや、物理外部キーと複合結合を作成することができます。

物理外部キーまたは複合結合を作成するには

1. Administration Tool のツールバーで、「Manage」→「Joins」を選択します。
2. 「Joins Manager」ダイアログ・ボックスで、次のいずれかの手順を実行します。
   - 「Action」→「New」→「Complex Join」を選択します。

     「Physical Complex Join」ダイアログ・ボックスが表示されます。
   - 「Action」→「New」→「Foreign Key」を選択します。「Browse」ダイアログ・ボックスで、テーブルをダブルクリックします。
3. 「Physical Foreign Key」ダイアログ・ボックスで、外部キーの名前を入力します。
4. ダイアログ・ボックスの左側にある「Table」フィールドの「Browse」ボタンをクリックし、外部キーによって参照されるテーブルを検索します。
5. キーが参照する列を左側のテーブルで選択します。
6. 右側のテーブルにおいて外部キー列で構成される列を選択します。
7. 必要に応じて、データベース・ヒントを指定します。

   詳細は、「データベース・ヒントの使用」（110 ページ）を参照してください。
8. 式ビルダーを開くには、「Expression」ペインの横にあるボタンをクリックします。

   式が「Expression」ペインに表示されます。
9. 「OK」をクリックして、作業内容を保存します。

Physical Diagram での物理結合の定義

外部キー、およびテーブル間の複合結合を定義できます。テーブルは、同一データベースのものであってもなくてもかまいません。ツールバーの Physical Diagram のアイコンをクリックすると、「Physical Table Diagram」ウィンドウが開き、選択したオブジェクトのみが表示されます。物理オブジェクトを右クリックすると、複数のオプションが使用できます。これらのオプションの詳細は、106 ページの表 15 を参照してください。

Physical Table Diagram を表示するには

1. Administration Tool の「Physical」レイヤーでテーブルを右クリックして、「Physical Diagram」を選択します。
2. ショートカット・メニューで、106 ページの表 15 のオプションを選択します。
3. 別のテーブルを「Physical Table Diagram」ウィンドウに追加するには、次の手順を実行します。
   a. 「Physical Table Diagram」ウィンドウは、開いたままにしておきます。

b 右クリックして、追加するテーブルを選択し、106 ページの表 15 に示された、Physical Diagram のいずれかのオプションを選択します。

必要なテーブルがすべて「Physical Table Diagram」ウィンドウに表示されるまで、この手順を繰り返します。

表 15. Physical Diagram のショートカット・メニューのオプション

Physical Diagram のメニュー	説明
Object(s) and all joins	選択したオブジェクト、および、結合パスによって選択されたオブジェクトと直接的または間接的に関連付けられている各オブジェクトが表示されます。スキーマ内のオブジェクトがすべて関連付けられている場合は、1 つのテーブルしか選択していない場合でも、このオプションによって、あらゆるテーブルがダイアグラム化されます。
Object(s) and direct joins	選択したオブジェクト、および選択したオブジェクトと結合しているテーブルが表示されます。
Object(s) and direct joins in the business model	このオプションは、現時点では使用できません。
Selected object(s) only	選択したオブジェクトのみが表示されます。結合は、選択したオブジェクト間に存在している場合にかぎり表示されます。

外部キー結合または複合結合を定義するには

1 Administration Tool の「Physical」レイヤーで、1 つ以上のテーブルを選択し、右クリック・メニューから Physical Diagram のいずれかのコマンドを実行します。

2 Administration Tool のツールバーにある次に示すアイコンのいずれかをクリックします。

- New foreign key
- New complex join

3 このアイコンを選択した状態で、「Physical Table Diagram」ウィンドウで、結合の最初のテーブル（1 対多の結合で「1」となっているテーブル）を左クリックして選択します。

4 結合するテーブル（1 対多の結合で「多」となっているテーブル）にカーソルを移動し、2 番目のテーブルを左クリックして選択します。

「Physical Foreign Key」ダイアログ・ボックスまたは「Physical Join」ダイアログ・ボックスが表示されます。

5 結合列を左右のテーブルで選択します。

SQL の結合条件が、ウィンドウの式ペインに表示されます。

注意： このウィンドウには駆動テーブルが表示されますが、選択できません。これは、Oracle BI Server では、「Business Model and Mapping」レイヤーにのみ駆動テーブルが実装されているためです。駆動テーブルの詳細は、「駆動テーブルの指定」（144 ページ）を参照してください。

6 必要に応じて、データベース・ヒントを指定します。

詳細は、「データベース・ヒントの使用」（110 ページ）を参照してください。

7 式ビルダーを開くには、「Expression」ペインの横にあるボタンをクリックします。

式が「Expression」ペインに表示されます。

8 「OK」をクリックすると、選択内容が適用されます。

不明瞭ビューのデプロイ

この項の内容は次のとおりです。

不明瞭ビューのデプロイについて

不明瞭ビューとは、SELECT 文で構成される物理レイヤー・テーブルです。新規テーブルが必要な場合は、物理テーブルまたはマテリアライズド・ビューを作成する必要があります。不明瞭ビューは、他の方法がない場合にのみ使用する必要があります。

不明瞭ビューは、リポジトリでは物理データベースのビュー・テーブルとして表示されますが、実際に存在するわけではありません。不明瞭ビューは、「Deploy View(s)」ユーティリティを使用して物理データベースにデプロイします。不明瞭ビューは、デプロイ後にはデプロイ済のビューと呼ばれます。不明瞭ビューは、デプロイしないで使用できますが、Oracle BI Server では、不明瞭ビューが出現したときには、より複雑なクエリーを生成する必要があります。

注意： XLS や非リレーショナル・データベースなどのデータベースでは、不明瞭ビューをサポートしていないため、このビューをデプロイするためのユーティリティを使用できません。

不明瞭ビューがデータベースでサポートされているかどうかを確認するには、「Database」ダイアログ・ボックスの「Features」タブで、CREATE_VIEW_SUPPORTED SQL 機能を選択してください。手順については、「データベースでサポートされている SQL 機能の指定」（68 ページ）を参照してください。

不明瞭ビューのオブジェクトのデプロイ

オフライン・モードでは、ODBC および DB2cli のデータソースを使用してデータベースからインポートしたときに、「Deploy View(s)」ユーティリティを使用できるようになります。Oracle ネイティブ（クライアント）ドライバも、オフライン・モードでビューをデプロイできます。オンライン・モードでは、サーバー経由でインポートしたデータベース（クライアント側の設定は無視されます）がサポートされていればビューをデプロイできます。

CREATE VIEW SELECT 文

不明瞭ビューをサポートしているデータベースでは、リポジトリの物理レイヤーに不明瞭ビューをデプロイする SQL 文を使用できます。どのデータベースが不明瞭ビューをサポートしているかを判断するには、システム管理者に問い合せるか、データベースのドキュメントを参照してください。

定義に使用できるのは、リポジトリ変数のみです。ビュー定義でセッション変数を使用するとエラーになります。

構文：

```
CREATE VIEW <view name> AS <select statement>,
```

内容は次のとおりです。

<select statement>	不明瞭ビューのオブジェクトでユーザーが入力した SQL です。SQL が無効の場合、CREATE VIEW 文は、ビューをデプロイしたときに失敗します。
<view name>	schema.viewname と viewname の 2 つのフォーマットがあります。接続プールの設定により、スキーマ名を追加するかどうかが決まります。

不明瞭ビューのオブジェクトの右クリック・メニューには、「Deploy View(s)」オプションが含まれています。「Deploy View(s)」を選択すると、CREATE VIEW SQL 文が実行され、デプロイ済ビューのオブジェクトの作成が試行されます。次に、ビューのデプロイを開始する方法および各方法の結果について説明します。

- 不明瞭ビューの 1 つのオブジェクトを右クリックします。「Deploy View(s)」を選択すると、CREATE VIEW SQL 文が実行され、オブジェクトのデプロイ済ビューの作成が試行されます。
- 複数のオブジェクトを右クリックします。選択したオブジェクトに 1 つでも不明瞭ビュー・オブジェクトが含まれている場合は、右クリック・メニューに「Deploy View(s)」オプションが含まれています。「Deploy View(s)」を選択すると、CREATE VIEW SQL 文が実行され、該当するオブジェクトのデプロイ済ビューの作成が試行されます。
- 物理スキーマまたは物理カタログを右クリックします。スキーマまたはカタログに不明瞭ビューのオブジェクトが含まれている場合、右クリック・メニューには、「Deploy View(s)」オプションが含まれています。「Deploy View(s)」を選択すると、該当するすべてのオブジェクトに対して CREATE VIEW SQL 文が実行され、選択したスキーマまたはカタログに含まれている該当オブジェクトのデプロイ済ビューの作成が試行されます。

デプロイ中に、ビューに名前が割り当てられます。事前に割り当てられた名前を変更する場合、新しい名前は 18 文字以下の英数字にする必要があります。これらのガイドラインが守られないと、オブジェクト名は、次の名前変換アルゴリズムを使用して、有効な名前に自動変換されます。

1. 英数字以外の文字はすべて削除します。
2. 108 ページの手順 1 後に 16 文字以上ある場合は、最初の 16 文字を残します。
3. 対応するコンテキストにおいて名前が一意になるように、00 から 99 で始まる 2 桁が名前に追加されます。

デプロイ・プロセスが完了すると、次の結果となります。

- デプロイが正常に行われたビューと正常に行われなかったビューがリストに表示されます。
- デプロイが正常に行われなかったものについては、簡潔な原因がリストに表示されます。

- 正しくデプロイされると、不明瞭ビューのオブジェクト・タイプが Select から None に変更され、デプロイ済ビューは通常のテーブルとして扱われます。

  注意：タイプを Select に戻すと、関連する不明瞭ビューがデータベースから削除されるか、エラー・メッセージが表示されます。デプロイ済ビューの削除方法の詳細は、「不明瞭ビューまたはデプロイ済ビューを削除するためのガイドライン」（110 ページ）を参照してください。

- Administration Tool では、正しくデプロイされたビューのアイコンがデプロイ済ビューのアイコンに変わります。

不明瞭ビューをデプロイするには

1. リポジトリの「Physical」レイヤーで、デプロイする不明瞭ビューを右クリックします。
2. 右クリック・メニューで「Deploy View(s)」を選択します。
3. 「View Deployment - Deploy View(s)」ダイアログ・ボックスで、次の手順を実行します。
   a. 「New Table Name」列で、必要に応じて、新しくデプロイしたビューの名前を変更します。

      変更した名前が命名ルールに準拠していない場合は、新しい名前が割り当てられ、再度ダイアログ・ボックスが表示されるので、それを受け入れるか変更します。すべての名前がチェックをパスするまで、この操作が繰り返されます。

   b. この時点でデプロイしておきたくないビューがある場合は、該当する行でチェック・ボックスの選択を解除します。
4. データベースに定義されている接続プールが複数ある場合は、「Select Connection Pool」ダイアログ・ボックスで 1 つの接続プールを選択して、「Select」をクリックします。

   SQL 文（CREATE VIEW）が実行され、「View Deployment Messages」ダイアログ・ボックスが表示されます。
5. 「View Deployment Messages」ダイアログ・ボックスでは、「Find」や「Find Again」を使用してビューを検索したり、内容をコピーできます。
6. 目的の作業を実行したら、「OK」をクリックします。

デプロイ済ビューのアンデプロイ

デプロイ済ビューに対して「Undeploy View(s)」ユーティリティを実行すると、ビューが削除され、元の SELECT 文を使用した不明瞭ビューにテーブルが戻ります。

デプロイ済ビューをアンデプロイするには

1. リポジトリの「Physical」レイヤーで、データベース、物理カタログ、スキーマまたは物理テーブルを右クリックします。

   選択したオブジェクトに関連するデプロイ済ビューがある場合、右クリック・メニューには、「Undeploy View(s)」オプションが含まれています。
2. 「Undeploy View(s)」を選択します。

   アンデプロイするビューのリストが表示されます。

3 この時点でアンデプロイしておきたくないビューがある場合は、該当する行でチェック・ボックスの選択を解除します。

4 「View Deployment - Undeploy View(s)」ダイアログ・ボックスで、「OK」をクリックしてビューを削除します。

アンデプロイが正常に終了すると、メッセージが表示されます。

5 「View Deployment Messages」ダイアログ・ボックスでは、「Find」や「Find Again」を使用してアンデプロイ済ビューを検索したり、内容をコピーできます。

6 目的の作業を実行したら、「OK」をクリックします。

不明瞭ビューまたはデプロイ済ビューを削除するためのガイドライン

リポジトリで不明瞭ビューまたはデプロイ済ビューを削除するには、次のガイドラインに従ってください。

- **アンデプロイされた不明瞭ビューをリポジトリから削除する**：不明瞭ビューがデプロイされていない場合は、リポジトリからそのビューを削除できます。
- **デプロイ済ビューを削除する**：不明瞭ビューをデプロイするときには、データベースのビュー・テーブルが、バックエンド・データベースおよびリポジトリに物理的に作成されます。したがって、ビューは削除前にアンデプロイする必要があります。Administration Tool の「Undeploy View(s)」ユーティリティを使用します。これによって、不明瞭ビューがバックエンド・データベースから削除され、オブジェクト・タイプが None から Select に変更され、リポジトリの物理レイヤーにあるオブジェクトの SELECT 文がリストアされます。

  警告：バックエンド・データベースにある物理データベース・ビューは削除しないでください。削除してしまうと、Oracle BI Server でビューのオブジェクトをクエリーできなくなります。ビューをアンデプロイすると、バックエンド・データベースから自動的に削除されます。

不明瞭ビューを再デプロイするためのガイドライン

不明瞭ビューは、削除後に再デプロイすることもできます。Administration Tool では、1 回目のデプロイと再デプロイが区別されません。不明瞭ビューを再デプロイする前に、デプロイ済ビューを必ず削除してください。これを行わないと、デプロイ処理が失敗し、データベースからエラー・メッセージが返されます。

データベース・ヒントの使用

データベース・ヒントとは、SQL 文内に置く指示のことで、文を実行する最も効率的な方法をデータベース・クエリー・オプティマイザに伝えます。ヒントはオプティマイザの実行計画よりも優先されるため、ヒントを使用すると、より効率的な計画の実行をオプティマイザに強制することにより、パフォーマンスを向上させることができます。

注意：ヒントはデータベースごとに異なります。Oracle BI Server では、Oracle 8*i*、9*i*、10*g* の各サーバーのヒントのみをサポートしています。

Administration Tool を使用すると、オンライン・モードでもオフライン・モードでも、ヒントをリポジトリに追加し、クエリーのパフォーマンスを最適化できます。リポジトリにヒントを追加する際に、ヒントをデータベース・オブジェクトに関連付けます。ヒントに関連付けられているオブジェクトがクエリーされると、Oracle BI Server ではヒントが SQL 文に挿入されます。

ヒントを関連付けられるデータベース・オブジェクトを 111 ページの表 16 に示します。そこでは、データベース・オブジェクトに対応する Administration Tool のダイアログ・ボックスも示します。これらのダイアログ・ボックスにはそれぞれ「Hint」フィールドが含まれています。リポジトリに追加するヒントは、このフィールドに入力できます。

表 16. ヒントを受け入れるデータベース・オブジェクト

データベース・オブジェクト	ダイアログ・ボックス
物理複合結合	Physical Join - Complex Join
物理外部キー	Physical Foreign Key
オブジェクト・タイプが Alias の物理テーブル	「Physical Table」の「General」タブ
オブジェクト・タイプが None の物理テーブル	「Physical Table」の「General」タブ

使用例

この項では、Oracle ヒントを Oracle BI Server と組み合せて使用する方法について、例をいくつか紹介します。Oracle ヒントの詳細は、ご使用の Oracle サーバーのバージョンに対応した Oracle の SQL リファレンスを参照してください。

Index ヒント

Index ヒントでは、テーブルではなく指定した索引をスキャンするようにオプティマイザに指示します。次の架空の例を使用して、Index ヒントの使用法を説明します。ORDER_ITEMS テーブルに対するクエリーの速度が低下していることに気づきました。クエリー・オプティマイザの実行計画を確認したところ、FAST_INDEX 索引が使用されていないことがわかりました。Index ヒントを作成して、オプティマイザが ORDER_ITEMS テーブルではなく FAST_INDEX 索引を強制的にスキャンするように設定しました。Index ヒントの構文は *index(table_name, index_name)* です。このヒントをリポジトリに追加するには、Administration Tool の「Physical Table」ダイアログ・ボックスにナビゲートし、「Hint」フィールドに次のテキストを入力します。

```
index(ORDER_ITEMS, FAST_INDEX)
```

Leading ヒント

Leading ヒントを使用すると、指定したテーブルを使用してクエリーの結合順序を作成するようにオプティマイザに強制します。Leading ヒントの構文は *leading(table_name)* です。Products テーブルと Sales Fact テーブルの間に外部キーの結合を作成し、オプティマイザで強制的に Products テーブルで結合を開始させる場合は、Administration Tool の「Physical Foreign Key」ダイアログ・ボックスにナビゲートし、「Hint」フィールドに次のテキストを入力します。

```
leading(Products)
```

パフォーマンスに関する考慮事項

しっかりと検討および計画されたヒントを使用すると、クエリーのパフォーマンスが大幅に向上することがあります。ただし、ヒントの実行計画が不十分な場合は、パフォーマンスに悪い影響を与えることもあります。クエリーのパフォーマンスを最適化するためのヒントを作成するときには、次のガイドラインに従ってください。

- ヒントは、次の方法でパフォーマンスの向上を試みた後でのみ、リポジトリに追加するようにしてください。
  - 物理的な索引（などの物理的な変更）を Oracle データベースに追加する。
  - サーバー内のモデリングを変更する。
- 頻繁にクエリーされる物理テーブルおよび結合オブジェクトには、ヒントを作成しないでください。

注意：ヒントに関連付けられた物理オブジェクトを削除または名前変更した場合は、それに従ってヒントも変更する必要があります。

ヒントの作成

次に、Administration Tool を使用してリポジトリにヒントを追加する手順について説明します。

ヒントを作成するには

1. 次のいずれかのダイアログ・ボックスにナビゲートします。
   - 「Physical Table」の「General」タブ
   - 「Physical Foreign Key」
   - 「Physical Join」の「Complex Join」
2. 「Hint」フィールドにヒントのテキストを入力し、「OK」をクリックします。

   使用可能な Oracle ヒントおよびヒントの構文の説明については、Oracle8*i* の SQL リファレンスを参照してください。

   注意：SQL のコメント・マーカー（/* または --）を使用してヒントを特定することができますが、ヒントのテキストを入力するときには、SQL のコメント・マーカーは入力しないでください。ヒントの実行時に、Oracle BI Server によってコメント・マーカーが挿入されます。

5 Oracle BI リポジトリの「Business Model and Mapping」レイヤーの作成と管理

次の各項で、ビジネス・モデルと論理オブジェクトの作成について説明します。

「Business Model and Mapping」レイヤーの作成について

この章は、リポジトリを計画および設定するためのロードマップの一部です。詳細は、「Oracle BI リポジトリの計画と作成」（41 ページ）を参照してください。

「Physical」レイヤーの要素をすべて作成した後で、「Physical」レイヤーのテーブルや列を、「Business Model and Mapping」レイヤーにドラッグすることができます。詳細は、「Oracle BI リポジトリの「Physical」レイヤーの作成と管理」（59 ページ）と「マルチディメンショナル・データソースのビジネス・モデル・レイヤーの作成」（114 ページ）を参照してください。

Administration Tool の「Business Model and Mapping」レイヤーでは、データのビジネス・モデル（または論理モデル）が定義され、ビジネス・モデルと物理レイヤーのスキーマとのマッピングが指定されます。

1 つ以上のビジネス・モデルを論理レイヤーに作成し、論理テーブルと論理列を各ビジネス・モデルに作成します。「Business Model and Mapping」レイヤーのオブジェクトを「Physical」レイヤーのソースに自動的にマッピングするには、「Physical」レイヤーのオブジェクトを論理レイヤーのビジネス・モデルにドラッグ・アンド・ドロップします。物理テーブルを「Business Model and Mapping」レイヤーにドラッグすると、対応する論理テーブルが作成されます。物理テーブルの各列に対応する論理列が作成されます。複数のテーブルを一度にドラッグした場合は、物理結合 1 つにつき 1 つの論理結合が作成されますが、作成されるのは新しいビジネス・モデルにテーブルを初めてドラッグしたときだけです。

マルチディメンショナル・データソースのビジネス・モデル・レイヤーの作成

マルチディメンショナル・データソースの場合も、「Business Model and Mapping」レイヤー（論理レイヤー）を設定する方法は、リレーショナル・データソースの設定方法に似ています。ビジネス・モデル・レイヤーを作成するには、物理レイヤーのキューブを論理レイヤーにドラッグ・アンド・ドロップします。ただし、物理キューブの内容は 1 つの論理テーブルとして追加されるので、列を並べ替えて適切な論理テーブルを作成し、階層を再作成する必要があります。

ビジネス・モデル・オブジェクトの作成

Administration Tool の「Business Model and Mapping」レイヤーには、1 つ以上のビジネス・モデル・オブジェクトを保持することができます。ビジネス・モデル・オブジェクトには、ビジネス・モデルの定義と、そのビジネス・モデルの論理テーブルから物理テーブルへのマッピングが含まれます。

注意：リポジトリをオフライン・モードで操作するときは、リポジトリを時々保存してください。ビジネス・モデルに一貫性がない場合でも、リポジトリはオフライン・モードで保存できます。

ビジネス・モデルを作成するには

1. 既存のオブジェクトより下にある「Business Model and Mapping」レイヤーを右クリックします。
2. ショートカット・メニューの「New Business Model」オプションを選択します。
3. ビジネス・モデルの名前を指定します。

4 対応するプレゼンテーション・レイヤーをクエリーで使用可能にする場合は、「Available for queries」オプションを選択します。

注意： このオプションを選択する前に、ビジネス・モデルで一貫性が維持されている必要があります。

5 （オプション）ビジネス・モデルの説明を入力して、「OK」をクリックします。

ビジネス・モデルとプレゼンテーション・カタログの複製

対応するビジネス・モデルとプレゼンテーション・カタログを選択して、複製を作成し、複製に名前を付けることができます。

注意： 別名はコピーされません。

ビジネス・モデルとそのプレゼンテーション・カタログをコピーするには

1 「Business Model and Mapping」レイヤーでビジネス・モデルを右クリックします。

2 右クリック・メニューの「Duplicate with Presentation Catalog」を選択します。

3 「Copy Business Model and Presentation Catalog」ダイアログ・ボックスで、コピーするビジネス・モデルを選択します。

4 ビジネス・モデルとそのカタログの新しい名前をそれぞれのフィールドに入力してから、「OK」をクリックします。

コピーされたビジネス・モデルが「Business Model and Mapping」レイヤー・ウィンドウに表示されます。

論理テーブルの作成と管理

論理テーブルは、「Business Model and Mapping」レイヤーに存在します。各ビジネス・モデルで定義される論理スキーマには、2 つ以上の論理テーブルが存在することと、これらのテーブルの間においてリレーションシップが定義されていることがそれぞれ必要です。

各論理テーブルには、1 つ以上の論理列と、1 つ以上の論理テーブル・ソースが関連付けられています。論理テーブル名の変更、論理テーブル・ソースの順序変更および論理キー（主キーと外部キー）の構成を行うことができます。

この項の内容は次のとおりです。

論理テーブルの作成

論理テーブルを作成するには通常、「Physical」レイヤーの物理テーブルを「Business Model and Mapping」レイヤーのビジネス・モデルにドラッグ・アンド・ドロップする方法を使用します。テーブルが物理スキーマに存在しない場合は、論理テーブルを手動で作成する必要があります。

「Business Model and Mapping」レイヤーにオブジェクトを作成する方法としては、通常はドラッグ・アンド・ドロップが最も簡単です。「Physical」レイヤーから「Business Model and Mapping」レイヤーに物理テーブルをドラッグ・アンド・ドロップすると、そのテーブルに属する列もコピーされます。オブジェクトを「Business Model and Mapping」レイヤーにドラッグ・アンド・ドロップした後は、そのオブジェクトを修正しても、「Physical」レイヤーのオブジェクトに影響しません。

キーと外部キー関係が定義されている物理テーブルをビジネス・モデルにドラッグ・アンド・ドロップすると、「Physical」レイヤーのキーと結合が反映される論理キーと結合が作成されます。作成されるのは、ドラッグ・アンド・ドロップしたテーブルに外部キーがある場合だけです。また、新しいテーブルを作成した場合や、後で別のテーブルを「Physical」レイヤーから「Business Model and Mapping」レイヤーにドラッグ・アンド・ドロップした場合は、新規に作成されたテーブルまたは追加でドラッグ・アンド・ドロップされたテーブルと、その前にドラッグ・アンド・ドロップされたテーブルとの間における論理リンクを手動で作成する必要があります。

結合の詳細は、「Joins Manager による論理結合の定義」（141 ページ）と「Business Model Diagram による論理結合の定義」（143 ページ）を参照してください。

ドラッグ・アンド・ドロップを使用して論理テーブルを作成するには

1. 「Physical」レイヤーのテーブルを 1 つ以上選択します。

   「Physical」レイヤーでのキーと結合を維持するには、テーブルに外部キーが作成されている必要があります。

2. テーブル・オブジェクトを、「Business Model and Mapping」レイヤーのビジネス・モデルにドラッグ・アンド・ドロップします。

   ドロップしたときに、テーブル・オブジェクト（物理ソース・マッピングを含む）が自動的に「Business Model and Mapping」レイヤーに作成されます。

論理テーブルを手動で作成するには

1. 「Business Model and Mapping」レイヤーで、テーブルの作成先ビジネス・モデルを右クリックして、「New Object」→「Logical Table」を選択します。

   「Logical Table」ダイアログ・ボックスが表示されます。

2. 「General」タブで、論理テーブルの名前を入力します。
3. このテーブルがブリッジ・テーブルの場合は、「Bridge table」オプションを選択します。

   詳細は、「ディメンション階層の識別」（46 ページ）を参照してください。

4. （オプション）テーブルの説明を入力します。
5. 「OK」をクリックします。

注意：論理テーブルを手動で作成した後で、すべてのキーと結合を手動で作成する必要があります。

論理テーブル・ソースの追加または編集

新規論理テーブル・ソースの追加、既存のテーブル・ソースの編集や削除、テーブル・ソースへのマッピングの作成や変更、および論理テーブル・ソースを使用するタイミングとコンテンツ集計方法の定義をそれぞれ実行できます。これらの作業の実行方法の詳細は、「論理テーブル・ソース（マッピング）の作成と管理」（121 ページ）を参照してください。

論理テーブルの主キーの指定

「Business Model and Mapping」レイヤーにテーブルを作成した後で、各テーブルの主キーを指定します。論理ディメンション・テーブルには、論理主キーが必要です。論理キーは、1 つ以上の論理列から構成できます。

注意：論理ファクト・テーブルの場合は、論理キーは必須ではありません。ただし、論理ファクト・テーブルの論理キーは指定しないことをお薦めします。詳細は、「論理テーブルの外部キーの確認」（117 ページ）を参照してください。

論理テーブルで主キーを指定するには

1. 「Business Model and Mapping」レイヤーで、テーブルをダブルクリックします。
2. 「Logical Table」ダイアログ・ボックスで「Keys」タブを選択してから「New」をクリックします。
3. 「Logical Key」ダイアログ・ボックスで、次の手順を実行します。
   a. キーの名前を入力します。
   b. 論理テーブルのキーとなる列のチェック・ボックスを選択します。
4. 「OK」をクリックします。

論理テーブルの外部キーの確認

「Foreign Keys」タブを使用すると、論理テーブルの外部キーを確認できます。

ファクト・テーブルでは、外部キーによる論理結合のかわりに、複合論理結合を使用することをお薦めします。複合論理結合を使用すると、柔軟に主キーを定義できます。物理テーブルに主キーがある場合は、このフィールドをファクト・テーブルの論理キーとして使用できます。Oracle BI リポジトリではこの方法をお薦めします。

警告：論理テーブルでは外部キーを使用しないことをお薦めします。ただし、論理外部キーや論理複合結合を作成するには、Joins Manager または Business Model Diagram を使用できます。ファクト・テーブルの論理キーは、属性テーブルに結合しているキー列で構成されている必要があります。詳細は、「論理結合の定義」（141 ページ）を参照してください。

論理テーブルの外部キー情報を確認するには

1. 「Business Model and Mapping」レイヤーで、テーブルをダブルクリックします。
2. 「Logical Table」ダイアログ・ボックスで「Foreign Keys」タブを選択します。

3 既存の外部キーを確認するには、「Foreign Keys」の一覧でそのキーを選択して「Edit」をクリックします。

「Logical Foreign Key」ダイアログ・ボックスが表示されます。このダイアログ・ボックスで情報を変更する方法の詳細は、「論理結合の定義」（141 ページ）を参照してください。

論理列の作成と管理

多くの論理列は、「Physical」レイヤーから「Business Model and Mapping」レイヤーにテーブルをドラッグ・アンド・ドロップすると自動的に作成されます。これ以外に、特に他の論理列に基づいて計算を行う列を後で作成することができます。

論理列が属する論理テーブルから展開されるツリー構造として、論理列は表示されます。主キーである列や、主キーに属する列では、キー・アイコンが表示されます。列に集計ルールがある場合は、シグマ記号のアイコンが表示されます。「Business Model and Mapping」レイヤーにある論理列の順序を変更することができます。

この項の内容は次のとおりです。

論理列の作成と移動

「General」タブを使用して、論理列の一般的なプロパティを作成または編集します。論理列オブジェクトを「Business Model and Mapping」レイヤーに作成してから、「Presentation」にドラッグ・アンド・ドロップすることができます。

論理列におけるソートについて

論理列では、ソートのベースとなる別の列を指定することができます。この方法は、辞書的な順序でない順序で列をソートする場合に利用します。辞書的な順序でソートを行うと、結果は辞書と同じようにアルファベット順になります。このソート方法では、数字は別のグループに分けられるのではなく、アルファベットのスペルによってソートされます。

たとえば、MONTH_NAME という列を使用して月をソートすると、結果は February、January、March のように辞書の順序になります。ただし、このソート順を時間順に変更する場合があります。それには、テーブルに月のキー（たとえば MONTH_KEY）を追加し、1 月（January）は 1、2 月（February）は 2、3 月（March）は 3 のように値を持たせます。目的のソート順を実現するために、MONTH_NAME 列の「Sort order column」フィールドを MONTH_KEY に設定します。その後のリクエストで、MONTH_NAME 順を指定すると、結果は January、February、March の順に返されます。

論理列を作成するには

1 「Business Model and Mapping」レイヤーで論理テーブルを右クリックします。

2 ショートカット・メニューの「New Object」→「Logical Column」を選択します。

3 「Logical Column」ダイアログ・ボックスで「General」タブを選択します。

4 「General」タブで、論理列の名前を入力します。

ビジネス・モデルの名前と対応する論理テーブルが「Belongs to Table」フィールドに表示されます。

5 （オプション）列のソート順序の基準として別の列を指定する場合は、次の手順を実行します。

a 「Sort order column」フィールドの横にある「Set」をクリックします。

b 「Browse」ダイアログ・ボックスで、列を選択します。

c 列の詳細情報を表示するには、「View」をクリックしてその列の「Logical Column」ダイアログ・ボックスを開いてから、「取消」をクリックします。

注意：このダイアログ・ボックスでは、変更を行うことができます。行った変更を反映するには、「取消」ではなく「OK」をクリックします。

d 「Browse」ダイアログ・ボックスで「OK」をクリックします。

6 （オプション）「Sort order column」の値を消去する場合は、「Clear」をクリックします。

7 （オプション）この論理列の値を他の論理列から導出する場合は、次の手順を実行します。

a 「Use existing logical columns as source」のチェック・ボックスを選択します。

b テキスト・ボックスの横にある省略記号ボタンをクリックして、式ビルダーを開きます。

c 「Expression Builder - Derived logical column」ダイアログ・ボックスで、論理列の導出元となる式を指定します。

d 「OK」をクリックします。

8 （オプション）「Logical Column」ダイアログ・ボックスで、論理列の説明を入力します。

注意：タイプと長さのフィールドは、列のソースに基づいて自動的に入力されます。

9 「OK」をクリックします。

論理列を移動またはコピーするには

1 「Business Model and Mapping」レイヤーで、ある論理列を別の論理テーブルにドラッグ・アンド・ドロップします。

注意：移動する列は複数選択可能です。

2 「Sources for moved columns」ダイアログ・ボックスの「Action」領域で、アクションを選択します。

3 「Ignore」を選択した場合は、ドラッグ先テーブルの「Sources」フォルダに論理ソースは追加されません。

4 「Create new」を選択した場合は、論理列に対応する論理ソースのコピーが、ドラッグ先テーブルの「Sources」フォルダに作成されます。

5 「Use existing」を選択した場合は、「Use existing」ドロップダウン・リストにおいて、ドラッグ先テーブルの「Sources」フォルダから論理ソースを選択する必要があります。

移動またはコピーした列は、この論理ソースに関連付けられます。

メジャー列のデフォルト集計レベルの設定

マッピングされた論理列がメジャー列の場合は、集計ルールを指定する必要があります。集計の対象はメジャー列のみですが、COUNT と Count Distinct の集計は例外です。メジャー列は、論理ファクト・テーブルにのみ存在する必要があります。

注意：マルチディメンショナル論理列の場合は、デフォルト集計ルールを Aggr_External に設定することができます。

特定の論理テーブル・ソースに対して、優先集計式を指定することができます。これによって、デフォルトの集計ルールが Count Distinct である場合に Oracle BI Server で集計テーブルが活用されます。優先ルールを指定しない場合は、デフォルトのルールが有効になります。

デフォルトでは、データがスパースであるとみなされます。ただし、まれに論理テーブル・ソースに稠密データが存在していることもあります。論理テーブル・ソースで稠密データが存在しているとみなされるのは、対応するディメンション・レベルの各組合せで行が存在している場合です。メジャー列の集計ルールを設定するときに、データが稠密であると指定できるのは、マッピング先の論理テーブル・ソースがすべて稠密である場合のみです。

警告：データが稠密であると指定したにもかかわらず、その列で使用されているテーブル・ソースにおいて稠密データがないものがある場合は、正しい結果は返されません。

メジャー列のデフォルト集計ルールを指定するには

1. 「Business Model and Mapping」レイヤーで、論理列をダブルクリックします。
2. 「Logical Column」ダイアログ・ボックスで、「Aggregation」タブをクリックします。
3. 「Aggregation」タブで、次のフィールドに値を入力します。
   - 時間ディメンションに基づく集計ルールの場合は、「Based on dimensions」チェック・ボックスを選択します。

     「Data is dense」チェック・ボックスが表示されます。
   - この列のマッピング先である論理テーブル・ソースがすべて稠密である場合は、「Data is dense」チェック・ボックスを選択してください。

     警告：このチェック・ボックスを選択すると、この列のマッピング先ソースはすべて、ディメンション・レベルの各組合せで行が存在することを示します。このチェック・ボックスを選択したにもかかわらず、その列で使用されているテーブル・ソースにおいて稠密データがないものがある場合は、正しい結果は返されません。
   - いずれかの集計関数を「Default Aggregation Rule」ドロップダウン・リストで選択します。

     選択した関数は、エンド・ユーザーまたはアプリケーションからのクエリーでこの列がリクエストされたときは必ず適用されます。
4. 「OK」をクリックします。

ディメンション・テーブルにおける論理レベルと属性との関連付け

「Levels」タブでディメンション・レベルを選択すると、属性を論理レベルに関連付けることができます。メジャーは複数のディメンションからのレベルを関連付けることができ、指定されたレベルで集計が必ず実行されます。

ディメンションが「Dimensions」リストに表示されます。この属性が論理レベルに関連付けられている場合は、そのレベルが「Levels」リストに表示されます。

メジャーにディメンションのレベルを関連付けるには、「Business Model and Mapping」レイヤーでディメンション・ツリーを開いてから、目的のレベルに列をドラッグ・アンド・ドロップする方法もあります。レベルベースのメジャーの詳細は、「レベルベースのメジャー計算の例」（135 ページ）を参照してください。

メジャーにディメンションの論理レベルを関連付けるには

1. 「Business Model and Mapping」レイヤーで、論理列をダブルクリックします。
2. 「Logical Column」ダイアログ・ボックスで、「Levels」タブをクリックします。
3. 「Levels」タブで、選択する論理レベルが属しているディメンションの「Logical Levels」フィールドをクリックします。

   注意：「Levels」タブのレベル・リストにおいて、列見出しをクリックすると行をソートすることができます（昇順または降順に切り替わる）。
4. 「Logical Levels」ドロップダウン・リストで、レベルを選択します。
5. 別のディメンションにおける別の論理レベルをこのメジャーに関連付けるには、この手順を繰り返します。

ディメンションとメジャーの関連付けを解除するには

1. 「Business Model and Mapping」レイヤーで、論理列をダブルクリックします。
2. 「Logical Column」ダイアログ・ボックスで、「Levels」タブをクリックします。
3. 「Levels」タブで、「Logical Levels」フィールドの横にある削除ボタンをクリックします。
4. 「OK」をクリックします。

論理テーブル・ソース（マッピング）の作成と管理

新規論理テーブル・ソースの追加、既存のテーブル・ソースの編集や削除、テーブル・ソースへのマッピングの作成や変更、および論理テーブル・ソースを使用するタイミングとコンテンツ集計方法の定義をそれぞれ実行できます。さらに、集計コンテンツを Windows のクリップボードにコピーしたり、別の論理テーブル・ソースからコピーしたり、論理ファクト・テーブル・ソースの集計コンテンツの確認もできます。

新規に論理テーブル・ソースを追加するのは、情報のソースとなる物理テーブルが複数存在する場合です。たとえば、多数のテーブルに売上情報が格納されている場合があります。3 つの事業部門があり、それぞれ専用の注文システムから売上情報を取得している場合です。別の例としては、売上情報を注文システムや財務システムから定期的にまとめ、このテーブルを使用して上層部用レポートを作成する場合などもあります。

各論理テーブルに、論理テーブルのソース・フォルダが 1 つ存在します。このフォルダには、論理テーブル・ソースが 1 つ以上あります。このソースによって、論理テーブルから物理テーブルへのマッピングが定義されます。計算式を使用するような複雑なマッピングも、論理テーブル・ソースにおいて構成されます。

論理テーブルにおいて多数の物理テーブル・ソースを持たせることもできます。1 つの論理列が複数の物理テーブルから多数の物理列にマッピングされる場合がありますが、列にマッピングする集計テーブルも物理テーブルに含まれます（その列に対する特定の集計レベルがクエリーで要求された場合）。

「Physical」レイヤーからドラッグ・アンド・ドロップして論理テーブルと列を作成すると、論理テーブル・ソースが自動的に生成されます。論理テーブルを手動で作成した場合は、ソースも手動で作成する必要があります。

集計ナビゲーション用のフラグメンテーション・コンテンツを設定する方法の例は、「フラグメンテーション・コンテンツの指定」（206 ページ）を参照してください。

この項の内容は次のとおりです。

論理テーブル・ソースの作成または削除

「Logical Table Source」ダイアログ・ボックスの「General」タブを使用して、論理テーブル・ソースの一般的なプロパティを定義します。

論理テーブル・ソースを作成するには

1. 「Business Model and Mapping」レイヤーで、論理テーブルを右クリックしてから、「New Object」→「Logical Table Source」を選択します。
2. 「Logical Table Source」ダイアログ・ボックスで「General」タブをクリックし、論理テーブル・ソースの名前を入力してから「Add」をクリックします。
3. 「Browse」ダイアログ・ボックスで、結合を確認したり、論理テーブル・ソースのテーブルを選択できます。

   1 つの論理テーブル・ソースに複数のテーブルがある場合は、関係するすべてのテーブルの間に結合が定義されている必要があります。
4. 結合を確認するには、「Browse」ダイアログ・ボックスでテーブルを選択してから、「View」をクリックします。
   - 「Physical Table」ダイアログ・ボックスで、結合を確認してから、「取消」をクリックします。
5. テーブルをテーブル・ソースに追加するには、テーブルを「Name」リストで選択してから「Select」をクリックします。
6. 「Logical Table Source」ダイアログ・ボックスで「Column Mapping」タブをクリックし、「物理テーブル・ソースから論理テーブル・ソースへのマッピングの定義」（123 ページ）の指示に従ってフィールドに値を入力します。
7. 「Logical Table」ダイアログ・ボックスで「Content」タブをクリックし、「論理テーブル・ソースのコンテンツの定義」（125 ページ）の指示に従ってフィールドに値を入力します。

8 「OK」をクリックします。

ソースとしてのテーブルを削除するには

1 「Business Model and Mapping」レイヤーで、論理テーブル・ソースを右クリックしてから「Properties」を選択します。

2 「Logical Table Source」ダイアログ・ボックスで、「General」タブをクリックします。

3 削除するテーブルをテーブル・リストで選択してから「Remove」をクリックします。

4 適切なテーブルが削除されたら、「OK」をクリックします。

集計ファクト・データのレベルごとにソースを作成する例

集計ファクト・テーブルのソースを作成することに加えて、対応する論理ディメンション・テーブル・ソースを、同じ集計レベルで作成する必要があります。

注意：集計コンテンツの指定において参照されているレベルごとに、1 つ以上のソースが必要です。各レベルのソースがすでに存在する場合は、新規に作成する必要はありません。

たとえば、月別売上テーブルに、毎月の各店舗における製品別の売上を合計した金額があらかじめ計算されて格納されていると仮定します。この場合、次に示す 3 つのソース（例において参照されている各論理ディメンション・テーブルに 1 つ）が必要です。

- Product 論理テーブルのソースでは、次のいずれかのコンテンツが指定されています。
  - 論理レベル：ProductDimension.ProductLevel
  - 列：Product.Product_Name
- Store 論理テーブルのソースでは、次のいずれかのコンテンツが指定されています。
  - 論理レベル：StoreDimension.StoreLevel
  - 列：Store.Store_Name
- Time 論理テーブルのソースでは、次のいずれかのコンテンツが指定されています。
  - 論理レベル：TimeDimension.MonthLevel
  - 列：Time.Month

物理テーブル・ソースから論理テーブル・ソースへのマッピングの定義

「Logical Table Source」ダイアログ・ボックスの「Column Mapping」タブを使用して、論理列を物理列にマッピングします。物理から論理へのマッピングは、「Physical」レイヤーと「Business Model and Mapping」レイヤーとの間で行われる変換を指定するためにも使用できます。この変換には、データ型を整数型から文字型に変更するという単純な変換もあれば、計算式を適用して単位人口当たりの売上比率を算出するといった複雑な変換もあります。

論理列を物理列にマッピングするには

1. 「Logical Table Source」ダイアログ・ボックスが開いていない場合は、「Business Model and Mapping」レイヤーで、論理テーブル・ソースをダブルクリックします。
2. 「Logical Table Source」ダイアログ・ボックスで、「Column Mapping」タブをクリックします。
3. 124 ページの図 11 に示すように、ダイアログ・ボックスを最大化するかそのサイズを大きくして、「Column Mapping」タブにあるすべての内容を表示します。

   注意：「Column Mapping」タブの「Logical column to physical column」領域で、列見出しをクリックすると行をソートすることができます（昇順、降順および元の順序のサイクルで切り替わる）。
4. 「Physical Table」列で、マッピングする列のあるテーブルを選択します。

   「Physical Table」列のセルを選択すると、ドロップダウン・リストが表示されます。ここには、この論理テーブル・ソースに現在含まれているテーブルのリストがあります。
5. 「Expression」リストで、各論理列に対応する物理列を選択します。

   「Expression」列のセルを選択すると、ドロップダウン・リストが表示されます。ここには、この論理テーブル・ソースに現在含まれているテーブルのリストがあります。
6. 式ビルダーを開くには、表示または編集する式の左にある省略記号ボタンをクリックします。

   注意：物理式の作成に使用される列はすべて、論理テーブル・ソースに含まれているテーブルに存在していることが必要です。ソースの外部にあるテーブルの列を使用して式を作成することはできません。
7. 列のマッピングを削除するには、削除ボタンをクリックします。

   削除ボタンが見つからない場合は、右にスクロールしてください。
8. 適切な列がマッピングされたら、「OK」をクリックします。

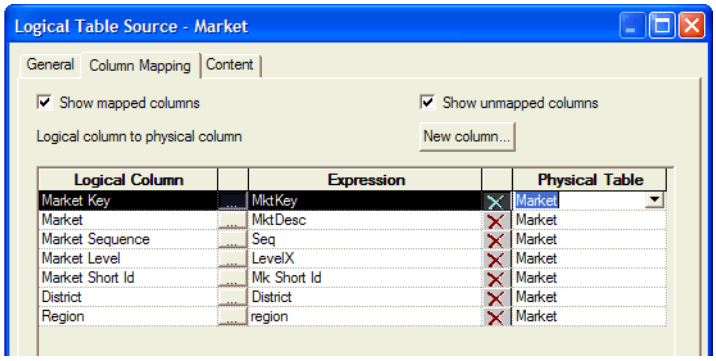


図 11.　「Logical Table Source」ダイアログ・ボックス

列マッピングを削除するには

1. 「Business Model and Mapping」レイヤーで、論理テーブルを右クリックしてから、「New Object」→「Logical Table Source」を選択します。
2. 「Logical Table Source」ダイアログ・ボックスで、「Column Mapping」タブをクリックします。

3 124 ページの図 11 に示すように、ダイアログ・ボックスを最大化するかそのサイズを大きくして、「Column Mapping」タブにあるすべての内容を表示します。

4 列マッピングを削除するには、「Physical Table」のセルの横にある削除ボタンをクリックします。

5 「OK」をクリックします。

論理列とソースとのマッピングの解除

「Logical Column」ダイアログ・ボックスの「Datatype」タブには、論理列に関する情報が表示されます。列データの導出元である論理テーブル・ソースを編集することや、ソースとのマッピングを解除することができます。

論理列とソースとのマッピングを解除するには

1 「Business Model and Mapping」レイヤーで、論理列をダブルクリックします。

2 「Logical Column」ダイアログ・ボックスで、「Datatype」タブをクリックします。

3 「Datatype」タブの「Logical Table Source」リストでソースを選択してから、「Unmap」をクリックします。

4 「OK」をクリックします。

論理テーブル・ソースのコンテンツの定義

ソースを正しく使用するには、Oracle BI Server において、各ソースに格納されているものがビジネス・モデルの観点で認識されている必要があります。そのためには、ファクト・テーブルの論理テーブル・ソースごとに集計コンテンツを定義する必要があります。集計コンテンツのルールとは、このファクト・テーブルに格納されているデータの粒度を定義するものです。このファクト論理テーブルに関係するディメンションごとに、粒度のレベルを定義して、関係するディメンションがすべて定義されるようにします。詳細は、「集計ファクト・データのレベルごとにソースを作成する例」（123 ページ）を参照してください。

論理テーブルのソースが一連のフラグメントである場合、すべてのフラグメントのマッピング先が同一の列セットにする必要はありません。ただし、列がどのようにマッピングされているかによって、サーバーから返される応答は異なります。

- ある論理テーブルのすべてのフラグメントが同一の列セットにマッピングされている場合、このフラグメントされたソースのセットが、その論理テーブルの論理テーブル・ソースの全体集合であるとみなされます。つまり、フラグメントのセットに基づいてメジャー集計を計算できるということです。
- マッピングされた列のセットがフラグメントによって異なる場合は、フラグメントの全体集合が存在しないことになるので、集計ロールアップを計算するのは適切ではなくなります（欠落しているフラグメントが存在するため）。

  この場合は、メジャー集計として NULL が返されます。

注意：すべてのフラグメントを同一の列セットにマッピングすることをお薦めします。

「Logical Table Source」ダイアログ・ボックスの「Content」タブを使用して、集計テーブル・コンテンツ定義、ソース用フラグメント化テーブル定義および Where 句（返される行の行数を制限する場合）を定義します。

注意：集計ナビゲーション用のフラグメンテーション・コンテンツを設定する方法の例は、「Oracle BI リポジトリでの集計ナビゲーション用のフラグメンテーション・コンテンツの設定」（205 ページ）を参照してください。

ディメンション・テーブルからファクト・テーブルへの既存の結合の確認

このソース・コンテンツ情報は、クエリーを適切な物理集計ファクト・テーブルに送信して、適切な物理集計ディメンション・テーブルとの結合を行ってそのテーブルの値により絞り込む処理を Oracle BI Server で行うために必要な情報です。「Physical」レイヤーにおいて、集計ファクト・テーブルと集計ディメンション・テーブルとの間に結合が存在していることを確認してください。

結合が存在することを確認する推奨方法は、ファクト論理テーブルを選択して、Business Model Diagram（Selected Tables and Direct Joins）をリクエストする方法です。このファクト論理テーブルに直接結合しているディメンション論理テーブルのみ、ダイアグラムに表示されます。同じ物理テーブルが論理ファクト・ソースとディメンション・ソースで使用されている場合は、ディメンション・テーブルは表示されません。

126 ページの図 12 は、Business Model Diagram（Selected Tables and Direct Joins）ビューにおける Fact - Assess ファクト論理テーブルの表示例です。

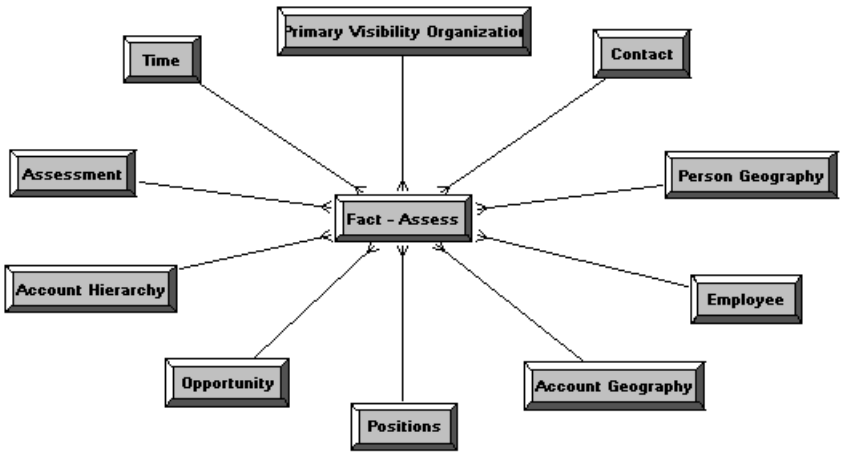


図 12. ファクト・テーブルの直接結合の図

127 ページの表 17 は、126 ページの図 12 に示した Fact - Assess ファクト・テーブルと直接結合している各ディメンション・テーブルの論理レベルの一覧です。

表 17. 「Content」タブに表示されるディメンションと論理レベル

ディメンション	論理レベル
Account Geography	Postal Code Detail
Person Geography	Postal Code Detail
Time	Day Detail
Account Organization	Account Detail
Opportunity	Opty Detail
Primary Visibility Organization	Detail
Employee	Detail
Assessment	Detail
Contact (W_PERSON_D)	Detail
FINS Time	Day
Positions	Details

論理テーブル・ソースのコンテンツ定義を作成するには

1. 「Business Model and Mapping」レイヤーで、論理テーブル・ソースをダブルクリックします。
2. 「Logical Table Source」ダイアログ・ボックスで、「Content」タブをクリックしてから、128 ページの表 18 の説明を参考にして次の手順を実行します。
3. 論理ソースが集計テーブルであり、論理ディメンションを定義した場合は、次の手順を実行します。
   a. 「Aggregation content, group-by」ドロップダウン・リストで「Logical Level」を選択します。

      警告：集計コンテンツを論理レベルまたは列で指定するオプションがありますが、論理レベルのみを使用することをお薦めします。

   b. 「Logical Level」ドロップダウン・リストで、ファクト論理テーブルの結合先である各ディメンション論理テーブルに対して適切なレベルを選択します。

      各ディメンションの論理レベルを指定してください（総計レベルを指定する場合を除く）。レベルが指定されていないディメンションは、総計レベルであると解釈されます。
4. 論理ソースが集計テーブルである場合に、列のコンテンツを定義するには、次の手順を実行します。
   a. 「Aggregation content, group-by」ドロップダウン・リストで「Column」を選択します。

      警告：集計コンテンツを論理レベルまたは列で指定するオプションがありますが、論理レベルのみを使用することをお薦めします。

   b. 「Table」ペインで、ソースの集計レベルを定義する論理ディメンション・テーブルをそれぞれ選択します。

c 「Column」ペインで、集計のグループ化方法を定義する各ディメンションに対応する論理列を選択します。

使用可能な論理列が複数ある場合は、ソース物理テーブルのキーにマッピングしている列を選択します。たとえば、データが Region 論理レベルで集計されている場合は、Region テーブルのキーにマッピングしている論理列を選択します。

注意：同一のビジネス・モデルにおいて、論理レベルによる集計と列による集計を混在させないでください。論理レベルによる集計を使用することをお薦めします。

5 ソースに対してフラグメント化テーブル定義を指定するには、ソースが特定の集計レベルにおけるデータの一部を表すときにそのソースに含まれる値の範囲を「Fragmentation content」ウィンドウで指定します。

計算式をウィンドウに直接入力するか、ウィンドウの右側にある「Expression Builder」ボタンをクリックします。「Fragmentation Content Expression Builder」で、既存の論理列を使用してコンテンツを指定できます。集計ナビゲーション用のフラグメンテーション・コンテンツを設定する方法の例は、「フラグメンテーション・コンテンツの指定」（206 ページ）を参照してください。

6 次のオプションを選択します。

This source should be combined with other sources at this level

注意：このオプションは、複数のソースの集計レベルが同一である場合にのみ使用できます。

7 （オプション）ソースに使用される結果テーブルの行数を制限する場合は、Where 句フィルタを「Where Clause Filter」ウィンドウで指定します。詳細は、「WHERE 句フィルタについて」（129 ページ）を参照してください。

a 「Expression Builder」ボタンをクリックして、「Physical Where Filter Expression Builder」を開きます。

b Where 句を入力してから「OK」をクリックします。

8 ソースの値が一意の場合は、「Select distinct values」オプションを選択します。

表 18. 論理テーブル・ソースの「Content」タブのフィールド

フィールド	説明
Aggregation content, group by	コンテンツの集計方法を示します。
Fragmentation content	データソースのコンテンツをビジネス・モデルの観点で説明したものです。データがフラグメント化されるのは、同一集計レベルの情報が、データの値に応じて複数のテーブルに分割されるときです。例として、期間によるデータのフラグメント化があります。集計ナビゲーション用のフラグメンテーション・コンテンツを設定する方法の例は、「フラグメンテーション・コンテンツの指定」（206 ページ）を参照してください。

表 18. 論理テーブル・ソースの「Content」タブのフィールド

フィールド	説明
「More」（ボタン）	「More」をクリックすると、次のオプションが表示されます。 ■ **Copy**: 集計コンテンツを Windows のクリップボードにコピーします（ファクト・テーブルの場合のみ使用可能）。Dimension.Level の情報をテキスト・エディタに貼り付けて、ドキュメントの検索やドキュメントへの追加に使用することができます。 ■ **Copy from**: 集計コンテンツを、同じビジネス・モデルにある別の論理テーブル・ソースからコピーします（ファクト・テーブルとディメンション・テーブルの場合のみ使用可能）。集計コンテンツのコピー元であるソースを指定する必要があります。複数のビジネス・モデルが表示されますが、現在のビジネス・モデルからの論理テーブル・ソースのみ、選択できます。 ■ **Get Levels**: 集計コンテンツを変更します（ファクト・テーブルの場合のみ使用可能）。ファクト・テーブル・ソースとディメンション・テーブル・ソースの間に結合が存在しない場合（たとえば、両ソースに同一の物理テーブルがある場合）は、管理ツールによって決まる集計コンテンツに、このディメンションの集計コンテンツが含まれなくなります。 ■ **Check Levels**: 論理ファクト・テーブル・ソース（ディメンション・テーブル・ソースではない）の集計コンテンツをチェックします（ファクト・テーブルの場合のみ使用可能）。返される情報は、論理レベルとレベル・キーがあるディメンションと階層が存在するかどうかによって異なり、ディメンション・テーブル・ソースのテーブルとファクト・テーブル・ソースのテーブルの間に物理結合が存在するかどうかによっても異なります。ファクトとディメンションの両方のソースに同一のテーブルが存在しているが、両方のソースのテーブルの間に物理結合が存在しない場合は、「Check Levels」の結果にこのディメンションの集計コンテンツは含まれません。
Select distinct values	ソースの値が一意である場合に使用します。
「This source should be combined with other sources at this level」（チェック・ボックス）	同一集計レベルにおける複数のデータソースに格納されている情報が重複していない場合に、このチェック・ボックスを選択します。この場合、この集計レベルの情報全体を取得するには、すべてのソースを結合する必要があります。

WHERE 句フィルタについて

WHERE 句フィルタは、論理テーブル・ソースにおいて参照される物理テーブルに絞込み条件を適用するときに使用します。集計ソースに対する絞込み条件がない場合は、WHERE 句フィルタを空欄のままにします。

各論理テーブル・ソースには、集計レベルの共通部分におけるデータが格納されます。たとえば、1 つのソースでブランド・レベルとメーカー・レベルの両方における売上データを持つソースを作成することはありません。物理テーブルに複数のレベルでデータが格納されている場合は、適切な WHERE 句絞込み条件を追加して値をフィルタ処理し、レベルが 1 つになるようにします。

WHERE 句フィルタにある絞込み条件は、ソースにおいて物理テーブルに適用されます。

ディメンションと階層レベルについて

ビジネス・モデルにおいてディメンションが表すのは、1 つの論理ディメンション・テーブルに属する論理列（属性）の階層構造です。一般的なディメンションには、期間、製品、市場、顧客、サプライヤ、販促条件、原材料、製造工場、輸送手段、メディアのタイプ、時間などがあります。ディメンションは、「Business Model and Mapping」レイヤー（論理レイヤー）に存在し、エンド・ユーザーからは見えません。

各ディメンションにおいて属性を階層レベルに編成します。これらの論理レベルが表すものは、組織のルールや、実際の業務における報告のニーズです。この構造（メタデータ）は、ディメンションにおけるドリル処理によってデータを詳細に調べるときに、Oracle BI Server によって使用されます。

ディメンションの階層レベルは、次の作業を行うために使用します。

- 集計ナビゲーションの設定
- レベルベースのメジャー計算の構成（「レベルベースのメジャー計算の例」（135 ページ）を参照）
- Oracle BI Presentation Services ユーザーがデータ・リクエストでドリルダウンしたときに表示される属性の設定

ディメンションの作成と管理のプロセス

各ビジネス・モデルには 1 つ以上のディメンションを持たせることができます。各ディメンションには 1 つ以上の論理レベルを持たせることができます。各論理レベルには、1 つ以上の属性（列）が関連付けられます。

注意： マルチディメンショナル・データソースにおけるディメンションの概念は、他のデータソースにおけるディメンションほど複雑ではありません。たとえば、ディメンション・レベル・キーは作成しません。ディメンションは、特定のマルチディメンショナル・データソースに固有です（複数のデータソースで使用できない）。また、個別に作成したり操作したりすることはできません。さらに、データソースの各キューブには、1 つ以上のディメンションと 1 つのメジャーが論理レイヤーに定義されている必要があります。

次の各項で、ディメンションの作成方法について説明します。

- ディメンションの作成
- ディメンション・レベルとキーの作成（131 ページ）
- ディメンションの自動作成（137 ページ）

ディメンションの作成

作成したディメンションには、1 つ以上の論理ディメンション・テーブルの属性（列）や、論理ファクト・テーブルのレベルベース・メジャーを関連付けることができます。論理列とディメンション・レベルとを関連付けると、その列が存在するテーブルが「Dimension」ダイアログ・ボックスの「Tables」タブに表示されるようになります。

ディメンションを作成するには

1 「Business Model and Mapping」レイヤーで、ビジネス・モデルを右クリックしてから、「New Object」→「Dimension」を選択します。

2 「Dimension」ダイアログ・ボックスの「General」タブで、ディメンションの名前を入力します。

3 時間ディメンションの場合は、「Time Dimension」チェック・ボックスを選択します。

注意：「Default root level」フィールドの値は、論理列とディメンション・レベルとが関連付けられた後で自動的に入力されます。

4 （オプション）ディメンションの説明を入力します。

5 「OK」をクリックします。

ディメンション・レベルとキーの作成

1 つのディメンションには、複数の論理レベルがあります。論理レベルを作成するには、総計レベルを作成してから、子レベルを上位レベルから下位レベルへ順に作成していくことをお薦めします。ディメンションを構成する各部分を次に示します。

- **総計レベル**：ディメンションの総計を表す特別なレベルです。各ディメンションが持つことのできる総計レベルは 1 つだけです。総計レベルには、ディメンション属性やレベル・キーはありません。ただし、メジャーを総計レベルに関連付けることは可能です。その場合、メジャーの集計レベルは必ず、ディメンションの総計になります。
- **レベル**：総計レベルを除いてすべてのレベルは、列が 1 列以上必要です。ただし、テーブルのすべての列に論理レベルを明示的に関連付ける必要はありません。論理レベルに関連付けられていない列には自動的に、ディメンション・テーブルに対応するディメンションの最下位レベルが関連付けられます。同じディメンション・テーブルの論理列はすべて、同じディメンションに関連付けられている必要があります。
- **階層**：各ビジネス・モデルの論理レベルにおいて、階層（レベルの親子関係）を作成する必要があります。たとえば、週データをロールアップしたものが年データとなるようにモデルを設定します。週をロールアップしないようにモデルを設定する場合もあります。たとえば、週データをロールアップして年データを作成するモデルでは、暗黙的に、各週に関連付けられる年は 1 つだけになります。これは、ある週が別々の年にまたがることもあるカレンダ週には該当しない場合があります。月と年の組合せが 1 四半期にロールアップすると、階層によっては複数の要素でロールアップが必要な場合があります。分析の結果が実際の業務のニーズと要件に適合するように業務の階層レベルを定義します。
- **レベル・キー**：各論理レベル（総計レベルとして定義される最上位レベルを除く）には、レベル・キーで構成されている属性が 1 つ以上必要です。レベル・キーでは、各論理レベルにおける一意の要素を定義します。ディメンション・テーブルの論理キーは、ディメンションの最下位レベルと関連付ける必要があります。また、そのレベルのレベル・キーにする必要もあります。

  論理レベルでは、複数のレベル・キーを持たせることができます。この場合は、そのレベルの主キーとなるキーを指定します。特定のレベルにおいて集計コンテンツがあるディメンション・ソースはすべて、そのレベルの主キーである列が必要です。Answers や Intelligence Dashboard のユーザーがドリルダウンするためにクリックしたときに表示されるレベル・キーを、各論理レベルにおいて 1 つ設定します。これは、そのレベルの主キーでも、主キーでなくてもかまいません。表示対象のレベル・キーを設定するには、「Level Key」ダイアログ・ボックスの「Use for drill down」チェック・ボックスを選択します。

  ドメインに January（1 月）、February（2 月）などの値が含まれる Month（月）のようなレベル・キーを使用する場合は注意してください。このような値は、特定の月に一意にならないで、毎年繰り返します。Month（月）をレベル・キーとして定義するには、上位レベルからの属性（たとえば Year（年））を追加する必要があります。Year（年）を追加するには、このダイアログの「Add」ボタンをクリックして、表示されたダイアログでその論理列を選択します。

- **時間ディメンションと時間順キー**：年（Year）などのディメンションを時間ディメンションとして指定することができます。時間ディメンションの 1 つ以上のレベルでは、時間順キーが必要です。時間ディメンションを設定して使用するときのガイドラインを次に示します。
  - 時間ディメンションの 1 つ以上のレベルでは、時間順キーが必要です。詳細は、「時間ディメンションにおける時間順キーの選択とソート」（134 ページ）を参照してください。
  - Ago 関数や ToDate 関数を使用する時系列メジャーはすべて、時間レベルにする必要があります。Ago 関数や ToDate 関数による集計は、導出論理列として作成されます。詳細は、「時系列の変換関数について」（201 ページ）を参照してください。
  - 時間論理テーブルの一部である物理テーブルが、別の論理テーブルに出現してはいけません。これによって、カレンダ・テーブルからの日付フィールドをメジャーとして使用することはできなくなります。
  - 時間ソースにおける物理テーブル（最も詳細なものを除く）と、ソースの外部にあるテーブルとを結合することはできません。結合は、時間テーブルとファクト・テーブルとの間で行う必要があります。結合ができるのは外部キーに基づいた場合のみです。複合結合であってはいけません。
  - Ago 関数や ToDate 関数は、フラグメント化論理テーブル・ソースではサポートされていません。詳細は、「時系列の変換関数について」（201 ページ）を参照してください。

ディメンション階層レベルの作成や管理を行うには、次の作業を行います。

ディメンションにおける論理レベルの作成

論理レベルをディメンションにおいて作成する場合は、レベルのタイプを指定してから子レベルを定義することによって階層も作成します。マルチディメンショナル・データソースの階層を作成する方法の詳細は、「マルチディメンショナル・データソースのビジネス・モデル・レイヤーの作成」（114 ページ）を参照してください。

ディメンションにおいて論理レベルの一般的プロパティを定義するには

1. 「Business Model and Mapping」レイヤーで、ディメンションを右クリックしてから、「New Object」→「Logical Level」を選択します。
2. 「Logical Level」ダイアログ・ボックスの「General」タブで、論理レベルの名前を指定します。
3. この論理レベルに存在する要素の数を指定します。このレベルが総計レベルの場合は、このフィールドを空欄のままにします。システムではデフォルトで値が 1 に設定されます。

   この数値は集計ソースの選択時に、Oracle BI Server によって使用されます。正確な数値にする必要はありませんが、複数の論理レベル間における比率が正確になるようにしてください。

4 次の条件が満たされている場合は、次に指定された作業を実行します。

- 論理レベルが総計レベルの場合は、「Grand total level」チェック・ボックスを選択します。

  注意：総計は、1 つのディメンションに 1 つのみです。

- 論理レベルを親レベルにロールアップする場合は、「Supports rollup to parent elements」チェック・ボックスを選択します。
- 論理レベルが総計レベルではなく、ロールアップも行わない場合は、どちらのチェック・ボックスも選択しないでください。

5 子論理レベルを定義するには、「Add」をクリックします。

6 「Browse」ダイアログ・ボックスで、子論理レベルを選択してから「OK」をクリックします。

子レベルが「Child Levels」ペインに表示されます。

7 定義済の子レベルを削除するには、そのレベルを「Child Levels」ペインで選択してから「Remove」をクリックします。

選択した子レベルと、その下のすべての子レベルが「Child Levels」ペインから削除されます。

8 （オプション）論理レベルの説明を入力します。

9 「OK」をクリックします。

ディメンション・レベルとの論理列とそのテーブルの関連付け

すべての論理レベルをディメンション内で作成したら、総計を除く各論理レベルにディメンション・テーブルから列を 1 つ以上ドラッグ・アンド・ドロップする必要があります。列をディメンションに初めてドラッグしたときに、論理テーブルがディメンションに関連付けられます。また、論理列が、ディメンションのそのレベルに関連付けられます。論理列に関連付けられる論理レベルを変更するには、列を別の論理レベルにドラッグします。

注意：ディメンション・テーブルの論理キーを構成する論理列は、ディメンションの最下位レベルに関連付けられている必要があります。

論理列とディメンション・レベルとを関連付けると、その列が存在するテーブルが「Dimensions」ダイアログ・ボックスの「Tables」タブに表示されるようになります。

ディメンションに関連付けられているテーブルを確認するには

1 「Business Model and Mapping」レイヤーで、ディメンションをダブルクリックします。

2 「Dimensions」ダイアログ・ボックスで、「Tables」タブをクリックします。

このディメンションに関連付けられているテーブルが、テーブル・リストに一覧表示されます。このテーブル・リストに含まれるのは、1 つの論理ディメンション・テーブルと、1 つ以上の論理ファクト・テーブル（レベルベースのメジャーを作成した場合）のみです。

3 「OK」または「取消」をクリックして「Dimensions」ダイアログ・ボックスを閉じます。

ディメンション・レベルの主キーの指定

「Logical Level」ダイアログ・ボックスの「Keys」タブを使用して、レベルの主キーを指定します。

ディメンション・レベルの主キーを指定するには

1. 「Business Model and Mapping」レイヤーで、ディメンションを開いてから、そのディメンションの最上位レベル（総計レベル）を開きます。
2. 総計レベルより下にある論理レベルをダブルクリックします。
3. 「Logical Level」ダイアログ・ボックスで、「Keys」タブをクリックします。
4. 「Keys」タブの「Primary key」ドロップダウン・リストで、レベル・キーを選択します。

   注意：存在するレベル・キーが 1 つのみの場合は、デフォルトでそのキーが主キーになります。
5. 列をリストに追加するには、次の手順を実行します。
   a. 「Logical Level」ダイアログ・ボックスで、「New」をクリックします。
   b. 「Logical Level Key」ダイアログ・ボックスで、キーの名前を入力します。
   c. 「Logical Level Key」ダイアログ・ボックスで、列を選択するか「Add」をクリックします。
   d. 「Add」をクリックした場合は、「Browse」ダイアログ・ボックスで列を選択してから「OK」をクリックします。

      選択した列が、「Logical Level Key」ダイアログ・ボックスの「Columns」リストに表示され、チェック・ボックスが自動的に選択されます。
6. 時間ディメンション内のレベルの場合は、時間順キーを選択することと、キーを名前別にソートすることができます。
7. （オプション）キーの説明を入力して、「OK」をクリックします。
8. 他の論理レベルに主キーを追加する場合は、134 ページの手順 2 から 134 ページの手順 7 までの手順を繰り返します。
9. 「Logical Level」ダイアログ・ボックスで、「OK」をクリックします。

時間ディメンションにおける時間順キーの選択とソート

時間ディメンションの 1 つ以上のレベルでは、時間順キーが必要です。どのレベルでも、1 つ以上の時間順キーを選択してから、そのレベルでキーをソートすることができますが、このときに使用されるのは最初の時間順キーのみです。

注意：ディメンションを時間ディメンションとして使用するには、「Dimension」ダイアログ・ボックスの「Time Dimension」チェック・ボックスを選択する必要があります。

時間ディメンションの時間順キーを選択してソートするには

1. 「Business Model and Mapping」レイヤーで、時間ディメンションを開いてから、そのディメンションの最上位レベル（総計レベル）を開きます。

   注意：ディメンションが時間ディメンションとして認識されるようにするには、「Dimension」ダイアログ・ボックスの「Time Dimension」チェック・ボックスを選択する必要があります。
2. 総計レベルより下にある論理レベルをダブルクリックします。
3. 「Logical Level」ダイアログ・ボックスで、「Keys」タブをクリックします。

4 時間順キーを選択するには、「Keys」タブの「Chronological Key」チェック・ボックスを選択します。

5 時間順キーをソートするには、「Keys」タブで時間順キーをダブルクリックします。

6 「Chronological Key」ダイアログ・ボックスで、時間順キーの列を選択し、「Up」または「Down」をクリックして列の順序を並べ替えてから「OK」をクリックします。

優先ドリル・パスへのディメンション・レベルの追加

「Preferred Drill Path」タブを使用すると、Oracle BI Presentation Services ユーザーがデータ・リクエストでドリルダウンしたときに使用されるドリル・パスを指定できます。この方法は、ディメンション・レベル階層によって定義される通常のドリル・パスの外部にあるドリル・パスを指定するときにのみ使用してください。最も一般的な用途は、あるディメンションから別のディメンションへのドリルです。論理レベルをドリル・パスから削除することや、論理レベルの順序をドリル・パスにおいて変更することが可能です。

ディメンション・レベルを優先ドリル・パスに追加するには

1 「Add」ボタンをクリックして「Browse」ダイアログ・ボックスを開きます。このダイアログ・ボックスで、ドリル・パスに追加する論理レベルを選択できます。論理レベルは、現在のディメンションから選択することも、別のディメンションから選択することもできます。

2 「OK」をクリックして「Level」ダイアログ・ボックスに戻ります。

レベルの名前が「Names」ペインに追加されます。

レベルベースのメジャー計算の例

レベルベースのメジャーとは、常に特定のレベルで集計された値のある列のことです。たとえば、ある企業の売上を国、地域および都市をベースにして把握する場合を考えてみます。この場合は、CountryRevenue、RegionRevenue および CityRevenue という列を設定できます。

AllProductRevenue メジャーは、総計レベルにおけるレベルベース・メジャーの例です。レベルベースのメジャーを利用すると、1 回のクエリーで複数の集計レベルのデータを返すことができます。また、売上の占有率を作成するときにも便利です。これは、メジャーをレベルベースのメジャーで除算してパーセンテージを算出することで計算されます。たとえば、営業担当者別の売上を地域別売上で除算すると、地域別売上の全体における各営業担当者別売上の割合を算出することができます。

このような計算を設定するには、Grandtotal（総計）、Country（国）、Region（地域）および City（都市）の各レベルを持つディメンション階層をリポジトリに作成する必要があります。この階層において格納されるメタデータでは、Country と Region における 1 対多関係と、Region と City における 1 対多関係がそれぞれ定義されます。それぞれの国には多数の地域がありますが、1 つの地域が属する国は 1 つのみです。同様に、それぞれの地域には多数の都市がありますが、1 つの都市が属する地域は 1 つだけです。

次に、3 つの論理列（CountryRevenue、RegionRevenue および CityRevenue）を作成する必要があります。この 3 つの列ではそれぞれ、Revenue 論理列をソースとして使用します。Revenue 列にはデフォルトの集計ルールである SUM があり、基礎となるデータベースにソースがあります。

この CountryRevenue、RegionRevenue および CityRevenue の各列を、Country レベル、Region レベルおよび City レベルにそれぞれドラッグします。各クエリーでこのいずれかの列のリクエストがあると、対応するレベルで集計された売上金額が返されます。

136 ページの図 13 に、この例のビジネス・モデルが「Business Model and Mapping」レイヤーにどのように表示されるかを示します。

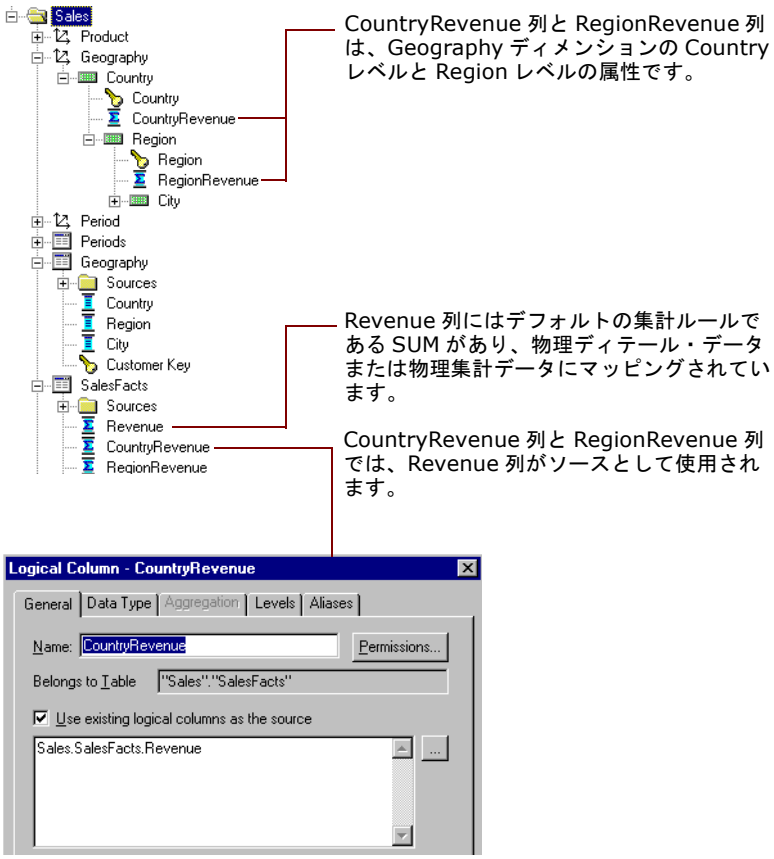


図 13. 「Business Model and Mapping」レイヤーにおけるビジネス・モデルの例

総計ディメンション階層の例

ある製品ディメンション階層に、TotalProducts（総計）、Brands（ブランド）および Products（製品）の各レベルがあると仮定します。また、Revenue（売上）という列には、デフォルトの集計ルールとして Sum が定義されているとします。この場合、AllProductRevenue 論理列を作成できます。この列では、Revenue をソースとして使用します（「Logical Column」ダイアログの「General」タブで指定）。次に、AllProductRevenue を総計レベルにドラッグします。この列を使用する各クエリーでは、すべての製品の売上合計金額が返されます。値は、Brands や Products における絞込み条件に関係なく返されます。他のテーブルの列における絞込み条件がある場合は、クエリーの有効範囲において総計が計算されます。たとえば、クエリーの有効範囲が 1999 年から 2000 年までのデータの場合、製品売上の総計は 1999 年と 2000 年に販売された全製品の合計金額となります。

A、B および C の 3 種類の製品があり、各製品の売上がそれぞれ 100、200 および 300 であると仮定します。このとき、製品売上の総計は 600 になります（各製品売上の合計金額）。この例で説明したとおりにリポジトリが設定されていれば、次に示すクエリーを実行すると、その次に示した結果が返されます。

```
select product, productrevenue, allproductrevenue
from sales_subject_area
where product in 'A' or 'B'
```

PRODUCT	PRODUCTREVENUE	ALLPRODUCTREVENUE
A	100	600
B	200	600

この例では、クエリーにおいて製品に対する絞込み条件に関係なく、AllProductRevenue 列から返される値は常に 600 です。

ディメンションの自動作成

論理ディメンション・テーブルのディメンションが存在しない場合に、そのテーブルから自動的にディメンションを設定することができます。ディメンションを自動的に作成するために、Administration Tool では、論理テーブル・ソースと、それらのソースにおける列マッピングを調べ、論理テーブル・ソースにおける複数の物理テーブル間の結合を使用して、論理レベルとレベル・キーを判断します。したがって、この方法でディメンションを作成するのに最適なタイミングは、すべての論理テーブル・ソースがディメンション・テーブル用に定義された後になります。

次のルールが適用されます。

- 「Create Dimension」が使用可能になるのは、選択した論理テーブルがディメンション・テーブルであり（1:N 論理結合で定義）、このテーブルに関連付けられたディメンションが存在しない場合です。
- 自動的に作成されたディメンションには、論理テーブルと同じ名前が使用され、接尾辞 Dim が追加されます。たとえば、テーブルの名前が Periods の場合、ディメンションの名前は Periods Dim になります。
- 総計レベルの名前は自動的に、論理テーブルの名前に空白と Total を追加した名前になります。たとえば、Periods Dim テーブルの総計レベルは、Periods Total になります。
- 複数のテーブルがソースにある場合は、ソースにある複数のテーブル間の結合関係により、最下位レベルの属性が保持される物理テーブルが決まります。階層の最下位レベルの名前は、論理テーブルの名前に空白と Detail を追加した名前になります。たとえば、Periods テーブルの最下位レベルは、Periods Detail になります。
- ディメンション・テーブルの論理キーは、階層の最下位レベルにマッピングされ、レベル・キーとして指定されます。この論理列は、ディメンション・ソースにおける最下位レベルのテーブルのキー列にマッピングされている必要があります。
  - 複数の物理テーブルが 1 つのソースにある場合は、これらのテーブルのキーにマッピングしている列も論理レベルになります。この追加されたレベル名には、キー列の論理列名が使用されます。
  - 結合の順序によって、論理レベルの階層配置が決まります。この新しい論理レベルのレベル・キーは、ソースにあるテーブルのキーにマッピングされる論理列に設定されます。
- 複数の論理テーブル・ソースがある場合は、属性マッピングと物理結合を使用して物理ソースにあるテーブルの階層順序が決まります。たとえば、3 つのソース（A、B および C）があり、それぞれ物理テーブルが 1 つあると仮定します。また、それぞれ 10 個、15 個および 3 個の属性の属性マッピングがあると仮定します（他の論理列から作成される列は含まれない）。このテーブルに対して自動的に作成されるディメンションは、次のようになります。

- 作成されるディメンションでは、4 つの論理レベル（総計レベルとディテール・レベル）が作成されます。
- ソース B（マッピングされた列の数が最大で、論理テーブル・キーの列マッピングが保持されている）にあるテーブルのキーが、ディテール・レベルのレベル・キーになります。
- ディテール・レベルの親となるのは、ソース A にある物理テーブルのキーにマッピングされる論理列に対して指定された論理レベルです。
- A と B の両方にマッピングされている属性は、レベル A に関連付けられます。
- レベル A の親となるのは、ソース C にある物理テーブルのキーにマッピングされる論理列に対して指定された論理レベルです。
- A と C の両方にマッピングされている列は、レベル C に関連付けられます。

- 物理ソースにおけるテーブル結合は、分割階層となるパターンを示すことがあります。たとえば、Product（製品）テーブルが Flavor（フレーバ）テーブルと Subtype（サブタイプ）テーブルに結合されていると仮定します。この場合、製品ディテール・レベルの親は、1 つのフレーバ・レベルと 1 つのサブタイプ・レベルになります。
- 次に示す場合、ディメンションを自動的に作成することはできません。
  - 結合とレベルがあるディメンションがすでに作成されている場合は、右クリック・メニューに「Create Dimension」は表示されません。
  - 他のテーブルと結合していないテーブルは、ファクト・テーブルとみなされるので、このオプションは使用できません。
- スノーフレーク・スキーマでは、テーブルのソースが 1 つのみの場合にディメンションを自動的に作成すると、子テーブルが自動的に階層に取り込まれます。子テーブルは、総計レベルとディテール・レベルとの間における中間レベルになります。1 つのディメンション・テーブルに複数の子テーブルがある場合は、ディメンションを自動的に作成すると、階層は分割されます。

ディメンションを自動的に作成するには

1. Administration Tool で、リポジトリを開きます。
2. リポジトリの「Business Model and Mapping」レイヤーで、論理テーブルを右クリックします。
3. 右クリック・メニューで「Create Dimension」を選択します。

   ディメンションが「Business Model and Mapping」レイヤーに表示されます。

ディメンション固有の論理列集計ルールの設定

メジャーの大半は、どのディメンションでも集計ルールは同じです。ただし、メジャーによっては、ディメンションごとに集計ルールが異なる場合があります。たとえば、銀行の預金残高の平均は期間ごとに計算し、合計は口座ごとに計算する場合があります。Oracle BI Server では、ディメンション固有の集計ルールを構成できます。あるディメンションで指定した集計ルールとは別の集計ルールを、別のディメンションで指定して適用することができます。

ディメンション固有の集計を設定するには、「Business Model and Mapping」レイヤーでディメンションを構成する必要があります。集計ナビゲーションの設定方法の詳細は、「Oracle BI リポジトリでの集計ナビゲーション用のフラグメンテーション・コンテンツの設定」（205 ページ）を参照してください。

1 つの論理列に対してディメンション固有の集計ルールを指定するには

1. 「Business Model and Mapping」レイヤーで、論理列をダブルクリックします。
2. 「Logical Column」ダイアログ・ボックスで、「Aggregation」タブをクリックします。
3. 「Aggregation」タブで、「Based on dimensions」チェック・ボックスを選択します。

   「Browse」ダイアログ・ボックスが自動的に表示されます。
4. 「Browse」ダイアログ・ボックスで「New」をクリックし、集計するディメンションを選択してから「OK」をクリックします。
5. 「Aggregation」タブの「Formula」ドロップダウン・リストで、ルールを選択します。

   注意：指定したディメンションのルールを選択した後で、残りのディメンションの集計ルールを設定するには「Other」というディメンションを使用します。
6. 複雑な計算式を作成する場合は、「Formula」列の右にある省略記号ボタンをクリックして式ビルダーを開きます。
7. 複数ディメンションがある場合に、ディメンション固有ルールの実行順序を変更するには、「Up」または「Down」をクリックします。

   メジャーを計算すると、ダイアログ・ボックスで確立された順序（上から下）で集計ルールが適用されます。
8. 「OK」をクリックします。

複数の論理ファクト列に対してディメンション固有の集計ルールを指定するには

1. 「Business Model and Mapping」レイヤーで、複数の論理ファクト列を選択します。
2. 右クリックして「Set Aggregation」を選択します。

   ファクト列に集計ルールがある場合は、メニューに「Set Aggregation」は表示されません。
3. 「Aggregation」ダイアログ・ボックスで「All columns the same」チェック・ボックスを選択するか、そのチェック・ボックスの選択を解除します。

   デフォルトでは、このチェック・ボックスが選択されています。選択すると、選択したすべての列に適用される集計ルールを設定できます。このチェック・ボックスの選択を解除した場合は、選択した列ごとに別の集計ルールを設定することができます。
4. 「Aggregation」タブで、「Use advanced options」チェック・ボックスを選択します。
5. 「Browse」ダイアログ・ボックスで、集計を実行するディメンションを選択してから「OK」をクリックします。

   特定のディメンションのルールを設定した後で、他のディメンションの集計ルールを指定するには「Other」というエントリを使用します。
6. 「Formula」列の右にある省略記号ボタンをクリックします。

7 「Expression Builder - Aggregate」ダイアログ・ボックスの「Formula」ドロップダウン・リストで、ディメンションに対して実行する集計を選択します。

8 ディメンション固有ルールの実行順序を変更するには、「Up」または「Down」をクリックしてから、「OK」をクリックします。

メジャーを計算すると、ダイアログ・ボックスで確立された順序（上から下）で集計ルールが適用されます。

「Business Model and Mapping」レイヤーにおける表示フォルダの設定

Oracle BI 管理者は、表示フォルダを作成して「Business Model and Mapping」レイヤーでオブジェクトを整理することができます。メタデータ的な意味はありません。表示フォルダを作成すると、選択したテーブルやディメンションがフォルダにショートカットとして表示され、ビジネス・モデル・ツリーではオブジェクトとして表示されます。表示フォルダにはショートカットのみが表示されるように、オブジェクトを非表示にできます。これらのオブジェクトを非表示にする方法の詳細は、「「Options」ダイアログ・ボックスにおける「Repository」タブの使用」（33 ページ）を参照してください。

注意：オブジェクトを表示フォルダで削除しても、削除されるのはそのオブジェクトのショートカットのみです。

論理表示フォルダを設定するには

1 「Business Model and Mapping」レイヤーで、ビジネス・モデルを右クリックして、「New Object」→「Logical Display Folder」を選択します。

2 「Logical Display Folder」ダイアログ・ボックスの「Tables」タブで、フォルダの名前を入力します。

3 テーブルを表示フォルダに追加するには、次の手順を実行します。

   a 「Add」をクリックします。

   b 「Browse」ダイアログ・ボックスで、フォルダに追加するファクト・テーブルまたはディメンション・テーブルを選択してから、「Select」をクリックします。

4 ディメンションを表示フォルダに追加するには、「Dimensions」タブをクリックしてから、次に示す手順を実行します。

   a 「Add」をクリックします。

   b 「Browse」ダイアログ・ボックスで、フォルダに追加するディメンションを選択してから、「Select」をクリックします。

5 「OK」をクリックします。

論理結合の定義

複数の論理テーブルは、互いに関連付けられています。複数のテーブル間における関係は、論理結合として表現されます。論理結合の重要なプロパティの 1 つが、カーディナリティです。カーディナリティとは、あるテーブルの行とその結合先テーブルの行との間における関係を表すものです。1 対多のカーディナリティは、第 1 の論理ディメンション・テーブルの各行に対応する第 2 の論理テーブルの行の行数が、0、1 または多数であることを示します。Administration Tool で、テーブルが論理ファクト・テーブルとみなされるのは、そのテーブルと他の論理テーブルを結ぶすべての論理結合において、そのテーブルが多の側にある場合です。

論理テーブル結合の指定が必要になるのは、ビジネス・モデルに対する論理リクエストを物理データソースに対する SQL クエリーに変換するために必要なメタデータを、Oracle BI Server で用意できるようにするためです。論理結合情報によって、複数の論理テーブル間における多対 1 関係があることを Oracle BI Server に示すことになります。この論理結合情報は、基礎となるデータベースに対するクエリーを Oracle BI Server が生成するときに使用されます。

論理レイヤーと物理レイヤーとの間における結合が自動的に作成されるのは、次の両方が真である場合です。

- 必要なすべての物理テーブルを同時に「Business Model and Mapping」レイヤーにドラッグ・アンド・ドロップして論理テーブルを作成した場合。
- 論理結合が、「Physical」レイヤーにおける結合と同じ場合。

ただし、多くの場合は「Business Model and Mapping」レイヤーにおいて論理結合を作成することが必要になります。すべての物理テーブルを同時にドラッグ・アンド・ドロップすることは、非常に単純なモデルの場合を除いて、ほとんどないからです。「Business Model and Mapping」レイヤーで、キー結合や外部キー結合ではなく、1 対多関係を持つ複合結合を作成してください。

論理外部キーや論理複合結合を作成するには、Joins Manager または Business Model Diagram を使用できます。「Physical」レイヤーで複合結合を作成する場合は、式を指定することや、結合を作成する対象となる列を指定することができます。「Business Model and Mapping」レイヤーで複合結合を作成する場合は、式を指定することや、結合を作成する対象となる列を指定することはできません。「Physical」レイヤーに結合が存在している場合に、対応する結合が「Business Model and Mapping」レイヤーに存在する必要はありません。

警告：論理テーブルでは外部キーを使用しないことをお薦めします。ただし、論理外部キーや論理複合結合を作成するには、Joins Manager または Business Model Diagram を使用できます。ファクト・テーブルの論理キーは、属性テーブルに結合しているキー列で構成されている必要があります。

論理結合を作成するには、次の作業を実行します。

- Joins Manager による論理結合の定義（141 ページ）
- Business Model Diagram による論理結合の定義（143 ページ）

Joins Manager による論理結合の定義

Joins Manager を使用すると、論理結合関係を確認することや、論理外部キーと複合結合を作成することができます。

この項の内容は次のとおりです。

- 論理外部キーの作成（142 ページ）
- 論理複合結合の作成（142 ページ）

論理外部キーの作成

論理外部キーの結合が必要になる場合は、サード・パーティ製クエリーとレポート・ツール用の ODBC データソースとして Oracle BI Server が使用される場合です。通常、論理外部キーの作成は不要です。Administration Tool にこの機能が用意されているのは、以前のリリースとの互換性を確保するためです。

論理外部キーを作成するには

1. Administration Tool のツールバーで、「Manage」→「Joins」を選択します。

   「Joins Manager」ダイアログ・ボックスが表示されます。

2. 「Action」→「New」→「Logical Foreign Key」を選択します。
3. 「Browse」ダイアログ・ボックスで、テーブルをダブルクリックします。

   「Logical Foreign Key」ダイアログ・ボックスが表示されます。

4. 外部キーの名前を入力します。
5. ダイアログ・ボックスの左側にある「Table」ドロップダウン・リストで、外部キーによって参照されるテーブルを選択します。
6. 外部キーが参照する列を左側のテーブルで選択します。
7. 右側のテーブルにおいて外部キー列で構成される列を選択します。
8. （オプション）キーの駆動テーブルを指定するには、「Driving」ドロップダウン・リストでテーブルを選択してから、適切なカーディナリティを選択します。

   これは、複数のデータベースにおける内部結合で一方のテーブルが非常に小さく他方が非常に大きい場合における Oracle BI Server の処理を最適化するために使用されます。複数のデータベースにおける結合が実行される場合を除いて、駆動テーブルを選択しないでください。駆動テーブルの詳細は、「駆動テーブルの指定」（144 ページ）を参照してください。

   警告：駆動テーブルを指定するかどうかを決める際は注意が必要です。駆動テーブルがクエリー最適化に使用されるのは、非常にまれな状況のみで、駆動テーブルが非常に小さい場合です（1,000 行未満）。駆動テーブルを適切に選択しないと、パフォーマンスが大きく低下する場合があります。

9. 「Type」ドロップダウン・リストで結合タイプを選択します。
10. 式ビルダーを開くには、「Expression」ペインの横にあるボタンをクリックします。

    式が「Expression」ペインに表示されます。

11. 「OK」をクリックして、作業内容を保存します。

論理複合結合の作成

論理キー結合と外部キー結合よりも、論理複合結合の使用をお薦めします。

論理複合結合を作成するには

1. Administration Tool のツールバーで、「Manage」→「Joins」を選択します。

   「Joins Manager」ダイアログ・ボックスが表示されます。

2 「Action」→「New」→「Logical Complex Join」を選択します。

「Logical Join」ダイアログ・ボックスが表示されます。

3 複合結合の名前を入力します。

4 ダイアログ・ボックスの左右両側にある「Table」ドロップダウン・リストで、複合結合によって参照されるテーブルを選択します。

5 （オプション）キーの駆動テーブルを指定するには、「Driving」ドロップダウン・リストでテーブルを選択してから、適切なカーディナリティを選択します。

これは、複数のデータベースにおける内部結合で一方のテーブルが非常に小さく他方が非常に大きい場合における Oracle BI Server の処理を最適化するために使用されます。複数のデータベースにおける結合が実行される場合を除いて、駆動テーブルを選択しないでください。駆動テーブルの詳細は、「駆動テーブルの指定」（144 ページ）を参照してください。

警告：駆動テーブルを指定するかどうかを決める際は注意が必要です。駆動テーブルがクエリー最適化に使用されるのは、非常にまれな状況のみで、駆動テーブルが非常に小さい場合です（1,000 行未満）。駆動テーブルを適切に選択しないと、パフォーマンスが大きく低下する場合があります。

6 「Type」ドロップダウン・リストで結合タイプを選択します。

7 「OK」をクリックします。

Business Model Diagram による論理結合の定義

Business Model Diagram には、複数の論理テーブルと、複数のテーブル間において定義されている結合が表示されます。論理外部キーの結合が必要になる場合は、サード・パーティ製クエリーとレポート・ツール用の ODBC データソースとして Oracle BI Server が使用される場合です。

注意：Business Model Diagram は、複合結合の定義に使用することをお薦めします。また、Business Model Diagram は、論理外部キーの作成に使用しないことをお薦めします。

Business Model Diagram を表示するには

1 Administration Tool で、ビジネス・モデルを右クリックしてから、「Business Model Diagram」→「Whole Diagram」を選択します。

2 Administration Tool のツールバーにある次に示すボタンのいずれかをクリックします。

- 「New complex join」（推奨）
- 「New foreign key」（推奨しない）：この機能が用意されている理由は、以前のリリースとの互換性を確保するためです。

3 ボタンのいずれかが選択された状態で、結合の第 1 テーブル（1 対多結合の 1）にカーソルを移動します。

4 左クリックしてから、結合するテーブル（1 対多結合の多）にカーソルを移動し、第 2 テーブルを左クリックします。

「Logical Foreign Key」ダイアログ・ボックスまたは「Logical Join」ダイアログ・ボックスが表示されます。

5 論理外部キーの場合は、結合列を左右のテーブルで選択します。

6 （オプション）キーの駆動テーブルを指定するには、「Driving」ドロップダウン・リストでテーブルを選択してから、適切なカーディナリティを選択します。

これは、複数のデータベースにおける内部結合で一方のテーブルが非常に小さく他方が非常に大きい場合における Oracle BI Server の処理を最適化するために使用されます。複数のデータベースにおける結合が実行される場合を除いて、駆動テーブルを選択しないでください。駆動テーブルの詳細は、「駆動テーブルの指定」（144 ページ）を参照してください。

警告：駆動テーブルを指定するかどうかを決める際は注意が必要です。駆動テーブルがクエリー最適化に使用されるのは、非常にまれな状況のみで、駆動テーブルが非常に小さい場合です（1,000 行未満）。駆動テーブルを適切に選択しないと、パフォーマンスが大きく低下する場合があります。

7 「Type」ドロップダウン・リストで結合タイプを選択します。

8 式ビルダーを開くには、「Expression」ペインの横にあるボタンをクリックします。

列、指定述語および演算子のみが、式の中で使用できます。詳細は、「式ビルダー」（195 ページ）を参照してください。

9 「OK」をクリックして、作業内容を保存します。

駆動テーブルの指定

論理結合の駆動テーブルは「Logical Joins」ウィンドウで指定できます。駆動テーブルは、異なるデータベース間の結合で一方のテーブルが非常に小さく他方が非常に大きい場合における Oracle BI Server の処理を最適化するために使用されます。駆動テーブルを指定した場合、駆動テーブルから選択される行の行数が、結合先のテーブルの行の行数を大幅に下回る場合にのみ、クエリーの最適化が行われます。

警告：問題を回避するには、駆動テーブルが非常に小さい（1,000 行未満）場合にのみ駆動テーブルを指定してください。

駆動テーブルを指定した場合、Oracle BI Server によって駆動テーブルが使用されるのは、駆動テーブルの使用によってクエリー処理が最適化されるとクエリー計画で判断される場合です。小さいテーブル（駆動テーブル）がスキャンされ、パラメータ化されたクエリーが大きいテーブルに対して発行され、一致する行が選択されます。その後で、他のテーブル（他の駆動テーブルも含む）が結合されます。

警告：駆動テーブルから選択される行の行数が多いと、駆動テーブルを指定するとパフォーマンスが大きく低下する場合があります。また、MAX_QUERIES_PER_DRIVE_JOIN の制限を超えると、クエリーが終了します。

一般的に、駆動テーブルは内部結合とともに使用できます。外部結合の場合は、左側の外部結合では駆動テーブルを左テーブルとして、右側の外部結合では右テーブルとして使用します。駆動テーブルは、完全外部結合では使用されません。駆動テーブルの指定方法の詳細は、「論理結合の定義」（141 ページ）を参照してください。

駆動テーブルのパフォーマンスを制御およびチューニングするための 2 つのエントリが、データベース機能テーブルにあります。

- MAX_PARAMETERS_PER_DRIVE_JOIN

  これは、パフォーマンス・チューニング用のパラメータです。一般的に、この値が大きいほど、生成が必要なパラメータ化されたクエリーの数は少なくなります。値が大きすぎると、バックエンド・データベースの制限が原因でパラメータ化されたクエリーに失敗することがあります。値を 0 に設定すると、駆動テーブルの結合は行われなくなります。

- MAX_QUERIES_PER_DRIVE_JOIN

  リソース集中型の駆動テーブル結合を防止するために使用します。パラメータ化されたクエリーの数がこの値を超えると、クエリーは終了し、エラー・メッセージがユーザーに返されます。

論理オブジェクトにマッピングする物理テーブルの指定

「Physical Diagram」には、選択されている論理オブジェクトにマッピングしている物理テーブルや、各テーブルの間の物理結合が表示されます。

結合オプションの 1 つである「Object(s) and Direct Joins within Business Model」は、論理レイヤーに固有のものです。これによって、次の条件を両方とも満たすテーブルの物理ダイアグラムが作成されます。

- 選択されているオブジェクトのテーブルおよび直接結合するテーブル
- ビジネス・モデルでマッピングされている（ビジネス・モデルの論理テーブル・ソースに存在する）テーブル

論理オブジェクトの物理ダイアグラムを開くには

1. 「Business Model and Mapping」レイヤーで、ビジネス・モデル、論理テーブルまたは論理テーブル・ソースを右クリックします。
2. 「Physical Diagram」を選択してから、結合オプションのいずれかを選択します。
3. オブジェクトをクリックしてドラッグすると、1 対多などのリレーションシップ線が明確に表示されます。

6 Oracle BI リポジトリの「Presentation」レイヤーの作成とメンテナンス

この章は、リポジトリを計画および設定するためのロードマップの一部です。詳細は、「Oracle BI リポジトリの計画と作成」（41 ページ）を参照してください。

「Business Model and Mapping」レイヤーの作成後は、そのレイヤーを Administration Tool の「Presentation」レイヤーにドラッグ・アンド・ドロップできます。「Business Model and Mapping」レイヤーの詳細は、「Oracle BI リポジトリの「Business Model and Mapping」レイヤーの作成と管理」（113 ページ）を参照してください。

この章では、リポジトリの「Presentation」レイヤーで、Administration Tool を使用してオブジェクトを作成および編集する手順について説明します。これは、リポジトリの設定における 4 番目の手順です。

この章の内容は次のとおりです。

リポジトリでの「Presentation」レイヤーの作成

「Presentation」レイヤーでは、ユーザーに対してビジネス・モデルのカスタマイズされたビューを表示できます。「Presentation」レイヤー内の「Presentation Catalogs」（Oracle Answers のサブジェクト領域）は、Oracle BI Presentation Services ユーザーにはビジネス・モデルとして表示されますが、Oracle BI Server を ODBC データソースとして使用するクライアント・ツールには、カタログとして表示されます。次の項では、「Presentation」レイヤーの作成手順について説明します。

注意：オフラインで編集する際には、リポジトリを随時保存してください。ビジネス・モデルに一貫性がない場合でも、リポジトリはオフライン・モードで保存できます。

ユーザーに公開するビジネス・モデルのコピー

「Presentation」レイヤーで「Presentation Catalog」を作成する方法は複数あります。その中で推奨されるのは、ビジネス・モデルを「Business Model and Mapping」レイヤーから「Presentation」レイヤーにドラッグ・アンド・ドロップして、ユーザーに何を表示するかに応じて「Presentation」レイヤーを変更する方法です。この方法では、プレゼンテーション・テーブル間で列を移動したり、ユーザーに表示する必要のない列を削除することができます。また、単一のプレゼンテーション・テーブルにすべてのデータを表示することもできます。プレゼンテーション・テーブルを作成すると、ユーザーにとって意味のある方法でメジャーを編成および分類することが可能です。

不必要または余分な列の削除

カスタムの「Presentation」レイヤーを使用する重要な理由の 1 つに、スキーマをできるだけ使いやすく、かつ理解しやすくするという目的があります。したがって、ユーザーにとって意味のない列は非表示にする必要があります。「Presentation」レイヤーから削除できる列の例を次に示します。

- ビジネス上の意味がないキー列
- ユーザーに表示する必要のない列（たとえば、テキスト説明が存在する場合のコード）
- ユーザーが参照権限のない列

注意：セキュリティ・レイヤー内のテーブルまたは列へのアクセスも制限することができます。詳細は、第 15 章「Oracle BI におけるセキュリティ」を参照してください。

プレゼンテーション列名のわかりやすい名前への変更

デフォルトでは、プレゼンテーション列は、「Business Model and Mapping」レイヤー内の対応する論理列と同じ名前になります。ただし、「Presentation Column」ダイアログ・ボックスで名前を変更することにより、別の名前を表示するように指定できます。プレゼンテーション列では、名前を変更するたびに古い名前の別名が自動的に作成されるため、古い名前との互換性は維持されます。

プレゼンテーション・カタログの論理キーのエクスポート

「Presentation」レイヤー内のプレゼンテーション・カタログごとに、アクセスするツールのキー列として論理キーをエクスポートするかどうかを「Presentation Catalog」ダイアログ・ボックスで決定します。論理キーのエクスポートは、Oracle BI Presentation Services ユーザーにとって関係のない操作ですが、一部のクエリー・ツールとレポート・ツールにはメリットがあります。論理キーをエクスポートする場合は、テーブル・フォルダに論理キー列が存在することを確認してください。この場合、ビジネス・モデルで論理キーと外部キーの結合を使用する必要があります。

「Presentation Catalog」ダイアログ・ボックスで「Export logical keys」オプションを選択すると、「Business Model and Mapping」レイヤーのキー列になっている「Presentation」レイヤーの列が、ODBC クライアントのキー列として一覧表示されます。これは、デフォルトの選択です。ほとんどの場合は、このオプションを選択します。

注意： Microsoft Access など、パラメータ化された SQL クエリーを発行するツールを使用している場合は、「Export logical keys」オプションを選択しないでください。これにより、ツールはパラメータ化されたクエリーを発行しなくなります。

「Presentation」レイヤーのオブジェクト

「Presentation」レイヤーは、「Business Model and Mapping」レイヤーの抽象化のレベルを上げたものです。このレイヤーは、クライアント・ツールおよびアプリケーション側から見たデータ・ビューを表します。

「Presentation」レイヤーは、「Business Model and Mapping」レイヤーをエンド・ユーザー用にさらに簡略化またはカスタマイズするための手段です。たとえば、キー列を非表示にしたり、スキーマを単一テーブルとして表示することができます。データ・ビューをユーザー用に簡略化することにより、ユーザーのビジネス・ニーズに基づいてクエリーを作成することが容易になります。

この項では、Administration Tool の「Presentation layer」ダイアログ・ボックスを使用してリポジトリ・オブジェクトを作成および編集する手順について説明します。

この項の内容は次のとおりです。

プレゼンテーション・カタログでの操作

「Presentation」レイヤーでは、プレゼンテーション・カタログ（サブジェクト領域）を使用して、異なるユーザー・セットに対してビジネス・モデルの異なるビューを表示できます。プレゼンテーション・カタログには、単一のビジネス・モデルの内容を移入する必要があります。複数のビジネス・モデルにわたって移入することはできません。

プレゼンテーション・カタログの作成時に「Export logical keys」オプションを選択すると、「Business Model and Mapping」レイヤーのキー列になっている「Presentation」レイヤーの列が、ODBC クライアントのキー列として一覧表示されます。これは、デフォルトの選択です。ほとんどの場合は、このオプションを選択します。多くのクライアント・ツールではキー列と非キー列が区別されており、「Export logical keys」オプションによって、キー列のメタデータにアクセスできます。ただし、クライアント・ツールでクエリーに追加した結合条件は無視されるため、Oracle BI Server では、リポジトリで定義されている結合条件が使用されます。

暗黙的なファクト列を設定すると、この列に 2 つ以上のディメンション・テーブルからの列が含まれていて、かつメジャーを含まない場合に、列がクエリーに追加されます。この列は、結果には表示されません。これは、複数の代替パスがあるときに、ディメンション・テーブルの間のデフォルトの結合パスを指定するために使用されます。

「Presentation Catalog」ダイアログ・ボックスには、「General」、「Presentation Tables」および「Aliases」という 3 つのタブがあります。次の表に、各タブの機能を示します。

タブ	コメント
General	プレゼンテーション・カタログを作成または編集するために使用します。
Presentation Table	Administration Tool ワークスペースで「Presentation」レイヤー・テーブルの並替えやソート、およびテーブルの削除を行うために使用します。また、「Presentation Table」ダイアログ・ボックスにアクセスし、テーブルを作成および編集することもできます。
Aliases	カタログ・フォルダの別名を指定または削除するために使用します。

プレゼンテーション・カタログを作成するには

1. 「Presentation」レイヤーで右クリックし、「New Presentation Catalog」を選択します。
2. 「Presentation Catalog」ダイアログ・ボックスの「General」タブで、プレゼンテーション・カタログの名前を入力してから「Permissions」をクリックします。
3. 「Permissions」ダイアログ・ボックスで、カタログ・フォルダにユーザー権限またはグループ権限を割り当て、「OK」をクリックします。

   プレゼンテーション・カタログへの権限の割当て方法の詳細は、「リポジトリ・オブジェクトに対する権限の設定」（35 ページ）を参照してください。
4. 「Presentation Catalog」ダイアログ・ボックスで、「Business Model」ドロップダウン・リストからビジネス・モデルを選択します。

   プレゼンテーション・カタログに列を追加すると、ドロップダウン・リストは無効になります。各プレゼンテーション・カタログでは、1 つのビジネス・モデルからのみ列を追加できるからです。
5. 他のアプリケーションに論理キーを公開するには、「Export logical keys」オプションを選択します。

   注意： Microsoft Access など、パラメータ化された SQL クエリーを発行するツールを使用している場合は、「Export logical keys」オプションを選択しないでください。論理キーをエクスポートしないことにより、ツールはパラメータ化されたクエリーを発行しなくなります。

6 （オプション）カタログ・フォルダの説明を入力します。

この説明は、Oracle Business Intelligence Answers でプレゼンテーション列にマウス・ポインタを置く際、ツールチップに表示されます。

警告： プレゼンテーション・カタログ・フォルダに列を移動する際は、カタログに同じ名前または同じ別名を持つ列が存在していないことを確認してください。

7 「Implicit Fact Column」を設定します。

8 「OK」をクリックします。

プレゼンテーション・テーブルでの操作

プレゼンテーション・テーブルを使用すると、ユーザー・コミュニティにとって意味のあるカテゴリに列を編成できます。「Presentation」レイヤーのプレゼンテーション・テーブルには列が含まれます。プレゼンテーション・テーブルには、1 つ以上の論理テーブルからの列を含められます。プレゼンテーション・テーブルの名前とオブジェクト・プロパティは、論理テーブル・プロパティに依存しません。

「Presentation Tables」ダイアログ・ボックスには、「General」、「Columns」および「Aliases」という 3 つのタブがあります。次の表に、各タブの機能を示します。

タブ	コメント
General	プレゼンテーション・テーブルを作成または編集するために使用します。
Columns	Administration Tool ワークスペースで「Presentation」レイヤー列の並替えやソート、および列の削除を行うために使用します。また、「Presentation Column」ダイアログ・ボックスにアクセスし、列を作成および編集することもできます。
Aliases	プレゼンテーション・テーブルの別名を指定または削除するために使用します。

プレゼンテーション・テーブルを作成するには

1 「Presentation」レイヤーのカタログ・フォルダを右クリックし、ショートカット・メニューから「New Presentation Table」を選択します。

「Presentation Table」ダイアログ・ボックスが表示されます。

2 「General」タブで、テーブルの名前を指定します。

3 「Permissions」ボタンをクリックして「Permissions」ダイアログ・ボックスを開き、テーブルのユーザー権限またはグループ権限を割り当てます。

プレゼンテーション・テーブルへの権限の割当て方法の詳細は、「リポジトリ・オブジェクトに対する権限の設定」（35 ページ）を参照してください。

4 （オプション）テーブルの説明を入力します。

注意： Answers でネストされたフォルダを表すには、ネストされるプレゼンテーション・フォルダ名の前にハイフンとスペースを付けてから（- <*folder name*>）、ネストするフォルダの後ろに置きます。たとえば、Facts フォルダに Sales Facts フォルダをネストする場合は、メタデータ内で Facts フォルダのすぐ後ろに Sales Facts フォルダを置いてから、その名前を - Sales Facts に変更します。Answers の左ペインにフォルダ名が表示される際には、フォルダ名からハイフンとスペースが省略されます。Facts フォルダに 2 つ目のフォルダ（Marketing Facts など）をネストする場合は、その名前を - Marketing Facts に変更してから、Sales Facts フォルダのすぐ後ろに置きます。事前構成済の標準リポジトリには、確認用のサンプルがほかにも用意されています。

プレゼンテーション・テーブルを削除するには

1 「Presentation」レイヤーで、カタログを右クリックしてから「Properties」を選択します。

2 「Presentation Catalog」ダイアログ・ボックスで、「Presentation Tables」タブをクリックします。

3 「Presentation Tables」タブで、テーブルを選択してから「Remove」をクリックします。

確認メッセージが表示されます。

4 「はい」をクリックしてテーブルを削除します。または、「いいえ」をクリックしてテーブルをカタログ内に残します。

5 「OK」をクリックします。

プレゼンテーション・カタログで、テーブルの並替えまたはすべてのテーブルをソートするには

1 「Presentation」レイヤーで、カタログを右クリックしてから「Properties」を選択します。

2 「Presentation Catalog」ダイアログ・ボックスで、「Presentation Tables」タブをクリックします。

3 テーブルを移動するには、次の手順を実行します。

a 「Presentation Tables」タブの「Name」リストで、並べ替えるテーブルを選択します。

b ドラッグ・アンド・ドロップでテーブルを再配置するか、「Up」および「Down」ボタンをクリックします。

4 すべてのテーブルを英数字順にソートするには、「Name」列見出しをクリックします。

ここをクリックするたびに、英数字のソートが昇順または降順に切り替わります。

プレゼンテーション列での操作

プレゼンテーション列名は、デフォルトでは、「Business Model and Mapping」レイヤーの論理列名と同じ名前になります。ただし、「Presentation Column」ダイアログ・ボックスの「Use Logical Column Name」および「Display Custom Name」チェック・ボックスの選択を両方とも解除することにより、別の名前を表示できます。

エンド・ユーザーに適した編成を実現するために、「Business Model and Mapping」レイヤーの単一の論理テーブルから複数のプレゼンテーション・テーブルへ列をドラッグ・アンド・ドロップできます。これにより、ユーザーにとって意味のあるカテゴリを作成できます。たとえば、異なるメジャー・クラスを含む複数のプレゼンテーション・テーブルを作成できます（出来高メジャー、株式メジャー、1 年前のメジャーを含む各種のテーブルなど）。

「Presentation Column」ダイアログ・ボックスには、次のタブがあります。

- **General**: プレゼンテーション列を作成または編集するために使用します。
- **Aliases**: プレゼンテーション列の別名を指定または削除するために使用します。

プレゼンテーション列を作成するには

1. 「Presentation」レイヤーでプレゼンテーション・テーブルを右クリックし、「New Presentation Column」を選択します。
2. プレゼンテーション列に論理列の名前を使用するために、「Presentation Column」ダイアログ・ボックスで、「Use Logical Column」チェック・ボックスを選択します。

   「Business Model and Mapping」レイヤーにおける列の名前と関連するパスが、「Logical Column Name」フィールドに表示されます。
3. 「Logical Column Name」とは異なる名前を指定する場合、「Use Logical Column」チェック・ボックスの選択を解除し、列の名前を入力します。
4. 列にユーザー権限またはグループ権限を割り当てる場合、「Permissions」をクリックします。
5. 「Permissions」ダイアログ・ボックスで権限を割り当てて、「OK」をクリックします。

   権限の割当て方法の詳細は、「リポジトリ・オブジェクトに対する権限の設定」（35 ページ）を参照してください。
6. 論理列を選択する場合、「Browse」をクリックします。
7. 「Browse」ダイアログ・ボックスで列を選択し、「Select」をクリックします。
8. （オプション）プレゼンテーション列の説明を入力します。
9. 論理列に別名を定義する場合、「Aliases」タブをクリックします。

プレゼンテーション列を編集するには

1. 「Business Model and Mapping」レイヤーで、プレゼンテーション列をダブルクリックします。
2. 「Presentation Column」ダイアログ・ボックスで、「Edit」をクリックします。
3. 「Logical Column」ダイアログ・ボックスで変更を加えるか、または情報を確認してから、「OK」をクリックします。

プレゼンテーション列を削除するには

1. 「Presentation」レイヤーでプレゼンテーション・テーブルを右クリックし、「Properties」を選択します。
2. 「Columns」タブをクリックします。
3. 削除する列を選択します。
4. 「Remove」をクリックするか、または [Delete] キーを押してから「はい」をクリックします。

プレゼンテーション列を並べ替えるには

1. 「Presentation」レイヤーでプレゼンテーション・テーブルを右クリックし、「Properties」を選択します。
2. 「Columns」タブをクリックします。
3. 並べ替える列を選択します。
4. ドラッグ・アンド・ドロップで列を再配置するか、「Up」および「Down」ボタンをクリックします。
5. 「OK」をクリックします。

「Presentation Layer」ダイアログ・ボックスでの「Alias」タブの使用

「Alias」タブは、「Presentation Catalog」、「Presentation Table」および「Presentation Column」ダイアログ・ボックスに表示されます。このタブを使用すると、「Presentation」レイヤーのオブジェクトの別名を指定または削除できます。

別名を追加するには

1. プレゼンテーション・カタログをダブルクリックします。
2. 「Presentation Layer」ダイアログ・ボックスで、「Aliases」タブをクリックします。
3. 新規ボタンをクリックしてから、別名に使用するテキスト文字列を入力します。
4. 「OK」をクリックします。

別名を削除するには

1. プレゼンテーション・カタログをダブルクリックします。
2. 「Presentation Layer」ダイアログ・ボックスで、「Aliases」タブをクリックします。
3. 「Aliases」リストで、削除する別名を選択します。
4. 削除ボタンをクリックし、「OK」をクリックします。

プレゼンテーション・テーブルからの XML ファイルの生成

Oracle BI プレゼンテーション・テーブルの構造は、Oracle Siebel Tools にインポートできます。そのためには、Oracle BI リポジトリの「Presentation」レイヤーのテーブルから XML ファイルを作成し、その XML ファイルを Oracle Siebel Tools にインポートします。

詳細は、Oracle Siebel Tools に関するドキュメントを参照してください。

7 Oracle BI リポジトリ・ファイルの設定完了と管理

この章は、リポジトリを計画および設定するためのロードマップの一部です。詳細は、「Oracle BI リポジトリの計画と作成」（41 ページ）を参照してください。リポジトリ・ファイル、「Physical」レイヤー、「Business Model and Mapping」レイヤー、および「Presentation」レイヤーの作成後は、いくつかの作業を実行して、初期のリポジトリ設定を完了する必要があります。この章では、この設定手順とリポジトリ・ファイルの管理について説明します。

この章の内容は次のとおりです。

リポジトリ・ファイルの設定完了のプロセス

リポジトリ・ファイルの設定を完了するには、次の作業を行います。

リポジトリの保存と一貫性のチェック

オフラインで編集する際には、リポジトリを随時保存してください。ビジネス・モデルに一貫性がない場合でも、リポジトリはオフライン・モードで保存できます。

ビジネス・モデルの一貫性が保たれていることを確認するには、「Check Consistency」コマンドを使用して、コンパイル・エラーを確認します。リポジトリ全体のエラーを確認する場合は、「File」→「Check Global Consistency」コマンド、特定の論理ビジネス・モデルのエラーを確認する場合は、ビジネス・モデルを選択してから右クリック・メニューの「Check Consistency」コマンドを使用します。

一貫性チェックでは、リポジトリを分析して特定のエラーと非一貫性を検出します。たとえば、一貫性チェックでは、論理ソースが構成されていない論理テーブルや物理ソースにマップされていない論理列の検出、未定義の論理結合条件のチェック、ビジネス・モデルの参照先である物理テーブルがそのビジネス・モデルの他の参照先テーブルに結合されていないかどうかの判断、および各ビジネス・モデルにおけるプレゼンテーション・カタログの有無のチェックが実行されます。

注意：一貫性チェックをパスしてもビジネス・モデルが正常に構成されていることは保証されませんが、一般的な問題の多くが除外されます。

一貫性チェックの実行中に発生するエラーや警告は、ダイアログ・ボックスに表示されます。エラー修正、一貫性チェックの再実行という処理を、エラーがなくなるまで繰り返してください。エラー・メッセージには、修正が必要な問題が示されます。警告メッセージでは、潜在的な問題が Oracle BI 管理者に通知されます。「リポジトリまたはビジネス・モデルの一貫性のチェック」（28 ページ）を参照してください。

注意：一貫性チェックのアルゴリズムは、Siebel Business Analytics 7.8.2 で拡張されました。そのため、以前のソフトウェア・バージョンからのアップグレード後にリポジトリの一貫性チェックを実行すると、以前の一貫性チェックにはなかったメッセージが表示される場合があります。これは通常、新しいエラーが発生したのではなく、アップグレード前には検出されなかった非一貫性が存在することを示すものです。

NQSConfig.INI ファイルへのエントリの追加

リポジトリを構築して一貫性が保証された後は、そのリポジトリの NQSConfig.INI ファイルにエントリを追加する必要があります。このエントリにより、Oracle BI Server が起動時にリポジトリをメモリーにロードできるようになります。したがって、変更内容をただちに有効にする場合は、Oracle BI Server を再起動します。NQSConfig.INI ファイルは次の場所にあります。

[*drive path*]:¥OracleBI¥Config¥

ファイルでエラーが発生すると、サーバーを再起動できない場合があります。次の場所にあるログ（NQServer.log）でログ・メッセージを確認できます。

[*drive path*]:¥OracleBI¥server¥Log¥

Oracle Application Server を使用する組織では、Oracle Application Server Control を使用して構成ファイルを変更することをお薦めします。その他のアプリケーション・サーバーを使用する組織では、JConsole を使用してください。詳細は、『Oracle Business Intelligence Infrastructure インストレーションおよび構成ガイド』を参照してください。

警告： Windows のメモ帳などのエディタの使用は可能ですが、お薦めしません。

NQSConfig.INI ファイルにエントリを追加するには

1 メモ帳などのエディタで、NQSConfig.INI ファイルを開きます。

2 リポジトリ・セクションで、新しいリポジトリのエントリを次の形式で追加します。

```
logical_name = repository_file_name ;
```

たとえば、リポジトリ・ファイルの名前が northwind.rpd で、割り当てる論理名が star の場合、エントリは次のようになります。

```
star = northwind.rpd ;
```

リポジトリの 1 つをデフォルトに指定する必要がありますが、同じリポジトリ名を使用する場合、エントリは次のようになります。

```
star = northwind.rpd, default;
```

論理名は、Oracle BI の ODBC セットアップ・ウィザードで DSN を構成する際にエンド・ユーザーが構成する必要のある名前です。複数の語で構成されるファイル名は、一重引用符で囲む必要があります。エントリの追加後は、構成ファイルを保存します。NQSConfig.INI ファイルの詳細は、『Oracle Business Intelligence Infrastructure インストレーションおよび構成ガイド』を参照してください。

3 ファイルを保存し、Oracle BI Server を再起動します。

詳細は、「Oracle BI Server の起動」（216 ページ）を参照してください。

データソースの作成

新規リポジトリにエンド・ユーザーのクライアント・アプリケーションを接続するために、各ユーザーは ODBC ドライバを使用してデータソースを定義する必要があります。

注意： Oracle BI Presentation Services と Oracle BI Server は、その関連性において、他のクライアント・アプリケーションと変わりがありません。

標準データソースとクラスタに参加するデータソースを作成できます。新規データソースの作成手順については、第 12 章「Oracle BI Server の接続性およびサード・パーティ製ツール」を参照してください。

Oracle BI Server の起動

Oracle BI Server を起動すると、NQSConfig.INI ファイルで指定されたリポジトリがロードされ、クエリーで使用できるようになります。サーバーの起動方法の詳細は、「Oracle BI Server の起動」（216 ページ）を参照してください。

リポジトリのテストと修正

リポジトリが作成されて接続できるようになったら、そのリポジトリに対してサンプル・クエリーを実行して、正常に作成されたかどうかをテストします。検出された問題の修正、再テストという処理を、満足な結果が得られるまで繰り返してください。

パフォーマンス・チューニングのヒント

物理データ・モデルは、トランザクション・データベース・システムよりも Oracle BI メタデータ・モデル（たとえば、スター・スキーマ）に類似しています。物理モデルが、基礎となるトランザクション・モデルのように設定されている場合、パフォーマンスと構成の問題が発生する可能性があります。その他のヒントについては、「リポジトリ設計のガイドライン」（52 ページ）を参照してください。

注意： パフォーマンス・チューニング・アクティビティ（ソースの追加など）は、メタデータが正常なレコード・セットを生成することを確認してから実行してください。

- メタデータの正確性は、パフォーマンスの向上よりも重要です。
- 通常、データベースにできるだけ多くの処理を配分します。これには、フィルタ処理や文字列の操作などのタスクと、加算メジャーが含まれます。
- ETL にできるだけ多くのクエリー・ロジックを移動して、システムのレスポンス時間を短縮します。加算メトリックと加算属性を事前計算することによりクエリーの複雑性が低減され、その結果、レスポンス時間が短縮されます。
- 実テーブルと拡張テーブルを使用して、アップグレード・パスをより明瞭にします。ランタイム・パフォーマンスを向上させるには、2 つのテーブルをマージして 3 つ目のテーブルを作成し、そのテーブルを Oracle BI にマップします。この方法には手間がかかり、ETL バッチ・ウィンドウのサイズも大きくなりますが、システムでアップグレード・エラーからの保護と最適なパフォーマンスの両立が実現されます。
- ETL プロセスを使用してデータを _DX（ディメンション拡張）テーブルに非正規化し、他のテーブルへの実行時の結合操作を減らします。

ユーザー・コミュニティへの公開

テストの完了後は、データ ソースをクエリーに使用できることをユーザー・コミュニティに通知します。Presentation Services ユーザーは、ブラウザに入力する URL を知っているだけで済みます。クライアントまたはサーバーのユーザー（たとえば、クエリー・ツールやレポート・ライター・クライアント・アプリケーションを使用して Oracle BI Server にアクセスするユーザー）は、サブジェクト領域名、サーバーの実行マシン、およびユーザー ID とパスワードを知っておく必要があり、各自の PC に ODBC 設定がインストールされている必要もあります。また、複数のリポジトリを使用する際に作成済のデータ ソース名（DSN）がデフォルト・リポジトリをポイントしない場合は、それらのリポジトリの論理名が必要になることがあります。

別のリポジトリからのインポート

プロジェクトは、インポート対象のオブジェクトを含むリポジトリで作成してから、リポジトリ・マージを使用して現行リポジトリにインポートすることをお薦めします。詳細は、「Oracle BI リポジトリのマージ」（167 ページ）を参照してください。

Repository Import Wizard を使用して、プレゼンテーション・カタログ（Answers のサブジェクト領域）とそれに関連付けられている子ビジネス・モデル、および別のリポジトリからの物理レイヤー・オブジェクトをインポートします。また、ユーザー、グループ、変数、初期化ブロックおよびプロジェクトもインポートできます。

注意：デフォルトでは、「File」メニューの「Import from repository」オプションは無効です。このオプションを有効にすると、ユーザーがインポートを開始するたびに、非推奨機能の警告ダイアログ・ボックスが表示されます。ダイアログ・ボックスで「はい」をクリックすると、インポート・プロセスが続行されます。「いいえ」をクリックすると、インポート・プロセスが終了します。このオプションの選択方法については、「「Options」ダイアログ・ボックスにおける「General」タブの使用」（31 ページ）を参照してください。

このオプションは、ビジネス・モデル・オブジェクトと物理レイヤー・オブジェクトが存在しない場合など、インポートするオブジェクトがリポジトリ内の既存オブジェクトに関連していない場合に使用します。同じ名前とタイプのオブジェクトが存在するときは、インポート・プロセスによって既存オブジェクトが新規オブジェクトで上書きされます。オブジェクトを 1 つのリポジトリから別のリポジトリへインポートする際には、インポート元リポジトリの一貫性が保たれている必要があります。

別のリポジトリからインポートするには

1. Administration Tool でリポジトリをオフライン・モードで開き、「File」→「Import from Repository」を選択します。

   このオプションは、リポジトリをオフラインで開いている場合にのみ使用できます。

2. Repository Import Wizard で、ファイル名でリポジトリを選択します。または、リポジトリが Oracle BI で使用されている場合、そのサーバーの該当リポジトリをポイントする ODBC DSN でリポジトリを選択し、「次へ」をクリックします。

3. Oracle BI 管理者のユーザー ID とパスワードを入力します。

4 Repository Import Wizard の「Objects to Update」ダイアログ・ボックスで、160 ページの表 19 に従ってドロップダウン・リストからカテゴリを選択します。

使用できるボタン（オプション）は、選択するカテゴリによって異なります。選択したオブジェクト、選択したオブジェクトとその子オブジェクト、または選択したオブジェクトとその親オブジェクトのみを追加できます。

5 インポートするオブジェクトがすべて追加されるまで、手順 4 を繰り返します。

オブジェクトをチェックアウトする必要がある場合は、「Check Out Objects」画面によって、オブジェクトのチェックアウトが求められます。

6 「次へ」をクリックして続行します。

7 「Finish」画面で、「Finish」をクリックします。

表 19. インポートするリポジトリ・オブジェクトのカテゴリ

カテゴリ	説明
Catalogs	カタログを選択すると、「Add with Children」ボタンが有効になります。プレゼンテーション・カタログは常に、すべての子オブジェクトとともに追加されます。「Presentation」レイヤーから「Physical」レイヤーまで、すべての関連オブジェクトが更新および同期化されます。
Groups	グループを選択すると、「Add」、「Add with Children」および「Add with Parents」ボタンが有効になります（グループのメンバーシップについての情報は、Security Manager から参照可能）。
Initialization Blocks	初期化ブロックを選択すると、「Add with Children」ボタンが有効になります。
List Catalogs	リスト・カタログを選択すると、「Add with Children」ボタンが有効になります。
Projects	プロジェクトを選択すると、更新および同期化の対象のすべてのリポジトリ・オブジェクトが表示されます。
Target levels	ターゲット・レベルを選択すると、「Add with Children」ボタンが有効になります。
Users	ユーザーを選択すると、左ペインで選択するユーザーに応じて、異なったボタンが有効になります。
Variables	変数を選択すると、「Add」および「Add with Parents」ボタンが有効になります。定義済のシステム変数とセッション変数が左ペインに表示されます。

リポジトリ・メタデータのクエリーと管理

リポジトリのオブジェクトのクエリーは、Query Repository ツールを使用して実行できます。「All Types」オプションを使用してクエリーを実行すると、リポジトリの公開済オブジェクト・タイプを含むリストが表示されます。このリストには、集計ルール、論理ソース・フォルダ、権限パッケージなどのオブジェクト、また内部オブジェクトとみなされているその他のオブジェクトは含まれません。

リポジトリ・クエリーを使用すると、次の方法でリポジトリ・メタデータを管理するのに役立ちます。

- リポジトリの内部構造を調査して更新する方法。たとえば、名前、タイプ（Catalog、Complex Join、Key、LDAP Server など）、または名前とタイプの組合せに基づいて、リポジトリ内のオブジェクトに対するクエリーを実行できます。その後、「Results」リストに表示されるオブジェクトを編集または削除できます。また、新規オブジェクトの作成や親階層の表示もできます。
- リポジトリをクエリーして、論理ソースにマップされたすべてのテーブル、特定の物理列へのすべての参照、論理ソースのコンテンツ・フィルタ、初期化ブロック、セキュリティ権限やユーザー権限などの項目を示すレポートを表示する方法。

  たとえば、リポジトリに影響を与える可能性がある物理的な変更をデータベースに加える前に、レポートを実行できます。レポートは、カンマ区切りの値（CSV）またはタブ区切り形式でファイルに保存できます。
- クエリーは、保存して後で再実行できます。また、クエリーの結果を外部ファイルに保存することもできます。外部ファイルに保存する場合、ANSI、Unicode および UTF-8 のエンコーディング・オプションを使用できます。

リポジトリをクエリーするには

1. Administration Tool を開いてから、リポジトリを開きます。
2. リポジトリで任意のオブジェクトを右クリックし、「Display Related」→「[検索対象のオブジェクト・タイプ]」を選択します。
3. 「Query Repository」ダイアログ・ボックスでは、オブジェクトのタイプが入力済です。
4. 「Query Repository」ダイアログ・ボックスで、162 ページの表 20 に従ってクエリー情報を完了します。

   162 ページの表 20 には、「Query Repository」ダイアログ・ボックスのほとんどのフィールドとボタンについての説明があります。
5. 「Query」をクリックします。

   クエリー結果は表示したり、外部ファイルに保存することができます。また、クエリーを変更して既存クエリーを上書きしたり、新規クエリーとして保存することもできます。

クエリー結果を外部ファイルに保存するには

1. クエリーの実行後に、「Query Repository」ダイアログ・ボックスで「Save」をクリックします。
2. 「Save As」ダイアログ・ボックスに名前を入力し、ファイル・タイプとエンコーディング値を選択します。
3. 結果に情報の新しい列を追加するには、「Add Columns」をクリックします。
4. 「Select information to add to the report」ダイアログ・ボックスで、リストから追加する列を選択します。
5. 選択済の列を選択してから「Up」および「Down」をクリックすると、列を並べ替えることができます。
6. 「OK」をクリックします。
7. 「Save as」ダイアログ・ボックスで、保存タイプを選択します。
8. 「Save」をクリックします。

クエリーを保存して後で再実行するには

1. クエリーの実行後に、「Query Repository」ダイアログ・ボックスで「Save Query As」をクリックします。
2. 「Save Query As」ダイアログ・ボックスにクエリーの名前を入力し、「Save」をクリックします。
3. 「Query Repository」ダイアログ・ボックスで、「閉じる」をクリックします。

保存したクエリーを削除するには

1. Administration Tool を開いてから、リポジトリを開きます。
2. 「Tools」メニューの「Query Repository」を選択します。
3. 「Query Repository」ダイアログ・ボックスで、「Saved Queries」をクリックします。
4. 「Saved Queries」ダイアログ・ボックスで右にスクロールするか、このダイアログ・ボックスを最大化します。
5. 削除するクエリーの削除ボタンをクリックしてから、「閉じる」をクリックします。

保存したクエリーを実行するには

1. Administration Tool を開いてから、リポジトリを開きます。
2. 「Tools」メニューの「Query Repository」を選択します。
3. 「Query Repository」ダイアログ・ボックスで、「Saved Queries」をクリックします。
4. 「Saved Queries」ダイアログ・ボックスで、実行するクエリーを含む行を選択し、「Select」をクリックします。
5. 「Query Repository」ダイアログ・ボックスで、「Query」をクリックします。

   クエリー結果は表示したり、外部ファイルに保存することができます。また、クエリーを変更して既存クエリーを上書きしたり、新規クエリーとして保存することもできます。

表 20. 「Query Repository」のフィールドと一部のボタン

フィールドまたはボタン	説明
Delete	クエリーの実行後に、クエリー結果のリスト内のオブジェクトを削除するために使用します。
Edit	クエリーの実行後に、クエリー結果のリスト内のオブジェクトを編集するために使用します。結果リストのリポジトリ・オブジェクトの中には、編集できないものもあります。たとえば、権限オブジェクトやユーザー・データベースのサインオン・オブジェクトがあります。オブジェクトを結果リストから編集できない場合、「Edit」ボタンは使用できません。
Filter	クエリーのフィルタを作成または編集するために使用します。フィルタの作成後は、ボタンの左側のテキスト・ボックスにフィルタ条件が表示されます。詳細は、「クエリー結果のフィルタの構成」（164 ページ）を参照してください。

表 20. 「Query Repository」のフィールドと一部のボタン

フィールドまたはボタン	説明
GoTo	クエリーの実行後に、リポジトリの Administration Tool ビューにあるオブジェクトに移動するために使用します。
Mark	クエリーの実行後に、選択したオブジェクトをマークするために使用します。オブジェクトのマーキングを解除するには、このボタンを再度クリックします。オブジェクトをマークすると、メタデータを作成する際にオブジェクトが視覚的に識別しやすくなります。
Name	オブジェクト名での検索が可能になります。アスタリスク（*）のワイルドカード文字を使用すると、任意の文字を指定できます。ワイルドカード文字では、検索文字列の最初と最後の文字を表せます。検索では、大文字と小文字は区別されません。
New	新規クエリーのリクエストに使用します。
Parent	クエリーの実行後に、オブジェクトの親階層を表示するために使用します。オブジェクトに親がない場合は、メッセージが表示されます。 「Parent Hierarchy」ダイアログ・ボックスでは、オブジェクトを編集または削除できます。ただし、オブジェクトを削除すると、選択したオブジェクトの子オブジェクトもすべて削除されます。
Query	クエリーを発行する準備ができたときに使用します。
Save Query As	クエリーを保存したり、以前に保存したクエリーを選択または削除することができるダイアログ・ボックスを開きます。クエリーの作成後にのみ使用可能です。
Saved Queries	クエリーの新しい名前を入力したり、既存の保存済クエリーを参照することができるダイアログ・ボックスを開きます。以前に保存したクエリーがある場合にのみ使用可能です。
Set Icon	クエリーの実行後に、オブジェクトの別のアイコンを選択するために使用します。アイコンを元に戻すには、このボタンを使用してから「Remove associated icon」を選択します。オブジェクトに特殊なアイコンを設定すると、それらのオブジェクトを共通の特性に基づいて視覚的に識別することが容易になります。たとえば、特殊なアイコンを選択して、特定のユーザー・グループのみが使用する列を識別できます。
Show Qualified Name	このチェック・ボックスは、クエリーで見つかったオブジェクトの完全修飾名を表示するために使用します。 たとえば、論理テーブルをクエリーした場合、「Name」リストのデフォルト値はテーブル名です。しかし、「Show Qualified Names」チェック・ボックスを選択すると、「Name」リストの値が「*businessmodelname.logicaltablename.columnname*」に変わります。
Type	ドロップダウン・リストからタイプを選択して、検索対象を特定のオブジェクト・タイプに絞り込みます。

新規オブジェクトを作成するには

1. Administration Tool のメニュー・バーから、「Tools」→「Query Repository」を選択します。
2. 「Query Repository」ダイアログ・ボックスの「Type」ドロップダウン・リストで、作成するオブジェクト・タイプを選択します。

3 「New」をクリックします。

表示されるダイアログ・ボックスは、選択したオブジェクト・タイプによって異なります。詳細は、このドキュメントのオブジェクト作成に関する項を参照してください。

クエリー結果のフィルタの構成

「Query Repository」ダイアログ・ボックスの「Results」リストの結果をフィルタするには、「Query Repository Filter」ダイアログ・ボックスを使用します。

「Query Repository Filter」ダイアログ・ボックスには、次の 5 つの列があります。「Item」列とその演算子または選択列、「Value」列とその演算子または選択列、および選択済のフィルタを削除する「Delete」列です。

「Query Repository Filter」ダイアログ・ボックスにアクセスするには

1 「Tools」メニューの「Query Repository」を選択します。

2 「Query Repository」ダイアログ・ボックスで、「Results」または「Type」リストの項目を選択し、「Filter」をクリックします。

フィルタを構成するには

1 「Tools」メニューの「Query Repository」を選択します。

2 「Query Repository」ダイアログ・ボックスで、「Results」または「Type」リストの項目を選択し、「Filter」をクリックします。

3 「Query Repository Filter」ダイアログ・ボックスで、「Item」フィールドをクリックします。

「Item」ドロップダウン・リストには、フィルタ処理するための項目が含まれています。

4 「Item」ドロップダウン・リストで、164 ページの手順 2 で選択した「Results」または「Type」オブジェクトに適用するフィルタを選択します。

「Item」ドロップダウン・リストでの選択に応じて、その他のオプションが使用可能になる場合があります。

ビジネス・モデルで参照されるすべてのデータベースを表示するフィルタを構成するには

1 「Tools」メニューの「Query Repository」を選択します。

2 「Query Repository」ダイアログ・ボックスの「Type」ドロップダウン・リストから「Database」を選択し、「Filter」をクリックします。

3 「Query Repository Filter」ダイアログ・ボックスで、「Item」フィールドをクリックします。

「Item」ドロップダウン・リストには、フィルタ処理するための項目が含まれています。

4 「Item」ドロップダウン・リストで、「Related to」を選択します。

「Item」フィールドの右側の列に等号記号（=）が表示されます。

5 「Value」フィールドの右側の省略記号ボタンをクリックし、ドロップダウン・リストから「Select」オブジェクトを選択します。

6 「Select」ダイアログ・ボックスで、フィルタ処理するビジネス・モデルを選択し、「Select」をクリックします。

選択した項目が「Value」フィールドに表示されます。

7 「OK」をクリックして「Query Repository」ダイアログ・ボックスに戻ります。

「Query Repository」ダイアログ・ボックスの「Filter」テキスト・ボックスにフィルタが表示されます。

論理列にマップされている、「Presentation」レイヤーのすべての列を表示するフィルタを構成するには

1 「Tools」メニューの「Query Repository」を選択します。

2 「Query Repository」ダイアログ・ボックスの「Type」ドロップダウン・リストから「Presentation Column」を選択し、「Filter」をクリックします。

3 「Query Repository Filter」ダイアログ・ボックスで、「Item」フィールドをクリックします。

「Item」ドロップダウン・リストには、フィルタ処理するための項目が含まれています。

4 「Item」ドロップダウン・リストで、「Column」を選択します。

「Item」フィールドの右側の列に等号記号（=）が表示されます。

5 「Value」フィールドの右側の省略記号ボタンをクリックし、ドロップダウン・リストから「Select」オブジェクトを選択します。

6 「Select」ダイアログ・ボックスで、フィルタ処理する列を選択し、「Select」をクリックします。

選択した項目が「Value」フィールドに表示されます。

7 「OK」をクリックして「Query Repository」ダイアログ・ボックスに戻ります。

「Query Repository」ダイアログ・ボックスの「Filter」テキスト・ボックスにフィルタが表示されます。

複数のフィルタを構成できますが、その場合、「Operator」フィールドが有効になります。「Operator」フィールドが有効になったら、AND および OR 条件の設定が可能になります。

ヒント： 複合フィルタを構成する場合は、絞込み条件を追加するたびに「OK」をクリックして、各絞込み条件でフィルタ構成が有効であることを確認できます。

リポジトリの比較

この項では、Administration Tool の「Compare Repositories」オプションの使用方法について説明します。このオプションを使用すると、2 つのリポジトリ間の内容を比較できます。それには、「Physical」、「Business Model and Mapping」および「Presentation」レイヤー内のすべてのオブジェクトが含まれます。

Oracle BI Applications リポジトリを使用する際に、その内容をカスタマイズした場合は、このオプションを使用して、カスタマイズ済のリポジトリを Oracle BI Applications で用意されているリポジトリの新規バージョンと比較できます。

カスタマイズ済の Oracle BI Applications リポジトリの内容をリポジトリの新規バージョンとマージする方法の詳細は、「Oracle BI リポジトリのマージ」（167 ページ）を参照してください。

2 つのリポジトリを比較するには

1 Administration Tool で、リポジトリをオフライン・モードで開きます。

この手順で開くリポジトリは、現行リポジトリと呼びます。リポジトリを開く方法の詳細は、「リポジトリのオンライン・モードとオフライン・モード」（25 ページ）を参照してください。

2 「File」メニューから、「Compare」を選択します。

3 「Select Original Repository」ダイアログ・ボックスで、開いたリポジトリと比較するリポジトリを選択します。

4 「Open Offline」ダイアログ・ボックスでパスワードを入力し、「OK」をクリックします。

5 「Compare repositories」ダイアログ・ボックスを使用して、2 つのリポジトリ間の差異を確認します。

次の一覧表に、「Change」列の値と説明を示します。

Change	説明
Created	現行リポジトリにオブジェクトが作成されましたが、元のリポジトリには存在しません。
Deleted	元のリポジトリにオブジェクトが存在しますが、現行リポジトリでは削除されています。
Modified	元のリポジトリにオブジェクトが存在しますが、現行リポジトリでは変更されています。

次の一覧表に、「Compare repositories」ダイアログ・ボックスの一部のボタンと、その機能の説明を示します。

ボタン	機能
Diff	現行リポジトリと元のリポジトリの間の差異を示します。
Edit 2	作成したオブジェクトを編集用に開きます。
Equalize	オブジェクトのアップグレード ID を均一にします。オブジェクト間のアップグレード ID が同一の場合、それらは同じオブジェクトであるとみなされます。リポジトリのマージでは使用できません。
Filter	「Comparison Filter」ダイアログ・ボックスを開いて、「Compare repositories」ダイアログ・ボックスに表示されるオブジェクトを、変更タイプおよびオブジェクト・タイプに応じてフィルタ処理できるようにします。表示する対象と非表示にする対象を指定できます。チェック・ボックス（「Group created and deleted objects」）を選択すると、作成済および削除済オブジェクトの子オブジェクトがフィルタで除外され、親オブジェクトのみを表示できます。デフォルトでは、すべての項目が表示されます。
Find	オブジェクトの「Name」および「Type」（「Initialization Block」など）で検索します。
Find Again	最新の「Find」値で再度検索します。
Mark	選択するオブジェクトをマークします。作成済および変更済オブジェクトの周辺にボックスが表示されます。マークを削除するには、「File」メニューの「Turn off Compare Mode」を選択します。リポジトリのマージでは使用できません。
Save	2 つのリポジトリ間の差異リストを保存します。

ボタン	機能
Select	リポジトリを選択して、現行リポジトリと比較できます。リポジトリのマージでは使用できません。
Stats	Change タイプごとの変更回数を示します。マルチユーザー開発マージでこのボタンを使用すると、これから実行するマージ決定の概要を表示できます。
View 1	削除済オブジェクトを読取り専用モードで開きます。

Turn Off Compare Mode

このオプションを使用すると、「Compare Repositories」および「Merge Repositories」オプションの使用時にオブジェクトに適用したマークを削除できます。「Turn off Compare Mode」オプションは、「File」→「Compare」操作で「Mark」をクリックした後にのみ使用できます。マークされたリポジトリ・オブジェクトがない場合、このオプションは使用できません。

「Turn Off Compare Mode」を有効にするには

- Administration Tool のツールバーから、「File」→「Turn Off Compare Mode」を選択します。

Oracle BI リポジトリのマージ

この項は、Oracle BI Applications リポジトリを使用する組織を対象にしています。ただし、カスタム・リポジトリをアップグレードするユーザーも「Merge Repository」オプションを使用できます。

マージ・プロセスには、Oracle BI リポジトリの 3 つのバージョンが関係します。次の説明の用語は、Administration Tool のユーザー・インタフェースで使用されています。

- **元のリポジトリ**：Oracle BI Applications の以前のバージョンで用意されているリポジトリです。この項では、Oracle の SiebelAnalytics.Original.rpd を例として使用します。
- **変更済リポジトリ**：元のリポジトリへのカスタマイズを含むリポジトリです。この項では、Oracle の SiebelAnalytics.Modified.rpd を例として使用します。
- **現行リポジトリ**：このバージョンにインストール済のリポジトリで、現在メイン・リポジトリとして開かれているものです。この項では、Oracle の SiebelAnalytics.Current.rpd を例として使用します。

マージ・プロセスでは、元のリポジトリと変更済リポジトリ、元のリポジトリと現行リポジトリを比較できます。「Merge Repository」オプションを使用すると、カスタマイズを現行リポジトリにマージするかどうかをオブジェクトごとに決定できます。

168 ページの図 14 は、「Merge repositories」ダイアログ・ボックスの各部分を示しています。169 ページの表 21 には、「Merge repositories」ダイアログ・ボックスの各列と各種ボタンの説明があります。

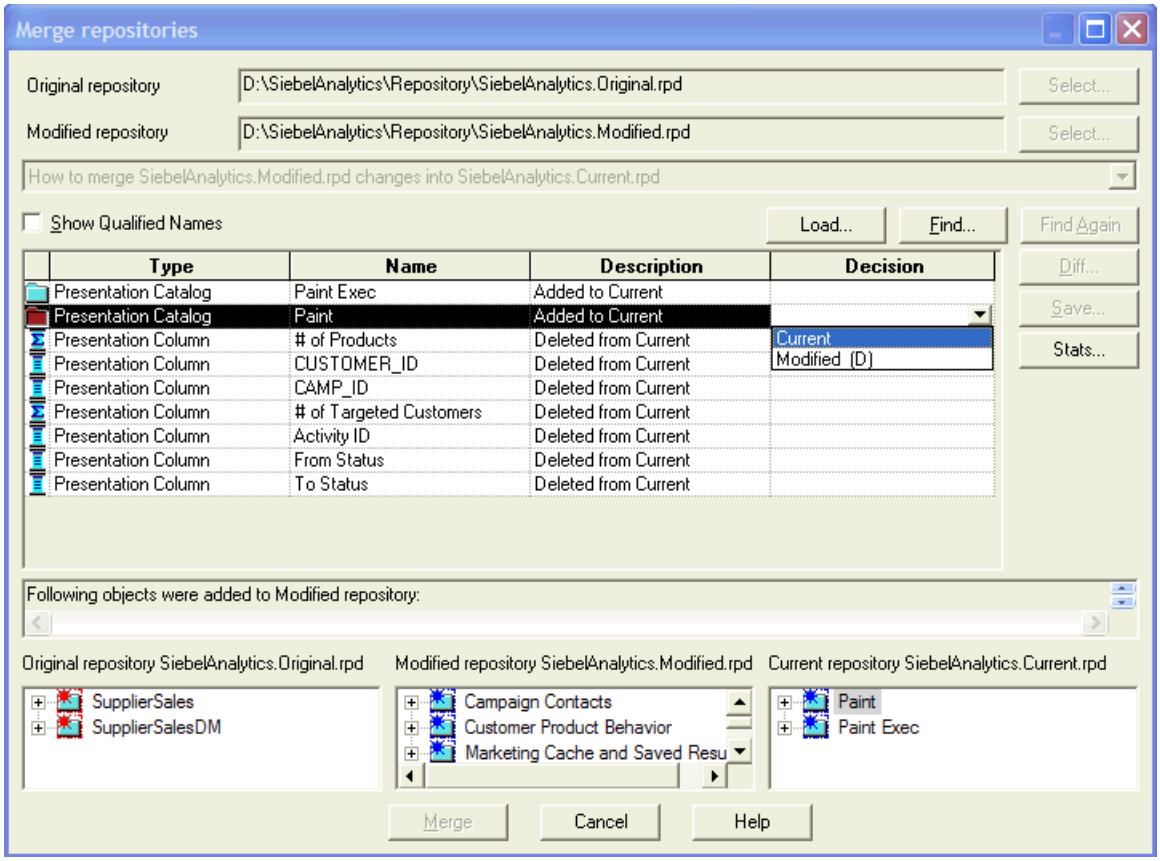


図 14. 「Merge repositories」ダイアログ・ボックス

- 「Original repository」および「Modified repository」フィールドは、ダイアログ・ボックスの最上部に表示されます。フィールドの右側にある「Select」ボタンを使用すると、リポジトリを選択できます。
- 読取り専用のテキスト・ボックスに、選択したマージのタイプが説明されます。

- ダイアログ・ボックスの中央にある決定テーブルは、このウィンドウで選択した内容に応じて動的に変更されます。たとえば、リポジトリの内容をマージするオプションを選択すると、決定テーブルに次の情報が表示されます。

表 21.「Merge repositories」の決定テーブル

列名	説明
Decision	選択したリポジトリ変更で実行する処理を選択できます。その処理の結果例については、「決定の選択肢の結果例」（170 ページ）を参照してください。 ■ **Current**: このタイプには接尾辞がありません。このタイプを選択すると、現行リポジトリのオブジェクトがそのまま残されます。 ■ **Modified**: このタイプには、A（追加）、D（削除）または AR（名前変更後）の接尾辞を付けることができます。 **注意：** AR は、変更済バージョンが許可されたものの、リポジトリ内の別の名前と競合するために名前が変更されたことを意味します。たとえば、ユーザーが現行リポジトリと変更済リポジトリの両方で、同じ名前を持つ同一のオブジェクトを追加してから両バージョンの許可を選択する場合、それらのオブジェクトは両方とも追加されますが、変更済リポジトリのオブジェクト名は変更されます。 ■ **Mix**: オブジェクトの追加も削除もされていませんが、少なくとも 1 つのプロパティが変更されました。たとえば、プロパティの選択肢を選択している場合があります。
Description	元のリポジトリと変更済リポジトリとの間、および元のリポジトリと現行リポジトリとの間の変更の説明です。
「Diff」（ボタン）	オブジェクトのどのプロパティが変更されたかを示します。「Mix」（追加も削除もされない）のラベルが付いたオブジェクトで使用できます。
「Find」（ボタン）	オブジェクトの「Name」および「Type」（「Initialization Block」など）で検索します。
「Find Again」（ボタン）	最新の「Find」値で再度検索します。
「Load」（ボタン）	Repository サブディレクトリから保存済の決定ファイルをロードして、リポジトリのマージ処理を続行できるようにします。
Name	オブジェクト名。
「Save」（ボタン）	仮変更を加えたファイルを Repository サブディレクトリに保存することにより、マージ作業を中断して後で続行できるようにします。変更（決定）内容の保存後は、「取消」をクリックして「Merge repositories」ダイアログ・ボックスを閉じる必要があります。
「Stats」（ボタン）	マルチユーザー開発でこのボタンを使用すると、これから実行するマージ決定の概要を表示できます。
Type	オブジェクト・タイプ。

- 決定テーブルでオブジェクトを選択すると、決定テーブルの下にある読取り専用テキスト・ボックスに、現行リポジトリのオブジェクトに対する変更内容が表示されます。

決定の選択肢の結果例

次の例では、現行リポジトリと変更済リポジトリが異なる場合の決定の選択結果を示します。

- オブジェクトの「Description」列に「Added to Current」が含まれる場合、「Decision」列の選択肢とその結果は次のとおりです。
  - 「Current」を選択すると、追加内容が現行リポジトリに保持されます。
  - 「Modified (D)」を選択すると、追加したオブジェクトが現行リポジトリから削除されます。
- オブジェクトの「Description」列に「Deleted from Modified」が含まれる場合、「Decision」列の選択肢とその結果は次のとおりです。
  - 「Current」を選択すると、オブジェクトは削除されず、リポジトリがそのまま保持されます。
  - 「Modified (D)」を選択すると、オブジェクトが現行リポジトリから削除されます。
- オブジェクトの「Description」列に「Deleted from Current」が含まれる場合、「Decision」列の選択肢とその結果は次のとおりです。
  - 「Current」を選択すると、オブジェクトは現行リポジトリに追加されず、リポジトリがそのまま保持されます。
  - 「Modified (A)」を選択すると、オブジェクトが現行リポジトリに追加されます。
- オブジェクトの「Description」列に「Added to Modified」が含まれる場合、「Decision」列の選択肢とその結果は次のとおりです。
  - 「Current」を選択すると、オブジェクトは現行リポジトリに追加されず、リポジトリがそのまま保持されます。
  - Modified (A): 「Modified (A)」を選択すると、オブジェクトが現行リポジトリに追加されます。

注意：選択した決定が「Mix」の場合、選択内容はオブジェクトの各プロパティ・レベルに適用されます。「Properties」ダイアログ・ボックスを開くには、省略記号ボタンをクリックします。

この項の手順では、Administration Tool の「Merge Repository」オプションを使用して、以前のリリースからのリポジトリのカスタマイズを新しいバージョンの Oracle BI リポジトリにマージする方法を示します。

Oracle BI リポジトリの複数のバージョンをマージするには

1. Administration Tool で、新しくインストールした Oracle BI リポジトリ（たとえば、SiebelAnalytics.Current.rpd）をオフライン・モードで開きます。
2. Administration Tool のメニュー・バーから、「File」→「Merge」を選択します。
3. 「Select Original Repository」ダイアログ・ボックスで、以前のバージョンのソフトウェアで用意されているリポジトリ（たとえば、SiebelAnalytics.Original.rpd）を選択します。
4. パスワードを入力し、「OK」をクリックします。
5. 「Modified repository」フィールドの「Select」をクリックします。

6 「Select Modified Repository」ダイアログ・ボックスで、以前のバージョンのリポジトリへのカスタマイズを含むリポジトリ（たとえば、SiebelAnalytics.Modified.rpd）を選択します。

7 「Open」をクリックしてパスワードを入力し、「OK」をクリックします。

8 「Decision」ドロップダウン・リストで、リポジトリ変更に対応する操作を選択するか、デフォルト操作を受け入れます。

9 「Decision」フィールドが空の後続の行を検出するには、「Decision」ヘッダー・セルをクリックします。

すべての行の「Decision」フィールドに値が入力されると、「Merge」ボタンが有効になります。

10 「Merge」をクリックします。

マージが完了したことを知らせるメッセージが表示されます。

11 「File」メニューの「Save As」を選択して、現行リポジトリを新しい名前で保存します。

IBM DB2 Cube Views への Oracle BI メタデータのエクスポート

Oracle BI 専用のメタデータは、XML ファイルに変換して、DB2 データベースにインポートできます。詳細は、「Oracle BI での IBM DB2 Cube Views の使用」（287 ページ）を参照してください。

注意： IBM DB2 Cube Views は、IBM 社の登録商標です。

プロジェクトへのメタデータ・サブセットの抽出について

プロジェクトは、メタデータの個々に定義されたサブセットで構成されています。プロジェクトは、「Presentation」レイヤーのカタログ（サブジェクト領域）とそれに関連付けられているビジネス・モデルの論理ファクト、ディメンション、グループ、ユーザー、変数および初期化ブロックで構成可能です。Oracle BI 管理者は、開発者および開発者グループが各自の職責でプロジェクトの作業を実行できるように、プロジェクトを作成します。次に、プロジェクトの作成が推奨される理由を示します。

- **ライセンス**：ソフトウェアの新規バージョンのリリース前に、プロジェクト内にライセンス・アプリケーションに関連するメタデータのみが存在し、すべてのメタデータの一貫性と完全性が保証されている必要があります。これは、このアプリケーション関連のファクト・テーブルのみを追加することで実現されます。
- **マルチユーザー開発**：開発プロセスにおいては、企業内の異なるチーム間で作業（メタデータ）を分割する必要があります。これは、メタデータをプロジェクトに抽出して、各プロジェクト・グループがメタデータの別の部分にアクセスできるようにすることで実現されます。

プロジェクトの抽出は、ファクト・テーブル集中型のプロセスです。これにより、プロジェクト抽出における一貫性が確保され、ライセンスの管理が容易になります。次に、ファクト・テーブルの種類を示します。

- **単純（ベース）ファクト・テーブル**：非導出列からなるファクト・テーブル、または式が定数で構成されている導出列からなるファクト・テーブル。
- **複合ファクト・テーブル**：別のファクト・テーブルのコンポーネントを含む導出列からなるファクト・テーブル。

- **コンポーネント・ファクト・テーブル**：複合ファクト・テーブルの一部である単純（ベース）ファクト・テーブル。複合ファクト・テーブルで、非導出列が使用されます。

「Project」ダイアログ・ボックスについて

ターゲット・レベル、リスト・カタログおよびプレゼンテーション・カタログは、プロジェクトで追加または削除できません。

左ペインではプレゼンテーション・カタログを追加できるように見えますが、実際には基礎となるファクト・テーブルを追加しているにすぎません。プレゼンテーション・カタログは、プロジェクトで必要とされる要素の追加を容易にするためにのみ、選択肢として表示されています。また、抽出の一貫性を保つために必要な、他のすべてのオブジェクトが追加されます。

ユーザー・インタフェースは、コード内で何が起こっているかを示します。「Project」ダイアログ・ボックスの左ペインには、プロジェクトの作成に使用できるオブジェクトが含まれます。右ペインのオブジェクトは、（直接的または間接的に）選択したすべてのオブジェクトで、それぞれの追加内容の一貫性を保証する完全なデータ・セットが反映されています。たとえば、プロジェクトに追加するプレゼンテーション・カタログを選択した場合、抽出の一貫性を保つために必要であれば、他のプレゼンテーション・カタログの基礎となるファクト・テーブルも自動的に追加されます。

次に、「Project」ダイアログ・ボックスの左ペインの内容について説明します。

- 単純（ベース）ファクト・テーブルとコンポーネント・ファクト・テーブルのみが表示されます。
- 複合ファクト・テーブルは表示されません。
- 「Catalog」または「Business Model」ごとにオブジェクトをグループ化できます。
- ビジネス・モデルごとにグループ化した場合、左ペインには、そのビジネス・モデルに属するファクトのみが表示されます。

次に、「Project」ダイアログ・ボックスの右ペインの内容について説明します。

- 単純（ベース）ファクト・テーブルとコンポーネント・ファクト・テーブルのみが表示されます。
- 複合ファクト・テーブルの全コンポーネントを追加せずに一部のコンポーネントのみを追加した場合、複合ファクト・テーブルは右ペインに表示されません。
- 「OK」をクリックすると、抽出されるカタログが表示されます。

リポジトリ・アップグレード時の古いプロジェクトの変換について

リリース 10.1.3.2 の Oracle BI にアップグレードすると、プロジェクト定義もアップグレードされます。アップグレード時には、プロジェクト定義、プレゼンテーション・カタログ、ターゲット・レベル、リスト・カタログおよび既存ファクト・テーブルが、次のように単純（ベース）ファクト・テーブルに自動的に変換されます。

- ターゲット・レベルに関連するプレゼンテーション列が制限付与キーを介して取得されます。
- リスト・カタログに関連するプレゼンテーション列が制限付与キーを介して取得されます。
- プレゼンテーション・カタログに関連するプレゼンテーション列が取得されます。
- すべてのプレゼンテーション列からベース論理列がすべて取得されます。
- プロジェクト内のファクト・テーブルからベース論理列がすべて取得されます。

- すべてのベース論理列からベース・ファクト・テーブルが取得されます。

注意：アップグレード後のプロジェクトには、単純（ベース）ファクト・テーブルのみが含まれます。セキュリティ・オブジェクトはすべて変更されずに残ります。

アップグレード後の古いリポジトリの使用について

古いバージョンのリポジトリが必要になる可能性がありますが、それをサポートするためには、現在のプロジェクト定義を古いプロジェクト定義へと変換する必要があります。プロジェクト定義をダウングレードするには、ファクト・テーブルを、プレゼンテーション・カタログ、ターゲット・レベル、リスト・カタログおよび複合ファクト・テーブルを含むプロジェクト定義に変換します。

リリース 10.1.2.3 より古いリリース番号のリポジトリを開くと、次の処理が行われます。

- ファクト・テーブルに関連するプレゼンテーション・カタログがすべて取得されます。
- プレゼンテーション・カタログのプレゼンテーション列がすべて取得されます。
- 制限付与キーの列がプレゼンテーション列に含まれるターゲット・レベルがすべて追加されます。
- 制限付与キーの列がプレゼンテーション列に含まれるリスト・カタログがすべて追加されます。
- プロジェクトにプレゼンテーション・カタログが追加されます。

Oracle BI のマルチユーザー開発環境の設定と使用

Oracle BI では、Oracle BI アプリケーションのグループ開発時に、複数の開発者が同じリポジトリ内のリポジトリ・オブジェクトに対して作業することを可能にします。次に、マルチユーザー開発環境の使用例を示します。

- 複数の開発者がメタデータのサブセットで同時に作業し、他の開発者と競合することなく、これらのサブセットをマスター・リポジトリにマージするケース。たとえば、企業でのデータ・ウェアハウスの実装後に、管理者が Oracle BI をその他の機能領域に配置する場合があります。
- 開発者 1 人が、すべての開発を担当するケース。開発者は、単純化とパフォーマンスの向上を目的として、Oracle BI マルチユーザー開発環境の使用により、メタデータ・コードを大規模なリポジトリではなく、小容量のチャンクに保持する場合があります。

どちらの例でも、管理者は Administration Tool でリポジトリ・ファイルにプロジェクトを作成し、このリポジトリ・ファイルを共有ネットワーク・ディレクトリにコピーします。開発者はプロジェクトをチェックアウトし、変更を行い、その変更内容をマスター・リポジトリにマージできます。

開発者がプロジェクトをチェックアウトすると、Administration Tool がバックグラウンドでファイルを自動的にコピーおよび上書きします。したがって、設定作業を実行する開発者や、チェックアウトおよびチェックイン手順を実行する開発者は、表示されるメッセージによく注意して行うことが重要です。

注意：この項の作業を実行するには、Oracle BI 管理者はメタデータ作成プロセスについて理解しておく必要があります。

この項の内容は次のとおりです。

マルチユーザー開発環境の設定（管理者）

管理者は、マルチユーザー開発を準備する際に次の作業を実行します。

- マルチユーザー開発環境用のプロジェクトの作成（174 ページ）：リポジトリに、開発者が必要とするプロジェクトを作成します。
- 共有ネットワーク・ディレクトリの設定（175 ページ）：マルチユーザー開発専用の共有ネットワーク・ディレクトリを指定または作成します。
- 共有ネットワーク・ディレクトリへのマスター・リポジトリのコピー（176 ページ）：すべてのプロジェクトの作成後、プロジェクトを作成したリポジトリ・ファイルを共有ネットワーク・ディレクトリにコピーし、マルチユーザー開発のマスター・リポジトリとして使用します。

  注意：この項では、共有ネットワーク・ディレクトリにコピーされたリポジトリを、マスター・リポジトリと呼びます。

マルチユーザー開発環境用のプロジェクトの作成

プロジェクトは、メタデータの個々に定義されたサブセットで構成されています。プロジェクトは、「Presentation」レイヤーのカタログ（サブジェクト領域）とそれに関連付けられているビジネス・モデルの論理ファクト、ディメンション、グループ、ユーザー、変数および初期化ブロックで構成可能です。

Oracle BI 管理者は、リポジトリにプロジェクトを作成し、このリポジトリを共有ネットワーク・ディレクトリにコピーします。プロジェクト作成のベスト・プラクティスは、プロジェクトをビジネス・モデルの個々の論理スター・スキーマに基づいた管理可能なサイズにすることです。作成したばかりの Oracle BI プロジェクトの場合、ベスト・プラクティスは、必要な物理テーブルおよび結合定義をすべて含んだリポジトリを使用することです。このリポジトリでは、論理ファクト・テーブルを「Business Model and Mapping」レイヤーのプレースホルダとして、プレゼンテーション・カタログを「Presentation」レイヤーのプレースホルダとして作成します。ビジネス・モデルおよびプレゼンテーション・カタログのメタデータが追加されるにつれて、個々のプレゼンテーション・カタログおよび論理ファクトに基づいた新規プロジェクトを作成できます。

注意：マスター・リポジトリにプロジェクトを作成できるのは、一度に 1 人のみです。

Oracle BI 管理者はプロジェクトの作成時に、プレゼンテーション・カタログ、または選択したプレゼンテーション・カタログに関連する論理ファクト・テーブルのサブセットを選択します。Administration Tool では、関連するすべてのビジネス・モデル・オブジェクトおよび物理レイヤー・オブジェクトが自動的に追加されます。オブジェクトは、複数のプロジェクトの一部にすることができます。

プロジェクトは、作成後にメタデータの一部となり、同じマスター・リポジトリ上で開発作業を行う必要がある、複数の開発者の使用が可能になります。プロジェクトをこの方法で定義すると、開発者がプロジェクトをチェックアウトして、新規リポジトリ・ファイルとして保存した後でも、そのプロジェクトは常に一貫性の取れたリポジトリになります。プロジェクトの作成方法の詳細は、「プロジェクトへのメタデータ・サブセットの抽出について」（171 ページ）を参照してください。

マルチユーザー開発環境用のプロジェクトを作成するには

1. Administration Tool メニューから、「File」→「Open」→「Offline」を選択します。
2. 「Open」ダイアログ・ボックスで、マルチユーザー開発に使用するリポジトリを選択し、「OK」をクリックします。
3. Administration Tool メニューから、「Manage」→「Projects」を選択します。
4. 「Project Manager」ダイアログ・ボックスの右パネルで右クリックし、「New Project」を選択します。

   左ペインには、プロジェクトに配置可能なオブジェクトがあります。右ペインには、プロジェクトの一部として選択したオブジェクトがあります。
5. 「Project」ダイアログ・ボックスに、プロジェクトの名前を入力します。
6. 次のいずれかの手順を実行して、プロジェクトの作成を終了します。
   - 「Project」ダイアログ・ボックスでカタログを開き、プレゼンテーション・カタログに関連するビジネス・モデル内の 1 つ以上の論理ファクト・テーブルを選択してから、「Add」をクリックします。

     プロジェクトは、論理ファクト・テーブルを明示的に含み、選択した論理ファクト・テーブルに結合されているすべての論理ディメンション・テーブルを（右ペインに表示されていなくても）暗黙的に含んでいるものとして定義されます。
   - 「Project」ダイアログ・ボックスでプレゼンテーション・カタログを選択し、「Add」をクリックします。

     Administration Tool によって、すべての論理ファクト・テーブルが自動的に追加されます。
7. プロジェクトからファクト・テーブルを削除するには、右ペインでファクト・テーブルを選択してから「Remove」をクリックします。
8. プロジェクトで必要とされるカタログ、グループ、ユーザー、変数または初期化ブロックを追加します。

   注意：プロジェクトでの作業が必要なすべての開発者を追加してください。そうしないと、オブジェクトのチェックアウトが許可されません。
9. 「OK」をクリックします。

共有ネットワーク・ディレクトリの設定

Oracle BI 管理者は、すべてのプロジェクトを定義して共有ネットワーク・ディレクトリを設定した後、全開発者がアクセス可能な共有ネットワーク・ディレクトリを指定または作成してから、その場所に新規マスター・リポジトリをアップロードする必要があります。共有ネットワーク・ディレクトリは、マルチユーザー開発にのみ使用します。このディレクトリには通常、複数の開発者が保持する必要がある、リポジトリのコピーが格納されます。

開発者は、自分のマシンに Administration Tool を設定する際に、この共有ネットワーク・ディレクトリへのポインタを作成します。

警告：Oracle BI 管理者は、マルチユーザー開発専用の独立した共有ネットワーク・ディレクトリを設定する必要があります。このディレクトリを指定どおりに設定および使用しない場合、重要なリポジトリ・ファイルが誤って上書きされ、リポジトリ・データが失われることがあります。

共有ネットワーク・ディレクトリへのマスター・リポジトリのコピー

マスター・リポジトリ・ファイルのコピーを作成し、それをマルチユーザー開発専用のディレクトリに貼り付けます。このマスター・リポジトリのプロジェクトは、開発者による変更の際に抽出およびダウンロードされます。また、その変更内容はマスター・リポジトリにマージされます。

リポジトリを共有ネットワーク・ディレクトリにコピーした後は、マルチユーザー開発環境の準備ができたことをすべての開発者に通知します。

マルチユーザー開発環境での変更の追加（開発者）

開発者は、プロジェクトをチェックアウトする前に、マスター・リポジトリを含む共有ネットワーク・ディレクトリをポイントするように Administration Tool を設定する必要があります。これは、Oracle BI 管理者が作成したマルチユーザー開発ディレクトリである必要があります。詳細は、「マルチユーザー開発環境の設定（管理者）」（174 ページ）を参照してください。

チェックアウトおよびチェックインの際には、マスター・リポジトリのコピーが開発者のローカル・リポジトリ・ディレクトリ（デフォルトでは、¥Oracle BI¥Repository）に一時的にコピーされます。プロジェクトをチェックアウトしてローカル・リポジトリ・ファイルに変更を加えた後は、各開発者はマスター・リポジトリに変更内容をチェックイン（マージ）したり、変更内容を破棄することができます。

マルチユーザー開発環境で変更を加えるには、次の作業を実行します。

- マルチユーザー開発ディレクトリへのポインタの設定（176 ページ）
- リポジトリ・プロジェクトのチェックアウト（177 ページ）
- メタデータの変更およびテストについて（178 ページ）

マルチユーザー開発ディレクトリへのポインタの設定

各開発者は、プロジェクトをチェックアウトする前に、ネットワーク上のマルチユーザー開発ディレクトリをポイントするように Administration Tool アプリケーションを設定する必要があります。Administration Tool では、このパスが開発者のワークステーション上の非表示の Windows レジストリ設定に格納されており、開発者が共有ディレクトリのオブジェクトをチェックアウトおよびチェックインする際に使用されます。

注意： Administration Tool のマルチユーザー・オプションは、ポインタが設定されるまで使用できません。

ネットワーク・ディレクトリの初期状態では、マスター・リポジトリが格納されています。この場所のリポジトリは、他の開発者と共有されます。

開発者はポインタの設定時に、「Full Name」フィールドにも入力できます。フィールドへの入力は任意ですが、このフィールドに入力することによって、リポジトリをロックしたユーザーが誰であるかを他の開発者に知らせることをお薦めします。「Full Name」の値は、レジストリの HKEY_CURRENT_USER に格納され、各ログインで一意となります。

マルチユーザー・デフォルト・ディレクトリにポインタを設定するには

1. Administration Tool メニューから、「Tools」→「Options」を選択します。
2. 「Options」ダイアログ・ボックスで、「Multiuser」タブをクリックします。

3 「Multiuser」タブの「Multiuser development directory」フィールドの横にある「Browse」をクリックします。

4 「Browse For Folder」ダイアログ・ボックスで、マルチユーザー開発ネットワーク・ディレクトリを検索および選択し、「OK」をクリックします。

5 「Options」ダイアログ・ボックスで、「Multiuser development directory 」フィールドに正しいディレクトリが表示されていることを確認します。

6 「Full Name」フィールドにユーザーのフルネームを入力し、「OK」をクリックします。

リポジトリ・プロジェクトのチェックアウト

マルチユーザー開発のデフォルト・ディレクトリへのポインタの設定後、開発者は、プロジェクトのチェックアウト、メタデータの変更およびメタデータのテストを実行できます。「File」→「Multiuser」サブメニューの「Checkout」オプションは、「Options」ダイアログ・ボックスの「More」タブでマルチユーザー開発ディレクトリが定義されている場合にのみ使用可能です。詳細は、「Oracle BI のマルチユーザー開発環境の設定と使用」（173 ページ）を参照してください。

開発者がローカル・リポジトリをチェックアウトして、ネットワーク上に公開するか、ローカルの変更内容を破棄する前にアプリケーションを終了しようとすると、操作の選択を求めるメッセージが表示されます。詳細は、「ネットワークへの公開前にリポジトリを閉じる操作について」（180 ページ）を参照してください。

プロジェクトをチェックアウトするには

1 Administration Tool メニューから、「File」→「Multiuser」→「Checkout」を選択します。

2 「Multiuser Development Checkout」ダイアログ・ボックスでリポジトリを選択し、「Open」をクリックします。

このダイアログ・ボックスは、マルチユーザー開発ディレクトリにリポジトリが 1 つしかない場合には表示されません。

3 「Extract from」ダイアログ・ボックスでユーザー名とパスワードの入力を求められたら、ユーザー名とパスワードを入力し、「OK」をクリックします。

リポジトリにプロジェクトが存在しない場合、メッセージが表示され、リポジトリは開きません。

4 「Browse」ダイアログ・ボックスで、抽出するプロジェクトの一部に含めるプロジェクトを選択し、「OK」をクリックします。

マスター・リポジトリに存在するプロジェクトが 1 つのみの場合、そのプロジェクトが自動的に選択され、「Browse」ダイアログ・ボックスは表示されません。

5 「New Repository」ダイアログ・ボックスで新規リポジトリの名前を入力し、「Save」をクリックします。

作業中のプロジェクト抽出リポジトリが、ローカル・マシンに保存されます。これは、指定した名前でオフライン・モードで開かれます。また、ログ・ファイルも作成されます。抽出されたリポジトリでは、一貫性が取れていない場合があります。

警告：プロジェクト抽出リポジトリの 2 つ目のコピーが、同じ場所に保存されます。このバージョンの名前は、リポジトリ抽出に割り当てた名前の先頭に「Original」が追加されたものになります。この「Original」プロジェクト抽出リポジトリは変更しないでください。このリポジトリは、変更内容を元のプロジェクトと比較する際に使用します。

メタデータの変更およびテストについて

標準リポジトリ・ファイルに対して追加可能な変更タイプのほとんどは、ローカル・リポジトリ・ファイルでもサポートされています。開発者は、新しい論理列や論理テーブルの追加、テーブル定義や論理テーブル・ソースの変更など、様々な操作を行えます。また、開発者は同じプロジェクトに対する同時作業をローカルで実行できます。ただし、Oracle BI では、これらの変更によるマスター・リポジトリへの影響を、個々の開発者が把握していることを前提にします。たとえば、開発者がローカル・リポジトリのオブジェクトを削除すると、この変更がマスター・リポジトリにも伝播されますが、警告プロンプトは表示されません。

次の変更は、ローカル・リポジトリでは行わないでください。

- 階層の定義。2 人の開発者が同時に変更すると、その変更内容はチェックイン時に正常にマージされません。
- プロジェクトの定義。これらは、Oracle BI 管理者のみがマスター・リポジトリで変更する必要があります。
- 物理接続の設定。これらは意図的に伝播されないため、開発者はローカル環境でテストできません。

ローカル・リポジトリに変更を加えた後は、ローカルの NQSConfig.INI ファイルを編集してリポジトリ名をデフォルト・リポジトリとして入力し、編集済のメタデータをテストできます。

注意：メタデータで指定されている DSN は、開発者のワークステーションに存在している必要があります。

メタデータのテスト方法の詳細は、「リポジトリのテストと修正」（158 ページ）を参照してください。

ローカル開発者は、変更の追加、テスト、およびローカルでのリポジトリの保存を実行した後、「File」→「Multiuser」サブメニューで次の作業を実行できます。

- **Compare with Original**: 作業中の抽出済ローカル・リポジトリと元の抽出済リポジトリとを比較します。このオプションを選択すると、「Compare repositories」ダイアログ・ボックスが開き、プロジェクトのチェックアウト後に作業中の抽出済リポジトリに加えた変更がすべて表示されます。詳細は、「リポジトリの比較」（165 ページ）を参照してください。
- **Discard local changes**: 変更内容は、チェックアウトからチェックインまでの任意の時点で破棄できます。このオプションを選択すると、作業内容を保存する機会がないまま作業中のリポジトリが閉じられます。

  警告：このオプションをいったん選択すると、操作を取り消すことはできません。たとえば、確認ダイアログ・ボックスは表示されません。
- **Merge local changes**: ネットワーク・マルチユーザー・ディレクトリのマスター・リポジトリをロックして、変更内容をチェックインできるようにします。詳細は、「マルチユーザー開発のリポジトリ・プロジェクトのチェックイン」（178 ページ）を参照してください。
- **Publish to the network**: 変更内容を正常にマージした後、マスター・リポジトリがローカルで開き、「Publish to Network」サブメニュー項目が使用可能になります。このオプションを選択すると、ロックが解除されます。また、リポジトリが公開されてから閉じられます。詳細は、「マルチユーザー開発のリポジトリ・プロジェクトのチェックイン」（178 ページ）を参照してください。

マルチユーザー開発のリポジトリ・プロジェクトのチェックイン

ローカル・マシンでのメタデータの変更およびテスト後、開発者はローカルの変更内容をローカル・マスターにマージしてから、プロジェクトを共有ネットワーク・ディレクトリのマスター・リポジトリにチェックインする必要があります。ローカル・リポジトリからマスター・リポジトリへとメタデータをマージできるのは、一度に 1 人の開発者のみです。したがって、マスター・リポジトリはマージ・プロセスの初めにロックされます。

プロジェクトのチェックインについて

チェックイン・プロセスが開始すると、次の処理が行われます。

- Administration Tool により、マスター・リポジトリが現在ロックされているかどうかが確認されます。ロックされていない場合、マスター・リポジトリをロックして、現在のマージが完了するまで他の開発者がマージを実行できないようにし、ロックしたことをログ・ファイルに記録します。
- その他の開発者は、現在のチェックイン・プロセスが正常に完了するまで、「File」→「Multiuser」メニューの「Merge Local Changes」オプションを使用できなくなります。
- Administration Tool により、現在のバージョンのマスター・リポジトリが共有ネットワーク・ディレクトリから開発者マシンの ¥¥Oracle BI¥Repository ディレクトリに自動的にコピーされ、ローカルおよび共有ネットワーク・ディレクトリのログ・ファイルが更新されます。これは、開発者がプロジェクトをチェックアウトした後に、共有ネットワーク・ディレクトリのマスター・リポジトリが変更された可能性があるため必要です。

マルチユーザー開発メタデータのマージについて

マージ・プロセスには、次のファイルが関係します。

- **元のローカル（サブセット）リポジトリ**：最初に抽出されたときのプロジェクト状態が含まれます。このリポジトリ名は、Original で始まります。たとえば、このコピーのファイル名を OriginalDevelopment2.rpd にすることができます。このバージョンは、変更済（または作業中）バージョンのローカル・リポジトリと同じ場所に格納されます。
- **変更済ローカル（サブセット）リポジトリ**：開発者が変更した後の抽出済プロジェクトが含まれます。このバージョンは、元のバージョンのローカル・リポジトリと同じ場所に格納されます。
- **ネットワーク共有ディレクトリのマスター・リポジトリ**：このマージの前に別の開発者によって変更されている可能性があります。このリポジトリの例として、Master_SiebelAnalytics.rpd があります。

Administration Tool では、マージ中に、追加されたオブジェクトの有無が確認され、検出された場合は、警告メッセージが表示されます。その際に次の処理が行われます。

- 追加されたオブジェクトについての警告。プロジェクトをチェックアウトすると、そのプロジェクトが自由に変更できるようになります。また、変更したプロジェクトはチェックインして戻すことができます。削除および変更の操作では、プロジェクトの整合性は確保されます。しかし、オブジェクトを追加する場合、リポジトリ内にどのプロジェクトにも属さないオブジェクトが作成されてしまうことがあります。したがって、プロジェクト関連のすべてのオブジェクトは確認され、新しいオブジェクトが検出されると、警告メッセージが表示されます。
- 関連オブジェクトの集計。警告メッセージでは、親オブジェクトのみがレポートされます。Administration Tool では、メッセージをより容易に使用できるようにするため、すべてのオブジェクトが集計されています。たとえば、開発者が新規ビジネス・モデルを追加した場合、ユーザーにビジネス・モデルのみの警告メッセージが表示され、テーブル、列、ディメンションなどの警告メッセージは表示されません。

開発者が Administration Tool を閉じると、次の処理が行われます。

- 共有ネットワーク・ディレクトリのマスター・リポジトリが、開発者の変更内容を含むマスター・リポジトリで上書きされます。

- [*master repository*].lck ファイルが削除されます。別の開発者がマスター・リポジトリから変更済のプロジェクトをチェックアウトすると、最初の開発者が変更した内容が、その開発者に公開されます。

  警告：Oracle BI 管理者は、新しく作成されたメタデータをマスター・リポジトリのプロジェクト定義に追加して、そのメタデータが今後のバージョンの抽出プロジェクトで表示されるようにする必要があります。たとえば、開発者がプロジェクトをチェックアウトして、新規オブジェクトの追加後にチェックインした場合、その新規オブジェクトは、プロジェクト定義に明示的に追加するまで、抽出されたプロジェクト・バージョンには表示されません。手順については、「マルチユーザー開発環境用のプロジェクトの作成」（174 ページ）を参照してください。

マスター・リポジトリへの変更のトラッキング

マスター・リポジトリに対する開発アクティビティのサマリーは、[*master_repository*].log にあります。このログには、次のアクティビティが記録されています。

- チェックインまたはチェックアウトされたプロジェクト、およびそれらの操作の実行日時
- トランザクションを開始した NT ログイン名とコンピュータ名
- ロックとロック解除の日時

マルチユーザー・マージ・プロセスと標準リポジトリ・マージ・プロセスとの相違点

マルチユーザー開発のチェックイン・プロセスでは、標準リポジトリ・マージ・プロセスと同じテクノロジが実装されていますが、いくつかの重要な相違点があります。標準リポジトリ・マージの詳細は、「Oracle BI リポジトリのマージ」（167 ページ）を参照してください。

次に、マルチユーザー開発マージでの相違点を説明します。

- 挿入が自動的に適用されます。マスター・リポジトリのサブセットが元のリポジトリとして使用されているため、マスター・リポジトリのほとんどのオブジェクトが新規として表示されます。その結果、開発者による手動の承認を要する、不必要な多数のプロンプトが表示されることになります。そのため、マルチユーザー開発マージでは、挿入がプロンプト表示なしで適用されます。
- 自動挿入の結果として解消される競合（挿入の競合以外の競合）は、マルチユーザー開発マージでは、プロンプト表示なしで適用されます。
- サーバー上のデータベースおよび接続プール・プロパティは、開発者マシン上の同一プロパティよりも優先されます。マルチユーザー開発マージでは、この優先処理がプロンプト表示なしで適用されます。

ネットワークへの公開前にリポジトリを閉じる操作について

「File」→「Multiuser」サブメニューのいずれかのオプションを選択せずに、未公開のロックされたリポジトリを閉じようとすると、「Closing MUD repository」ダイアログ・ボックスが開き、次のオプションが表示されます。

- **Publish to Network**: マージ済リポジトリを新規マスターとしてネットワーク共有で公開し、マスターのロックを解除してからイベントをログに記録します。このオプションは、「Merge Local Changes」イベントの発生後に使用できます。このオプションは、「File」→「Multiuser」サブメニューでも使用できます。
- **Discard Local Changes**: マスター・リポジトリのロックを解除し、イベントをログに記録します。このオプションは、「Checkout」または「Merge Local Changes」の実行後に使用できます。このオプションは、「File」→「Multiuser」サブメニューでも使用できます。

- **Close repository and keep lock**: マスター・リポジトリをロックしたままの状態でリポジトリを閉じます。

マスター・リポジトリにプロジェクトをチェックインするには

1. Administration Tool メニューから、「File」→「Multiuser」→「Merge Local Changes」を選択します。
2. 「Lock Information」ダイアログ・ボックスの「Comment」フィールドに変更内容の説明を入力し、「OK」をクリックします。
3. 「Merge repositories」ダイアログ・ボックスで、元のローカル・リポジトリおよび変更済ローカル・リポジトリのファイル名が正しいことを確認します。

   「Merge repositories」ダイアログ・ボックス上では、リポジトリ間に差異がないように思える場合がありますが、これは変更内容のチェックインに、開発者による明示的な決定が不要であることを意味しています。
4. 実行されるマージ決定の概要を参照するには、「Stats」をクリックします。
5. 「Results」ダイアログ・ボックスで、「閉じる」をクリックします。
6. 「Merge」をクリックします。
7. グローバルな一貫性をチェックするかどうかを尋ねられたら、「はい」または「いいえ」をクリックします。

   変更内容がマージされ、マージ済のローカル・リポジトリ・ファイルが開きます。開発者のローカル・ディレクトリには、マージされた変更内容の詳細を含んだ CSV ファイルが作成されます。
8. 変更済ローカル・リポジトリの変更内容が、このマージ済ローカル・リポジトリに反映されていることを確認します。

   警告：マージ済リポジトリは共有リポジトリと同じ名前ですが、それでもローカル・コピーであることには変わりありません。
9. すべての変更内容を確認した後、「Save」をクリックします。

   これにより、マージ済リポジトリがローカルで保存されます。このリポジトリは、ファイル拡張子（000）付きで共有ネットワーク・ディレクトリにアップロードされます。たとえば、Master_Sales.000 となります。

   この時点では、開発者が加えた変更は、共有ネットワーク・ディレクトリのマスター・リポジトリに保存されていません。
10. これらの変更内容を共有ネットワーク・ディレクトリのマスター・リポジトリにコミットするには、Administration Tool を閉じます。
11. 「Closing MUD repository」ダイアログ・ボックスで、「Publish to Network」を選択します。

    共有ネットワーク・ディレクトリのマスター・リポジトリが、開発者の変更内容を含むリポジトリ・コピーで上書きされます。このダイアログ・ボックスの各オプションの詳細は、「ネットワークへの公開前にリポジトリを閉じる操作について」（180 ページ）を参照してください。

マルチユーザー開発履歴表示と削除

マルチユーザー開発リポジトリの開発履歴を表示できます。Administration Tool では、マルチユーザー開発履歴は、開いているリポジトリがなく、かつ管理者が共有ネットワーク・ディレクトリを設定した場合にのみ参照できます。これにより、ユーザーが開いた履歴ログが、現在開かれている関連性のないリポジトリと一致しないために混乱が生じるという事態を防げます。

マルチユーザー開発履歴を表示するには

1. Administration Tool を開きます。
2. リポジトリを開かずに、Administration Tool メニューから「File」→「Multiuser」→「History」を選択します。
3. 「Multiuser Development History」ダイアログ・ボックスで、リポジトリを選択します。

   マルチユーザー開発ディレクトリ内のすべてのマスター・リポジトリが一覧表示されます。ディレクトリ内のマスター・リポジトリが 1 つのみの場合、そのリポジトリがデフォルトで選択されるため、リポジトリの一覧は表示されません。
4. 「Open Offline」ダイアログ・ボックスで、リポジトリのパスワードを入力します。
5. 「Multi User History」ダイアログ・ボックスの行を右クリックし、次のいずれかの項目を選択します。

操作	説明
View Repository	Administration Tool で選択したリポジトリのマスター・バージョンを、読取り専用モードでロードします。
View Projects	Administration Tool で選択した変更済サブセット・リポジトリのバージョンを、読取り専用モードでロードします。
View Conflict Resolution	選択したバージョンの必要なリポジトリをすべてロードします。また、「Merge」ダイアログ・ボックスを読取り専用モードで開き、「Merge Local Changes」アクティビティのときのまま、選択したすべての決定内容を表示します。 **注意：** このメニュー項目を有効にするには、ダイアログで「Conflict Resolution」チェック・ボックスを選択している必要があります（選択していない場合は、ユーザーによる決定事項がないため何も表示されません）。
View Details	選択したバージョンの詳細を含むログを表示します。特定のバージョンを選択していない場合は、すべてのバージョンの詳細を表示します。
View Changes	選択したバージョンの変更済サブセット・リポジトリを元のサブセット・リポジトリと比較し、ユーザーがその選択バージョンに加えた変更をすべて表示します。
「Find」および「Find Again」	一覧を検索できます。

操作	説明
Select All	ダイアログに表示されるすべての項目を選択します。
Delete	管理者のみが使用できます。

マルチユーザー開発履歴の削除

履歴を削除できるのは、管理者のみです。管理者は、マルチユーザー開発ディレクトリ内の特殊な隠しオプション・ファイルで定義されます。このオプション・ファイルには、次のプロパティと特性があります。

- HIDDEN フラグがオンであること。
- マルチユーザー開発管理者にのみネットワーク・アクセス権限が割り当てられていること。
- 拡張子が OPT である点を除けば、マスター・リポジトリと同じ基準名が付けられていること。たとえば、¥¥network¥MUD¥sales.rpd の場合、管理者は ¥¥network¥MUD¥sales.opt という隠しファイルを作成できます。
- OPT ファイルは、次の形式の TXT ファイルであること。

  ```
  [Options]

  Admin=admin1;admin2
  ```

  管理者は、ネットワーク・ログイン名で定義されます。複数の管理者が存在する場合、管理者の名前はセミコロンで区切ります。次に例を示します。

  ```
  [Options]

  Admin=jsmith;mramirez;plafleur
  ```

管理者は、MUD 履歴全体または最も古い順から 1 ～ n までのバージョンを削除できます。この範囲の途中でバージョンを削除することはできません。たとえば、バージョン 2 および 1 が存在する場合、管理者はバージョン 3 を削除できません。管理者が MUD 履歴全体を削除すると、新しく割り当てられるバージョン番号は 1 から再開します。1 つ以上のバージョンが削除されていない場合、新しく割り当てられるバージョンはその次の番号から続きます。

Delivers で操作するためのリポジトリの設定

Oracle BI Presentation Services では、Delivers からのアラートをグループ全体に、または指定の電子メール・アドレス、電話番号などに配信する必要があります。Delivers では、アラート配信に iBot というツールを使用します。iBot は、スケジュールドリブンまたはイベントドリブンのソフトウェアベース・エージェントで、定義された条件に基づいてデータのアクセス、フィルタ処理および分析を実行します。iBot の詳細は、『Oracle Business Intelligence Answers, Delivers, and Interactive Dashboards ユーザーズ・ガイド』の Delivers の使用に関する章を参照してください。

これが正常に機能するには、Oracle BI リポジトリの「Presentation」レイヤーで、SA System という名前のプレゼンテーション・カタログ（Answers のサブジェクト領域）を設定する必要があります。

注意： BI Presentation Server パラメータの設定方法については、『Oracle Business Intelligence Presentation Services 管理ガイド』を参照してください。

この項の内容は次のとおりです。

SA System サブジェクト領域について

Oracle BI では、ユーザーが属するグループごとにデータの表示方法を定義できます。たとえば、Oracle BI アプリケーションに顧客がログオンすると、GROUP システム・セッション変数が移入され、その変数内に存在するグループに応じて、特定のサブジェクト領域と列が表示されます。GROUP システム・セッション変数により、各ユーザーが属するグループは特定されますが、各グループに存在するユーザーは識別されません。

SA System は、Delivers にグループ内のユーザーを公開することによってこのデータ表示の問題を解決するプレゼンテーション・カタログ（Answers のサブジェクト領域）です。また、電子メール・アドレスなどの連絡先情報をデータベースから取り出して、Delivers の配信デバイスとして使用できるようにします。これにより、Oracle BI 管理者は、Delivers の「My Account」画面の更新をユーザーに求めるかわりに、Oracle BI Server を設定して、ユーザーの配信デバイスおよびプロファイルを自動的に移入できます。

グループ・メンバーシップは多くの場合、Oracle BI リポジトリではなく、Oracle の Siebel トランザクション・データベースなどの外部データベースに保持されています。この情報は、GROUP セッション変数によって Oracle BI Server および Oracle BI Presentation Services に伝播できます。SA System サブジェクト領域を GROUP セッション変数とともに使用すると、Delivers にグループ・メンバーシップと外部電子メール・アドレスが提供されます。

SA System サブジェクト領域の設定

Oracle BI Presentation Services は、起動時に SA System サブジェクト領域を検索するように事前構成されており、存在する場合はこのサブジェクト領域を使用します。グループ・メンバーシップ、ユーザーの第 1 電子メール・アドレスおよび SMTP デバイス（携帯電話、ハンドヘルド機器など）の情報を保持する外部スキーマをインポートして、SA System サブジェクト領域にマップできます。

SA System テーブルに（プロバイダ名などの）列を追加するには、基礎となる物理テーブルおよび列を拡張します。

この項の内容は次のとおりです。

SA System サブジェクト領域の実装ガイドライン

サブジェクト領域の名前は、必ず SA System にする必要があります。SA System サブジェクト領域を使用する際には、次の設定ガイドラインに従ってください。

- **Users and groups**: データ内にすべてのユーザーとグループが存在している必要があります。Oracle BI では、SA System サブジェクト領域内に Oracle BI リポジトリの内部ユーザーと外部ユーザーが混在するグループ・メンバーシップはサポートしていません。
- **Delivery profile**: ユーザーによる配信プロファイルの設定はお薦めしません。Oracle BI 管理者がこのプロファイルを管理する方法がないためです。たとえば、Oracle BI 管理者は、電子メール・アドレスの大量更新を実行できなくなります。ユーザーが配信プロファイルを設定すると、その配信プロファイルは SA System サブジェクト領域の表示項目よりも優先されます。
- **Authorization and authentication**: このオプションは、システムでユーザーが実行できること（認可）に影響を与えますが、ユーザーが誰であるか（認証）には影響しません。データベース認証の関連情報については、「Oracle BI Delivers とデータベース認証について」（337 ページ）を参照してください。

  警告： instanceconfig.xml の既存の構成オプション（Alerts 要素）は、SA System サブジェクト領域にアクセスする前のログイン名の処理方法を制御します。たとえば、ログイン名をすべて大文字に変換できます。詳細は、『Oracle Business Intelligence Presentation Services 管理ガイド』を参照してください。
- **Security settings**: SA System サブジェクト領域では、管理者アカウントへの読取りアクセスのみが必要です。そのため、セキュリティ設定が損なわれることはありません。グループ・メンバーシップを機密データとして扱う場合は、このサブジェクト領域へのアクセスを Oracle BI 管理者にのみ許可できます。

スタンドアロン実装での SA System サブジェクト領域の設定

Oracle BI をスタンドアロンで使用する顧客は、SA System サブジェクト領域を Oracle BI リポジトリに作成および移入する必要があります。

Oracle BI Server をスタンドアロン環境に（Oracle BI Applications なしで）インストールする顧客は、ここに示す順番で次の作業を実行してください。

- データソース（たとえば、外部データベース）にテーブルおよび列を作成します。
- Oracle BI リポジトリにサブジェクト領域を作成および構築します。
  - 該当情報が格納されているスキーマをインポートします。手順については、「Oracle BI リポジトリの「Physical」レイヤーの作成と管理」（59 ページ）を参照してください。
  - テーブルおよび列を、「Physical」レイヤーから「Business Model and Mapping」レイヤーへマップします。手順については、「Oracle BI リポジトリの「Business Model and Mapping」レイヤーの作成と管理」（113 ページ）を参照してください。
  - テーブルおよび列を、「Business Model and Mapping」レイヤーから「Presentation」レイヤーへマップします。手順については、「Oracle BI リポジトリの「Presentation」レイヤーの作成とメンテナンス」（147 ページ）を参照してください。

    「Presentation」レイヤーのメタデータには、SA System フォルダ、ユーザー・テーブルおよび列が含まれている必要があります。

注意： Oracle BI Application の顧客の場合、SA System サブジェクト領域は Oracle BI リポジトリに事前構成されています。

8 Oracle BI Administration Tool のユーティリティと式ビルダー

この章では、Administration Tool に用意されている各種のユーティリティおよびウィザードについて説明します。また、式ビルダーの説明と、リポジトリ内で絞込み条件、集計およびその他の定義を作成する手順についても説明します。

この章の内容は次のとおりです。

ユーティリティとウィザード

Administration Tool には、様々な作業を支援する多数のウィザードおよびユーティリティが用意されています。

この項では、次のユーティリティおよびウィザードについて説明します。

Replace Column or Table Wizard

Replace Wizard では、Oracle BI 管理者が表示されたソースから選択することにより、論理テーブル・ソースにおける物理列またはテーブルの置換プロセスが自動化されます。このウィザードでは、列およびテーブルの置換を要求するプロンプトが Oracle BI 管理者に対して表示されます。

「Replace Column or Table Wizard」を起動するには

1. 「Tools」メニューから、「Utilities」→「Replace Column or Table in Logical Table Sources」を選択します。
2. 「Execute」をクリックします。

Oracle BI Event Tables

このユーティリティを使用すると、テーブルを Oracle BI イベント・ポーリング・テーブルとして識別できます。イベント・ポーリング・テーブルは、1 つ以上の物理テーブルが更新されたことを Oracle BI Server に通知する方法を提供します。イベント・テーブルに追加される行は、それぞれ 1 つの更新イベントを説明します。キャッシュ・システムでは、イベント・テーブルの行を読み取り（ポーリング）、その行から物理テーブルの情報を抽出して、それらの物理テーブルを参照しているキャッシュ・エントリを消去します。イベント・テーブルの詳細は、「イベント・ポーリング・テーブルによるキャッシュ・イベント処理」（257 ページ）を参照してください。

Oracle BI Event Tables・ユーティリティを起動するには

- 「Tools」メニューから、「Utilities」→「Event Tables」を選択します。
- 「Execute」をクリックします。

Externalize Strings

このユーティリティは主に、翻訳者が「Presentation」レイヤーのカタログ、テーブル、列およびその説明を他言語に翻訳するために使用します。これらのテキスト文字列は、ANSI、Unicode および UTF-8 のコーディング・オプションを使用して外部ファイルに保存できます。

警告：このユーティリティを使用する際には、翻訳者は Oracle BI の窓口と緊密に連携して、各状況で適切なプロセスが遵守されていることを確認する必要があります。

文字列を翻訳するには

1. 「Presentation」レイヤーで、プレゼンテーション・カタログを右クリックし、「Externalize Display Names」を選択します。

   注意：Administration Tool では、プレゼンテーション・カタログという用語は、サブジェクト領域のことを示します。

2. 「Presentation」レイヤーで、同じプレゼンテーション・カタログを右クリックし、「Externalize Descriptions」を選択します。

   右クリック・メニューでは、両方のオプションの横にチェック・マークが表示されます。

3. 「Tools」メニューから、「Utilities」→「Externalize Strings」を選択し、「Execute」をクリックします。

4. 「Externalize Strings」ダイアログ・ボックスの左ペインでプレゼンテーション・カタログをクリックします。

   注意：すべてのカタログを一度に選択するか、個々のカタログを選択して、それぞれに個別の文字列ファイルを作成できます。

   右ペインに、翻訳済の値と元の文字列（名前）が表示されます。これらは、Oracle BI Presentation Services で使用されるセッション変数に配置されます。

5. 「Save」をクリックします。

6. 「Save As」ダイアログ・ボックスで、ファイル・タイプとエンコーディング値を選択し、「Save」をクリックします。

7. 「Externalized Strings」ダイアログ・ボックスで、「閉じる」をクリックします。

8. （オプション）これらのオプションの横にあるチェック・マークの選択を解除するには、プレゼンテーション・カタログを右クリックして、各オプションをクリックします。

Rename Wizard

Rename Wizard を使用すると、「Presentation」および「Business Model and Mapping」レイヤーにあるテーブルと列の名前を変更できます。これは、物理名をわかりやすい名前に変換する際に便利です。

注意：プレゼンテーション・レイヤーの列の名前を変更すると、「Use Logical Column Name」プロパティが False にリセットされます。かわりに、ビジネス・モデル・レイヤーの論理列の名前を変更することをお薦めします。

Rename Wizard を起動するには

■ 「Tools」メニューから、「Utilities」→「Rename Wizard」を選択し、「Execute」をクリックします。

Update Physical Layer Wizard

このウィザードを使用すると、リポジトリの「Physical」レイヤーのデータベース・オブジェクトを、バックエンド・データベースの現在の定義に基づいて更新できます。

注意： このウィザードは、読取り専用モードで開かれているリポジトリでは使用できません。読取り専用モードのリポジトリは更新できないからです。

ウィザードで更新処理が行われると、Administration Tool を実行しているサーバーは、各バックエンド・データベースに接続します。「Physical」レイヤーのオブジェクトは、バックエンド・データベースのオブジェクトと比較されます。説明テキストには、「Physical」レイヤーのデータベース内のオブジェクト定義とバックエンド・データベース内のオブジェクト定義との差異（データ型の長さの不一致やバックエンド・データベースには見つからないオブジェクトなど）を示すアラートが表示されます。たとえば、リポジトリの「Physical」レイヤーのデータベースに存在するオブジェクトがバックエンド・データベースに存在しない場合、次のテキストが表示されます。

```
Object does not exist in the database
```

注意： このウィザードは、バックエンド・データベースに存在するがリポジトリに存在しない列またはテーブルを、リポジトリに追加しません。さらに、列のキー割当ても更新しません。ウィザードは、データベース内の列と一致する列がリポジトリに存在することを確認して、それらの値が一致しない場合は、リポジトリ内の列のタイプと長さを更新します。

各データベースの接続プール設定は、オブジェクトをバックエンド・データベースから「Physical」レイヤーに前回インポートしたときの接続プール設定と一致している必要があります。たとえば、Oracle の場合、接続プールがネイティブ OCI に設定されていることがありますが、更新には Oracle ODBC ソースを使用する必要があります。この場合、接続プールをインポートに使用した Oracle ODBC 設定に設定します。接続プール設定の詳細は、「接続プールの設定」（69 ページ）を参照してください。

「Physical」レイヤーのオブジェクトを更新するには

1. 「Tools」メニューから、「Utilities」→「Update Physical Layer」を選択し、「Execute」をクリックします。

   リポジトリの「Physical」レイヤーにあるデータベースが、ウィザードの左ペインに一覧表示されます。

2. 「Update Physical Layer Wizard」ダイアログ・ボックスの左ペインで、更新するデータベースを選択してから「Add」をクリックします。

   そのデータベースが右ペインに移動します。

3. 更新リストからデータベースを削除する場合は、そのデータベースを選択して「Remove」をクリックします。

4. 「Physical」レイヤーで更新するオブジェクトを選択したら、「次へ」をクリックします。

5. 次のウィンドウで、更新する各データベースの接続プールを選択し、「次へ」をクリックします。

   チェックアウトが必要なオブジェクトがある場合は、アラートが表示されます。

6 それぞれの更新情報を確認します。

注意：「Name」列見出しをクリックすると、行をソートできます（昇順または降順に切り替わる）。

7 ウィザードで、「Physical」レイヤーの特定のオブジェクトを更新しない場合は、「戻る」ボタンをクリックしてオブジェクトを削除します。

8 「Finish」をクリックします。

「Physical」レイヤーのオブジェクトが更新され、ウィザードが自動的に閉じられます。

9 Administration Tool のツールバーで、「File」→「Save」をクリックして、「Physical」レイヤーの更新済オブジェクトを保存します。

リポジトリ・マッピングのドキュメント生成

Repository Documentation・ユーティリティは、プレゼンテーション列から対応する論理列および物理列へのマッピングをドキュメント化します。このドキュメントの対象には、列に関連付けられている条件式も含まれます。ドキュメントは、カンマ区切り（CSV）またはタブ区切り（TXT）形式で保存できます。

Repository Documentation・ユーティリティを使用すると、Oracle BI メタデータをフラット・ファイルに抽出して、Excel および RDBMS にロードできるようになります。これにより、結果ファイルをクエリーして、「物理列 X を削除すると影響を受ける論理列はどれか」または「ビジネス・モデル内には物理テーブル W_SRVREQ_F への参照が何箇所あるか」などの質問に回答できます。その後、リポジトリの要素間の依存関係を確立できます。

Excel では、1,000,000 行のデータセットしか処理できませんが、大規模なリポジトリでは、この行数を超えてしまう場合があります。そのため、該当ビジネス・モデルを新規プロジェクトに抽出することにより、リポジトリのサブセットに対して Repository Documentation・ユーティリティを実行することをお薦めします。詳細は、「Oracle BI のマルチユーザー開発環境の設定と使用」（173 ページ）を参照してください。

Repository Documentation・ユーティリティでは、現行リポジトリのプレゼンテーション・レイヤーと物理レイヤーとの間の接続を示す、カンマ区切り値またはタブ区切り値のファイルが作成されます。このファイルは、物理レイヤーとしてリポジトリにインポートできます。

注意：このファイルでは、リポジトリ変数および Marketing オブジェクトに関する情報はすべて除外されます。

Repository Documentation・ユーティリティを実行するには

1 「Tools」メニューの「Utilities」を選択します。

2 「Utilities」ダイアログ・ボックスで、「Repository Documentation」をクリックします。

3 「Save As」ダイアログ・ボックスで、ファイルの保存先ディレクトリを選択します。

4 ファイルの名前を入力します。

5 ファイルのタイプおよびエンコーディング値を選択し、「Save」をクリックします。

現在のエンコーディング・オプションは、「ANSI」、「Unicode」および「UTF-8」です。

メタデータ・ディクショナリの生成と配置

Oracle BI を使用する際には、リポジトリ・オブジェクトのメトリックまたは属性について、より多くの情報が必要になる場合があります。たとえば、メタデータ・オブジェクトの名前が紛らわしいことが原因で発生する問題を解決したり、属性が複雑な方法で導出される際に詳細情報を取得する場合があります。メタデータ・ディクショナリは、矛盾する情報の解決やリポジトリ・オブジェクトのメトリックと属性の理解に役立ちます。

メタデータ・ディクショナリは、XML 文書の静的セットです。各 XML 文書には、物理テーブルや論理列などのメタデータ・オブジェクトについて、そのプロパティや他のメタデータ・オブジェクトとの関係もあわせて記述されています。

これらの XML 文書は、ブラウザの索引ファイルを使用して表示します。そのため、BI Presentation Server にメタデータ・ディクショナリを配置する必要があります。ディクショナリの生成前にその場所がわかる場合は、ディクショナリのファイルとフォルダを生成する前に「Save As」ダイアログ・ボックスで指定できます。ディクショナリの生成後に配置場所がわかった場合は、その時点でファイルを指定の配置場所にコピーできます。

ディクショナリは、リポジトリの変更に応じて動的に変更されることはありません。そのため、ディクショナリを定期的に生成して、その内容を更新する必要があります。

メタデータ・ディクショナリの生成

ディクショナリを生成する際には、出力先を最終的な配置場所または一時的な配置場所のいずれかに設定できます。

警告：リポジトリは、数万のオブジェクトを含む大規模なものになる場合があるため、大規模なリポジトリのディクショナリ生成には、かなりの時間がかかることがあります。

ディクショナリの出力先は、次の方法で選択できます。

- ディクショナリの生成先となる、ローカルまたはネットワーク上の配置場所を選択します。ディクショナリを生成する際には、リポジトリと同じ名前のサブディレクトリがその場所に作成されます。ディクショナリのフォルダとファイルは、そのサブディレクトリに作成されます。

  たとえば、リポジトリ名が demo1.rpd で J:¥BI_DataDictionary¥ を選択した場合、ディクショナリ・ファイルとスタイル・シートは、次の場所に配置されます。

  J:¥BI_DataDictionary¥demo1¥

  注意：使用するディクショナリが生成されている場合は、そのフォルダ全体を目的の配置場所にコピーしてください。

- IIS 仮想ディレクトリを使用する場合は、ディクショナリを生成する前に、IIS で仮想ディレクトリを作成または選択します。ディクショナリを生成する際は、IIS 仮想ディレクトリに関連付けられている物理ディレクトリを選択します。

  注意：ディクショナリが生成されている場合は、そのフォルダ全体を、IIS 仮想ディレクトリに関連付けられている物理ディレクトリにコピーしてください。

メタデータ・ディクショナリの生成後は、そのディクショナリのスタイル・シートと索引ファイルが作成されます。

関連するスタイル・シート（XSL ファイル）が作成され、次の場所に格納されます。

[drive]:¥[path]¥[repository name]¥xsl

名前索引とツリー索引が作成され、[drive]:¥[path]¥[repository name] のルート・フォルダに格納されます。オブジェクトの検索と表示には、次の URL を使用します。

- http://<hostname>:<portname>/<repository name>/NameIndex.xml。リポジトリ・オブジェクトを名前で検索できます。
- http://<hostname>:<portname>/<repository name>/TreeIndex.xml。トップレベル・オブジェクトから表示するオブジェクトにドリルダウンできます。

注意：各索引ファイルには、表示を迅速に切り替えられるように、もう一方の索引へのリンクがあります。

メタデータ・ディレクトリを生成するには

1. リポジトリをオフライン・モードで開きます。

   オンライン・モードでは、メタデータ・ディクショナリを生成できません。

2. 「Tools」メニューから、「Utilities」→「Generate Metadata Dictionary」を選択し、「Execute」をクリックします。
3. 「Choose Directory」ダイアログ・ボックスの「Browse」をクリックし、ディクショナリの格納場所を検索および選択します。

   「Metadata dictionary has been successfully created in [drive]:¥[path]¥¥¥」というメッセージが表示されます。

4. 「OK」をクリックします。

使用されていない物理オブジェクトの削除

リポジトリが大規模になると、サーバーのメモリー消費量が増え、メンテナンスが困難になります。さらに、大規模なリポジトリでは開発アクティビティに、より多くの時間がかかります。このユーティリティを使用すると、リポジトリ内の不要になったオブジェクトを削除できます。データベース、初期化ブロック、物理カタログおよび変数を削除できます。

使用されていない物理オブジェクトを削除するには

1. 「Tools」メニューから、「Utilities」→「Remove Unused Physical Objects」を選択し、「Execute」をクリックします。
2. 「Remove Unused Physical Objects」ダイアログ・ボックスの「Type」ドロップダウン・リストで、オブジェクト・タイプを選択します。
3. オブジェクトのリストで、削除するオブジェクトにのみチェック・マークが付いていることを確認します。

   オブジェクトのリストの下に、チェック済のオブジェクト数とオブジェクトの合計数が表示されます。

4. チェック済のオブジェクトを削除するには、「はい」をクリックします。
5. 取り消すには、「いいえ」をクリックします。

Aggregate Persistence Wizard

集計は、Oracle BI Server メタデータおよびバックエンド・データベースで作成および保存されます。この項では、Aggregate Persistence Wizard で集計テーブルの作成に使用する SQL ファイルを作成した後、作成された集計テーブルをメタデータにマップする方法について説明します。

Aggregate Persistence Wizard の使用ガイドライン

Oracle BI Server クエリーの集計の作成は、従来手動で行われていたプロセスで、データベースにテーブルを作成および移入する複雑な DDL および DML ファイルを作成する必要がありました。これに加え、これらのテーブルは、クエリーに使用できるように、リポジトリのメタデータにマップする必要があります。この作業は時間がかかるうえ、エラーが発生しやすいプロセスです。Oracle BI の管理者は、Aggregate Persistence Wizard を使用することで、これらの集計テーブルの作成とメタデータへのマッピングを自動化できます。次に、Aggregate Persistence Wizard を使用する際のガイドラインを示します。

- 「Tools」メニューから、「Utilities」→「Aggregate Persistence Wizard」を選択し、「Execute」をクリックします。
- 「Select file location」ダイアログ・ボックスで、集計作成スクリプトの完全パスとファイル名を指定します。ここでは、新規または既存のファイル名を指定できます。DDL ファイルを生成する場合は、「Generate DDL file」チェック・ボックスを選択します。
- 「Select Business Model & Measures」ダイアログ・ボックスで、項目を選択します。

  注意：「View Script」ボタンは、最初の集計テーブル・ブロックの作成中は使用できません。
- 「Select Dimensions & Levels」ダイアログ・ボックスでウィンドウを開いて、すべての列を表示します。ファクトとディメンションの結合に使用するサロゲート・キーの指定が必要な場合があります。

  ファクト集計テーブルとレベル集計テーブルとの結合オプションのデフォルトでは、レベル集計の主キーが使用されます。レベルの主キーが大きくて複雑なものであると、ファクト・テーブルへの結合は負荷の大きな処理となります。サロゲート・キーは人為的に生成するキーで、通常は数字を使用します。たとえば、レベル集計テーブルにサロゲート・キーが生成されると、この結合操作が簡素化され、ファクト・テーブルから不要な列（レベルの主キー）が削除され、ファクト・テーブルのサイズが小さくなります。
- 「Select Output Connection Pool, Container & Name」で、項目を選択します。デフォルトの集計テーブル名が表示され、接頭辞（NQSConfig.INI に定義されている）がファイル名に追加されます。
- 「Aggregate Definition」ダイアログ・ボックスの「View Script」ボタンが使用可能になり、確認用の論理 SQL が表示されます。ここで、別の集計を定義するか（デフォルト）、ウィザードを終了します。
- 「Complete Aggregate Script」ダイアログ・ボックスに、完全パスとファイル名が表示されます。

SQL ファイルを使用した集計テーブルの作成方法の詳細は、「Oracle BI Server のクエリーに関する集計の作成」（250 ページ）を参照してください。

Calculation Wizard

Calculation Wizard を使用すると、既存の 2 列を比較する新しい計算列を作成したり、メトリックを一括処理（集計処理）して作成することができます（NULL やゼロ除算ロジックなどの既存のエラー・トラップも含む）。

Calculation Wizard を「Business Model and Mapping」レイヤーで起動するには、数値データ型のディメンション列または論理ファクト列を右クリックし、「Calculation Wizard」オプションを選択します。Calculation Wizard の設定方法については、「「Options」ダイアログ・ボックスにおける「General」タブの使用」（31 ページ）を参照してください。

式ビルダー

Administration Tool で、式ビルダーの各ダイアログ・ボックスを使用すると、リポジトリ内で絞込み条件、集計およびその他の定義を作成できます。式ビルダーで作成する式は、SQL の式に類似しています。式ビルダーで作成したすべての式は、特に注記がないかぎり、Oracle BI Server に対する SQL クエリーで使用できます。

式ビルダーでの SQL の使用方法については、「SQL 構文およびセマンティクス」（366 ページ）を参照してください。Oracle BI Server でサポートされる SQL 関数については、「SQL リファレンス」（376 ページ）を参照してください。

この項の内容は次のとおりです。

式ビルダーのダイアログ・ボックスについて

式ビルダーは、次のダイアログ・ボックスからアクセスできます。

- Logical Table Source（「Content」タブ）
- Logical Table Source（「Column Mapping」タブ）
- Logical Column（「General」タブ）
- Logical Column（「Aggregation」タブ）
- Logical Foreign Key
- Physical Foreign Key
- Session Variable（「Variable」タブ）
- Static Repository Variable（「Variable」タブ）

式ビルダーのダイアログ・ボックスで式を作成する際には、カテゴリ・ペインと構築ブロック・ペインで検索します。検索ボックスに値を入力すると、一致しない文字列が除外され、一致するもののみが表示されます。検索ボックスに検索条件を入力した後は、矢印を使用してリストの上下に移動したり、タブで最初の検索ボックスと 2 つ目の検索ボックスの間を移動することができます。結果の完全なリストに戻すには、「Search」フィールドから検索文字列を削除します。

式に挿入する項目が見つかったら、その項目を選択し、ダイアログ・ボックスの「Insert」をクリックするか、キーボードの [Enter] キーを押します。選択した項目が、式ボックスの式に表示されます。

「Expression Builder」ダイアログ・ボックスを最初に開くときは、項目はソートされていません。「Sort Panes」チェック・ボックスを選択すると、ペイン内のすべての項目がソートされます。このチェック・ボックスを選択すると、ペインの自動的な再描画がすぐに実行されます。この際、ペインの内容やフィルタ条件は変更されません。

196 ページの図 15 に、式ビルダーの例を示します。このダイアログ・ボックスには、次のセクションがあります。

- ダイアログ・ボックスの上部の編集ペイン。現在の式を編集できます。
- ダイアログ・ボックスの中央のツールバー。一般的に使用される式の構築ブロックが含まれます。
- ダイアログ・ボックスの下半分にある「Selection」左ペイン。式ビルダーのアクセス元であるダイアログ・ボックスに応じたフォルダが表示されます。
- 下中央の「Categories」ペイン。「Selection」ペインで選択したフォルダで使用できるカテゴリが表示されます。中央ペインの下にある「Search」フィールドを使用すると、中央ペインの値を検索できます。
- 右下の「Building Blocks」ペイン。「Category」ペインで選択したカテゴリの個々の構築ブロックが表示されます。右ペインの下にある「Search」フィールドを使用すると、右ペインの値を検索できます。

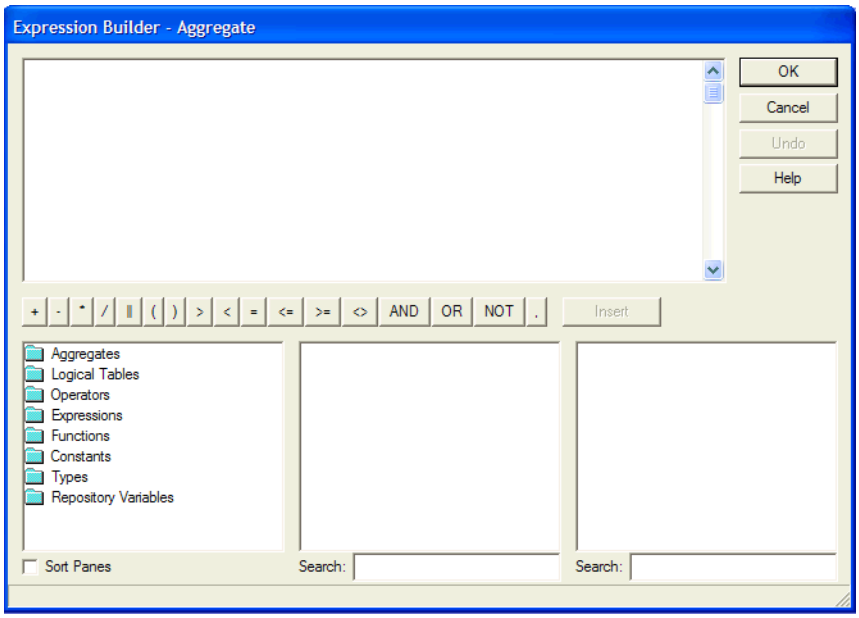


図 15. 式ビルダーの例

式ビルダーのツールバー

ツールバーは、式ビルダーの中ほどにあります。197 ページの表 22 では、式の各アイコンとその機能について説明します。

表 22. 式ビルダーのツールバー

演算子	説明
+	加算のプラス記号。
-	減算のマイナス記号。
*	乗算の乗算記号。
/	除算の除算記号。
||	文字列の連結。
(	開きカッコ。
)	閉じカッコ。
>	大なり記号。値が比較対象よりも大きいことを示す。
<	小なり記号。値が比較対象よりも小さいことを示す。
=	等号記号。値が同じであることを示す。
<=	以下記号。値が比較対象よりも小さいまたは等しいことを示す。
>=	以上記号。値が比較対象よりも大きいまたは等しいことを示す。
<>	不等号記号。値より大きいまたは小さいが同じでないことを示す。
AND	AND 連結語。1 つ以上の条件の論理積で複合条件を形成することを示す。
OR	OR 連結語。1 つ以上の条件の論理和で複合条件を形成することを示す。
NOT	NOT 連結語。条件に合致しないことを示す。
,	カンマ。リスト内の各要素を区切るために使用する。

「Selection」ペインのフォルダ

「Selection」ペインに表示されるフォルダは、式ビルダーのアクセス元のダイアログ・ボックスに応じて異なります。この項では、表示される可能性があるフォルダについて説明します。

Aggregate Content

「Aggregate Content」フォルダには、使用可能な集計関数があります。集計ソースは、ここに表示される関数の 1 つを使用してコンテンツ・レベルを指定する必要があります。

Dimensions

「Dimensions」フォルダには、ビジネス・モデルで構成されているディメンションがあります。ビジネス・モデルにディメンションが存在しない場合、または特定の式ビルダーにディメンション・フォルダが関連しない場合、「Dimension」フォルダは表示されません。

「Dimensions」フォルダを選択すると、構成済の各ディメンションが中央のペインに表示され、選したディメンションの各レベルが右ペインに表示されます。

Logical Tables

「Logical Tables」フォルダには、ビジネス・モデルで構成されている論理テーブルがあります。特定の式ビルダーに論理テーブルが関連しない場合、「Logical Tables」フォルダは表示されません。

「Logical Tables」フォルダを選択すると、ビジネス・モデルの各論理テーブルが中央のペインに表示され、選択した論理テーブルの各列が右ペインに表示されます。

Operators

「Operators」フォルダには、使用可能な SQL 論理演算子があります。

Expressions

「Expressions」フォルダには、使用可能な式があります。

Functions

「Functions」フォルダには、使用可能な関数があります。表示される関数は、選択したオブジェクトによって異なります。

Constants

「Constants」フォルダには、使用可能な定数があります。

Types

「Types」フォルダには、使用可能なデータ型があります。

Repository Variables

このフォルダには、使用可能なリポジトリ変数があります。リポジトリ変数が定義されていない場合、このフォルダは表示されません。

Session Variables

このフォルダには、使用可能なシステム・セッション変数および非システム・セッション変数があります。セッション変数が定義されていない場合、このフォルダは表示されません。

式の設定例

199 ページの図 16 に、導出された論理列の式ビルダーを示します。

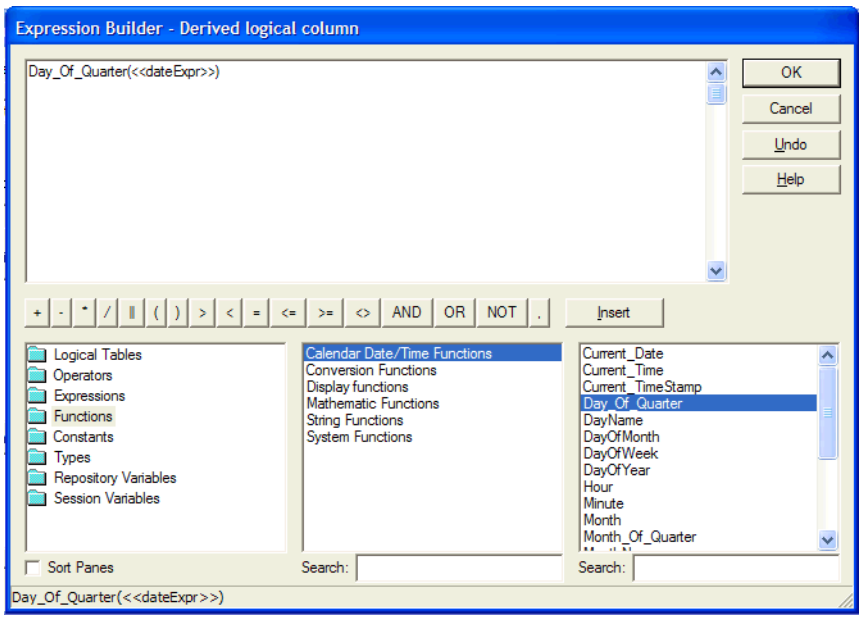


図 16. 導出された論理列の式ビルダー

左ペインで、「Functions」フォルダを選択します。右ペインの関数をダブルクリックして、編集ボックスに関数を貼り付けます。式ビルダーの編集ボックスで、関数のカッコの間を一度クリックして、その領域を関数の引数を追加するための挿入ポイントとして選択します。

論理列をダブルクリックして、その論理列を関数の引数として挿入ポイントに貼り付けます。200 ページの図 17 に、ウィンドウのどこに式が表示されるかを示します。

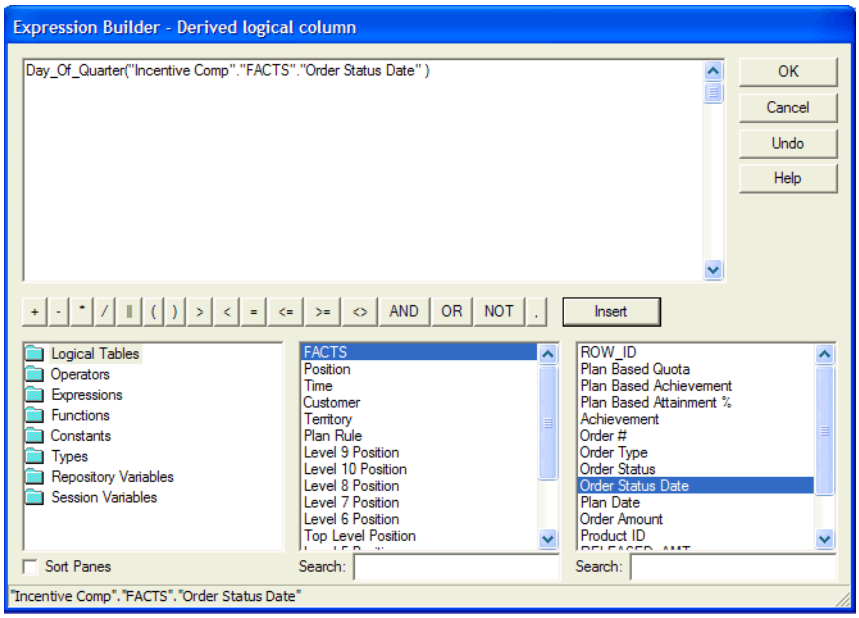


図 17. 編集ペインの論理列関数の例

式ビルダー内のナビゲート

次の手順で、「Expression Builder」ダイアログ・ボックス内をナビゲートします。

式ビルダー内をナビゲートするには

1. 「Selection」ペインで、作成する式タイプに適したフォルダを選択します。

   「Categories」ペインに、そのフォルダで使用できるカテゴリが表示されます。

2. 作成する式に適したカテゴリを選択します。

   「Building Blocks」ペインに、そのカテゴリで使用できる構築ブロックが表示されます。

3. 構築ブロックをダブルクリックして、「Editing」ペインに移動します。
4. 式に演算子を挿入する場合は、式ビルダーのツールバーで演算子をダブルクリックします。

式の作成

次の手順で、「Expression Builder」ダイアログ・ボックスで式を作成します。

式を作成するには

1. 式に含める個々の構築ブロックにナビゲートします。

   「Expression Builder」ダイアログ・ボックスの最下部にある「Syntax」バーには、式の構文が表示されます。

2. 構築ブロックを「Editing」ペインに追加します。
3. 構築ブロックを編集して、作成する式の構文を反映させます。
4. 式ビルダーのツールバーを使用して、演算子を式に挿入します。
5. 式が完成するまで前述の手順を繰り返し、「OK」をクリックします。

   式に構文エラーがある場合、Administration Tool によってメッセージが表示されます。式の構文が正しい場合、式ビルダーのアクセス元のダイアログ・ボックスにその式が追加されます。

時系列の変換関数について

時系列関数は、時系列的なディメンションを操作します。これらの関数を特定のディメンションで使用するには、そのディメンションを時間ディメンションに指定してから、1 つ以上のキーを 1 つ以上のレベルで時間順キーとして設定する必要があります。これにより、そのディメンションが、ある時間において単調増加する値である（時間順になる）ことが認識されます。

注意：時間順キーは、時系列クエリーへの回答に使用できるレベルで定義する必要があります。パフォーマンス上の理由により、関連するその他のレベルでも時間順キーを定義することをお薦めします。

現時点では、AGO と TODATE は、時系列変換関数タイプになっています。そのため、Administration Tool の式ビルダーでのみ、AGO と TODATE 関数を入力できます。これらは、コード化された SQL では使用できません。

Ago および ToDate 関数では、式ビルダーを使用して論理関数をコールすることによって時系列計算を実行できます。物理テーブルの別名を使用して論理的なモデリングを行う必要はありません。時系列関数は、標準的な SQL 日付操作関数ではなく、ユーザーが指定したカレンダ・テーブルに基づいて、Period Ago および Period to Date 関数を計算します。

次のクエリー例を使用して、時間クエリーでのナビゲートにおける重要な単位について説明します。

```
Select quarter, YearAgoSales:
```

- **クエリー単位**：リクエストの単位。このクエリー例では、クエリー単位は Quarter になります。
- **時系列単位**：リクエストされた集計の単位。このクエリー例では、時系列単位は Year になります。

  注意：時系列クエリーは、その時系列単位がクエリー単位かそれ以上である場合にのみ有効です。
- **ストレージ単位**：この例のクエリーは、日次売上、月次売上または四半期の売上から計算できます。集計ソースの単位は、集計単位と呼ばれます。

  注意：時間順キーは、このレベルで定義する必要があります。

202 ページの図 18 に、「Expression Builder」ダイアログ・ボックスの時系列関数を示します。

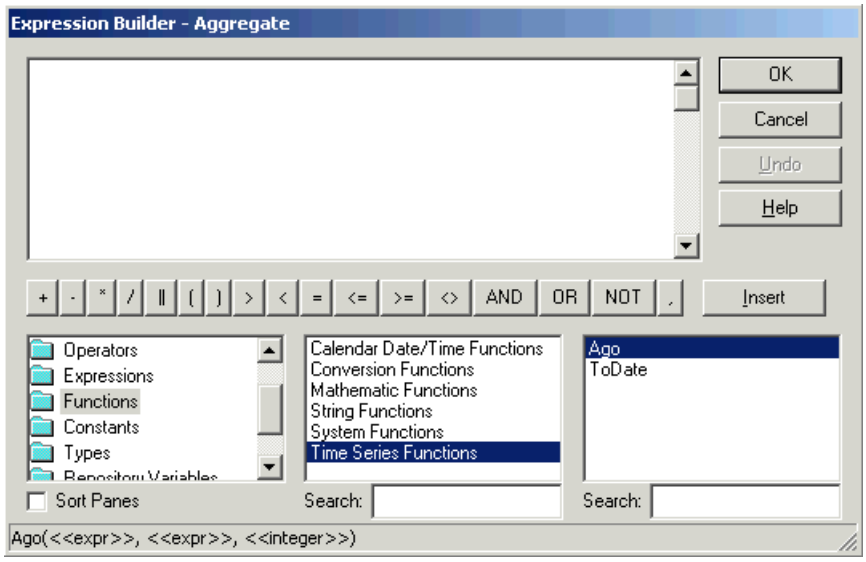


図 18. 式ビルダーの時系列関数の例

Ago

リレーショナル・データソースでのみ使用される時系列集計関数です。現在の時刻から指定された期間にさかのぼって集計値を計算します。たとえば、Ago 関数を使用すれば、現四半期の各月の売上と、対応する 1 つ前の四半期の売上を生成できます。すべての Ago 関数が同じレベルの引数を持つ場合は、複数の Ago 関数をネストできます。

注意：同じレベルの引数を持つ ToDate 関数と Ago 関数では、1 つの ToDate 関数と複数の Ago 関数をネストできます。

構文：

```
AGO(<measure_expression>, <model_id>.<dimension_id>.<level_id>, <integer_literal>)
```

この例では、<measure_expression> は少なくとも 1 つのメジャーを含む式、<model_id> はモデル識別子、<dimension_id> はディメンション識別子、<level_id> はレベル識別子、<integer_literal> は整数リテラルになります。次に、この構文の例を示します。

```
AGO(model.sales.revenue + 5, model.time.month, 3)
```

ToDate

リレーショナル・データソースでのみ使用される時系列集計関数です。ToDate は、指定された期間の最初から現在表示されている時刻までのメジャー属性を集計します。たとえば、この関数を使用すると、1 月 1 日から今日までの売上を計算できます。

サポートされていないメトリックがリクエストされると、NULL 値が返され、ロギング・レベルが 3 以上に設定されている場合は NQQuery.log ファイルに警告エントリが書き込まれます。ToDate 関数は、別の ToDate 関数内でネストすることはできません。

注意： 同じレベルの引数を持つ ToDate 関数と Ago 関数では、1 つの ToDate 関数と複数の Ago 関数をネストできます。

構文：

```
TODATE(<measure_expression>, <model_id>.<dimension_id>.<level_id>)
```

IndexCol 変換関数について

IndexCol 関数では、導出された論理列を作成できます。IndexCol を選択すると、次の関数テンプレートが自動的に生成されます。

```
IndexCol( <<integer literal>>, <<expr1>> [, <<expr2>>, ?-] )
```

詳細は、「IndexCol」（409 ページ）を参照してください。

9 Oracle BI リポジトリでの集計ナビゲーション用のフラグメンテーション・コンテンツの設定

この章の内容は次のとおりです。

集計ナビゲーションについて

集計テーブルには、一連のディメンション属性から集計したメジャーの事前計算された結果が格納されます。集計テーブルの各列には、指定された一連のレベルのデータが含まれます。たとえば、月次売上高のテーブルには、各製品の各店舗での月々の収益を事前に計算した合計が含まれます。このメタデータは、「Logical Table Source」ダイアログ・ボックスで構成します。論理ソースの詳細な作成手順については、「論理テーブル・ソース（マッピング）の作成と管理」（121 ページ）を参照してください。

この項では、集計ナビゲーションの使用方法と設定手順について説明します。

フラグメンテーション・コンテンツの指定

論理テーブル・ソースに特定レベルの一連のデータ全体が含まれていない場合は、含まれているデータの一部、つまりフラグメントを指定する必要があります。論理列のコンテンツは、「Logical Table Source」ダイアログ・ボックスの「Content」タブにある「Fragmentation Content」編集ボックスで記述します。

次の各例では、ソースのフラグメンテーション・コンテンツを指定する方法およびルールについて説明します。

単一列、値ベースの述語

述語 IN は、1 つの等価述語に置き換えるか、または複数の等価述語を OR 連結語で区切って置き換えることができます。

フラグメント 1:

```
logicalColumn IN <valueList1>
```

フラグメント *n*:

```
logicalColumn IN <valueListN>
```

単一列、範囲ベースの述語

フラグメント 1:

```
logicalColumn >= valueof(START_VALUE) AND logicalColumn < valueof(MID_VALUE1)
```

フラグメント 2:

```
logicalColumn >= valueof(MID_VALUE1) AND logicalColumn < valueof(MID_VALUE2)
```

フラグメント *n*:

```
logicalColumn >= valueof(MID_VALUEN-1) AND logicalColumn < valueof(END_VALUE)
```

開始ポイント、中間ポイント、終了ポイントは慎重に選択してください。

注意：述語 >= および < の使用方法に注意して、フラグメント・コンテンツの記述が重複しないことを確認してください。フラグメントごとに、上限値は < で表現する必要があります。<= を使用するとエラーが発生します。同様に、述語 BETWEEN を使用してフラグメント・コンテンツの範囲を記述することはできません。

valueof は、リポジトリ変数の値を表します（変数の詳細は、第 13 章「Oracle BI リポジトリの変数の使用」を参照してください）。式の中でリポジトリの値を使用する場合は、次の構文がフラグメント 2 について機能しないことに注意してください。

```
logicalColumn >= valueof(MID_VALUE1)+1 AND logicalColumn < valueof(MID_VALUE2)
```

valueof(MID_VALUE1)+1 のかわりに別のリポジトリ変数を使用してください。

valueof(MID_VALUE1) などの同じ変数を両方のフラグメントのコンテンツに表示する必要はありません。別の変数を指定して、次の形式の文を作成できます。

フラグメント 1:

```
logicalColumn >= valueof(START_VALUE) AND logicalColumn < valueof(MID_VALUE1)
```

フラグメント 2:

```
logicalColumn >= valueof(MID_VALUE2) AND logicalColumn < valueof(MID_VALUE3)
```

複数列コンテンツの記述

異なる列に対する任意の数の述語を、各コンテンツ・フィルタに含めることができます。それぞれの列の述語は値ベースでも範囲ベースでも構いません

フラグメント 1:

```
<logicalColumn1 predicate> AND <logicalColumn2 predicate > ... AND <logicalColumnM
predicate>
```

フラグメント *n*:

```
<logicalColumn1 predicate> AND <logicalColumn2 predicate > ... AND <logicalColumnM
predicate>
```

すべてのフラグメントに、同じ M 列の述語があることが理想です。論理列に述語制約がない場合、Oracle BI Server では、その論理列のすべての値に対応するデータがフラグメントに含まれているとみなされます。OR 述語を使用する例外については、「パラレル・コンテンツの記述」（207 ページ）を参照してください。

パラレル・コンテンツの記述

残念ながら、日付を処理するには前述の方法では不十分です。これは、year > year month > date、month > year month > date のように、論理列間に複数の階層関係があるためです。たとえば、年と月のように、別々の時点で表されるフラグメントについて考えます。年を基準に十分にさかのぼって制約すれば、履歴フラグメントのみの選択には十分です。次の例に示すように、これはパラレル OR 手法によってサポートされます。この例では、スナップショット月が 1999 年 4 月 1 日の午前 12 時であったと仮定します。関連する OR 連結語と述語は太字で示してあります。

フラグメント 1（履歴）:

```
EnterpriseModel.Period."Day" < VALUEOF("Snapshot Date") OR

EnterpriseModel.Period.MonthCode < VALUEOF("Snapshot Year Month") OR

EnterpriseModel.Period."Year" < VALUEOF("Snapshot Year") OR
```

```
EnterpriseModel.Period."Year" = VALUEOF("Snapshot Year") AND
EnterpriseModel.Period."Month in Year" < VALUEOF("Snapshot Month") OR

EnterpriseModel.Period."Year" = VALUEOF("Snapshot Year") AND
EnterpriseModel.Period."Monthname" IN ('Mar', 'Feb', 'Jan')
```

フラグメント 2（現在）：

```
EnterpriseModel.Period."Day" >= VALUEOF("Snapshot Date") OR

EnterpriseModel.Period.MonthCode >= VALUEOF("Snapshot Year Month") OR

EnterpriseModel.Period."Year" > VALUEOF("Snapshot Year") OR

EnterpriseModel.Period."Year" = VALUEOF("Snapshot Year") AND
EnterpriseModel.Period."Month in Year" >= VALUEOF("Snapshot Month") OR

EnterpriseModel.Period."Year" = VALUEOF("Snapshot Year") AND
EnterpriseModel.Period."Monthname" IN ('Dec', 'Nov', 'Oct', 'Sep', 'Aug', 'Jul',
'Jun', 'May', 'Apr')
```

論理モデルに日付レベルの詳細が不要な場合は、前述の例で EnterpriseModel.Period."Day" の述語を省略してください。

パラレル・コンテンツの記述トラックをサポートする OR 連結語の使用方法に注意してください。

例と説明

この項では、後述の説明と簡単に対応付けられるように、例にトラック n のラベルを付けてあります。これらのラベルは、実際のフラグメンテーション・コンテンツ文には含めないでください。

フラグメント 1（履歴）：

トラック 1 EnterpriseModel.Period."Day" < VALUEOF("Snapshot Date") OR

トラック 2 EnterpriseModel.Period.MonthCode < VALUEOF("Snapshot Year Month") OR

トラック 3 EnterpriseModel.Period."Year" < VALUEOF("Snapshot Year") OR

トラック 4 EnterpriseModel.Period."Year" = VALUEOF("Snapshot Year") AND
EnterpriseModel.Period."Month in Year" < VALUEOF("Snapshot Month") OR

トラック 5 EnterpriseModel.Period."Year" = VALUEOF("Snapshot Year") AND
EnterpriseModel.Period."Monthname" IN ('Mar', 'Feb', 'Jan')

たとえば、トラック 1 の EnterpriseModel.Period."Day" について考えます。履歴フラグメントでは、述語 < により、Snapshot Date より前の日付（Day）に基づいて制約するすべてのクエリーが履歴フラグメントに該当することが Oracle BI Server に対して示されます。反対に、現在のフラグメントでは、Day に述語 >= を指定することにより、現在のフラグメントに Snapshot Date より前のデータは含まれないことが示されます。

トラック 2 の MonthCode（199912 など）は Day に類似しています。スナップショットの日付が 4 月 1 日であり、月の記述は重複しないため、< および >= の述語を使用しています。覚えておくべき重要なルールとして、追加のパラレル・トラックでは、それぞれ別々の列セットを参照する必要があります。共通する列を使用できますが、全体的な列セットは一意である必要があります。Oracle BI Server では、列セットを使用して最適なトラックが選択されます。

トラック 3 の Year（履歴フラグメントでは <、現在のフラグメントでは >）では、年のみに基づいて制約するクエリーで、最適な（単一の）フラグメントを選択できることが Oracle BI Server に対して示されます。たとえば、Year IN (1997, 1998) の論理クエリーでは履歴フラグメントのみがヒットします。同様に、Year = 2000 のクエリーでは、現在のフラグメントのみがヒットする必要があります。ただし、このトラックに記述されているコンテンツでは 1999 年にヒットするクエリーに回答できないため、後続のトラックに追加情報がないかぎり、両方のフラグメントがヒットします。

トラック 4 では、Year および Month in Year（月を示す整数）に関してフラグメント・セットを記述します。前述の複数列コンテンツの記述方法を使用します。< および >= の述語の使用方法に注目してください。これら 2 つの列には、あいまい性および重複がありません。

トラック 5 では、Year および Monthname に関してフラグメント・コンテンツを記述します。値ベースの IN 述語を使用します。

応用として、スナップショットの日付が、ある月の特定の日に当たるため、年や月のみに基づく複数列コンテンツの記述が、特定のスナップショット月で重複する場合を考えます。このあいまい性を指定するには、<= および >= の述語を使用します。

フラグメント 1（履歴）：

```
EnterpriseModel.Period."Day" < VALUEOF("Snapshot Date") OR

EnterpriseModel.Period.MonthCode <= VALUEOF("Snapshot Year Month") OR

EnterpriseModel.Period."Year" < VALUEOF("Snapshot Year") OR

EnterpriseModel.Period."Year" = VALUEOF("Snapshot Year") AND
EnterpriseModel.Period."Month in Year" <= VALUEOF("Snapshot Month") OR

EnterpriseModel.Period."Year" = VALUEOF("Snapshot Year") AND
EnterpriseModel.Period."Monthname" IN ('Apr', 'Mar', 'Feb', 'Jan')
```

フラグメント 2（現在）：

```
EnterpriseModel.Period."Day" >= VALUEOF("Snapshot Date") OR

EnterpriseModel.Period.MonthCode >= VALUEOF("Snapshot Year Month") OR

EnterpriseModel.Period."Year" > VALUEOF("Snapshot Year") OR

EnterpriseModel.Period."Year" = VALUEOF("Snapshot Year") AND
EnterpriseModel.Period."Month in Year" >= VALUEOF("Snapshot Month") OR

EnterpriseModel.Period."Year" = VALUEOF("Snapshot Year") AND
EnterpriseModel.Period."Monthname" IN ('Dec', 'Nov', 'Oct', 'Sep', 'Aug', 'Jul',
'Jun', 'May', 'Apr')
```

不均衡なパラレル・コンテンツの記述

受注入力アプリケーションでは、通常、履歴フラグメントと現在のフラグメントの間で時間ベースのフラグメンテーションを行うだけでは不十分です。たとえば、スナップショット日より前にデータベースに入力された履歴レコードでも、まだ揮発的である場合があります。

次の例では、出荷されるか取り消されるまで、未出荷の注文はアプリケーションによって直接更新されるものと仮定します。ただし出荷された後、注文に対して可能な変更は、個別に相殺する返送処理の入力だけです。

次のコンテンツの記述には、2 つのパラレル・トラックがあります。1 つ目のトラックでは、前述の項で説明した複数列のパラレル・トラック手法を使用します。Shipped または Canceled の注文状況を示す複数列コンテンツの記述内にある、パラレル・カレンダ記述を囲むカッコに注意してください。

2 つ目のパラレル・トラックは現在のフラグメントのみに存在し、未出荷（Open）のレコードがすべて現在のフラグメントのみにあることを示します。

フラグメント 1（履歴）：

```
Marketing."Order Status"."Order Status" IN ('Shipped', 'Canceled') AND

Marketing.Calendar."Calendar Date" <= VALUEOF("Snapshot Date") OR

Marketing.Calendar."Year" <= VALUEOF("Snapshot Year") OR

Marketing.Calendar."Year Month" <= VALUEOF("Snapshot Year Month")
```

フラグメント 2（現在）：

```
Marketing."Order Status"."Order Status" IN ('Shipped', 'Canceled') AND

Marketing.Calendar."Calendar Date" > VALUEOF("Snapshot Date") OR

Marketing.Calendar."Year" >= VALUEOF("Snapshot Year") OR

Marketing.Calendar."Year Month" >= VALUEOF("Snapshot Year Month")

OR Marketing."Order Status"."Order Status" = 'Open'
```

パラレル・トラックがある場合は重複が許可されるため、2 つのフラグメントで Year と Month の記述が重複していても問題はありません。ルールは、少なくとも 1 つのトラックが重複していないことです。その他のトラックには重複があっても構いません。

集計テーブルのフラグメント

集計の特定レベルの情報が、複数の物理テーブルに格納される場合があります。特定レベルの個々のソースにドメインの一部またはフラグメントの情報が含まれている場合、クエリーの適切なソースを選択するには、Oracle BI Server でソースのコンテンツを認識する必要があります。

たとえば、全店舗のソフト・ドリンクの売上を追跡するデータベースがあるとします。データのディテール・レベルは店舗レベルです。211 ページの図 19 に示すように、Coke と Pepsi の売上については集計情報が都市レベルで格納されますが、7-Up やその他のソーダの売上については集計情報はありません。

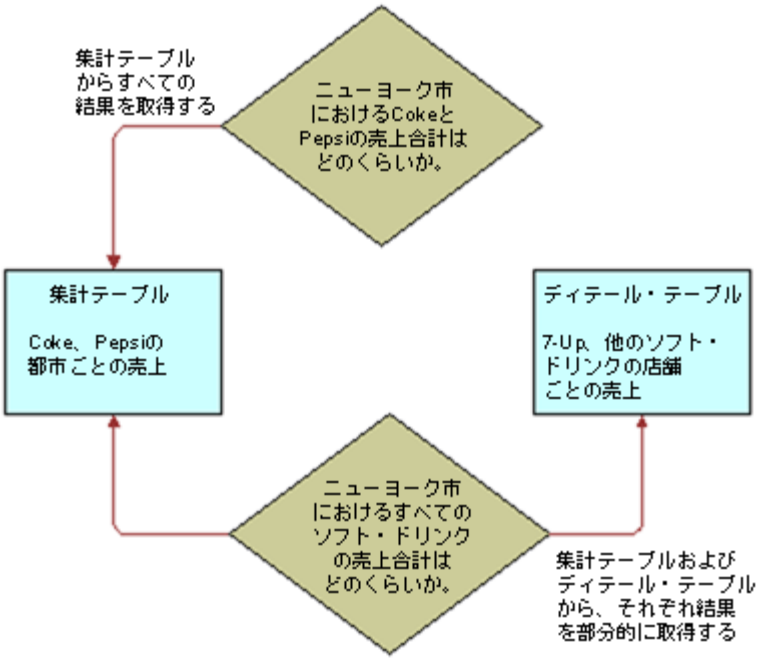


図 19. 集計情報

この構成の目的は、集計テーブルを最大限利用することにあります。Coke と Pepsi の販売総額をクエリーした場合、データは集計テーブルから返されます。すべてのソフト・ドリンクの販売総額をクエリーした場合、データは、Coke と Pepsi については集計テーブルを使用して、その他のブランドについてはディテール・テーブルを使用して返されます。

Oracle BI Server は、このような部分集計ナビゲーションに対応しています。ドメインが複数のフラグメントにまたがるクエリーに対して集計フラグメントを使用するようにリポジトリを構成するには、集計データのレベルごとにドメイン全体を定義する必要があります。これは、要約が不十分な物理ソースに基づいているとして集計フラグメントを構成する必要がある場合でも同様です。

集計テーブルのコンテンツの指定

集計テーブルのナビゲーションは、論理テーブル・ソース・マッピングで構成します。ソフト・ドリンクの例では、集計テーブルには Coke と Pepsi の売上データが都市レベルで含まれています。「Aggregate content」の指定（「Logical Table Source」ウィンドウの「Content」タブ）は次のようになります。

「Group by logical level:」

```
GeographyDim.CityLevel, ProductDim.ProductLevel
```

「Fragmentation content」の指定（「Logical Table Source」ダイアログの「Content」タブ）は次のようになります。

```
SoftDrinks.Products.Product IN ('Coke', 'Pepsi')
```

このコンテンツの指定により、2 つの製品の都市レベルおよび製品レベルのデータがソース・テーブルに含まれることが Oracle BI Server に対して示されます。また、このソースは、このレベルのデータのフラグメントであるため、「Logical Table Source」ダイアログ・ボックスの「Content」タブで「This source should be combined with other sources at this level」オプションを選択して、ソースが同じレベルの他のソースと結合することを示す必要があります。詳細は、「フラグメンテーション・コンテンツの指定」（206 ページ）を参照してください。

SELECT 文で物理レイヤー・テーブルを定義してドメインを完成

ドメインの残りのデータ（その他の種類のソーダ）は、すべて店舗レベルで格納されます。集計レベル（この例では都市と製品）でドメイン全体を定義するには、このレベルのドメインの残りを含むソースが必要です。店舗レベルのデータは都市レベルのデータよりもレベルが下（つまり、より詳細）であるため、ある都市の全店舗の製品売上データを合計することにより、店舗と製品の詳細から都市と製品レベルの詳細を計算できます。これは、店舗と製品レベルのテーブルを使用するクエリーで実行できます。

これを行う方法の 1 つは、「Physical」レイヤーで、店舗レベルの計算を返す SELECT 文を使用してテーブルを定義することです。テーブルを定義するには、SELECT 文でクエリーを行う物理スキーマ・フォルダを選択することによって「Physical」レイヤーにテーブルを作成し、New Table コマンドを実行します。「Object Type」ドロップダウン・リストから「Select」を選択して、右ペインに SQL 文を入力します。

この SQL では、他の集計テーブルのレベルでドメインを完成させる仮想テーブルを定義する必要があります。ここでは既存の集計テーブルは 1 つであり、Coke と Pepsi のデータが都市ごとに含まれています。したがって、SQL 文では、Coke と Pepsi のデータを除くすべてのデータを都市レベルで返す必要があります。

SQL 仮想テーブルのコンテンツの指定

次に、ドメインの残りを都市と製品レベルで網羅する、Sales 列用の論理テーブル・ソースを新規に作成します。このソースには、前述の項で作成した仮想テーブルが含まれます。この仮想テーブルで、Dollars 論理列を USDollars 物理列にマップします。

このソースの「Aggregate content」の指定（「Logical Table Source」ダイアログの「Content」タブ）は次のとおりです。

「Group by logical level:」

```
GeographyDim.CityLevel, ProductDim.ProductLevel
```

これにより、このソースに都市と製品レベルのデータが含まれることが Oracle BI Server に対して示されます。

「Fragmentation content」の指定は次のようになります。

```
SoftDrinks.Products.Product = '7-Up'
```

また、Coke および Pepsi の都市と製品レベルのデータを含む集計テーブルと結合してドメインを完成させるため、「Logical Table Source」ダイアログの「Content」タブで、このソースが同じレベルの他のソースと結合することを示すオプションを選択する必要があります。

仮想テーブルの物理結合

仮想テーブルの正しい物理結合を構成します。213 ページの図 20 では、CityProductSales2 は Cities テーブルおよび Products テーブルと結合しています。

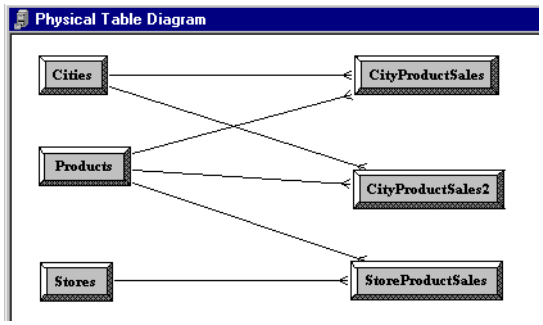


図 20. 物理結合の例

この例では、ソーダ売上のドメイン全体が 2 つのソースで構成されています。1 つのドメインに複数のソースを含めることができます。すべてのソースに該当するルールとして、互いに結合した場合に、該当するレベルの値のドメイン全体を構成するソースが各レベルに含まれるようにする必要があります。各レベルでドメイン全体を設定すると、Coke、Pepsi および 7-Up を要求するクエリーで 7-Up が除外されません。また、事前計算されて集計テーブルに格納されている情報をリクエストするクエリーで、その情報を集計テーブルから取得できます。これは、クエリーで集計テーブルに格納されていない他の情報をリクエストした場合でも同様です。

10 Oracle BI Server のクエリー環境の管理

Oracle BI Server はサーバーベースのクエリー環境で、サーバーベースのシステムの管理に関連する数多くのツールを備えています。この章では、これらのツールを使用して様々な管理アクションを実行する方法を説明します。このアクションには、サーバーの起動と停止、ログ・ファイルの確認と分析など、マルチユーザー環境の管理に関連するタスクがあります。

この章の内容は次のとおりです。

Oracle BI Server の起動

クエリーを処理する前に、Oracle BI Server を起動しておく必要があります。次の項では、Oracle BI Server の起動方法、Oracle BI Server のユーザー ID の変更方法、および Oracle BI Server の起動に失敗した場合の処置の 3 点について説明します。

Windows サービスからのサーバーの起動

この手順を実行するには、ユーザー ID が、Oracle BI Server がインストールされているローカル・マシンの Windows 管理者グループのメンバーであることが必要です。

Oracle BI Server を起動するには

1. サーバーがインストールされているマシンで、「スタート」→「設定」→「コントロール パネル」→「サービス」を選択します。

   「管理ツール」サブメニューを開くことが必要になる場合もあります。

2. 「Oracle BI Server」サービスを選択して「開始」をクリックします。

   サービスを開始していることを示すメッセージが表示されます。NQSConfig.INI ファイルの Repositories セクションにリポジトリがロードされるまで、サーバーの起動に数分かかることがあります。

   起動が完了すると、サービスの開始メッセージが消え、サービスの状態が「開始」に変わります。Oracle BI のインストール・フォルダの Log サブディレクトリにある NQServer.log ファイルに、次の情報が記録されます。

   - 起動時刻
   - ロードされたビジネス・モデル
   - 発生したエラー

   起動が完了しない場合は NQSConfig.INI ファイルを調べ、リポジトリのファイル名の間違いなどのエラーがないか確認します。サーバーが起動していないことを示すメッセージが表示された場合は、状態が「開始」に変わるまで定期的に状態をリフレッシュします。

Windows でのサーバーの自動スタートアップの構成

次の手順では、Windows NT または Windows 2000 を起動したときに Oracle BI Server が自動的に起動するように構成する方法について説明します。

Oracle BI Server の自動スタートアップを構成するには

1 サーバーがインストールされているマシンで、「スタート」→「設定」→「コントロール パネル」→「サービス」を選択します。

「管理ツール」サブメニューを開くことが必要になる場合もあります。

2 「サービス」ダイアログ・ボックスで「Oracle BI Server」サービスをダブルクリックします。

3 「Oracle BI Server のプロパティ」ダイアログ・ボックスにある「スタートアップの種類」ドロップダウン・リストで「自動」を選択し、「OK」をクリックします。

これで、Windows を起動したときに Oracle BI Server サービスも自動的に起動するようになります。

UNIX でのサーバー起動スクリプトの実行

次のいずれかのスクリプトを実行して、Oracle BI Server を起動します。

- sh または bash を使用する場合：

```
run-sa.sh start
```

- csh を使用する場合：

```
run-sa.csh start
```

- 環境の設定に sa.sh または sa.csh を使用した場合：

```
nqscomgateway.exe &
```

- 標準シェルの場合：

```
nohup nqscomgateway.exe >/dev/null 2>&1 &
```

- C シェルの場合：

```
nohup nqscomgateway.exe >&/dev/null &
```

Oracle BI Server を実行するユーザー ID の変更

Windows では、Oracle BI Server は Windows サービスの 1 つとして実行されます。デフォルトでは、ローカル・システム・アカウントで実行されます。このサーバーからリモート・マシン上のデータベースにアクセスする必要がある場合は、適切なネットワーク権限が設定されたユーザー ID でサーバーを実行する必要があります。さらに、このユーザー ID は、ローカル・マシンの Windows NT 管理者グループのメンバーであることが必要です。

Oracle BI Server を実行するユーザー ID を変更するには、コントロール・パネルのサービス・ユーティリティを使用します。

ユーザー ID を変更するには

1 Oracle BI Server がインストールされているマシンで、「スタート」→「設定」→「コントロール パネル」を選択します。

2 コントロール・パネルで「サービス」アイコンをダブルクリックします。

3 「Oracle BI Server」サービスをダブルクリックして「開始」をクリックします。

4 「Oracle BI Server のプロパティ」ダイアログ・ボックスの「ログオン」領域で「アカウント」を選択し、そのテキスト・ボックスの右にあるボタンをクリックします。

5 「ユーザーの追加」ダイアログ・ボックスで、Oracle BI Server を実行するユーザー・アカウントを選択して「追加」をクリックし、「OK」をクリックします。

6 「サービス」ダイアログ・ボックスでこのユーザーのパスワードを入力し、確認のためにもう一度そのパスワードを入力して「OK」をクリックします。

これで、新しいユーザー ID でサーバーが実行されるようになりました。このサービスを次回開始すると、サービスは新しいユーザー・アカウントで開始しようとします。

サーバーの起動に失敗した場合

起動処理が失敗した場合は、原因を示すメッセージが次のログ・ファイルに記録されていないか確認します。

- Windows NT および Windows 2000 では、「スタート」→「プログラム」→「管理ツール」→「イベントビューア」を選択して、Windows イベント・ログを確認します。
- Windows NT、Windows 2000 および UNIX では、NQServer.log ファイルを確認します。このファイルは、Oracle BI Server ソフトウェアのインストール・フォルダ（¥OracleBI）の下の Log フォルダにあります。このファイルは、テキスト・エディタを使用して表示できます。
- UNIX では、/usr/sbin/syslogd を実行して、システムおよび Oracle BI Server に関連するメッセージが記録されていないか確認します。

これらのログ・ファイルには、サーバーの起動が失敗した原因を示すメッセージが記録されています。たとえば、NQSConfig.INI ファイルに構文エラーがあると、オペレーティング・システムのログ・ファイルと NQServer.log ファイルの両方に、構文エラーに関するメッセージが記録されます。これらのログ・メッセージを調べ、問題を修正した後、サーバーを再起動します。

Oracle BI Server の停止

Oracle BI Server は、次のいずれかの方法で停止できます。

- Windows サービスでのサーバーの停止（219 ページ）
- Windows のコマンド・プロンプトからのサーバーの停止（219 ページ）
- UNIX でのサーバー停止スクリプトの実行（220 ページ）
- Administration Tool の使用による Oracle BI Server の停止（220 ページ）

サーバーを停止すると、アクティブなクライアント接続はエラー・メッセージを受け取り、ただちに終了します。また、基礎となるデータベースに対する未処理のクエリーは取り消されます。

Windows サービスでのサーバーの停止

次の手順では、Windows のコントロール・パネルにあるサービス・アプレットから Oracle BI Server を停止する方法について説明します。

サービス・アプレットからサーバーを停止するには

1. Oracle BI Server がインストールされているマシンで、「スタート」→「設定」→「コントロール パネル」を選択します。
2. コントロール・パネルで「サービス」アイコンをダブルクリックして、サービス・アプレットを起動します。
3. 「Oracle BI Server」サービスを選択して「停止」をクリックします。

   サービスを停止していることを示すメッセージが表示されます。

停止が完了すると、コントロール・パネルに表示されているサービスの状態は空白となり、NQServer.log にログ・メッセージが書き込まれます。このファイルは、ソフトウェアのインストール・フォルダの下の Log フォルダ（¥OracleBI¥Log）にあります。

Windows のコマンド・プロンプトからのサーバーの停止

次の手順に従って、Windows のコマンド・プロンプトから Oracle BI Server を停止します。

Windows のコマンド・プロンプトから Oracle BI Server を停止するには

- Oracle BI Server を実行しているマシンで次のコマンドを入力します。

  ```
  nqsshutdown {-d <data_source_name> -u <user ID> -p <password>}
  ```

  内容は次のとおりです。

  data_source_name — 停止処理を実行するためにサーバーへの接続に使用される、クラスタ化されていない ODBC データ ソースの名前。このデータ ソースは、Oracle BI の管理者グループのメンバーである Oracle BI ユーザーとしてサーバーに接続する必要があります。サーバーを停止できるのは、Oracle BI の管理者として定義されているユーザーのみです。

  注意： クラスタ化された DSN に対しては、nqsshutdown コマンドは無効です。（クラスタ化されていない）標準の DSN を渡すと、目的の Oracle BI Server は、クラスタを構成していても停止します。

  user ID — 停止処理を実行するために Oracle BI Server に接続するユーザー。

  password — 停止処理を実行するために Oracle BI Server に接続するユーザー ID のパスワード。

  停止が完了すると、それを示すメッセージがコマンド・ウィンドウに表示されます。

UNIX でのサーバー停止スクリプトの実行

次のいずれかのスクリプトを実行して、Oracle BI Server を停止します。

- sh または bash を使用する場合

  ```
  run-sa.sh stop
  ```

- csh を使用する場合

  ```
  run-sa.csh stop
  ```

- 環境の設定に sa.sh または sa.csh を使用した場合

  ```
  nqsshutdown.exe -uAdministrator
  ```

Administration Tool の使用による Oracle BI Server の停止

次の手順では、Administration Tool を使用してサーバーを停止する方法について説明します。クラスタ化されていない DSN を使用して、オンライン・モードでリポジトリを開く必要があります。

注意： サーバーを停止するには、DSN は管理者権限が設定されたユーザーとしてログインする必要があります。

Administration Tool を使用してサーバーを停止するには

1. 「スタート」→「プログラム」→「Oracle BI」→「Oracle BI Administration Tool」を選択して Administration Tool を起動します。
2. サーバーにロードされたリポジトリをオンライン・モードで開きます。
3. 「File」→「Shut Down Server」を選択します。
4. 停止してよいか確認するダイアログ・ボックスが表示されたら、「はい」をクリックします。

   これによりサーバーが停止し、Administration Tool のオンライン・セッションが終了します。クラスタ化された DSN を使用して接続している場合は、Cluster Manager を使用して個々の Oracle BI Server インスタンスをオフラインにします。詳細は、「Cluster Manager の使用」（316 ページ）を参照してください。

サーバーへのユーザーの接続

Oracle BI のリポジトリ内でビジネス・モデルに接続するユーザーは、データソースを設定する必要があります。DSN 接続の設定の詳細は、「Oracle BI ODBC データソース名（DSN）の構成」（268 ページ）を参照してください。

Administration Tool は、リモート・マシンからオンライン・モードまたはオフライン・モードで実行することもできます。

クエリー・ログの管理

Oracle BI Server では、個々のユーザー・レベルでクエリー・アクティビティをログに記録できます。このクエリーのログ・ファイルには、NQQuery.log というファイル名が付いています。このファイルは、Oracle BI のインストール・フォルダの Log サブディレクトリにあります。ログの記録は、Oracle テクニカル・サポートが実施する品質保証テスト、デバッグおよびトラブルシューティングで使用します。本番モードでは、クエリー・ロギングは無効にすることが普通です。

Oracle BI Server のクエリー・ロギングは、ユーザー・レベルで追跡されます。ユーザーのコミュニティ全体を追跡すると、リソースを多大に消費するプロセスとなります。

注意：本番のシステムでは、目的のユーザー・コミュニティに対してのみ、クエリー・ロギングを有効にすることをお薦めします。ほとんどのユーザーに対しては、ロギング・レベルを 0（ゼロ）に設定します。本番システムでは、本番レベルのロギング機能として Usage Tracking を使用できます。詳細は、「使用状況トラッキングの管理」（226 ページ）を参照してください。

テスト対象のユーザーであることがユーザー名から明らかなユーザーのみをテストし、クエリー・ロギングが有効であることを確認しておくことをお薦めします。このようなユーザーに対してロギングを有効にした場合は、sales_admin_with_logging、sales_dev_with_logging、sales_test_with_logging など、ユーザーを容易に識別できるような名前をこれらのユーザーに付けることをお薦めします。クエリー・ロギングはリソースを消費するので、本番の Oracle BI の管理者ログインであっても、クエリー・ロギングは有効にしないようにします。

次の場合でも、クエリー・ロギングを無効にします。

- 初期化文字列における SQL 文。初期化文字列のテキスト・ボックスは、「Initialization」ダイアログ・ボックスの「General」タブにあります。

  注意：LOGGING 列は、ロギング・レベルとして格納されている値を参照します。

- 本番環境におけるユーザーごとに、ロギング・レベルを 0（ゼロ）に設定する必要があります。「Logging level」フィールドは、「User」ダイアログ・ボックスの「User」タブにあります。

ロギング・システムの構成

この項では、ロギング・システムについて説明し、クエリー・ログのサイズ設定、ロギング・レベルの選択、およびユーザーに対するクエリー・ログの有効化に関する情報を扱います。

クエリー・ロギングでは巨大なサイズのログ・ファイルが生成されることがあるので、デフォルトでは、ロギング・システムはオフになっています。ロギングを有効にすると役に立つのは、リポジトリの構成が正しいことをテストする場合、システム上のアクティビティを監視する場合、パフォーマンス上の問題を解決する場合、テクニカル・サポートを支援する場合などです。クエリーをログに記録するユーザーごとに、システム上でロギングを有効にする必要があります。

ログ・ファイルのサイズの制御

NQSConfig.INI ファイルの User Log セクションにある USER_LOG_FILE_SIZE パラメータで、NQQuery.log ファイルのサイズが決まります。このログ・ファイルのサイズが大きくなり、USER_LOG_FILE_SIZE パラメータで指定されたサイズの半分に達すると、そのファイル名は NQQuery.log.old に変更され、新しいログ・ファイルが自動的に作成されます（これによって、このログ・ファイルに割り当てられたディスク領域は、構成ファイルで指定されたサイズを超過しなくなります）。保持される古いファイルのコピーは 1 つのみです。

このファイル・サイズは、デバイス上で使用できる領域の大きさを上限として、任意の値に設定できます。USER_LOG_FILE_SIZE パラメータの値を変更した場合、その変更を有効にするには、Oracle BI Server を再起動する必要があります。USER_LOG_FILE_SIZE パラメータの構文については、『Oracle Business Intelligence Infrastructure インストレーションおよび構成ガイド』を参照してください。

ロギング・レベルの設定

ロギング・レベルは個々のユーザーで有効にできますが、グループ単位で構成することはできません。

注意： セッション変数はユーザーのロギング・レベルより優先されます。たとえば、Oracle BI の管理者のロギング・レベルが 4 に定義され、セッション変数のロギング・レベルがリポジトリでデフォルトの 0（ゼロ）に定義されていると、Oracle BI の管理者のロギング・レベルは 0 になります。

目的とするロギングの量に基づいてロギング・レベルを設定します。通常の運用では、ロギングを無効にすることが普通です（ロギング・レベルを 0 に設定します）。ロギングを有効にする場合は、1 または 2 のロギング・レベルを選択します。この 2 つのレベルは Oracle BI の管理者向けに設計されています。

クエリーのロギング・レベルを一時的に設定して、パフォーマンスやデータの問題を診断することが必要になる場合もあります。SELECT 文の前に次の文を記述することで、特定のクエリーに対してロギングを有効にできます。

```
Set Variable LOGLEVEL=n;
```

次に例を示します。

```
Set Variable LOGLEVEL=5; select year, product, sum(revenue) from time, products,
facts
```

このクエリーでは、基礎となる LOGLEVEL 変数の値に関係なく、ロギング・レベル 5 が使用されます。

注意： 3 以上のロギング・レベルは、テクニカル・サポートの助言に従って使用してください。

ロギング・レベルの説明は、222 ページの表 23 に記載されています。

表 23. ロギング・レベル

ロギング・レベル	ログに記録される情報
レベル 0	ロギングなし。
レベル 1	クライアント・アプリケーションから発行された SQL 文を記録します。 クエリーのコンパイル、クエリーの実行、クエリー・キャッシュの処理、バックエンド・データベースの処理の各経過時間を記録します。 クエリーのステータス（成功、失敗、終了、またはタイムアウト）を記録します。各クエリーのユーザー ID、セッション ID およびリクエスト ID を記録します。

表 23. ロギング・レベル

ロギング・レベル	ログに記録される情報
レベル 2	レベル 1 で記録された情報をすべて記録します。 これに加え、クエリーごとに、リポジトリ名、ビジネス・モデル名、プレゼンテーション・カタログ（Answers のサブジェクト領域）名、物理データベースに対して発行されたクエリーの SQL、キャッシュに対して発行されたクエリー、物理データベースに対して発行された各クエリーおよびキャッシュに対して発行されたクエリーから返された行数、およびクライアント・アプリケーションに返された行数を記録します。
レベル 3	レベル 2 で記録された情報をすべて記録します。 これに加え、キャッシュに置かれると思われるクエリーがキャッシュに挿入されなかった場合、現在のクエリーを置くためにキャッシュから既存のエントリが削除された場合、および完全一致検出の更新に失敗した場合に、該当の論理クエリー計画のログ・エントリを追加します。 このレベルは、テクニカル・サポートの助言がない場合は選択しないでください。
レベル 4	レベル 3 で記録された情報をすべて記録します。 これに加え、クエリー実行計画を記録します。このレベルは、テクニカル・サポートの助言がない場合は選択しないでください。
レベル 5	レベル 4 で記録された情報をすべて記録します。 これに加え、実行計画の様々な時点における中間行カウントを記録します。このレベルは、テクニカル・サポートの助言がない場合は選択しないでください。
レベル 6 および 7	将来使用するために予約されています。

ユーザーのロギング・レベルを設定するには

1. Administration Tool で、「Manage」→「Security」を選択します。

   「Security Manager」ダイアログ・ボックスが表示されます。

2. 目的のユーザーのユーザー ID をダブルクリックします。

   「User」ダイアログ・ボックスが表示されます。

3. 「Logging Level」フィールドの横にある上矢印または下矢印をクリックしてロギング・レベルを設定します。

ユーザーのロギング・レベルを無効にするには

- ロギング・レベルを 0 に設定します。

ログ・ビューアの使用

クエリー・ログを表示するには、Oracle BI のログ・ビューア・ユーティリティである nQLogViewer（またはテキスト・エディタ）を使用します。クエリー・ログの各エントリは、クエリーを発行したユーザーのユーザー ID、クエリーが開始されたセッションのセッション ID、および個々のクエリーのリクエスト ID でタグ付けされています。

nQlogViewer ユーティリティを実行するには、コマンド・ウィンドウを開き、引数を任意に組み合せて「nQlogviewer」と入力します。この構文は次のとおりです。

```
nqlogviewer [-u<user_ID>] [-f<log_input_filename>]
            [-o<output_result_filename>]
            [-s<session_ID>] [-r<request_ID>]
```

内容は次のとおりです。

user_ID	Oracle BI リポジトリにおけるユーザーの名前。これによって、表示する範囲が特定のユーザーのエントリに制限されます。これを指定しない場合は、クエリー・ログが有効になっているユーザーがすべて表示されます。
log_input_filename	既存のログ・ファイルの名前。このパラメータは必須です。
output_result_filename	ログ・ビューアの出力結果を格納するファイルの名前。指定した名前のファイルが存在する場合は、結果がそのファイルに追加されます。ファイルが存在しない場合は、新しいファイルが作成されます。これを指定しない場合、出力結果はモニター画面に送られます。
session_ID	該当するユーザー・セッションのセッション ID。Oracle BI Server では、各セッションの開始時に一意の ID が割り当てられます。これによって、表示するログ・エントリの範囲が、指定したセッション ID に制限されます。これを指定しない場合は、すべてのセッション ID が表示されます。
request_ID	個々のクエリーのリクエスト ID。Oracle BI Server では、各クエリーの開始時に一意の ID が割り当てられます。これによって、表示するログ・エントリの範囲が、指定したリクエスト ID に制限されます。これを指定しない場合は、すべてのリクエスト ID が表示されます。 **注意：** リクエスト ID は、アクティブなリクエストの中では一意ですが、セッション中に一意であるとはかぎりません。リクエスト ID は巡回形式で生成されます。リクエストが終了した場合やセッションが長すぎる場合、同じリクエスト ID が再利用されます。

ユーザー ID、セッション ID およびリクエスト ID は、Session Manager を使用して検索することもできます。詳細は、「Session Manager の使用」（232 ページ）を参照してください。

注意： Oracle BI Presentation Services の管理者は、Presentation Services Administration の「Manage Sessions」オプションを使用してクエリー・ログを表示できます。

ログ記録の解釈

いくつかのクエリー情報をログに記録し、ログ・ビューアを起動した後で、そのログを分析できます。ログは複数のセクションに分かれていますが、次の各項ではそのうちのいくつかについて説明します。レベル 1 および 2 で生成されるログのエントリは、一般的には読めば理解できる内容です。このログ・エントリには、基礎となるデータベースを担当するデータベース管理者が、クエリーに対して最適なパフォーマンスが得られるようにデータベースをチューニングする際に役立つ情報が収められています。また、Oracle BI Server を使用するアプリケーションの精度を確認するときにも、クエリー・ログが役に立ちます。

SQL Request

このセクションには、クライアント・アプリケーションから発行された SQL が記述されます。これは、同じアプリケーションまたは別のアプリケーションからクエリーを再実行する際に使用できます。

General Query Information

このセクションには、クエリーの実行場所となったリポジトリ、ビジネス・モデルおよびプレゼンテーション・カタログが記述されます。この情報を使用してクエリーの使用状況に関する統計を得ることができます。この統計は、今後のアプリケーション開発とシステム管理の優先度を設定する際に使用できます。

Database Query

ログのこのセクションは、「<data_source_name> というデータベースへのクエリー送信」という意味のエントリで始まります。data_source_name は、Oracle BI Server が接続しているデータソースの名前です。1 つ以上のデータソースに複数のデータベース・クエリーを送信できます。それぞれのクエリーについて、ログにエントリが作成されます。

Database Query セクションにはいくつかの用途があります。ここには基礎となるデータベースに送信された SQL が記録されるので、ログに記録された SQL を使用して、そのデータベースに対してクエリーを直接実行し、パフォーマンスのチューニングや結果の検証などのテストに役立てることが可能です。これによって、クエリーの対象となったテーブルを調べ、集計ナビゲーションが意図したとおりに機能しているかどうかを検証できます。基礎となるデータベースの構造がわかっていれば、有用な集計テーブルや索引の構築など、パフォーマンスの向上策に結びつく情報が得られることもあります。

Query Status

クエリーの成功に関するこのログ内のエントリには、クエリーが問題なく終了したか、失敗したかが記述されます。失敗したクエリーをログの中で検索し、失敗の原因を判断できます。たとえば、データベースに発生した停止時間のために、特定の時間帯ですべてのクエリーが失敗していることが考えられます。

使用状況トラッキングの管理

Oracle BI Server では、使用状況トラッキングの統計を蓄積して様々な用途に利用できます。たとえば、データベースの最適化、集計方針の策定、消費したリソースに応じたユーザーや部署への課金などが可能です。Oracle BI Server では、詳細なクエリー・レベルで使用状況を追跡します。

使用状況トラッキングを有効にすると、あらゆるクエリーの統計がデータベース・テーブルに挿入されるか、使用状況トラッキングのログ・ファイルに書き込まれます。直接挿入する方法を使用すると、使用状況トラッキングのデータがリレーショナル・データベース・テーブルに直接挿入されます。直接挿入する方法を使用してデータベース・テーブルに統計を書き込むことをお薦めします。

Oracle BI Server が起動すると、使用状況トラッキング・テーブルの有効な列のリストと比較して、メタデータにある列名が検証されます。varchar の長さは検証されません。これにより、次の処理が行われます。

- **列名**：テーブルにない列名がメタデータにあると、Oracle BI Server のログにエラーが書き込まれ、使用状況トラッキングが無効になります。メタデータにない列がテーブルにあると、メタデータにあるその列が使用状況トラッキング・テーブルに書き込まれます。
- **varchar の長さ**：メタデータの長さがテーブルに設定されている長さと一致しない場合は、NQServer.log ファイルにエラーが書き込まれ、使用状況トラッキングが無効になります。

この項の内容は次のとおりです。

使用状況トラッキングで情報を収集するための直接挿入の設定

使用状況トラッキングの設定でお薦めできる方法について説明します。

使用状況トラッキングに直接挿入を設定するには、この項のガイドラインに従ってください。詳細は、『Oracle Business Intelligence Infrastructure インストレーションおよび構成ガイド』を参照してください。直接挿入を設定および管理するには、次の項を参照してください。

直接挿入の有効化

NQSConfig.INI ファイルの Usage Tracking セクションで DIRECT_INSERT パラメータを設定することで、以降のロードでクエリーの統計をデータベース・テーブルに直接挿入するか、ファイルに書き込むかを指定します。直接挿入を有効にするには、DIRECT_INSERT パラメータと ENABLE パラメータを YES に設定します。

注意：直接挿入を有効にすることを強くお薦めします。

データベース・テーブルの構成

クエリーの統計情報をテーブルに挿入するには、テーブルの名前とテーブルへのアクセスに使用する接続プールを構成する必要があります。

物理テーブルの完全修飾名は、データベース名、カタログ名、スキーマ名、テーブル名という最大 4 つの要素で構成されます。各要素は二重引用符（"）で囲み、他の要素とはピリオド（.）で区切ります。物理テーブルの名前は完全修飾する必要があります。この物理テーブルの完全修飾名は、ロードされたリポジトリの物理レイヤーにあるテーブル名と一致する必要があります。次に、Oracle BI リポジトリで使用状況トラッキング・テーブルを表す物理テーブル名の例を示します。

```
PHYSICAL_TABLE_NAME = "Oracle BI Usage"."Catalog"."dbo"."S_NQ_ACCT" ;
```

この例では、Oracle BI Usage がデータベース要素を、Catalog がカタログ要素を、dbo がスキーマ要素を、S_NQ_ACCT がテーブル名をそれぞれ表しています。

接続プールの構成

完全指定の接続プール名は、データベース名と接続プール名の 2 つの部分で構成されます。各部分は二重引用符（"）で囲み、他の部分とはピリオド（.）で区切ります。この接続プールの完全修飾名は、ロードされたリポジトリの物理レイヤーにある接続プール名と一致する必要があります。次に、Oracle BI リポジトリにある接続プール名の例を示します。

```
CONNECTION_POOL = "Oracle BI Usage"."Connection Pool" ;
```

この例では、Oracle BI Usage がデータベース要素を、Connection Pool が接続プール名本体をそれぞれ表しています。

Usage Tracking を正常に挿入するには、バックエンド・データベースに対する書込み権限が設定されたユーザー ID を使用して、接続プールを構成する必要があります。

注意：国際的なデータをサポートするタイプの接続を使用することをお薦めします。

バッファ・サイズ構成パラメータ

BUFFER_SIZE 構成パラメータは、挿入文のバッファ処理用に Oracle BI Server で割り当てるメモリー量を指定します。このようなバッファを使用することで、Oracle BI Server では 1 つのトランザクションの一部として複数の挿入文を送信できるので、Usage Tracking 挿入のスループットが向上します。また、通常のクエリー・リクエストでは Usage Tracking の挿入を待つ必要がないので、クエリーに対する平均レスポンス時間も短縮されます。この値は、サーバー・マシンで利用できるメモリー量とその使用状況に基づいて調整できます。

バッファ時間制限構成パラメータ

BUFFER_TIME_LIMIT_SECONDS 構成パラメータは、Usage Tracking サブシステムから挿入文が発行されるまで、その挿入文をバッファに保持しておく最長時間を指定します。この時間制限を設定することで、処理が長時間にわたって発生しない間でも、挿入文が Oracle BI Server から適切な時点で発行されるようになります。

挿入スレッド数構成パラメータ

NUM_INSERT_THREADS 構成パラメータは、バッファから挿入文を削除して Usage Tracking データベースに発行するスレッドの数を指定します。読取りと挿入でそれぞれ独立した接続プールを使用すると仮定すると、挿入スレッドの数は、その接続プールの「Maximum Connections」の設定と普通は等しくなります。

トランザクション当たり最大挿入数構成パラメータ

MAX_INSERTS_PER_TRANSACTION 構成パラメータは、1 つのトランザクションの一部として Usage Tracking サブシステムから発行する挿入文の最大数を指定します。この数値が大きいほど、Usage Tracking 挿入で可能なスループットも大きくなります。ただし、デッドロックが原因でトランザクションが失敗する可能性も高くなります。BUFFER_TIME_LIMIT_SECONDS に小さい値を設定すると、トランザクション当たりの挿入数が制限されることがあります。

使用状況トラッキングで情報を収集するためのログ・ファイルの設定

使用状況トラッキングを設定するもう 1 つの方法について説明します。使用状況トラッキングでの情報収集では直接挿入の使用をお薦めします。詳細は、「使用状況トラッキングで情報を収集するための直接挿入の設定」（226 ページ）を参照してください。

この項の内容は次のとおりです。

出力場所の選択

NQSConfig.INI ファイルの Usage Tracking セクションにある STORAGE_DIRECTORY パラメータは、使用状況トラッキング・ログ・ファイルの場所を指定します。使用状況トラッキングが有効になっていても、記憶域フォルダを指定しないと、ファイルはソフトウェアのインストール・フォルダ（¥OracleBI）の下の Log フォルダに書き込まれます。

現在のファイルは定期的にディスクに書き込まれ、新しいファイルが作成されます。CHECKPOINT_INTERVAL_MINUTES パラメータは、使用状況トラッキングのデータをディスクにフラッシュする頻度を制御します。また、FILE_ROLLOVER_INTERVAL_MINUTES パラメータは、現在の使用状況トラッキング・ログ・ファイルを閉じて新しいファイルを作成する頻度を制御します。

使用状況トラッキングが有効になっていると、あらゆるクエリーが使用状況トラッキング・ログ・ファイルに記録されます。これには、巨大な空き容量を持つ記憶域が必要になります。たとえば、データ出力量が平均 300 バイトのクエリーが 1 秒当たり 10 回発生する状況で、そのクエリーを 8 時間にわたって記録するとします。この場合、1 日当たりで記憶域に書き込まれる使用状況トラッキング・データの量は約 83MB になります。この例を年中無休 24 時間運用として考えると、1 日当たりに必要な記憶域は約 25GB にもなります。

Oracle BI Server では、指定した場所に置くことができる使用状況トラッキング・ログ・ファイルのサイズや数には制限がありません。十分なディスク容量があることを確認し、古い使用状況トラッキング・ファイルを削除したりアーカイブする処理は、ユーザー側で実施する必要があります。

注意：記憶領域が不足すると、使用状況トラッキング・データが失われることがあります。Oracle BI Server から使用状況トラッキング出力ファイルへのアクセスでエラーが発生すると、使用状況トラッキング統計の収集はただちに中止され、エラー・メッセージが NQServer.log に発行されます。Windows では、Windows のイベント・ログにもエラー・メッセージが発行されます。追加の記憶領域が利用可能になっても、使用状況トラッキング統計の収集を再開するにはサーバーを再起動する必要があります。

ファイルのネーミング規則

使用状況トラッキング・ログ・ファイルには、NQAcct.yyyymmdd.hhmmss.log というネーミング・スキーマでファイル名が付けられます。yyyy、mm、dd、hh、mm および ss は、ファイル作成時のタイムスタンプの年、月、日、時、分および秒です。たとえば、2003 年 2 月 12 日午前 7 時 15 分 00 秒にサーバーで作成された使用状況トラッキング・ログ・ファイルの名前は、NQAcct.20030212.071500.log になります。指定したロールオーバー間隔が経過すると、このファイルはディスクにフラッシュされて閉じ、現在の日付とタイムスタンプで新しいファイルが作成されます。

出力ファイル・フォーマット

使用状況トラッキング・ログ・ファイルは、セミコロン区切り（;）フォーマットのテキスト・ファイルです（列の区切り記号としてセミコロンを使用します。これは、カンマが論理 SQL テキストで使用されているためです）。各データ行の区切りには改行を使用します。

このスキーマの説明は、230 ページの表 24 に記載されています。各列の内容の詳細は、「使用状況トラッキング・データの説明」（417 ページ）を参照してください。

表 24. 使用状況トラッキング出力ファイルのフォーマット

列番号	列名	データ型	データの最大サイズ	NULL 値可能
1	User name	Varchar	128	不可
2	Repository name	Varchar	128	不可
3	Subject area name	Varchar	128	不可
4	Node ID	Varchar	15	不可
5	Start timestamp	Char（タイムスタンプ）	19	不可
6	Start date	Char（yyyy-mm-dd）	10	不可
7	Start hourMin	Char（hh:mm）	5	不可
8	End timestamp	Char（タイムスタンプ）	19	不可
9	End date	Char（yyyy-mm-dd）	10	不可
10	End hourMin	Char（hh:mm）	5	不可
11	Query Text	Varchar	1024	不可
12	Success indicator	Integer（次の「注意」を参照）	4	不可
13	Row count	Integer（次の「注意」を参照）	4	可
14	Total time (secs)	Integer（次の「注意」を参照）	4	可
15	Compilation time (secs)	Integer（次の「注意」を参照）	4	可
16	Number of database queries	Integer（次の「注意」を参照）	4	可
17	Cumulative db time (secs)	Integer（次の「注意」を参照）	4	可
18	Cumulative db rows	Integer（次の「注意」を参照）	4	可
19	Cache indicator	Char	1	不可
20	Query source	Varchar	30	不可
21	Presentation Catalog path	Varchar	250	不可
22	Interactive Dashboard name	Varchar	150	可

注意：この出力ファイルに記述されるデータはすべて文字フォーマットです。列 12 から列 18 までのデータは、整数のテキスト表現として出力されます。したがって、これらのデータの動作は整数よりも Varchar(10) に近いものになります。たとえば、行カウントが 100 万の場合、出力ファイルの列 13（Row count）には 1000000 と記述されます。このデータは、4 バイトの内部整数値を表していますが、出力ファイルでは 7 バイトのデータで構成されます。

- 列 12 の Success indicator の値が 0 であれば、クエリーは正常に処理されています。この値が 0 以外の場合、クエリーは失敗です。現在定義されている失敗インジケータは次のとおりです。
  - 1: タイムアウト
  - 2: 行制限違反
  - 3: 不明なエラー

以降の列は、クエリーが成功したことが Success indicator で示されている場合（値が 0）にのみ有効です。

- Start timestamp 列と End timestamp 列は、それぞれ論理クエリーの開始日時と終了日時を示します。それぞれの値は、SQL-92 タイムスタンプを表現する 19 バイトの文字データです。そのフォーマットは、*yyyy-mm-dd-hh:mm:ss* です。これに関連する Start date と End date は、それぞれ開始と終了のタイムスタンプの日付部分を *yyyy-mm-dd* のフォーマットで記述した列です。最後に、これに関連する Start hourMin と End hourMin は、それぞれ開始と終了のタイムスタンプの時刻部分を *hh:mm* のフォーマットで記述した列です。

使用状況トラッキング・データには一意性を保証するキーはありませんが、通常は User name、Node ID、Start timestamp および Query Text を組み合せることで十分な一意性が得られます。

使用状況トラッキング・ログ・ファイルからデータを抽出して、適切なフォーマットのリレーショナル・データベース・テーブルにロードするためのスクリプト例の詳細は、「Oracle BI Server の使用状況トラッキング・データの説明とログ・ファイル・メソッドの使用」（415 ページ）を参照してください。

パフォーマンスに関する考慮事項

使用状況トラッキングが有効になっていると、Oracle BI Server では、あらゆるクエリーについて使用状況トラッキング・データが収集されます。ただし、このデータは、「チェックポイント」という、ユーザー指定の間隔でのみディスクに書き込まれます。デフォルト設定のチェックポイントは 5 分間隔です。

この値は NQSConfig.INI ファイルで変更できますが（『Oracle Business Intelligence Infrastructure インストレーションおよび構成ガイド』を参照）、間隔を短くするとオーバーヘッドが増加し、極端に短い間隔を設定すると、サーバーのパフォーマンスが低下することもあります。また、大きな値を設定すると、Oracle BI Server の異常停止などの予期しない事態が発生したときに、大量の使用状況トラッキング・データが失われることになります。

Oracle BI Server では、定期的に使用状況トラッキング・ログ・ファイルのロールオーバーが実行されます。ロールオーバーでは、現在の使用状況トラッキング・ログ・ファイルを閉じ、新しく作成したログ・ファイルを開いて、以降のデータをそのファイルに書き込みます。ロールオーバーの実行頻度を「ロールオーバー間隔」と呼びます。デフォルトのロールオーバー間隔は 240 分（4 時間間隔）です。

閉じた使用状況トラッキング・ログ・ファイルは分析に使用できます。ロールオーバー間隔を短くすると、分析に使用できる使用状況トラッキング・ログ・ファイルを短時間で入手できますが、オーバーヘッドが増加するという欠点も発生します。

チェックポイント間隔をロールオーバー間隔以上に設定すると、明示的に発生するのはロールオーバーのみとなり、チェックポイントは、古い使用状況トラッキング・ログ・ファイルが閉じられるときにのみ、暗黙的に発生します。

サーバー・セッションの管理

Session Manager をオンライン・モードで使用してアクティビティを監視します。Session Manager では、セッションにログインしているすべてのユーザー、各ユーザーの現在のすべてのクエリー・リクエスト、および選択したセッションの変数とその値が表示されます。また、Oracle BI の管理者であれば Session Manager を使用して、任意のユーザーの接続を切断し、任意のクエリー・リクエストを中断できます。

Session Manager のデータがリフレッシュされる頻度は、システムに発生するアクティビティの量によって変化します。明示的に表示をリフレッシュするには、「Refresh」をクリックします。

Session Manager の使用

Session Manager は、上下 2 つのペインで構成されます。

- 上部のウィンドウである「Session」ウィンドウには、Oracle BI Server に現在ログインしているユーザーが表示されます。更新速度を制御するには、「Update Speed」ドロップダウン リストで「Normal」、「High」または「Low」を選択します。「Pause」を選択すると、表示がリフレッシュされなくなります。
- 下部のウィンドウには 2 つのタブがあります。
  - 「Request」タブには、「Session」ウィンドウで選択されたユーザーのアクティブなクエリー・リクエストが表示されます。
  - 「Variable」タブには、選択したセッションの変数とその値が表示されます。列のヘッダーをクリックすると、その列のデータをソートできます。

    注意：変数に関する情報を返すのは、バージョン 7.7 の Oracle BI Server のみです。これより古いバージョンのサーバーに Administration Tool がオンラインで接続している場合、Session Manager のダイアログ・ボックスには「Request」タブのみが表示されます。

Session Manager の各ウィンドウに表示される列については、232 ページの表 25 および 233 ページの表 26 で説明します。

表 25. 「Session」ウィンドウのフィールド

列名	説明
Client Type	サーバーに接続しているクライアントのタイプ。
Last Active Time	セッションにおける最後のアクティビティのタイムスタンプ。
Logon Time	セッションが最初に Oracle BI Server に接続した日時を示すタイムスタンプ。
Repository	セッションの接続先となるリポジトリの論理名。
Session ID	セッションが開始したときに、Oracle BI Server が各セッションに割り当てた一意の内部識別子。
User	接続しているユーザーの名前。

表 26. 「Request」タブにあるフィールドの一部

列名	説明
Last Active Time	クエリーにおける最後のアクティビティのタイムスタンプ。
Request ID	クエリーが開始したときに、Oracle BI Server が各クエリーに割り当てた一意の内部識別子。
Session ID	セッションが開始したときに、Oracle BI Server が各セッションに割り当てた一意の内部識別子。
Start Time	個々のクエリー・リクエストの日時。

セッションの変数を表示するには

1. Administration Tool で、リポジトリをオンライン・モードで開き、「Manage」→「Sessions」を選択します。
2. セッションを選択して「Variable」タブをクリックします。

   変数の詳細は、「Oracle BI リポジトリの変数の使用」（291 ページ）を参照してください。
3. 表示をリフレッシュするには、「Refresh」をクリックします。
4. Session Manager を終了するには、「閉じる」をクリックします。

ユーザーをセッションから切断するには

1. Administration Tool で、リポジトリをオンライン・モードで開き、「Manage」→「Sessions」を選択します。
2. Session Manager の上部ウィンドウでユーザーを選択します。
3. 「Disconnect」をクリックします。

   ユーザーのセッションは、Oracle BI の管理者がそのセッションを終了したことを通知するメッセージを受け取ります。現在実行中のクエリーはただちにすべて終了し、基礎となるデータベースに対する未処理のクエリーはすべて取り消されます。
4. Session Manager を終了するには、「閉じる」をクリックします。

アクティブなクエリーを中断するには

1. Administration Tool で、リポジトリをオンライン・モードで開き、「Manage」→「Sessions」を選択します。
2. Session Manager の上部ウィンドウで、目的のクエリーを開始したユーザー・セッションを選択します。

   ユーザーが強調表示された後、そのユーザーが発行したアクティブなクエリー・リクエストがすべて下部のウィンドウに表示されます。
3. 中断するリクエストを選択します。

4 「Kill Request」をクリックすると、強調表示されているリクエストが終了します。

ユーザーは、Oracle BI の管理者がそのクエリーを終了したことを通知するメッセージを受け取ります。そのクエリーはただちに終了し、基礎となるデータベースに対する未処理のクエリーはすべて取り消されます。

他のリクエストをすべて中断するまでこの手順を繰り返します。

5 Session Manager を終了するには、「閉じる」をクリックします。

サーバーの構成とチューニング

どのような意思決定支援システムでもパフォーマンスはきわめて重要な考慮事項ですが、Web 経由でクエリーを実行できるシステムでは特に重要になります。この項では、Oracle BI Server でのクエリーのパフォーマンスを向上するうえで重要ないくつかの考慮事項について説明します。ここで記述しているのは概要です。詳細は、『Oracle Business Intelligence Infrastructure インストレーションおよび構成ガイド』を参照してください。

NQSConfig.INI ファイルのパラメータ

NQSConfig.INI ファイルには、Oracle BI Server の構成とチューニングのためのパラメータが記述されています。これらのパラメータでは、一時記憶域のディスク領域構成、ソート・メモリーのバッファ・サイズ設定、キャッシュ・メモリー・バッファの設定、仮想テーブルのページ・サイズ設定など、使用しているハードウェアの性能を最大限に引き出すための構成を設定します。

NQSConfig.INI ファイルのパラメータの詳細は、『Oracle Business Intelligence Infrastructure インストレーションおよび構成ガイド』を参照してください。

集計テーブル

クエリーのパフォーマンスを向上させるには、集計テーブルを使用する必要があります。集計テーブルには、データの事前計算済の要約が収められています。数千行に及ぶ詳細から回答を再計算するよりも、集計テーブルから回答を取得するほうが、処理がはるかに速く完了します。リポジトリで集計テーブルを適切に指定しておけば、Oracle BI Server では自動的にその集計テーブルが使用されます。

リポジトリでの集計ナビゲーション設定の詳細と例は、「Oracle BI リポジトリでの集計ナビゲーション用のフラグメンテーション・コンテンツの設定」（205 ページ）を参照してください。

クエリー・キャッシュ

Oracle BI Server でクエリー・キャッシュを有効にすると、クエリーの結果を格納しておき、以降のクエリーで再利用できます。キャッシュを利用することで、ユーザーから見たシステムのパフォーマンスが大幅に向上します。クエリー・キャッシュの概念と設定の詳細は、第 11 章「Oracle BI Server のクエリー・キャッシュ」を参照してください。

基礎となるデータベースのチューニングと索引作成

クエリーは Oracle BI Server からデータベースに送信されます。クエリーに対する回答が適切な時間で得られるようにするには、基礎となるデータベースを正しく構成、チューニングおよび索引作成する必要があります。データベースのチューニングが必要な、問題の領域を特定するために、基礎となるデータベースの管理者との共同作業が必要な場合もあります。

チューニングでの考慮事項は、データベース製品によって様々です。基礎となるデータベースからの回答に時間のかかるクエリーがある場合は、クエリー・ログからそのクエリーの SQL を取り込んでデータベース管理者に提出し、分析を依頼します。使用しているシステムでクエリー・ロギングを構成する方法の詳細は、「クエリー・ログの管理」（221 ページ）を参照してください。

11 Oracle BI Server のクエリー・キャッシュ

意思決定支援のクエリーでは、データベースで大量の処理が必要になることがあります。Oracle BI Server では、クエリーの結果をキャッシュ・ファイルに保存しておき、類似のクエリーがリクエストされたときにその結果を再利用できます。キャッシュを使用すると、クエリーに対するデータベースでの処理が 1 回のみで済み、クエリーが実行されるたびに同じ処理を繰り返す必要がなくなります。

この項では、クエリー・キャッシュとそれが Oracle BI Server でどのように実装されるかについて説明します。

注意： Delivers を使用して Oracle BI Server キャッシュを生成する方法の詳細は、『Oracle Business Intelligence Presentation Services 管理ガイド』を参照してください。

この章の内容は次のとおりです。

Oracle BI Server クエリー・キャッシュについて

Oracle BI の管理者は、クエリー結果セットをローカルで保存するディスクベースのキャッシュ（クエリー・キャッシュ）を維持するように Oracle BI Server を構成できます。クエリー・キャッシュを使用すると、（Oracle や DB2 などの）バックエンド・データベースにアクセスしなくても、以降の数多くのクエリー・リクエストに Oracle BI Server で対応できるようになります。通信面のコストが軽減されることで、クエリーのレスポンス時間を大幅に短縮できます。

バックエンド・データベース側で更新が発生すると、クエリー・キャッシュのエントリが最新でなくなる可能性があります。したがって、Oracle BI の管理者は、次のいずれかの方法でクエリー・キャッシュのエントリを定期的に削除する必要があります。

- **手動処理**：Administration Tool の「Manage」メニューで、「Cache」を選択して Cache Manager を起動します。Cache Manager では、消去するキャッシュ・エントリとそれを消去する時期をきわめて柔軟に選択できますが、人手による直接操作が必要になります。詳細は、「Cache Manager の使用」（263 ページ）を参照してください。
- **自動処理**：Administration Tool では、システムのキャッシュを無効にして、特定の物理テーブルのキャッシュ属性を設定し、Oracle BI のイベント・テーブルを使用してキャッシュを自動的に消去できます。キャッシュの管理方法の詳細は、「キャッシュの監視と管理」（242 ページ）を参照してください。
- **プログラムによる処理**：Oracle BI Server には、キャッシュのエントリをプログラムで消去するための ODBC 拡張関数が用意されています。これらの関数は、イベント・テーブルの自動化によって Cache Manager が持つ選択機能と処理タイミングの柔軟性を実現します。目的に合ったタイミングでこれらの関数をコールする専用のスクリプトを記述できます。詳細は、「ODBC 手順を使用したキャッシュの消去と保持」（244 ページ）を参照してください。

クエリー・キャッシュを制御するパラメータは、『Oracle Business Intelligence Infrastructure インストレーションおよび構成ガイド』で説明している NQSConfig.INI ファイルに記述されています。

注意： Delivers を使用して Oracle BI Server キャッシュを生成する方法の詳細は、『Oracle Business Intelligence Presentation Services 管理ガイド』を参照してください。

キャッシュのメリット

クエリーを短時間で処理するには、大量の処理を省略し、事前計算した回答を使用することが最良の方法です。

事前計算した回答の一例が集計テーブルです。集計テーブルには、特定の集計レベルで事前計算した結果が収められています。たとえば、データベースの詳細を日次レベルの細密度で管理している場合、集計テーブルには製品ごとの売上結果が月別に格納されます。この集計テーブルを作成するには、プロセス（普通はクエリー）で結果を計算し、それをデータベースにテーブルで格納します。

Oracle BI Server のクエリー・キャッシュでは、クエリーを事前計算した結果がローカル・キャッシュに格納されます。別のクエリーでこの結果を使用できる場合、そのクエリーに対するデータベース側での処理はすべて省略されます。その結果、クエリーの平均レスポンス時間を大幅に短縮できます。

パフォーマンスの向上に加え、ローカル・キャッシュでクエリーに回答できることで、データベース・サーバーでのネットワーク・リソースと処理時間も節約されます。ネットワーク・リソースを節約できるのは、ネットワークを介して中間結果を Oracle BI Server に送信する必要がないためです。データベースでクエリーを実行しないことにより、データベースでは他の処理を実行できるようになります。チャージ・バック制を採用しているデータベースでは、予算面での支払額も削減できます。

キャッシュを使用してクエリーに回答する方法のメリットとして、Oracle BI Server での処理時間を削減できるという点もあげられます。特に、複数のデータベースからクエリーの結果を取得する場合に大きなメリットとなります。クエリーの内容によっては、相当量の結合処理とソート処理がサーバーで発生することがあります。クエリーが計算済であればこのような処理は発生せず、サーバーのリソースを他のタスクに割り当てることができます。

まとめると、クエリー・キャッシュには次のメリットがあります。

- クエリー・パフォーマンスの大幅な向上
- ネットワーク・トラフィックの低減
- データベース処理とチャージ・バックの削減
- Oracle BI Server の処理オーバーヘッドの減少

ユーザー ID でのキャッシュ・エントリの初期化

ユーザー ID でキャッシュ・エントリを初期化するには、ログイン・プロパティのセッション変数 VALUEOF(NQ_SESSION.PASSWORD) および VALUEOF(NQ_SESSION.USER) を使用し、共有ログインについて Connection Pool を設定する必要があります。共有ログインを無効にし、ユーザー固有のデータベース・ログインを使用すると、キャッシュが共有されます。

セキュリティの詳細は、第 15 章「Oracle BI におけるセキュリティ」を参照してください。

キャッシュのデメリット

クエリー・キャッシュには明白なメリットが数多くありますが、次のようなデメリットもあります。

- キャッシュ用のディスク領域が必要
- キャッシュ管理の手間が発生
- キャッシュ結果が最新ではない可能性が発生
- サーバー・マシンの CPU 使用率とディスク I/O がわずかに増加

キャッシュを適切に管理することで、これらのデメリットを上回るメリットが得られます。

ディスク領域

クエリー・キャッシュでは専用のディスク領域が必要です。このディスク領域の大きさは、クエリーの量、クエリー結果セットのサイズ、およびユーザー側でキャッシュに割り当てるディスク領域の大きさによって異なります。高いパフォーマンスを得るために、高いパフォーマンスと信頼性を持つ、キャッシュ専用のディスク・システムでキャッシュを使用する必要があります。

管理作業

キャッシュの使用に伴う管理作業がいくつかあります。各物理テーブルのデータ更新頻度を確認し、そのテーブルのキャッシュ保持時間を適切に設定する必要があります。更新頻度が変動する場合は、変更の発生時期を把握し、必要に応じてキャッシュを手動で消去する必要があります。キャッシュ・イベントのポーリング・テーブルを作成し、データベースに変更が発生したときにそのポーリング・テーブルを更新するようにアプリケーションを変更することもできます。これにより、システムはイベントドリブンとなります。

Oracle BI Server には、キャッシュのエントリをプログラムで消去するための ODBC 拡張関数も用意されています。適切なタイミングでこれらの関数をコールする専用のスクリプトを記述できます。

キャッシュの最新性維持

基礎となるデータベースのデータに変更があったときにキャッシュの該当エントリを消去しておかないと、クエリーに対して最新ではない結果が返される可能性があります。得られた結果が妥当なものかどうかを評価する必要があります。古いデータがいくつかキャッシュに存在しても問題とならない場合もあります。どのレベルの古さのデータまで許容できるか判断し、そのレベルを反映したルールを設定（および運用）する必要があります。

たとえば、巨大な複合企業の企業データを分析するアプリケーションがあり、その企業の様々な部署の年次集計を実行するとします。新しいデータがあってもその影響を受けるのは翌年分の集計なので、その新しいデータでこのクエリーが影響を受けることは実質的にはありません。この場合は、キャッシュを消去するかどうかという判断は、キャッシュのエントリをそのまま残すことが妥当であるという結論になります。

ただし、1 日に 3 回の頻度で更新されるデータベースに対して、本日のアクティビティに関するクエリーを実行する場合は事情が異なります。この場合は、キャッシュを頻繁に消去するか、キャッシュを使用しないようにする必要があります。

また、（週に 1 回など）一定の間隔でデータ・マートを新規に再構築するというシナリオも考えられます。この例では、データ・マートを再構築するプロセスの一環としてキャッシュ全体を消去すると、古いデータがキャッシュに残ることはなくなります。

どのような状況でも、最新ではない情報がユーザーに返される事態がどの程度許容できるかを評価する必要があります。

CPU 使用率とディスク I/O

大半の状況では問題になりませんが、クエリー・キャッシュではわずかな CPU 時間が必要となり、ディスク I/O が増加します。多くの場合、CPU 使用率の増加は無視できますが、ディスク I/O は、特に大量のデータセットを返すクエリーに影響することがあります。

クエリー・キャッシュのアーキテクチャ

クエリー・キャッシュは、キャッシュ記憶領域、キャッシュ・メタデータ、およびクエリーのコンパイルでのキャッシュ検出で構成されます。

キャッシュ・メタデータへのアクセスはきわめて高速です。メタデータにキャッシュ・ヒットがあれば、クエリー処理の大部分は不要となり、結果がただちにユーザーに返されます。新しい結果をキャッシュに追加するプロセスは、ユーザーに返される結果から独立しています。実行中のクエリーに対する唯一の影響は、キャッシュした結果を書き込むプロセスで消費されるリソースです。

クエリー・キャッシュの構成

デフォルトでは、クエリー・キャッシュは無効になっています。キャッシュを有効にするには、キャッシュ記憶域を構成し、古くなったエントリをフラッシュする方針を決定する必要があります。この項では、クエリー・キャッシュについて Oracle BI Server を構成するうえで必要な作業について説明します。

クエリー・キャッシュを制御するパラメータは、『Oracle Business Intelligence Infrastructure インストレーションおよび構成ガイド』で説明している NQSConfig.INI ファイルに記述されています。

キャッシュ記憶域の構成

NQSConfig.INI ファイルで、キャッシュ記憶域について次の項目を設定する必要があります。

- キャッシュ・ファイルを格納するディレクトリ
- キャッシュ・メタデータを格納するファイル

キャッシュ・データ記憶域のディレクトリ

NQSConfig.INI ファイルの CACHE セクションの DATA_STORAGE_PATHS パラメータで、クエリー・キャッシュ記憶域のディレクトリを 1 つ以上指定します。キャッシュしたクエリー結果をこのディレクトリに格納し、キャッシュ・ヒットが発生したときにこのディレクトリにアクセスします。詳細は、「キャッシュ・ヒット」（247 ページ）を参照してください。

キャッシュ記憶域のディレクトリは、高いパフォーマンスを持つストレージ・デバイスに置く必要があります。専用のキャッシュ記憶域を使用できれば理想的です。キャッシュ記憶域のディレクトリの空き領域が少なくなってくると、新しいエントリを格納する領域を確保するために、最近の使用頻度が最も低い（LRU）エントリが破棄されます。MAX_CACHE_ENTRIES パラメータで、クエリー・キャッシュに格納できるキャッシュ・エントリの最大数を指定します。キャッシュに割り当てる領域が大きいほど、クエリーの結果を得るときに、基礎となるデータベースへのアクセスが必要となる頻度は低くなります。

この構成パラメータの詳細は、『Oracle Business Intelligence Infrastructure インストレーションおよび構成ガイド』を参照してください。

最大キャッシュ・エントリ数

NQSConfig.INI ファイルのパラメータ MAX_ROWS_PER_CACHE_ENTRY および MAX_CACHE_ENTRIES で、任意のキャッシュ・エントリの最大行数および最大キャッシュ・エントリ数をそれぞれ制御できます。大量の行を返す処理量の大きいクエリーでキャッシュ領域を消費しないようにするには、行数の制限が効果的です。キャッシュ・エントリの総数を制限する場合は、キャッシュ記憶域を管理する別のパラメータを使用できます。MAX_ROWS_PER_CACHE_ENTRY パラメータで指定した値を超える数の行を返すクエリーはキャッシュされません。

これらのパラメータの構文については、『Oracle Business Intelligence Infrastructure インストレーションおよび構成ガイド』を参照してください。

集計

以前に実行されたクエリーからキャッシュ・ヒットを取得したクエリーの場合、普通はその新しいクエリーはキャッシュに追加されません。POPULATE_AGGREGATE_ROLLUP_HITS パラメータを指定すると、キャッシュ・ヒットが発生したとき、以前に実行したクエリーから集計がロールアップされ、このデフォルトの設定が無視されます。

クエリー・キャッシュの有効化

「キャッシュの監視と管理」（242 ページ）で説明したように、キャッシュ記憶域を構成して 1 つ以上のキャッシュ管理方針を決定すれば、クエリー・キャッシュを有効にできます。

クエリー・キャッシュを有効にするには

1. NQSConfig.INI ファイルの CACHE セクションで、ENABLE パラメータを YES に設定します。
2. Oracle BI Server を再起動します。

クエリー・キャッシュの無効化

この項では、クエリー・キャッシュを無効にする方法について説明します。

クエリー・キャッシュを無効にするには

1. NQSConfig.INI ファイルの CACHE セクションで、ENABLE パラメータを NO に設定します。
2. Oracle BI Server を再起動します。

キャッシュの監視と管理

基礎となるデータベースに発生する変更を管理し、キャッシュ・エントリを監視するには、キャッシュ管理方針を確立する必要があります。不要なキャッシュ・エントリを監視、特定および削除するプロセスのほか、基礎となるテーブルの中で、キャッシュ・エントリの元となったデータが変更されたときに該当のキャッシュ・エントリを無効にするプロセスが必要です。

キャッシュ管理方針の選択

キャッシュ管理方針は、基礎となるデータベースにあるデータの変動性とその変動の原因となるデータ変更の予測可能性に基づいて選択します。また、キャッシュを構成するクエリーの数とタイプ、およびこれらのクエリーが受け取る使用状況によっても、選択する方針は異なってきます。この項では、キャッシュ管理の様々な手法の概要について説明します。

システムのキャッシュの無効化

NQSConfig.INI ファイルで ENABLE パラメータを NO に設定して Oracle BI Server を再起動することで、システム全体でキャッシュを無効にすることができます。キャッシュを無効にすると、新しいキャッシュ・エントリは作成されなくなり、既存のキャッシュは新しいクエリーで使用できなくなります。キャッシュを無効にすると、キャッシュに格納済のエントリをすべて保持したまま、後でそのキャッシュを有効にすることができます。

キャッシュ・エントリが古くなっている可能性が高い状況で、そのようなエントリまたはキャッシュ全体を消去する前に、実際に古い状態になっているかどうかを検証する必要がある場合は、キャッシュを一時的に無効にする方法が効果的です。キャッシュに格納されているデータがまだ使用可能であることが判明した場合や、問題のあるエントリを安全に消去した後は、キャッシュを有効にしても問題ありません。必要に応じて、キャッシュ全体、またはビジネス・モデル全体に関連するキャッシュを消去した後、キャッシュを再び有効にします。

指定した物理テーブルのキャッシュおよびキャッシュ保持時間

物理テーブルごとにキャッシュ可能属性を設定できます。これにより、今後発生するクエリーに答えるために、そのテーブルに対するクエリーをキャッシュに追加するかどうかを指定できます。あるテーブルに対するクエリーを有効にすると、そのテーブルを対象としたクエリーはすべてキャッシュに追加されます。デフォルトではすべてのテーブルがキャッシュ可能になっていますが、キャッシュ保持時間の設定を使用しないと、キャッシュの対象とするにはふさわしくないテーブルが存在することがあります。たとえば、株式相場のデータを格納したテーブルは分単位で更新されます。このようなテーブルでは、そのエントリが 59 秒ごとに消去されるようにキャッシュ保持時間を設定できます。

「Cache persistence time」フィールドを使用して、このテーブルのエントリをクエリー・キャッシュに保持する時間を指定することもできます。これは、頻繁に更新されるデータソースに便利です。

特定の物理テーブルのキャッシュ属性を設定するには

1. 「Physical」レイヤーで、目的の物理テーブルをダブルクリックします。
2. 「Physical Table properties」ダイアログ・ボックスの「General」タブで、次のいずれかを選択します。
   - キャッシュを有効にするには、「Make table cachable」チェック・ボックスを選択します。
   - テーブルがキャッシュされないようにするには、「Make table cachable」チェック・ボックスの選択を解除します。
3. 有効期間（最大保持時間）を設定するには、次の手順を実行します。
   a. 「Cache persistence time」ドロップダウン・リストで、値を選択します。

      「Infinite」を選択した場合、または具体的な値を選択していない場合、「Cache persistence time」フィールドは使用できません。

   b. 「Cache persistence time」フィールドに値を入力します。
4. 「OK」をクリックします。

Oracle BI Server イベント・ポーリング・テーブルの構成

Oracle BI Server のイベント・ポーリング・テーブルには、基礎となるデータベースで発生した更新に関する情報が格納されます。（データをデータ・マートにロードするような）アプリケーションは、データベース・テーブルが更新されるたびにイベント・ポーリング・テーブルに行を追加するように構成できます。Oracle BI Server では、設定した間隔でこのテーブルがポーリングされ、更新されたテーブルに対応するキャッシュ・エントリはすべて無効化されます。イベント・ポーリング・テーブルのみでキャッシュを管理できるほか、他のキャッシュ管理スキームとイベント・ポーリング・テーブルの組合せで管理することもできます。キャッシュ・エントリと消去タイミングの選択の面で、イベント・ポーリング・テーブルの柔軟性は限られています。イベント・ポーリング・テーブルの詳細は、「物理データベースに対するイベント・ポーリング・テーブルの設定」（257 ページ）を参照してください。

ODBC 手順を使用したキャッシュの消去と保持

Oracle BI の管理者がキャッシュ・エントリの消去に使用できるように、Oracle BI Server には ODBC 拡張関数が用意されています。

これらの関数の中には、特に抽出、変換およびロード（ETL）の作業への埋込みに便利なものがあります。たとえば、夜間の ETL を実施した後では、Oracle BI Server のキャッシュをすべて消去できます。ファクト・テーブルのみが変更されている場合は、そのテーブルに関連するキャッシュのみを消去できます。特定のデータベースに関連するキャッシュ・エントリの消去が必要なこともあります。

注意：キャッシュを消去する権限を持っているのは、Oracle BI の管理者のみです。したがって、これらの ODBC 拡張関数をコールするスクリプトは、Oracle BI の管理者のログオン ID で実行する必要があります。

次の ODBC 関数は、ODBC 接続で指定したリポジトリに関連付けたキャッシュ・エントリに影響します。

- **SAPurgeCacheByQuery**: 指定のクエリーに完全一致するキャッシュ・エントリを消去します。たとえば、次のクエリーを使用すると、収入が 100,000 ドルを超えている従業員の名前を取得したクエリー・キャッシュ・エントリが得られます。

  ```
  select lastname, firstname from employee where salary > 100000;
  ```

  次のコールによって、このクエリーに関連するキャッシュ・エントリが消去されます。

  ```
  Call SAPurgeCacheByQuery('select lastname, firstname from employee where salary > 100000' );
  ```

- **SAPurgeCacheByTable**: クライアントが接続するリポジトリについて、指定した物理テーブル名（完全修飾）に関連したキャッシュ・エントリをすべて消去します。

  この関数は、完全修飾物理テーブル名の 4 つの要素（データベース、カタログ、スキーマおよびテーブル名本体）を表す最大 4 つのパラメータをとります。たとえば、完全修飾名が DBName.CatName.SchName.TabName というテーブルがあるとします。Oracle BI のリポジトリの物理レイヤーでこのテーブルに関連付けられたキャッシュ・エントリを消去するには、スクリプトで次のコールを実行します。

  ```
  Call SAPurgeCacheByTable( 'DBName', 'CatName', 'SchName', 'TabName' );
  ```

  注意：Oracle BI Server では、この関数にワイルド・カードは使用できません。また、DBName と TabName に null は指定できません。いずれか 1 つに null を指定しても、エラー・メッセージを受け取ります。

- **SAPurgeAllCache**: すべてのキャッシュ・エントリを消去します。次に、このコールの例を示します。

  ```
  Call SAPurgeAllCache();
  ```

- **SAPurgeCacheByDatabase**: 特定の物理データベース名に関連付けられたキャッシュ・エントリをすべて消去します。キャッシュを消去する ODBC プロシージャであれば、どれであってもコールの結果としてレコードが返されます。この関数は、物理データベース名を表すパラメータを 1 つとりますが、このパラメータに null は指定できません。次に、このコールの構文を示します。

  ```
  Call SAPurgeCacheByDatabase( 'DBName' );
  ```

Presentation Server キャッシュの共有について

クエリーを実行するためにユーザーが Intelligence Dashboard にアクセスすると、そのクエリーの結果は Oracle BI Presentation Services によってキャッシュされます。Oracle BI Presentation Services では、リクエスト・キーと論理 SQL 文字列を使用して、以降に実行するクエリーで、キャッシュされている結果を使用できるかどうかを判断します。キャッシュを共有できる場合、以降に実行するクエリーは格納されません。

- **SAGetSharedRequestKey**: Oracle BI Presentation Services から論理 SQL 文を取得し、リクエスト・キー値を返す ODBC プロシージャ。

  次に、このプロシージャの構文を示します。

  ```
  SAGetSharedRequestKey('sql-string-literal)
  ```

結果レコードについて

結果レコードには 2 つの列があります。最初の列は結果コードで、2 番目の列は消去操作の結果を説明する短いメッセージです。次の一覧表に、結果レコードの例を示します。

結果コード	結果メッセージ
1	SAPurgeCacheByDatabase returns successfully.
E_Execution_CacheNotEnabled	Operation not performed because caching is not enabled.
E_Execution_NonExistingDatabase	The database specified does not exist.

SAP/BW データソースのキャッシュの格納と消去

Microsoft Analysis Services では、メンバーのキャプション名はメンバーの一意の名前と同じです。ただし、SAP/BW データソースでは、メンバーのキャプション名と一意の名前は異なります。したがって、Oracle BI Server では、SAP/BW メンバーの一意の名前のキャッシュ・サブシステムを維持しています。デフォルトでは、このサブシステムはオフになっています。構成情報については、『Oracle Business Intelligence Infrastructure インストレーションおよび構成ガイド』で、NQSConfig.INI ファイルの MDX Member Name Cache セクションに関する項を参照してください。

メンバーの一意の名前に対するクエリーを受け取ったサブシステムは、キャッシュを確認して、そのクエリーに対するキャッシュが存在するかどうかを判断します。キャッシュが存在する場合は、キャッシュされている一意の名前のレコードを返します。クエリーと一致するキャッシュが存在しない場合、サブシステムは SAP/BW に調査クエリーを送信します。

ロギング・レベルが 2 以上に設定されていると、この調査クエリーはログに記録されます。サブシステムが有効になっているかどうかなどのサブシステムのステータス、および起動イベントや停止イベントなどのイベントも、サーバー・ログに書き込まれます。

警告：ロギング・レベルを高くするほど、パフォーマンスに対する影響が発生します。ユーザーのロギング・レベルを高くするときは注意が必要です。

SAP/BW データソースのキャッシュの消去

ODBC 消去プロシージャを実行する権限を持っているのは、Oracle BI の管理者のみです。キャッシュの保持は、Oracle BI の管理者が担当します。したがって、Oracle BI の管理者は、次の点を認識している必要があります。

- マルチディメンショナルなキャッシュ・エントリのサイズは、きわめて大きくなることがあります。したがって、各メンバー・セットのサイズには、NQSConfig.INI ファイルの MDX_MEMBER_CACHE セクションで制限が設定されています。
- アップグレードを実行すると、存続しているキャッシュのフォーマットから一貫性が失われることがあります。したがって、Oracle BI の管理者は、すべてのキャッシュを消去した後でソフトウェアをアップグレードする必要があります。
- 各クエリーを初めて実行したときに、キャッシュにデータが移入されます。Oracle BI の管理者は、パフォーマンスへの影響を最小限とするために、サーバーの使用率が低い時間帯に、キャッシュへのデータ移入を調整する必要があります。

注意： Oracle BI の管理者は、Oracle BI Administration Tool で個々のキューブ・テーブルを右クリックし、「Purge Member Cache」を選択することで、そのキューブ・テーブルのキャッシュを消去できます。この操作は、Oracle BI の管理者権限を持つユーザーがオンライン・モードで実行する必要があります。

次の消去プロシージャは、SAP/BW データソース専用です。

- **SAPurgeALLMCNCache**: すべての SAP/PW キャッシュ・エントリを消去します。

  次に、このプロシージャの構文を示します。

  ```
  SAPurgeALLIMCNCache ()
  ```

- **SAPurgeMCNCacheByCube**: 特定の物理キューブに関連付けられたキャッシュ・エントリをすべて消去します。データベース名とキューブ名はリポジトリ・オブジェクトの外部名です。次に、このプロシージャの構文を示します。

  ```
  SAPurgeMCNCacheByCube( 'DBName', 'CubeName')
  ```

次のメッセージが返されます。

リターン・コード	リターン・メッセージ
1	SAPurgeALLMCNCache returns successfully.
1	SAPurgeMCNCacheByCube returns successfully.
E_Execution_NonExistingDatabase(*)	The database specified does not exist. **注意：** データベースと物理キューブの両方が間違っていると、この結果コードが返されます。
E_Execution_NonExistingPhysicalCube	The physical cube specified does not exist.

キャッシュ使用の方針

クエリー・キャッシュを使用すると、クエリーのパフォーマンスが目に見えて向上するという大きなメリットが得られます。使用率が低い時間帯にクエリーを実行してその結果をキャッシュし、キャッシュをシードしておくと効果的です。良好なシード方針を確立するには、キャッシュ・ヒットが発生する条件を知ることが必要です。

すべてのユーザーで効果が得られるようにキャッシュをシードするには、次のクエリーを実行してキャッシュをシードします。

```
Select User, SRs
```

Select User, SRs を使用してキャッシュをシードしておくと、次のクエリーがすべてキャッシュ・ヒットになります。

```
Select User, SRs where user = valueof(nq_SESSION.USER) (and the user was USER1)

Select User, SRs where user = valueof(nq_SESSION.USER) (and the user was USER2)

Select User, SRs where user = valueof(nq_SESSION.USER) (and the user was USER3)
```

キャッシュ・ヒット

キャッシュを有効にしておくと、クエリーごとに、そのクエリーがキャッシュ・ヒットに該当するかどうかが評価されます。キャッシュ・ヒットとは、キャッシュを使用してサーバーからクエリーに答えることができ、その際、データベースにいっさいアクセスしなかったことを指します。

注意： Oracle BI Server では、クエリー・キャッシュを使用し、同等以上の集計レベルでクエリーに答えることができます。

キャッシュ・ヒットが発生するのは、この項で説明している条件がすべて成立した場合のみです。

- **WHERE 句が意味的に同一か、論理サブセットであること。** クエリーがキャッシュ・ヒットに該当するには、その WHERE 句による絞込み条件が、キャッシュされている結果と同一か、キャッシュされている結果のサブセットであることが必要です。

  キャッシュされているクエリーの論理サブセットである WHERE 句は、次の条件のいずれかが成立する場合にキャッシュ・ヒットに該当します。

  - IN リスト値のサブセットであること。

    キャッシュされているクエリーの IN リストより少ない数の要素をリクエストするクエリーが、キャッシュ・ヒットに該当します。例として、次のようなクエリーを考えてみます。

    ```
    select employeename, region
    from employee, geography
    where region in ('EAST', 'WEST')
    ```

    このクエリーは、キャッシュされている次のクエリーに対するヒットに該当します。

    ```
    select employeename, region
    from employee, geography
    where region in ('NORTH', 'SOUTH', 'EAST', 'WEST')
    ```

  - キャッシュされている結果より少ない数の同等な OR 絞込み条件を持つこと。

- リテラル比較の論理サブセットを持つこと。

  例として、次のような述語を考えてみます。

  ```
  where revenue < 1000
  ```

  この述語は、次の述語による比較クエリーに対してキャッシュ・ヒットとなります。

  ```
  where revenue < 5000
  ```

- WHERE 句がないこと。

  WHERE 句のないクエリーがキャッシュされていると、他のすべてのキャッシュ・ヒット・ルールを満足するクエリーは、その WHERE 句に関係なく、キャッシュ・ヒットに該当します。

- **SELECT 構文のリストにある列のサブセットであること。**新しいクエリーがキャッシュ・ヒットに該当するためには、その SELECT 構文のリストにあるすべての列が、キャッシュされているクエリーの中に存在しているか、そのクエリーにある列から計算できる必要があります。
- **SELECT 構文のリストにある列が、キャッシュされているクエリーの列に対する式から得られること。**Oracle BI Server では、キャッシュされている結果に対する式を計算して新しいクエリーに答えることができますが、計算に使用するすべての列が、キャッシュされている結果に存在している必要があります。

  例として、次のようなクエリーを考えてみます。

  ```
  select product, month, averageprice from sales where year = 2000
  ```

  このクエリーは、次のクエリーに対するキャッシュとヒットします。

  ```
  select product, month, dollars, unitsales from sales where year = 2000
  ```

  これは、averageprice = dollars/unitsales により、dollars と unitsales から averageprice を計算できるからです。

- **結合条件が同等であること。**新しいクエリーがキャッシュ・ヒットに該当するには、そのリクエストで得られる結合した論理テーブルが、キャッシュされている結果と同一か、そのサブセットであることが必要です。
- **DISTINCT 属性が同じであること。**キャッシュされているクエリーで、DISTINCT 処理（SELECT DISTINCT... など）を使用して重複レコードを削除している場合は、そのキャッシュされた列に対するリクエストでも、DISTINCT 処理を指定する必要があります。DISTINCT 処理を指定せずにその同じ列をリクエストすると、キャッシュ・ミスとなります。
- **相互に集計レベルの互換性があること。**情報の集計レベルをリクエストするクエリーでは、その集計レベルより低い集計レベルで、キャッシュされている結果を使用できます。

  例として、次のようなクエリーを考えてみます。

  ```
  select supplier, region, city, qtysold
  from suppliercity
  ```

  このクエリーでは、仕入先、地域および市区町村の各レベルの売上げ数量をリクエストします。一方、次のようなクエリーを考えてみます。

  ```
  select city, qtysold
  from suppliercity
  ```

  このクエリーでは、市区町村レベルの売上げ数量をリクエストします。この 2 番目のクエリーは、最初のクエリーに対するキャッシュ・ヒットとなります。

- **追加の集計が限定されていること。**たとえば、列が *qtysold* になっているクエリーがキャッシュされていると、RANK（qtysold）に対するリクエストはキャッシュ・ミスとなります。また、国レベルで qtysold をリクエストするクエリーは、国、地域のレベルで qtysold をリクエストするクエリーとキャッシュ・ヒットします。
- **ORDER BY 句が、SELECT 構文のリストにある列で構成されていること。**SELECT 構文のリストにない列の順序を指定するクエリーはキャッシュ・ミスとなります。

キャッシュにデータを移入するクエリーのスイートの実行

キャッシュ・ヒットが発生する可能性を高くするための方針として、キャッシュにデータを移入する目的でのみ、クエリーのスイートを実行することが考えられます。キャッシュをシードするクエリーのスイートを作成する際に使用するクエリーのタイプについて、推奨事項のいくつかを次にあげます。

- 共通のクエリーをビルトインしておく。

  頻繁に実行するクエリー、特に処理負荷が高いものは、キャッシュをシードするクエリーとして優れています。Intelligence Dashboard に埋め込まれる結果を返すクエリーは、共通のクエリーの好例です。

- SELECT 構文のリストに式を含めない。

  SELECT 構文のリストの列から式を除外すると、キャッシュ・ヒットの確率が高くなります。式を持ったままキャッシュされている列は、異なる式を持つ新しいクエリーには答えることができませんが、式を持たずにキャッシュされている列は、その列に対するリクエストがどのような式を指定していても、それに答えることができます。たとえば、次のリクエストがキャッシュされているとします。

  ```
  SELECT QUANTITY, REVENUE...
  ```

  このリクエストは、次のような新しいクエリーに答えることができます。

  ```
  SELECT QUANTITY/REVENUE...
  ```

  ただし、2 番目のクエリーは最初のクエリーに答えることはできません。

- WHERE 句を含めない。

  WHERE 句を持たずにキャッシュされている結果は、予測リストの列を扱う任意の WHERE 句を持つ SELECT 構文のリストに対するキャッシュ・ヒット・ルールを満足するクエリーへの回答に使用できます。

一般的に、キャッシュのシードに最適なクエリーとは、データベースの処理リソースを大量に消費し、さらに繰り返し発行される可能性が高いものです。大量の行を返すだけの単純なクエリーでキャッシュをシードしないようにします。このようなクエリーで必要となるデータベース処理量はわずかです。たとえば、SELECT * FROM PRODUCTS というクエリーで、PRODUCTS が単一のデータベース・テーブルに直接マップされている場合です。これらのクエリーではネットワークとディスクに大きなオーバーヘッドが発生しますが、これはキャッシュの機能では軽減できません。

注意：Oracle BI Server でリポジトリ変数がリフレッシュされるとき、ビジネス・モデルでそのリポジトリ変数が参照されていないか確認されます。参照されている場合は、それらのビジネス・モデルのキャッシュがすべて消去されます。

Oracle BI Server のクエリーに関する集計の作成

集計テーブルには、一連のディメンション属性にわたる集計メジャー（普通は合計）である事前計算した結果が格納されます。集計テーブルは、意思決定支援システムのレスポンス時間短縮でよく使用される手法です。

SQL クエリーを記述する場合や、物理テーブルが存在することのみ認識していて、その意味を理解していないツールを使用する場合は、集計テーブルの数が増えるので、その使用方法は複雑になります。Oracle BI Server の集計ナビゲーション機能を使用すると、集計テーブルに格納された情報をクエリーで自動的に使用できます。Oracle BI Server では、ユーザーはビジネス上の正しい質問に集中できます。迅速に回答を提供するテーブルがどれであるかはサーバーが判断します。

Oracle BI Server のクエリーについて集計を作成する従来のプロセスは手動です。この方法では複雑な DDL と DML を記述して、関連するデータベースにテーブルを作成する必要があります。これに加え、これらのテーブルは、クエリーに使用できるように、リポジトリのメタデータにマップする必要があります。この作業は時間がかかるうえ、エラーが発生しやすいプロセスです。Oracle BI の管理者は、Aggregate Persistence モジュールを使用すると、集計テーブルの作成とメタデータへのマッピングを自動化できます。

集計の作成は、クラスタのマスター・サーバーに対して実行されます。メタデータの変更がスレーブに移入されるまで少し時間がかかります。クラスタのリフレッシュ時間はユーザー制御のオプションなので、スレーブ・サーバーがリフレッシュされる前の内容にクエリーが一致していると、正しくない結果が得られることがあります。適切なクラスタ・リフレッシュ間隔の設定は、Oracle BI の管理者が担当します。

NQSConfig.INI ファイルの GENERAL セクションには、次のオプションのパラメータがあります。

```
AGGREGATE_PREFIX = "user specified short prefix for dimension aggregates" ;
```

このパラメータに指定できる文字数は 8 文字までです。自動的に生成されるディメンション（レベル）の集計には、ここで指定した接頭辞が付加されます。これを指定しない場合、デフォルトの接頭辞 SA_ が使用されます。

注意：集計を管理できるのは、Oracle BI の管理者グループのみです。

この項の内容は次のとおりです。

集計のクエリー候補の特定

集計を作成するときは、集計したデータから大きな効果を得ることができるクエリーを特定する必要があります。可能なかぎり高いレベルまで集計することで、最適な結果が得られます。動作が遅いクエリーを特定するには、次の作業を実行します。

- Oracle BI Server で使用状況トラッキングを有効にします。

  使用状況トラッキングの統計は様々な用途に利用できます。たとえば、データベースの最適化、集計方針の策定、消費したリソースに応じたユーザーや部署への課金などが可能です。Oracle BI Server では、詳細なクエリー・レベルで使用状況を追跡します。使用状況トラッキングを有効にすると、あらゆるクエリーの統計が使用状況トラッキングのログ・ファイルに書き込まれるか、データベース・テーブルに挿入されます。

  注意：データベースへの直接挿入を使用することを強くお薦めします。手順については、「使用状況トラッキングの管理」（226 ページ）を参照してください。

- クエリーの実行時間を分析して、集計の候補として動作の最も遅いクエリーを特定します。

  集計を作成するための実行時間は、ユーザーが選択した集計のタイプによって異なります。大きなファクト・テーブルから集計を作成する場合は、小さなテーブルから作成する場合よりも時間がかかります。Oracle BI の管理者は、作成する集計を慎重に選択する必要があります。

Create Aggregates 仕様の記述について

スクリプト・ファイルを作成するには、Administration Tool で Aggregate Persistence Wizard を使用するか、ファイルを手動で記述します。

注意：Aggregate Persistence Wizard を使用することをお薦めします。手順については、「Aggregate Persistence Wizard」（194 ページ）を参照してください。

集計の作成中にデータベースが変更されないようにする場合は、Aggregate Persistence Wizard を使用して SQL ファイルを作成します。SQL を作成した後、データベース管理プロセスを使用して集計テーブルを作成できます。

作成プロセスで発生する制約

この項では、作成プロセスで発生する制約について説明します。

有効なメジャー

有効なメジャーには有効な集計ルールが必要です。

- レベルベースのメジャーに適用される制約は次のとおりです。
  - レベルが総計の別名である場合、そのディメンションは、その集計仕様のレベルのリストに存在しないことが必要です。
  - このメジャーに定義した他のすべてのレベルは、その集計仕様のレベルのリストに存在する必要があります。

  前述の制約を満足しないと、集計仕様全体が破棄されます。

- 次のいずれかに該当するメジャーは、作成プロセスで無視されます。
  - セッション変数またはリポジトリ変数にマップされているメジャー。

- 導出されたメジャー。

無視されたメジャーは、必然的に集計仕様には影響しなくなります。集計の作成には残りのメジャーが使用されます。

有効なレベル

有効なレベルには有効な主キーが必要です。

- レベルが無効な場合、集計仕様は破棄されます。
- 次のいずれかに該当する場合、レベルの属性またはその主キーは無視されます。
  - セッション変数またはリポジトリ変数にマップされている属性。
  - 同じ論理テーブルから得られたものではない属性。

有効な集計仕様

有効な集計仕様には次の特性があります。

- 名前の長さが 1 ～ 18 文字です。
- 有効なレベルを 1 つ以上指定する必要があります。
- 有効なメジャーを 1 つ以上指定する必要があります。
- 有効な接続プールが必要です。
- 有効な出力コンテナ（データベース、カタログ、スキーマ）が必要です。
- 接続プールとコンテナは、同じデータベースに属している必要があります。
- ディメンションごとに指定できるレベルは 1 つのみです。
- 同じファクト・テーブルにあるメジャーのみを使用できます。
- 仕様の論理要素は、同じサブジェクト領域にあるもののみです。

集計仕様は、それと同じ名前が出力コンテナにすでに存在すると無視されます。これは、レベル集計の有効範囲がデータベース全体に及ぶためです。ただし、同じファクト集計名に対して複数の異なるカタログまたはスキーマを指定すると、同じデータベースの中であっても有効範囲が異なっていれば、名前の同じファクトを複数設定できます。

Create Aggregates 仕様を記述するためのガイドライン

論理ファクト列を除き、すべてのメタデータ名は完全修飾とします。操作のモードには、Create と Delete の 2 つがあります。

すべての集計仕様を単一の Create Aggregates 文に記述することを強くお薦めします。

- スクリプト・ファイルでは、先頭に Delete 文を記述します。新しい集計を作成する前に、システムで生成された集計をすべて削除する必要があります。これによってデータの一貫性を確保でき、無効な集計や不完全な集計を削除してから Create 操作を実行できます。集計を削除する構文は次のとおりです。

  ```
  Delete aggregates;
  ```

- この文の後には Create 文を記述します。集計を作成する構文は次のとおりです。

```
Create|Prepare aggregates

<aggr_name_1>

for  logical_fact_table_1 [(logical_fact_column_1, logical_fact_column_2,…)]

at levels (level_1, level_2, …)

using connection pool <connection_pool_name_1>

in <schema_name_1>

[ ,<aggr_name_2>

for logical_fact_table_3 [(logical_fact_column_5, logical_fact_column_2,…)]

at levels (level_3, level_2, …)

using connection pool <connection_pool_name_2>

in <schema_name_2>] ;
```

- 複数の集計の作成。単一の Create Aggregates 文で複数の集計を指定するには、次のガイドラインに従ってください。
  - 各集計仕様はカンマで区切り、集計作成スクリプト全体の末尾にはセミコロンを付けます。
  - このファイルでは、その先頭で Delete Aggregates 文を 1 つだけ指定します。Oracle BI の管理者は、ETL の実行ごとに発行される削除が、（リセットが要求されないかぎり）1 回のみであることを確認する必要があります。

    警告：最初の集計スクリプトの後で実行するどの集計スクリプトにも、Delete Aggregates 文を記述しないようにします。記述すると、それまでに作成した集計がすべて削除されます。
- サロゲート・キーを使用した集計の作成。手順の詳細は、「ディメンション集計テーブルへのサロゲート・キーの追加について」（253 ページ）を参照してください。

ディメンション集計テーブルへのサロゲート・キーの追加について

ファクト集計テーブルとレベル集計テーブルとの結合オプションのデフォルトでは、レベル集計の主キーが使用されます。レベルの主キーが大きく複雑なもの（数多くの列の複合）であると、ファクト・テーブルへの結合は負荷の大きな処理となります。サロゲート・キーは人為的に生成するキーで、通常は数字を使用します。レベル集計テーブルにサロゲート・キーが生成されると、この結合操作が簡素化され、ファクト・テーブルから不要な列（レベルの主キー）が削除され、ファクト・テーブルのサイズが小さくなります。ディメンション（レベル）集計テーブルにサロゲート・キーを追加することで、ファクト・テーブルへの結合を簡素化でき、クエリーのパフォーマンス向上が期待できます。さらに、サロゲート・キーによって、各集計テーブルに一意の識別子が必ず存在するようになります。

1 つのレベルを複数のファクト・テーブルで共有することがあります。この場合、1 つのファクトで複数のサロゲート・キーを使用し、別のファクトでディメンション集計の主キーを使用することがあります。次に、この問題の解決方法をいくつか示します。

- サロゲート・キーと主キーのどちらを使用するかを示すメタデータのプロパティを、レベルに設定します。
- レベル集計のサロゲート・キーを必ず作成します（運用コストが下がります）。後で、サロゲート・キーまたは主キーを使用してファクト集計をレベル集計に結合するかどうかを判断します。

ディメンションごとに結合タイプを指定する方法に代えて、スター全体でサロゲート・キーを使用するかどうかを指定する方法もあります。これによって構文はシンプルになりますが、ユーザーが利用できるオプションが制限され、集計作成プロセスの動作が遅くなります。

Create/Prepare Aggregates へのサロゲート・キー入力

Oracle BI の管理者は、次の結合オプションを使用して集計スターを作成できます。

- 主キー（オプションを指定しない場合のデフォルト）
- サロゲート・キー

Create/Prepare Aggregates の構文

次に示す Create/Prepare Aggregates の構文では、[Using_Surrogate_Key] の変更が記述されています。サロゲート・キーのオプションはレベルごとに指定できます。指定しない場合、ファクト・テーブルとディメンション・テーブルは、レベル集計の主キーを使用して結合されます。

```
Create|Prepare aggregates

<aggr_name_1>

[file <output_file_name>]

for  logical_fact_table_1 [(logical_fact_column_1, logical_fact_column_2,…)]

at levels (level_1 [Using_Surrogate_Key], level_2, …)

using connection pool <connection_pool_name_1>

in <schema_name_1>

[ ,<aggr_name_2>

for logical_fact_table_3 [(logical_fact_column_5, logical_fact_column_2,…)]

at levels (level_3, level_2, …)

using connection pool <connection_pool_name_2>

in <schema_name_2>] ;
```

Create/Prepare Aggregates からのサロゲート・キー出力

現在のプロセスの変更可能な範囲は、リポジトリの物理メタデータ・レイヤーおよびデータベースに制限されています。

UseSurrogateKeys 結合オプションを使用すると、次に説明する結果が得られます。

- レベル集計については、次の結果となります。
  - 物理メタデータでは、次の結果となります。
    - レベル集計テーブルには、<level_name>_SKEY という新しい列が作成されます（衝突の確認用）。これは、ディメンション集計のサロゲート・キー列です。
    - この列のタイプは UINT です。

- データベースでは、次の結果となります。
  - レベル集計テーブルにも、<level_name>_SKEY という対応する列が作成されます。
  - この列には、RCOUNT () を使用してデータを移入できます。
- ファクト集計については、次の結果となります。
  - 物理メタデータでは、次の結果となります。
    - ファクト集計テーブルには、レベルの主キーから得た列は存在しなくなります。
    - かわりに、レベル集計のサロゲート・キーに対応する新しい列が、このテーブルに追加されます。
    - この列のタイプは、レベルのサロゲート・キーと同じになります。
    - この列の名前は、レベル集計の対応する列と同じ名前になります（衝突の確認用）。
    - ファクト・テーブルとレベル・テーブルは、このサロゲート・キーのみを使用して結合されます。
  - データベースでは、次の結果となります。
    - ファクト集計テーブルにも、対応するサロゲート・キーが作成されます。
    - この列には、Populate で利用できる新しい機能を使用してデータが移入されます。

SQL スクリプト・ファイルの生成

集計の指定方法が理解できていれば、集計論理 SQL を記述できます。また、Aggregate Persistence Wizard を使用して SQL スクリプト・ファイルを生成することもできます。

注意： Aggregate Persistence Wizard を使用すると、前述の項で説明した制約の多くが自動的に適用されるので、このウィザードの使用を強くお薦めします。SQL スクリプトを手動で記述する場合は、「Create Aggregates 仕様を記述するためのガイドライン」（252 ページ）の説明にある構文を使用します。

ロギング・レベルの設定について

ロギング・レベルが 2 以上であれば、トレース・ログが NQQuery.log に記録されます。ロギング・イベントには、集計の実行計画や、集計の作成順序と削除順序などがあります。ロギング・レベルが高いほど、クエリーと実行計画に関する情報が詳しく記録されます。

エラー・ログは、ロギング・レベルが 1 以上であれば NQQuery.log に記録されるほか、ロギング・レベルに関係なく NQServer.log に記録されます。

SQL スクリプト・ファイルの実行による集計の作成と削除

SQL スクリプト・ファイルを生成し、ロギング・レベルを設定した後、その SQL スクリプトを実行する必要があります。コマンドライン・プロンプト、または Administration Tool の Job Manager ユーティリティから nQCmd.exe を実行できます。

注意： nQCmd.exe の実行には Job Manager の使用を強くお薦めします。詳細は、『Oracle Business Intelligence Scheduler ガイド』を参照してください。nQCmd.exe を実行する場合、Oracle BI のインストール・フォルダである [*installdrivepath*]:¥OracleBI¥server¥Bin でバージョンを確認できます。

SQL スクリプトを実行すると集計が作成され、Oracle BI Server メタデータとバックエンド・データベースに保持されます。

作成後のアクティビティ

この項の内容は次のとおりです。

- データベースの索引作成について（256 ページ）
- エラー処理について（256 ページ）

データベースの索引作成について

現時点では、データベース・テーブルの索引は自動生成されません。必要に応じて、データベース管理者がデータベース・テーブルにこれらの索引を手動で作成します。ディメンション・テーブルは自動生成されるので、ロギング・レベルを 2 以上に設定すると、NQQuery.log で Aggregate Creation Plan を確認するうえで便利です。この計画は、集計仕様と並んで、データベースに自動生成されたテーブルを探すための参照として使用できます。

EXECUTE PHYSICAL 機能を使用すると、論理 SQL スクリプトに索引の削除と構築の処理を手動で埋め込むことができます。スクリプトの先頭に追加する文として、次の例が考えられます。

```
EXECUTE PHYSICAL CONNECTION POOL "SQL_Paint"."SQL_Paint" DROP INDEX demo_index1;

CREATE AGGREGATES......;

EXECUTE PHYSICAL CONNECTION POOL "SQL_Paint"."SQL_Paint" CREATE INDEX demo_index1
ON table1(col1);
```

エラー処理について

エラーの発生原因としては、次のようなものが考えられます。

- ネットワークの障害
- データベース上のディスク領域不足
- 集計リクエストが不適切

集計の作成でエラーが発生すると、その集計リクエスト全体が中断され、それ以降は集計が作成されなくなります。作成とチェックインが済んでいる集計は、チェックインされたままです。エラーが発生した場合は、エラー発生時または次回の ETL 実行時に、次のいずれかの方法でその集計を削除する必要があります。

- メタデータおよびデータベースから、該当の集計を手動で削除する
- Delete Aggregates 仕様を使用して、すべての集計を自動的に削除する

イベント・ポーリング・テーブルによるキャッシュ・イベント処理

Oracle BI Server のイベント・ポーリング・テーブル（イベント・テーブル）を使用して、更新された物理テーブルが 1 つ以上あることを Oracle BI Server に通知できます。イベント・テーブルに追加された行それぞれには 1 件の更新イベントが記述されています。たとえば、11308Production データベースの Product テーブルに発生した更新などです。Oracle BI Server のキャッシュ・システムでは、イベント・テーブルの行を読み取り（ポーリング）、その行から物理テーブルの情報を抽出して、それらの物理テーブルを参照している古いキャッシュ・エントリを消去します。

イベント・テーブルは、Oracle BI Server からアクセス可能なデータベースに存在する物理テーブルです。その存在する場所がイベント・テーブル専用のデータベースであるか、また他のテーブルも格納されているデータベースであるかに関係なく、イベント・テーブルには 258 ページの表 27 の説明にあるように、固定したスキーマが必要です。通常、このスキーマは Administration Tool の「Physical」レイヤーのみに表示され、「Physical Table」ダイアログ・ボックスで Oracle BI Server のイベント・テーブルとして識別されます。

古くなったキャッシュ・エントリを無効にするうえで、イベント・テーブルを使用する方法は最も精密な結果が得られるもので、また高い信頼性を持つ方法であるといえます。ただし、この方法では、データベース・テーブルを更新するたびに、イベント・テーブルにデータを移入する必要があります（「Oracle BI Server のイベント・ポーリング・テーブルへの移入」（261 ページ）を参照）。また、ポーリング間隔ではキャッシュが完全には最新の状態になっていないので、古いデータがキャッシュに存在している可能性が常にあります。

イベント・テーブルを更新する際には、データベースにデータを移入する抽出スクリプトおよびロード・スクリプトまたはプログラムに、SQL の INSERT 文を記述する方法が一般的です。この INSERT 文は、物理テーブルが変更されるたびにイベント・テーブルに行を 1 つ追加します。このプロセスを配置し、Oracle BI のリポジトリにイベント・テーブルを構成すると、キャッシュが自動的に無効化されるようになります。イベント・テーブルに対する変更が、このテーブルを更新するスクリプトによって正確に記録されていれば、古いキャッシュ・エントリは指定したポーリング間隔で自動的に消去されます。

物理データベースに対するイベント・ポーリング・テーブルの設定

この項では、物理データベースに対して、Oracle BI Server のイベント・ポーリング・テーブルを設定する方法について説明します。

ポーリング・テーブルの構造

物理データベースごとに物理的なイベント・ポーリング・テーブルを設定して、各データベースに発生する変更を監視できます。また、専用のデータベースにイベント・テーブルを設定することもできます。イベント・テーブルは、データベースを構成するテーブルが変更されるたびに更新する必要があります。イベント・テーブルは、258 ページの表 27 に示す構造を持つ必要があります。イベント・テーブルが存在する場所によっては、いくつかの列には null 値を格納できます。

イベント・テーブルの列名には名前の候補が示されますが、任意の名前を使用できます。ただし、列の順序は 258 ページの表 27 に示すとおりにする必要があります。「イベント・ポーリング・テーブルの CREATE TABLE 文のサンプル」（259 ページ）には、イベント・ポーリング・テーブルを作成する CREATE TABLE 文のサンプルがあります。

表 27. イベント・ポーリング・テーブルの列名

イベント・テーブルの列名	データ型	説明
CatalogName	CHAR または VARCHAR	更新された物理テーブルが存在するカタログの名前。 イベント・テーブルが存在するデータベースと、更新された物理テーブルが存在するデータベースが異なる場合にのみ、「CatalogName」列にデータを移入します。それ以外の場合は、null 値を設定します。
DatabaseName	CHAR または VARCHAR	更新された物理テーブルが存在するデータベースの名前。これは、Administration Tool の「Physical」レイヤーで定義したデータベースの名前です。たとえば、物理データベース名が 11308Production で、それを Administration Tool では SQL_Production というデータベース名で表現している場合、イベント・テーブルのポーリング行にはデータベース名として SQL_Production が表示される必要があります。 イベント・テーブルが存在するデータベースと、更新された物理テーブルが存在するデータベースが異なる場合にのみ、「DatabaseName」列にデータを移入します。それ以外の場合は、null 値を設定します。
Other	CHAR または VARCHAR	将来の拡張のために予約されています。この列には null 値を設定する必要があります。
SchemaName	CHAR または VARCHAR	更新された物理テーブルが存在するスキーマの名前。 イベント・テーブルが存在するデータベースと、更新される物理テーブルが存在するデータベースが異なる場合にのみ、「SchemaName」列にデータを移入します。それ以外の場合は、null 値を設定します。
TableName	CHAR または VARCHAR	更新された物理テーブルの名前。この名前は、Administration Tool の「Physical」レイヤーでこのテーブルに定義した名前と一致している必要があります。 null 値は設定しないでください。

表 27. イベント・ポーリング・テーブルの列名

イベント・テーブルの列名	データ型	説明
UpdateTime	DATETIME	イベント・テーブルの更新が発生した日時。これは、イベント・テーブルに行が追加されるたびに増加するキー（一意）値とする必要があります。一意な増加する値となるようにするには、列のデフォルト値として現在のタイムスタンプを指定します。たとえば、Oracle 8*i* では DEFAULT CURRENT_TIMESTAMP と指定します。 null 値は設定しないでください。
UpdateType	INTEGER	標準の更新であることを示すには、更新スクリプトで値 1 を指定します（他の値は将来使用するために予約されています）。 null 値は設定しないでください。

Oracle BI Server では、イベント・ポーリング・テーブルに対する読取り権限と書込み権限を設定する必要があります。サーバーは、指定された間隔でイベント・テーブルを読み取り、変更されたデータを探します。（ロード操作などで）データベース・テーブルが変更されると、アプリケーションによってイベント・テーブルに行が追加されます。イベント・テーブルに行が存在する場合は、基礎となるデータベースのデータが変更されているということになります。サーバーはさらに、変更された物理テーブルに対応するキャッシュ・エントリをすべて無効にし、古い行をイベント・テーブルから定期的に削除します。イベント・テーブルの次回確認時にも、このプロセスが繰り返されます。

注意： 1 つのイベント・ポーリング・テーブルを複数の Oracle BI Server で共有することはできません。複数の Oracle BI Server を設定した場合は、サーバーごとにイベント・ポーリング・テーブルを作成する必要があります。

Oracle BI Server を、イベント・ポーリング・テーブルのみに書込みアクセスができ、データベースにある他のテーブルには書込みアクセスできないようにするには、次の手順を実行します。

- Administration Tool の「Physical」レイヤーで、権限付きの接続プールを使用して、独立した物理データベースを作成します。
- 削除権限のある接続プールにユーザーを割り当てます。
- この権限付きのデータベースにイベント・テーブルのデータを移入します。

Oracle BI Server は、イベント・ポーリング・テーブルには書込みできますが、ユーザーのクエリーに答えるために使用されているテーブルには書込みアクセスできなくなります。

イベント・ポーリング・テーブルの CREATE TABLE 文のサンプル

SQL Server 7.0 用および Oracle 8*i* 用の CREATE TABLE 文のサンプルを次に示します。これらの CREATE TABLE 文では、Oracle BI Server のイベント・ポーリング・テーブルに必要な構造を作成します。これらの文で作成したテーブルには、UET という名前が付けられます。このテーブルは、更新される物理テーブルと同じデータベースに存在します。

注意： 列は、リポジトリにあるオブジェクト名を表現するうえで十分な長さにする必要があります。

SQL Server 7.0 用の CREATE TABLE 文を次に示します。

```
// SQL Server 7.0 Syntax
create table UET (
  UpdateType Integer not null,
  UpdateTime datetime not null DEFAULT CURRENT_TIMESTAMP,
  DBName     char(40) null,
  CatalogName varchar(40) null,
  SchemaName  varchar(40) null,
  TableName   varchar(40) not null,
  Other       varchar(80) null DEFAULT NULL
)
```

Oracle 8*i* 用の CREATE TABLE 文を次に示します。

```
// Oracle 8i syntax
create table UET (
  UpdateType Integer not null,
  UpdateTime date DEFAULT SYSDATE not null,
  DBName     char(40) null,
  CatalogName varchar(40) null,
  SchemaName  varchar(40) null,
  TableName   varchar(40) not null,
  Other       varchar(80) DEFAULT NULL
);
```

SQL Server と Oracle の別バージョンや他のデータベースでは、これらの CREATE TABLE 文を多少変更する必要が生じることがあります。さらに、記憶域に関する句を明示的に指定するには、適切な句をこの文に追加する必要があります。

イベント・ポーリング・テーブルのアクティブ化

物理データベースにテーブルを作成すると、Oracle BI Server でそのテーブルをアクティブにすることができます。

ポーリング・テーブルをアクティブにするには

1. テーブルを「Physical」レイヤーにインポートします。
2. 「Tools」→「Utilities」→「Oracle Event Tables」メニュー項目を使用して、Oracle BI Server のイベント・ポーリング・テーブルのグループにテーブルを追加し、ポーリング間隔を設定します。

ポーリング・テーブルを「Physical」レイヤーにインポートするには、開いたリポジトリから次の手順を実行します。

テーブルを「Physical」レイヤーにインポートするには

1. 「File」→「Import...」を選択します。
2. インポートするイベント・テーブルが格納されているデータソースを選択して「OK」をクリックします。

   「Import」ダイアログ・ボックスが表示されます。
3. 「Tables」オプションを選択して、テーブルのメタデータをインポートします。

4 イベント・ポーリング・テーブルにナビゲートしてテーブルを選択し、「Import」ボタンをクリックするか、そのテーブルを「Physical」レイヤーにドラッグ・アンド・ドロップします。

これで、「Physical」レイヤーにイベント・テーブルがインポートされます。複数のイベント・ポーリング・テーブルがある場合は、イベント・テーブルごとにこの手順を繰り返します。

注意：イベント・テーブルに対して指定したデータソースが、そのイベント・テーブルに対して読取りと書込みのアクセス権限があることを確認します。このリポジトリでは、テーブルの読取りとそこからの行の削除の両方を実行するので、書込み権限が必要です。イベント・テーブルを、「Business Model and Mapping」レイヤーで表示できるようにする必要はありません。

1 つ以上のポーリング・テーブルを「Physical」レイヤーに置いた後、それらをイベント・ポーリング・テーブルのグループに追加する必要があります。

テーブルのオブジェクトをイベント・ポーリング・テーブルとしてマークするには

1 「Tools」→「Utilities」メニュー項目をクリックします。

2 オプションの一覧から、オプション「Oracle BI Event Tables」を選択します。

3 「Execute」をクリックします。

4 イベント・テーブルとして登録するテーブルを選択して「>>」ボタンをクリックします。

5 ポーリング間隔を分単位で指定して「OK」をクリックします。

デフォルト値は 60 分です。

注意：ポーリング間隔は、10 分未満には設定しないようにしてください。極端に短いポーリング間隔が必要な場合は、一部またはすべてのテーブルをキャッシュ不可とすることを検討してください。

Oracle BI Server のイベント・テーブルとして登録したテーブルでは、プロパティが変化します。イベント・ポーリング・テーブルで得られた結果をキャッシュする理由はないので、テーブルをイベント・テーブルとして登録すると、テーブルをキャッシュ可能とするオプションはなくなります。

Oracle BI Server のイベント・ポーリング・テーブルへの移入

Oracle BI Server では、イベント・ポーリング・テーブルへの移入は行われません。テーブルに更新が発生するたびに行が挿入されることで、イベント・テーブルへの移入が行われます。通常、このプロセスはデータベース管理者が設定します。普通は、テーブルが変更されるたびにポーリング・テーブルに行が挿入されるように、ロード・プロセスを変更します。この処理は、ロード・スクリプト、データベース・トリガ（トリガをサポートしているデータベースの場合）、アプリケーションまたは手動操作で可能です。Oracle BI Server 側ではポーリング・テーブルに収められている情報は正確で最新であるとみなしているので、イベント・テーブルへの移入プロセスが正しく実行されていないと、キャッシュの消去が影響を受けます。

イベント・ポーリング・テーブルに発生した問題のトラブルシューティング

キャッシュのポーリングに問題が発生した場合は、Oracle BI Server のアクティビティ・ログで、サーバーとイベント・テーブルとのやり取りを記録したエントリを探します。

- NQServer.log ファイルには、Oracle BI Server に関するアクティビティが自動的に記録されます。このファイルのデフォルトの格納場所は、Oracle BI Server ソフトウェアのインストール・フォルダ直下の Log フォルダ（¥OracleBI¥Log）です。ログのエントリはわかりやすく記述されていて、テキスト・エディタで表示できます。
- Oracle BI Server でイベント・テーブルをポーリングしていれば、Oracle BI の管理者のロギング・レベルを 0 に設定していないかぎり、Oracle BI の管理者のユーザー ID を使用してクエリーが NQQuery.log に記録されます。最も有用なレベルの情報を得るには、Oracle BI の管理者のユーザー ID についてロギング・レベルを 2 に設定します。NQQuery.log ファイルのデフォルトの格納場所は、Oracle BI Server ソフトウェアのインストール・フォルダ（¥OracleBI）直下の Log フォルダです。ユーザーレベルのロギングの詳細は、「Oracle BI Server のクエリー・キャッシュ」（237 ページ）を参照してください。

リポジトリの変更

Oracle BI のリポジトリを変更すると、その変更によって、キャッシュに格納されているエントリが影響を受けることがあります。たとえば、物理オブジェクトや動的リポジトリ変数の定義を変更すると、そのオブジェクトや変数を参照しているキャッシュ・エントリが無効になります。このような変更の結果、キャッシュの消去が必要になることがあります。ここでは、オンライン・モードで変更が発生した場合、オフライン・モードで変更が発生した場合、およびリポジトリを切り替える場合の 3 種類のシナリオを考慮する必要があります。

オンライン・モード

オンライン・モードで Oracle BI のリポジトリを変更したとき、キャッシュ・エントリに影響する変更があると、変更したオブジェクトを参照するキャッシュ・エントリがすべて、自動的に消去されます。この消去処理は、変更をチェックインしたときに実行されます。たとえば、リポジトリから物理テーブルを削除すると、チェックイン時に、そのテーブルを参照しているキャッシュ・エントリがすべて消去されます。「Business Model and Mapping」レイヤーでビジネス・モデルを変更すると、そのビジネス・モデルのキャッシュ・エントリがすべて消去されます。

オフライン・モード

オフライン・モードで Oracle BI のリポジトリを変更したとき、その変更の中には、キャッシュに格納されているクエリーに影響し、キャッシュされているこれらの結果を古いものにする内容が含まれている可能性があります。オフライン・モードでの編集作業ではリポジトリがサーバーにロードされないため、キャッシュ・エントリが変更の影響を受けるかどうかをサーバー側では判断できません。したがって、オフラインで変更した後では、サーバーでのキャッシュの自動消去は実行されません。キャッシュを消去しないと、リポジトリを次回ロードしたときに無効なエントリが存在する可能性があります。オフラインで行った変更の影響を受けるエントリがキャッシュに存在しないことが確かである場合を除き、変更したビジネス・モデルのキャッシュは消去する必要があります。

リポジトリの切替え

Oracle BI Server の構成からリポジトリを削除する場合は、そのリポジトリを参照しているすべてのキャッシュ・エントリのキャッシュを必ず消去します。これを実行しないと、キャッシュが破損します。詳細は、「キャッシュの消去」（265 ページ）を参照してください。

Cache Manager の使用

Cache Manager を使用すると、Oracle BI の管理者は、クエリー・キャッシュ全体に関する情報のほか、クエリー・キャッシュの中で、開いているリポジトリに関連する個々のエントリに関する情報を確認できます。また、特定のキャッシュ・エントリを選択して、キャッシュされている SQL コールの表示や保存、エントリの消去などの様々な操作を、そのエントリに対して実行できます。

Cache Manager を起動するには

- Administration Tool のツールバーで、「Manage」→「Cache」を選択します。

左のエクスプローラ・ペインで「Cache」タブを選択して、現在のリポジトリ、ビジネス・モデルおよびユーザーのキャッシュ・エントリを表示します。関連するキャッシュ・エントリが右のペインに表示され、エントリの総数が上部の表示専用フィールドに表示されます。

キャッシュ・エントリの情報とその表示順序は、「Options」での設定で制御します（Cache Manager で「Edit」→「Options...」を選択するか、Administration Tool で「Tools」→「Options」→「Cache Manager」タブを選択します）。この情報には、263 ページの表 28 に示すオプションが含まれる場合があります。

表 28. キャッシュのオプション

オプション	説明
Business model	キャッシュ・エントリに関連付けられたビジネス・モデルの名前。
Column count	このキャッシュ・エントリの結果セットの各行に含まれる列の数。
Created	キャッシュ・エントリの結果セットが作成された日時。
Creation elapsed time	このキャッシュ・エントリの結果セットの作成に要した時間（秒）。 **注意：** この値は、ディスク上のキャッシュ・オブジェクト記述子にミリ秒の単位で格納されています。表示されるときに秒単位に変換されます。
Full size	「Full size」は、可変長の列や圧縮アルゴリズムなどの要因を考慮した、使用する最大サイズです。結果セットの実際のサイズは、このキャッシュ・エントリの結果セットの作成に必要な最大サイズ（秒単位）よりも小さくなります。
Last used	このキャッシュ・エントリの結果セットでクエリーを最後に満足した日時（Oracle BI Server が予期せずに停止した後は、最後の使用日時として、正しい日時の値よりも古い値が一時的に記録されていることがあります）。
Query Server	クエリーを処理した Oracle BI Server。
Row count	クエリーで生成された行の数。
Row size	このキャッシュ・エントリの結果セットに含まれる各行のサイズ（バイト）。

表 28. キャッシュのオプション

オプション	説明
SQL	このキャッシュ・エントリに関連付けられた SQL 文。
Use count	このキャッシュ・エントリの結果セットが（Oracle BI Server の起動後に）クエリーを満足した回数。
User	このキャッシュ・エントリを生成したクエリーを発行したユーザーの ID。

リポジトリ・ツリーを開くと、キャッシュ・エントリのあるビジネス・モデルがすべて表示され、ビジネス・モデルを開くと、キャッシュ・エントリのあるユーザーがすべて表示されます。右ペインには、階層ツリーで選択した項目に関連付けられたキャッシュ・エントリのみが表示されます。

グローバル・キャッシュ情報の表示

グローバル・キャッシュ情報を表示するには、「Action」→「Show Info...」を選択します。264 ページの表 29 では、「Global Cache Information」ウィンドウに表示される情報について説明します。

表 29. グローバル・キャッシュ情報

列	説明
Amount of space still available for cache storage use	キャッシュ記憶域にまだ使用できる領域（MB）。
Amount of space used on disks containing cache related files	キャッシュ関連ファイルを格納したディスクの全領域のうち、キャッシュ関連ファイルだけでなく、すべてのファイルで使用している領域の合計（MB）。
Maximum allowable number of entries in cache	キャッシュに格納できるエントリの最大数。NQSConfig.INI ファイルの MAX_CACHE_ENTRIES パラメータで指定します。
Maximum allowable number of rows per cache entry result set	各キャッシュ・エントリの結果セットに格納できる行の最大数。NQSConfig.INI ファイルの MAX_ROWS_PER_CACHE_ENTRY パラメータで指定します。
Number of entries currently in cache	グローバル・キャッシュにある現在のエントリの数。これらのエントリは、複数のリポジトリに関連していることがあります。
Number of queries not satisfied from cache since startup of Oracle BI Server	前回 Oracle BI Server が起動した時点からのキャッシュ・ミスの件数。
Number of queries satisfied from cache since startup of Oracle BI Server	前回 Oracle BI Server が起動した時点からのキャッシュ・ヒットの件数。

Cache Manager をアクティブなウィンドウにし、[F5] を押すか、「Action」→「Refresh」を選択して、表示をリフレッシュします。これにより、開いているリポジトリの現在のキャッシュ・エントリ、および現在のグローバル・キャッシュ情報が表示されます。DSN がクラスタ化されている場合は、クラスタにあるすべてのリポジトリに関する情報が表示されます。

キャッシュの消去

キャッシュの消去とは、クエリー・キャッシュからエントリを削除するプロセスです。キャッシュ・エントリは、次の方法で消去できます。

- Administration Tool の Cache Manager 機能をオンライン・モードで使用する手動処理。
- 特定のテーブルについて「Physical Table」ダイアログ・ボックスの「Cache Persistence Time」フィールドを設定することによる自動処理。
- Oracle BI Server のイベント・ポーリング・テーブルを設定することによる自動処理。
- キャッシュの記憶域領域が不足してきたときの自動処理。

Cache Manager 機能を使用してキャッシュを手動で消去するには

1. Administration Tool を使用して、リポジトリをオンライン・モードで開きます。
2. 「Manage」→「Cache」を選択して、「Cache Manager」ダイアログ・ボックスを開きます。
3. 左ペインで適切なタブを選択して、「Cache」モードまたは「Physical」モードを選択します。
4. エクスプローラ・ツリーをナビゲートして、関連するキャッシュ・エントリを右ペインに表示します。
5. 消去するキャッシュ・エントリを選択し、「Edit」→「Purge」を選択して削除します。
   - 「Cache」モードでは、右ペインに表示されているエントリから、消去するものを選択します。
   - 「Physical」モードでは、消去するデータベース、カタログ、スキーマまたはテーブルを、左ペインのエクスプローラ・ツリーから選択します。

   「Cache」モードで消去できるものは次のとおりです。

   - 開いているリポジトリに関連付けられた、1 つ以上の選択したキャッシュ・エントリ
   - 指定したビジネス・モデルに関連付けられた、1 つ以上の選択したキャッシュ・エントリ
   - ビジネス・モデル内で指定したユーザーに関連付けられた、1 つ以上の選択したキャッシュ・エントリ

   「Physical」モードで消去できるものは次のとおりです。

   - 1 つ以上の選択したデータベースに関連付けられた、すべてのテーブルのすべてのキャッシュ・エントリ
   - 1 つ以上の選択したカタログに関連付けられた、すべてのテーブルのすべてのキャッシュ・エントリ
   - 1 つ以上の選択したスキーマに関連付けられた、すべてのテーブルのすべてのキャッシュ・エントリ
   - 1 つ以上の選択したテーブルに関連付けられた、すべてのキャッシュ・エントリ

消去によって、選択したキャッシュ・エントリ、およびそれに関連付けられたメタデータが削除されます。
キャッシュの表示をリフレッシュするには、「Action」→「Refresh」を選択するか、[F5] を押します。

XML データソースのリフレッシュ間隔について

この項では、XML データソースのリフレッシュ間隔について説明します。

詳細は、「Oracle BI Server のデータソースとしての XML の使用」（345 ページ）を参照してください。

普通、XML データソースは頻繁にリアル・タイムで更新されます。XML データソースのリフレッシュ間隔設定は、データベース・テーブルのキャッシュ保持設定に類似しています。リフレッシュ間隔とは、キャッシュにある結果を使用しないで、XML データソースに対して再度クエリーが直接発行されるようになるまでの時間間隔です。リフレッシュ間隔は、「Connection Pool」ダイアログ・ボックスの「XML」タブで指定します。

デフォルトの間隔設定は「Infinite」で、XML データソースは自動的にはリフレッシュされません。

リフレッシュ間隔の設定では、Oracle BI Server の XML ゲートウェイ接続がリフレッシュされるまでの時間間隔を指定します。

- http:// または https:// で始まる URL の場合は、この間隔が経過したことが検出されると、ゲートウェイがリフレッシュされます。
- ローカル・ドライブまたはネットワーク・ドライブに存在する URL の場合は、この間隔が経過し、さらにその URL で変更が発生していることが検出されると、ゲートウェイがリフレッシュされます。

12 Oracle BI Server の接続性およびサード・パーティ製ツール

Oracle BI Server には、多くのクライアント・ツールおよびアプリケーションを介して接続できます。この章の内容は次のとおりです。

Oracle BI ODBC データ ソース名（DSN）の構成

英語以外の環境では、Oracle BI Server への直接の ODBC 接続は使用できません。英語以外の環境で Oracle BI Server に直接接続できるのは、Oracle BI Presentation Services クライアントのみです。

ここで説明する手順は、Windows ベースのオペレーティング・システムに該当します。

新しいデータ ソースを作成するには

1. 「スタート」→「設定」→「コントロール パネル」を選択し、「ODBC データ ソース」アイコンをダブルクリックして、Windows ODBC アプレットを開きます。

   Windows 2000 または XP を実行している場合は、「管理ツール」の下に「データ ソース (ODBC)」アイコンがあります。

2. 「ODBC データ ソース アドミニストレータ」ダイアログ・ボックスで、「システム DSN」タブをクリックして、「追加」をクリックします。

3. 「データ ソースの新規作成」ダイアログから Oracle BI Server ドライバを選択して、「完了」をクリックします。

   最初の「DSN Configuration」画面が表示されます。

4. 「Name」フィールドにデータ ソースの名前を入力します。

5. （オプション）「Description」フィールドに説明を入力します。

6. このデータ ソースをクラスタに参加させない場合は、画面下部にある「Server」フィールドで、Oracle BI Server が実行されているマシンを選択します。

   サーバー名がドロップダウン・リストに表示されない場合は、「Server」フィールドに名前を入力します。これは、該当するマシンの NetBIOS 名（コンピュータ名）である必要があります。

7. このデータ ソースをクラスタに参加させる場合は、次の手順を実行します。

   a. 「Is this a Clustered DSN?」オプションを選択します。

      「Primary Controller」フィールドと「Secondary Controller」フィールドがアクティブになり、「Server」フィールドは非アクティブになります。

   b. NQClusterConfig.INIファイルのパラメータPRIMARY_CONTROLLERでプライマリCluster Controllerとして指定されているマシンの名前を入力します。これは、該当するマシンの NetBIOS 名（コンピュータ名）である必要があります。

   c. NQClusterConfig.INI ファイルのパラメータ SECONDARY_CONTROLLER でセカンダリ Cluster Controller が指定されている場合は、「Secondary Controller」フィールドにそのマシンの名前を入力します。このコンピュータ名は、プライマリ Cluster Controller とは異なる一意の名前にする必要があります。

   d. Cluster Controller への接続をテストするには、「Test Connect」をクリックします。

      接続が正常にテストされたかどうかを示すメッセージが表示されます。テストが失敗した場合は、メッセージで示されているエラーを修正して、再度接続をテストします。

8 次の「DSN Configuration」画面で、データソースの接続先とするリポジトリの有効なユーザー ID とパスワードを入力します。Windows オペレーティング・システムの認証を使用している場合、このフィールドは空白のままにします。その場合は、Windows アカウントのログオン ID を使用して Oracle BI Server にログインします。

9 ログオン ID をリポジトリに保存する場合は、「Save login ID」オプションを選択します。

このオプションを選択すると、接続するたびにログオン情報を入力する必要がなくなります。

10 「Port」フィールドに、Oracle BI Server でクライアント / サーバー通信に使用している TCP/IP ポートを指定します。

デフォルトのポートは 9703 です。このポート番号は、NQSConfig.INI ファイルの Server セクションのパラメータ RPC_SERVICE_OR_PORT で指定されているポート番号と一致させる必要があります。構成ファイルのポート番号を変更する場合は、その新しいポート番号を使用するように、影響を受けるすべての ODBC データソースを再構成してください。

注意： Oracle BI Server のデフォルトのクライアント / サーバー通信方法は、分散コンポーネント・オブジェクト・モデル（DCOM）から TCP/IP に変更されました。DCOM のサポートは、今後のリリースで中止される予定です。すでに Oracle BI Server を実行しているサイトで、サポートの中止まで引き続き DCOM を使用する場合は、このフィールドをデフォルト値のままにし、NQUIRE_DCOM という名前の Windows システム環境変数を定義して、DCOM を強制的に使用するようにします。この変数の値は 1 に設定します（システム環境変数を定義するには、「コントロール パネル」の「システム」を選択し、「詳細設定」タブをクリックし、「環境変数」ボタンをクリックして「環境変数」ダイアログ・ボックスを開きます）。

11 デフォルト以外のリポジトリに接続する場合は、「Change the default repository to」オプションを選択し、接続先のリポジトリの論理名を、NQSConfig.INI ファイルに表示されているとおりにチェック・ボックスの下のフィールドに入力します。

このオプションを選択しない場合、データソースは、NQSConfig.INI ファイルでデフォルトのリポジトリとしてマークされているリポジトリに接続します。デフォルトとしてマークされているエントリがない場合は、リストの最初にあるリポジトリに接続します。

12 追加構成のデフォルト設定を取得するには、「Connect to Oracle BI Server」オプションを選択します。

サーバーに接続し、リポジトリ内のビジネス・モデルに関する情報が取得されます。このオプションを選択しなくても、次の構成画面に情報を手動で入力することにより、DSN を構成できます。

13 「Next」をクリックして次のウィンドウに進みます。

14 デフォルトのカタログを変更するには、「Change the default catalog to」オプションを選択して、チェック・ボックスの下のフィールドにカタログの名前を入力します。

デフォルトのカタログは、Administration Tool の「Presentation」レイヤーの最上位に表示されるカタログ・フォルダです。Oracle BI Presentation Services で使用される DSN については、このチェック・ボックスを選択せず、ドロップダウン・ボックスにエントリが何も表示されない状態にしておくことをお薦めします。

Oracle BI Server の接続先の、基礎となるデータベースのユーザー ID とパスワードも指定できます。データベースのユーザー ID とパスワードを指定すると、これらがデータベースへの接続に使用されます（「接続プールの作成または変更」（72 ページ）で説明しているように、ユーザー固有のデータベース・ログオン情報が接続プールで構成されている場合）。データベース固有のユーザー ID とパスワードを使用すると、特権ユーザーは、データベースで付与されている認可レベルに基づいて基礎となるデータベースに接続できます。

15 この時点で、DSN のログインに使用する Oracle BI ユーザーのパスワードを変更できます（サーバーが書込み可能モードで実行されている場合）。パスワードを変更するには、前の画面で、ログオン情報を入力し、「Connect to Oracle BI」オプションを選択している必要があります。新しいパスワードは、暗号化された形式でリポジトリに格納されます。

ODBC 準拠レベル

Oracle BI Server では、クライアント・アプリケーションからの次の ODBC コールがサポートされています。

- SQLAllocConnect
- SQLAllocEnv
- SQLAllocStmt
- SQLBindCol
- SQLCancel
- SQLColumns
- SQLConnect
- SQLDescribeCol
- SQLDisconnect
- SQLDriverConnect
- SQLError
- SQLExecDirect
- SQLExecute
- SQLExtendedFetch
- SQLFetch
- SQLFreeConnect
- SQLFreeEnv
- SQLFreeStmt
- SQLGetConnectOption
- SQLGetCursorName
- SQLGetData
- SQLGetFunctions
- SQLGetInfo
- SQLGetStmtOption
- SQLGetTypeInfo
- SQLColAttributes

- SQLNumResultCols
- SQLPrepare
- SQLRowCount
- SQLSetConnectOption
- SQLSetStmtOption
- SQL Tables

Oracle BI ODBC では、静的、動的、転送専用、およびキー・セット・ドリブンの、完全なスクロール可能カーソルがサポートされています。

Oracle BI ODBC では、非同期および同期の処理と取消がサポートされています。

サード・パーティ製ツールとリレーショナル・データソース・アダプタ

Oracle BI Server では、様々なクライアント・ツールやデータソースの間での接続が可能です。詳細は、『Oracle Business Intelligence Suite Enterprise Edition システム要件およびサポートされるプラットフォーム』を参照してください。

メタデータのインポート

メタデータをデータソースから Oracle BI リポジトリにインポートできます。メタデータは、Administration Tool の「Physical」レイヤーで物理テーブル情報を確立するために使用します。

Oracle BI リポジトリへのメタデータのインポートは、基礎となるデータソースへの ODBC 接続またはネイティブのデータベース接続を使用して行う必要があります。メタデータは、Microsoft Excel などのソフトウェアから ODBC 接続を介してインポートすることもできます。

メタデータのインポート手順については、「リレーショナル・データソースから物理レイヤーを作成するプロセス」（60 ページ）を参照してください。

クエリー・ツールおよびレポート・ツールの使用

Oracle BI Server へは、ODBC 準拠の様々なクエリー・ツールやレポート・ツールを使用して接続できます。クエリー・ツールで接続するには、Oracle BI ODBC ドライバを使用してデータソースを構成してから、Oracle BI DSN を使用してクエリー・ツールからリポジトリへ接続します。

「Presentation」レイヤーを使用すると、ツールのルールおよび規則に一致するようにビジネス・モデルのプレゼンテーションを構成して、Oracle BI Server の分析エンジンやデータ抽象化の機能を利用できます。これにより、複雑な集計や計算ルールを使用する列を、クエリーやレポートに簡単に含めることができます。また組織内でクエリー・ツールやレポート・ツールを使用している場合は、Oracle BI Server をデータソースとして使用すると、これらのツールの価値が高まり、ツール使用時の作業を簡略化できます。

データベースとのメタデータの交換

Oracle Database または IBM DB2 データベースをインストールしている組織では、これらのデータベースを使用して、データ・ウェアハウスのパフォーマンスを向上したり、Oracle BI Server で実行されるクエリー機能を強化したりできます。Oracle BI Server の Oracle BI メタデータを Oracle Database または IBM DB2 データベースと交換することにより、次のツールを使用して、データベースでのデータ・ウェアハウス・クエリーのパフォーマンスを高速化できます。

- Oracle Database では、Oracle Database サマリー・アドバイザを使用して、マテリアライズド・ビューおよびパフォーマンス最適化に関する索引の推奨が作成されます。
- IBM DB2 データベースでは、IBM DB2 Cube Views を使用してマテリアライズ照会表（MQT）が作成されます。

どちらのツールでも、リレーショナル・データが事前に集計され、クエリーのパフォーマンスが向上します。

メタデータの交換に関する情報の参照

メタデータの交換に関する情報については、次の各項で説明しています。

- Oracle BI での Oracle Database のマテリアライズド・ビューの使用（283 ページ）
- Oracle BI での IBM DB2 Cube Views の使用（287 ページ）

「インポート・ファイルの生成」では、両方のデータベースに該当する情報について示します。

インポート・ファイルの生成

Oracle Database Metadata Generator および DB2 Cube Views Generator の両方で、Oracle BI から Oracle Database サマリー・アドバイザまたは IBM DB2 データベースへメタデータをインポートするために必要なファイルが作成されます。この項を読む前に、「メタデータの交換に関する情報の参照」（272 ページ）を読んでください。

この項は、2 つのジェネレータに共通する次のトピックで構成されています。

- ジェネレータの実行（272 ページ）
- メタデータ入力ファイルについて（273 ページ）
- 出力ファイルについて（274 ページ）
- ジェネレータからのエラーのトラブルシューティング（275 ページ）
- メタデータの変換ルールおよびエラー・メッセージ（275 ページ）

ジェネレータの実行

Oracle Database Metadata Generator および DB2 Cube Views Generator は、コマンドラインから起動されるか、またはバッチ・ファイルに埋め込まれています。コマンドライン実行可能ファイルの名前は SAMetaExport.exe で、構文は次のとおりです。

```
SAMetaExport -r "PathAndRepositoryFileName" -u <UserName> -p <Password>
-f "InputFileNameAndPath" [-t "ORACLE" or "DB2"]
```

表 30 に、このコマンドライン実行可能ファイルのパラメータの説明を示します。

表 30. SAMetaExport.exe のパラメータ

パラメータ	定義	追加情報
-r	リポジトリ・ファイル名およびフルパス	ファイル・パスの形式が長いかパスに空白が含まれる場合のみ、ファイル名とパスを引用符で囲む必要があります。ファイルが現在のディレクトリに存在しない場合は、フルパスを使用します。
-u	ユーザー名	リポジトリへのアクセスが可能なユーザー名。
-p	パスワード	ユーザー名のパスワード。リポジトリ・パスワードが空の場合、パスワード・パラメータは使用しないでください。
-f	入力ファイル名およびフルパス	ファイル・パスの形式が長いかパスに空白が含まれる場合のみ、ファイル名とパスを引用符で囲む必要があります。ファイルが現在のディレクトリに存在しない場合は、フルパスを使用します。入力ファイルを指定すると、必要な情報すべてをコマンドラインに入力する必要がなくなります。また各国語の文字を入力できます。入力ファイルの詳細は、「メタデータ入力ファイルについて」（273 ページ）を参照してください。
-t	ORACLE または DB2	Oracle Database または IBM DB2 データベースを指定します。データベースのタイプはリポジトリ内の情報に基づいて自動的に検出されるため、このパラメータは省略できます。

メタデータ入力ファイルについて

入力ファイルは、表 31 で説明するパラメータを含むテキスト・ファイルです。

表 31. キューブ・メタデータ入力ファイルのパラメータ

入力ファイル名	説明
BUSINESS_MODEL	エクスポートするメタデータを含む、Oracle BI リポジトリの論理レイヤーにあるビジネス・モデルの名前。リポジトリ内にビジネス・モデルが見つからない場合は、エラー・メッセージが表示されます。
PHYSICAL_DATABASE	エクスポートするメタデータを含む、Oracle BI リポジトリの物理レイヤーにあるデータベースの名前。ビジネス・モデルが複数のデータベースから導出されている場合は、ここで指定した以外のすべてのデータベースからのメタデータが削除されます。リポジトリ内に物理データベースが見つからない場合は、エラー・メッセージが表示されます。

表 31. キューブ・メタデータ入力ファイルのパラメータ

入力ファイル名	説明
RUN_AS_USER	メタデータのエクスポート用に重複した可視性を持つ必要のあるユーザーのユーザー名。このパラメータは必ず指定してください。リポジトリ内にユーザーが見つからない場合は、エラー・メッセージが表示されます。
OUTPUT_FOLDER	SQL ファイルの書込み先フォルダのフルパスおよびファイル名。このフォルダが Oracle Database Metadata Generator の実行時に存在しない場合は作成されます。出力ファイルの詳細は、「出力ファイルについて」（274 ページ）を参照してください。

メタデータ入力ファイルのサンプルは次のとおりです。

```
BUSINESS_MODEL= "Paint"

PHYSICAL_DATABASE    ="SQL_Paint"

RUN_AS_USER =     "Administrator"

OUTPUT_FOLDER = "C:¥OracleBI"
```

出力ファイルについて

次に説明するように、作成される出力ファイルのタイプはジェネレータごとに異なります。

- Oracle Database Metadata Generator: UTF8 でエンコードされ、指定した出力フォルダに格納される SQL ファイルを生成します。ファイル名には、my_business_model.sql のように、キューブ・モデル内のビジネス・モデルの名前が含まれます。
- DB2 Cube Views Generator: 指定した出力フォルダに次のファイルを生成します。
  - **XML ファイル（UTF8 でエンコード）**: 指定したビジネス・モデルごとに XML ファイルが 1 つ作成されます。このファイルには、キューブに変換されたすべてのオブジェクトが含まれます。また、リポジトリ内のオブジェクトは、IBM Cube Views メタデータ内の同様のオブジェクトにマップされます。変換されないオブジェクトのリストは、「メタデータの変換ルールおよびエラー・メッセージ」（275 ページ）を参照してください。

    XML ファイルの名前はビジネス・モデル名（空白なし）と一致し、XML 拡張子が付きます。たとえば、SalesResults.xml などです。XML ファイル名の構文は次のとおりです。

    [空白を除いたビジネス・モデル名].xml

  - **別名生成 DLL を含む SQL ファイル**: 指定したビジネス・モデルで参照されている物理レイヤーのデータベースに別名が存在する場合のみ、指定したビジネス・モデルごとに SQL ファイルが 1 つ作成されます。別名ファイルに含まれる SQL コマンドによって、DB2 データベースに別名が作成されます。SQL ファイルの名前はビジネス・モデル名（空白なし）と一致し、SQL 拡張子が付きます。たとえば、SalesResults-alias.sql などです。別名 SQL ファイル名の構文は次のとおりです。

    [空白を除いたビジネス・モデル名]-alias.sql

ジェネレータからのエラーのトラブルシューティング

エラー・メッセージは、ジェネレータで一部またはすべてのタスクを完了できなかったことを示します。ジェネレータの起動後に、次のエラー・メッセージが表示される場合があります。

- Unable to write to Log file: <log_file_name>.

  NQSConfig.INI ファイルで指定されているログ・ファイルに誤ったパスが含まれているか、ユーザーにそのフォルダに対する書込み権限がないか、またはディスク容量が不足しています。

- Run_as_user, <user_name>, is invalid.

  ユーザー名が正しくありません。

- Repository, <repository_name>, is invalid or corrupt.

  リポジトリ名が正しくないか、指定されたパスにリポジトリが存在しないか、またはユーザーにリポジトリの読取り権限がありません。

- Physical Database, <database_name>, is invalid.

  物理データベース名が正しくありません。

- Business Model, <model_name>, is invalid.

  ビジネス・モデル名が正しくありません。

- Authentication information provided is invalid.

  指定したユーザー名とパスワードが正しくありません。

- Path: "<path_name>" is invalid.

  パスまたはファイル名が正しくありません。

メタデータの変換ルールおよびエラー・メッセージ

ジェネレータで出力ファイルを作成する際、Oracle BI リポジトリのメタデータ・オブジェクトは、Oracle Database または IBM DB2 データベースのメタデータの同様のオブジェクトにマップされます。

この項では、SQL、XML のどちらの形式にも変換できない Oracle BI メタデータを識別する際のルールについて説明します。このルールは、Oracle BI で許可されているメタデータ構造の一部が、Oracle Database および IBM Cube Views ではサポートされないために必要になります。

SQL ファイルまたは XML ファイルのディメンション・メタデータは、論理ファクト・テーブル・ソース・レベルで生成されます。論理ファクト・テーブル・ソースに無効な論理ディメンション・テーブル・ソースがある場合、この論理ディメンション・テーブル・ソースは無効になります。論理ファクト・テーブル・ソースが無効な場合は、それにリンクするすべての論理ディメンション・テーブル・ソースも無効になります。無効な Oracle BI リポジトリ・メタデータ要素は、SQL ファイルや XML ファイルでキューブに変換されません。

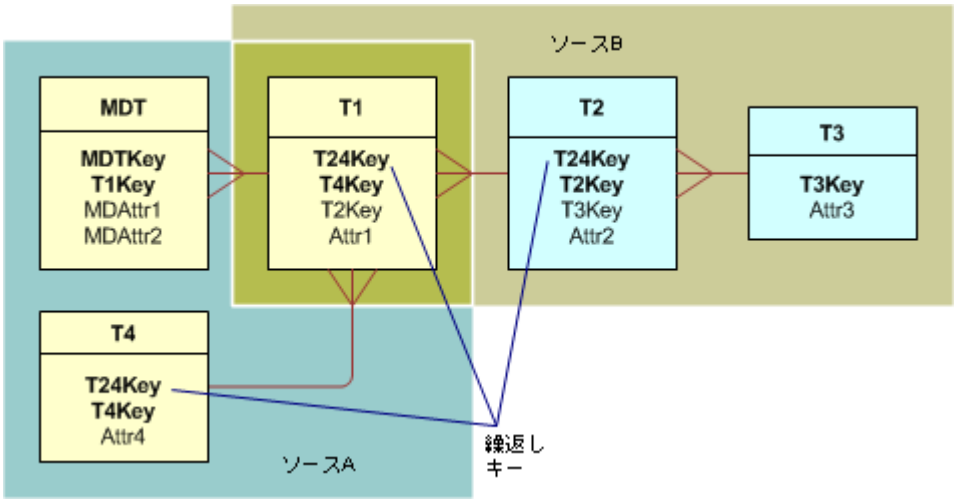
ルール違反が発生すると、ルールに違反しているメタデータがエラー・メッセージとともにログ・ファイルに書き込まれます。このログ・ファイルの名前は、NQSConfig.INI ファイルの LOG_FILE_NAME パラメータで指定します。NQSConfig.INI ファイルの Cube Views セクションにあるパラメータの詳細は、『Oracle Business Intelligence Infrastructure インストレーションおよび構成ガイド』を参照してください。

Oracle Database の変換ルール

Oracle BI メタデータを Oracle Database のオブジェクトに変換するためのルールは次のとおりです。

- 論理テーブルに式を含む属性はエクスポートできません。
- 複合結合を使用して結合されたテーブルは考慮されません。
- 不明瞭ビューのテーブルは考慮されません。
- あるレベルでキーの一部として使用されている列は、別レベルのキーの一部としては使用できません。例として、ソース A およびソース B というディメンションが含まれる、次に示すリポジトリの図について考えます。ソース A ディメンションは、T1 テーブルをソース B と共有しています。T24Key は、T2 テーブルへの結合（T24Key と T2Key）、および T4 テーブルへの結合（T24Key と T4Key）の両方で使用されています。

  Oracle Database では、列を複数レベルのキーとして使用することは禁止されています。この禁止ルールにより、Oracle Database Metadata Generator では、2 つの結合のいずれかを削除する必要があります。通常は最初に見つかった結合が削除されます。したがって、T1-T4 結合が最初に見つかったため、T2 および T3 への結合は失われ、テーブル T2 および T3 の Attr2 属性と Attr3 属性はエクスポートされません。



IBM DB2 データベースの変換ルール

表 32 に、Oracle BI リポジトリのメタデータ要素の検証に使用するルール、ルール違反が発生した場合にログ・ファイルに書き込まれるエラー・メッセージ、およびルール違反の原因の説明を示します。エラー・メッセージは、特定の Oracle BI メタデータ・オブジェクトが XML ファイルにエクスポートされなかった理由を特定する際に役立ちます。

表 32. メタデータ要素の検証ルール

ルール	メッセージ	説明
ComplexJoinFactsRule	[Fact Logical Table Source]Complex Physical Joins not supported %qn has a complex Join %qn between Physical Tables %qn and %qn	物理ファクト・テーブルが複合結合で接続されている場合、この結合はサポートされません。複合結合は、外部キーのリレーションシップを持たない 2 つのテーブル間の結合として定義されます。
ComplexJoinDimsRule	[Dimension Logical Table Source]Complex Physical Joins not supported %qn has a complex Join %qn between Physical Tables %qn and %qn	ディメンション物理テーブルが複合結合で接続されている場合、この結合はサポートされません。
ComplexJoinFactDimRule	[Fact Logical Table Source -> Dimension Logical Table Source] Complex Physical Joins not supported. %qn has a complex Join %qn between Physical Tables %qn and %qn.	ディメンション物理テーブルとファクト物理テーブルが複合結合で接続されている場合、この結合はサポートされず、ディメンション・テーブル・ソースは無効になります。
OpaqueViewFactRule	[Fact Logical table Source] Physical SQL Select Statements not supported. %qn uses the SQL Select Statement %qn.	物理ファクト・テーブルが SQL の SELECT 文によって生成される場合、このテーブルを含む論理ファクト・テーブル・ソースは無効になります。この論理ファクト・テーブル・ソースに接続されている論理ディメンション・テーブル・ソースもすべて無効になります。この構造では、サブクエリー処理が可能です。
OpaqueViewDimRule	[Dimension Logical table Source] Physical SQL Select Statements not supported. %qn uses the SQL Select Statement %qn.	物理ディメンション・テーブルが SQL の SELECT 文によって生成される場合、このテーブルを含む論理ディメンション・テーブル・ソースは無効になります。

表 32. メタデータ要素の検証ルール

ルール	メッセージ	説明
OuterJoinFactRule	[Fact Logical Table Source] Physical Outer Joins not supported. %qn has an outer join %qn between physical tables %qn and %qn.	論理ファクト・テーブル・ソースに外部結合リンクが含まれている場合、この論理ファクト・テーブル・ソースは無効になり、このソースにリンクされている論理ディメンション・テーブル・ソースもすべて無効になります。
OuterJoinDimRule	[Dimension Logical Table Source] Physical Outer Joins not supported. %qn has an outer join %qn between physical tables %qn and %qn.	論理ディメンション・テーブル・ソースに外部結合リンクが含まれている場合、この論理ディメンション・テーブル・ソースは無効になります。
WhereClauseFactRule	[Fact Logical Table Source] WHERE clauses are not supported. %qn has a where condition %s.	ファクト・テーブル・ソースで、ロードされるデータを WHERE 句を使用してフィルタ処理する場合、このテーブル・ソースは無効になります。
WhereClauseDimRule	[Dimension Logical Table Source] WHERE clauses are not supported. %qn has a where condition %s.	ディメンション・テーブル・ソースで、ロードされるデータを WHERE 句を使用してフィルタ処理する場合、このテーブル・ソースは無効になります。
TwoJoinFactDimRule	[Fact Logical Table Source -> Dimension Logical Table Source] Multiple Joins between sources not supported. %qn and %qn have at least the following joins : %qn, %qn.	1 つの物理ファクト・テーブルが、同じディメンション・ソースの 2 つのディメンション・テーブルにリンクされている場合（ファクト・テーブルが、テーブル・ソースの最も詳細なテーブルに排他的にリンクされていない場合）、ディメンション・テーブル・ソースは無効になります。

表 32. メタデータ要素の検証ルール

ルール	メッセージ	説明
HiddenManyManyRule	[Fact Logical Table Source -> Dimension Logical Table Source] Join between (physical or logical?) fact and dimension is not on the most detailed table. %qn between %qn and %qn is not on the most detailed table %qn {Join name, facttable, dimtable).	これは TwoJoinFactDimRule に関連しています。ファクト・テーブルが、テーブル・ソース内の最も詳細なテーブルでないディメンション・テーブルに結合されていると、このディメンション・テーブル・ソースは無効になります。
ComplexMeasureRule	[Column] Complex Aggregation Rules not supported. %qn uses an aggregation rule of %s which is not supported.	サポートされている集計は、SUM、COUNT、AVG、MIN、MAX、STDDEV、COUNT-DISTINCT および COUNT です。
CountDistMeasureRule	[Column] COUNT-DISTINCT Aggregation Rule not supported. %qn uses an aggregation rule of %s which is not supported.	COUNT-DISTINCT 集計はサポートされていません。
InvalidColumnLevelRule	[Level] Some columns that are part of the Primary Level Key are invalid. %qn has %qn as part of its primary key, when %qn has already been marked invalid.	COUNT-DISTINCT 集計はサポートされていません。
VariableBasedColumnRule	[Logical Table Source -> Column] Column uses a Variable in the Expression Column %qn uses a variable in its mapping.	COUNT-DISTINCT 集計はサポートされていません。論理列では、式にリポジトリ変数およびセッション変数を使用します。

表 32. メタデータ要素の検証ルール

ルール	メッセージ	説明
OneFactToManyDimRule	[Fact Logical Table Source -> Dimension Logical Table Source] There must be a unique join path between the most detailed tables in the (logical or physical?) fact and the dimension. No join paths found between %qn and %qn (both physical table names). Found at least the following join paths: (%qn->%qn....), (%qn->%qn....)	TwoJoinFactDimRule または HiddenManyManyRule と同じです。
ManyMDTinFactRule	[Fact Logical Table Source] Fact Logical Table Source must have a unique most detailed table. %qn has at least the following most detailed tables : %qn,%qn.	ファクト・ソースに最も詳細なテーブルが複数あります。
NoMeasureFactRule	[Fact Logical Table Source] Fact Logical Table Source does not have any Measures. %qn does not have any deployable measures.	すべてのメジャーが無効になっているため、ファクト・テーブルにメジャーがありません。
NoInActiveFactRule	[Fact Logical Table Source] Fact Logical Table Source is not marked Active.	ファクト・ソースがアクティブではありません。
NoInActiveDimRule	[Dimension Logical Table Source] Fact Logical Table Source is not marked Active.	ディメンション・ソースがアクティブではありません。
NoAttributeInFactRule	[Fact Logical Table Source -> Column] Attribute found in Fact. %qn in a fact source %qn does not have an aggregation rule.	ファクト・ソース内に属性がありません。

表 32. メタデータ要素の検証ルール

ルール	メッセージ	説明
NoMeasureInDimRule	[Dimension Logical Table Source -> Column] Measure found in Dimension. %qn in a dimension source %qn has an aggregation rule.	ディメンション・ソース内にメジャーがありません。
VisibleColumnsAttrRule	[Column] -> The run_as_user does not have visibility to this Logical Column. %qn is not accessible to the run_as_user %qn due to visibility rules.	列にこのユーザーの可視性がありません。
VisibleColumnsMeasRule	[Column] -> The run_as_user does not have visibility to this Logical Column. %qn is not accessible to the run_as_user %qn due to visibility rules.	列にこのユーザーの可視性がありません。
MultiplePrimaryKeysDimRule	[Dimension Logical Table Source] A Join uses an alternate key in the Dimension Logical Table Source. %qn between %qn and %qn in %qn uses the alternate key %qn.	ディメンション物理テーブルに含めることができる主キーは 1 つだけです。異なる一意なキーを使用して別のディメンション物理テーブルに結合されているため、その結合は無効です。 IBM Cube Views は、外部結合に使用される一意のキーを受け入れません。常に主キーを使用する必要があります。
MultiplePrimaryKeysFactRule	[Dimension Logical Table Source] A Join uses an alternate key in the Dimension Logical Table Source. %qn between %qn and %qn in %qn uses the alternate key %qn.	ファクト物理テーブルに含めることができる主キーは 1 つだけです。異なる一意なキーを使用して別のファクト物理テーブルに結合されているため、その結合は無効です。 IBM Cube Views は、外部結合に使用される一意のキーを受け入れません。常に主キーを使用する必要があります。

表 32. メタデータ要素の検証ルール

ルール	メッセージ	説明
MultiplePrimaryKeysFactDimRule	[Fact Logical Table Source -> Dim Logical Table Source] A Join uses an alternate key between the Logical Table sources. %qn between %qn and %qn for sources %qn and %qn uses the alternate key %qn.	ファクト物理テーブルに含めることができる主キーは 1 つだけです。異なる一意なキーを使用してディメンション物理テーブルに結合されているため、その結合は無効です。 IBM Cube Views は、外部結合に使用される一意のキーを受け入れません。常に主キーを使用する必要があります。
NotDB2ExpressionAttrRule	[Dimension Logical Table Source -> Column] The Column contains an Expression not supported. %qn has expression %s which is not supported.	IBM Cube Views でサポートされていない式が属性に含まれています。 これには、DateTime 関数（たとえば、CURRENT_DATE）を使用するメタデータ式などがあります。
NotDB2ExpressionMeasRule	[Fact Logical Table Source -> Column] The Column contains an Expression not supported. %qn has expression %s which is not supported.	IBM Cube Views でサポートされていない式がメジャーに含まれています。 これには、DateTime 関数（たとえば、CURRENT_DATE）を使用するメタデータ式などがあります。
NoAttributeDimRule	[Dimension Logical Table Source] Dimension Logical Table Source does not have any attributes visible to the run_as_user. %qn can not be queried by user %qn since none of its attributes are visible.	ディメンションに属性がありません。

Oracle BI での Oracle Database のマテリアライズド・ビューの使用

この項では、Oracle BI から Oracle Database サマリー・アドバイザへメタデータをエクスポートし、Oracle Database Metadata Generator を使用してマテリアライズド・ビューを作成する方法について説明します。この項を読む前に、「データベースとのメタデータの交換」（272 ページ）を読んでください。

この項の内容は次のとおりです。

Oracle Database サマリー・アドバイザでのマテリアライズド・ビューの使用について

この機能により、データ・ウェアハウスのパフォーマンスやデータベースの機能性が向上します。Oracle Database サマリー・アドバイザを使用すれば、データベース内に存在するデータの論理的なリレーションシップに関するメタデータを格納できます。また、効率的な Oracle のマテリアライズド・ビューを使用することにより、データ・ウェアハウス・クエリーの処理を高速化できます。これらのマテリアライズド・ビューでは、リレーショナル・データが事前に集計され、サブクエリーのパフォーマンスが向上します。Oracle Database サマリー・アドバイザにメタデータが格納されると、データベース管理者は、データベース・オブジェクトを最適化してサブクエリーのパフォーマンスを向上できます。

Oracle Database でクエリーを処理する場合、可能であれば、そのクエリーはマテリアライズド・ビューを含むテーブルにルーティングされます。マテリアライズド・ビューのテーブルは基礎となる実テーブルよりも小さく、そのデータは事前集計されているため、マテリアライズド・ビューに再ルーティングすることでクエリー時間が短縮される場合があります。

Oracle Database Metadata Generator はメタデータ・ブリッジの役割を果たし、Oracle BI 独自のメタデータを、Oracle Database サマリー・アドバイザでディメンションを生成するための PL/SQL コマンドを含む SQL ファイルに変換します。メタデータを SQL ファイルに変換したら、SQL*Plus などのツールを使用して、変換されたメタデータを Oracle Database サマリー・アドバイザにインポートし、メタデータ・カタログ・テーブルに格納します。メタデータをインポートしたら、マテリアライズド・ビューを作成します。マテリアライズド・ビューは、受信するアプリケーションのクエリーを最適化するために使用します。

この機能は、Oracle Database9*i* 以降で使用できます。プラットフォームの互換性の詳細は、『Oracle Business Intelligence Suite Enterprise Edition システム要件およびサポートされるプラットフォーム』を参照してください。

Oracle でのメタデータのデプロイ手順

注意： Oracle Database でメタデータをデプロイする前に、Oracle Database とそのツールについてよく理解しておいてください。詳細は、Oracle Database のドキュメントを参照してください。

メタデータをデプロイする前に、「インポート・ファイルの生成」（272 ページ）の手順を完了しておいてください。キューブ・メタデータをデプロイするには、ここに示す順番で次の手順を実行してください。

Oracle 用 SQL ファイルの実行

この手順は、「Oracle でのメタデータのデプロイ手順」（283 ページ）の一部です。

Oracle Database サマリー・アドバイザへインポートするための SQL ファイルを実行する前に、Oracle Database のインポート・ツールについてよく理解してください。詳細は、Oracle Database のドキュメントを参照してください。

Oracle Database Metadata Generator で生成された SQL ファイルを実行するには、SQL*Plus などのツールを使用します。ディメンションがすでに存在する場合や、データベース・スキーマが RPD ファイルと異なる場合は、エラー・メッセージが表示されることがあります。スクリプトが正常に実行されると、データベース Web コンソールまたは Oracle Enterprise Manager コンソールを使用して、作成されたディメンションを確認できます。Oracle Enterprise Manager コンソールで、「ネットワーク」、「データベース」、「*データベース名*」、「ウェアハウス」、「サマリー管理」、「ディメンション」、「システム」の各ノードを開きます。

SQL ファイルの実行後、次の処理が必要な場合があります。

- メタデータの増分変更はできません。スキーマを変更するには、Oracle Database でキューブ・モデル・メタデータを手動で削除して、Oracle BI メタデータを再度変換する必要があります。たとえば、Oracle BI メタデータ・リポジトリ内のキューブでディメンションを変更する場合は、Oracle Database でキューブ・モデルを削除し、Oracle BI リポジトリから SQL ファイルを再生成して、この SQL ファイルを Oracle Database サマリー・アドバイザにインポートする必要があります。
- Oracle Database Metadata Generator を使用してメタデータを削除することはできません。Oracle BI 管理者は、Oracle Enterprise Manager コンソールを使用してキューブ・モデルを手動で削除する必要があります。

結合の存在に関する制約の定義

この手順は、「Oracle でのメタデータのデプロイ手順」（283 ページ）の一部です。詳細は、Oracle Database のドキュメントを参照してください。

ディメンション・テーブルとファクト・テーブル間の結合が Oracle Database で認識されることを確認する必要があります。これを行うには、SQL*Plus または Oracle Enterprise Manager コンソールで制約を作成します。Oracle Enterprise Manager コンソールで制約対象となるテーブルを選択して、「制約」タブを選択します。

作成する制約のタイプは、テーブルによって次のように異なります。

- ディメンション・テーブルには、UNIQUE キー制約を作成します。
- ファクト・テーブルには、FOREIGN キー制約を作成して、参照先のスキーマおよびテーブルを指定します。「制約定義」領域で、外部キー列をファクト・テーブルに含め、対応する一意キーをディメンション・テーブルに含めます。外部キー列のデータがディメンション・テーブルの一意キー列のデータと一致しない場合、ディメンション・テーブルでの外部キーの作成が失敗することがあります。

クエリー・ワークロードの作成

この手順は、「Oracle でのメタデータのデプロイ手順」（283 ページ）の一部です。詳細は、Oracle Database のドキュメントを参照してください。

クエリー・ワークロードは、最適化を必要とする物理クエリーのサンプル・セットです。ワークロードを作成する前に、実行速度の最も遅いクエリーに関する情報を使用して、トレース・ファイルを生成します。

トレース・ファイルを生成するには

実行速度の最も遅いクエリーのトレース・ファイルを生成するには、次の説明に従って、データベースのバージョンに適したツールを使用します。

- Usage Tracking: Oracle BI でこの機能を使用すると、クエリーとその実行時間をログに記録できます。その後、長時間実行される Oracle BI のクエリーをスクリプトとして実行し、Oracle Database のトレース機能をともに使用すれば、これらのクエリー用の Oracle Database SQL コードを取得できます。
- Oracle Database Trace: このツールを使用すると、最も遅い物理クエリーを特定できます。このトレース機能は、Oracle Enterprise Manager Database Control 内で、または DBMS_MONITOR パッケージを含む SQL コマンドを入力することによって有効化できます。このトレース機能を有効にしたら、スクリプトを使用してトレース・ファイルを作成し、クエリー・ワークロード・テーブル内のクエリー用の SQL コードを取得します。
- Oracle Enterprise Manager: このツールを使用すると、実行速度の遅いクエリーをトラッキングできます。

ヒント： 次の各項で説明する機能は、Oracle BI の一部としてではなく、Oracle Database で使用可能です。

トレース・ファイルの情報を分析するには

1. トレース・ファイルを確認する場合は、次のガイドラインに従ってください。
   - バッチ・プロセスなどで多くの文を一度にトレースした場合は、クエリーの実行時間が許容範囲内である文をすぐに破棄します。実行時間が最も長い文に注目してください。
   - ブロック参照の Query 列で、すべてのクエリーおよびサブクエリーの処理を含めた読取り一貫性をチェックします。非効率な文には、多くの場合、多数のブロック参照が関連付けられています。Current 列には、更新されるセグメント・ヘッダーやブロックを含む、読取り一貫性に関係のない参照が示されます。
   - Disk 列で、ディスクから読み取られたブロック数をチェックします。ディスクの読取りにはメモリーの読取りより時間がかかるため、この値は、Query 列と Current 列の合計を大きく下回る傾向があります。そうでない場合は、バッファ・キャッシュに問題がないかどうかチェックしてください。
   - ロックの問題がある場合や非効率な PL/SQL ループでは、ブロック参照の数が少なくても、CPU 時間の値が大きくなることがあります。
   - 単一の文で複数の解析コールが行われる場合は、ライブラリ・キャッシュに問題があります。
2. ファイル内で問題のある文を特定したら、実行計画をチェックして、問題の発生理由を確認してください。

クエリーをワークロードにロードするには

- トレース・ユーティリティを使用して最も遅い物理クエリーの名前を確認したら、それらを USER_WORKLOAD テーブルに挿入します。

  286 ページの表 33 に、USER_WORKLOAD テーブルの列について説明します。

- INSERT 文を使用して、最も遅い物理クエリーの SQL 文を QUERY 列に移入し、適切な所有者名を OWNER 列に移入します。

表 33. USER_WORKLOAD テーブルの列

列	データ型	必須かどうか	説明
QUERY	LONG または VARCHAR（すべてのキャラクタ・タイプ）	はい	クエリーの SQL 文。
OWNER	VARCHAR2 (30)	はい	クエリーを最後に実行したユーザー。
APPLICATION	VARCHAR2 (30)	いいえ	クエリーのアプリケーション名。
FREQUENCY	NUMBER	いいえ	クエリーが実行された回数。
LASTUSE	DATE	いいえ	クエリーが最後に実行された日付。
PRIORITY	NUMBER	いいえ	ユーザーが指定したクエリーのランキング。
RESPONSETIME	NUMBER	いいえ	クエリーの実行時間（秒）。
RESULTSIZE	NUMBER	いいえ	クエリーで選択されたバイトの合計数。
SQL_ADDR	NUMBER	いいえ	クエリーのキャッシュ・アドレス。
SQL_HASH	NUMBER	いいえ	クエリーのキャッシュ・ハッシュ値。

マテリアライズド・ビューの作成

この手順は、「Oracle でのメタデータのデプロイ手順」（283 ページ）の一部です。

クエリー・ワークロード・テーブルを作成したら、Oracle Database のバージョンに適したツールを使用してマテリアライズド・ビューを作成します。Oracle Database 10*g* では、Oracle Enterprise Manager コンソールでサマリー・アドバイザを使用して、作成したクエリー・ワークロード・テーブルを指定します。

サマリー・アドバイザでは、指定したファクト・テーブルのパフォーマンスを向上させるための推奨事項が生成されます。サマリー・アドバイザには、適切なマテリアライズド・ビューを作成する SQL コードが表示されます。サマリー・アドバイザでマテリアライズド・ビューを作成する前に、次のヒントを確認してください。

- SQL コードに CAST 文が含まれている場合、マテリアライズド・ビューの作成が失敗することがあります。
- CREATE MATERIALIZED VIEW 文に、ワークロード・テーブルとして指定した同じクエリーが指定されていないことを確認してください。同じクエリーが指定されていると、マテリアライズド・ビューによるパフォーマンスの向上が実際には反映されません。ただし、このクエリーを頻繁に実行する場合は、マテリアライズド・ビューを作成する価値がある場合もあります。

- SQL 文の後の CREATE MATERIALIZED VIEW 文の最後にスラッシュ（/）を追加してください。そうしないと、SQL*Plus ワークシートで有効な文として認識されません。

ヒント： SQL アクセス・アドバイザも、適切な索引付けスキームの特定に役立つ場合があります。

Oracle BI での IBM DB2 Cube Views の使用

この項では、DB2 Cube Views Generator を使用して、Oracle BI から IBM DB2 データベースへメタデータをエクスポートする方法について説明します。この項を読む前に、「データベースとのメタデータの交換」（272 ページ）を読んでください。

この項の内容は次のとおりです。

Oracle BI での IBM DB2 Cube Views の使用について

IBM DB2 Cube Views は、IBM 社の登録商標です。プラットフォームの互換性の詳細は、『Oracle Business Intelligence Suite Enterprise Edition システム要件およびサポートされるプラットフォーム』を参照してください。

この機能により、データ・ウェアハウスのパフォーマンスやデータベースの機能性が向上します。DB2 データベースを使用すれば、データベース内に存在するデータの論理的なリレーションシップに関するメタデータを格納できます。また、効率的な DB2 のマテリアライズ照会表（MQT）を使用することにより、データ・ウェアハウス・クエリーの処理を高速化できます。これらの MQT では、リレーショナル・データが事前に集計され、クエリーのパフォーマンスが向上します。

DB2 のクエリー・リライト機能でクエリーを処理する場合、可能であれば、そのクエリーは MQT にルーティングされます。これらのテーブルは基礎となる実テーブルよりも小さく、そのデータは事前集計されているため、MQT に再ルーティングすることでクエリー時間が短縮される場合があります。

DB2 Cube Views Generator はメタデータ・ブリッジの役割を果たし、Oracle BI 独自のメタデータを IBM Cube Views XML ファイルに変換します。メタデータを XML ファイルに変換したら、IBM Cube Views を使用して、変換されたメタデータを DB2 データベースにインポートし、IBM Cube Views メタデータ・カタログ・テーブルに格納します。メタデータをインポートしたら、IBM 最適化アドバイザを使用して、マテリアライズ照会表（MQT）とその索引を作成するスクリプトを生成します。デプロイされた MQT は、DB2 Query Reroute Engine で、受信するアプリケーションのクエリーを最適化するために使用されます。

注意： DB2 には、XML ドキュメントを引数として渡し、メタデータ・オブジェクトの作成、変更、削除または読取りを行う API（ストアド・プロシージャとして実装）が用意されています。IBM Cube Views の詳細は、IBM DB2 のドキュメントを参照してください。

キューブ・メタデータのデプロイ手順

XML ファイルをインポートする前に、DB2 Cube Views Generator で生成された別名 SQL ファイルを実行する必要があります。DB2 Cube Views Generator で生成された XML ファイルには、キューブ・メタデータが XML 形式で含まれます。XML ファイルを DB2 データベースにインポートしたら、マテリアライズ照会表を作成する必要があります。

注意： XML ファイルをインポートする前に、IBM Cube Views とそのツールについて理解しておくことを強くお薦めします。詳細は、IBM のドキュメントを参照してください。

メタデータをデプロイする前に、「インポート・ファイルの生成」（272 ページ）の手順を完了しておいてください。キューブ・メタデータをデプロイするには、ここに示す順番で次の手順を実行してください。

1. IBM Cube Views 用の別名 SQL ファイルの実行（288 ページ）
2. XML ファイルのインポート（288 ページ）
3. マテリアライズ照会表（MQT）作成のガイドライン（290 ページ）

IBM Cube Views 用の別名 SQL ファイルの実行

この手順は、「キューブ・メタデータのデプロイ手順」（288 ページ）の一部です。XML ファイルを DB2 データベースにインポートする前に、別名 SQL ファイルを実行する必要があります。詳細は、該当する IBM のドキュメントを参照してください。

DB2 Cube Views Generator で生成された別名 SQL ファイルは、データ・ウェアハウスが存在するデータベース上で、SQL クライアントによって実行する必要があります。実行すると、データベース内にテーブルの別名（シノニム）が作成されます。

XML ファイルのインポート

この手順は、「キューブ・メタデータのデプロイ手順」（288 ページ）の一部です。

別名 SQL ファイルを実行したら、XML ファイルをデータベースにインポートできます。詳細は、IBM のドキュメントを参照してください。

注意： XML ファイルをインポートする前に、IBM Cube Views とそのツールについて理解しておくことを強くお薦めします。詳細は、IBM のドキュメントを参照してください。

このファイルは、次の IBM ツールを使用してインポートできます。

- **IBM OLAP Center（推奨）**：詳細は、「IBM OLAP Center を使用して XML ファイルをインポートするためのガイドライン」（289 ページ）および IBM のドキュメントを参照してください。
- **IBM コマンドライン・クライアント・ユーティリティ（db2mdapiclient.exe）**：このユーティリティは DB2 に付属しています。コマンドライン・クライアント・ユーティリティの使用方法の詳細は、IBM のドキュメントを参照してください。
- **IBM DB2 ストアド・プロシージャ**：IBM Cube Views には、SQL ベースおよび XML ベースのアプリケーション・プログラミング・インタフェース（API）が用意されており、これを使用すると、単一のストアド・プロシージャを実行してメタデータ・オブジェクトを作成、変更および取得できます。詳細は、IBM のドキュメントを参照してください。

IBM OLAP Center を使用して XML ファイルをインポートするためのガイドライン

IBM OLAP Center を使用すると、キューブ・メタデータを DB2 データベースにインポートできます。IBM OLAP Center では、ウィザードを使用してファイルをインポートできます。詳細は、IBM のドキュメントを参照してください。

XML ファイルをインポートするには、次のガイドラインに従ってください。

- IBM OLAP Center ツールを使用して、DB2 データベースに接続します。
- インポート・ウィザードで、インポートする XML ファイルを選択します。
- データベースにないデータベース構造を参照するメタデータが存在する場合は、エラー・メッセージが表示されます。
- ウィザードでインポート・オプションを選択する指示が表示されたら、既存のオブジェクトを置換するように選択します。
- IBM OLAP Center に戻ると、キューブ・モデルの図が表示されます。

XML ファイルのインポート後にキューブ・メタデータを変更するためのガイドライン

XML ファイルのインポート後、次の処理が必要な場合があります。

- Oracle Business Analytics Warehouse では外部キーがメタデータとして格納されないため、DB2 データベース内の変換されたメタデータには存在しません。IBM Cube Views 対応の IBM 参照整合性ユーティリティを使用して、外部キーの情報制約を生成する必要があります。このユーティリティは IBM 社の Web サイトで入手できます。
- 外部キー結合列が NULL 値可能になるなど、その他の問題が発生する場合があります。この問題は、次の方法を使用して解決できます。
  - これらの列のデータが NULL でない場合は、これらの列を NOT NULL 列に変換することをお薦めします。
  - これらの列のデータが NULL である場合や、列のデータが NULL でなくても列のデータ型を変換しないことが望ましい場合は、次のガイドラインに従ってキューブ・モデルを変更することをお薦めします。
    - ファクトとディメンションの結合では、このディメンション・オブジェクトを変換後のキューブ・モデルから手動で削除して、この結合の外部キーで構成されるデジェネレート・ディメンション・オブジェクトを作成する必要があります。
    - ディメンションとディメンションの結合では、結合の主キー側のディメンション・オブジェクトを変換後のキューブ・モデルから手動で削除して、この結合の外部キーで構成されるデジェネレート・ディメンション・オブジェクトを作成する必要があります。
    - ファクトとファクトの結合では、結合の主キー側のファクト・オブジェクトを変換後のキューブ・モデルから手動で削除して、この結合の外部キーで構成されるデジェネレート・ディメンション・オブジェクトを作成する必要があります。
- Cube Generator では、メタデータの増分変更はできません。スキーマを変更するには、DB2 データベースでキューブ・モデル・メタデータを手動で削除して、Oracle BI メタデータを再度変換する必要があります。たとえば、Oracle BI メタデータ・リポジトリ内のキューブでディメンションを変更する場合は、DB2 データベースでキューブ・モデルを削除し、Oracle BI リポジトリから XML ファイルを再生成して、この XML ファイルを DB2 データベースにインポートする必要があります。
- DB2 Cube Views Generator を使用してメタデータを削除することはできません。Oracle BI 管理者は、IBM OLAP Center を使用してキューブ・モデルを手動で削除する必要があります。

- IBM 統計ツールおよび IBM 最適化アドバイザを定期的に実行する必要があります。

詳細は、該当する IBM のドキュメントを参照してください。

マテリアライズ照会表（MQT）作成のガイドライン

この手順は、「キューブ・メタデータのデプロイ手順」（288 ページ）の一部です。詳細は、IBM のドキュメントを参照してください。

キューブ・メタデータをデータベースにインポートしたら、Oracle BI 管理者は、IBM 最適化アドバイザを実行して SQL スクリプトを生成し、これらのスクリプトを実行して MQT を作成します。Oracle BI 管理者は、実装から最適な結果を得るために、IBM 最適化アドバイザに特定のパラメータを指定する必要があります。IBM 最適化アドバイザ・ウィザードは、メタデータを分析し、SQL クエリーの集計データを格納して索引付けするサマリー・テーブルの構築方法を推奨します。IBM 最適化アドバイザを実行すると、MQT を最新の状態に維持できます。また、ETL が終了するたびに、データベースをリフレッシュする必要があります。

MQT を作成するには、次のガイドラインに従ってください。

- IBM OLAP Center で、最適化するキューブ・モデルを選択して、IBM 最適化アドバイザ・ウィザードを起動します。
- 次の表を参考に、ウィザードの指示に従います。

項目	選択方法
Summary Tables	「Deferred」（または「Immediate」）を選択して、テーブルの表領域を指定します。
Limitations	最適化パラメータに適切な値を選択します。「Data-sampling」オプションを選択する必要があります。
SQL Scripts	サマリー・テーブルを作成するために実行するスクリプトの作成。ファイル名と場所を選択します。

- IBM最適化アドバイザが閉じたら、Oracle BI管理者はSQLスクリプトを実行してMQTを作成する必要があります。

13 Oracle BI リポジトリの変数の使用

リポジトリの変数を使用すると、管理タスクを合理化し、メタデータの内容を動的に修正してデータ環境の変更に対応できます。Administration Tool には、変数を定義するための Variable Manager が含まれています。

この章の内容は次のとおりです。

Variable Manager の使用

Variable Manager を使用すると、変数を定義できます。「Variable Manager」ダイアログ・ボックスには 2 つのペインがあります。左ペインには変数と初期化ブロックを表示するツリー、右ペインには左ペインで選択した項目の詳細が表示されます。

変数には、リポジトリ変数とセッション変数の 2 種類のクラスがあります。

- リポジトリ変数は常に単一の値を持ちます。リポジトリ変数には、静的変数と動的変数の 2 種類があります。リポジトリ変数は、疑問符アイコンで示されます。
- セッション変数は、各ユーザーがログオンする際に作成され、値が割り当てられます。セッション変数には、システム変数と非システム変数の 2 種類があります。

  システム変数および非システム変数は、疑問符アイコンで示されます。

初期化ブロックは、動的リポジトリ変数、システム・セッション変数および非システム・セッション変数の初期化に使用されます。初期化ブロックのアイコンは、*i* というラベルの付いた立方体です。

この項の内容は次のとおりです。

リポジトリ変数の概要と作成

リポジトリ変数は常に単一の値を持ちます。リポジトリ変数は、Administration Tool の式ビルダーでリテラルまたは定数のかわりに使用できます。Oracle BI Server によって、メタデータ内の変数自身がリポジトリ変数の値に置き換えられます。

この項の内容は次のとおりです。

静的リポジトリ変数

静的リポジトリ変数の値は「Variable」ダイアログ・ボックスで初期化されます。この値は永続し、Oracle BI 管理者が変更しないかぎり変わりません。

例

1 日の時間をまとめていくつかのセグメントに分ける式を作成するとします。そのセグメントの 1 つが Prime Time であり、対応する時間帯が 5:00 ～ 10:00 PM である場合は、次のような CASE 文を作成します。

```
CASE WHEN "Hour" >= 17 AND "Hour" < 23 THEN 'Prime Time' WHEN...ELSE...END
```

ここで Hour は、日付と時間の Hour(<<timeExpr>>) 関数を使用してタイムスタンプの物理列にマップされている論理列です。

この式に定数の 17 および 23 を入力するかわりに、「Variable」ダイアログ・ボックスの「Variable」タブで prime_begin および prime_end という静的リポジトリ変数を設定して、それぞれの値を 17 および 23 で初期化できます。

式ビルダーでの変数の使用

作成した変数は、式ビルダーで使用できます。式ビルダーで、左ペインにある「Repository Variables」フォルダをクリックすると、中央のペインにすべてのリポジトリ変数（静的変数と動的変数）が名前順に表示されます。

リポジトリ変数を式で使用するには、変数を選択してダブルクリックします。式ビルダーによって、式内のアクティブなカーソル挿入ポイントに変数が貼り付けられます。

変数は、関数 VALUEOF( ) の引数として使用する必要があります。これは、変数をダブルクリックして式に貼り付けるときに自動的に実行されます。

たとえば、次の CASE 文は前述の例と同じですが、定数のかわりに変数を使用している点が異なります。

```
CASE WHEN "Hour" >= VALUEOF("prime_begin")AND "Hour" < VALUEOF("prime_end") THEN
'Prime Time' WHEN ...ELSE...END
```

注意：変数を使用して、列オブジェクトまたは別のリポジトリ・オブジェクトを表すことはできません。

動的リポジトリ変数

動的リポジトリ変数の初期化は、静的リポジトリ変数と同じ方法で行います。ただしその値は、クエリーから返されるデータによってリフレッシュされます。動的リポジトリ変数を定義する場合は、初期化ブロックを作成するか、または SQL クエリーを含む既存の初期化ブロックを使用します。Oracle BI Server でクエリーを実行して変数の値を定期的にリフレッシュするためのスケジュールも設定します。

注意：動的リポジトリ変数の値が変わると、その変数の値を参照するビジネス・モデルに関連付けられているすべてのキャッシュ・エントリが自動的に削除されます。

クエリーの各列に変数を 1 つ指定して、クエリーごとに複数の変数をリフレッシュできます。これらのクエリーは Oracle BI Server で実行するようにスケジュールします。

例

動的リポジトリ変数は、論理テーブル・ソースのコンテンツを定義する場合に非常に役立ちます。たとえば、注文情報に関する 2 つのソースがあるとします。一方のソースには最新の注文が含まれ、もう一方のソースには履歴データが含まれています。

これらのソースのコンテンツは、「Logical Table Source」ダイアログ・ボックスの「Content」タブに記述する必要があります。動的リポジトリ変数を使用しない場合は、次のような式を使用して、最新データが含まれるソースのコンテンツを記述します。

```
Orders.OrderDates."Order Date" >= TIMESTAMP '2001-06-02 00:00:00'
```

このコンテンツの文は、最新のソースに新しいデータが追加され、古いデータが履歴ソースに移動されると無効になります。最新のソースの新しいコンテンツを適切に反映するには、フラグメンテーション・コンテンツの記述を手動で変更する必要があります。これを自動的に行うように動的リポジトリ変数の値を設定できます。

動的リポジトリの値は、論理テーブル・ソースの WHERE 句フィルタでも使用できます。その場合は、「Logical Table Source」ダイアログ・ボックスの「Content」タブで定義します。

動的リポジトリ変数の値は、変数の初期化ブロックに定義されたクエリーによって設定されます。動的リポジトリ変数を定義する場合は、初期化ブロックを作成するか、またはクエリーを含む既存のブロックを使用します。Oracle BI Server でクエリーを実行して変数の値を定期的にリフレッシュするためのスケジュールも設定します。

一般的に、これらの変数は Oracle BI Presentation Services で使用するフィルタ設定に使用されます。たとえば、動的リポジトリ変数 CurrentMonth の値に基づいて列をフィルタ処理するには、フィルタに変数 CurrentMonth を設定します。

リポジトリ変数の作成

リポジトリ変数を作成するには、次の手順を実行します。詳細は、「リポジトリ変数の概要と作成」（292 ページ）を参照してください。

リポジトリ変数を作成するには

1. Administration Tool のメニュー・バーから、「Manage」→「Variables」を選択します。
2. 「Variable Manager」ダイアログ・ボックスのメニュー・バーから、「Action」→「New」→「Repository」→「Variable」を選択します。
3. 「Variable」ダイアログ・ボックスに変数名を入力します。

   すべての変数名は一意にする必要があります。システム・セッション変数の名前は予約されており、他の種類の変数として使用することはできません。
4. 「Variables」ダイアログ・ボックスで、「Static」と「Dynamic」のいずれかの変数の種類を指定します。

   選択した変数の種類に応じて、ダイアログ・ボックスの名前が変更されます。
5. （動的リポジトリ変数）「Initialization Block」ドロップダウン・リストを使用して、定期的な値のリフレッシュに使用される既存の初期化ブロックを選択します。

   新しい初期化ブロックを作成するには、「New」をクリックします。詳細は、「初期化ブロックの作成プロセス」（301 ページ）を参照してください。
6. （動的変数または静的変数）デフォルト・イニシャライザの値を追加するには、次のいずれかの手順を実行します。
   - 式ビルダーを使用するには、デフォルト・イニシャライザ・ワークスペースの横にある省略記号ボタンをクリックします。値の作成の詳細は、「SQL 論理演算子」（374 ページ）を参照してください。
   - 「Default initializer」テキスト・ボックスに値を入力します。

   静的リポジトリ変数では、「Default initializer」ウィンドウで指定した値が永続化されます。この値は管理者が変更しないかぎり変わりません。文字列を使用して変数を初期化する場合は、文字列を一重引用符（'）で囲みます。
7. 「OK」をクリックします。

セッション変数の概要と作成

セッション変数は、その値を初期化ブロックから取得する点では動的リポジトリ変数に似ています。ただし、動的リポジトリ変数と異なり、セッション変数の初期化はスケジュールされません。ユーザーがセッションを開始すると、Oracle BI Server により、セッション変数の新しいインスタンス作成されて初期化されます。

リポジトリ変数と異なり、セッション変数のインスタンスは、Oracle BI Server 上のアクティブなセッションの数と同じだけ存在します。セッション変数の各インスタンスは、異なる値に初期化できます。

セッション変数は、主に、データベース・テーブルや LDAP サーバーなどの外部ソースに対するユーザー認証に使用されます。ユーザーが正常に認証されると、セッション変数を使用して、そのセッションのフィルタおよび権限を設定できます。セッション変数を使用したセキュリティ設定の詳細は、第 15 章「Oracle BI におけるセキュリティ」を参照してください。

この項の内容は次のとおりです。

詳細は、「リポジトリ変数の作成」（294 ページ）を参照してください。

システム・セッション変数の使用

システム・セッション変数は、Oracle BI Server および Oracle BI Presentation Services によって特殊な目的に使用されるセッション変数です。システム・セッション変数の名前は予約されており、静的または動的リポジトリ変数や非システム・セッション変数など、他の種類の変数として使用することはできません。

SA System サブジェクト領域とともに GROUP システム・セッション変数を使用して、Oracle BI Delivers に対してグループ・メンバーシップおよび外部電子メール・アドレスを提供する方法については、「Delivers で操作するためのリポジトリの設定」（183 ページ）を参照してください。

注意： Oracle BI Presentation Services でこれらの変数を使用する場合は、その変数名の前に NQ_SESSION を付けます。たとえば、変数 LOGLEVEL の値に基づいて列をフィルタ処理するには、フィルタに変数 NQ_SESSION.LOGLEVEL を設定します。

296 ページの表 34 に、使用可能なシステム・セッション変数を示します。

表 34. システム・セッション変数

変数	説明
DISPLAYNAME	Oracle BI Presentation Services で使用されます。この変数には、Oracle BI Presentation Services ユーザー・インタフェースの挨拶文に表示される名前が含まれます。また、カタログ・オブジェクトの作成者フィールドとしても保存されます。内部の Oracle BI リポジトリ・ユーザー（非データベース・ユーザー）の場合、この変数にはユーザーのフルネームが移入されます。
EMAIL	Answers で使用されるユーザーのデフォルトの電子メール・アドレスが含まれます。Answers の配信オプションが有効な場合は、このアドレスを使用した電子メール・デバイスが、ユーザーの最初のログイン時に作成されます。ユーザーは、Oracle BI Presentation Services のアカウント設定を変更することで、このアドレスを無効にできます。
GROUP	ユーザーが所属するグループが含まれます。これらは、Oracle BI Server および Oracle BI Presentation Services の両方で使用されます。 ユーザーが複数グループに所属している場合はグループ名をセミコロンで区切ります。テキストは、一重引用符または二重引用符などで囲まないでください。グループ・メンバーシップを含めるには、データベース・テーブルの Varchar 列を使用します。 たとえば、ユーザーが Sales US、Sales UK、QA and Dev および Doc というグループに所属している場合、データベース・テーブルの Varchar データ型の列に入力されるテキストは次のようになります。 `Sales US;Sales UK;QA and Dev;Doc` 注意：Oracle BI Presentation Services の管理者は、Presentation Services のグループ名が、Oracle BI Presentation Services へのログオンに使用されるユーザー ID と同じにならないように注意する必要があります。ユーザーと Presentation Services のグループで同じ名前を使用すると、ユーザーが Oracle BI Presentation Services にログオンする際、「Invalid Account」メッセージが表示されます。
LAST_SYNCH_TIME および THIS_SYNCH_TIME	これらの 2 つの変数は、Oracle BI Disconnected Analytics の同期化を管理するために Oracle BI Presentation Services で設定および追跡されます。詳細は、『Oracle Business Intelligence Disconnected Analytics 管理および構成ガイド』を参照してください。
LOGLEVEL	LOGLEVEL の値（0 ～ 5 の数字）は、Oracle BI Server でユーザー・クエリーに使用されるロギング・レベルを決定します。 このシステム・セッション変数は、ユーザー・オブジェクトで定義されている変数よりも優先されます。管理者ユーザー・オブジェクトのロギング・レベルが 4 で、リポジトリに定義されているセッション変数 LOGLEVEL が 0（ゼロ）の場合は、値 0 が適用されます。
PORTALPATH	Oracle BI Presentation Services で使用されます。この変数は、ログイン時に表示されるユーザーのデフォルト・ダッシュボードを識別します（ログオン後、ユーザーはこのプリファレンスを無効にできます）。

表 34. システム・セッション変数

変数	説明
REQUESTKEY	Oracle BI Presentation Services で使用されます。空でない同じリクエスト・キーを持つユーザーは、Presentation Server の同じキャッシュ・エントリを共有します。これにより、Oracle BI Presentation Services に対して、これらのユーザーが Oracle BI Server で同じコンテンツ・フィルタおよびセキュリティを持つことが示されます。Presentation Server のキャッシュ・エントリの共有は、Oracle BI Server との不要な通信を最小限に抑える 1 つの方法です。
SKIN	Oracle BI Presentation Services ユーザー・インタフェースの外観の特定要素を決定します。ユーザーは、Oracle BI Presentation Services へのログオン時にスタイルを選択することで、ユーザー・インタフェースの一部の要素を変更できます。SKIN 変数は、変更できない要素（たとえば、GIF ファイルなどの画像）が含まれる Oracle BI Presentation Services フォルダをポイントします。このようなディレクトリは、名前の最初に sk_ が付きます。たとえばフォルダ名が sk_companyx である場合、SKIN 変数には companyx を設定します。
USER	ユーザーがログオン名として入力した値を保持します。
WEBGROUPS	Oracle BI Presentation Services 固有のその他のグループがある場合は指定します。Presentation Services グループを使用すれば、より詳細にコンテンツを制御できます。

非システム・セッション変数の使用

非システム・セッション変数を定義する手順は、システム・セッション変数の場合と同じです。

非システム・セッション変数は、一般的にユーザー・フィルタの設定に使用されます。たとえば、ユーザーの販売地域名で初期化される SalesRegion という非システム変数を定義します。

その後、グループのすべてのメンバーに対してセキュリティ・フィルタを設定し、そのメンバーの地域に関係するデータのみが表示されるように指定できます。

注意： Oracle BI Presentation Services でこれらの変数を使用する場合は、その変数名の前に NQ_SESSION を付けます。たとえば、変数 SalesRegion の値に基づいて列をフィルタ処理するには、フィルタに変数 NQ_SESSION.SalesRegion を設定します。

セッション変数の作成

セッション変数を作成するには、次の手順を実行します。

セッション変数を作成するには

1. Administration Tool のメニュー・バーから、「Manage」→「Variables」を選択します。
2. 「Variable Manager」ダイアログ・ボックスのメニュー・バーから、「Action」→「New」→「Session」→「Variable」を選択します。

3 「Session Variable」ダイアログ・ボックスに変数名を入力します。

すべての変数名は一意にする必要があります。システム・セッション変数の名前は予約されており、他の種類の変数として使用することはできません。

4 セッション変数について、次のチェック・ボックスを選択できます。

Enable any user to set the value	初期化ブロックで（ユーザーのログイン時に）ODBC ストアド・プロシージャ NQSSetSessionValue() をコールして値が移入された後、セッション変数を設定できるようにします。たとえば、このチェック・ボックスを選択すると、Oracle BI 管理者以外のユーザーがサンプリングのためにこの変数を設定できるようになります。
Security Sensitive	この変数を、仮想物理データベース（VPD）のセキュリティに依存する変数として識別します。キャッシュ・テーブルの一致をフィルタ処理する場合、Oracle BI Server では、論理リクエストの予測リストで参照されている各列またはテーブルの親データベースが確認されます。この物理データベース・ソースが VPD である場合は、セキュリティ依存変数のリストを、それぞれの予期されるキャッシュ・ヒットと一致させます。キャッシュ・ヒットは、すべてのセキュリティ依存変数に含まれ、一致したキャッシュ・エントリについてのみ発生します。

5 「Initialization Block」ドロップダウン・リストを使用して、定期的な値のリフレッシュに使用される初期化ブロックを選択します。

新しい初期化ブロックを作成するには、「New」をクリックします。詳細は、「初期化ブロックの作成プロセス」（301 ページ）を参照してください。

6 デフォルト・イニシャライザの値を追加するには、次のいずれかの手順を実行します。

- 式ビルダーを使用するには、デフォルト・イニシャライザ・ワークスペースの横にある省略記号ボタンをクリックします。値の作成の詳細は、「SQL 論理演算子」（374 ページ）を参照してください。
- 「Default initializer」テキスト・ボックスに値を入力します。

7 「OK」をクリックします。

変数を持つ初期化ブロックの使用について

初期化ブロックは、動的リポジトリ変数、システム・セッション変数および非システム・セッション変数の初期化に使用されます。たとえば、NQ_SYSTEM 初期化ブロックはシステム・セッション変数のリフレッシュに使用されます。

初期化ブロックには、そのブロックに関連付けられている変数を初期化またはリフレッシュするために実行される SQL が含まれています。この SQL は、「Initialization Block」ダイアログ・ボックスの「Connection Pool」フィールドに指定されている接続プールを使用してアクセス可能な物理テーブルを参照する必要があります。

初期化ブロックのクエリーにデータベース固有の SQL を含める場合は、そのクエリーのデータベース型を選択できます。初期化ブロックをインスタンス化する際にそのデータベース型の SQL 初期化文字列が定義されている場合は、この文字列が使用されます。それ以外の場合は、デフォルトの SQL 初期化文字列が使用されます。

警告：デフォルトでは、オンライン・モードで編集するために「Initialization Block」ダイアログ・ボックスを開くと、その初期化ブロック・オブジェクトが自動的にチェック・アウトされます。初期化ブロックがチェック・アウトされている間、Oracle BI Server は、この初期化ブロックによってリフレッシュされる動的変数の値を、設定された間隔に従って継続的にリフレッシュします。初期化ブロックをチェック・インすると、動的変数の値は、「Default initializer」に示されている値にリセットされます。この処理が必要ない場合は、「Undo Check Out」オプションを選択します。

動的リポジトリ変数の初期化

動的リポジトリ変数の値は、「Initialization Block」ダイアログ・ボックスの「Initialization string」フィールドに定義されたクエリーによって設定されます。Oracle BI Server でクエリーを実行して変数の値を定期的にリフレッシュするためのスケジュールも設定します。Oracle BI Server を停止して再起動すると、リポジトリ変数の初期化ブロック内の SQL が自動的に実行され、リポジトリ変数が再初期化されます。

Oracle BI 管理者のロギング・レベルが 2 以上に設定されている場合、Oracle BI Server では、リポジトリ変数情報を取得するために発行されたすべての SQL クエリーが NQQuery.log ファイルに記録されます。最も有用なレベルの情報を得るには、Oracle BI 管理者のユーザー ID についてロギング・レベルを 2 に設定する必要があります。NQQuery.log ファイルのデフォルトの格納場所は、Oracle BI Server ソフトウェアのインストール・フォルダ（¥OracleBI）直下の Log フォルダです。ユーザーレベルのロギングの詳細は、「クエリー・ログの管理」（221 ページ）を参照してください。

セッション変数の初期化

動的リポジトリ変数と同様、セッション変数もその値を初期化ブロックから取得します。ただし、動的リポジトリ変数とは異なり、セッション変数はスケジュールされた時間間隔でリフレッシュされません。かわりに、新しいユーザー・セッションが開始されるたびに、Oracle BI Server によってこれらの変数の新しいインスタンスが作成されます。セッションが継続する間、値は変更されずに維持されます。

Oracle BI Server では、Security Manager のユーザー・オブジェクトでロギング・レベルが 2 以上に設定されている場合、または Variable Manager で LOGLEVEL システム・セッション変数が 2 以上に設定されている場合、すべての SQL クエリーがログに記録されます。

NQQuery.log ファイルのデフォルトの格納場所は、Oracle BI Server ソフトウェアのインストール・フォルダ（¥OracleBI）直下の Log フォルダです。ユーザーレベルのロギングの詳細は、「クエリー・ログの管理」（221 ページ）を参照してください。

行ベクトルの初期化

行ベクトルの初期化オプションを使用すると、セッション変数を動的に作成して、セッションの開始時に値を設定できます。このセッション変数の名前と値は、接続プールを介してアクセスする外部データベースに格納されます。変数の値は、「Initialization Block」ダイアログ・ボックスに入力した初期化文字列から取得されます。

たとえば、RW_SESSION_VARS という名前のテーブルに含まれる値を使用してセッション変数を作成します。テーブルには 3 つの列が含まれています。USERID 列にはユーザーの一意の識別子を表す値、NAME 列にはセッション変数の名前を表す値、VALUE 列にはセッション変数の値を表す値が含まれています。

テーブルのコンテンツは次のとおりです。

USERID	NAME	VALUE
JOHN	LEVEL	4
JOHN	STATUS	FULL-TIME
JANE	LEVEL	8
JANE	STATUS	FULL-TIME
JANE	GRADE	AAA

初期化ブロックを作成して、「Row-wise initialization」チェック・ボックスを選択します（「初期化ブロックの作成プロセス」（301 ページ）を参照してください）。

初期化文字列には次の SQL 文を入力します。

```
select NAME, VALUE
from RW_SESSION_VARS
where USERID='VALUEOF(NQ_SESSION.USERID)'
```

NQ_SESSION.USERID は、すでに別の初期化ブロックを使用して初期化されています。

次のセッション変数が作成されます。

- John が Oracle BI Server に接続すると、John のセッションには、行ベクトルの初期化によって、値 4 に設定された LEVEL と値 FULL_TIME に設定された STATUS の 2 つのセッション変数が含まれます。
- Jane が Oracle BI Server に接続すると、Jane のセッションには、行ベクトルの初期化によって、値 8 に設定された LEVEL、値 FULL-TIME に設定された STATUS、および値 AAA に設定された GRADE の 3 つのセッション変数が含まれます。

値リストによる変数の初期化

行ベクトルの初期化オプションを使用すると、値リストを使用して変数を初期化することもできます。その後、SQL の IN 演算子を使用して、指定したリストの値をテストできます。

例：前述のテーブルの値を使用する場合、初期化文字列には次の SQL 文を入力します。

```
select 'LIST_OF_USERS', USERID
from RW_SESSION_VARS
where NAME='STATUS' and VALUE='FULL-TIME'
```

この SQL 文では、変数 LIST_OF_USERS に、値 JOHN および JANE がコロンで区切られたリスト（たとえば、JOHN:JANE）として移入されます。この変数は、次の WHERE 句に示すようにフィルタで使用できます。

```
where TABLE.USER_NAME = valueof(NQ_SESSION.LIST_OF_USERS)
```

変数 LIST_OF_USERS には、値のリスト、つまり 1 つ以上の値が含まれています。次の文に示すように、この論理 WHERE 句は物理 IN 句に展開されます。

```
where TABLE.USER_NAME in ('JOHN', 'JANE')
```

初期化ブロックを使用したユーザー認証について

初期化ブロックを使用して、カスタマイズされた認証モジュールを作成できます。認証子は、Oracle BI の認証 API 仕様に従って顧客または開発者によって記述された DLL（または UNIX の共有オブジェクト）であり、Oracle BI Server において認証やその他の作業をランタイムで実行するために使用されます。動的にロード可能な認証子フレームワーク（認証モジュール）とは、認証子を使用して認証および関連作業をランタイムで実行する、キャッシュ・レイヤーを持つ Oracle BI Server モジュールです。

各リポジトリに使用できる認証子は 1 つのみです。ユーザー Administrator に対する認証は、常にリポジトリに対して実行されます。リポジトリ内のユーザーは、常にリポジトリに対して認証されます。

Oracle BI をインストールすると、2 つのサンプル認証子プラグインがインストールされます。1 つは Windows プラットフォームでのみ使用できます。もう 1 つのプラグインでは、テキスト・ファイルを使用してユーザー情報が格納されます。これは、すべてのプラットフォームで使用できます。動的にロード可能な認証子で使用されるすべての型のためにヘッダー・ファイルが用意されています。

Oracle BI 管理者は開発者に対して Oracle BI の認証 API 仕様に従って動的にロードできる認証モジュールの実装をリクエストします。この仕様の詳細は、「Oracle BI Server 認証 API」（419 ページ）を参照してください。

Oracle BI 管理者は、認証オブジェクト（認証子プラグイン）を作成したら、構成ファイルのパス、キャッシュ・エントリ数およびキャッシュの有効期限などの一連のパラメータを認証モジュールに指定します。次に、この認証オブジェクトを初期化ブロックに関連付けます。また Oracle BI 管理者は、USER 変数（必須）とその他の変数を初期化ブロックに関連付けます。

ユーザーのログイン時、認証が成功すると、Oracle BI Server によって初期化ブロックに指定された変数のリストが移入されます。

初期化ブロックの作成プロセス

初期化ブロックには専用の接続プールを作成することをお薦めします。詳細は、「接続プールの作成または変更」（72 ページ）を参照してください。

初期化ブロックの詳細は、「変数を持つ初期化ブロックの使用について」（298 ページ）を参照してください。

初期化ブロックを作成するには、次の手順を実行します。

1. 初期化ブロックの名前とスケジュールの割当て（302 ページ）
2. データソースと接続プールの選択およびテスト（302 ページ）
3. 変数と初期化ブロックの関連付け（306 ページ）
4. 実行の優先順位の確立（307 ページ）

初期化ブロックの名前とスケジュールの割当て

この作業は、「初期化ブロックの作成プロセス」（301 ページ）の 1 つの手順です。

リポジトリ変数には、開始日の曜日、日付および時間と、リフレッシュ間隔を指定できます。

初期化ブロックに名前とスケジュールを割り当てるには

1. Administration Tool のメニュー・バーから、「Manage」→「Variables」を選択します。
2. 「Variable Manager」ダイアログ・ボックスの「Action」メニューから、「New」→「Repository」（または、「Session」）→「Initialization Block」を選択します。
3. 「Variable Init Block」ダイアログ・ボックスで、ブロックの名前を入力します（初期化ブロック名 NQ_SYSTEM は予約されています）。
4. （リポジトリ初期化ブロック）「Schedule」領域で、開始日の日付および時間と、リフレッシュ間隔を選択します。
5. （セッション初期化ブロック）次の適切なチェック・ボックスを選択します。
   - 「Disabled」：選択されている場合、初期化ブロックは無効になります。

     注意：「Variables Manager」で既存の初期化ブロックを右クリックすると、「Disable」または「Enable」のいずれかの切替え値がメニューに表示されます。これを使用すれば、初期化ブロックのダイアログ・ボックスを開かずにプロパティを変更できます。
   - 「Required for authentication」：ユーザー認証用の初期化ブロックを作成する際に使用されます。

次の手順では、データ ソースと接続プールを選択します。

データ ソースと接続プールの選択およびテスト

この作業は、「初期化ブロックの作成プロセス」（301 ページ）の 1 つの手順です。

データ ソース・タイプにデータベースを選択すると、SQL 文の列に対してデータベースから返される値が、初期化ブロックに関連付けられている変数に割り当てられます。セッション変数の初期化ブロックについては、「LDAP」または「Custom Authenticator」を選択できます。

データ ソース・タイプにデータベースを選択する場合、変数のリフレッシュに使用する SQL では、「Connection Pool」フィールドに指定されている接続プールを使用してアクセス可能な物理テーブルを参照する必要があります。このテーブルは、メタデータの物理レイヤーに含まれている必要はありません。実行時にそのデータベース型の初期化文字列が定義されている場合は、この文字列が使用されます。それ以外の場合は、そのデータベース型のデフォルトの初期化 SQL が使用されます。この文字列は上書きできます。

SQL を作成してデータベースに直接発行する場合（初期化ブロックでデータベース固有の SQL を使用するなど）、その SQL は Oracle BI Server を経由しません。SQL 文の列の順序と初期化ブロックに関連付けられている変数の順序によって、各変数に割り当てられる列が決まります。

注意：この SQL は、「Variable Init block Data Source」ダイアログ・ボックスの「Test」ボタンを使用してテストする必要があります。SQL に問題がある場合は、データベースからエラー・メッセージが返されます。

次の各項では、Delivers で使用される初期化文字列の例を示します。

サイトで Delivers を使用する場合の SQL 文の例

```
select username, groupname, dbname, schemaname from users
where username=':USER'
NQS_PASSWORD_CLAUSE(and pwd=':PASSWORD')NQS_PASSWORD_CLAUSE
```

この SQL 文には WHERE 句に絞込み条件が 2 つあります。

':USER'（コロンおよび一重引用符に注意）は、ユーザーがログイン時に入力する ID です。

':PASSWORD'（ここでもコロンおよび一重引用符に注意）は、ユーザーが入力するパスワードです。これは、USER システム・セッション変数が使用されている場合に常に存在することが想定される、別のシステム変数です。PASSWORD 変数を設定する必要はありません。また、この変数をデータベース接続プールで使用すると、このユーザーのユーザー ID とパスワードを使用してパススルー・ログインを実行できます。この変数は、必要に応じて SQL 文で使用することもできます。

外部テーブルによる認証を Delivers で使用する場合は、:PASSWORD 絞込み条件を構成する SQL 文の一部を NQS_PASSWORD_CLAUSE 句の間に埋め込む必要があります。

ユーザー ID とパスワードが、指定されたテーブルにある値と一致した場合のみ、クエリーではデータが返されます。SQL 文は、USER および PASSWORD 変数に有効な値を代入し、NQS_PASSWORD_CLAUSE 句を削除したうえで、Oracle BI Server の外部でテストする必要があります。

詳細は、「Oracle BI Delivers とデータベース認証について」（337 ページ）を参照してください。

サイトで Delivers を使用しない場合の SQL 文の例

```
select username, groupname, dbname, schemaname from users
where username=':USER'
and pwd=':PASSWORD'
```

この SQL 文には WHERE 句に絞込み条件が 2 つあります。

':USER'（コロンおよび一重引用符に注意）は、ユーザーがログイン時に入力する ID です。

':PASSWORD'（ここでもコロンおよび一重引用符に注意）は、ユーザーが入力するパスワードです。これは、USER システム・セッション変数が使用されている場合に常に存在することが想定される、別のシステム変数です。PASSWORD 変数を設定する必要はありません。また、この変数をデータベース接続プールで使用すると、このユーザーのユーザー ID とパスワードを使用してパススルー・ログインを実行できます。この変数は、必要に応じて SQL で使用することもできます。

ユーザー ID とパスワードが、指定されたテーブルにある値と一致した場合のみ、クエリーではデータが返されます。SQL 文は、USER および PASSWORD 変数に有効な値を代入したうえで、Oracle BI Server の外部でテストする必要があります。

初期化ブロックのデータ ソースおよび接続プールを選択するには

1. Administration Tool のメニュー・バーから、「Manage」→「Variables」を選択します。
2. 「Variable Manager」ダイアログ・ボックスで、目的の変数をダブルクリックします。
3. 「Variable Initialization Block」ダイアログ・ボックスで、「Edit Data Source」をクリックします。

4 「Variable Initialization Block Data Source」ダイアログ・ボックスで、「Data Source Type」ドロップダウン・リストから、次のいずれかを選択します。

データソース・タイプ	説明
Database	リポジトリ変数またはセッション変数。
XML	リポジトリ変数またはセッション変数。
LDAP	セッション変数。
Custom Authenticator	セッション変数。詳細は、「初期化ブロックを使用したユーザー認証について」（301 ページ）を参照してください。

5 「Data Source Connection」ドロップダウン・リストで「Database」を選択した場合は、次の手順を実行します。

a 「Browse」をクリックして、対象情報が格納されているデータベースに関連付けられた接続プールを選択します。

警告：初期化文字列を入力する前に接続プールを選択しないと、接続プールを選択するように要求するメッセージが表示されます。

b 「Browse」ダイアログ・ボックスで接続プールを選択し、「OK」をクリックします。

注意：初期化文字列を入力する前に、接続プールを選択してください。

（オプション）「Use Database Specific SQL」チェック・ボックスを選択し、「Database」ペインで、データベースとデータベースに関連付けられた文字列を展開して選択します。

c 「Initialization string」テキスト・ボックスで、変数へのデータの移入に必要な SQL 初期化文字列を入力します。

d （オプション）「Test」をクリックします。SQL 文のデータソースへの接続性をテストします。

6 「Data Source Connection」領域で「XML」を選択した場合は、次の手順を実行します。

a 「Browse」をクリックして、対象情報が格納されているデータベースに関連付けられた接続プールを選択します。

b 「Initialization string」テキスト・ボックスで、変数へのデータの移入に必要な SQL 初期化文字列を入力します。

7 「Data Source Connection」領域で「LDAP」を選択した場合は、次の手順を実行します。

a 「Browse」をクリックして既存の LDAP サーバーを選択するか、「New」をクリックし、「LDAP Server」ダイアログ・ボックスの「General」タブを開いて LDAP サーバーを作成します。

b 「OK」をクリックして「Initialization Block」ダイアログ・ボックスに戻ります。

LDAP サーバー名と関連付けられたドメイン識別子が、それぞれ Name 列と Domain identifier 列に表示されます。

8 「Data Source Connection」領域で「Custom Authenticator」を選択した場合は、次のリストを参考にしてフィールドに値を入力します。

フィールド	説明
Authenticator plug-in	DLL 認証子ファイルを直接入力するか、参照して入力します。
Configuration parameters	構成ファイルを指定します。
Cache never expires	このオプションを選択すると、キャッシュが無期限に有効になります。その場合、キャッシュ・エントリを手動で消去する必要があります。
Cache persistence time	このオプションを選択するとテキスト・ボックスおよびドロップダウン・リストが使用可能になり、テキスト・ボックスに数値を入力したり、時間の増分として日にち、時間、分または秒を選択したりできるようになります。この期間を経過すると、キャシュは自動的に期限切れになります。
Number of cache entries	キャッシュ・エントリの最大数を指定します。

9 「OK」をクリックします。

初期化ブロックのテスト

SQL は、「Test」ボタンまたは Oracle BI Client ユーティリティなどの SQL ツールを使用してテストする必要があります。SQL ツールを使用する場合は、同じ DSN か、指定された接続プールの DSN と同一に設定されている DSN を使用します。

オンライン編集モードの場合、ユーザー名およびパスワードに :USER および :PASSWORD を使用するように設定された接続プールでは、初期化ブロックのテストは機能しません。オフライン・モードの場合は「Set values for variables」ダイアログ・ボックスが表示され、:USER および :PASSWORD に値を設定できます。

初期化ブロックをテストするには（オプション）

1 Administration Tool のメニュー・バーから、「Manage」→「Variables」を選択します。

2 「Variable Manager」ダイアログ・ボックスで、初期化する最後の変数をダブルクリックします。

3 「Variable Initialization Block」ダイアログ・ボックスで、「Test」をクリックします。

4 「Set value for the variables」ダイアログ・ボックスの情報が正しいことを確認し、「OK」をクリックします。

5 「View Data from Table」ダイアログ・ボックスに行数とクエリーの開始行を入力し、「Query」をクリックします。

「Results」ダイアログ・ボックスに変数と値が表示されます。

次の手順では、変数を初期化ブロックに関連付けます。

変数と初期化ブロックの関連付け

この作業は、「初期化ブロックの作成プロセス」（301 ページ）の 1 つの手順です。

「Default initializer」のリストに表示される SQL SELECT 文には、複数の列が含まれることがあります。SQL 文の列の順序と初期化ブロックに関連付けられている変数の順序によって、各変数に割り当てられる列の値が決まります。したがって、初期化ブロックに変数を関連付けると、最初の列に返される値がリスト内の最初の変数に割り当てられます。

詳細は、「変数を持つ初期化ブロックの使用について」（298 ページ）を参照してください。

リポジトリ変数

リポジトリをオンライン・モードで開くと、Oracle BI Server で認識されている現在の変数の値が、「Initialization Block」ダイアログ・ボックスの「Default initializer」フィールドに表示されます。

注意：関連付けられる変数の数は、取得される列の数によって異なります。列よりも変数の数が少ない場合、余分な列の値は無視されます。列よりも変数の数が多い場合、余分な変数はリフレッシュされません（どのような変数であっても、元の値が保持されます）。データベースへの書込みやデータベース構造の変更などを実行するために接続プールに関連付けられているユーザー ID がデータベースで許可される場合、それらの操作を含む適切な SQL が初期化ブロックを使用して実行されます。

Oracle BI Server を停止して再起動すると、リポジトリ変数の初期化ブロック内の SQL が自動的に実行され、リポジトリ変数が再初期化されます。

セッション変数

セッション変数の初期化ブロックについては、「Row-wise initialization」を選択できます。「Row-wise initialization」チェック・ボックスを選択すると、「Cache variables」チェック・ボックスが自動的に選択されます。キャッシュ変数オプションを選択すると、クエリーの結果がメイン・メモリーのキャッシュに格納されるようになります。詳細は、「行ベクトルの初期化」（299 ページ）を参照してください。

Oracle BI Server では、キャッシュされた結果が後続のセッションで使用されます。これにより、セッションの開始時間が短縮されます。ただし、キャッシュされている結果には、セッション変数の最新の値が含まれていない可能性があります。すべての新しいセッションに、最新の一連のセッション変数とその対応する値が必要な場合は、このチェック・ボックスの選択を解除します。

変数を初期化ブロックに関連付けるには

1. Administration Tool のメニュー・バーから、「Manage」→「Variables」を選択します。
2. 「Variable Manager」ダイアログ・ボックスで、目的の変数をダブルクリックします。
3. 「Variable Initialization Block」ダイアログ・ボックスで、「Edit Data Target」をクリックします。
4. 「Variable Initialization Block Variable Target」ダイアログ・ボックスで、次のいずれかを選択します。
   - **「Variables」**：変数を初期化ブロックに関連付けます。
   - **「Row-wise initialization」**：セッション初期化ブロックでのみ使用されます。詳細は、「行ベクトルの初期化」（299 ページ）を参照してください。

     「Row-wise initialization」を選択すると、「Use caching」チェック・ボックスが使用可能になります。

5 「Variables」オプションを選択した場合は、次のいずれかの手順を実行します。

   a 「New」をクリックし、「Variable」ダイアログ・ボックスで新しい変数を作成します。

   注意：データソース・タイプが「Custom Authentication」（セッション変数）である場合は、変数 USER が必要です。

   変数の作成の詳細は、「Variable Manager の使用」（292 ページ）を参照してください。

   b 「Link」をクリックして、既存の変数を初期化ブロックに関連付けます。

      ❑ 「Browse」ダイアログ・ボックスで、この初期化ブロックでリフレッシュする変数を選択し、「OK」をクリックします。

6 変数を並べ替えるには、変数を選択して、「Up」または「Down」をクリックします。

7 このブロックとの関連付けから変数を削除するには、変数を選択して、「Remove」ボタンをクリックします。

8 「OK」をクリックします。

次の手順では、実行の優先順位を確立します。

実行の優先順位の確立

この作業は、「初期化ブロックの作成プロセス」（301 ページ）の 1 つの手順です。

リポジトリに複数の初期化ブロックがある場合は、ブロックを初期化する順序を設定（優先順位を確立）できます。

まず、最後に実行するブロックを開きます。次に、そのブロックの前に実行する初期化ブロックを追加します。たとえば、リポジトリに 2 つの初期化ブロック A および B があるとします。初期化ブロック B を開いてから、ブロック A をブロック B より前に実行するように指定します。これにより、ブロック A は自身のスケジュールに加えて、ブロック B のスケジュールにも従って実行されるようになります。

実行の優先順位を確立するには

1 Administration Tool のメニュー・バーから、「Manage」→「Variables」を選択します。

2 「Variable Manager」ダイアログ・ボックスで、初期化する最後の変数をダブルクリックします。

3 「Variable Initialization Block」ダイアログ・ボックスで、「Edit Execution Precedence」をクリックします。

4 「Variable Initialization Block Execution Precedence」ダイアログ・ボックスで、「Add」をクリックします。

   注意：「Add」は、まだ選択されていない初期化ブロックがある場合にのみ使用できます。

5 「Browse」ダイアログ・ボックスで、開いているブロックよりも前に実行するブロックを選択して、「OK」をクリックします。

   警告：ブロックは初期化する順序で追加してください。

6 ブロックを削除するには、「Variable Initialization Block Execution Precedence」ダイアログ・ボックスで、削除するブロックを選択し、「Remove」をクリックします。

7 「OK」をクリックします。

8 初期化ブロックを必須にする場合は、「Variable Initialization Block」ダイアログ・ボックスで、「Required for authentication」チェック・ボックスを選択します。

9 「OK」をクリックします。

14 Oracle BI Server のクラスタ化

この章では、Cluster Server の概要と、複数のサーバーのクラスタ化を設定および構成する手順について説明します。

この章の内容は次のとおりです。

Cluster Server について

Cluster Server により、ネットワーク・ドメインにおいて最大 16 台の Oracle BI Server を単一のサーバーとして機能させることができます。クラスタ内のサーバーは、Answers や Delivers を含めて、複数の Oracle BI クライアントからのリクエストを共有します。

Cluster Controller は、Cluster Server の主要なコンポーネントです。クラスタ内のリソースの状態を監視し、リソースが変更されたときにセッションの割当てを実行します。また、サーバーと Oracle Business Intelligence Scheduler の障害を検出したり、障害が発生したサーバーの ODBC クライアントをこれらの管理対象サービスのクライアントにフェイルオーバーすることもサポートされています。

Cluster Server のコンポーネント

クラスタ化環境では、次のコンポーネントが利用可能です。

- **2 種類の Cluster Controller**: 詳細は、「Cluster Controller について」（310 ページ）を参照してください。
- **1 台以上のサーバー**：詳細は、「クラスタ化で使用されるサーバーについて」（311 ページ）を参照してください。
- **1 つ以上の Scheduler**: アクティブな Scheduler（ジョブを実行する Scheduler）は 1 つのみです。詳細は、「クラスタ化で使用される Scheduler について」（311 ページ）を参照してください。
- **Cluster Manager**: Administration Tool のユーティリティです。
- **リポジトリ公開ディレクトリ**：このディレクトリは、クラスタに参加しているすべての Oracle BI Server により共有されます。このディレクトリには、オンライン・モードで編集されたリポジトリのマスター・コピーが格納されます。クラスタ化された Oracle BI Server は、起動時にこのディレクトリを確認して、リポジトリに変更がないかどうかを調べます。このディレクトリは通常、クラスタ内のすべてのサーバーから参照可能な共有ファイル・システムに配置されます。この公開ディレクトリに対して、次のようなアクセス権限を設定してください。
  - マスター・サーバーでは、読取り権限と書込み権限を設定する必要があります。
  - すべてのスレーブ・サーバーでは、読取り権限を設定する必要があります。

  NQSConfig.INI ファイルの REPOSITORY_PUBLISHING_DIRECTORY パラメータにより、リポジトリ公開ディレクトリの場所を指定します。

Cluster Controller について

Cluster Controller の種類とその説明を、次に示します。

- **プライマリ Cluster Controller**: プライマリ Cluster Controller のロールは、クラスタ内のサーバーと Scheduler の運用を監視して、クラスタ内でセッションを割り当てることです。プライマリ Cluster Controller は、クラスタ内の Oracle BI Server と同じマシンに配置することができ、また、クラスタと同じサブネットにある別のマシン上に配置することもできます。1 台のマシンで、1 つの Oracle BI Server、1 つの Cluster Controller、1 つの Scheduler またはそのいずれか 1 つをホストできます。また、プライマリ・コントローラは、クラスタ内でアクティブな Scheduler を調べて、アクティブなインスタンスが変更された場合に Scheduler インスタンスに通知します。

  NQClusterConfig.INI ファイルの PRIMARY_CONTROLLER パラメータにより、プライマリ Cluster Controller をホストするマシンを指定します。

- **セカンダリ Cluster Controller**: セカンダリ Cluster Controller は、プライマリ Cluster Controller が使用できない場合に、そのロールを引き受けます。セカンダリ Cluster Controller は、クラスタ内の Oracle BI Server と同じマシンに配置することができ、また、クラスタと同じサブネットにある別のマシン上に配置することもできます。

  NQClusterConfig.INI ファイルの SECONDARY_CONTROLLER パラメータにより、セカンダリ Cluster Controller をホストするマシンを指定します。プライマリ Cluster Controller をホストするマシンと異なるマシンを指定する必要があります。セカンダリ Cluster Controller の指定は任意です。ただし、プライマリ Cluster Controller が使用できなくなると、セカンダリ Cluster Controller が構成されていない場合にクラスタは機能しなくなります。

  注意： NQClusterConfig.ini ファイルでは、完全修飾されたマシン名を使用しないでください。これは、すべてのサーバーを同じ LAN 上で実行する必要があるためです。`machinename.domain` 構文ではなく、`machinename` 構文を使用してください。

クラスタ化で使用されるサーバーについて

クラスタ化で使用されるサーバーの種類について、次に説明します。

- **マスター・サーバー**：マスター・サーバーは、クラスタ化された Oracle BI Server で、リポジトリの変更をオンラインで行うための Administration Tool の接続先となります。NQClusterConfig.INI ファイルの MASTER_SERVER パラメータにより、マスター・サーバーとして機能する Oracle BI Server を指定します。
- **スレーブ・サーバー**：スレーブ・サーバーは、クラスタ化された Oracle BI Server で、オンラインでリポジトリを変更できません。このサーバーは、Oracle BI Server クラスタに対して ODBC セッションのロード・バランシングを行うために使用されます。マスター・サーバーが停止した場合、Administration Tool は利用可能なスレーブ・サーバーに接続しますが、読取り専用モードになります。

クラスタ化で使用される Scheduler について

クラスタ化で使用される Scheduler の種類とその説明を、次に示します。

- **アクティブな Scheduler**: アクティブな Scheduler は、クラスタ化された Oracle BI Scheduler インスタンスで、これにより Scheduler ジョブがアクティブに処理されます。Cluster Controller は、実行時にアクティブなインスタンスを決定し、このインスタンスの Scheduler クラスタに通知します。

- **非アクティブな Scheduler**: 非アクティブな Scheduler は、クラスタ化された Oracle BI Scheduler インスタンスで、Scheduler ジョブをアクティブに処理するのではなく、アクティブな Scheduler に障害が発生した場合に処理を引き継ぐために待機しているインスタンスです。障害が発生したとき以外は、アイドル状態になっています。

Cluster Manager について

Cluster Manager は、オンライン・モードでリポジトリが開いている場合に Administration Tool で使用できるユーティリティです。これによって、Oracle BI 管理者がクラスタ内の操作やアクティビティの監視と管理ができます。

Cluster Manager からサービスの起動、停止または再起動を行うことはできません。ご使用のオペレーティング・システムに用意されている方法を使用して、Oracle BI サービスの停止や再起動を行ってください。Oracle BI Scheduler インスタンスの場合、右クリック・メニューから選択できるのは、「Activate」のみです。

Oracle BI Server インスタンスの場合、右クリック・メニューから選択できるのは、「Quiesce」（新しいセッションの取得の停止）または「Enable」（新しいセッションの取得）のみです。Oracle BI Server インスタンスが実行中の場合は、「Quiesce」を選択できます。このインスタンスが停止している場合は、「Enable」を選択できます。

Cluster Server の実装

Cluster Server を実装するための手順の概要を次に示します。

1. Cluster Controller コンポーネントを含めて、Oracle BI コンポーネントをインストールします。
2. NQSConfig.INI ファイルのパラメータを設定します。
3. NQClusterConfig.INI ファイルのパラメータを設定します。
4. クラスタ化に使用する Oracle BI ODBC データソースを設定します。
5. クラスタに参加するように Scheduler を構成します。
6. Scheduler および Cluster Controller を参照するように Oracle BI Presentation Services を構成します。
7. NQClusterConfig.INI ファイルをクラスタ内の Cluster Controller と Oracle BI Server にコピーします。
8. クラスタのマシンを起動します。

Cluster Server のインストールおよび構成の詳細は、『Oracle Business Intelligence Enterprise Edition デプロイメント・ガイド』を参照してください。

Cluster Server コンポーネントのインストール

このコンポーネントをインストールするには、『Oracle Business Intelligence Enterprise Edition デプロイメント・ガイド』を参照してください。

NQSConfig.INI ファイルにおけるパラメータの設定

NQSConfig.INI ファイルの Server セクションで、クラスタに参加するすべての Oracle BI Server のパラメータを設定する必要があります。NQSConfig.INI ファイルは、Oracle BI ソフトウェアのインストール・フォルダの Config ディレクトリにあります。このファイルとそのパラメータの詳細は、『Oracle Business Intelligence Infrastructure インストレーションおよび構成ガイド』を参照してください。

NQClusterConfig.INI ファイルにおけるパラメータの設定

NQClusterConfig.INI ファイルには、クラスタ構成パラメータが含まれます。Oracle BI Server と Scheduler は、NQSConfig.INI ファイルを読み込んだ後で（NQSConfig.INI ファイルで CLUSTER_PARTICIPANT が YES に設定されている場合）、このファイルを読み込みます。Cluster Controller もこのファイルを読み込みます。

NQClusterConfig.INI ファイルは、Oracle BI のインストール・フォルダの Config ディレクトリにあります。このファイルとそのパラメータの詳細は、『Oracle Business Intelligence Enterprise Edition デプロイメント・ガイド』を参照してください。

Oracle BI ODBC データソース名の構成

Oracle BI Presentation Services クライアントを含めてすべてのクライアントがクラスタと通信するためには、クラスタ化されたデータソース名（DSN）が構成されている必要があります。DSN の設定には、「Oracle BI Server の接続性およびサード・パーティ製ツール」（267 ページ）で説明している Oracle BI DSN 構成ウィザードを使用します。

NQClusterConfig.INI ファイルのコピー

構成済の NQClusterConfig.INI ファイルは、クラスタに参加するすべての Oracle BI Server と Cluster Controller の Config ディレクトリに配置する必要があります。

NQClusterConfig.INI ファイルの構成の詳細は、『Oracle Business Intelligence Enterprise Edition デプロイメント・ガイド』を参照してください。

Cluster Controller と Oracle BI Server の起動

Administration Tool を使用してリポジトリをオンライン・モードで開いている場合は、Cluster Manager を使用してクラスタの Oracle BI Server の操作を監視および管理できます。ただし、リポジトリをオンライン・モードで開いても、クラスタ化された Oracle BI Server または Cluster Controller は自動的に起動しません。したがって、Oracle BI Server と Cluster Controller をそれぞれ 1 つずつ手動で起動する必要があります。その後、Cluster Manager を使用して別のクラスタ化されたサーバーを起動することができます。

注意：すべてのプラットフォームで、Oracle BI Server、Scheduler および Cluster Controller をそれぞれ手動で起動する必要があります。

Oracle BI Server、Scheduler および Cluster Controller を Windows のコマンド・ウィンドウから起動するには

- コマンド・ウィンドウを開いて、次のコマンドを入力します。

```
net start "Oracle BI SERVER"

net start "Oracle BI CLUSTER"

net start "Oracle BI SCHEDULER"
```

注意：リモートでのサービスの操作用に設計されたサード・パーティ製ツールを使用することもできます。

サービスと Cluster Controller を起動したら、テキスト・エディタを使用して Log ディレクトリにあるログ・ファイル（NQServer.log、NQScheduler.log および NQCluster.log）を調べ、すべてのプロセスが問題なく起動されたこと、および動作中のクラスタ構成に正常に参加したことを確認します。ログ・ファイルにエラーが出力された場合は、エラーに対処してからサーバーを再起動します。

Oracle BI Server、Scheduler および Cluster Controller を UNIX のコマンドラインから起動するには

1. コマンド・ウィンドウ（xterm）に移動します。
2. コマンド・ウィンドウで、次のコマンドを入力します（csh を使用する場合は、コマンドは若干異なります）。

```
cd INSTALLDIR/setup

./run-sa.sh start

./run-ccs.sh start

./run-sasch.sh start
```

クラスタ処理（時系列順）

この項では、Oracle BI Cluster Server の起動プロセスの概要について説明します。

1. 各 Oracle BI Server が起動すると、その NQSConfig.INI ファイルが読み込まれます。ファイルの読込み中にサーバーが構文エラーを検出した場合は、エラーはサーバーの Log ディレクトリにある NQServer.log ファイルに記録されます。起動を続行するには、すべての構文エラーに対処する必要があります。
2. CLUSTER_PARTICIPANT が NQSConfig.INI ファイルで YES に設定されている場合、各 Oracle BI Server ではその NQClusterConfig.INI ファイルと Scheduler ファイルが読み込まれます。Cluster Controller もこのファイルを読み込みます。
   - ファイルの読込み中に Oracle BI Server が構文エラーを検出した場合は、エラーはサーバーの NQServer.log ファイルに記録されます。
   - ファイルの読込み中に Cluster Controller が構文エラーを検出した場合は、エラーは NQCluster.log ファイルに記録されます。
   - ファイルの読込み中に Scheduler が構文エラーを検出した場合は、エラーは NQScheduler.log に記録されます。
   - マシンが Oracle BI Server と Cluster Controller の両方をホストしている場合、メッセージは両方のログに書き込まれます。
   - 起動を続行するには、すべての構文エラーに対処する必要があります。

3 Oracle BI Server または Scheduler のインスタンスが起動すると、このインスタンスはプライマリ Cluster Controller およびセカンダリ Cluster Controller からの接続を待機します。

- Oracle BI Server は、Cluster Controller サービスのインスタンスがなくても実行できます。ただし、実行中の Cluster Controller サービスがない場合は、Oracle BI Server に対してクラスタ化された ODBC 接続を確立することができません。
- Scheduler サービスは、起動後に Cluster Controller サービスのインスタンスがオンラインになって状態が変化したことが Scheduler に通知されるまで、非アクティブな状態のままになります。

4 プライマリ Cluster Controller とセカンダリ Cluster Controller がハートビート・メッセージの交換を開始します。この手順は、セカンダリ Cluster Controller が定義されていない場合は省略されます。

5 Oracle BI Server では、リポジトリ公開ディレクトリが使用可能かどうかが確認されます。リポジトリ公開ディレクトリが使用できない場合、各サーバーが実行する処理は、その NQSConfig.INI ファイルの REQUIRE_PUBLISHING_DIRECTORY パラメータの設定によって異なります。

- YES に設定されているときに、公開ディレクトリが起動時に使用できない場合や、サーバーがディレクトリのファイルを読み取るときにエラーが発生した場合は、エラー・メッセージが NQServer.log ファイルに記録され、サーバーが停止します。
- NO に設定されているときは、サーバーはクラスタに参加し、警告メッセージが NQServer.log ファイルに記録されます。ただし、オンラインでのリポジトリの更新は、サーバーの Repository ディレクトリには反映されません。

6 プライマリ Cluster Controller とセカンダリ Cluster Controller が、クラスタ内の各コンポーネントとハートビート・メッセージの交換を開始します。

- 接続ステータスが、クラスタ化された適切なインスタンス（Scheduler または Oracle BI Server）のログ・ファイルに記録されます。また、メッセージも Cluster Controller の NQCluster.log ファイルに記録されます。
- 接続に問題が発生すると、クラスタに参加できません。
- オンライン・リポジトリの MASTER_SERVER として定義されたサーバーが利用できない場合は、リポジトリをオンライン・モードで編集することができません。

7 クラスタの各 Oracle BI Server が起動すると、リポジトリ公開ディレクトリを確認して更新されたリポジトリがないかどうかが調べられます。これは、日付とタイムスタンプを比較して行われます。

注意： Oracle BI Server 管理者は、すべての Oracle BI Server と Cluster Controller でクロックが同期していることを確認する必要があります。

- サーバーで既存リポジトリの新しいバージョンが検出された場合、リポジトリがその Repository ディレクトリにコピーされます。
- サーバーでは、新しいリポジトリの存在は検出されません。新しいリポジトリを作成したら、そのリポジトリをクラスタ化されたすべてのサーバーに手動で伝播する必要があります。これを行うと、各サーバーがそれ以降に起動されたときにオンラインでの変更が検出されます。

8 Cluster Controller がセッションを特定の Oracle BI Server に割り当てると、バックエンド・データベース用の「Connection Pool」ダイアログ・ボックスで定義された接続を使用して、サーバーはこのデータベースと通信します。クラスタ化されたサーバーでは、共通の接続プールを共有しません。ODBC セッションは、1 つの Oracle BI Server セッションとのアフィニティをそのセッションの存続期間中保持します。

9 Oracle BI Server がそのキャッシュ・ファイルからのすべてのクエリーまたはクエリーの一部を満足することができると判断した場合は、これを実行します。クラスタ化されたサーバーでは、共通のキャッシュを共有しません。また、クラスタ化されたサーバーでは、共通の非定型クエリー・キャッシュを共有しません。各サーバー・インスタンスでシード済クエリーのキャッシュを共有するように構成されている場合、これらのインスタンスではこのキャッシュを共有します。

Cluster Manager の使用

Cluster Manager を使用すると、クラスタの操作を監視、分析および管理できます。Cluster Manager により、クラスタを構成するサーバーとコントローラのステータス、キャッシュおよびセッションの情報が用意されます。Cluster Manager は、クラスタ化された DSN に Administration Tool が接続されているときにのみ利用できます。

注意：クラスタ内のすべての Cluster Controller または Oracle BI Server が現在停止しているかオフラインの場合、Cluster Manager にアクセスして起動することはできません。1 台の Cluster Controller（通常はプライマリ）と 1 台の Oracle BI Server を手動で起動する必要があります。

Cluster Manager のグラフィカル・ユーザー・インタフェース（GUI）にはペインが 2 つあります。1 つは「Explorer」ペインで左側にあり、もう 1 つは「Information」ペインで右側にあります。「Explorer」ペインには、クラスタを構成するサーバー、スケジューラおよびコントローラに関する情報が階層表示されます。「Information」ペインには、「Explorer」ペインで選択された項目に関する詳細情報が表示されます。

Cluster Manager のウィンドウは、デフォルトでは 1 分間隔でリフレッシュされます。この間隔は変更することができます。

画面のリフレッシュ間隔を設定するには

1 Administration Tool で、リポジトリをオンライン・モードで開きます。

2 「Manage」→「Clusters」を選択します。

3 この値を変更するには、「Refresh」→「Every」を選択してから、別の値をリストから選択します。

4 画面を必要に応じてリフレッシュするには、Cluster Manager がアクティブ・ウィンドウであることを確認してから [F5] キーを押すか、「Refresh」→「Now」を選択します。

これによって、クラスタの最新情報が取得されます。

非アクティブな Scheduler インスタンスをアクティブにするには

1 Administration Tool で、リポジトリをオンライン・モードで開きます。

2 「Manage」→「Clusters」を選択します。

3 「Cluster Manager」ダイアログ・ボックスで、Scheduler インスタンスを右クリックします。

4 選択した Scheduler インスタンスが非アクティブな場合は、「Activate」を選択します。

これによって、管理者は、実行時にアクティブな Scheduler インスタンスに対して操作できるようになります。

クラスタ情報の表示と管理

この項では、クラスタに関するステータス、キャッシュおよびセッションの情報の表示方法と、表示された情報の意味について説明します。

ステータス情報

最初に「Cluster Manager」ウィンドウを開くと、「Status」ビューが自動的に表示されます。「Status」ビューには、「Cluster Manager」ウィンドウで「View」→「Status」を選択してアクセスすることもできます。

「Information」ペインに表示される情報のカテゴリは、Administration Tool の接続先サーバーによって異なる場合があります。317 ページの表 35 は、表示されるカテゴリについて説明しています。

表 35. 「Status」の列

列	説明
Last Reported Time	Cluster Controller または Oracle BI Server が、制御 Cluster Controller と通信を行った時刻です。サーバーまたはコントローラがオフラインの場合、このフィールドは空欄になる場合があります。
Name	Oracle BI Server または Cluster Controller をホストしているマシンの名前です。
Role	クラスタにおけるオブジェクトのロールです。 ■ **Controlling**: 現在、クラスタを制御するロールが割り当てられている Cluster Controller を示します。 ■ **Primary**: プライマリ Cluster Controller を示します。プライマリ Cluster Controller が現在、制御 Cluster Controller である場合、このロールは表示されません。 ■ **Secondary**: セカンダリ Cluster Controller を示します。セカンダリ Cluster Controller が現在、制御 Cluster Controller である場合、このロールは表示されません。 ■ **Clustered server**: クラスタのメンバーである Oracle BI Server を示します。このロールは、マスター・サーバーとして定義されているクラスタ化されたサーバーには表示されません。 ■ **Master**: クラスタ化されたサーバーが Administration Tool の接続先であることを示し、リポジトリをオンライン・モードで編集するために接続されています。 ■ **Active**:Scheduler がアクティブになっていることを示します。
Sessions	このフィールドは、Cluster Server または個別のサーバーが、「Explorer」ペインで選択された場合に利用可能になります。クラスタ化されたサーバーに現在ログオンしているセッションの数を示します。
Start Time	Cluster Controller または Oracle BI Server が最後に起動したときを示すタイムスタンプです。このフィールドは、Cluster Controller またはクラスタ化されたサーバーがオフラインの場合は空欄になります。

表 35. 「Status」の列

列	説明
Status	クラスタにおけるオブジェクトのステータスです。 ■ **Online**:Cluster ControllerまたはOracle BI Serverがオンラインであることを示します。Cluster Controller の場合は、コントローラがセッション・リクエストを受け入れて、そのリクエストをクラスタ内の利用可能なサーバーに割り当てることができることを意味します。クラスタ化されたサーバーの場合は、Cluster Controller によってサーバーにセッションが割り当てられている可能性があることを意味します。 ■ **Quiesce**: このステータスは、クラスタ化されたサーバーにのみ該当します。このステータスは、未処理のセッションで処理中のアクティビティを完了してから、サーバーをオフラインのステータスに移行できることを意味します。 ■ **Offline**:Cluster Controller または Oracle BI Server がオフラインであることを示します。Cluster Controller の場合は、コントローラがセッション・リクエストを受け入れることができないか、セッションをクラスタ内の利用可能なサーバーに割り当てることができないことを意味します。クラスタ化されたサーバーの場合は、サーバーが制御 Cluster Controller と通信していないため、制御 Cluster Controller によって割り当てたセッションを受け入れることができないことを意味します。サーバーがこの後で利用可能になると、クラスタに参加できるようになります。Cluster Controller またはクラスタ化されたサーバーをオフラインの状態から停止する場合は、Stop コマンドを発行する必要があります。 ■ **Forced Offline**: このステータスは、クラスタ化されたサーバーにのみ該当します。Oracle BI Server は停止しています。これはオフラインのステータスと同じですが、Oracle BI Server がオンライン状態に戻っても、リクエストが割り当てられない点が異なります。このサーバーに対して Administration Tool の Cluster Manager から Start コマンドを発行するか、両方の Cluster Controller を停止してから再起動するまで、サーバーはこの状態のままになります。 ■ **Online: Active**:Scheduler インスタンスがオンラインで実行中であり、Scheduler クライアントの接続先インスタンスであることを示します。このインスタンスではジョブが実行されます。 ■ **Online: Inactive**:Scheduler はオンラインですが、実行中ではありません。このインスタンスは、アクティブなインスタンスが利用できない状態になった場合に、引き継ぐために待機しています。 ■ **Online: Inactive Pending**: アクティブな状態だった Scheduler が、非アクティブの状態に移行しようとしています。この処理は数分かかる場合があります（複数のジョブを実行している場合など）。
Type	クラスタを「Explorer」ペインで選択すると、このフィールドは利用可能になります。次に示す 3 つのタイプがあります。 ■ **Controller**: オブジェクトが Cluster Controller であることを示します。 ■ **Server**: オブジェクトが Oracle BI Server であることを示します。 ■ **Scheduler**: オブジェクトが Scheduler Server であることを示します。

キャッシュ情報

キャッシュが有効である場合は、「Cluster Manager」ウィンドウの「Cache」ビューを使用できます。

情報のカテゴリとその表示順序は、「Options」の設定によって制御されます。319 ページの表 36 は、表示されるカテゴリについて説明しています。

表 36. 「Cache」ビューの列

列	説明
Business Model	キャッシュ・エントリに関連付けられたビジネス・モデルの名前です。
Column count	このキャッシュ・エントリの結果セットの各行に含まれる列の数です。
Created	キャッシュ・エントリの結果セットが作成された時刻です。
Creation elapsed time	このキャッシュ・エントリの結果セットを作成するために必要な時間（ミリ秒単位）です。
Full size	「Full size」は、可変長の列や圧縮アルゴリズムなどの要因を考慮した、使用する最大サイズです。結果セットの実際のサイズは、このキャッシュ・エントリの結果セットの作成に必要な最大サイズ（秒単位）よりも小さくなります。
Last used	キャッシュ・エントリの結果セットがクエリーを満足したときの最後の時刻です。Oracle BI Server が予期せずに停止した後は、一時的に「Last used」の時刻が古い値（つまり、実際の時刻値よりも古い時刻値）になることがあります。
Row count	クエリーによって生成された行の行数です。
Row size	このキャッシュ・エントリの結果セットに含まれる各行のサイズ（バイト単位）です。
SQL	キャッシュ・エントリを生成した SQL 文のテキストです。
Use count	このキャッシュ・エントリの結果セットが（Oracle BI Server の起動後に）クエリーを満足した回数です。
User	キャッシュ・エントリにキャッシュされたクエリーを送信したユーザーの ID です。

キャッシュ情報を表示するには

- 個別のサーバーを「Explorer」ペインでクリックしてから、「View」→「Cache」を選択します。

セッション情報

「Session」ビューを Oracle BI Server で利用できます。情報は、320 ページの表 37 および表 38 で説明されている 2 つのウィンドウに表示されます。

- 「Session」ウィンドウ：上部に表示されます。Oracle BI Server に現在ログオンしているユーザーが表示されます。
- 「Request」ウィンドウ：下部に表示されます。「Session」ウィンドウで選択されたユーザーのアクティブなクエリー・リクエストが表示されます。

320 ページの表 37 では、「Session」ウィンドウに表示される情報について説明します。

表 37. 「Session」ウィンドウの列（上部ウィンドウ）

列	説明
Catalog	セッションの接続先となる「Presentation」レイヤー・カタログの名前です。
Client Type	クライアント・セッションのタイプです。クライアント・タイプ「Administration」は、Oracle BI の管理者ユーザー ID を使用してログインするユーザー用に予約されています。
Last Active Time	セッションまたはクエリーにおける最後のアクティビティのタイムスタンプです。
Logon Time	セッションが Oracle BI Server にログオンしたときのタイムスタンプです。
Repository	セッションの接続先となるリポジトリの論理名です。
Session ID	セッションが開始したときに、Oracle BI Server が各セッションに割り当てた一意の内部識別子です。
User	接続するユーザーの名前です。

320 ページの表 38 では、「Request」ウィンドウに表示される情報について説明します。

表 38. 「Request」ウィンドウの列（下部ウィンドウ）

列	説明
Last Active Time	セッションまたはクエリーにおける最後のアクティビティのタイムスタンプです。
Request ID	クエリーが開始したときに、Oracle BI Server が各クエリーに割り当てた一意の内部識別子です。
Session ID	セッションが開始したときに、Oracle BI Server が各セッションに割り当てた一意の内部識別子です。
Start Time	最初のクエリー・リクエストの時刻です。
Status	可能な値を次に示します。処理が完了するまでの処理速度により、指定されたリクエストまたはセッションのすべての値が表示されるわけではありません。 ■ **Idle**: 現在、リクエストまたはセッションにアクティビティがありません。 ■ **Fetching**: リクエストを取得中です。 ■ **Fetched**: リクエストが取得されました。 ■ **Preparing**: リクエストは処理のために準備している最中です。 ■ **Prepared**: リクエストは処理の準備が完了して、実行待ち状態にあります。 ■ **Executing**: リクエストは現在実行中です。リクエストを中断するには、リクエストを選択してから「Kill Request」ボタンをクリックします。Oracle BI 管理者がリクエストをキャンセルしたことを示すメッセージを、ユーザーは受信します。 ■ **Executed**: リクエストの実行が終了しました。 ■ **Succeeded**: リクエストが正常に完了しました。 ■ **Canceled**: リクエストはキャンセルされました。 ■ **Failed**: リクエストの処理中または実行中にエラーが発生しました。

クラスタ化されたサーバーを管理するには

1. 「Explorer」ペインで「Server」アイコンの左側にあるプラス記号（+）をクリックして、クラスタ内のサーバーを表示します。
2. 「Information」ペインで、サーバーを選択します。
3. 「Action」を選択してから、いずれかの利用可能なオプションを選択します。

   操作が完了すると、クラスタ化されたサーバーのステータスが自動的にリフレッシュされます。

セッション情報を表示するには

- サーバーを「Explorer」ペインでクリックしてから、「View」→「Sessions」を選択します。

  サーバーのセッション情報が「Information」ペインに表示されます。サーバーにログインしているすべてのユーザーと、各ユーザーの現在のクエリー・リクエストすべてが表示されます。

セッションを切断するには

- 「Session」ビューの「Session」ウィンドウ（上部ウィンドウ）でセッションを右クリックしてから、「Disconnect」をクリックします。

クエリー・リクエストを中断するには

- 「Session」ビューの「Request」ウィンドウ（下部ウィンドウ）でリクエストを右クリックしてから、「Kill Request」をクリックします。

サーバー情報

「Server info」を「View」メニューから選択すると、クラスタ・サーバーに関する情報（サーバーのバージョン番号など）が表示されます。

パフォーマンスに関する考慮事項

この項では、クラスタ化された Oracle BI Server のパフォーマンスに影響を与える場合がある Cluster Server の特性について説明します。Cluster Server を実装する際は、これらの点を考慮してください。

- セッションが確立すると、複数のセッションが Oracle BI Server に割り当てられます。セッションは、セッションの処理数が最も少ないサーバーに割り当てられます。その結果、Oracle BI Server の負荷にばらつきが発生して、実行中のクラスタでオンライン状態になった Oracle BI Server は、ある期間に処理リクエストを受信できないことがあります。
- 各 Oracle BI Server には固有のローカル・クエリー結果キャッシュがあるため、結果がキャッシュされている場合でも、バックエンド・データベースは複数の Oracle BI Server から同じクエリーを受信する場合があります。クラスタ対応のキャッシュを使用している場合、キャッシュをシードする iBot を介して明示的にシードされたクエリーは、クラスタ内のノード全体で共有されます。詳細は、『Oracle Business Intelligence Enterprise Edition デプロイメント・ガイド』を参照してください。

- 各 Oracle BI Server には固有のローカル・クエリー結果キャッシュがあるため、そのローカル・キャッシュに移入中であるとき、実行中のクラスタでオンライン状態になった Oracle BI Server では、クエリーに対するレスポンスが遅くなることがあります。
- 各 Oracle BI Server には、各リポジトリの独立したコピーと、固有のバックエンド接続プールがあるため、バックエンド・データベースに N*M の数の接続が発生することがあります。ここで、N はクラスタ内のアクティブなサーバーの数を表し、M は単一のリポジトリの接続プールで可能な最大セッション数を表します。したがって、セッション・プールに構成されている最大セッション数を減らすことが適している場合があります。

15 Oracle BI におけるセキュリティ

Oracle BI Server では、すべてのレベルでアクセスがセキュアに制御されます。この章の内容は次のとおりです。

Oracle BI Security Manager

Oracle BI Security Manager には、リポジトリのすべてのセキュリティ情報が表示されます。Security Manager を使用すると、ユーザーとグループの設定、LDAP ユーザーとグループの同期化、テーブルや列などのオブジェクトへのアクセス権限の設定、および情報に対するフィルタの設定ができます。さらに、ユーザーがデータにアクセスできる状況を広範囲に制御する管理されたクエリー環境を設定できます。

注意：この項を読むと、セキュリティと認証設定の基礎を理解できます。Oracle BI アプリケーションのセキュリティ構成の詳細は、この項を読んだ後に『Oracle Business Intelligence Applications Installation and Administration Guide』を参照してください。

Oracle BI Server と Oracle BI Presentation Services クライアントでは、ログインとパスワードの暗号化において業界標準のセキュリティがサポートされています。エンド・ユーザーがログイン名とパスワードを Web ブラウザに入力すると、Oracle BI Server では HTTPS（Hyper Text Transport Protocol Secure）標準を使用して、Oracle BI Presentation Services 上のセキュアなポートに情報を送信します。情報は Oracle BI Presentation Services から ODBC を経由し、トリプル DES（Data Encryption Standard）を使用して Oracle BI Server に渡されます。この高レベルのセキュリティ（168 ビット）により、権限のないユーザーがデータや Oracle BI メタデータにアクセスすることを防止します。

Oracle BI 管理者は、データベース・レベルでデータベース・セキュリティと認証を実装できます。さらに、独自のキーベースでの暗号化により、認証されていないユーザーは Oracle BI メタデータ・リポジトリにアクセスできなくなります。

この項の内容は次のとおりです。

ユーザーに対する操作

ユーザー・アカウントは明示的に、Oracle BI リポジトリまたは外部ソース（データベースのテーブルや LDAP サーバーなど）に定義できます。ユーザー・アカウントが定義されていても、セッションを開始するには Oracle BI Server から認証を受ける必要があります。

ユーザーがリポジトリで明示的に定義されていると、そのリポジトリのビジネス・モデルにアクセスできますが、他のリポジトリにはアクセスできません。

Oracle BI 管理者のユーザー・アカウントは、リポジトリの作成時に自動的に作成され、削除はできません。Oracle BI 管理者のユーザー・アカウントの詳細は、「Oracle BI 管理者アカウントについて」（326 ページ）を参照してください。

この項の内容は次のとおりです。

リポジトリへの新しいユーザーの追加

次の手順に従って、新しいユーザーをリポジトリに追加します。

新しいユーザーをリポジトリに追加するには

1. リポジトリを Administration Tool で開きます。
2. 「Manage」→「Security」を選択し、Security Manager を表示します。
3. 「Action」→「New」→「User」を選択し、「User」ダイアログ・ボックスを開きます。
4. ユーザー名とパスワードを入力します。
5. このユーザーのクエリーをクエリー・ログに記録するには、クエリー・ロギングのレベルを 1 または 2 に変更します。

   クエリー・ロギングの詳細は、「ロギング・レベルの設定」（222 ページ）を参照してください。
6. 「OK」をクリックします。

   これによって新しいユーザーが作成され、デフォルトの権限が付与されます。NQSConfig.INI ファイルでは、デフォルトの権限が DEFAULT_PRIVILEGES エントリにより指定されます。
7. ユーザーの権限を変更するには、変更対象となるユーザーのアイコンをダブルクリックして、「User」ダイアログを開きます。「Permissions」をクリックすると、複数の列に対応する権限を変更できます。
8. パスワードの有効期限のオプションを指定します。
   - ユーザーのパスワードに有効期限を設定しない場合、「Password Never Expires」オプションを選択します。
   - ユーザーのパスワードに有効期限を設定する場合、「Days」ドロップダウン・リストを使用して、ユーザーのパスワードが期限切れになるまでの日数を選択します。最長期間は 365 日です。

     パスワードの作成または変更後、指定した日数が経過すると、パスワードは有効期限切れとなり変更が必要になります。
9. ユーザーには、ユーザー単位またはグループ単位（あるいはその両方の組合せ）を使用して権限を付与できます。グループのメンバーシップを付与するには、ダイアログ・ボックスの「Group Membership」の部分においてユーザーの参加先グループを必要な数だけ選択します。
10. 特定のデータベース・ログオン ID を 1 つ以上のデータベースに対して指定する場合、「User」ダイアログ・ボックスの「Logons」タブにユーザー用の適切なユーザー ID とパスワードを入力します。

    注意： Oracle BI 管理者が接続プールにデータベース固有のデフォルトのログオン ID とパスワードを構成していないときに、Oracle BI Server への接続に使用する DSN においてデータベース固有のログオン ID をユーザーが指定した場合、指定したログオン ID が使用されます。接続プールにデータベース固有のログオン ID を構成する方法の詳細は、「接続プールの作成または変更」（72 ページ）を参照してください。
11. ユーザー用のクエリー権限を設定します。詳細は、「クエリーの実行権限の管理」（339 ページ）を参照してください。

Oracle BI 管理者アカウントについて

Oracle BI 管理者アカウント（ユーザー ID が Administrator）は、すべての Oracle BI リポジトリにおけるデフォルトのユーザー・アカウントです。これは永続アカウントです。変更や削除はできません（パスワードとロギング・レベルの変更を除く）。リポジトリの管理作業（物理スキーマのインポート、ビジネス・モデルの作成、ユーザーとグループの作成など）をすべて実行できるよう設計されています。

注意： Oracle BI 管理者アカウントは、Windows NT や Windows 2000 の管理者アカウントとは異なります。このアカウントに付与されている管理者権限は、Oracle BI Server 環境においてのみ機能します。

新しいリポジトリを作成すると、Oracle BI 管理者アカウントが自動的に作成されますが、パスワードは割り当てられていません。リポジトリを作成したら、ただちに Oracle BI 管理者アカウントにパスワードを割り当ててください。Oracle BI に同梱されている Oracle BI リポジトリのインストール時に作成される Oracle BI 管理者アカウントには、SADMIN というデフォルトのパスワードがあります。

Oracle BI 管理者アカウントは管理者グループに属し、所属グループから削除できません。Oracle BI 管理者のユーザー ID を使用してログオンしたユーザーや、Oracle BI の管理者グループのすべてのメンバーには、リポジトリにあるすべてを変更する権限があります。Oracle BI 管理者アカウントから発行されるすべてのクエリーは、データにフル・アクセスできます。オブジェクトに対する制限はありません。

注意： パスワードの最短文字列長は、NQSConfig.INI ファイルで MINIMUM_PASSWORD_LENGTH を使用すると設定できます。

グループに対する操作

Oracle BI Server では、グループを作成してそのグループのメンバーシップをユーザーまたは他グループに付与できます。

グループを一組のセキュリティ属性と考えることができます。Oracle BI Server のグループは、Windows NT や Windows 2000 のグループ、およびデータベース管理システム（DBMS）のグループやロールと似ています。Windows NT、Windows 2000 およびデータベースのグループやロールと同様に、Oracle BI Server グループではオブジェクトへのアクセスを許可できます。さらに Oracle BI Server グループではメンバーに対し特定のセキュリティ属性を明示的に拒否できます。

グループにより、大量のユーザーの管理を簡略化できます。一組の権限を付与または拒否したグループに、個々のユーザーをメンバーとして割り当てることができます。その後、そのグループに対して変更を行うと、グループに属するすべてのユーザーが影響を受けます。外部で定義されたユーザーには、GROUP セッション変数を使用してグループのメンバーシップを付与できます。セッション変数の詳細は、「システム・セッション変数の使用」（295 ページ）を参照してください。

この項の内容は次のとおりです。

事前定義済の管理者グループ

Oracle BI Server には、事前定義されたグループ（Oracle BI の管理者グループ）が 1 つあります。このグループのメンバーは、リポジトリ内のどのオブジェクトにもアクセスして変更することができます。事前定義された Oracle BI 管理者ユーザー ID は、自動的に Oracle BI の管理者グループのメンバーになっています。

Oracle BI の管理者グループのメンバーシップをユーザーまたは他のグループに付与する場合は注意が必要です。Oracle BI の管理者グループのメンバーシップは、グループを介したり明示的にユーザー権限を介してユーザーに割り当てられたすべての権限よりも優先されます。Oracle BI の管理者グループのメンバーはすべて、Oracle BI 管理者ユーザーとしての権限がすべて付与されます。

定義済グループ

Oracle BI リポジトリには無制限にグループを作成できます。各グループに権限を明示的に付与したり、別のグループにおけるメンバーシップを使用して権限を暗黙的に付与することができます。グループの設定の詳細は、「新しいグループの追加」（329 ページ）を参照してください。

たとえば、月曜と水曜にリポジトリへのアクセスが拒否されるグループ（グループ 1）、土曜と日曜にアクセスが拒否されるグループ（グループ 2）、火曜、木曜および金曜にアクセスが拒否されるグループ（グループ 3）をそれぞれ作成できます。この場合、グループ 2 のメンバーは平日のみシステムにアクセスできます。グループ 1 とグループ 3 の両方に属するメンバーは週末のみアクセスできます。

グループの継承

ユーザーには権限を明示的に割り当てることができます。グループのメンバーシップを介して権限を割り当てることもできます。また、それによって次々と他のグループのメンバーシップを介してユーザーに権限を付与することもできます。ユーザーに明示的に割り当てられた権限は、グループを介して割り当てられた権限よりも優先されます。また、グループに明示的に割り当てられた権限は、他のグループを介して割り当てられた権限よりも優先されます。

ユーザーまたはグループが同じレベルの複数のグループに属し、それらのセキュリティ属性が競合する場合、最も制限の少ないセキュリティ属性がユーザーまたはグループに付与されます。ユーザーに対して付与された明示的な権限はすべて、グループを介してそのユーザーに割り当てられた同じオブジェクトの権限よりも優先されます。

例 1

ユーザー（ユーザー 1）に特定のテーブル（テーブル A）の読取り権限が明示的に付与されているとします。またユーザー 1 はグループ 1 のメンバーであり、そのグループのメンバーにはテーブル A へのアクセスが明示的に拒否されているとします。328 ページの図 21 に示すように、結果としてユーザー 1 はテーブル A の読取り権限を持ちます。

ユーザーに直接付与されている権限は、グループを介して付与されている権限よりも優先されるので、ユーザー 1 が持つ権限はテーブル A の読取り権限となります。

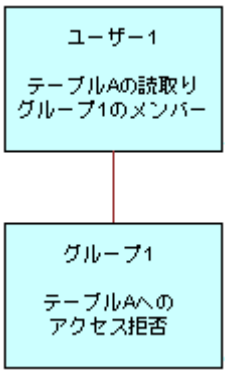


図 21. ユーザー権限とグループ権限

例 2

328 ページの図 22 に示されている状況があると仮定します。

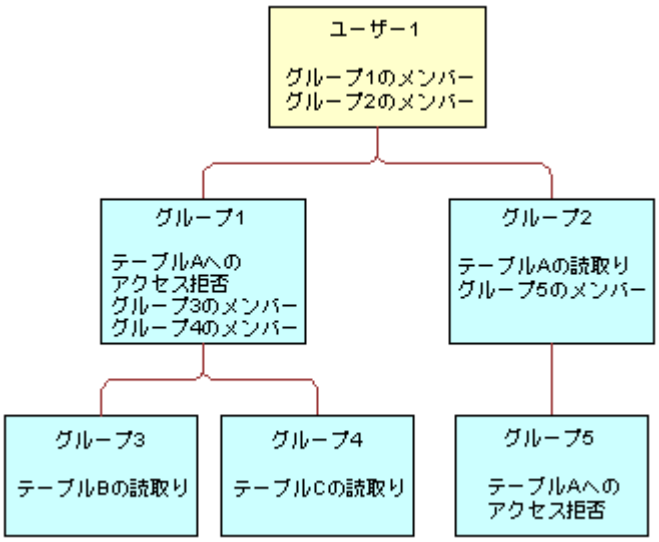


図 22. 権限の例

結果として権限は次のようになります。

- ユーザー1 はグループ 1 とグループ 2 の直接のメンバーで、グループ 3、グループ 4 およびグループ 5 の間接的なメンバーです。
- グループ 5 の優先順位はグループ 2 よりも低いので、グループ 2 を介して割り当てられた読取り権限は、テーブル A へのアクセス拒否よりも優先されます。結果として、ユーザーはグループ 2 からのテーブル A の読取り権限を持ちます。

- グループ 1 から付与される権限は、テーブル A へのアクセス拒否、テーブル B の読取りおよびテーブル C の読取りの権限になります。
- グループ 1 とグループ 2 の優先順位は同じで、権限が相互に相殺されるので（グループ 1 はテーブル A へのアクセスが拒否され、グループ 2 はテーブル A へのアクセスが許可される）、ユーザー 1 は制限の少ないレベルの権限を継承します。つまり、ユーザー 1 にはテーブル A の読取り権限があります。
- 総合するとユーザー 1 には、テーブル A、テーブル B およびテーブル C の読取り権限があります。

新しいグループの追加

次の手順では、新しいグループをリポジトリに追加する方法について説明します。

新しいグループをリポジトリに追加するには

1. リポジトリを Administration Tool で開きます。リポジトリはオンライン・モードでもオフライン・モードでも開くことができます。
2. 「Manage」→「Security」を選択し、「Security」ウィンドウを表示します。
3. メニューから「Action」→「New」→「Group」を選択します。

   「Group」ダイアログ・ボックスが表示されます。

   注意：左ペインから「Group」アイコンを選択し、左ペインの空いているスペースで右クリックしてから、右クリックのメニューから「New Security Group」を選択することもできます。
4. グループ名を入力してから「OK」をクリックします。

   これによって新しいグループが権限なしで作成されます。
5. グループの権限を変更するには、変更対象となるグループのアイコンをダブルクリックして、「Group」ダイアログを開きます。「Permissions」をクリックすると、複数の列に対応する権限を変更できます。
6. グループに権限を付与するには、他のグループを追加するか、グループに対し明示的に構成するか、またはそれらの両方の組合せで行うことができます。メンバーをグループに追加するには「Add」をクリックし、メンバーシップの付与先となるユーザーまたはグループを選択します。グループまたはユーザーを選択してから「OK」をクリックします。
7. グループ用のクエリー権限を設定します。詳細は、「クエリーの実行権限の管理」（339 ページ）を参照してください。

   注意：「User」ダイアログ・ボックスとは異なり、「Group」ダイアログ・ボックスではロギング・レベルを選択できません。ロギング・レベルはユーザー属性のためグループでは指定できません。

メンバー階層の表示

次の手順に従って、メンバー階層を表示します。

メンバー階層を Security Manager で表示するには

- Security Manger の左ペインで階層のアイコンをクリックし、ツリーを右ペインで展開します。

「Query Repository」ダイアログ・ボックスでメンバー階層を表示するには

1. Administration Tool のメイン・メニューから「Tools」→「Query Repository」を選択します。
2. すべてのグループを表示するには、「Type」ドロップダウン・リストから「Security Groups」を選択し、「Query」をクリックします。
3. すべてのユーザーを表示するには、「Type」ドロップダウン・リストから「Users」を選択し、「Query」をクリックします。
4. 別のグループがメンバーである所属先グループを表示するには、メンバー・グループを選択し「Parent」をクリックします。たとえば、グループ 1 がメンバーとなっているグループを表示するには、グループ 1 を選択し「Parent」ボタンをクリックします。

LDAP からのユーザーとグループのインポート

組織で Lightweight Directory Access Protocol（LDAP）を使用している場合、既存の LDAP ユーザーとグループをリポジトリにインポートできます。インポートすると、通常のすべての Oracle BI Server ユーザー機能とグループ機能が使用できるようになります。インポートしたリストはいつでも再び同期化できます。

外部ソースとして LDAP で認証することもできます。この際、ユーザーはリポジトリにインポートされません。ユーザーはログオン時に認証されグループ権限が判別されます。LDAP 認証の使用方法の詳細は、「LDAP 認証の設定」（333 ページ）を参照してください。

注意：変数を同一ユーザーのためにリポジトリと LDAP サーバーの両方において作成すると、ローカルのリポジトリ・ユーザー定義が優先され、LDAP 認証は行われません。これによって、Oracle BI 管理者は外部セキュリティ・システムに存在するユーザーを確実に無視することができます。

この項の内容は次のとおりです。

LDAP サーバーの設定

この項ではリポジトリに対して LDAP 認証を設定する方法について説明します。

注意：セキュリティの基本および認証の設定方法の詳細は、「Oracle BI Security Manager」（324 ページ）を参照してください。

ADSI を認証方式として使用する Oracle BI インスタンスでは、AD インスタンスを設定するとき次のオプションを使用してください。

- 「Log On To」で「All Computers」チェック・ボックスを選択します。一部のコンピュータを選択する場合は、ログオン・ワークステーションとして AD サーバーを含めます。
- 次のオプションは選択しないでください。

  User must change password at next logon

Administration Tool で、「LDAP Server」セクションの「BIND DN」で使用する CN ユーザーには、ldap_bind と ldap_search の両方の権限が必要です。

注意： Oracle BI Server では LDAP 認証でクリアテキスト・パスワードが使用されます。LDAP サーバーの設定でこれが許可されていることを確認します。

リポジトリに対して LDAP 認証を設定するには

1. リポジトリを Administration Tool でオフライン・モードかオンライン・モードで開きます。
2. アプリケーション・メニューから「Manage」→「Security」を選択します。
3. Security Manager のメニューから「Action」→「New」→「LDAP Server」を選択します。
4. 「LDAP Server」ダイアログ・ボックスの「General」タブで必要なフィールドを入力します。次のリストでは、LDAP サーバーの設定に役立つフィールドやボタンとそれらの説明に関する追加情報を示しています。
   - **Host name**:LDAP サーバーの名前です。
   - **Port number**: デフォルトの LDAP ポートは 389 です。
   - **LDAP version**:LDAP 2 または LDAP 3（バージョン）です。デフォルトは LDAP 3 です。
   - **Base DN**: ベース識別名（DN）により認証検索の開始点が判別されます。たとえば、ディレクトリ内の o=Oracle.com サブツリーのエントリをすべて検索する場合は、o=Oracle.com がベース DN です。
   - 「**Bind DN**」と「**Bind Password**」: LDAP サーバーのバインドに必要なオプションの DN とそれに関連付けられているユーザー・パスワードです。

     これら 2 つのエントリが空欄の場合、匿名バインディングとみなされます。セキュリティ上の理由からすべての LDAP サーバーで匿名バインディングが許可されているとはかぎりません。

     これらのフィールドは LDAP V3 ではオプションですが、LDAP V2 では匿名バインディングがサポートされていないので必須です。

     これらのフィールドは「ADSI」チェック・ボックスを選択した際は必須です。これらのフィールドを空欄にすると、パスワードを空のままにすることを確認する警告メッセージが表示されます。「はい」をクリックすると匿名バインディングとみなされます。
   - **Test Connection**: このボタンを使用して、LDAP サーバーへの接続をテストしパラメータが正しいことを検証します。
5. 「Advanced」タブをクリックし、必要な情報を入力します。次のリストでは、LDAP サーバーの設定に役立つフィールドとそれらの説明に関する追加情報を示しています。

   注意： Oracle BI Server では認証キャッシュがメモリーに格納されるので、LDAP を使用して多数のユーザーを認証する際におけるパフォーマンスの向上に役立ちます。この認証キャッシュを無効にすると、何百というセッションを認証する際におけるパフォーマンスが低下する場合があります。

   - **Connection timeout**: Administration Tool においてインポートのために LDAP サーバーに接続したり、Oracle BI Server においてユーザーを認証するために LDAP サーバーに接続する場合、ここで指定した時間が経過すると接続はタイムアウトになります。

- **Domain identifier**: 識別子は通常 1 つの単語で、これにより LDAP オブジェクトが存在するドメインを一意に識別します。LDAP オブジェクトを複数使用している場合、これは特に便利です。別々のユーザーが同じユーザー ID を持ち、それぞれが異なる LDAP サーバー上に存在する場合、ドメイン識別子を指定すると別々のユーザーを区別できます。ユーザーは次のフォーマットを使用して Oracle BI Server にログインします。

  `domain_id/user_id`

  ユーザー ID をドメイン識別子なしで入力した場合、使用可能なすべての LDAP サーバーに対して順々に認証を行います。複数のユーザーが同じ ID を使用する場合、ユーザーは 1 人のみ認証されます。

- **ADSI**:（Active Directory Service Interfaces）。LDAP サーバーの一種です。「ADSI」チェック・ボックスを選択すると、「Bind DN」と「Bind Password」が必須になります。
- **SSL**:（Single Socket Layer）。このチェック・ボックスを選択すると、有効になります。
- **User Name Attribute Type**: これによりユーザーを一意に識別します。多くの場合、これは RDN（相対識別名）です。通常はデフォルト値のままにします。ほとんどの LDAP サーバーではユーザー ID を使用します。ADSI には sAMAccountName を使用します。

注意：キャッシュ設定と SSL の構成方法の詳細は、『Oracle Business Intelligence Enterprise Edition デプロイメント・ガイド』を参照してください。

LDAP からのユーザーのインポート

選択したユーザーやグループ、またはすべてのユーザーやグループをインポートすることができます。以前にインポートを実行した場合、リポジトリを LDAP サーバーと同期化するよう選択できます。

LDAP ユーザーとグループをリポジトリにインポートするには

1. リポジトリを Administration Tool でオフライン・モードかオンライン・モードで開きます。
2. アプリケーション・メニューから「Manage」→「Security」を選択します。
3. Security Manager の左ペインで「LDAP Servers」を選択し、右ペインに既存の LDAP サーバーを表示します。ユーザーやグループのインポート元 LDAP サーバーを選択し、右クリック・メニューから「Import...」を選択します。サーバーを選択してから「LDAP」→「Import」を選択することもできます。

   選択したユーザーやグループ、またはすべてのユーザーとグループをインポートすることができます。以前にインポートを実行した場合、リポジトリを LDAP サーバーと同期化するよう選択できます。
4. インポート対象となるユーザーを選択し、「Import」をクリックします。

   グループをインポートするには、「Users」のかわりにドロップダウン・リストから「Groups」を選択します。

LDAP でのユーザーとグループの同期化

リポジトリのユーザーとグループを LDAP サーバー上の現在のユーザーとグループに対してリフレッシュできます。適切な LDAP サーバーを選択したら、「LDAP」→「Synchronize」を選択します（または右クリック・メニューから「Synchronize」を選択）。

同期化すると、リポジトリのユーザーとグループのリストが更新され、現在の LDAP ユーザーとグループがミラー化されます。LDAP サーバー上にないユーザーとグループは、リポジトリから削除されます。特別なユーザーである Administrator と特別なグループである Administrators は、常にリポジトリに残り削除されません。

リポジトリにすでにあるユーザーのプロパティは、同期化によって変更されません。別のユーザーのログイン名を再利用した場合、同期化前にその名前をリポジトリから削除してください。これによって LDAP の新しいユーザー定義のインポートが確実に行われます。

注意：外部 LDAP 認証（次の項で説明）では、インポートと同期化は実際には必要ありません。インポートを行う主な目的は、テストを行うために LDAP ユーザーを Oracle BI ユーザーとしてコピーする作業を容易にするためです。

認証オプション

認証とは、システムでユーザー ID とパスワードを使用して、ユーザーがログインしデータにアクセスする権限があり認可されていることをシステムが検証するプロセスのことです。Oracle BI Server では、受信するすべての接続リクエストの認証が行われます。

Oracle BI Server では次の認証タイプがサポートされています。

- LDAP 認証の設定（333 ページ）
- 外部テーブル認証の設定（335 ページ）
- データベース認証の設定（336 ページ）
- Oracle BI Delivers とデータベース認証について（337 ページ）
- Oracle BI Server のユーザー認証のメンテナンス（338 ページ）

LDAP 認証の設定

ユーザー ID とパスワードが Oracle BI リポジトリに格納されるかわりに、ユーザーが入力したユーザー ID とパスワードを LDAP サーバーに渡して認証を行うよう Oracle BI Server を設定できます。サーバーでは LDAP 認証でクリアテキスト・パスワードが使用されます。LDAP サーバーの設定でこれが許可されていることを確認します。

基本的なユーザー認証以外に、LDAP サーバーでは Oracle BI Server に対してユーザーの表示名（Oracle BI Presentation Services で使用）やユーザーが所属するグループの名前などの情報を提供することもできます。また、データのクエリーを行う際に各ユーザーが使用する特定のデータベース・カタログやスキーマの名前を提供することもできます。この情報は LDAP 変数に含まれており、ユーザー認証を処理する際に Oracle BI セッション変数に渡されます。セッション変数の詳細は、「セッション変数の概要と作成」（295 ページ）を参照してください。

LDAP 認証では Oracle BI セッション変数が使用されますが、これは Administration Tool の Variable Manager を使用して定義します。Variable Manager の詳細は、「Variable Manager の使用」（292 ページ）を参照してください。

次の手順に従って LDAP 認証を設定する必要があります。

1. Administration Tool のメニューで「Manage」→「Security」を選択し、LDAP サーバーを作成します。手順の詳細は、「LDAP サーバーの設定」（330 ページ）を参照してください。
2. LDAP 初期化ブロックを作成してから LDAP サーバーと関連付けます。LDAP 初期化ブロックの設定方法の詳細は、「初期化ブロックの作成プロセス」（301 ページ）を参照してください。
3. USER という名前のシステム変数を定義し、USER 変数を LDAP 属性（uid または sAMAccountName）にマッピングします。

   ユーザーがログオンしてセッションを開始すると、セッション変数には値が設定されます。システム・セッション変数というセッション変数には特別の使用方法があります。USER 変数は LDAP 認証で使用されるシステム変数です。USER システム変数の詳細は、「システム・セッション変数の使用」（295 ページ）と「LDAP 認証用の USER セッション変数の定義」（334 ページ）を参照してください。
4. 必要に応じて、ユーザーを Oracle BI リポジトリ・ファイルから削除します。
5. USER システム変数を LDAP 初期化ブロックと関連付けます。詳細は、「初期化ブロックを使用したユーザー認証について」（301 ページ）を参照してください。

LDAP 認証用の USER セッション変数の定義

LDAP 認証を設定するには、USER というシステム変数を定義してから、LDAP 初期化ブロック（LDAP サーバーと関連付けられている）と関連付けます。ユーザーが Oracle BI Server にログインすると、ユーザー ID とパスワードが LDAP サーバーに渡され認証が行われます。ユーザーの認証が成功すると、LDAP サーバーから返された情報でそのユーザーの他のセッション変数も移入できます。

注意：変数を同一ユーザーのためにリポジトリと LDAP サーバーの両方において作成すると、ローカルのリポジトリ・ユーザー定義が優先され、LDAP 認証は行われません。

この項の情報は、LDAP 初期化ブロックがすでに定義済であるということを前提にしています。

リポジトリに定義されていないユーザーは、USER セッション・システム変数が定義されているかどうかによって外部認証の実行が決まります。USER を LDAP 初期化ブロックに関連付けると、ユーザーの認証は LDAP で行われます。他の形式の認証を行うには、外部のデータベースや XML ソースと関連付けられた初期化ブロックに USER 変数を関連付けます。詳細は、「外部テーブル認証の設定」（335 ページ）を参照してください。

LDAP 認証用に USER セッション・システム変数を定義するには

1. Administration Tool のメニューから「Manage」→「Variables」を選択します。
2. 左ペインでツリーの「System」リーフを選択します。
3. 右ペインで右クリックし、「New USER」を選択します。
4. 「Session Variable - USER」ダイアログ・ボックスで「Initialization Block」ドロップダウン・リストから適切な LDAP 初期化ブロックを選択します。

   選択した初期化ブロックによって USER セッション・システム変数に値が設定されます。
5. 「OK」をクリックして USER 変数を作成します。

ロギング・レベルの設定

LOGLEVEL システム変数を使用して、LDAP サーバーに認証されるユーザーのロギング・レベルを設定します。詳細は、「ロギング・レベルの設定」（222 ページ）を参照してください。

外部テーブル認証の設定

ユーザー ID とパスワードを Oracle BI リポジトリに格納するかわりに、ユーザーとユーザーのパスワードを外部データベース・テーブルに保持して認証に使用することができます。この外部データベース・テーブルにはユーザー ID とパスワードが格納されますが、Oracle BI Presentation Service ユーザーのために使用するグループ・メンバーシップと表示名などの情報を含めることもできます。また、各ユーザーがデータのクエリーを実行する際に使用する特定のデータベース・カタログやスキーマの名前を含めることもできます。

注意：ユーザーが複数のグループに属している場合、それらのグループ名はセミコロンで区切って同じ列に記録する必要があります。

外部テーブル認証はデータベース認証とともに使用できます。外部テーブル認証が成功すると、データベース認証は行われません。外部テーブル認証が失敗すると、データベース認証は行われます。

詳細は、「データベース認証の設定」（336 ページ）と「認証の順序」（339 ページ）を参照してください。

外部テーブル認証では Oracle BI セッション変数が使用されますが、この変数は Administration Tool の Variable Manager を使用して定義します。Variable Manager の詳細は、「Variable Manager の使用」（292 ページ）を参照してください。

ユーザーがログオンしてセッションを開始すると、セッション変数には値が設定されます。システム変数というセッション変数には特別の使用方法があります。USER 変数は、外部テーブル認証で使用されるシステム変数です。

外部テーブル認証を設定するには、USER というシステム変数を定義して、外部データベース・テーブルと関連付けられている初期化ブロックと関連付けます。ユーザーがログインすると、ユーザー ID とパスワードの認証が、このデータベース・テーブルのクエリーを行う SQL を使用して行われます。ユーザーの認証が成功すると、ユーザーのその他のセッション変数も、SQL クエリーの結果により移入できます。セッション変数の詳細は、「セッション変数の概要と作成」（295 ページ）を参照してください。

USER システム変数を定義すると外部認証が行われます。USER を外部データベース・テーブルの初期化ブロックと関連付けると、ユーザーの認証がこのテーブルの情報を使用して行われます。認証を他の形式で行うには、USER システム変数を LDAP サーバーまたは XML ソースと関連付けられた初期化ブロックと関連付けます。詳細は、「LDAP 認証の設定」（333 ページ）を参照してください。

外部テーブル認証を設定するには

1. 外部テーブルの情報を「Physical」レイヤーにインポートします。この図では、データベース sql_nqsecurity には securitylogons という名前のテーブルがあり、External Table Security という名前の接続プールがあります。
2. 「Manage」→「Variables」を選択し、Variable Manager を起動します。
3. 左のツリー・ペインから「Initialization Blocks」を選択します。

4 右ペインの空欄で右クリックを行い、右クリック・メニューの「New Initialization Block」をクリックします。

5 「Initialization Block」ダイアログ・ボックスに、初期化ブロックの名前を入力します。

6 「Data Source Connection」ドロップダウン・リストから「Database」を選択します。

7 「Browse」をクリックして、このブロックで使用する接続プールの名前を検索します。

8 「Initialization String」領域で認証時に発行する SQL 文を入力します。

SQL 文を使用してデータベースから返された列値が、変数に割り当てられます。変数の順序と列の順序により、どの列をどの変数に割り当てるかが決まります。次の SQL 文を例に考えてみます。

```
select username, grp_name, SalesRep, 2 from securitylogons where username =
':USER' and pwd = ':PASSWORD'
```

この SQL 文には WHERE 句に絞込み条件が 2 つあります。

- :USER（コロンがあることに注意）は、ユーザーがログオン時に入力した ID です。
- :PASSWORD（ここでもコロンがあることに注意）は、ユーザーが入力したパスワードです。

ユーザー ID とパスワードが、指定されたテーブルにある値と一致した場合のみ、クエリーではデータが返されます。

SQL 文を Oracle BI Server の外部でテストし、:USER と :PASSWORD に有効な値が代入されて、データ行が返されることを確認してください。

9 このクエリーによりデータが返されると、ユーザーは認証されセッション変数に値が移入されます。このクエリーでは列が 4 つ返されるので、4 つのセッション変数が移入されます。ダイアログの「Variables」タブで「New」をクリックして、これらの変数（USER、GROUP、DISPLAYNAME および LOGLEVEL）を作成します。

変数が適切な順序で並んでいない場合、順序を変更する変数をクリックしてから、「Up」と「Down」のボタンを使用して移動します。

10 「OK」をクリックして、初期化ブロックを保存します。

データベース認証の設定

Oracle BI Server では、データベースへのログオンでユーザーを認証できます。ユーザーに特定のデータベースの読取り権限がある場合、ユーザーは Oracle BI Server によって信頼されます。オペレーティング・システムでの認証とは異なり、この認証は Oracle BI Presentation Services ユーザーに適用できます。詳細は、「Oracle BI Delivers とデータベース認証について」（337 ページ）を参照してください。

データベース認証は外部テーブル認証とともに使用できます。外部テーブル認証が成功すると、データベース認証は行われません。外部テーブル認証が失敗すると、データベース認証は行われます。

詳細は、「外部テーブル認証の設定」（335 ページ）と「認証の順序」（339 ページ）を参照してください。

データベース認証ではユーザー ID を Oracle BI リポジトリに格納する必要があります。

データベース認証を設定するには

1. データベースのユーザーと同じ名前のユーザーをリポジトリに作成します。パスワードはリポジトリには格納しません。
2. 必要となるユーザー権限（該当する場合、グループのメンバーシップを含む）をユーザーに割り当てます。
3. 認証データベースを NQSConfig.INI ファイルの Security セクションで指定します。

   詳細は、『Oracle Business Intelligence Infrastructure インストレーションおよび構成ガイド』を参照してください。
4. データベース用に DSN を作成します。
5. データベースを「Physical」レイヤーにインポートします。物理テーブル・オブジェクトはインポートする必要はありません。「Physical」レイヤーのデータベース名は、（手順 3 で示したように）NQSConfig.INI ファイルのデータベース名と一致する必要があります。
6. 接続プールを共有ログオンなしで設定します。

ユーザーが Oracle BI Server にログオンすると、サーバーではログオン名とパスワードを使用して、関連付けられている最初の接続プールを使用して認証データベースへの接続を試みます。接続が成功すると、ユーザーの認証は正常に行われたものとみなされます。

ログオンが拒否された場合、Oracle BI Server ではユーザー ID またはパスワードが無効であることを示すメッセージをユーザーに発行します。

Oracle BI Delivers とデータベース認証について

Oracle BI Applications では、ユーザーは必ず、稼動中のアプリケーション・データベースに作成されます。Oracle BI リポジトリには作成されません。Oracle BI リポジトリはデータベースの認証用に事前構成されています。

Oracle BI Scheduler Server では、アクセス権のないユーザーやパスワードを格納していないユーザーのために Delivers ジョブを実行します。Scheduler は偽装というプロセスを使用して、Oracle BI 管理者権限を持つユーザーの ID とパスワードを使用して他のユーザーのために処理を実行できます。Scheduler は、その Oracle BI 管理者の ID とパスワードを使用し、Oracle BI Presentation Services にログオンすることで iBot を起動します。

Delivers が機能するには、すべてのデータベース認証を単一の接続プールでのみ実行する必要があります。その接続プールは、USER システム・セッション変数の初期化ブロックでのみ選択できます。これは通常、認証初期化ブロックと呼ばれます。偽装が使用されると、この初期化ブロックはスキップされます。他のすべての初期化ブロックは、データベース認証を使用しない接続プールを使用する必要があります。

警告： 認証初期化ブロックは、接続プールを使用できる唯一の初期化ブロックで、接続プールでは :USER と :PASSWORD が物理データベースに渡されます。

他の初期化ブロックの場合、SQL 文では :USER と :PASSWORD を使用できます。ただし、Oracle BI Scheduler Server にはユーザーのパスワードは格納されないので、WHERE 句を次の例のように構成する必要があります。

```
select username, groupname, dbname, schemaname from users
where username=':USER'
NQS_PASSWORD_CLAUSE(and pwd=':PASSWORD')NQS_PASSWORD_CLAUSE
```

注意：偽装が使用されていると、カッコ内の変数はすべて実行時に SQL 文から抽出されます。

詳細は、「データソースと接続プールの選択およびテスト」（302 ページ）の Oracle BI Delivers の例を参照してください。

Oracle BI Server のユーザー認証のメンテナンス

Oracle BI リポジトリのユーザーとパスワードのリストは、Administration Tool を使用するとメンテナンスできます。ユーザーがログオンすると Oracle BI Server はこのリストでユーザーの認証を試みます。ただし、別の認証方法が正常に実行されたり、データベース認証が NQSConfig.INI ファイルで指定されている場合を除きます。

詳細は、「認証の順序」（339 ページ）を参照してください。

Oracle BI Server のユーザー ID は、暗号化されずに Oracle BI リポジトリに格納され、大文字と小文字は区別されません。パスワードは暗号化されて格納され、大文字と小文字は区別されます。ユーザーに必要な権限があると、Oracle BI Server ユーザー ID を使用してリポジトリにあるビジネス・モデルにアクセスできます。ユーザー ID はユーザー ID を設定したリポジトリにおいてのみ有効です。他のリポジトリでは使用できません。

注意：LDAP または外部テーブルの認証を使用している場合、パスワードは Oracle BI リポジトリには格納されません。

ユーザー認証の構成方法の詳細は、『Oracle Business Intelligence Enterprise Edition デプロイメント・ガイド』を参照してください。

Oracle BI Server ユーザーのパスワードの変更

ユーザーのパスワードを Administration Tool で変更できます。

ユーザーのパスワードを変更するには

1. 「Manage」→「Security」を選択します。
2. 「Security Manager」ダイアログ・ボックスの左ペインで「User」を選択します。
3. 右ペインで、変更対象となるユーザーのパスワードを右クリックします。
4. ショートカット・メニューから「Properties」を選択します。
5. 「User」タブで、新しいパスワードを入力します。
6. 「Confirm Password」テキスト・ボックスにパスワードを再び入力してから、「OK」をクリックします。

認証の順序

ユーザーがログオン名を入力しない場合、OS 認証が NQSConfig.INI ファイルにおいて明示的にオフに設定されている場合を除いて、OS 認証がトリガーされます。詳細は、『Oracle Business Intelligence Enterprise Edition デプロイメント・ガイド』を参照してください。また OS 認証は Oracle BI Presentation Services ユーザーには使用されません。

Oracle BI Server では初期化ブロックを使用してセッション変数が移入されます。その移入順序は、初期化ブロックに定義されている依存性ルールにより指定された順序です。サーバーでは USER セッション変数を検索して、USER 変数が関連付けられている初期化ブロックの構成に応じて、LDAP サーバーまたは外部データベース・テーブルに対して認証を行います。

Oracle BI Server の内部認証（または、データベース認証）は、これらの可能性を考慮した後でのみ実行されます。

クエリーの実行権限の管理

Oracle BI Server では、ユーザーのリポジトリ情報へのアクセスをどの程度許可するかを管理できます。

クエリー権限を制御すると、クエリー環境を管理できます。ユーザーに対するクエリーの使用状況は、高度に管理することも、管理しないことも、ある程度管理することもできます。次のリストで、制限を設定できるいくつかのアクティビティを示します。

- 特定のオブジェクト（行や列など）または特定の期間におけるクエリー・アクセスの制限
  - **オブジェクト**の場合：子オブジェクトを持つオブジェクトへのアクセスを明示的に拒否すると、ユーザーは子オブジェクトへのアクセスが拒否されます。たとえば、特定の物理データベース・オブジェクトへのアクセスを拒否すると、そのカタログ内のすべての物理テーブルと物理列へのアクセスは暗黙的に拒否されます。

    ユーザーまたはグループに対し、あるオブジェクトに対する権限が複数のソースから付与または拒否されている場合（たとえば、1 つ以上のグループから明示的に権限が割り当てられているなど）、「グループの継承」（327 ページ）で説明しているように、権限は優先順位に従って使用されます。

    データベース・リクエストをユーザーまたはグループが直接実行する権限を許可または禁止できます。
  - **期間**の場合：期間を選択しない場合、アクセス権は変更されません。1 つ以上のグループに対しアクセスを明示的に許可または拒否した場合、定義した期間の間、ユーザーは最も制限が少ないアクセス権が付与されます。たとえば、すべての月曜日に明示的にアクセスを許可されているユーザーが、毎日 24 時間アクセスが禁止されているグループに属していると仮定します。このユーザーは月曜日のみアクセスできることになります。
- クエリーの行数や最大実行回数の制限によるリソース集中型のクエリーの制御
- オブジェクトに対してフィルタを設定することによるクエリーの制限

すべての制限と制御は、ユーザー・レベル、グループ・レベルまたはこの両方の組合せで適用できます。

ユーザーやグループに対しオブジェクトごとにクエリーを制限するには

1 Administration Tool のメニュー・バーから、「Manage」→「Security」を選択します。

2 「Security Manager」ダイアログ・ボックスのツリー・ペインで「Users」または「Groups」を選択します。

3 右ペインで、変更する名前を右クリックしてから「Properties」を選択します。

4 「User」ダイアログ・ボックスまたは「Group」ダイアログ・ボックスで、「Permissions」をクリックします。

5 「User/Group Permissions」ダイアログ・ボックスで、「General」タブをクリックし、次の手順を実行します。

   a リポジトリ内の 1 つ以上のオブジェクトに対し、アクセスを明示的に許可または禁止するには、「General」タブで「Add」をクリックします。

   b 「Browse」ダイアログ・ボックスの「Name」リストで、変更するオブジェクトを選択し「Select」をクリックします。

   c 「User/Group Permissions」ダイアログ・ボックスで、各オブジェクトに対応する「Read」チェック・ボックスを選択または解除して権限を割り当てます。

      チェック・ボックスが選択されている場合（デフォルトは選択）、ユーザーにはオブジェクトの読取り権限があります。チェック・ボックスに X の印がある場合、ユーザーにはオブジェクトの読取り権限がありません。空欄の場合、そのオブジェクトに対する既存の権限（たとえばグループを介した権限）が適用されます。

   権限の割当て方法の詳細は、「リポジトリ・オブジェクトに対する権限の設定」（35 ページ）を参照してください。

6 移入権限、または特定のデータベース・オブジェクトを直接データベースにリクエストする権限を、明示的に許可または禁止するには、次の手順を実行します。

   a 「Query Limits」タブをクリックし、データベースを選択します。

   b 「Populate Privilege」ドロップダウン・リストで「Allow」または「Disallow」を選択します。

      注意： これによって、選択したユーザーまたはグループに対する「Allow populate queries for all」のデータベース・プロパティが無視されます。

   c 直接データベースに特定のデータベース・オブジェクトをリクエストすることを明示的に許可または禁止するには、「Execute Direct Database Requests」ドロップダウン・リストで「Allow」または「Disallow」を選択します。

      注意： これによって、選択したユーザーまたはグループに対する「Allow direct database requests for all」のデータベース・プロパティが無視されます。

7 「OK」を 2 回クリックして、「Security Manager」ダイアログ・ボックスに戻ります。

ユーザーやグループから受信するクエリーを行数で制限するには

1 Administration Tool のメニュー・バーから、「Manage」→「Security」を選択します。

2 「Security Manager」ダイアログ・ボックスのツリー・ペインで「Users」または「Groups」を選択します。

3 右ペインで、変更する名前を右クリックしてから「Properties」を選択します。

4 「User」ダイアログ・ボックスまたは「Group」ダイアログ・ボックスで、「Permissions」タブをクリックします。

5 「User/Group Permissions」ダイアログ・ボックスで、「Query Limits」タブをクリックしてからダイアログ・ボックスを展開し、すべての列を表示します。

6 各クエリーがデータベースから取得する行の最大行数を指定または変更するには、「Query Limits」タブで次の手順を実行します。

   a 「Max Rows」列で、最大行数を入力します。

   b 「Status Max Rows」フィールドで 342 ページの表 39 を参考にしてステータスを選択します。

7 「OK」を 2 回クリックして、「Security Manager」ダイアログ・ボックスに戻ります。

ユーザーやグループに対しクエリーの最大実行回数や期間を制限するには

1 Administration Tool のメニュー・バーから、「Manage」→「Security」を選択します。

2 「Security Manager」ダイアログ・ボックスのツリー・ペインで「Users」または「Groups」を選択します。

3 右ペインで、変更する名前を右クリックしてから「Properties」を選択します。

4 「User」ダイアログ・ボックスまたは「Group」ダイアログ・ボックスで、「Permissions」タブをクリックします。

5 「User/Group Permissions」ダイアログ・ボックスで、「Query Limits」タブをクリックしてからダイアログ・ボックスを展開し、すべての列を表示します。

6 クエリーをデータベースに対して実行できる最長期間を指定するには、「Query Limits」タブで次の手順を実行します。

   a 「Max Time」列で、分単位の時間を選択します。

   b 「Status Max Time」ドロップダウン・リストで 342 ページの表 39 を参考にしてステータスを選択します。

7 特定の期間の間データベースへのアクセスを制限するには、「Restrict」列で省略記号ボタンをクリックします。

8 「Restrictions」ダイアログ・ボックスで次の手順を実行します。

   a 期間を選択するには、開始時間をクリックしてから終了時間までドラッグします。

   b アクセスを明示的に許可するには、「Allow」をクリックします。

   c アクセスを明示的に禁止するには、「Disallow」をクリックします。

9 「OK」を 2 回クリックして、「Security Manager」ダイアログ・ボックスに戻ります。

ユーザーやグループに対し、オブジェクトにフィルタを設定してクエリーを制限するには

1 Administration Tool のメニュー・バーから、「Manage」→「Security」を選択します。

2 「Security Manager」ダイアログ・ボックスのツリー・ペインで「Users」または「Groups」を選択します。

3 右ペインで、変更する名前を右クリックしてから「Properties」を選択します。

4 「User」ダイアログ・ボックスまたは「Group」ダイアログ・ボックスで、「Permissions」をクリックします。

5 「User/Group Permissions」ダイアログ・ボックスで、「Filters」タブをクリックします。

6 「Filters」タブで、次の手順を実行してオブジェクトにフィルタを追加します。

   a 「Add」をクリックします。

   b 「Browse」ダイアログ・ボックスの「Names」リストから、フィルタされるオブジェクトを検索してからダブルクリックします。

   c オブジェクトを選択し、「Select」をクリックします。

7 「User/Group Permissions Filters」ダイアログ・ボックスで、次の手順を実行します。

   a 右にスクロールし「Business Model Filter」列を表示します。

   b 選択したオブジェクトに対し「Business Model Filter」省略記号ボタンをクリックします。

8 「Expression Builder」ダイアログ・ボックスで論理フィルタを作成してから「OK」をクリックします。

9 「User/Group Permissions Filters」ダイアログ・ボックスの「Status」ドロップダウン・リストから、342 ページの表 39 を参考にしてステータスを選択します。

10 「OK」を 2 回クリックして、「Security Manager」ダイアログ・ボックスに戻ります。

表 39. クエリー権限のステータス・フィールド

ステータス	説明
Disable	■ 「**Status Max Rows**」または「**Status Max Time**」: 選択すると、「Max Rows」または「Max Time」のフィールドに設定されている制限がすべて無効になります。 ■ **Filter**: フィルタは使用されず、オブジェクトには（たとえばグループを介して）優先順位の高いレベルで適用されている他のフィルタも使用されません。
Enable	■ 「**Status Max Rows**」または「**Status Max Time**」: これによって、指定した値まで行数または期間が制限されます。行数が「Max Rows」の値を超えると、クエリーは終了します。 ■ **Filter**: オブジェクトにアクセスするすべてのクエリーにフィルタが適用されます。
Ignore	■ 「**Status Max Rows**」または「**Status Max Time**」: 親グループから制限を継承します。継承する制限がない場合、制限は適用されません。 ■ **Filter**: フィルタは使用されませんが、オブジェクトに適用される（たとえばグループを介しての）他のフィルタは使用されます。これ以外のフィルタが有効になっていない場合、フィルタ処理は行われません。

ユーザーまたはグループへの移入権限の割当て

条件ブロックがキャッシュされている場合、Populate Stored プロシージャによって Cache/Saved Result Set 値がデータベースに書き込まれます。

注意： キャッシュ・エントリを書き込んだり結果セットを保存するすべての Marketing ユーザーに、対象となるデーターベースの POPULATE 権限を割り当てる必要があります。Marketing セグメントのすべてのユーザーとグループに、この権限を割り当てる必要があります。通常、すべての Marketing ユーザーはこの権限が付与されているグループに関連付けられます。マーケティング・キャッシュの詳細は、Oracle Siebel Marketing アプリケーションのドキュメントでターゲット・レベルのキャッシュの設定に関する項を参照してください。

移入権限をユーザーやグループに割り当てるには

1. Administration Tool のメニュー・バーから、「Manage」→「Security」を選択します。
2. 「Security Manager」ダイアログ・ボックスのツリー・ペインで「Users」または「Groups」を選択します。
3. 右ペインで、変更する名前を右クリックしてから「Properties」を選択します。
4. 「User」ダイアログ・ボックスまたは「Group」ダイアログ・ボックスで、「Permissions」をクリックします。
5. 「User/Group Permissions」ダイアログ・ボックスで、「Query Limits」タブをクリックします。
6. ダイアログ・ボックスを開いて「Query Limits」リストのすべての列を表示します。
7. 「Populate Privilege」ドロップダウン・リストで「Allow」または「Disallow」を選択します。

   注意：すべての Marketing データ・ウェアハウスの「Populate Privilege」を「Allow」に設定します。
8. 「OK」を 2 回クリックして、「Security Manager」ダイアログ・ボックスに戻ります。

16 Oracle BI Server のデータソースとしての XML の使用

この章では、eXtensible Markup Language（XML）をデータソースとして使用する方法について説明します。XML は、Web 上の構造化された文書とデータ用の汎用フォーマットです。これは、構造化されたデータを保存するためのデータベースとしても使用できます。

Oracle BI Server では、Oracle BI Server XML Gateway とその拡張機能の Data Mining Adapter を介したアクセス、および XML ODBC ドライバを介したアクセスを含めて、様々な XML アクセス・モードがサポートされています。

この章の内容は次のとおりです。

XML URL の特定

Oracle BI Server では、リポジトリの「Physical」レイヤーのデータソースとして XML データの使用がサポートされています。XML データソースのアクセスに使用する方法によっては、データソースは次のいずれかのソースを指す URL で表す場合があります。

- インターネット（イントラネットやエクストラネットを含む）上の XML データ・アイランドを含む静的な XML ファイルまたは HTML ファイル。次に例を示します。
  tap://216.217.17.176/[DE0A48DE-1C3E-11D4-97C9-00105AA70303].XML
- サーバー・サイトで生成される動的 XML。次に例を示します。
  tap://www.aspserver.com/example.asp
- ローカル・ドライブまたはネットワーク・ドライブ上の XML データ・アイランドを含む XML ファイルまたは HTML ファイル。次に例を示します。
  d:¥xmldir¥example.xml
  d:¥htmldir¥island.htm

  ローカルまたはネットワーク上の XML ファイルのディレクトリ・パスを指定したり、ファイル名のワイルドカードとしてアスタリスク（*）を使用することもできます。ディレクトリ・パスをファイル名なしで指定すると（d:¥xmldir など）、拡張子が XML であるファイルがすべてインポートされます。次に例を示します。
  d:¥xmldir¥
  d:¥xmldir¥exam*.xml
  d:¥htmldir¥exam*.htm
  d:¥htmldir¥exam*.html
- *<table>* タグと *</table>* タグのペアで定義されたテーブルを含む HTML ファイル。この HTML ファイルは、インターネット（イントラネットやエクストラネットを含む）上、またはローカル・ドライブやネットワーク・ドライブ上に配置することができます。詳細は、「HTML テーブルへのアクセス」（354 ページ）を参照してください。

URL には、リポジトリまたはセッション変数を含めることができます。これにより、受け入れるユーザー ID とパスワードを URL に埋め込む HTTP データソースがサポートされます。たとえば、http://somewebserver/cgi.pl?userid=valueof(session_variable1)&password= valueof(session_variable2) と指定できます。この機能により、Oracle BI 管理者は、実行時パラメータで動的に決まる場所を使用して XML データソースを作成することもできます。変数の詳細は、第 13 章「Oracle BI リポジトリの変数の使用」を参照してください。

Oracle BI Server では、HTML ページの XML データ・アイランドや XML ファイルを変換する XSL 変換ファイル（XSLT）または XPath 式の使用もサポートされています。

XSLT は、HTML 文書用のカスケード・スタイル・シート（CSS）が XML 文書またはテキストのフラグメントに適用されるよう汎用的にしたものです。XPath は、1 行の文で表現できる XSLT の簡易版です。たとえば、XPath 式の //xml により、XML プロセッサではルート要素の xml 配下にあるすべての要素が抽出されます。XSLT ファイルにも XPath 式を含めることができます。

注意： Oracle BI Server でローカルにないファイル（ネットワーク・ファイル、インターネット上のファイルなど）にアクセスする必要がある場合、それらのリモート・ファイルへのアクセスに必要な権限を持つ有効なユーザー ID とパスワードを使用して、Oracle BI Server を実行する必要があります。Windows NT と Windows 2000 では、このユーザー ID はローカル・マシンの Windows 管理者権限も持つ必要があります。サーバー実行時に使用するアカウントを変更するには、「Oracle BI Server を実行するユーザー ID の変更」（217 ページ）の説明にある手順を実行します。

Oracle BI Server XML Gateway の使用

Oracle BI Server XML Gateway を使用して、メタデータのインポート処理で XML 文書をテーブル形式にフラット化します。その際、XML ファイル名の本体（ファイル名の拡張子以外の部分）をテーブル名に使用し、XML 文書の第 2 レベルの要素を行のデリミタに使用します。すべてのリーフ・ノードは、テーブルに属する列としてインポートされます。リーフ・ノードへの階層アクセス・パスもインポートされます。

Oracle BI Server XML Gateway では、XML スキーマが含まれるメタデータ情報を使用します。XML スキーマは、XML 文書に含まれるか、XML 文書のルート要素の中で参照されます。現時点では、Microsoft 社によって定義され Internet Explorer 5 ブラウザのファミリに実装されている XML スキーマのバージョンのみサポートされています。

有効なスキーマがない場合、すべての XML データはテキスト・データとしてインポートされます。リポジトリを構築する際に、「Physical」レイヤーの列のデータ型を変更して、スキーマで定義された対応する列のデータ型より優先することができます。ゲートウェイにより、入力データは、「Physical」レイヤーで指定した目的の型に変更されます。また、Administration Tool の「Business Model and Mapping」レイヤーで CAST 演算子を使用して、テキストのデータ型を他のデータ型にマップすることもできます。

現時点では、Oracle BI Server XML Gateway では次はサポートされていません。

- XML 文書に含まれる外部参照の解決（「Oracle BI Server XML Gateway の例」（348 ページ）のサンプル・ファイルの説明にある、外部 XML スキーマへの参照を除く）
- Microsoft XML スキーマに含まれる要素と属性の継承
- 混合コンテンツ・モデルの要素タイプ（要素と CDATA が混在する XML 要素など。たとえば hello Joe, how are you doing? ）

注意： Oracle BI Server XML Gateway には、Data Mining Adapter が含まれます。これを使用すると、取得した各レコードに対し実行可能ファイルや DLL をコールしてデータソースにアクセスできます。詳細は、「Data Mining Adapter の使用」（356 ページ）を参照してください。

Oracle BI Server XML Gateway を使用して XML データをインポートするには

1. Administration Tool のツールバーから「File」→「Import」を選択します。

   「Select ODBC Data Source」ダイアログ・ボックスが表示されます。

2. 「Connection Type」ドロップダウン・リストから「XML」を選択します。

   「Connection Type」が「XML」に設定された状態で、「Type In Uniform Resource Locator」ダイアログ・ボックスが表示されます。

3. 「URL」フィールドで、XML データソースを指定します。

   Oracle BI Server XML Gateway では、「XML URL の特定」（346 ページ）に説明のある、すべてのデータソースがサポートされます。

   URL にはリポジトリ変数またはセッション変数を含めることができます。「Browse」ボタンをクリックすると、「Select XML File」ダイアログ・ボックスが表示されます。ここで、ファイルを 1 つ選択できます。変数の詳細は、第 13 章「Oracle BI リポジトリの変数の使用」を参照してください。

4 オプションで、eXtensible Stylesheet Language Transformation（XSLT）ファイルまたは XPath 式を入力します。

「Browse」ボタンを使用して、XSLT ソース・ファイルを参照します。

5 HTTP Basic 認証のセキュリティ・モードを使用する HTTP サイトに接続する場合、適切なフィールドにオプションのユーザー ID とパスワードを入力します。

HTTP Basic 認証のセキュリティ・モードに加えて、Oracle BI Server XML Gateway では、Secure HTTP プロトコルと統合 Windows 認証（Windows 2000 用で、旧称は NTLM または Windows NT チャレンジ / レスポンス認証）もサポートされています。

6 「OK」をクリックすると、「Import」ダイアログ・ボックスが開きます。

7 テーブルと列を選択してから、インポートするメタデータのタイプを選択します。

デフォルト設定では、すべてのオブジェクトとすべてのメタデータがインポートされます。

8 「Import」をクリックすると、インポート処理が開始します。

9 「Connection Pool」ダイアログ・ボックスの「General」タブで、名前を入力し、必要に応じて接続の説明を入力します。詳細は、「接続プールの設定」（69 ページ）を参照してください。

10 「XML」タブをクリックして、さらに別の接続プロパティ（URL リフレッシュ間隔や URL ロードのタイムアウト待ち時間など）を設定します。

通常、XML データソースはリアルタイムで頻繁に更新されるため、Oracle BI Server XML Gateway でこれらのデータソースのリフレッシュ間隔を指定できます。

詳細は、「XML データソースのリフレッシュ間隔について」（266 ページ）を参照してください。

クエリーのタイムアウト時間（URL ロードにおけるタイムアウト）は、デフォルトでは 15 分です。

11 「OK」をクリックすると、インポートが完了します。

12 XML データソースを詳細に制御するために、「Physical Table」ダイアログ・ボックスでデータソースの個々のテーブルに対して XSLT ファイルまたは XPath 式を指定できます。これらのエントリが指定されると、これを使用して個々の物理テーブルに対して接続プールの対応する XSLT または XPath エントリが上書きされます。

Oracle BI Server XML Gateway の例

XML データの次のサンプル文書（mytest.xml）では、外部ファイルに含まれる XML スキーマを参照しています。スキーマ・ファイルはデータ文書の後に示します。リポジトリへのインポートに使用可能な生成済 XML スキーマ情報を最後に示します。

```
<?xml version="1.0"?>
<test xmlns="x-schema:mytest_sch.xml">

<row>
<p1>0</p1>
<p2 width="5">
      <p3>hi</p3>
      <p4>
         <p6>xx0</p6>
```

```
            <p7>yy0</p7>
         </p4>
         <p5>zz0</p5>
   </p2>
   </row>

   <row>
   <p1>1</p1>
   <p2 width="6">
         <p3>how are you</p3>
         <p4>
            <p6>xx1</p6>
            <p7>yy1</p7>
         </p4>
         <p5>zz1</p5>
   </p2>
   </row>

   <row>
   <p1>a</p1>
   <p2 width="7">
         <p3>hi</p3>
         <p4>
            <p6>xx2</p6>
            <p7>yy2</p7>
         </p4>
         <p5>zz2</p5>
   </p2>
   </row>

   <row>
   <p1>b</p1>
   <p2 width="8">
         <p3>how are they</p3>
         <p4>
            <p6>xx3</p6>
            <p7>yy3</p7>
         </p4>
         <p5>zz2</p5>
   </p2>
   </row>
   </test>
```

次に対応するスキーマ・ファイルを示します。

```
<Schema xmlns="urn:schemas-microsoft-com:xml-data"
      xmlns:dt="urn:schemas-microsoft-com:datatypes">
      <ElementType name="test" content="eltOnly" order="many">
         <element type="row"/>
      </ElementType>
      <ElementType name="row" content="eltOnly" order="many">
      <element type="p1"/>
         <element type="p2"/>
      </ElementType>
```

```
      <ElementType name="p2" content="eltOnly" order="many">
         <AttributeType name="width" dt:type="int" />
         <attribute type="width" />
         <element type="p3"/>
         <element type="p4"/>
         <element type="p5"/>
      </ElementType>
      <ElementType name="p4" content="eltOnly" order="many">
         <element type="p6"/>
         <element type="p7"/>
      </ElementType>
      <ElementType name="p1" content="textOnly" dt:type="string"/>
      <ElementType name="p3" content="textOnly" dt:type="string"/>
      <ElementType name="p5" content="textOnly" dt:type="string"/>
      <ElementType name="p6" content="textOnly" dt:type="string"/>
      <ElementType name="p7" content="textOnly" dt:type="string"/>
</Schema>
```

前述の XML データ文書（mytest.xml）から生成されるテーブルの名前は mytest となり、列名は p1、p3、p6、p7、p5 および width となります。

また、文書の各列で出現するコンテキストを保存し、異なるコンテキストで出現する同じ名前の XML 要素から導出される複数の列を区別するため、World Wide Web Consortium の XPath に関する提案に基づいて、次のように列の完全修飾名のリストが生成されます。

```
//test/row/p1
//test/row/p2/p3
//test/row/p2/p4/p6
//test/row/p2/p4/p7
//test/row/p2/p5
//test/row/p2@width
```

次の例は、XML 文書でネストされたテーブル構造の使用方法を示す複雑な例です。オプションで、外部スキーマ・ファイルへの参照を省略することもできます。この場合、すべての要素は Varchar 文字型として処理されます。

```
===Invoice.xml===
<INVOICE>
   <CUSTOMER>
      <CUST_ID>1</CUST_ID>
      <FIRST_NAME>Nancy</FIRST_NAME>
      <LAST_NAME>Fuller</LAST_NAME>
      <ADDRESS>
         <ADD1>507 - 20th Ave. E.,</ADD1>
         <ADD2>Apt.2A</ADD2>
         <CITY>Seattle</CITY>
         <STATE>WA</STATE>
         <ZIP>98122</ZIP>
      </ADDRESS>
      <PRODUCTS>
           <CATEGORY>
              <CATEGORY_ID>CAT1</CATEGORY_ID>
              <CATEGORY_NAME>NAME1</CATEGORY_NAME>
              <ITEMS>
```

```
                <ITEM>
                   <ITEM_ID>1</ITEM_ID>
                   <NAME></NAME>
                   <PRICE>0.50</PRICE>
                   <QTY>2000</QTY>
                </ITEM>
                <ITEM>
                   <ITEM_ID>2</ITEM_ID>
                   <NAME>SPRITE</NAME>
                   <PRICE>0.30</PRICE>
                   <QTY></QTY>
                </ITEM>
             </ITEMS>
          </CATEGORY>
            <CATEGORY>
             <CATEGORY_ID>CAT2</CATEGORY_ID>
             <CATEGORY_NAME>NAME2</CATEGORY_NAME>
             <ITEMS>
                <ITEM>
                   <ITEM_ID>11</ITEM_ID>
                   <NAME>ACOKE</NAME>
                   <PRICE>1.50</PRICE>
                   <QTY>3000</QTY>
                </ITEM>
                <ITEM>
                   <ITEM_ID>12</ITEM_ID>
                   <NAME>SOME SPRITE</NAME>
                   <PRICE>3.30</PRICE>
                   <QTY>2000</QTY>
                </ITEM>
             </ITEMS>
          </CATEGORY>
      </PRODUCTS>
</CUSTOMER>
<CUSTOMER>
      <CUST_ID>2</CUST_ID>
      <FIRST_NAME>Andrew</FIRST_NAME>
      <LAST_NAME>Carnegie</LAST_NAME>
      <ADDRESS>
         <ADD1>2955 Campus Dr.</ADD1>
         <ADD2>Ste.300</ADD2>
         <CITY>San Mateo</CITY>
         <STATE>CA</STATE>
         <ZIP>94403</ZIP>
      </ADDRESS>
      <PRODUCTS>
           <CATEGORY>
             <CATEGORY_ID>CAT22</CATEGORY_ID>
             <CATEGORY_NAME>NAMEA1</CATEGORY_NAME>
             <ITEMS>
                <ITEM>
                   <ITEM_ID>122</ITEM_ID>
                   <NAME>DDDCOKE</NAME>
                   <PRICE>11.50</PRICE>
```

```
                    <QTY>2</QTY>
                </ITEM>
                <ITEM>
                    <ITEM_ID>22</ITEM_ID>
                    <NAME>PSPRITE</NAME>
                    <PRICE>9.30</PRICE>
                    <QTY>1978</QTY>
                </ITEM>
            </ITEMS>
        </CATEGORY>
          <CATEGORY>
            <CATEGORY_ID>CAT24</CATEGORY_ID>
            <CATEGORY_NAME>NAMEA2</CATEGORY_NAME>
            <ITEMS>
                <ITEM>
                    <ITEM_ID>19</ITEM_ID>
                    <NAME>SOME COKE</NAME>
                    <PRICE>1.58</PRICE>
                    <QTY>3</QTY>
                </ITEM>
                <ITEM>
                    <ITEM_ID>15</ITEM_ID>
                    <NAME>DIET SPRITE</NAME>
                    <PRICE>9.30</PRICE>
                    <QTY>12000</QTY>
                </ITEM>
            </ITEMS>
        </CATEGORY>


    </PRODUCTS>
</CUSTOMER>
<CUSTOMER>
    <CUST_ID>3</CUST_ID>
    <FIRST_NAME>Margaret</FIRST_NAME>
    <LAST_NAME>Leverling</LAST_NAME>
    <ADDRESS>
        <ADD1>722 Moss Bay Blvd.</ADD1>
        <ADD2> </ADD2>
        <CITY>Kirkland</CITY>
        <STATE>WA</STATE>
        <ZIP>98033</ZIP>
    </ADDRESS>
    <PRODUCTS>
          <CATEGORY>
            <CATEGORY_ID>CAT31</CATEGORY_ID>
            <CATEGORY_NAME>NAMEA3</CATEGORY_NAME>
            <ITEMS>
                <ITEM>
                    <ITEM_ID>13</ITEM_ID>
                    <NAME>COKE33</NAME>
                    <PRICE>30.50</PRICE>
                    <QTY>20033</QTY>
                </ITEM>
```

```
                    <ITEM>
                        <ITEM_ID>23</ITEM_ID>
                        <NAME>SPRITE33</NAME>
                        <PRICE>0.38</PRICE>
                        <QTY>20099</QTY>
                    </ITEM>
                </ITEMS>
            </CATEGORY>
              <CATEGORY>
                <CATEGORY_ID>CAT288</CATEGORY_ID>
                <CATEGORY_NAME>NAME H</CATEGORY_NAME>
                <ITEMS>
                    <ITEM>
                        <ITEM_ID>19</ITEM_ID>
                        <NAME>COLA</NAME>
                        <PRICE>1.0</PRICE>
                        <QTY>3</QTY>
                    </ITEM>
                    <ITEM>
                        <ITEM_ID>18</ITEM_ID>
                        <NAME>MY SPRITE</NAME>
                        <PRICE>8.30</PRICE>
                        <QTY>123</QTY>
                    </ITEM>
                </ITEMS>
            </CATEGORY>
        </PRODUCTS>
    </CUSTOMER>
</INVOICE>
```

次に示すように生成される XML スキーマは、次の列名と対応する完全修飾名を持つ 1 つのテーブル（INVOICE）で構成されます。

列	完全修飾名
ADD1	//INVOICE/CUSTOMER/ADDRESS/ADD1
ADD2	//INVOICE/CUSTOMER/ADDRESS/ADD2
CITY	//INVOICE/CUSTOMER/ADDRESS/CITY
STATE	//INVOICE/CUSTOMER/ADDRESS/STATE
ZIP	//INVOICE/CUSTOMER/ADDRESS/ZIP
CUST_ID	//INVOICE/CUSTOMER/CUST_ID
FIRST_NAME	//INVOICE/CUSTOMER/FIRST_NAME
LAST_NAME	//INVOICE/CUSTOMER/LAST_NAME
CATEGORY_ID	//INVOICE/CUSTOMER/PRODUCTS/CATEGORY/CATEGORY_ID
CATEGORY_NAME	//INVOICE/CUSTOMER/PRODUCTS/CATEGORY/CATEGORY_NAME
ITEM_ID	//INVOICE/CUSTOMER/PRODUCTS/CATEGORY/ITEMS/ITEM/ITEM_ID

列	完全修飾名
NAME	//INVOICE/CUSTOMER/PRODUCTS/CATEGORY/ITEMS/ITEM/NAME
PRICE	//INVOICE/CUSTOMER/PRODUCTS/CATEGORY/ITEMS/ITEM/PRICE
QTY	//INVOICE/CUSTOMER/PRODUCTS/CATEGORY/ITEMS/ITEM/QTY

値を持つタグのみが列として抽出されます。XML クエリーではタグの完全修飾名が生成されます。これにより、適切な列が取得されたことを確認することができます。

次に、INVOICE テーブルに対するサンプル・クエリーの結果を示します。

```
select first_name, last_name, price,  qty, name from invoice
-------------------------------------------------------------
FIRST_NAME    LAST_NAME          PRICE    QTY    NAME
-------------------------------------------------------------
Andrew        Carnegie            1.58      3    SOME COKE
Andrew        Carnegie           11.50      2    DDDCOKE
Andrew        Carnegie            9.30   12000   DIET SPRITE
Andrew        Carnegie            9.30    1978   PSPRITE
Margar        Leverling           0.38   20099   SPRITE33
Margar        Leverling           1.0        3   COLA
Margar        Leverling          30.50   20033   COKE33
Margar        Leverling           8.30     123   MY SPRITE
Nancy         Fuller              0.30           SPRITE
Nancy         Fuller              0.50     2000
Nancy         Fuller              1.50     3000   ACOKE
Nancy         Fuller              3.30     2000   SOME SPRITE
-------------------------------------------------------------
Row count: 12
```

HTML テーブルへのアクセス

Oracle BI Server XML Gateway では、HTML ファイルのテーブルをデータソースとして使用することもサポートされています。HTML ファイルは、インターネット（イントラネットやエクストラネットを含む）上のファイルを指す URL として指定でき、またローカル・ドライブまたはネットワーク・ドライブ上のファイルとしても指定できます。

<table> と *</table>* のタグのペアで定義されたテーブルが HTML 4.0 仕様のネイティブ構成であっても、Web デザイナはこれを単なるデータ構造としてではなく一般的なフォーマット・デバイスとして使用し、特定の視覚効果を実現できます。Oracle BI Server XML Gateway は、一般的には、*<th>* と *</th>* のタグのペアで定義された特定の列ヘッダーを含むテーブルの抽出に最も効果的です。

特定の列ヘッダーを含まないテーブルに対して、Oracle BI Server XML Gateway では単純な経験則を使用して、実際のデータ・テーブルのように HTML ファイルの部分を特定するよう処理します。

次に 1 つのテーブルを含む HTML ファイルのサンプルを示します。

```
<html>
   <body>
      <table border=1 cellpadding=2 cellspacing=0>
         <tr>
            <th colspan=1>Transaction</th>
            <th colspan=2>Measurements</th>
         </tr>
         <tr>
            <th>Quality</th>
            <th>Count</th>
            <th>Percent</th>
         </tr>
         <tr>
            <td>Failed</td>
            <td>66,672</td>
            <td>4.1%</td>
         </tr>
         <tr>
            <td>Poor</td>
            <td>126,304</td>
            <td>7.7%</td>
         </tr>
         <tr>
            <td>Warning</td>
            <td>355,728</td>
            <td>21.6%</td>
         </tr>
         <tr>
            <td>OK</td>
            <td>1,095,056</td>
            <td>66.6%</td>
         </tr>
         <tr>
            <td colspan=1>Grand Total</td>
            <td>1,643,760</td>
            <td>100.0%</td>
         </tr>
      </table>
   </body>
</html>
```

テーブル名は HTML ファイル名から導出します。列名は、対応する列（*<th>* と *</th>* のタグのペアで定義）の見出しを連結して形成します。連結の区切りには、アンダースコアを使用します。

使用するサンプル・ファイルの名前を *18.htm* とすると、テーブル名は 18_0 になり（その HTML ファイルの最初のテーブルであるため）、列名とそれに対応する完全修飾名は次のようになります。

列	完全修飾名
Transaction_Quality	¥¥18_0¥Transaction_Quality
Measurements_Count	¥¥18_0¥Measurements_Count
Measurements_Percent	¥¥18_0¥Measurements_Percent

テーブルの列見出しが複数行ある場合、列名はそれらのヘッダー行の対応するフィールド・コンテンツを連結して形成します。

見出しタグのペアがないテーブルの場合、Oracle BI Server XML Gateway では最初の行のフィールド値（*<td>* と *</td>* のタグのペアで区切られた値）が列名とみなされます。列は出現順に名前が付けられます（c0、c1 など）。

XML の追加のサンプルは、「XML の例」（360 ページ）を参照してください。

Data Mining Adapter の使用

Data Mining Adapter は、Oracle BI Server XML Gateway の拡張機能です。これを使用して、取得した各レコードに対し実行可能ファイルや DLL をコールして外部データソースに選択的にアクセスできます。

Data Mining Adapter は、あるテーブルを、駆動テーブルとして機能する別のテーブルと論理結合するためにのみ使用できます。Data Mining Adapter を使用したテーブルは、論理結合を介して駆動テーブルから、パラメータ化されたクエリーを受け取ります。Data Mining Adapter を使用したテーブルは、バックエンド・データベースに物理的に存在するテーブルではありません。アダプタでは、入力列のパラメータとしてパラメータ化されたクエリーの WHERE 句の列値を使用して、WHERE 句にない列（出力列）の値が生成されます。論理結合の設定方法の詳細は、「駆動テーブルの指定」（144 ページ）を参照してください。

Data Mining Adapter は、次のように動作します。

- **DLL ファイルをコールする場合の動作** :Data Mining Adapter では、Data Mining Adapter API を実装する DLL、共有オブジェクトまたは共有ライブラリを指定できます。実行時に、アダプタでは DLL がロードされ、一度に 1 行のレコードを取得する API がコールされます。クエリーの結果は、API のパラメータを介して XML ゲートウェイに返されます。
- **実行可能ファイルをコールする場合の動作** :Data Mining Adapter では、実行可能ファイルを指定できます。実行時に、アダプタではこのファイルが実行され、一度に 1 行のレコードを取得します。また、出力ファイルの列値を区切るデリミタも指定します。

テーブルごとに実行可能ファイルまたは DLL を 1 つ指定します。

DLL ファイルを使用した Data Mining Adapter API のコール

この API は現在、1 つの機能のみで構成されています。これは、パラメータ化されたクエリーの入力列の値と、入力列および出力列の両方のメタ情報を使用します。戻り値として、API では出力列の値が outputColumnValueBuffer に格納されます。すべてのバッファはコール元が割り当てます。

この API で使用されるデータ型と構造体の定義の詳細は、IterativeGatewayDll.h を参照してください。このファイルは、次のパスにあります。

[*installation root*]¥Sample¥TestExternalGatewayDll¥IterativeGatewayDll.h

357 ページの表 40 では API 要素について説明します。

表 40. API 要素

要素	説明
inputColumnCount	入力列の数です。
inputColumnValueBuffer	入力列の値が格納されるバッファのバイト数です。各列値の実際のサイズは、OracleBIColumnMetaInfo の columnWidth フィールドで指定します。列値は、pInputColumnMetaInfoArray における列の出現順にバッファに格納されます。
modelId	「Physical Table」ダイアログ・ボックスの「XML」タブにある「Search Utility」フィールドで指定できるオプション引数です。
OutputColumnCount	出力列の数です。
outputColumnValueBuffer	出力列の値が格納されるバッファのバイト数です。各列値の実際のサイズは、OracleBIColumnMetaInfo の columnWidth フィールドで指定します。列値は、pOutputColumnMetaInfoArray における列の出現順にバッファに格納されます。
pInputColumnMetaInfoArray	入力列のメタ情報の配列です。OracleBIColumnMetaInfo は、パブリックのヘッダー・ファイル IterativeGatewayDll.h（Oracle BI とともにインストールされる）で宣言します。
pOutputColumnMetaInfoArray	出力列のメタ列情報の配列です。OracleBIColumnMetaInfo は、パブリックのヘッダー・ファイル IterativeGatewayDll.h（Oracle BI とともにインストールされる）で宣言します。API のコール元は列名を渡し、コール先では列のデータ型（現在は VarCharData のみをサポート）と列値のサイズが設定されます。

サンプル実装

Data Mining Adapter API のサンプル実装は、サポートされるすべてのプラットフォームの Oracle BI インストール・フォルダ（¥OracleBI）の Sample サブディレクトリにあります。次のファイルがサンプルに含まれます。

- hpacc.mak（HPUX 版のサンプル作成用メイク・ファイル）
- IterativeGatewayDll.h（DLL のインクルード用ヘッダー・ファイル）
- ReadMe.txt（Data Mining Adapter API について説明しているテキスト・ファイル）
- StdAfx.cpp（Windows 版のファイル）
- StdAfx.h（Windows 版のヘッダー・ファイル）
- sunpro.mak（Solaris 版のサンプル作成用メイク・ファイル）
- TestExternalGatewayDll.cpp（DLL のサンプル実装）
- TestExternalGatewayDll.dsp（サンプル作成用の Microsoft Visual C++ プロジェクト・ファイル）
- TestLibraryUnix.cpp（UNIX プラットフォームで DLL をロードするテスト・ドライブ）
- xlC50.mak（AIX 版のサンプル作成用メイク・ファイル）

ValueOf() 式の使用

ValueOf() 式をコマンドラインの引数で使用して、任意の追加パラメータを実行可能ファイルまたは DLL API に渡すことができます。

次の例で、ユーザー ID とパスワードを実行可能ファイルに渡す方法を示します。

```
executable_name valueof(USERID) valueof(PASSWORD)
```

列値の指定（実行可能ファイル）

実行可能ファイルを指定するときに、列名を $() で囲んで列値を実行可能ファイルに渡すことができます。

たとえば、Car_Loan、Credit、Demand、Score および Probability という列を含むテーブルがあるとします。入力列 Car_Loan、Credit および Demand の値は、結合関係を介して別のテーブルから取得されます。出力列 Score および Probability の値は、実行可能ファイルから返されます。コマンドラインは、次のようになります。

```
executable_name $(Car_Loan) $(Credit) $(Demand)
```

実行可能ファイルがコールされるたびに、列値の 1 行が返されます。列値は、指定したデリミタで区切られ、1 行で出力されます。

デフォルトでは、実行可能ファイルは stdout に出力されます。また、Data Mining Adapter で出力を一時出力ファイルから読み込むように指定することもできます。一時出力ファイルでは、プレースホルダ $(NQ_OUT_TEMP_FILE) で指定して引数として実行可能ファイルに渡すと、実行可能ファイルではそのファイルに結果行が出力されます。Data Mining Adapter で実行可能ファイルがコールされると、プレースホルダ $(NQ_OUT_TEMP_FILE) は実行時に生成される一時ファイル名に置換されます。次の例でこれについて説明します。

```
executable_name $(Car_Loan) $(Credit) $(Demand)  $(NQ_OUT_TEMP_FILE)
```

最初に実行可能ファイルへの入力でない列の値が、物理テーブルのそのままの列順に出力されます。前述の例では、Score 列の値の後が Probability 列の値になります。

実行可能ファイルで入力でない列の数より多くの列値が出力される場合、Data Mining Adapter では物理テーブルの列順のまま列値が読み込まれます。対応する入力列の値と競合がある場合、実行可能ファイルから返される値を使用して、入力列を上書きします。

区切られたクエリー出力における各列のデータ長は、物理テーブルの対応する列で指定されたサイズを超えることはできません。

Data Mining Adapter の構成

次の手順に従って、Data Mining Adapter を構成します。

Data Mining Adapter を構成するには

1. Administration Tool でデータベースを作成し、データベース・タイプとして「XML Server」を選択して、「OK」をクリックします。

   データベースの作成方法の詳細は、「物理レイヤーでのデータベース・オブジェクトの手動作成」（65 ページ）を参照してください。

2 接続プールを構成します。

注意：「Connection Pool」ダイアログ・ボックスの「XML」タブでは、すべてのフィールドで情報を入力しないでください。空のフィールドは、Oracle BI Server により Data Mining Adapter の機能が呼び出されることを示します。

a 手順 1 で作成したデータベースを右クリックして、「New Object」→「Connection Pool」を選択します。

b 「General」タブで、接続プールの名前を入力します。

コール・インタフェースのデフォルトは XML です。

c データソース名を入力してから「OK」をクリックします。

3 手順 1 で作成したデータベースを右クリックして、「New Object」→「Table」を選択します。

4 「Physical Table」ダイアログ・ボックスで、「XML」タブをクリックします。

5 「XML」タブで、次のいずれかの作業を実行します。

- 「Executable」を選択して、「Search Utility」フィールドに実行可能ファイルへのパスを入力し、出力値のデリミタを指定します。
- 「DLL」を選択して、「Search Utility」フィールドに DLL へのパスを入力します。

  パスの中に空白がある場合、パスを引用符で囲みます。次に例を示します。

  ```
  "C:¥Program Files¥OracleBI¥Bin¥SADataMining.dll"
  ```

  DLL パスの後にあるすべての文字は、modelid 文字列として API に渡されます。modelid 文字列を使用して、API を介して静的パラメータまたは動的パラメータを DLL に渡すことができます。次に例を示します。

  ```
  "C:¥Program Files¥OracleBI¥Bin¥SADataMining.dll" VALUEOF(Model1) VALUEOF(Model2)
  ```

XML ODBC の使用

XML ODBC データベース・タイプを使用すると、ODBC インタフェースを介して XML データソースにアクセスできます。物理テーブルの物理列を表す XML 要素のデータ型は、XML スキーマで定義された XML 要素のデータ型から導出されます。適切な XML スキーマがないと、デフォルトのデータ型である文字列型が使用されます。「Physical」レイヤーのデータ型を設定しても、XML データソースで定義された設定より優先されません。XML スキーマのない XML データにアクセスする場合、CAST 演算子を使用して Administration Tool の「Business Model and Mapping」レイヤーにおけるデータ型変換を実行します。

ODBC を使用して XML データをインポートするには

1 ODBC を介して XML データソースにアクセスするには、XML ODBC ドライバのライセンスを使用してインストールする必要があります。

2 次に、アクセスする XML データソースを指す ODBC DSN を作成し、XML ODBC データベース・タイプが選択されていることを確認します。

3 「File」メニューで、「Import」→「from Database」を選択します。

4 ダイアログ・ボックスの指示に従って、ODBC DSN をリポジトリにインポートします。

警告：「Import」ダイアログ・ボックスで「Synonyms」オプションが選択されていることを確認します。

XML ODBC の例

次に、Microsoft ADO 永続ファイル形式における XML ODBC データソースの例を示します。データとスキーマは両方とも同じ文書の中に含めることができます。

```
<xml xmlns:s='uuid:BDC6E3F0-6DA3-11d1-A2A3-00AA00C14882'
   xmlns:dt='uuid:C2F41010-65B3-11d1-A29F-00AA00C14882'
   xmlns:rs='urn:schemas-microsoft-com:rowset'
   xmlns:z='#RowsetSchema'>
<s:Schema id='RowsetSchema'>
   <s:ElementType name='row' content='eltOnly' rs:CommandTimeout='30'
rs:updatable='true'>
      <s:AttributeType name='ShipperID' rs:number='1' rs:writeunknown='true'
rs:basecatalog='Northwind' rs:basetable='Shippers'
          rs:basecolumn='ShipperID'>
         <s:datatype dt:type='i2' dt:maxLength='2' rs:precision='5'
rs:fixedlength='true' rs:maybenull='false'/>
      </s:AttributeType>
      <s:AttributeType name='CompanyName' rs:number='2' rs:writeunknown='true'
rs:basecatalog='Northwind' rs:basetable='Shippers'
          rs:basecolumn='CompanyName'>
         <s:datatype dt:type='string' rs:dbtype='str' dt:maxLength='40'
rs:maybenull='false'/>
      </s:AttributeType>
      <s:AttributeType name='Phone' rs:number='3' rs:nullable='true'
rs:writeunknown='true' rs:basecatalog='Northwind'
          rs:basetable='Shippers' rs:basecolumn='Phone'>
         <s:datatype dt:type='string' rs:dbtype='str' dt:maxLength='24'
rs:fixedlength='true'/>
      </s:AttributeType>
      <s:extends type='rs:rowbase'/>
   </s:ElementType>
</s:Schema>

<rs:data>
   <z:row ShipperID='1' CompanyName='Speedy Express' Phone='(503) 555-9831          '/>
   <z:row ShipperID='2' CompanyName='United Package' Phone='(503) 555-3199          '/>
   <z:row ShipperID='3' CompanyName='Federal Shipping' Phone='(503) 555-9931        '/>
</rs:data>
</xml>
```

XML の例

次の XML 文書では様々な状況の例を示すとともに、Oracle BI Server XML のアクセス方式によりこれらの状況に対応する方法について説明します。

- XML 文書の *83.xml* および *8_sch.xml* は、異なる有効範囲で同じ要素の宣言を使用する例を示します。たとえば、<p3> は <p2> の中にも <p4> の中にも出現できます。

  前述の例における要素 <p3> は異なる有効範囲に出現するため、インポート処理時に 2 番目に出現した要素にインデックス番号を付加して、各要素に個別の列名を付与します。この場合、2 番目に出現した要素は p3_1 となります。ここでさらに <p3> が出現すると、p3_2、p3_3 のようになります。

- XML 文書の *83.xml* および *84.xml* では、複数の XML ファイルで同じスキーマを共有できる例を示します（*8_sch.xml*）。

- Internet Explorer バージョン 5 以上では、XML アイランドと呼ばれる XML フラグメントが埋め込まれた HTML 文書がサポートされています。

  XML 文書 island2.htm は、複数の XML データ・アイランドと複数のテーブルが 1 つの文書から生成できる簡単な例を示します。XML アイランドのインスタンスごとに 1 つのテーブルが生成されます。適切なインデックスを文書名に付加して、テーブルを識別します。*island2.htm* では、生成される 2 つの XML テーブルは、island2_0 と island2_1 になります。

83.xml

```
===83.xml===

<?xml version="1.0"?>
<test xmlns="x-schema:8_sch.xml">|
<row>
<p1>0</p1>
<p2 width="5" height="2">
   <p3>hi</p3>
   <p4>
      <p3>hi</p3>
      <p6>xx0</p6>
      <p7>yy0</p7>
   </p4>
   <p5>zz0</p5>
</p2>
</row>


<row>
<p1>1</p1>
<p2 width="6" height="3">
   <p3>how are you</p3>
   <p4>
      <p3>hi</p3>
      <p6>xx1</p6>
      <p7>yy1</p7>
   </p4>
   <p5>zz1</p5>
</p2>
</row>
</test>
```

8_sch.xml

===8_sch.xml===

```
<Schema xmlns="urn:schemas-microsoft-com:xml-data" xmlns:dt="urn:schemas-microsoft-
com:datatypes">
         <AttributeType name="height" dt:type="int" />
   <ElementType name="test" content="eltOnly" order="many">
      <AttributeType name="height" dt:type="int" />
      <element type="row"/>
   </ElementType>
   <ElementType name="row" content="eltOnly" order="many">
         <element type="p1"/>
      <element type="p2"/>
   </ElementType>
   <ElementType name="p2" content="eltOnly" order="many">
         <AttributeType name="width" dt:type="int" />
      <AttributeType name="height" dt:type="int" />
         <attribute type="width" />
      <attribute type="height" />
      <element type="p3"/>
      <element type="p4"/>
      <element type="p5"/>
   </ElementType>
   <ElementType name="p4" content="eltOnly" order="many">
      <element type="p3"/>
      <element type="p6"/>
      <element type="p7"/>
   </ElementType>
   <ElementType name="test0" content="eltOnly" order="many">
      <element type="row"/>
   </ElementType>
      <ElementType name="p1" content="textOnly" dt:type="string"/>
      <ElementType name="p3" content="textOnly" dt:type="string"/>
      <ElementType name="p5" content="textOnly" dt:type="string"/>
      <ElementType name="p6" content="textOnly" dt:type="string"/>
      <ElementType name="p7" content="textOnly" dt:type="string"/>
</Schema>
```

84.xml

```
===84.xml===

<?xml version="1.0"?>
<test0 xmlns="x-schema:8_sch.xml">
<row>
<p1>0</p1>
<p2 width="5" height="2">
   <p3>hi</p3>
   <p4>
      <p3>hi</p3>
      <p6>xx0</p6>
      <p7>yy0</p7>
   </p4>
   <p5>zz0</p5>
</p2>
</row>


<row>
<p1>1</p1>
<p2 width="6" height="3">
   <p3>how are you</p3>
   <p4>
      <p3>hi</p3>
      <p6>xx1</p6>
      <p7>yy1</p7>
   </p4>
   <p5>zz1</p5>
</p2>
</row>
</test0>
```

Island2.htm

```
===island2.htm===

<HTML>
   <HEAD>
<TITLE>HTML Document with Data Island</TITLE>
</HEAD>
   <BODY>
<p>This is an example of an XML data island in I.E. 5</p>
   <XML ID="12345">
   test>
      <row>
         <field1>00</field1>
         <field2>01</field2>
   </row>
      <row>
         <field1>10</field1>
         <field2>11</field2>
   </row>
      <row>
         <field1>20</field1>
         <field2>21</field2>
      </row>
   </test>
</XML>
<p>End of first example.</p>
<XML ID="12346">
   <test>
      <row>
         <field11>00</field11>
         <field12>01</field12>
      </row>
      <row>
         <field11>10</field11>
         <field12>11</field12>
      </row>
      <row>
         <field11>20</field11>
         <field12>21</field12>
      </row>
   </test>
</XML>
<p>End of second example.</p>
</BODY>
</HTML>
```

17 Oracle BI Server SQL リファレンス

Oracle BI Server はクライアント・ツールからの SQL SELECT 文を受け入れます。また Administration Tool を使用すれば、ユーザーは複雑な式を使用して論理テーブルを定義することもできます。この章では、SELECT 文の構文とセマンティクス、および論理テーブルを定義するために Administration Tool で使用できる式の構文とセマンティクスについて説明します。

SQL 構文およびセマンティクス

この項では、SELECT 文の構文とセマンティクスについて説明します。この項の内容は次のとおりです。

注意： このマニュアルにおける構文の説明は包括的なものではありません。Oracle BI Server 固有の基本的な構文と機能のみを説明しています。SQL 構文のより詳細な解説については、SQL に関するサード・パーティの参考書籍か、使用しているデータベース・ベンダーの発行する SQL リファレンス・マニュアルを参照してください。

SELECT クエリーの指定構文

SELECT 文は、任意の SQL（Structured Query Language）データベースにクエリーを行うための基礎となります。Oracle BI Server は論理リクエストを受け入れて、リポジトリ内のオブジェクトのクエリーを行います。これらの論理リクエストは、ユーザー（またはクエリー・ツール）が通常の SQL SELECT 文を使用して作成します。サーバーは次に、これらの論理リクエストを 1 つ以上のデータソースに対する物理クエリーに変換し、論理リクエストに一致するように結果を組み合せて、エンド・ユーザーに回答を返します。

SELECT 文（クエリー指定とも呼ばれる）は、Oracle BI Server を介して意思決定支援システムにクエリーを行う方法です。SELECT 文は、そのクエリーに一致するテーブルをクライアントに返します。結果が行と列の形式で格納されているため、それはテーブルであると言えます。

次に、SELECT 文の基本構文を示します。それぞれの句については、後続の各項で定義しています。

```
SELECT [DISTINCT] select_list

FROM from_clause

[WHERE search_condition]

[GROUP BY column {, column}

     [HAVING search_condition]]

[ORDER BY column {, column}]
```

内容は次のとおりです。

select_list	クエリーで指定される列のリストです。「SELECT リスト構文」（367 ページ）を参照してください。
from_clause	クエリーで指定されるテーブルのリスト、またはカタログ・フォルダ名です。クエリーの特定の結合情報をオプションで含みます。「FROM 句の構文」（368 ページ）を参照してください。
search_condition	条件テストを生成するための条件の任意の組合せを指定します。「WHERE 句の構文」（368 ページ）を参照してください。
column	データソースで定義されているテーブルに属する列（または別名）です。

SELECT の使用上の注意

Oracle BI Server では、SELECT 文が論理リクエストとして処理されます。SELECT 文で集計データがリクエストされる場合は、サーバーにより自動的に GROUP BY 句が仮定されます。クエリー内で指定される結合条件はすべて無視されます。結合条件はすべてリポジトリ内で事前定義されています。

Oracle BI Server は、コメントについて次の SELECT 構文を受け入れます。

- C スタイル・コメント（/* */）
- 1 行コメントに対する二重スラッシュ（//）
- 1 行コメントに対する番号記号（#）

Oracle BI Server は、論理リクエスト内の特定のサブクエリーと、UNION、UNION ALL、INTERSECT および EXCEPT 操作をサポートします。この機能により、回答可能なビジネス上の調査範囲が拡大し、クエリーの生成が容易になり、複数のビジネス・モデル間でクエリーを実行できるようになります。

Oracle BI Server は、任意の条件式（WHERE、HAVING または CASE 文など）において次のサブクエリー述語をサポートします。

- IN、NOT IN
- >Any、>=Any、=Any、<Any、<=Any、<>Any
- >Some、>=Some、=Some、<Some、<=Some、<>Some
- >All、>=All、=All、<All、<=All、<>All
- EXISTS、NOT EXISTS

SELECT リスト構文

SELECT リスト構文は、クエリーの列をリストします。

構文：

```
SELECT [DISTINCT] select_list
```

内容は次のとおりです。

select_list　クエリーで指定される列のリストです。すべての列は単一のビジネス・モデルから派生している必要があります。

テーブル名を含めることができます（*Table.Column* など）。これは列名がビジネス・モデル内で一意でない場合は必須ですが、一意の場合はオプションです。

列名に空白が含まれる場合は、列名を二重引用符で囲みます。Oracle BI Server では常に個別のクエリーが実行されるため、DISTINCT キーワードを含める必要はありません。

集計ルールはサーバーで認識され、集計は自動的に実行されるため、集計が行われる列に集計関数（SUM など）を含める必要はありません。

FROM 句の構文

Oracle BI Server は、任意の有効な SQL FROM 句の構文を受け入れます。FROM 句の作成を容易にするため、テーブル・リストのかわりにカタログ・フォルダの名前を指定できます。クエリーで検索する列と Oracle BI リポジトリの構成に基づいて、適切なテーブルと適切な結合指定が決定されます。

WHERE 句の構文

Oracle BI Server は、任意の有効な SQL WHERE 句の構文を受け入れます。結合はすべて Oracle BI リポジトリ内で構成されるため、WHERE 句に結合条件を指定する必要はありません。WHERE 句で指定された結合条件はすべて無視されます。

Oracle BI Server は、任意の条件式（WHERE、HAVING または CASE 文など）において次のサブクエリー述語をサポートします。

- IN、NOT IN
- >Any、>=Any、=Any、<Any、<=Any、<>Any
- >All、>=All、=All、<All、<=All、<>All
- EXISTS、NOT EXISTS

GROUP BY 句の構文

Oracle BI Server の自動集計機能により、GROUP BY 句を発行する必要はありません。GROUP BY 句が指定されていない場合は、GROUP BY 指定によって、SELECT リスト内のすべての非集計列がデフォルトで使用されます。SELECT リスト内で集計関数を明示的に使用する場合は、異なる列に GROUP BY 句を指定できます。これにより、GROUP BY 句で指定されたレベルに基づいて結果が計算されます。Oracle BI Server に対するクエリーで GROUP BY 句を使用する方法の詳細と例については、「集計関数を使用するクエリーのルール」（368 ページ）を参照してください。

ORDER BY 句の構文

Oracle BI Server は、SELECT リスト内の順序による列の参照（ORDER BY 3, 1, 5 など）を含め、任意の有効な SQL ORDER BY 句の構文を受け入れます。

集計関数を使用するクエリーのルール

Oracle BI Server では、集計クエリーの作成に必要な SQL が簡略化されます。この項では、クエリーに GROUP BY 句が含まれるかどうかによって異なる Oracle BI Server の動作ルールと、GROUP BY 句が指定されている場合に予期されるクエリーからの結果について概要を示します。この項で説明するルールは、SQL 文で使用されるすべての集計（SUM、AVG、MIN、MAX、COUNT(*) および COUNT）に適用されます。

ベースライン列の集計の計算

ベースライン列とは、リポジトリの「Logical Column」ダイアログの「Aggregation」タブで集計ルールが定義されていない列を指します。ベースライン列は、それらが属する論理テーブルの粒度のレベルで、非集計データにマップされます。SQL リクエストを介してベースライン列に対する集計（SUM、AVG、MIN、MAX または COUNT）を実行すると、Oracle BI Server では次のルールに基づいたレベルで集計が計算されます。

- GROUP BY 句が指定されていない場合、集計のレベルは、SELECT リスト内のすべての非集計列でグループ化されます。
- GROUP BY 句が指定されている場合、集計のレベルは、GROUP BY 句で指定された列に基づいて決定されます。

たとえば、次のクエリーを考えます。ここでは、*revenue* 列がベースライン列としてリポジトリで定義されています（「Logical Column」→「Aggregation」タブで集計ルールが指定されていない）。

```
select year, product, sum(revenue)

from time, products, facts

YEAR     PRODUCT      SUM(REVENUE)

1998     Coke          500

1998     Pepsi         600

1999     Coke          600

1999     Pepsi         550

2000     Coke          800

2000     Pepsi         600
```

このクエリーは、*year* および *product* によってグループ化された結果を返します。つまり、製品と年の組合せごとに 1 行を返します。各行で計算される合計は、その年におけるその製品のすべての売上の合計です。このクエリーは、論理的には次のクエリーと同じです。

```
select year, product, sum(revenue)

from time, products, facts

group by year, product
```

year によってのみグループ化するように GROUP BY 句を変更すると、次に示すように、計算される合計はその年におけるすべての製品の合計になります。

```
select year, product, sum(revenue)

from time, products, facts

group by year


YEAR     PRODUCT      SUM(REVENUE)
1998     Coke         1100
1998     Pepsi        1100
1999     Coke         1150
1999     Pepsi        1150
2000     Coke         1400
2000     Pepsi        1400
```

このクエリーの結果セットでは、その年のすべての売上を表す *revenue* の合計が、特定の年に対応する各行で同じになります。この例では、Coke の売上に Pepsi の売上を加算した合計が表示されます。

revenue の COUNT をリクエストする列をクエリーに追加すると、各グループの結果を計算するために使用されたレコード数が計算されます。次に示すように、この例での対象レコードは year になります。

```
select year, product, sum(revenue), count(revenue)
from time, products, facts
group by year

YEAR    PRODUCT      SUM(REVENUE)  COUNT(REVENUE)
1998    Coke         1100          6000
1998    Pepsi        1100          6000
1999    Coke         1150          6500
1999    Pepsi        1150          6500
2000    Coke         1400          8000
2000    Pepsi        1400          8000
```

メジャー列の集計の計算

メジャー列とは、リポジトリの「Logical Column」ダイアログの「Aggregation」タブでデフォルトの集計ルールが定義されている列を指します。メジャー列では、それらに定義されている集計が常に計算されます。SQL リクエストを介してメジャー列に対する明示的な集計（SUM、AVG、MIN、MAX または COUNT）を実行すると、実際には集計の集計を要求することになります。このようなネストされた集計の場合、Oracle BI Server では次のルールに基づいて集計が計算されます。

- SQL 文で集計関数を定義せずにメジャー列をリクエストすると、クエリーに GROUP BY 句が指定されているかどうかにかかわらず、常に SELECT リスト内の非集計列のレベルでグループ化されます。
- GROUP BY 句が指定されていない場合、ネストされた集計は、SELECT リスト内のすべての非集計列によって決定される各グループの総計になります。
- GROUP BY 句が指定されている場合、ネストされた集計は、GROUP BY 句で指定されている各グループの合計になります。

たとえば、次のクエリーを考えます。ここでは、SumOfRevenue 列が SUM のデフォルトの集計ルール（「Logical Column」ダイアログの「Aggregation」タブで指定されている SUM 集計ルール）を使用するメジャー列としてリポジトリで定義されています。

```
select year, product, SumOfRevenue, sum(SumOfRevenue)
from time, products, facts

YEAR    PRODUCT      SUMofREVENUE  SUM(SUMofREVENUE)
1998    Coke         500           3650
1998    Pepsi        600           3650
1999    Coke         600           3650
1999    Pepsi        550           3650
2000    Coke         800           3650
2000    Pepsi        600           3650
```

このクエリーは、*year* および *product* によってグループ化された結果を返します。つまり、製品と年の組合せごとに 1 行を返します。メジャー列のレベルは常にクエリーの非集計列で決定されるため、SumOfRevenue 列の各行で計算される合計は、その年におけるその製品のすべての売上の合計になります。このクエリーは、論理的には次のクエリーと同じです。

```
select year, product, SumOfRevenue, sum(SumOfRevenue)
from time, products, facts
group by year, product
```

year によってのみグループ化するように GROUP BY 句を変更すると、次に示すように、SumOfRevenue 列で計算される合計はその年の製品ごとの合計となり、SUM（SumOfRevenue）列で計算される合計は、その年のすべての製品売上の合計になります。

```
select year, product, SumOfRevenue, sum(SumOfRevenue)
from time, products, facts
group by year

YEAR    PRODUCT      SUMofREVENUE   SUM(SUMofREVENUE)
1998    Coke         500            1100
1998    Pepsi        600            1100
1999    Coke         600            1150
1999    Pepsi        550            1150
2000    Coke         800            1400
2000    Pepsi        600            1400
```

この結果セットでは、メジャー列のレベルは常にクエリーの非集計列で決定されるため、SumOfRevenue 列の各行で計算される合計は、その年におけるその製品のすべての売上の合計になります。その年の売上の合計を表す SUM（SumOfRevenue）は、特定の年に対応する各行で同じになります。この例では、Coke の売上に Pepsi の売上を加算した合計が表示されます。

表示関数のリセット動作

表示関数とは、クエリーの結果セットに対して動作する関数です。Oracle BI Server でサポートされる表示関数（RANK、TOP*n*、BOTTOM*n*、PERCENTILE、NTILE、MAVG、MEDIAN および様々な標準偏差）を、SQL クエリーの SELECT リストに指定します。表示関数を使用するクエリーは、次のルールに従います。

- GROUP BY 句が指定されていない場合、表示関数は結果セット全体に対して動作します。つまり、表示関数のグループ化のレベルはクエリーのレベルと同じになります。
- GROUP BY 句が指定されている場合、表示関数は、GROUP BY 句で指定されているように自身の値を各グループに対してリセットします。

たとえば次のクエリーでは、SumOfRevenue は、SUM のデフォルトの集計ルールを使用するメジャー列として定義されています。

```
select year, product, SumOfRevenue, rank(SumOfRevenue)
from time, products, facts

YEAR    PRODUCT      SUMOFREVENUE  RANK(SUMOFREVENUE)
1998    Coke         500           6
1998    Pepsi        600           2
1999    Coke         600           2
```

```
1999    Pepsi       550         5
2000    Coke        800         1
2000    Pepsi       600         2
```

このクエリーの結果セットでは、GROUP BY 句が指定されていないため、ランクは結果セット全体に対して計算されます。このクエリーは、論理的には次のクエリーと同じです。

```
select year, product, SumOfRevenue, rank(SumOfRevenue)
from time, products, facts
group by year, product
```

year によってのみグループ化するように GROUP BY 句を変更すると、次に示すように、ランクは年ごとにリセットされます。

```
select year, product, sum(revenue), rank(sum(revenue))
from time, products, facts
group by year
```

```
YEAR    PRODUCT     SUM(REVENUE) RANK(SUM(REVENUE))
1998    Coke        500         2
1998    Pepsi       600         1
1999    Coke        600         1
1999    Pepsi       550         2
2000    Coke        800         1
2000    Pepsi       600         2
```

この結果セットでは、年が変わるごとにランクがリセットされます。年ごとに 2 行ずつあるため、ランクの値は常に 1 または 2 になります。

代替構文

集計関数を使用すると、集計関数内で BY を使用することにより特定レベルの集計を計算できます。これを行う場合、GROUP BY 句を指定する必要はありません。

例として、次のクエリーを考えます。

```
select year, product, revenue, sum(revenue by year) as year_revenue from softdrinks
```

このクエリーは、year ごとに集計された売上を表示する year_revenue 列を返します。

同じ構文を表示関数で使用できます。次のクエリーを考えます。

```
select year, product, revenue, rank(revenue), rank(revenue by year) from softdrinks
order by 1, 5
```

このクエリーは、各年の各製品の売上（結果セット全体の各行）の全体的なランクと、年ごとの各製品の売上のランクを計算します。

FILTER を使用した条件集計の計算

SQL クエリー言語では、SUM、COUNT、MIN および MAX などの従来の集計は、GROUP BY 句によって決定されるタプル（指定された種類のオブジェクトの順序付けされたリスト）のグループに対して評価されます。クエリーの SELECT 句で指定されるすべての集計は、タプルの同じサブセットに対して評価されます。条件集計は、述語を使用して入力を制限することで SQL を拡張します。

FILTER 演算子は、集計の引数を計算するために使用される行セットを USING 条件に基づいて絞り込みます。FILTER 演算子は論理 SQL 構文です。メタデータを参照する論理クエリーや、既存の論理列をソースとして使用する論理列内で使用される場合があります。

構文

条件集計は表記上の概念であり、実行可能な演算子を表している訳ではありません。次の文に示すように、条件集計は関数の形式で表されます。

```
FILTER(<measure_expression> USING <boolean_expression>)
```

内容は次のとおりです。

- <measure_expression> は、少なくとも 1 つのメジャーを含む式です。次に例を示します。
  - 式 Sales + 1 は、Sales がメジャーである場合に許可されます。
  - 式 productid は、productid がスカラー属性である場合は許可されません。
- <boolean_expression> は、メジャーを含まないブール式です（TRUE または FALSE に評価されます）。この式に、ネストされたクエリーを含めることはできません。

FILTER 関数の例

次に FILTER 関数の簡単な例を示します。

```
SELECT year,
FILTER(sales USING product = 'coke'),
FILTER(sales USING product = 'pepsi')
FROM logBeverages
```

ナビゲーションの実行後、クエリーは次のように実行されます。

```
SELECT year,
SUM(CASE WHEN product = 'coke' THEN sales),
SUM(CASE WHEN product = 'pepsi' THEN sales)
FROM physBeverages
WHERE product = 'coke' OR product = 'pepsi'
GROUP BY year
```

エラー処理

例の FILTER（X USING Y）では、次の場合にエラー・メッセージが返されます。

- Y 式がブール式ではない。
- Y 式にメジャーが含まれる。
- FILTER が外部のクエリー・ブロックで使用されている。
- 明示的な集計が X（メジャー）式で使用されている。たとえば、FILTER(COUNT(product), C) など。

SQL 論理演算子

次の SQL 論理演算子は、式間の比較を指定するために使用されます。

Between: 条件の境界を定義します。それぞれの境界は式で、それらが「より小さい」や「より大きい」で表される場合は、(「以下」や「以上」とは対照的に）境界の範囲には上限と下限の値は含まれません。この条件を否定する場合は、BETWEEN の前に NOT を付けることができます。

In: 列の値を、一連の値と比較します。

Is Null: 列の値を、NULL 値と比較します。

Like: リテラル値に対する比較を指定します。多くの場合、ワイルドカード文字とともに使用して、ゼロ文字以上の任意の文字列（%）に一致するか、または任意の単一文字（_）に一致することを指定します。

条件式

Expressions フォルダには、CASE、WHEN、THEN および ELSE 文を使用する条件式を作成するための構築ブロックが含まれています。

CASE（Switch）

この形式の CASE 文は、CASE（Lookup）形式とも呼ばれます。

構文：

```
CASE expression1
     WHEN expression2 THEN expression3
     {WHEN expression...THEN expression...}
     ELSE expression...
END
```

expression1 の値が検証された後、WHEN 式が検証されます。expression1 が任意の WHEN 式に一致すると、その対応する THEN 式の値が割り当てられます。

一致する WHEN 式がない場合は、ELSE 式に指定されているデフォルト値が割り当てられます。ELSE 式が指定されていない場合は、システムによって自動的に ELSE NULL が追加されます。

expression1 が複数の WHEN 句の式に一致する場合は、最初に一致した式のみが割り当てられます。

注意： CASE 文では、AND は OR よりも優先されます。

CASE

CASE 文を開始します。この CASE 文には、1 つの式、1 つ以上の WHEN 文と THEN 文、ELSE 文（オプション）、および END キーワードを指定する必要があります。

WHEN

満たす条件を指定します。

THEN

対応する WHEN 式が満たされる場合に割り当てる値を指定します。

ELSE

満たされる WHEN 条件がない場合に割り当てる値を指定します。省略すると、ELSE NULL が指定されたものとみなされます。

END

CASE 文を終了します。

CASE （If）

この形式の CASE 文の構文は次のとおりです。

```
CASE
      WHEN search_condition1 THEN expression1
      {WHEN search_condition2 THEN expression2}
      {WHEN search_condition...THEN expression...}
      ELSE expression
END
```

各 WHEN 条件を評価して、それが満たされる場合、対応する THEN 式の値を割り当てます。

一致する WHEN 条件がない場合は、ELSE 式に指定されているデフォルト値が割り当てられます。

ELSE 式が指定されていない場合は、システムによって自動的に ELSE NULL が追加されます。

注意： CASE 文では、AND は OR よりも優先されます。

Switch 形式の CASE 文とは異なり、If 形式の WHEN 文では比較演算子が許可されます。たとえば、WHEN < 0 THEN 'Under Par' という WHEN 条件を使用できます。

CASE
CASE 文を開始します。この CASE 文には、1 つ以上の WHEN 文と THEN 文、ELSE 文（オプション）、および END キーワードを指定する必要があります。

WHEN
満たす条件を指定します。

THEN
対応する WHEN 式が満たされる場合に割り当てる値を指定します。

ELSE
満たされる WHEN 条件がない場合に割り当てる値を指定します。省略すると、ELSE NULL が指定されたものとみなされます。

END
CASE 文を終了します。

SQL リファレンス

SQL 関数は、列の値について様々な計算を実行します。この項では、Oracle BI Server でサポートされる関数の構文について説明します。また、リテラルの表現方法についても説明します。これらの関数には、集計関数、文字列関数、算術関数、カレンダ日付 / 時刻関数、変換関数およびシステム関数があります。

この項の内容は次のとおりです。

集計関数

集計関数は複数の値に対して計算を実行し、サマリー結果を作成します。「Logical Column」ダイアログ・ボックスの「Aggregation」タブでデフォルトの集計ルールが定義されている論理列に対して、ネストされた集計の式を集計関数を使用して作成することはできません。ネストされた集計を指定するには、デフォルトの集計ルールを使用する列を定義してから、SQL 文で列の集計をリクエストする必要があります。

Avg

式の結果セット内の平均値を計算します。引数には数式を指定する必要があります。

構文：

```
AVG (n_expression)
```

内容は次のとおりです。

n_expression　　数値に評価される任意の式を指定します。

AvgDistinct

式のすべての一意な値について平均値を計算します。引数には数式を指定する必要があります。

構文：

```
AVG (DISTINCT n_expression)
```

内容は次のとおりです。

n_expression　　数値に評価される任意の式を指定します。

BottomN

引数に指定した式の下位 n 件の値を 1 ～ n でランク付けします。1 が最下位の数値になります。BOTTOMN 関数は、結果セット内の返された値に対して計算を実行します。

構文：

```
BOTTOMN (n_expression, n)
```

内容は次のとおりです。

n_expression　　数値に評価される任意の式を指定します。

n　　任意の正の整数を指定します。結果セットで表示するランク付けの対象数を示します。1 が最も低いランクを表します。

注意： 1 つのクエリーに含めることができる BOTTOMN 式は 1 つのみです。

Count

式について NULL でない値を持つ行数を計算します。通常、式には列名を指定します。その場合は、その列の NULL でない値を持つ行数が返されます。

構文：

```
COUNT (expression)
```

内容は次のとおりです。

expression　　任意の式を指定します。

CountDistinct

COUNT 関数に、一意な値の行数を返す処理を追加します。

構文：

```
COUNT (DISTINCT expression)
```

内容は次のとおりです。

expression　　任意の式を指定します。

Count (*) (CountStar)

行数を計算します。

構文：

```
COUNT(*)
```

たとえば、Facts という名前のテーブルに 200,000,000 の行が含まれる場合、クエリーとその結果は次のようになります。

```
SELECT COUNT(*) FROM Facts

COUNT(*)

200000000
```

First

引数に指定した式の最初に返される値を選択します。FIRST 関数の使用は、リポジトリ内でディメンション固有の集計ルールを定義する場合に限定されます。SQL 文で使用することはできません。

FIRST 関数は、明示的に定義されたディメンションに指定されている、最も詳細なレベルで計算を実行します。たとえば、日、月、年の階層レベルで定義された時間ディメンションがある場合、FIRST 関数は各レベルで最初の日を返します。

FIRST 関数は、最初のディメンション固有の集計ルールとして使用しないでください。クエリーによって Oracle BI Server で処理するための膨大な数の行が返され、パフォーマンスが低下する可能性があります。

構文：

FIRST (expression)

内容は次のとおりです。

expression 任意の式を指定します。

GroupByColumn

集計ナビゲーションの設定に使用します。物理集計テーブルに存在する集計データのレベルを定義する論理列を指定します。

たとえば、集計テーブルに店舗と月によってグループ化されたデータが含まれる場合、コンテンツ・フィルタ（「Logical Source」ダイアログの「General」タブ）で次の構文を指定します。

```
GROUPBYCOLUMN(STORE, MONTH)
```

GROUPBYCOLUMN 関数は、リポジトリの構成のためにのみ使用します。SQL 文の作成に使用することはできません。

GroupByLevel

集計ナビゲーションの設定に使用します。物理集計テーブルに存在する集計データのレベルを定義するディメンション・レベルを指定します。

たとえば、集計テーブルに店舗レベルと月レベルのデータが含まれており、これらのレベルを含むディメンション（地理および顧客）を定義している場合は、コンテンツ・フィルタ（「Logical Source」ダイアログの「General」タブ）で次の構文を指定します。

```
GROUPBYLEVEL (GEOGRAPHY.STORE, CUSTOMERS.MONTH)
```

GROUPBYLEVEL 関数は、リポジトリの構成のためにのみ使用します。SQL 文の作成に使用することはできません。

Last

引数に指定した式の最後に返される値を選択します。LAST 関数の使用は、リポジトリ内でディメンション固有の集計ルールを定義する場合に限定されます。SQL 文で使用することはできません。

LAST 関数は、明示的に定義されたディメンションに指定されている、最も詳細なレベルで計算を実行します。たとえば、日、月、年の階層レベルで定義された時間ディメンションがある場合、LAST 関数は各レベルで最後の日を返します。

LAST 関数は、最初のディメンション固有の集計ルールとして使用しないでください。クエリーによって Oracle BI Server で処理するための膨大な数の行が返され、パフォーマンスが低下する可能性があります。

構文：

```
LAST (expression)
```

内容は次のとおりです。

expression 任意の式を指定します。

Max

引数に指定した数式を満たす行の最大値を計算します。

構文：

```
MAX (expression)
```

内容は次のとおりです。

expression　　任意の式を指定します。

MAX 関数は、「表示関数のリセット動作」（371 ページ）で説明しているルールに基づいて、クエリー内の各グループに対して自身の値をリセットします。

Median

引数に指定した数式を満たす行の中央値を計算します。行数が偶数の場合、中央値は 2 つの中央の行の平均になります。この関数は常に倍精度の値を返します。

構文：

```
MEDIAN (n_expression)
```

内容は次のとおりです。

n_expression　　数値に評価される任意の式を指定します。

MEDIAN 関数は、「表示関数のリセット動作」（371 ページ）で説明しているルールに基づいて、クエリー内の各グループに対して自身の値をリセットします。

Min

引数に指定した数式を満たす行の最小値を計算します。

構文：

```
MIN (expression)
```

内容は次のとおりです。

expression　　任意の式を指定します。

MIN 関数は、「表示関数のリセット動作」（371 ページ）で説明しているルールに基づいて、クエリー内の各グループに対して自身の値をリセットします。

NTile

NTILE 関数は、ユーザーの指定した範囲における値のランクを決定します。任意のランクを表す整数を返します。つまり、結果としてソートされるデータ・セットは複数のグループに分割されており、各グループにほぼ同じ数の値が含まれます。

構文：

```
NTILE (n_expression, n)
```

内容は次のとおりです。

n_expression　　数値に評価される任意の式を指定します。
n　　グループの数を表す、NULL でない正の整数を指定します。

n_expression 引数が NULL でない場合、この関数はリクエストされた範囲内のランクを表す整数を返します。

n=100 を指定した NTile では、一般的にパーセンタイルと呼ばれる値が返されます（1 ～ 100 の値で、100 はソート範囲の上限を表します）。この値は、SQL 92 のパーセント・ランクに準拠した、0 ～ 1 の値を返す、Oracle BI Server の Percentile 関数の結果とは異なります。

Percentile

引数に指定した数式を満たす各値のパーセント・ランクを計算します。パーセント・ランクの範囲は 0（最初のパーセンタイル）から 1（100 番目のパーセンタイル）までです（これらの値を含む）。

PERCENTILE 関数は、クエリーの結果セット内の値に基づいてパーセンタイルを計算します。

構文：

```
PERCENTILE (n_expression)
```

内容は次のとおりです。

n_expression　　数値に評価される任意の式を指定します。

PERCENTILE 関数は、「表示関数のリセット動作」（371 ページ）で説明しているルールに基づいて、クエリー内の各グループに対して自身の値をリセットします。

PeriodAgo

リレーショナル・データソースでのみ使用される時系列集計関数です。現在の時刻から指定された期間にさかのぼって集計値を計算します。たとえば、PeriodAgo 関数を使用すれば、現四半期の各月の売上と、対応する 1 つ前の四半期の売上を生成できます。

サポートされていないメトリックがリクエストされると、NULL 値が返され、ロギング・レベルが 3 以上に設定されている場合は NQQuery.log ファイルに警告エントリが書き込まれます。複数の PeriodAgo 関数が同じレベルの引数を持つ場合は、それらの PeriodAgo 関数をネストできます。

同じレベルの引数を持つ PeriodToDate 関数と PeriodAgo 関数では、1 つの PeriodToDate 関数と複数の PeriodAgo 関数をネストできます。

構文：

PeriodAgo(<time-level>, <offset>, <measure>)

PeriodToDate

リレーショナル・データソースでのみ使用される時系列集計関数です。PeriodToDate は、指定された期間の最初から現在表示されている時刻までのメジャー属性を集計します。たとえば、この関数を使用すると、1 月 1 日から今日までの売上を計算できます。

サポートされていないメトリックがリクエストされると、NULL 値が返され、ロギング・レベルが 3 以上に設定されている場合は NQQuery.log ファイルに警告エントリが書き込まれます。PeriodToDate 関数を、別の PeriodToDate 関数内でネストすることはできません。

同じレベルの引数を持つ PeriodToDate 関数と PeriodAgo 関数では、1 つの PeriodToDate 関数と複数の PeriodAgo 関数をネストできます。

構文：

PeriodToDate(<time-level>, <measure>)

Rank

引数に指定した数式を満たす各値のランクを計算します。最高値はランク 1 に割り当てられ、次に続く整数はそれぞれ後続のランク（2, 3, 4, ...）に割り当てられます。同じ値が複数ある場合、それらは同じランクに割り当てられます（たとえば、1, 1, 1, 4, 5, 5, 7... のようになります）。

RANK 関数は、クエリーの結果セット内の値に基づいてランクを計算します。

構文：

```
RANK (n_expression)
```

内容は次のとおりです。

n_expression　　数値に評価される任意の式を指定します。

RANK 関数は、「表示関数のリセット動作」（371 ページ）で説明しているルールに基づいて、クエリー内の各グループに対して自身の値をリセットします。

カレンダの絶対フィールドの計算

abs_month のようなカレンダの絶対フィールドを計算するには、ビルトインの Rank 演算子を使用します。

構文：

```
Rank(<ordering key>, <partitioning key>, <at_distinct key>)
```

内容は次のとおりです。

<ordering key>	ランクの割当てを目的とした、入力の順序付けに使用されるキー。
<partitioning key>	データのパーティション化に使用されるキー。ランク付けは、各パーティションの先頭でゼロから再度開始されます。
<at_distinct key>	このキーと異なる値を持つ行においてのみランク付けが増加するよう指示します。このキーと同じ値を持つ行は同じランクになります。

次に例を示します。

- Rank(<chronological key>, null, <year key columns>) は abs_year を返します。
- Rank(<chronological key>, null, <month key columns>) は abs_month を返します。
- Rank(<chronological key>, <year key columns>, <month key columns>) は month_in_year を返します。

StdDev

STDDEV 関数は、一連の値に対する標準偏差を返します。常に倍精度型の値を返します。

構文：

```
STDDEV([ALL | DISTINCT] n_expression)
```

内容は次のとおりです。

n_expression　　数値に評価される任意の式を指定します。

- ALL を指定すると、標準偏差がセット内のすべてのデータに対して計算されます。
- DISTINCT を指定すると、計算ですべての重複が無視されます。
- 何も指定しない場合は（デフォルト）、すべてのデータが計算されます。

STDDEV に関連して、次の 2 つの関数があります。

STDDEV_POP([ALL | DISTINCT] n_expression)

STDDEV_SAMP([ALL | DISTINCT] n_expression)

STDDEV と STDDEV_SAMP はシノニムです。

STDDEV 関数は、「表示関数のリセット動作」（371 ページ）で説明しているルールに基づいて、クエリー内の各グループに対して自身の値をリセットします。

Sum

引数に指定した数式を満たす、すべての値の合計を計算します。

構文：

```
SUM (n_expression)
```

内容は次のとおりです。

n_expression　　数値に評価される任意の式を指定します。

SUM 関数は、「表示関数のリセット動作」（371 ページ）で説明しているルールに基づいて、クエリー内の各グループに対して自身の値をリセットします。

SumDistinct

引数に指定した数式を満たす、すべての一意な値の合計を計算します。

構文：

```
SUM(DISTINCT n_expression)
```

内容は次のとおりです。

n_expression	数値に評価される任意の式を指定します。

TopN

引数に指定した式の上位 n 件の値を 1 ～ n でランク付けします。1 が最上位の数値になります。

TOPN 関数は、結果セット内の返された値に対して計算を実行します。

構文：

```
TOPN (n_expression, n)
```

内容は次のとおりです。

n_expression	数値に評価される任意の式を指定します。
n	任意の正の整数を指定します。結果セットで表示するランク付けの対象数を示します。1 が最も高いランクを表します。

1 つのクエリーに含めることができる TOPN 式は 1 つのみです。

TOPN 関数は、「表示関数のリセット動作」（371 ページ）で説明しているルールに基づいて、クエリー内の各グループに対して自身の値をリセットします。

集計実行関数

集計実行関数は、入力に一連のレコードを取る点では関数による集計と似ていますが、レコード・セット全体に対して単一の集計を出力するのではなく、それまでに検出したレコードに基づいて集計を出力します。

この項では、Oracle BI Server でサポートされる集計実行関数について説明します。

Mavg

結果セット内のデータの最後の n 行について移動平均（中央値）を計算します（現在の行を含む）。

構文：

```
MAVG (n_expression, n)
```

内容は次のとおりです。

n_expression	数値に評価される任意の式を指定します。
n	任意の正の整数を指定します。データの最後の n 行の平均を表します。

MAVG 関数は、「表示関数のリセット動作」（371 ページ）で説明しているルールに基づいて、クエリー内の各グループに対して自身の値をリセットします。

最初の行の平均は、最初の行の数式と一致します。2 番目の行の平均は、データの最初の 2 行の平均を取ることで計算されます。3 番目の行の平均は、データの最初の 3 行の平均を取ることで計算され、n 番目の行に到達するまで同様に続きます。n 番目の行では、平均はデータの最後の n 行に基づいて計算されます。

MSUM

データの最後の n 行の移動合計を計算します（現在の行を含む）。

最初の行の合計は、最初の行の数式と一致します。2 番目の行の合計は、データの最初の 2 行の合計を取ることで計算されます。3 番目の行の合計は、データの最初の 3 行の合計を取ることで計算されます。以降の行についても同様に続きます。n 番目の行に到達すると、合計はデータの最後の n 行に基づいて計算されます。

この関数は、「表示関数のリセット動作」（371 ページ）で説明しているルールに基づいて、クエリー内の各グループに対して自身の値をリセットします。

構文：

```
MSUM (n_expression, n)
```

内容は次のとおりです。

n_expression	数値に評価される任意の式を指定します。
n	任意の正の整数を指定します。データの最後の n 行の合計を表します。

例：

次の例は、MSUM 関数を使用するクエリーとその結果を示します。

```
select month, revenue, MSUM(revenue, 3) as 3_MO_SUM from sales_subject_area

MONTH       REVENUE     3_MO_SUM
JAN         100.00      100.00
FEB         200.00      300.00
MAR         100.00      400.00
APRIL       100.00      400.00
MAY         300.00      500.00
JUNE        400.00      800.00
JULY        500.00      1200.00
AUG         500.00      1400.00
SEPT        500.00      1500.00
OCT         300.00      1300.00
NOV         200.00      1000.00
DEC         100.00      600.00
```

RSUM

この関数は、それまでに検出したレコードに基づいて累積合計を計算します。最初の行の合計は、最初の行の数式と一致します。2 番目の行の合計は、データの最初の 2 行の合計を取ることで計算されます。3 番目の行の合計は、データの最初の 3 行の合計を取ることで計算されます。以降の行についても同様に続きます。

この関数は、「表示関数のリセット動作」（371 ページ）で説明しているルールに基づいて、クエリー内の各グループに対して自身の値をリセットします。

構文：

```
RSUM (n_expression)
```

内容は次のとおりです。

n_expression　　数値に評価される任意の式を指定します。

例：

次の例は、RSUM 関数を使用するクエリーとその結果を示します。

```
select month, revenue, RSUM(revenue) as RUNNING_SUM from sales_subject_area
```

MONTH	REVENUE	RUNNING_SUM
JAN	100.00	100.00
FEB	200.00	300.00
MAR	100.00	400.00
APRIL	100.00	500.00
MAY	300.00	800.00
JUNE	400.00	1200.00
JULY	500.00	1700.00
AUG	500.00	2200.00
SEPT	500.00	2700.00
OCT	300.00	3000.00
NOV	200.00	3200.00
DEC	100.00	3300.00

RCOUNT

この関数は、入力に一連のレコードを取り、それまでに検出したレコードの数を計算します。

この関数は、「表示関数のリセット動作」（371 ページ）で説明しているルールに基づいて、クエリー内の各グループに対して自身の値をリセットします。

構文：

```
RCOUNT (Expr)
```

内容は次のとおりです。

Expr 任意のデータ型の式を指定します。

例：

次の例は、RCOUNT 関数を使用するクエリーとその結果を示します。

```
select month, profit, RCOUNT(profit) from sales_subject_area where profit > 200.

MONTH       PROFIT      RCOUNT (profit)
MAY         300.00      2
JUNE        400.00      3
JULY        500.00      4
AUG         500.00      5
SEPT        500.00      6
OCT         300.00      7
```

RMAX

この関数は、入力に一連のレコードを取り、それまでに検出したレコードに基づいて最大値を示します。順序付け可能なデータ型を指定する必要があります。

この関数は、「表示関数のリセット動作」（371 ページ）で説明しているルールに基づいて、クエリー内の各グループに対して自身の値をリセットします。

構文：

```
RMAX (expression)
```

内容は次のとおりです。

expression 任意のデータ型の式を指定します。関連付けられたソート順序を持つデータ型を指定する必要があります。

例：

次の例は、RMAX 関数を使用するクエリーとその結果を示します。

```
select month, profit, RMAX(profit) from sales_subject_area

MONTH       PROFIT      RMAX (profit)
JAN         100.00      100.00
FEB         200.00      200.00
MAR         100.00      200.00
APRIL       100.00      200.00
MAY         300.00      300.00
JUNE        400.00      400.00
```

JULY	500.00	500.00
AUG	500.00	500.00
SEPT	500.00	500.00
OCT	300.00	500.00
NOV	200.00	500.00
DEC	100.00	500.00

RMIN

この関数は、入力に一連のレコードを取り、それまでに検出したレコードに基づいて最小値を示します。順序付け可能なデータ型を指定する必要があります。

この関数は、「表示関数のリセット動作」（371 ページ）で説明しているルールに基づいて、クエリー内の各グループに対して自身の値をリセットします。

構文：

```
RMIN (expression)
```

内容は次のとおりです。

expression 任意のデータ型の式を指定します。関連付けられたソート順序を持つデータ型を指定する必要があります。

例：

次の例は、RMIN 関数を使用するクエリーとその結果を示します。

```
select month, profit, RMIN(profit) from sales_subject_area
```

MONTH	PROFIT	RMIN (profit)
JAN	400.00	400.00
FEB	200.00	200.00
MAR	100.00	100.00
APRIL	100.00	100.00
MAY	300.00	100.00
JUNE	400.00	100.00
JULY	500.00	100.00
AUG	500.00	100.00
SEPT	500.00	100.00
OCT	300.00	100.00
NOV	200.00	100.00
DEC	100.00	100.00

文字列関数

文字列関数は様々な文字操作を行い、文字列に対して動作します。

ASCII

単一文字列を、0 ～ 255 の対応する ASCII コードに変換します。

構文：

```
ASCII (character_expression)
```

内容は次のとおりです。

character_expression　ASCII 文字に評価される任意の式を指定します。

文字式が複数の文字に評価される場合は、式の最初の文字に対応する ASCII コードが返されます。

Bit_Length

指定された文字列の長さをビット単位で返します。各 Unicode 文字の長さは 2 バイトです（16 ビットに相当）。

構文：

```
BIT_LENGTH (character_expression)
```

内容は次のとおりです。

character_expression　文字列に評価される任意の式を指定します。

Char

0 ～ 255 の数値を、ASCII コードに対応する文字値に変換します。

構文：

```
CHAR (n_expression)
```

内容は次のとおりです。

n_expression　0 ～ 255 の数値に評価される任意の式を指定します。

Char_Length

指定された文字列の長さを文字数で返します。先頭と末尾の空白は、文字列の長さとして計算されません。

構文：

```
CHAR_LENGTH (character_expression)
```

内容は次のとおりです。

character_expression　0 ～ 255 の数値に評価される任意の式を指定します。

Concat

この関数には 2 つの形式があります。最初の形式は、2 つの文字列を連結します。2 番目の形式は、文字列の連結文字を使用して、3 つ以上の文字列を連結します。

形式 1 の構文：

```
CONCAT (character_expression1, character_expression2)
```

内容は次のとおりです。

character_expression 文字列に評価される式を指定します。

形式 2 の構文：

```
CONCAT (string_expression1 || string_expression2 || ... string_expressionxx)
```

内容は次のとおりです。

string_expression 文字列に評価される式を、文字列の連結演算子（||）で区切って指定します。最初の文字列が 2 番目の文字列に連結されて中間の文字列が生成され、次にこの文字列が次の文字列と連結されます。

Insert

指定された文字列を、別の文字列の指定された場所に挿入します。

構文：

```
INSERT(character_expression, n, m, character_expression)
```

内容は次のとおりです。

character_expression 文字列に評価される任意の式を指定します。

n 2 番目の文字列が挿入される位置を示す、1 番目の文字列の先頭からの文字数を任意の正の整数で指定します。

m 2 番目の文字列全体によって置き換えられる、1 番目の文字列の文字数を任意の正の整数で指定します。

Left

文字列の左側から、指定された数の文字を返します。

構文：

```
LEFT(character_expression, n)
```

内容は次のとおりです。

character_expression 文字列に評価される任意の式を指定します。

n 最初の文字列の左側から返す文字数を、任意の正の整数で指定します。

Length

指定された文字列の長さを文字数で返します。末尾の空白文字を除いた長さが返されます。

構文：

```
LENGTH(character_expression)
```

内容は次のとおりです。

character_expression 文字列に評価される任意の式を指定します。

Locate

文字式での character_expression1 の数値位置を返します。character_expression1 が文字式で見つからない場合、Locate 関数は値 0 を返します。検索の開始位置を指定する場合は、かわりに LocateN 関数を使用します。

構文：

```
LOCATE(character_expression1, character_expression2)
```

内容は次のとおりです。

character_expression1 文字列に評価される任意の式を指定します。文字式で検索する式です。
character_expression2 文字列に評価される任意の式を指定します。検索される文字式です。

LocateN

文字式での character_expression1 の数値位置を返します。この関数の動作は、パターン検索が整数の引数で指定された位置から開始されることを除き、Locate 関数と同じです。character_expression1 が文字式で見つからない場合、LocateN 関数は値 0 を返します。返される数値位置は、整数の引数の値に関係なく、文字列内の最初から該当する文字までを数えることで決定されます。

構文：

```
LOCATE(character_expression1, character_expression2, n)
```

内容は次のとおりです。

character_expression1 文字列に評価される任意の式を指定します。文字式で検索する式です。
character_expression2 文字列に評価される任意の式を指定します。検索される文字式です。
n Locate 式の検索の開始位置を表す、任意のゼロ以外の正の整数を指定します。

Lower

文字列を小文字に変換します。

構文：

```
LOWER (character_expression)
```

内容は次のとおりです。

character_expression 文字列に評価される任意の式を指定します。

Octet_Length

指定された文字列のビットを、基数 8 の単位（バイト数）で返します。

構文：

```
OCTET_LENGTH (character_expression)
```

内容は次のとおりです。

character_expression 文字列に評価される任意の式を指定します。

Position

文字式での character_expression1 の数値位置を返します。character_expression1 が見つからない場合、関数は 0 を返します。

構文：

```
POSITION(character_expression1 IN character_expression2)
```

内容は次のとおりです。

character_expression1 文字列に評価される任意の式を指定します。2 番目の文字列内での検索に使用されます。

character_expression2 文字列に評価される任意の式を指定します。

Repeat

指定された式を n 回繰り返します。ここで n は正の整数です。

構文：

```
REPEAT(character_expression, n)
```

内容は次のとおりです。

character_expression 文字列に評価される任意の式を指定します。

n 任意の正の整数を指定します。

Replace

指定された文字式の指定された文字を、他の指定された文字で置換します。

構文：

```
REPLACE(character_expression, change_expression, replace_with_expression)
```

内容は次のとおりです。

character_expression	文字列に評価される任意の式を指定します。最初の文字列は元の文字列です。
change_expression	文字列に評価される任意の式を指定します。2 番目の文字列は、最初の文字列内の置換される文字を指定します。
replace_with_expression	文字列に評価される任意の式を指定します。3 番目の文字列は、最初の文字列に代入される文字を指定します。

Right

文字列の右側から、指定された数の文字を返します。

構文：

```
RIGHT(character_expression, n)
```

内容は次のとおりです。

character_expression	文字列に評価される任意の式を指定します。
n	最初の文字列の右側から返す文字数を、任意の正の整数で指定します。

Substring

元の文字列内に、固定された文字数から始まる新しい文字列を作成します。

構文：

```
SUBSTRING (character_expression FROM starting_position)
```

内容は次のとおりです。

character_expression	文字列に評価される任意の式を指定します。
starting_position	処理の開始位置を示す文字列の最初からの文字数を、任意の正の整数で指定します。

TrimBoth

指定された文字を文字列の冒頭と末尾から削除します。

構文：

```
TRIM (BOTH 'character' FROM character_expression)
```

内容は次のとおりです。

character	任意の単一文字を指定します。文字の指定（および一重引用符）を省略すると、空白文字がデフォルトで使用されます。
character_expression	文字列に評価される任意の式を指定します。

TrimLeading

指定された文字を文字列の冒頭から削除します。

構文：

```
TRIM (LEADING 'character' FROM character_expression)
```

内容は次のとおりです。

character	任意の単一文字を指定します。文字の指定（および一重引用符）を省略すると、空白文字がデフォルトで使用されます。
character_expression	文字列に評価される任意の式を指定します。

TrimTrailing

指定された文字を文字列の末尾から削除します。

構文：

```
TRIM (TRAILING 'character' FROM character_expression)
```

内容は次のとおりです。

character	任意の単一文字を指定します。文字の指定（および一重引用符）を省略すると、空白文字がデフォルトで使用されます。
character_expression	文字列に評価される任意の式を指定します。

Upper

文字列を大文字に変換します。

構文：

```
UPPER (character_expression)
```

内容は次のとおりです。

character_expression	文字列に評価される任意の式を指定します。

算術関数

算術関数は、数学的な操作を実行します。

Abs

数式の絶対値を計算します。

構文：

```
ABS (n_expression)
```

内容は次のとおりです。

n_expression　　数値に評価される任意の式を指定します。

Acos

数式の逆余弦を計算します。

構文：

```
ACOS (n_expression)
```

内容は次のとおりです。

n_expression　　数値に評価される任意の式を指定します。

Asin

数式の逆正弦を計算します。

構文：

```
ASIN (n_expression)
```

内容は次のとおりです。

n_expression　　数値に評価される任意の式を指定します。

Atan

数式の逆正接を計算します。

構文：

```
ATAN (n_expression)
```

内容は次のとおりです。

n_expression　　数値に評価される任意の式を指定します。

Atan2

y/x の逆正接を計算します。ここで y は最初の数式、x は 2 番目の数式です。

構文：

```
ATAN2 (n_expression1, n_expression2)
```

内容は次のとおりです。

n_expression（*1* および *2*）　数値に評価される任意の式を指定します。

Ceiling

整数でない数式を、次に高い整数値に丸めます。数式が整数に評価される場合はその整数を返します。

構文：

```
CEILING (n_expression)
```

内容は次のとおりです。

n_expression　　数値に評価される任意の式を指定します。

Cos

数式の余弦を計算します。

構文：

```
COS (n_expression)
```

内容は次のとおりです。

n_expression　　数値に評価される任意の式を指定します。

Cot

数式の余接を計算します。

構文：

```
COT (n_expression)
```

内容は次のとおりです。

n_expression　　数値に評価される任意の式を指定します。

Degrees

式をラジアンから度に変換します。

構文：

```
DEGREES (n_expression)
```

内容は次のとおりです。

n_expression　　数値に評価される任意の式を指定します。

Exp

値 e を指定された値でべき乗します。

構文：

```
EXP (n_expression)
```

内容は次のとおりです。

n_expression　　数値に評価される任意の式を指定します。

Floor

整数でない数式を、次に低い整数値に丸めます。数式が整数に評価される場合はその整数を返します。

構文：

```
FLOOR (n_expression)
```

内容は次のとおりです。

n_expression　　数値に評価される任意の式を指定します。

Log

式の自然対数を計算します。

構文：

```
LOG (n_expression)
```

内容は次のとおりです。

n_expression　　数値に評価される任意の式を指定します。

Log10

式の対数（基数 10）を計算します。

構文：

```
LOG10 (n_expression)
```

内容は次のとおりです。

n_expression　　数値に評価される任意の式を指定します。

Mod

最初の数式を 2 番目の数式で除算し、商の剰余を返します。

構文：

```
MOD (n_expression1, n_expression2)
```

内容は次のとおりです。

n_expression（*1* および *2*）　　数値に評価される任意の式を指定します。

Pi

パイ（円周を円の直径で割った率）の定値を返します。

構文：

```
PI()
```

Power

最初の数式を、2 番目の数式で指定された値でべき乗します。

構文：

```
POWER(n_expression1, n_expression2)
```

内容は次のとおりです。

n_expression（*1* および *2*） 数値に評価される任意の式を指定します。

Radians

式を度からラジアンに変換します。

構文：

```
RADIANS (n_expression)
```

内容は次のとおりです。

n_expression 数値に評価される任意の式を指定します。

Rand

0 ～ 1 の擬似乱数を返します。

構文：

```
RAND()
```

RandFromSeed

シード値に基づいた擬似乱数を返します。任意のシード値に対して、乱数の同一のセットが生成されます。

構文：

```
RAND (n_expression)
```

内容は次のとおりです。

n_expression 数値に評価される任意の式を指定します。

Round

数式を n 桁の精度まで丸めます。

構文：

```
ROUND (n_expression, n)
```

内容は次のとおりです。

n_expression　　数値に評価される任意の式を指定します。

n　　丸める精度の桁数を表す任意の正の整数を指定します。

Sign

引数に指定した数式が正数に評価される場合は値 1 を返し、負数に評価される場合は値 -1 を返し、ゼロに評価される場合は値 0 を返します。

構文：

```
SIGN (n_expression)
```

内容は次のとおりです。

n_expression　　数値に評価される任意の式を指定します。

Sin

数式の正弦を計算します。

構文：

```
SIN (n_expression)
```

内容は次のとおりです。

n_expression　　数値に評価される任意の式を指定します。

Sqrt

引数に指定した数式の平方根を計算します。数式は、負数でない数値に評価される必要があります。

構文：

```
SQRT (n_expression)
```

内容は次のとおりです。

n_expression　　負数でない数値に評価される任意の式を指定します。

Tan

数式の正接を計算します。

構文：

```
TAN (n_expression)
```

内容は次のとおりです。

n_expression　数値に評価される任意の式を指定します。

Truncate

小数を切捨てて、小数点から指定された桁数の数値を返します。

構文：

```
TRUNCATE (n_expression, n)
```

内容は次のとおりです。

n_expression　数値に評価される任意の式を指定します。
n　少数位の右側から返す文字数を、任意の正の整数で指定します。

カレンダ日付 / 時刻関数

カレンダ日付 / 時刻関数は、データ型 DATE および DATETIME のデータを操作します。

Current_Date

現在の日付を返します。日付は Oracle BI Server を実行しているシステムによって決定されます。

構文：

```
CURRENT_DATE
```

Current_Time

現在の時刻を返します。時刻は Oracle BI Server を実行しているシステムによって決定されます。

構文：

```
CURRENT_TIME (n)
```

内容は次のとおりです。

n　小数秒を表示する精度の桁数を表す任意の整数を指定します。この引数の指定はオプションです。何も指定しなければ、デフォルトの精度で時刻が返されます。

Current_TimeStamp

現在の日付 / タイムスタンプを返します。タイムスタンプは Oracle BI Server を実行しているシステムによって決定されます。

構文：

```
CURRENT_TIMESTAMP (n)
```

内容は次のとおりです。

n　　小数秒を表示する精度の桁数を表す任意の整数を指定します。この引数の指定はオプションです。何も指定しなければ、デフォルトの精度で時刻が返されます。

Day_Of_Quarter

指定された日付について、四半期の通算日に対応する数字（1 ～ 92）を返します。

構文：

```
DAY_OF_QUARTER (date_expression)
```

内容は次のとおりです。

date_expression　　日付に評価される任意の式を指定します。

DayName

指定された日付の曜日を返します。

構文：

```
DAYNAME (date_expression)
```

内容は次のとおりです。

date_expression　　日付に評価される任意の式を指定します。

DayOfMonth

指定された日付について、月の通算日に対応する数字を返します。

構文：

```
DAYOFMONTH (date_expression)
```

内容は次のとおりです。

date_expression　　日付に評価される任意の式を指定します。

DayOfWeek

指定された日付について、曜日（日曜日から土曜日）に対応する 1 ～ 7 の数字を返します。たとえば、数字 1 は日曜日に対応し、数字 7 は土曜日に対応します。

構文：

DAYOFWEEK (date_expression)

内容は次のとおりです。

date_expression　日付に評価される任意の式を指定します。

DayOfYear

指定された日付について、年の通算日に対応する数字（1 ～ 366）を返します。

構文：

```
DAYOFYEAR (date_expression)
```

内容は次のとおりです。

date_expression　日付に評価される任意の式を指定します。

Hour

指定された時刻について、時間に対応する数字（0 ～ 23）を返します。たとえば、0 は午前 12 時に対応し、23 は午後 11 時に対応します。

構文：

```
HOUR (time_expression)
```

内容は次のとおりです。

time_expression　時刻に評価される任意の式を指定します。

Minute

指定された時刻について、分に対応する数字（0 ～ 59）を返します。

構文：

```
MINUTE (time_expression)
```

内容は次のとおりです。

time_expression　時刻に評価される任意の式を指定します。

Month

指定された日付について、月に対応する数字（1 ～ 12）を返します。

構文：

```
MONTH (date_expression)
```

内容は次のとおりです。

date_expression　日付に評価される任意の式を指定します。

Month_Of_Quarter

指定された日付について、四半期の通算月に対応する数字（1 ～ 3）を返します。

構文：

```
MONTH_OF_QUARTER (date_expression)
```

内容は次のとおりです。

date_expression　日付に評価される任意の式を指定します。

MonthName

指定された日付の月の名前を返します。

構文：

```
MONTHNAME (date_expression)
```

内容は次のとおりです。

date_expression　日付に評価される任意の式を指定します。

Now

現在のタイムスタンプを返します。NOW 関数は、CURRENT_TIMESTAMP 関数と同じです。

構文：

```
NOW ()
```

Quarter_Of_Year

指定された日付について、年の四半期に対応する数字（1 ～ 4）を返します。

構文：

```
QUARTER_OF_YEAR (date_expression)
```

内容は次のとおりです。

date_expression　日付に評価される任意の式を指定します。

Second

指定された時刻について、秒に対応する数字（0 ～ 59）を返します。

構文：

```
SECOND (time_expression)
```

内容は次のとおりです。

time_expression　時刻に評価される任意の式を指定します。

TimestampAdd

TimestampAdd 関数は、指定されたタイムスタンプに、指定された間隔の数を加算します。単一のタイムスタンプを返します。

構文：

```
TimestampAdd (interval, integer-expression, timestamp-expression)
```

内容は次のとおりです。

interval　タイムスタンプ間隔を指定します。有効な値は次のとおりです。

- SQL_TSI_SECOND
- SQL_TSI_MINUTE
- SQL_TSI_HOUR
- SQL_TSI_DAY
- SQL_TSI_WEEK
- SQL_TSI_MONTH
- SQL_TSI_QUARTER
- SQL_TSI_YEAR

integer_expression　整数に評価される任意の式を指定します。

timestamp_expression　計算のベースに使用されるタイムスタンプを指定します。

NULL の整数式または NULL のタイムスタンプ式がこの関数に渡されると、NULL 値が返されます。

この関数の最も簡単なシナリオでは、指定された整数値（integer-expression）が、間隔（interval）に基づいて適切なタイムスタンプ・コンポーネントに加算されます。1 週は 7 日として加算され、1 四半期は 3 か月として加算されます。負の整数値を追加すると減算されます（時間をさかのぼります）。

特定のコンポーネントでオーバーフローが発生する場合（60 秒、24 時間、12 か月を超える場合など）は、次のコンポーネントに適切に値を加算する必要があります。たとえば、タイムスタンプの日のコンポーネントを加算する場合、この関数ではオーバーフローを考慮して、特定月の日数（2 月が 29 日ある、うるう年など）が計算時に確認されます。

タイムスタンプの月のコンポーネントを加算する場合は、結果として生じるタイムスタンプについて、日のコンポーネントの日数が適切であるかどうか検証されます。たとえば、2005-05-31 に 1 か月を加算しても 2006-06-31 にはなりません。これは、6 月には 31 日がないためです。この例では、日のコンポーネントが月の最後の日に減算され、2006-06-30 となります。

同様の問題は、月のコンポーネントが 2 月、日のコンポーネントが 29 であるタイムスタンプ（つまり、うるう年の 2 月の最終日）において、タイムスタンプの年のコンポーネントを加算する場合にも発生します。結果として生じるタイムスタンプがうるう年に該当しない場合は、日のコンポーネントが 28 に減算されます。

この処理は、Microsoft 社の SQL Server および Oracle のネイティブ OCI インタフェースに準拠します。

次の例は、TimestampAdd 関数を使用するクエリーとその結果を示します。

次のクエリーは、2002-02-27 14:30:00 に 3 日を追加したタイムスタンプを求めています。2000 年はうるう年であるため、2000-03-01 14:30:00 が単一のタイムスタンプとして返されます。

```
Select TimestampAdd(SQL_TSI_DAY, 3,

TIMESTAMP'2000-02-27 14:30:00')

From Employee where employeeid = 2;
```

次のクエリーは、1999-07-31 0:0:0 に 7 か月を追加したタイムスタンプを求めています。2000-02-29 00:00:00 が単一のタイムスタンプとして返されます。2 月は短い月であるため、日のコンポーネントが 29 に減算されていることに注意してください。

```
Select TimestampAdd(SQL_TSI_MONTH, 7,

TIMESTAMP'1999-07-31 00:00:00')

From Employee where employeeid = 2;
```

次のクエリーは、2000-07-31 23:35:00 に 25 分を追加したタイムスタンプを求めています。2000-08-01 00:00:00 が単一のタイムスタンプとして返されます。月のコンポーネントにオーバーフローが伝播されていることに注意してください。

```
Select TimestampAdd(SQL_TSI_MINUTE, 25,

TIMESTAMP'2000-07-31 23:35:00')

From Employee where employeeid = 2;
```

警告： Microsoft SQL Server、ODBC、IBM DB2 および Oracle Database では、TIMESTAMPADD 関数がデフォルトで有効になっています。DB2 と Oracle のセマンティクスではこの関数が完全にはサポートされていないため、Oracle BI Server の計算結果と正確には一致しない場合があります。

TimeStampDiff

TimestampDiff 関数は、2 つのタイムスタンプ間の指定された間隔の合計を返します。

構文：

```
TimestampDiff (interval, timestamp-expression1, timestamp-expression2)
```

内容は次のとおりです。

interval	タイムスタンプ間隔を指定します。有効な値は次のとおりです。 SQL_TSI_SECOND SQL_TSI_MINUTE SQL_TSI_HOUR SQL_TSI_DAY SQL_TSI_WEEK SQL_TSI_MONTH SQL_TSI_QUARTER SQL_TSI_YEAR
timestamp_expression1	関数で使用するタイムスタンプを指定します。
timestamp_expression2	関数で使用する最初のタイムスタンプを指定します。

NULL のタイムスタンプ式のパラメータがこの関数に渡されると、NULL 値が返されます。

この関数では、最初に、指定された間隔（interval）パラメータに対応するタイムスタンプ・コンポーネントが判別されます。たとえば、SQL_TSI_DAY は日のコンポーネントに対応し、SQL_TSI_MONTH は月のコンポーネントに対応します。

次に、両方のタイムスタンプで次に高い順位のコンポーネントが確認され、各タイムスタンプについて間隔の合計数が計算されます。たとえば、指定された間隔が月のコンポーネントに対応する場合は、月のコンポーネントに年のコンポーネントを 12 回加算することで各タイムスタンプの月の合計数を計算します。

最後に、最初のタイムスタンプの間隔の合計数を、2 番目のタイムスタンプの間隔の合計数から減算します。

間隔の小数部が間隔の境界を越える場合は、次の整数に丸められます。たとえば 1999-12-31 と 2000-01-01 の間の年の差は、年の小数部が翌年にまたがるため（1999 年から 2000 年）、1 年となります。対照的に、1999-01-01 と 1999-12-31 の間の年の差は、年の小数部が同じ年内に存在するため（1999 年）、0 年となります。

Microsoft SQL Server でも同様の丸めが発生します。IBM DB2 では常に切り捨てられます。Oracle では、タイムスタンプの差を求める一般化された関数が実装されていません。

週の差を計算する場合は、差を日数で計算してから、丸める前に 7 で除算します。また、Oracle BI 管理者が NQSConfig.INI ファイルで FIRST_DAY_OF_THE_WEEK パラメータを使用して構成した新しい週の開始方法も考慮されます（デフォルトは日曜日）。

たとえば、週の開始を日曜日とすると、2007-07-06（木曜日）と 2000-07-10（次の月曜日）の週の差は 1 週間となります。しかし週の開始を火曜日とすると、間隔の小数部が同じ週内に収まるため、週の差は 0 週間となります。

四半期の差を計算する場合は、差を月数で計算してから、丸める前に 3 で除算します。

IBM DB2 ではタイムスタンプの差を求める一般化された関数（TIMESTAMPDIFF）が提供されていますが、常に 1 年を 365 日、52 週、および 1 か月を 30 日とみなして計算を簡略化しています。

TimestampmDiff 関数と結果の例

次のクエリーは、タイムスタンプ 1998-07-31 23:35:00 と 2000-04-01 14:24:00 の日の差を求めています。値 610 が返されます。2000 年がうるう年であるため、日が 1 日追加されていることに注意してください。

```
Select TimestampDIFF(SQL_TSI_DAY, TIMESTAMP'1998-07-31 23:35:00', TIMESTAMP'2000-04-01 14:24:00') From Employee where employeeid = 2;
```

警告： Microsoft SQL Server、ODBC、IBM DB2 および Oracle Database では、TIMESTAMPDIFF 関数がデフォルトで有効になっています。DB2 と Oracle のセマンティクスではこの関数が完全にはサポートされていないため、Oracle BI Server の計算結果と正確には一致しない場合があります。

Week_Of_Quarter

指定された日付について、四半期の通算週に対応する数字（1 ～ 13）を返します。

構文：

```
WEEK_OF_QUARTER (date_expression)
```

内容は次のとおりです。

date_expression　日付に評価される任意の式を指定します。

Week_Of_Year

指定された日付について、年の通算週に対応する数字（1 ～ 53）を返します。

構文：

```
WEEK_OF_YEAR (date_expression)
```

内容は次のとおりです。

date_expression　日付に評価される任意の式を指定します。

Year

指定された日付の年を返します。

構文：

```
YEAR (date_expression)
```

内容は次のとおりです。

date_expression　日付に評価される任意の式を指定します。

変換関数

変換関数は、ある値の形式を別の形式に変換します。

Cast

式または Null リテラルのデータ型を、別のデータ型に変更します。たとえば、customer_name（Char または Varchar データ型）や birthdate（日時リテラル）をキャストできます。次に、値の変更が可能なサポートされたデータ型を示します。

```
CHARACTER、VARCHAR、INTEGER、FLOAT、SMALLINT、DOUBLE PRECISION、DATE、TIME、
TIMESTAMP、BIT、BIT VARYING
```

注意：元のデータ型によっては、一部のデータ型が変換後の型としてサポートされません。たとえば、元のデータ型が BIT 文字列である場合、変換後のデータ型は文字列または別の BIT 文字列である必要があります。

次に、CHAR および VARCHAR データ型に固有の特性を示します。

- **CHAR データ型へのキャスト**：サイズ・パラメータを指定する必要があります。サイズ・パラメータを指定しない場合は、デフォルトの 30 が追加されます。次に、構文のオプションについて示します。
  - 次の構文を使用することをお薦めします。

    ```
    CAST (expression|NULL AS CHAR(n) )
    ```

    例：`CAST (companyname AS CHAR(35) )`
  - 次の構文を使用できます。

    ```
    CAST (expression|NULL AS datatype )
    ```

    例：`CAST (companyname AS CHAR )`

    注意：この構文を使用すると、データが CAST (expression|NULL AS CHAR(30)) として明示的に変換され、格納されます。
- **VARCHAR データ型へのキャスト**：Administration Tool ではサイズ・パラメータを指定する必要があります。サイズ・パラメータを省略すると変更を保存できません。

Choose

任意の数のパラメータを取り、ユーザーが表示権限を持つ、リスト内の最初の項目を返します。ただし、この動作を有効にするには、Oracle BI 管理者が Administration Tool で列の権限をモデル化する必要があります。別の方法については、「IndexCol」（409 ページ）を参照してください。

構文：

```
CHOOSE (expression1, expression2, ..., expressionN)
```

たとえば、組織全体に対するセキュリティベースの収益数を返す単一のクエリーを作成できます。関数は次のようになります。

```
choose(L1-Revenue, L2-Revenue, L3-Revenue, L4-Revenue)
```

この関数を発行するユーザーが列 L1-Revenue に対するアクセス権を持っている場合は、その列の値が返されます。ユーザーに列 L1-Revenue を表示する権限がなく、L2-Revenue を表示する権限がある場合は、L2-Revenue が返されます。

IfNull

ある式が Null 値に評価されるかどうかをテストし、評価される場合は、指定された値をその式に割り当てます。

構文：

```
IFNULL (expression, value)
```

IndexCol

IndexCol を使用すると、外部情報を使用して、ログイン・ユーザーが表示する適切な列を返すことができます。Oracle BI Server では、この関数は次のように処理されます。

- **ODBC プロシージャ**: NQSGetLevelDrillability および NQSGenerateDrillDownQuery は、IndexCol から変換された式に基づいてコンテキスト固有のドリルダウン情報を返します。これは、論理 SQL クエリーで指定される IndexCol 式と、導出された論理列で指定される IndexCol 式の両方に適用されます。
- **クエリー・ログとキャッシュ**: IndexCol 関数を使用する論理 SQL は、クエリー・ログの SQL 文字列内に表示されます。ただし、Oracle BI Server では IndexCol が論理リクエスト・ジェネレータ内の式のリストにある式の 1 つに変換されるため、論理リクエストに IndexCol 関数が表示されません。

  注意：クエリー・キャッシュでは、結果として変換された式がキャッシュ・ヒット検出に使用されます。
- 使用状況トラッキング：使用状況トラッキングにより、IndexCol 関数を使用する論理 SQL クエリー文字列が挿入されます。
- セキュリティ：ユーザーに IndexCol から変換された式の列に対するアクセス権があれば、クエリーは実行されます。

  IndexCol の最初の引数がセッション変数であり、初期化ブロックが失敗してもデフォルトの式が返されることが予測される場合、Oracle BI 管理者はセッション変数に対してデフォルト値を設定する必要があります。そうしないと、セッション変数に値が定義されないため、クエリーは失敗します。

構文：

```
IndexCol( integer literal, expression_list )
```

内容は次のとおりです。

```
expression_list equals expr1 [, expression_list ]
```

IndexCol 関数は、最初の引数として整数のリテラル値（その後に可変長の式リストが続く）を取り、式リストの中の 1 つの式に変換します。リテラル値は、変換先の式リスト内の式のゼロベースの索引です。次の式を考えます。

```
IndexCol( integer literal, expr1, expr2, … )
```

リテラル値が 0 の場合、この式は expr1 と同じになります。リテラル値が 1 の場合、値は expr2 と同じになり、以降のリテラル値についても同様に処理されます。

注意： IndexCol を使用する場合は、主に最初の引数にセッション変数を含めます。定数リテラルを指定すると、IndexCol では常に同じ式が選択されます。

階層レベルの例

ABC 社には、国、都道府県、市区町村の階層を持つ地理ディメンションがあります。CEO は国レベルから市区町村レベルまでアクセスでき、営業マネージャは都道府県および市区町村レベルにアクセスでき、販売担当者は市区町村レベルにのみアクセスできます。410 ページの表 41 は、ABC 社のバックエンド・データベースを示しています。

表 41. 階層レベルを持つ IndexCol の例

USER_NAME	TITLE	GEO LEVEL	CURRENCY	CURRENCY_COL
Bob	CEO	0	US ドル	0
Harriet	営業マネージャ	1	日本円	1
Jackson	営業マネージャ	1	日本円	1
Mike	販売担当者	2	日本円	1
Jim	販売担当者	2	US ドル	0

次に、各ユーザーがアクセス権を持つ最上位レベルの情報を表示する、単一のクエリーを作成する 1 つの手順について説明します。

- Oracle BI 管理者は、新しいセッション変数 GEOGRAPHY_LEVEL を作成します。この変数の値は、初期化ブロック SELECT GEO_LEVEL from T where USER_NAME = ':USER' によって移入されます。

  これは、Oracle BI Server インスタンスに同じユーザー名があることを前提としています。

- SELECT IndexCol(VALUEOF(NQ_SESSION.GEOGRAPHY_LEVEL), Country, State, City), Revenue from Sales を使用すると、次のことが起こります。
  - Bob がログインすると、GEOGRAPHY_LEVEL セッション変数が 0 であるため、IndexCol は Country 列に変換されます。Bob は SELECT Country, Revenue from Sales を実行した場合と同じ結果を取得し、国から都道府県にドリルダウンできます。
  - Jackson がログインすると、Jackson の GEOGRAPHY_LEVEL セッション変数が 1 であるため、IndexCol は State 列に変換されます。Jackson は SELECT State, Revenue from Sales を実行した場合と同じ結果を取得し、都道府県から市区町村にドリルダウンできます。
  - Mike がログインすると、Mike の GEOGRAPHY_LEVEL セッション変数が 2 であるため、IndexCol は City 列に変換されます。Mike は SELECT City, Revenue from Sales を実行した場合と同じ結果を取得します。市区町村でのドリルダウンはできません。

VALUEOF( )

VALUEOF 関数を式ビルダーまたはフィルタで使用すると、サーバーの Administration Tool を使用して定義されたリポジトリ変数の値を参照できます。また、Oracle BI Answers の「Advanced」タブからリクエストの SQL を編集する場合にも、VALUEOF 関数を使用できます。

VALUEOF 関数の引数には変数を使用する必要があります。静的リポジトリ変数は名前で参照します。たとえば、prime_begin および prime_end という名前の静的リポジトリ変数の値を使用するには、次のように指定します。

```
CASE WHEN "Hour" >= VALUEOF("prime_begin")AND "Hour" < VALUEOF("prime_end") THEN
'Prime Time' WHEN ...ELSE...END
```

動的リポジトリ変数は、完全修飾名で参照する必要があります。動的リポジトリ変数を使用している場合は、初期化ブロック名とリポジトリ変数名を二重引用符（"）で囲み、ピリオドで区切ってカッコ内に含める必要があります。たとえば、Region Security という名前の初期化ブロックに含まれる REGION という名前の動的リポジトリ変数の値を使用する場合、適切な構文例は次のようになります。

```
SalesSubjectArea.Customer.Region =

VALUEOF("Region Security"."REGION")
```

セッション変数名は、前に NQ_SESSION を付けてピリオドで区切り、カッコで囲む必要があります。変数名に空白が含まれる場合は、名前を二重引用符（"）で囲みます。たとえば、REGION という名前のセッション変数の値を使用する場合、式ビルダー（フィルタ）で使用する適切な構文例は次のようになります。

```
"SalesSubjectArea"."Customer"."Region" = VALUEOF(NQ_SESSION.REGION)
```

注意：他のリポジトリ変数と同様に、セッション変数を初期化ブロック名とともに指定しても動作しますが、NQ_SESSION を使用する必要があります。NQ_SESSION は、すべての初期化ブロック名に一致するワイルドカードのような役割を果たします。これにより Oracle BI 管理者は、レポートに影響を与えずに、ローカライズされた方法で初期化ブロックの構造を変更できます。

システム関数

システム関数は、セッションに関連する値を返します。

User

ログインしている Oracle BI リポジトリのユーザー ID を返します。

構文：

```
USER ()
```

Database

ログインしている Oracle BI Presentation Catalog の名前を返します。

構文：

```
DATABASE ()
```

リテラルの表現

リテラルとは、任意のデータ型に対応する NULL でない値です。リテラルは通常定数値です。つまり、何も変更せずに、文字通りそのまま使用される値です。リテラル値は、それが表すデータ型に準拠する必要があります。

SQL では、SQL 文でリテラルを表現するメカニズムが用意されています。この項では、SQL で各種リテラルを表現する方法について説明します。

文字リテラル

文字リテラルは、CHARACTER または VARCHAR のデータ型の値を表します。文字リテラルを表現するには、文字列を一重引用符（'）で囲みます。リテラルの長さは、一重引用符の間の文字数で決定されます。

日時リテラル

SQL 92 標準では、次の形式で 3 種類の型指定された日時リテラルが定義されています。

DATE 'yyyy-mm-dd'

TIME 'hh:mm:ss'

TIMESTAMP 'yyyy-mm-dd hh:mm:ss'

これらの形式は固定されており、NQSConfig.INI ファイルの DATE_DISPLAY_FORMAT、TIME_DISPLAY_FORMAT または DATE_TIME_DISPLAY_FORMAT パラメータに指定された形式には影響されません。型指定された日時リテラルを表現するには、キーワードの DATE、TIME または TIMESTAMP を使用して、その後に一重引用符で囲んだ日時文字列を指定します。値が 1 桁の場合でも、年以外のすべてのコンポーネントには 2 桁が必要です。

数値リテラル

数値リテラルは、数値データ型（INTEGER、DECIMAL、FLOAT など）の値を表します。数値リテラルを表現するには、数字を SQL 文の一部として入力します。

数値リテラルは一重引用符で囲まないでください。一重引用符で囲むと、リテラルを文字リテラルとして表現することになります。

整数

整数定数をリテラルとして表現するには、整数を SQL 文の一部として入力します（たとえば SELECT リスト内で入力します）。整数の前にプラス記号（+）かマイナス記号（-）を付けると、その整数がそれぞれ正の数であるか負の数であるかを示すことができます。

小数

小数リテラルを表現するには、小数を入力します。小数の前にプラス記号（+）かマイナス記号（-）を付けると、その小数がそれぞれ正の数であるか負の数であるかを示すことができます。

浮動小数点

浮動小数点数をリテラル定数として表現するには、小数リテラルの後に、文字 E（大文字と小文字のどちらでも可）と、指数が正か負かを示すためのプラス記号（+）またはマイナス記号（-）を続けて入力します。整数部と、文字 E、および指数の符号の間に空白は許可されません。

A Oracle BI Server の使用状況トラッキング・データの説明とログ・ファイル・メソッドの使用

Oracle BI Server では、使用状況トラッキング・データの収集がサポートされています。使用状況トラッキングが有効になっている場合、Oracle BI Server では各クエリーに対応する使用状況トラッキング・データを収集し、統計を使用状況トラッキング・ログ・ファイルに書き込むか、データベースのテーブルに直接挿入します。

注意： ログ・ファイルに書き込むかわりに、直接挿入する方法を使用することを強くお薦めします。

詳細は、「使用状況トラッキングの管理」（226 ページ）を参照してください。

以前のバージョンの Usage Tracking からアップグレードする場合の詳細は、『Oracle Business Intelligence Infrastructure アップグレード・ガイド』で使用状況トラッキングに関する項を参照してください。

使用状況トラッキング・データ用 Create Table スクリプト

OracleBI¥server¥Schema フォルダには、Oracle、DB2、SQL Server および Teradata 用の次の Create Table スクリプトがあります。

- SAACCT.Oracle.sql（Oracle 用）
- SAACCT.DB2.sql（DB2 用）
- SAACCT.MSSQL.sql（SQL Server 用）
- SAACCT.Teradata.sql

サンプル・スクリプトによって使用状況トラッキング・テーブルの名前が S_NQ_ACCT に設定されます。Oracle BI Applications があるサイトの場合、Oracle BI リポジトリでこの名前が使用されます。独自のリポジトリが構築されるサイトでは、使用状況トラッキング・テーブルの名前は変更できます。テーブルの名前は対応するリポジトリで使用されるテーブルの名前と一致する必要があります。

ログ・ファイルからの使用状況トラッキング・テーブルのロード

使用状況トラッキング・テーブルのロードでは Direct-Insert オプションの使用を強くお薦めします。使用状況トラッキング・テーブルをログ・ファイルからロードする必要がある場合のために、Oracle BI では、次のサンプル JavaScript ファイルが OracleBI¥server¥Scripts¥Common サブディレクトリに用意されています。

- sblAcctLoaderMSSQL（SQL Server 用）
- sblAcctLoaderOCL.js（Oracle 用）
- sblAcctLoaderADO.js（DB2 など他のデータベース・サーバー用）

これらの JavaScript ファイルはご使用の環境に合せて変更する必要があります。これらのファイルの冒頭にあるコメントを参考にしてください。

使用状況トラッキング・ログ・ファイルのデータを抽出する前に、データを格納するテーブルを作成する必要があります。詳細は、「使用状況トラッキング・データ用 Create Table スクリプト」（416 ページ）を参照してください。

注意：UNIX インストールではこれらの JavaScript ファイルは使用できません。UNIX オペレーティング・システムでは、一般的にこれらのスクリプトで使用される高度な機能がサポートされていないためです。UNIX インストールの場合は、使用状況トラッキング・ログ・ファイルをロードするスクリプトを作成するようデータベース管理者に依頼してください。

使用状況トラッキング・データの説明

417 ページの表 42 では、使用状況トラッキング・テーブルの各列について説明します。存在する場合、データ型と長さも説明されています。

表 42. 使用状況トラッキング・データ

列	説明
CACHE_IND_FLG	デフォルトは N です。 Y はクエリーに対してキャッシュ・ヒットがあったことを示し、N はキャッシュ・ヒットがなかったことを示します。
COMPILE_TIME_SEC	クエリーのコンパイルに要した時間（秒単位）です。
CUM_DB_TIME_SEC	論理クエリーのために Oracle BI Server がバックエンドの物理データベースを待機した時間の合計（秒単位）です。
CUM_NUM_DB_ROW	バックエンド・データベースによって返された行の合計数です。
END_DT	論理クエリーが完了した日の日付です。
END_HOUR_MIN	論理クエリーが完了した時刻の時間と分です。
END_TS	論理クエリーが終了したときの日付と時刻です。開始タイムスタンプと終了タイムスタンプの時間には、リソースが使用可能になるまでクエリーが待機した時間も含まれます。
ERROR_TEXT	デフォルトは NULL です。Varchar(250) です。 バックエンド・データベースからのエラー・メッセージです。この列は、SUCCESS_FLG が 0（ゼロ）以外の値に設定されている場合のみ該当します。複数のメッセージは連結され Oracle BI Server によって解析されません。
NODE_ID	将来使用するために予約されています。
NUM_CACHE_HITS	デフォルトは NULL です。Number(10,0) です。 **注意：** DB2 の場合、データ型と長さは Decimal(10,0) です。 既存のキャッシュが返された回数です。
NUM_CACHE_INSERTED	デフォルトは NULL です。Number(10,0) です。 **注意：** DB2 の場合、データ型と長さは Decimal(10,0) です。 クエリーによって生成されたキャッシュが返された回数です。
NUM_DB_QUERY	論理クエリーのリクエストに対応するためにバックエンド・データベースに発行したクエリーの回数です。クエリーが成功した場合（SuccessFlag = 0）、この数値は 1 以上になります。

表 42. 使用状況トラッキング・データ

列	説明
PRESENTATION_NAME	デフォルトは NULL です。Varchar(128) です。 Oracle BI Presentation Services における Presentation Catalog の名前です。
QUERY_SRC_CD	リクエストのソースです（ドリルやレポートなど）。
QUERY_TEXT	クエリーのために発行された SQL 文です。
REPOSITORY_NAME	クエリーがアクセスするリポジトリの名前です。
ROW_COUNT	クエリーを行ったクライアントに返された行の数です。
RUNAS_USER_NAME	デフォルトは NULL です。Varchar(128) です。 代理ユーザーのユーザー ID です。リクエストが代理ユーザーとして実行されない場合、値は NULL になります。
SAW_DASHBOARD	ダッシュボードのパスです。クエリーが Interactive Dashboard を介して発行されていない場合、値は NULL になります。
SAW_DASHBOARD_PG	デフォルトは NULL です。Varchar(150) です。 Interactive Dashboard におけるページ名です。リクエストがダッシュボードのリクエストでない場合、値は NULL になります。
SAW_SRC_PATH	Oracle BI Presentation Catalog におけるリクエストのパス名です。
START_DT	論理クエリーが発行された日の日付です。
START_HOUR_MIN	論理クエリーが発行された時刻の時間と分です。
START_TS	論理クエリーが発行されたときの日付と時刻です。
SUBJECT_AREA_NAME	アクセスされたビジネス・モデルの名前です。
SUCCESS_FLG	クエリーの完了ステータスです。0 の場合、クエリーはエラーなしで正常に完了しました。1 の場合、クエリーはタイムアウトで終了しました。2 の場合、行の制限を超過したためにクエリーは失敗しました。3 の場合、他の理由によりクエリーは失敗しました。
TOTAL_TIME_SEC	クライアントがクエリー・リクエストのレスポンスを待機した間に Oracle BI Server においてクエリーを処理した時間（秒単位）です。
USER_NAME	クエリーを発行したユーザーの名前です。

B Oracle BI Server 認証 API

Oracle BI 管理者は開発者に対して Oracle BI の認証 API 仕様に従って動的にロードできる認証モジュールの作成をリクエストします。この付録では、認証子で実装が必要な API について説明します。実装の詳細は、「初期化ブロックを使用したユーザー認証について」（301 ページ）を参照してください。

すべての API は SAChar をパラメータにしており、定義は次のとおりです。

```
typedef wchar_t SAChar;
```

他の関連する型定義は次のように定義されています。

```
typedef  unsigned int  SAUInt32;

enum

{

    SAAuthenticatorTrue,

    SAAuthenticatorFalse,

    SAAuthenticatorNotSupported

} SAReturnType;
```

動的にロード可能な認証子で使用されるすべての型のためにヘッダー・ファイルが用意されています。次のすべての API はスレッド・セーフである必要があります。

- SAUInt32 SAAuthenticatorGetVersion()

  説明：これによって認証子の現在のバージョンが返されます。バージョン番号は、1 に事前定義されています。

  - 引数

    なし。

  - 戻り値

    バージョン番号は、1 に事前定義されています。

- void SAAuthenticateFreeString(SAChar *p)

  この関数はユーティリティ関数で、p が指すメモリーを解放します。

  - 引数

入力	p	SAChar 文字列へのポインタです。

  - 戻り値

    なし。

- void SAAuthenticateFreeStringArray(SAChar **pp)

  この関数はユーティリティ関数です。pp が指すメモリーを解放します。

  - 引数

入力	pp	SAChar 配列へのポインタです。

  - 戻り値

    なし。

- SAReturnType SAAuthenticatorInit(const SAChar * pConfigParams, SAChar **ppErrorMessage)

  この関数は認証モジュールに対して基本的な初期化処理を実行します。Oracle BI Server から一度だけコールされます。

  - 引数

入力	pConfigParams	認証子に渡されるパラメータへのポインタです。文字列は Configuration パラメータを連結した文字列で、Encrypted パラメータが復号化されたものです。
出力	ppErrorMessage	エラー・メッセージが格納されている場合、エラーが発生しています。認証子を記述するプログラマはエラー・メッセージ用メモリーを割り当てる必要があります。

  - 戻り値

    - 初期化が成功すると、SAAuthenticatorTrue という値が返されます。
    - 初期化が失敗すると、SAAuthenticatorFalse という値が返され、エラー・メッセージが *ppErrorMessage に格納されます。

    注意：認証子フレームワークにより、ppErrorMessage のために割り当てられたメモリーが解放されます。

- SAReturnType SAAuthenticatorLogin(const SAChar * pUserID, const SAChar *pPassword, SAChar **ppErrorMessage)

  この関数によりユーザーが認証されます。

  - 引数

入力	pUserID	ユーザーのログイン名です。
入力	pPassword	ユーザーのパスワードです。
出力	ppErrorMessage	エラー・メッセージが格納されている場合、エラーが発生しています。認証子を記述するプログラマは、エラー・メッセージ用にメモリーを割り当てる必要があります。

  - 戻り値

    - 認証が成功すると、SAAuthenticatorTrue という値が返されます。

- 認証が失敗すると、SAAuthenticatorFalse という値が返され、エラー・メッセージが *ppErrorMessage に格納されます。

認証子フレームワークにより、ppErrorMessage のために割り当てられたメモリーが解放されます。この API は、SAAuthenticatorNotSupported を返すことはできません。

- bool SAAuthenticatorIsAValidUser(const SAChar * pUserID, SAChar **ppErrorMessage)

この関数により、指定したユーザー ID が有効であるかどうかを調べます。

- 引数

入力	pUserID	ユーザーのログイン名です。
出力	ppErrorMessage	エラー・メッセージが格納されている場合、エラーが発生しています。認証子を記述するプログラマは、エラー・メッセージ用にメモリーを割り当てる必要があります。

- 戻り値
  - ユーザー ID が有効な場合、SAAuthenticatorTrue という値が返されます。
  - ユーザー ID が無効な場合、SAAuthenticatorFalse という値が返され、エラー・メッセージが *ppErrorMessage に格納されます。

  認証子フレームワークにより、ppErrorMessage のために割り当てられたメモリーが解放されます。

  注意：この API は、SAAuthenticatorNotSupported を返すことはできません。

- SAReturnType SAAuthenticatorGetUserProps(const SAChar * pUserID, const SAChar **ppKeys, SAChar *** pppValues, SAChar **ppErrorMessage)

この関数ではプロパティ値のリストが含まれます。キーが GROUP で、ユーザーが複数のグループに属する場合、値はセミコロンで区切る必要があります。

- 引数

入力	pUserID	ユーザーのログイン名です。
入力	ppKeys	返すプロパティ値を示す文字列の配列（NULL が終端）へのポインタです。
出力	pppValues	プロパティ値が ppKeys に対応する文字列の配列（NULL が終端）へのポインタです。認証子を記述するプログラマは、この配列用にメモリーを割り当てる必要があります。
出力	ppErrorMessage	エラー・メッセージが格納されている場合、エラーが発生しています。認証子を記述するプログラマは、エラー・メッセージ用にメモリーを割り当てる必要があります。

- 戻り値
  - 関数コールが成功すると、SAAuthenticatorTrue という値が返されます。Oracle BI Server の認証子フレームワークにより、pppValues のために割り当てられたメモリーが解放されます。

- 関数コールが失敗すると、SAAuthenticatorFalse という値が返され、エラー・メッセージが *ppErrorMessage に格納されます。Oracle BI Server の認証子フレームワークにより、ppErrorMessage のために割り当てられたメモリーが解放されます。

注意： この API は、SAAuthenticatorNotSupported を返すことはできません。

- SAReturnType SAAuthenticatorGetAllGroups(SAChar *** pppGroups, SAChar **ppErrorMessage)

  この関数によりすべてのグループのリストを取得します。

  - 引数

出力	pppGroups	ユーザーが属するグループを示す文字列の配列（NULL が終端）へのポインタです。認証子を記述するプログラマは、この配列用にメモリーを割り当てる必要があります。
出力	ppErrorMessage	エラー・メッセージが格納されている場合、エラーが発生しています。認証子を記述するプログラマは、エラー・メッセージ用にメモリーを割り当てる必要があります。

  - 戻り値
    - 関数コールが成功すると、SAAuthenticatorTrue という値が返されます。Oracle BI Server の認証子フレームワークにより、pppGroups のために割り当てられたメモリーが解放されます。
    - 関数コールが失敗すると、SAAuthenticatorFalse という値が返され、エラー・メッセージが *ppErrorMessage に格納されます。Oracle BI Server の認証子フレームワークにより、ppErrorMessage のために割り当てられたメモリーが解放されます。

  注意： サポートされていない場合、API は SAAuthenticatorNotSupported を返すことができます。

- void SAAuthenticatorShutdown()

  この関数は Oracle BI Server が停止するときに、クリーンアップを実行します。Oracle BI Server から一度だけコールされます。

  - 引数

    なし。

  - 戻り値

    なし。

索引

D

M

N

O

P

Q

R

S

Y

あ

か

さ

た

な

は

ま

や

ら

わ